

A DEEP LEARNING COMPUTER VISION SYSTEM FOR IMAGE
BASED CYTOMETRY

by

Michal Lisicki

Submitted in partial fulfillment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
December 2014

© Copyright by Michal Lisicki, 2014

To my always supportive family, who never let me give up

Table of Contents

List of Tables	vi
List of Figures	vii
Abstract	xi
List of Abbreviations and Symbols Used	xii
Acknowledgements	xiv
Chapter 1 Introduction	1
1.1 Objectives and scope of the thesis	1
1.2 Complete Blood Count	3
1.3 Contribution	3
1.4 Outline	5
Chapter 2 Background analysis	6
2.1 Physiology background	6
2.1.1 Complete Blood Count	6
2.1.2 Differential WBC count	10
2.2 Standard biological image analysis pipeline	12
2.3 Related work in computer vision	13
2.3.1 Problems related to low quality images	13
2.3.2 Examples of image cytometry systems	14
2.4 Related work in machine learning	16
Chapter 3 Tools and related applications	19
3.1 Background	19
3.2 ImageJ and Fiji	19
3.3 OpenCV	21
3.4 MATLAB image processing and computer vision toolboxes	22
3.5 Pylearn2, Cuda Convnet and GPGPU programming	22

3.6	Popular cell analyzing applications	23
Chapter 4	Algorithms and methods	26
4.1	Background	26
4.2	Image pre-processing	26
4.2.1	Flat-field correction	26
4.2.2	Contrast enhancement	27
4.2.3	Denoising	29
4.2.4	Resolution enhancement	31
4.2.5	Artifact removal	34
4.2.6	Color space	36
4.3	Segmentation and candidates extraction	37
4.3.1	Hough transform	37
4.3.2	Voronoi diagram	39
4.4	Feature extraction	40
4.4.1	Size and nucleus - cytoplasm ratio	40
4.4.2	Image moments	44
4.5	Classification	46
4.5.1	Rule-based classifier	46
4.5.2	Classical analysis	48
4.5.3	Convolutional Neural Network	50
4.5.4	Pre-training and filter initialization	53
4.5.5	Dropout	56
Chapter 5	Experiments and Results	57
5.1	Image Preparation	57
5.2	Cell analysis and final results	59
5.2.1	Erythrocyte segmentation and analysis	59
5.2.2	Thrombocyte segmentation and analysis	65
5.2.3	Leukocyte classification	66
5.3	Discussion on results	76
5.3.1	Erythrocyte	76
5.3.2	Thrombocyte	78
5.3.3	Leukocyte	79
Chapter 6	General discussion and future prospects	82
Appendices	85

Appendix A	Anisotropic diffusion	85
Appendix B	Super-resolution	88
Appendix C	Image moments	90
Appendix D	Backpropagation in Convolutional Neural Network . .	98
Bibliography	101

List of Tables

2.1	Analytic goals and state of the art (Buttarelli and Plebani, 2008)	11
2.2	Types of leukocytes. Data based on Wikipedia (2013b)	12
5.1	RBC R^2 summary table.	65
5.2	Confusion matrix - predictions (top) vs actual class (left)	73
5.3	Platelet R^2 summary table (CNN)	75
5.4	Rule-based classification	79
5.5	CNN after training on validation set	79
C.1	Image moments of examples on Fig.C.6 proving invariance of features under specified transformations	94

List of Figures

2.1	CBC print-out example (<i>Online resource, 2012a</i>)	6
2.2	CBC fishbone diagram	7
2.3	Idealistic model of red blood cell distribution histogram	8
2.4	RBC differentiation based on laser illumination of cells - signal correlation plot (<i>Online resource, 2012b</i>)	9
2.5	WBC differentiation based on laser illumination of cells - signal correlation plot (<i>Online resource, 2012b</i>)	9
2.6	Normal WBC differential plot showing volume against light scatter of the cells (McClatchey, 2002)	10
2.7	Volume distribution of WBCs, RBCs and Plts (Bain, 2006).	11
2.8	Common high-level architecture of biological image analysis systems	13
3.1	WBC analysis with Fiji Weka plugin using low quality image	20
3.2	GUI developed as part of the thesis for testing variety of segmentation methods available in OpenCV and MATLAB	22
3.3	CellC GUI	25
4.1	Flat field correction	27
4.2	Original image	28
4.3	Histogram stretch	28
4.4	Histogram equalization	29
4.5	Histogram with custom look up table	29
4.6	Anisotropic diffusion example	30
4.7	Lanczos kernel for window size $a = 3$	32
4.8	Example application of Lanczos filter application	32
4.9	Super-resolution image with resolution enhancing factor too large	34
4.10	Power spectrum of the image on Fig.4.9	35

4.11	Image corrected by FFT filtering	36
4.12	L a b color space channels	36
4.13	Edge detection with different methods.	38
4.14	Example of Hough transform for line detection	39
4.15	Line-fitting to the edge of WBC in log-polar coordinates . . .	41
4.16	Log-polar image profile peaks	42
4.17	Result of WBC contour tracking in log-polar coordinates . . .	42
4.18	Region growing results for set of WBCs	44
4.19	(a) Original image. (b) Hu moment invariant map	45
4.20	Nucleus to cytoplasm ratio segmentation.	48
4.21	Comparison of softplus and rectified linear functions	52
4.22	Hand crafted filters	54
5.1	Results of super-resolution	57
5.2	Comparison of filtering of the original image	58
5.3	Visuals of the main objects of interest cropped out from images after the last step of pre-processing	59
5.4	RBC threshold	60
5.5	RBC declustering	61
5.6	Histogram of object sizes	62
5.7	RBC declustering methods	62
5.8	RBC segmentation. Violet channel of the image with overlaid borders (red) of the segmented red blood cells.	63
5.9	Squared Pearson correlation of total (a) RBC (b) MCV (c) RDW (d) MCH (e) Hct count with Coulter Counter readings for given patients	64
5.10	Platelet segmentation	66
5.11	Squared Pearson correlation of total platelet count with Coulter Counter readings for given patients	66

5.12	Representatives of examples of three classes in the dataset - anomalies, lymphocytes and myeloids	67
5.13	Random WBCs examples displaying their (a) RGB and (b) fourth 405nm violet channel	67
5.14	Ground truth class and localization markers manually positioned by expert.	68
5.15	Data Augmentation by rotation, flipping, shifting and intensity modification by PCA in RGB space.	69
5.16	SVM grid search in space of regularization parameter C and RBF kernel parameter γ	70
5.17	Convolutional Neural Network architecture used for WBC recognition	70
5.18	(a) Lymphocyte generalization curve. (b) Myeloids generalization curve. (c) Mean negative log likelihood	71
5.19	Filters trained with regular CNN (a) RGB (b) Violet.	72
5.20	Filters trained by CNN with dropout (a) RGB (b) Violet.	72
5.21	RGB filters of convolutional RBM	73
5.22	Best accuracy achieved on the network	74
5.23	Classification results comparison showing per-class recall of classical and deep learning approaches.	75
5.24	Squared Pearson correlation of WBC parameters	76
5.25	Red blood cells fading due to death or loss of hemoglobin.	77
5.26	(a) Anomaly caused by erroneous data acquisition and mask (b) for removing those and similar areas from further processing.	77
5.27	Typical appearance of platelet (a) and faded platelet commonly occurring within blood samples of patient 6 (b)	78
5.28	Selected positive examples	80
5.29	Examples of $0.5 \leq p(y x) < 0.75$	80
5.30	Most similar examples from other class	80
C.1	2D Gaussian distribution with varied moment $M_{10} = \mu_1$	91
C.2	2D Gaussian distribution with varied moment M_{20}	92

C.3	2D Gaussian distribution with varied moment M_{30}	92
C.4	2D Gaussian distribution with varied moment M_{11}	93
C.5	2D Gaussian distribution with varied moment M_{21}	93
C.6	Examples of an image under transformations of scaling, mirroring, which can be described with exactly the same moment invariants.	95

Abstract

A Complete Blood Count (CBC) is one of the most commonly deployed tests for checking the overall medical condition of a human body. It results in a summary of blood cell related measures allowing for a quick preliminary detection of common diseases. Image based cytometry is a modern way of overcoming common efficiency issues related to current gold standard method of Coulter Counter analysis.

This thesis describes a computer vision pipeline for measuring main components of CBC based on methods with varied level of human influence in feature design. The low quality input images are acquired from a prototype lensless microscope. Rule-based approaches proved sufficient in cases of red blood cells and platelets analysis and their configuration is explored in more details at the segmentation stage.

Leukocyte differentiation is a subproblem of CBC requiring more sophisticated classification approach. The comparison of results based on rule-based, classical and deep learning methods is presented. Support Vector Machines and k-Nearest Neighbours were chosen as representatives of classical approaches. The deep learning scheme has the advantage of finding quick solutions to complex problems involving highly dimensional and hierarchical structure of underlying solutions. It also alleviates the amount of work needed by human experts to design appropriate descriptors.

The results discussed here include promising enumeration and feature description of the objects of interest. Their linear dependence with gold standard clinical results proved their usability for point of care blood analysis. Classification results showed superiority of convolutional neural networks in several important aspects. Further analysis explored the learned parameters and analyzed wrongly classified cases in order to understand better what kind of cytological objects are most fragile for particular classifier and why.

List of Abbreviations and Symbols Used

CBC	Complete Blood Count
CNN	Convolutional Neural Network
GPGPU	General Purpose Graphical Processing Unit
Hct	Hematocrit
Hgb	Hemoglobin
HQ	High Quality
HSV	Hue Saturation Value
ICA	Independent Component Analysis
kNN	k Nearest Neighbours
LQ	Low Quality
Ly	Lymphocyte
MCH	Mean Cell Hemoglobin / Mean Corpuscular Hemoglobin
MCV	Mean Cell Volume / Mean Corpuscular Volume
MLP	Multi-Layer Perceptron
My	Myeloid
PCA	Principal Component Analysis
Plt	Platelet
POC	Point-of-Care
RBC	Red Blood Cell
RDW	Red Cell Distribution Width

ReLU	Rectified Linear Unit
RGB	Red Green Blue
ROI	Region Of Interest
SVM	Support Vector Machine
WBC	White Blood Cell

Acknowledgements

I would like to thank my supervisor, dr. Thomas Trappenberg, for introducing me to the fascinating world of biologically plausible and statistical machine learning. Being thrown into the deep water since the beginning helped me to understand the holistic view on this field, see where the real scientific problems are and how to approach them. Big thanks also to dr. Alan Fine for large exposure on physiology research and leading me through the applied research project. It greatly influenced my current interest in fascinating multidisciplinary field of medical data analysis. The great long discussions with Paul Hollensen helped me to analyze the problems in greater details, get better view on the available literature and move forward whenever I got stuck. I am extremely grateful to have such a great colleagues both in HAL lab and Alentic Microscience. None of this work could have been done without your constant support and discussions. In the end I would like to show my great appreciation for my girlfriend, Maja Wizer, and my close and further family, especially my mother Elzbieta, my father Tomasz, sister Agnieszka and aunt Halina, who encouraged me to stay in health and showed how to stay happy even in harsh conditions.

Chapter 1

Introduction

1.1 Objectives and scope of the thesis

The main goal of this study is to develop software for lensless microscopic device capable of performing the Complete Blood Count (CBC) test in the Point Of Care (POC device). The device is currently under the development process by Alentic Microscience Inc., a startup company based in Halifax, Canada. The full hardware design cannot be disclosed at this stage due to privacy issues. Their effort is put particularly on solving the time and throughput problems of blood tests by letting physician to make a test within only few minutes during patient's first visit. By using the hand-held device the health worker could prick the patient's finger and collect as little as $10\mu l$ of blood, which could be then mixed with diluent and transported onto the image sensor. The images would be collected directly without lens by illuminating sample with lights of chosen wavelengths.

The work presented in this thesis is a result of my cooperation with this company and focuses on presenting the system pipeline purely from software perspective. Research and development nature of the company triggered many design changes done in parallel with system development. The effort was made to adapt for those changes in parts particularly improving the results. In the other parts - the quality of the data was kept at the level satisfactory for explanation and testing. Other issues are mentioned below.

CBC test comprises of measures related to three main blood components - red blood cells, white blood cells and platelets. From the computer vision perspective, the blood sample image needs to be first pre-processed to remove noise and apply algorithms for obtaining higher resolution by combining images with different angles of illumination. Further, the image has to be segmented to get the probable locations and coverage of specific types of cells. The features of found objects can be then extracted and classification algorithms can then be used to differentiate between them.

The main challenge is currently the limited resolution of the images which prevents using most of the methods developed in the literature (Bikhet et al., 2000, Hiremath et al., 2010, Young, 1972) directly. The other challenges include motion blur, magnified by a long time of acquisition, stain attaching to other objects resembling cells, bubbles and light reflections on the sensor surface. During the development of the system reported here the quality of images was changing. For some of the tasks initial quality provided the information needed to describe the concept and perform successful analysis. For other ones, only after improving staining practice and acquisition procedure the objects could be differentiated sufficiently to perform good analysis. As the work on algorithms performing cell analysis was developed in parallel with image enhancement the decision was made to show the results on both low quality images - for simpler problems - and higher quality - for more complex ones, like finer leukocyte differentiation. Different perspectives on the problem can also provide more insight for further research on optimization.

The first part of experiments involves localizing, separating, counting and extracting detailed features of red blood cells (erythrocytes) and platelets (thrombocytes). This part is mostly focused on segmentation techniques in low quality data and later the methods are slightly modified to obtain similar results using HQ data. The further analysis here was done purely with rule-based approaches. Also the preliminary work on white blood cell (leukocyte) detection is presented.

The second part provides much more extensive examination of WBC differentiation problem using HQ information. It is often considered as a most informative part of the test and provides the best field for testing state of the art classification methods. For that the three popular approaches to medical image analysis are presented - rule-based, classical and deep learning.

The best of obtained results are further correlated with gold standard measures, to test their real usability. For each patient the gathered numbers specify enumeration of the cells in questions - RBCs, Plts and two main types of WBCs - lymphocytes and myeloids. The same data is gathered with current standard procedure from blood analysis laboratory, using different blood samples of the same patients. The hope is to prove the linear dependence between those two measures by obtaining high r^2 coefficient of determination (i.e. squared Pearson correlation in this case).

1.2 Complete Blood Count

One of the most popularly deployed tests for checking the overall medical condition of a human body is the Complete Blood Count. This is a common medical test panel measuring various components of the blood, resulting in summary which allows easy detection of many general organism malfunctions. The most popular factors include enumeration of the red blood cells (RBCs), platelets (Plts), most common types of white blood cells (WBCs), and parameters usually calculated indirectly, like hemoglobin concentration or hematocrit (volume percentage of RBCs in blood). Different conditions affecting those parameters may include for example infection, inflammation, coagulopathies, neoplasm or toxic substance exposure (George-Gay and Parker, 2003). For example color intensity of the RBCs indicate the oxygenation level of hemoglobin. Besides the parameters can indicate blood disorders directly, like anemia related to RBCs and Hgb measures, leukocytosis, leukopenia and leukemia associated with leukocyte measures and thrombocytosis and thrombocytopenia associated with platelet measures. Among those often the most informative are leukocyte factors as they constitute the blood defensive system and are very prone to direct changes in response to unwanted substances appearing in the organism.

The current procedure for taking the test involves many steps and can take many hours for return of the data. The throughput is also quite low. It happens often that patient needs to wait for appointment to have his or her blood drawn. Later the blood must be often transported to laboratory where automatic analyzer, like Coulter counter, is used to perform CBC. This is costly, takes a lot of time and often doesn't provide enough details about cell morphology. Before the results are available the patient is scheduled for the next visit - usually within couple of days.

1.3 Contribution

The contribution of this thesis is two-fold. First the pipeline for image based cytometry was designed and demonstrated in the context of a complete blood count. The second contribution is a proof of concept application and comparison to show that modern machine learning algorithms can learn features from raw data to obtain similar or better classification results than when features are selected by trained engineers.

Although the recent literature shows superiority of this approach for commonly used evaluation datasets, the thesis presents its performance in real world application. The results showed that some of the measures are already clinically valid. Other ones will need additional work, but are encouraging. I believe they will contribute to better understanding of the processes driving the choice of the features, which are not always directly interpretable by humans, and present advantages of algorithms making use of them.

Segmentation of red blood cells was particularly successful. The violet color channel allowed for easy application of automatic threshold. The clusters of touching cells were separated using Hough transform and Voronoi diagram. The size histogram of separated cells didn't appear to augment well the one of freely moving cells, because of occlusion and pressure from the side of surrounding cells which affected the final shape. The decision was then made to use all of the cells for enumeration, but only freely floating for extraction of other properties. After segmentation all of the parameters were successfully extracted. Most of them appeared to be associated more with the blood treatment and acquisition procedure than with computer vision processing.

Platelet enumeration was highly significant apart from one outlier. One of the patient's blood had unusual conditions which prevented accurate count using current method and largely affected the linear dependency approximation with ground truth data. This suggests the need for larger data collection to better understand the nature of those conditions.

White blood cell analysis was based on comparison of three popular classification algorithms. The parameters of rule-based approach were highly adjusted by expert and gave very good correlation results with ground truth. kNN and SVM methods were used for quantification of classical approach results. Even though SVM's hyperparameters were optimized using grid search method, the accuracy obtained for lymphocyte class was particularly low. This might be caused by morphological similarity to both neighbouring classes - myeloids containing also small cells with large nuclei and anomalies containing sometimes large platelets or contaminants resembling lymphocytes. Convolutional neural networks proved superior in terms of time needed for hyperparameters adjustment, accuracy and correlation with ground truth. Additional analysis of wrongly classified samples revealed small inconsistencies in initial

labeling and showed a few cases indistinguishable even by human expert.

1.4 Outline

The thesis is divided into chapters defining extensive background knowledge on the topic, hierarchy of computer vision methods used in the process, defined in order of standard medical image analysis system, and experiments showing how those methods were deployed to achieve the specified goals of RBC, WBC and Plt analysis.

The first chapter focuses on overview defining the main concept needed to understand the higher goals meant to be achieved by the systems. Those topics are mainly drawn from physiology field. This is immediately followed by section providing background and related work in the field of computer vision tackling problems of image based cytometry from perspectives of microscopy, biomedical engineering, low level computer vision and machine learning.

The next two chapters define all the programmatic tools and main methods used in the application project. The chapter providing the algorithm overview is presented in order specific for most of known computer vision systems. Beginning with algorithms used for pre-processing and image enhancement, through segmentation, feature extraction and finishing on different machine learning, particularly connectionists', approaches.

These methods are then of course applied in order specific for given application - RBC, Plt or WBC analysis - and the process of deployment is described in experiments and results chapter. Even though method overview chapter provides examples of utilization of the algorithms on the same data, this chapter combines them together and provides the final results of the analysis, producing full blood count.

I conclude with summary and discussion on those results and provide potential future prospects for further research.

Chapter 2

Background analysis

2.1 Physiology background

2.1.1 Complete Blood Count

Complete Blood Count (CBC), also called Full Blood Exam (FBE) or Blood Panel, is a test panel measuring the main properties of the cells in a patient's blood. Such a test panel (or profile) consists of a predetermined group of medical tests which aids in diagnosis and treatment of different diseases. CBC is one of the most popular procedures performed frequently and is capable of detecting such hematological diseases like e.g. leukemias, anemias or inherited blood disorders.

TESTS	RESULT	FLAG	UNITS	REFERENCE INTERVAL	LAB
CBC With Differential/Platelet					
WBC	5.7		x10E3/uL	4.0-10.5	01
RBC	5.27		x10E6/uL	4.10-5.60	01
Hemoglobin	15.4		g/dL	12.5-17.0	01
Hematocrit	44.1		%	36.0-50.0	01
MCV	84		fL	80-98	01
MCH	29.2		pg	27.0-34.0	01
MCHC	34.9		g/dL	32.0-36.0	01
RDW	13.7		%	11.7-15.0	01
Platelets	268		x10E3/uL	140-415	01
Neutrophils	47		%	40-74	01
Lymphs	46		%	14-46	01
Monocytes	6		%	4-13	01
Eos	1		%	0-7	01
Basos	0		%	0-3	01
Neutrophils (Absolute)	2.6		x10E3/uL	1.8-7.8	01
Lymphs (Absolute)	2.6		x10E3/uL	0.7-4.5	01
Monocytes (Absolute)	0.4		x10E3/uL	0.1-1.0	01
Eos (Absolute)	0.1		x10E3/uL	0.0-0.4	01
Baso (Absolute)	0.0		x10E3/uL	0.0-0.2	01
Immature Granulocytes	0		%	0-1	01
Immature Grans (Abs)	0.0		x10E3/uL	0.0-0.1	01

Figure 2.1: CBC print-out example (*Online resource, 2012a*). The main parameters meaning is described below. The second and fourth columns provide measures for particular patient and their units. Further is the reference specifying the standard ranges for humans.

The main types of cells in human bloodstream, which are measured by the process, are leukocytes (white blood cells, WBC), erythrocytes (red blood cells, RBC) and

thrombocytes (platelets). The fishbone diagram for the main types of measurements is shown on Fig. 2.2.

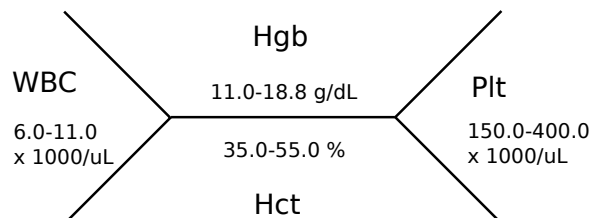


Figure 2.2: CBC fishbone diagram

The smaller hematology instruments measure only the number of RBCs, WBCs and platelets. Example of standard CBC print-out can be seen on Fig.2.1. The most commonly used eight measures and calculated parameters (Turgeon, 2004) are:

- WBC - number of white blood cells
- RBC - number of red blood cells
- Hbg (Hemoglobin)
- Hct (Hematocrit) - (also Packed Cell Volume (PCV) or Erythrocyte Volume Fraction (EVF)) volume percentage of red blood cells in blood.
- Mean Cell Volume (MCV) - measure of average red blood cell size ($MCV = \frac{Hct}{RBC}$)
- Mean Cell Hemoglobin (MCH) - average hemoglobine per red blood cell ($MCH = \frac{Hgb}{RBC}$)
- Mean Cell Hemoglobin Concentration (MCHC) - measure of the concentration of hemoglobin in a given volume of packed red blood cells ($MCHC = \frac{MCH}{MCV}$)
- Plt - number of platelets
- Additional parameters - e.g. Erythrocyte morphology (RDW), leukocyte histogram differential, erythrocyte sedimentation rate (ESR), main types of white blood cells.

The methods for obtaining those measurements are well described by Bain (2006) and Turgeon (2004). They are divided into manual blood counts and automated blood counts. The former involve using blood films with peripheral blood smears and watching specimens with diluted blood under the microscope in counting chambers. Manual counting is useful mostly when automated analyzers cannot reliably count abnormal cells or differentiate their types correctly. Automated blood counts make use of automated hematology analyzers. These are the medical laboratory instruments used for measuring different types of blood cells together with their characteristics quickly and with minimal human assistance. The samples can be processed one-by-one, in batches or continuously (by time lapse analysis). The most popular analyzers are based on flow cytometry, where machine puts a very small amount of the specimen through tube narrow enough to not allow cells to overlap. While specimen flows through it, the attached sensors perform different measurements for variety of elements in the blood. Two main sensors being put in almost every analyzer are electrical impedance (Coulter counter) sensor and laser-based optical sensor. First one is generally used for counting and sizing particles suspended in electrolytes. It allows cells to pass through electrical current to measure their impedance. The assumptions are made that only one cell would go through it at the time and that bigger cells should cause larger impedance. After that the reagent is usually added to lyse RBCs and leave only WBCs, which are counted next together with platelets. WBC count is separated from platelet count by measuring the height of the impedance peaks. Example measure of RBC count is shown on Fig. 2.3. The number of cells of particular size is stated in femtolitres.

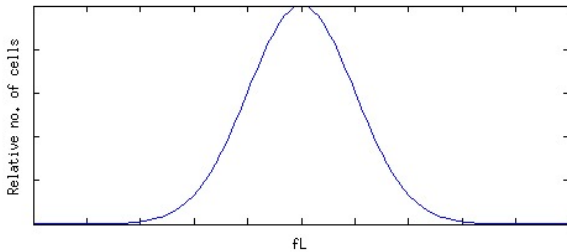


Figure 2.3: Idealistic model of red blood cell distribution histogram

The optical techniques are mainly used to perform differential count of the population of WBC. Diluted specimen is passed once at a time onto a laser beam. The

measurements include reflectance, transmission and scattering of light. Results of such differentiation are shown on Fig. 2.4, 2.5. Fig.2.6 shows more historical and very standard WBC differentiation plot, where lymphocytes, monocytes, neutrophils, eosinophils and debris can be distinguished easily.

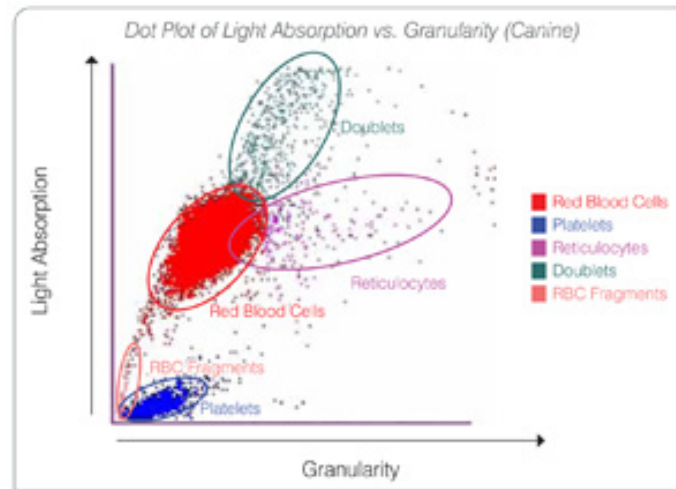


Figure 2.4: RBC differentiation based on laser illumination of cells - signal correlation plot (*Online resource, 2012b*)

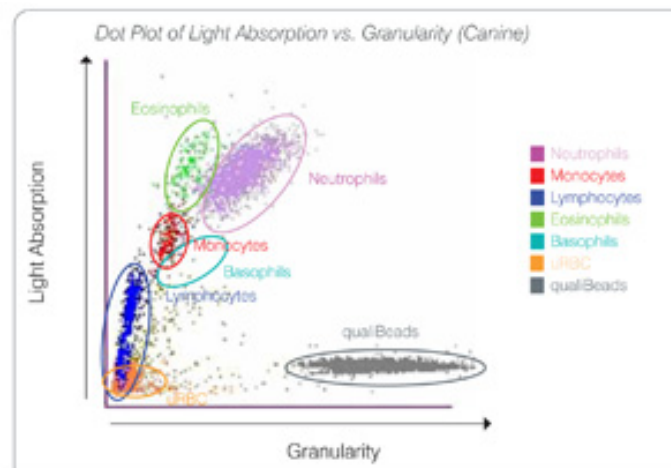


Figure 2.5: WBC differentiation based on laser illumination of cells - signal correlation plot (*Online resource, 2012b*)

The analyzers might also be based on digital imaging (image cytometry) technique which performance developed very rapidly during the last years and offers certain potential advantages including access to parameters not easily measured by flow

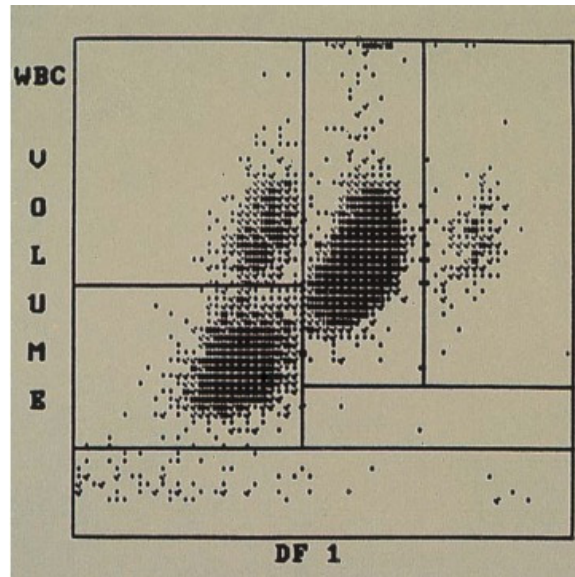


Figure 2.6: Normal WBC differential plot showing volume against light scatter of the cells (McClatchey, 2002)

cytometers.

Tab.2.1 shows current state of the art in performing differential blood counts and should be base for further development of image analysis techniques. The analytic goals specifies allowable error and state of the art current error being made in the tests.

2.1.2 Differential WBC count

There are many diseases associated with amount of different types of leukocytes present in blood. It is therefore very important to distinguish between them in CBC. The main categories of white blood cells can be reviewed by looking at Tab. 2.2.

The possible samples are taken from low quality data set based on closest visual appearance, size and approximated occurrence factor among WBCs. Fig. 2.7 shows the example size distribution for WBCs, RBCs and platelets - in this order. The first histogram at the first look gives us already a very important information. The two peaks are visible. In healthy blood the first constitutes the lymphocyte and the second myeloid (non lymphocyte) count, which consists mostly of neutrophils. This kind of differentiation can be very important. Besides, the cell types have different specialties

Parameter	Analytic goal (error %)	State of the art (error %)
<i>CBC Count</i>		
Leukocytes	16.5	5.4 - 8.8
Erythrocytes	3.75	1.5 - 1.8
Hemoglobin	4.0	1.2 - 1.9
MCV	2.23	2.0 - 2.4
Platelets	6.32	5.2 - 9.8
<i>Leukocyte Differential Count</i>		
Neutrophils	23.4	3.06 - 7.0
Lymphocytes	15.0	4.0 - 11.9
Monocytes	14.8	13.4 - 58.7
Eosinophils	26.0	16.0 - 37.3
Basophils	15.7	35.5 - 155.5
Reticulocyte count	13.0	8.9 - 41.3

Table 2.1: Analytic goals and state of the art (Buttarelli and Plebani, 2008)

and abnormal distribution of one of those can carry more specific information about particular disease or infection. For example lymphocytes can be differentiated as B-cells, T-cells and natural killer cells. All of them work together to prevent infections and regulate immune system. It is in oppose, for example, to neutrophils, which are responsible for elimination of bacteria and fungi.

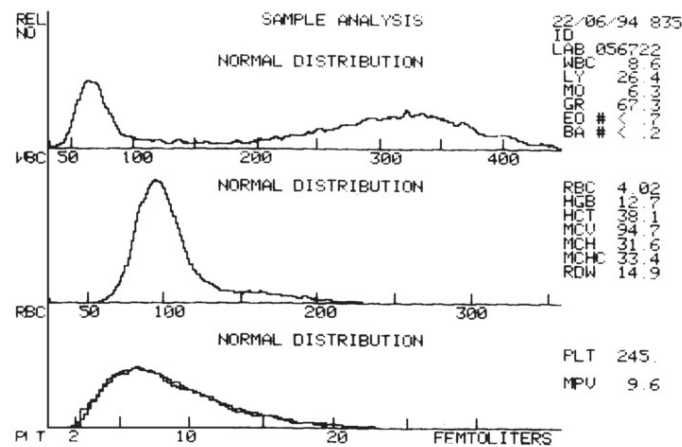


Figure 2.7: Volume distribution of WBCs, RBCs and Plts (Bain, 2006).

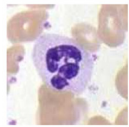


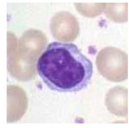


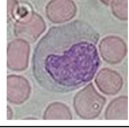


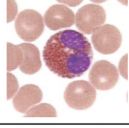
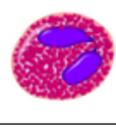


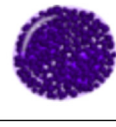

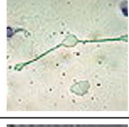
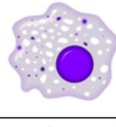


Type	Microscopic appearance	Diagram	Possible LQ sample (guess)	Approximated occurrence	Diameter [μm]
Neutrophil				62%	10-12
Lymphocyte				30%	7-8 small 12-15 large
Monocyte				5.3%	12-20
Eosinophil				2.3%	10-12
Basophil				0.4%	12-15
Macrophage					≈ 21
Dendritic cells					

Table 2.2: Types of leukocytes. Data based on Wikipedia (2013b). The middle column represents examples from our lowest quality data resembling the morphology of the real counterparts.

2.2 Standard biological image analysis pipeline

Most biological image analysis systems, including microscope-based analyzers, consist of modular structure summarized on Fig. 2.8. First image signal is pre-processed, mainly to reduce noise and to make interesting regions more visible. This results in preparation appropriate for segmentation. Then methods are deployed to separate background from foreground and label all of the region of interest (ROIs) on the image, leaving everything else unlabeled. Further chosen features are extracted from the detected objects and classification is performed based on them. After evaluation of

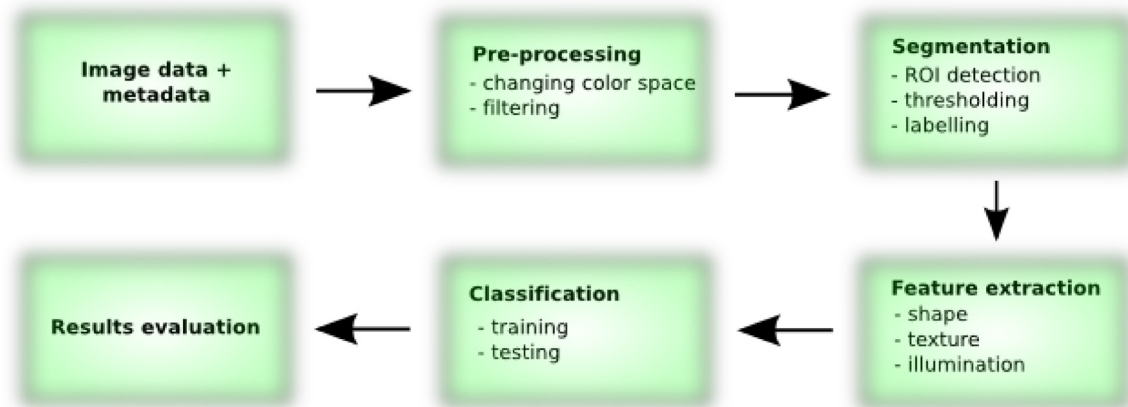


Figure 2.8: Common high-level architecture of biological image analysis systems

the result user or algorithm may decide to further optimize parameters of any of the previous module. This is often iterative process practiced until desired convergence.

2.3 Related work in computer vision

2.3.1 Problems related to low quality images

A lot of good work has been done on different problems related to the project. To programmatically solve the low resolution problem the approach described by Farsiu et al. (2004) was taken. This is a commonly used super resolution algorithm, well suited for camera imperfections, as the main factors taken into account are atmospheric blur, motion and camera blur effects, which is shown in the equation:

$$Y[m, n] = [H_{cam}(x, y) * F(H_{atm}(x, y) * X(x, y))] \downarrow + V[m, n] \quad (2.1)$$

where $*$ is a 2D convolution operation, \downarrow is a discretizing operator, V is the system noise, H is the point spread function (PSF) for the effects, F is the warping factor, X is the real world scene and Y is the obtained noisy image. The original world scene model is optimized based on registration and deconvolution to give the closest resemblance to the noisy image. The data fusion is done by L_p norm and regularization based on bilateral prior is used to remove artifacts and improve the rate of convergence. The initial noise model assumption should be modified to fit the needs of microscopic application.

One of the other challenges we encountered besides low resolution was cell jiggling and random direction flow. This could be solved by image warping, although optimizing deformation for each image can take a lot of time. Partial solution was presented by Kozubek and Matula (2000). The authors used the Delaunay triangulation for nearest neighbour and error detection. After that the calculation of motion vector was performed separately for each ROI what reduced the resource utilization significantly.

2.3.2 Examples of image cytometry systems

Some of the latest approaches to different types of cell recognition in the microscopic images include extracting them by predefined shape templates resembling those popular for specific types of cells (Basu et al., 2013). Besides, Sui and Wang (2013) proposes the use Support Band Filters (SBF), which allows to extract the support region for template based on Convergence Index (CI) for accurate cell nuclei extraction. It is a template matching based method. The other popular approach is based feature extraction. The project by Vink et al. (2013) describes the way of recognizing malaria by detecting the influenced RBCs with fluorescent light, Haar-like features and supervised learning AdaBoost classification which requires manual marking of training set in the beginning. The interesting approach is also presented by Wu et al. (2013), where the authors develop a modified edge detection method where boundary and region information are integrated with active contour algorithm.

To move further towards hematological computer vision systems, useful study was made to classify different types of erythrocytes in patients with anemia by DAS et al. (2012). The developed and compared methods can be used for differentiation of WBCs as well - although they were used with light microscopy and low resolution images are limited in terms of border description possibilities. Going through the system levels first the gray world assumption was used to correct the illumination. After that some of the most popular noise filtering algorithms were compared, including median, Wiener, geometric mean, max and M3 filters, with use of MSE and SNR. The geometric mean filter proved to be the best. The marker watershed algorithm was used as the main segmentation technique. 25 object features were then extracted and compared on their "usefulness" for classification. At the same time different classifiers were used, among which we can find multivariate logistic regression, RBF network,

MLP, Naive Bayes and regression tree, and distinguish the first one as particularly well suited even for the small amount of features. For all of them the F-measure and information gain measure were used to choose the features and adjust parameters.

The work presented by Ongun et al. (2001a) is even more suited to our problem as it deals directly with different types of leukocytes. For extracting the correct borders it uses the popular active contours models. The algorithm is initialized using the morphological operations and, after segmentation, the feature vector is created based on nucleus shape, color and texture. This data is then passed to one of the following classification algorithms: k-NN, learning vector quantization, MLP and SVM.

Similar works include ones by Bikhet et al. (2000) and Hiremath et al. (2010). Both are using the light microscopy and obtain high resolution images. This eliminates most of the problems and allows for reliable segmentation even using only gray-scale images. The enhancement in both cases is performed by using simple (e.g. median) filtering and histogram equalization. Some of the morphological operations are used in the second case. Most usable features include shape descriptions and nucleus to cytoplasm ratio. No sophisticated classification needs to be made in the end. In the second case for example it is only based on simple knowledge based rules where parameters are adjusted based on training data.

Quite educational was also study of Young (1972), which explores similar approaches using the analog camera. He later performs the thresholding and extracts features like color, size, shape of the nucleus and size of the cytoplasm, for cluster analysis. It's interesting to see how good results (93% of accuracy in comparison to about 95%-99% in other described studies) he achieves with low computational resources and simple algorithm. This suggests two things. First - that problem was less complex than it was initially thought. Second - the area of very low resolution recognition is less explored, which creates a lot of space for image enhancement techniques and new robust segmentation methods.

As the low resolution images often don't provide enough information to distinguish them even by humans the feature extraction method with unsupervised learning techniques seems correct. To get the examples reliable enough for testing the hypothesis we need the very accurate segmentation. The template matching methods are the current state of the art and as we deal with few different cell categories it's useful to

get few most distinguishable templates. For that the PCA method was used in some approaches (Yampri et al., 2006, Brunelli, 2009).

2.4 Related work in machine learning

Although almost no longer reported in cell counting research designs nowadays, the rule-based approaches, like ones presented by Prewitt and Mendelsohn (1966), Young (1972), Bikhel et al. (2000), are still often parts of common testing platforms and proof of concept engineering designs. The reason they remain in the process is mainly the clear representation of features, large control over internal classification processes and achievement of often very good accuracy for very specific applications with completely closed and stabilized environment. Unfortunately those methods suffer with development time, adaptation to even small changes and therefore are often useless for rapid prototyping during the design process. In the following chapters we go through the development stages of such system using decision tree consisting of carefully prepared IF...THEN... rules and report comparable high accuracy results for our problem.

The other common solution to the problem of cell classification are what we call classical methods, which can be mostly represented by different kinds of large margin and instance based classifiers. Those are the ones which prevailed for most of the time and usually consist of large number of easy to extract features, often categorized within three modalities of color, shape and texture (Shi et al., 2013, Ongun et al., 2001b, Ramoser et al., 2006). The first one is often based on transforming the image into different color spaces (most often RGB, HSV and $L^*a^*b^*$) and extracting statistical moments of histograms (Shi et al., 2013, Xiong et al., 2010). We believe those kind of features do not preserve much of the important information of the image by themselves. Practically all the information about structures of the elements is lost and as we are interested in distinguishing objects of very similar colors, those features are not preferred. Shape analysis is recently dominated by active contours (or in general level set) methods (Möller et al., 2014, Dzyubachyk et al., 2010). It is not surprising that the energy-based contour delineating methods achieve high accuracy for fluorescent and clear-edge images. They are also needed in cases where the shape of the outer bound of the object is main classification feature. For our purposes none of the most

popular approaches (Chan and Vese, 2001, Paragios and Deriche, 2002) perform well out of the box as the borders of our objects are very blurry, inconsistent and the visual features of cytoplasm are very similar to the background. Also most of the cells in question are mostly circular, so the membrane layout is not as good feature as its relation to nucleus. We could apply the method only to nuclei, but this wouldn't tell us anything about the total area, nor would one shape descriptor suffice to capture many different kinds of nuclei. Last but not least recent analyses (Gurari et al., 2014) show that level set methods need to be tuned to particular cases and our approach here is to present method as general and automated as possible. The last types of commonly used features are texture based and those usually involve measurements of recurring patterns in the image or how much "busy" the area is by applying the edge detection methods or calculating the energy of the region. A lot of information about direction of the recurring patterns can also be extracted by calculating the power spectrum of the image. Shi et al. (2013) lists energy, entropy, contrast and divergence as major texture features, but also Thibault et al. (2014) provides a good example of custom designed texture-based features for cell analysis.

For our purposes we try to restrict ourselves to only one type of features, but to choose those which can make the process as automated as possible - as we believe this is the approach which would provide the most representative results for classic method category. Further we propose combining texture and intensity based information and make the classification rely purely on image moments, sometimes categorized as morphometric features. Although recently forgotten within approaches to cell analysis, it was successfully applied to WBC classification problem before (e.g. Ongun et al. (2001*b*)), still contributes to other very similar problems Chevretils et al. (2009) and is well recognized within most of the common frameworks and textbooks (Hall et al., 2009, Wu et al., 2010, Bankman, 2008). This method treats the full image as 2D joint probability distribution and characterizes its statistical moments, like mean, standard deviation and kurtosis. The nice thing about them is that they can be treated as components of the image and thus can approximate it with desired accuracy. The most popularly applied are, so called, Hu moments (Hu, 1962). However the author derived only the first 7 moments and didn't specify any clear algebraic base from which the further functions for calculating them can be generated. Later this work

was attempted by Flusser (2006). Based on the given formulas we implemented and generated 11 transition, scale and rotation invariant moments and juxtaposed the results obtained by their utilization with those obtained by Hu's method. In the following sections we name them "Flusser moments", after the name of the author.

On top of the feature extraction stage usually a shallow machine learning classifier is applied. Most commonly this involves SVM (Dundar et al., 2011, Xiong et al., 2010, Basavanhally et al., 2010), kNN (Ongun et al., 2001*b*, Ge et al., 2014), random forest (Thibault et al., 2014) or MLP (Ongun et al., 2001*b*). As the first one is the most common and second constitutes a good example of usually weaker approach we chose those two as representatives for obtaining accuracy for automated classic approach. We believe kNN classification is important as it provides a measure of how complex the solution really is. As it doesn't exploit hierarchy of the objects nor applicability of different kernel spaces, it provides a popular bare base line for automatic solutions.

My main contribution from the machine learning perspective is to show the advantages of deep learning approaches to cytological classification. Deep convolutional networks recently achieve the best performance on variety of natural image classification tasks including large international competitions on general (Krizhevsky et al., 2012, Zeiler and Fergus, 2013) and domain-specific (Ciresan et al., 2013) tasks. The work on CNNs goes back to 1980s beginning with famous Neocognitron model (Fukushima, 1988) and later supported by very successful application to hand-written digits recognition problem (LeCun et al., 1998). However their recent success is mainly due to dawn of cheap and accessible GPGPU programming allowing for very fast and efficient matrices manipulations and application of so called "tricks of the trade" targeting and attempting to solve very specific problems or inefficiencies in network design, good examples of which would be use of dropout and rectified linear units.

The similar way of leukocyte classification using the Convolutional Neural Networks (LeCun et al., 1998) was recently published by Habibzadeh et al. (2013). However my work incorporates lower quality images for which the best effort was made to increase this quality, rather than simulating it by reducing color depth and downsampling and thus it is better representative of real world applications. This thesis also provides more thorough analysis of the results and uses different methods to answer questions about applicability of range of feature extraction approaches.

Chapter 3

Tools and related applications

3.1 Background

The chapter provides description of available tools, how they were used during the development process and how they can facilitate further research. GUIs like ImageJ provide a platform for quick testing of the application of certain popular transformation to the images. Libraries like OpenCV and those provided within toolboxes for MATLAB language give access to much larger variety of tools, allow of rapid development of pipelines and often incorporate more recent methods. Moreover the machine learning frameworks, like Pylearn2, are community supported project directed to researches, where often unstable, state of the art methods are available to the user. The chapter finishes with small comparison of available fully fledged applications attempting to solve similar problem to my. They are useful for getting initial ideas and obtaining baselines.

3.2 ImageJ and Fiji

ImageJ (Schneider et al., 2012) is a Java based image processing application with modular structure. It was developed mainly to aid medical image analysis research, but is also widely used as a testing platform for any image analysis task. "Fiji is just ImageJ" is "battery included" version (Schindelin et al., 2012), where many of freely available plugins were incorporated to provide nicely structured extended standard application. Few of those modules appeared to be crucial during our application development.

Cell Counter is a popular ImageJ addition allowing for quick manual marking of objects with different classes in the image and exporting their locations in XML format. In our case it separated my work from the work of hematology expert who was able to quickly gather the information necessary to obtain labeled training set

and present it in a way easy for further automatic processing.

2D Fast Hartley Transform is an FFT counterpart allowing for operations only in real space as oppose to complex space of original Fourier transform. The tools based on it are irreplaceable for designing custom filters - especially for removing any recurrently occurring unwanted pattern, like one presented on Fig.4.9.

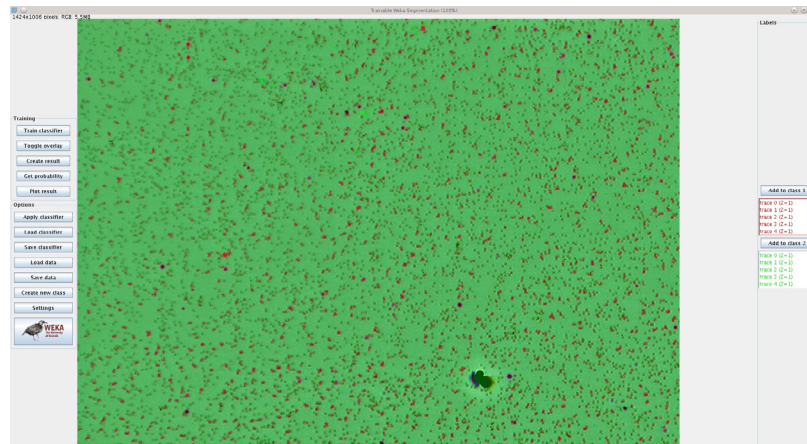


Figure 3.1: WBC analysis with Fiji Weka plugin using low quality image

The Trainable Weka Segmentation plugin in Fiji is a pixel based classifier in which user can specify random areas belonging to each of the chosen classes in the image and obtain automatic classification of regions. This process can be done iteratively until satisfactory results are reached. The feature vector consists of singular numbers each representing either value of the pixel after certain image filtering (e.g. Gaussian, DoG, Gabor, anisotropic diffusion) or local histogram derived statistics. After obtaining feature vector informative enough for every pixel, the ones marked by the user are used as an input to chosen classifier. This can be any classifier implemented in Weka. The rest of the image is used as a test set and the results are presented to the user as probability map of pixels belonging to given class. It can be further thresholded and presented as segmentation (Fig.3.1).

This may sound like a perfect solution to our problem, but although it seems as appealing I believe it has few disadvantages and eliminating them made our method much faster and more robust. Besides limited number of features and classification algorithms choices, the presented feature set is almost completely ignoring the spatial information and thus is not dependent at all on shape, structure and only partially on

size. For our application those are very important features and aiding classification by descriptors like image moments, PCA/ICA components or even the raw pixel values, like in case of deep neural network, provides a good alternative solution by its own. The second important thing is that in most of the cases pixel based classification takes a very long time and our problem can be solved much faster performing more basic and faster initial segmentation and concentrating only on small regions of interest.

3.3 OpenCV

Open Source Computer Vision Library is a library of programming functions for real time computer vision developed by Intel and maintained later by Willow Garage. They define their main goals as (Wikipedia, 2012) :

- Advance vision research by providing not only open but also optimized code for basic vision infrastructure.
- Disseminate vision knowledge by providing a common infrastructure that developers could build on, so that code would be more readable and transferable.
- Advance vision-based commercial applications by making portable, performance - optimized code available for free with a license that did not require to be open or free themselves.

The framework proved useful many times during the development process. Its main advantage is that in combination with Python back-end it allows for rapid development of stand-alone testing platforms. Fig.3.2 shows GUI of application developed as part of the project for the purpose of quick comparison of segmentation methods applied to blood images.

It also provides a good alternative for MATLAB in which different set of methods is available and different development approach can be undertaken. Particular examples used during development of this project include Voronoi diagram (not available in previous versions of MATLAB), larger choice of pre-defined color spaces to view the image, more efficient and providing more control watershed method and gradient based morphological transform. OpenCV also provides built in open source implementation of super resolution method by Farsiu et al. (2004). Unfortunately it is much slower and less efficient than closed source MATLAB version provided by the authors.

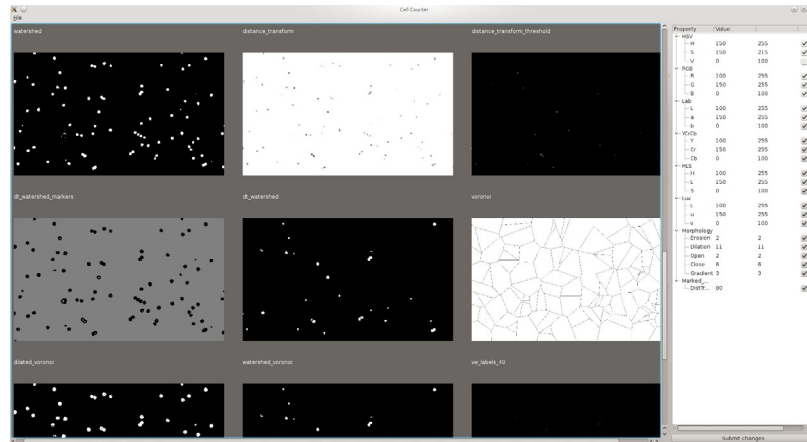


Figure 3.2: GUI developed as part of the thesis for testing variety of segmentation methods available in OpenCV and MATLAB

3.4 MATLAB image processing and computer vision toolboxes

Most of the application development from image processing and segmentation perspective was done in Mathworks MATLAB. The environment, as OpenCV, provides a large library of computer vision methods, which aid testing and rapid development. During testing phase state of the art active contour methods were deployed for contour analysis. Also Fig.4.13 shows a comparison of edge detection method available in image processing toolbox, like ones based on Sobel filtering, Roberts, Prewitt, Canny and Laplacian of Gaussians. Even though the basic version of Hough transform was developed for the purpose of this thesis, the one included in IP toolbox appeared to be faster and more accurate. The toolbox also provides a set of basic filters used in the project, like median filtering for noise removal or Lanczos filter for resolution enhancement. MATLAB environment also provides a great testing platform for many computer vision methods developed as part of the outside research, like Farsiu et al. (2004) superresolution, anisotropic diffusion, calculations on MRFs, superpixel segmentation, graph cuts and more basic like custom histogram adjustments, converting between color spaces, data visualization and plotting the final results.

3.5 Pylearn2, Cuda Convnet and GPGPU programming

All of the classification tasks were performed using external tools other than ones implemented in MATLAB. The last part of the thesis focuses on development of

robust convolutional neural network for leukocyte differentiation. The real advent of such networks began when they started to be trained using GPU architecture, mostly implemented with CUDA library. This allowed to obtain the results in reasonable time for multiple tests and better adjustment of hyperparameters. The most popularly deployed, ready to use, CNN implementation was Cuda-Convnet by Krizhevsky et al. (2012). The framework provides a lot of extensions to standard CNN, like momentum, weight decay, different activation functions to choose (including rectified linear), weights normalization, ways to initialize ones own weight matrices, splitting the network and much more. Everything can be described within a simple configuration file and examples are provided for networks winning famous ImageNet competition. At the time of the first release this was also the most optimized publicly available and open source code for training CNN and it earned itself places in other popular machine learning frameworks like Pylearn2 (Goodfellow et al., 2013) or Caffe (Jia et al., 2014). By itself the code is pretty self contained though and doesn't provide easy ways of extension.

The advantage of Pylearn2 is its modularity and specific design allowing for easy interaction. The developers focus on researchers as the main target group, which allow us to directly focus on solving certain problems by writing own extensions instead of digging through low level parts of code and changing its whole structure. The design pattern combines three most important machine learning modules - the dataset - converting any data to common format, model - describing the actual network architecture or other designs appropriate for variety of machine learning methods, often specified within YAML configuration file, and algorithm - defining how the model should be trained. In the application development I took advantage of both frameworks, mostly to better visualize the results, search for inefficiencies, extend the networks and test latest training approaches, and finally to develop my own models and train them in the same fashion.

3.6 Popular cell analyzing applications

Besides mentioned frameworks there are many other great applications, often supplied with fully fledged GUIs and created mainly for cell analysis, partially used in the

development process and providing baselines for our approaches. Cell Profiler (Carpenter et al., 2006) is definitely the most compound one. It is based on construction of full pipelines (called recipes) for cell analysis from the set of available modules. This can be thought of as a kind of specialized visual programming system. Many methods are available for basic pre-processing and threshold-based segmentation, after certain filtering, like Otsu's method, basic adaptive threshold or mixture of Gaussians. More advanced methods are also applied for edge and contour detection to aid in separation and shape description. For that, among the others, the application provides Laplacian of Gaussians filtering and energy based methods, especially geodesic active contours.

Pixcavator is based on applied algebraic topology and constructs a level set method in this framework in order to find connected components in the image. This approach works well when the image can be nicely filtered from noise before.

Cell Tracer (Wang et al., 2005) is a MATLAB-based application developed at Duke University primarily to segment yeast and E.coli bacteria on very clear fluorescent microscopy pictures. The main leverage for finding initial candidates is a range filter, which can be very subjective to noise. Even though this noise is later partially filtered by morphological operations, I found that the concave disk shape of the red blood cells prevents some of the cells to be correctly recognized. Although in overall the algorithm gave good results on our RBC low quality test image.

Cell Cognition Cecog Analyzer (Held et al., 2010) is another large application with multiple components similar to Cell Profiler. It is based on VIGRA - general purpose computer vision library and mainly just provides a user interface, where user can specify functions parameters to be called on the input image. The selection of tools is again matched directly with those needed for cell analysis. Although very powerful and useful for bio-engineering, it doesn't provide fully automated pipeline and can be done with the same efficiency using other available scripting languages.

CellC (Selinummi et al., 2005) is a simple MATLAB software which worked best in our RBC test. Even though it was meant to work on fluorescent images, just after simple pre-processing on our side - choosing one or two of the HSV channels and stretching the histogram - it achieved remarkable accuracy. The test consisted of counting erythrocytes on small crop of a very poor quality image in which the cells were first counted manually to provide a ground truth. The algorithm achieved 99%

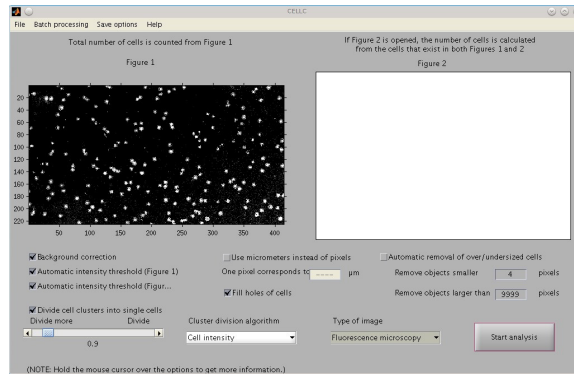


Figure 3.3: CellC GUI

accuracy on this task and the methods involved mainly adaptive thresholding for initial count and marked watershed for cluster separation. Even though the image was clear from other objects resembling red blood cells and didn't contain anomalies present on some of our other images (like moving objects or reflections), it provided a very encouraging result which inspired further development of erythrocyte analysis part of our cytometry pipeline. GUI during our analysis process can be seen on Fig.3.3.

The other popular applications which had less direct influence on our process, but still are worth noting here, are Ilastik (Sommer et al., 2011), Cell-ID and GemIdent (Kapelner et al., 2007).

Chapter 4

Algorithms and methods

4.1 Background

This chapter provides derivations and detailed description of selected methods used to facilitate or suggested as alternatives to achieve the final results. It is meant to serve as a guide for implementing components of computer vision pipeline for cell analysis. The results chapter highly depends on references to this chapter, whenever more detailed explanation was needed. Additionally the derivations of equations not directly needed for implementation, but providing insights into their motivation and purpose, were presented in appendices.

The following sections are presented in order common for standard medical image analysis system (as detailed on Fig.2.8). It starts with image processing methods needed for noise removal and extraction of crucial information from the original images. Then it switches to segmentation, where the description of all the methods needed to extract the location and shape of the cells was presented with particular focus on applications to red blood cell analysis. We concluded with features extraction - mainly by image moment descriptors - and three popular approaches to classification.

4.2 Image pre-processing

4.2.1 Flat-field correction

Immediately after acquisition the images have uneven intensity. To correct for that the pseudo flat field correction was used. In FFC we use flat field image collected from camera (I_{FF}), which is usually the picture of background with even intensity we want to correct to. Later we use it to correct the future images by applying:

$$I_{FFC} = \frac{I_{org}}{I_{FF}} * g \quad (4.1)$$

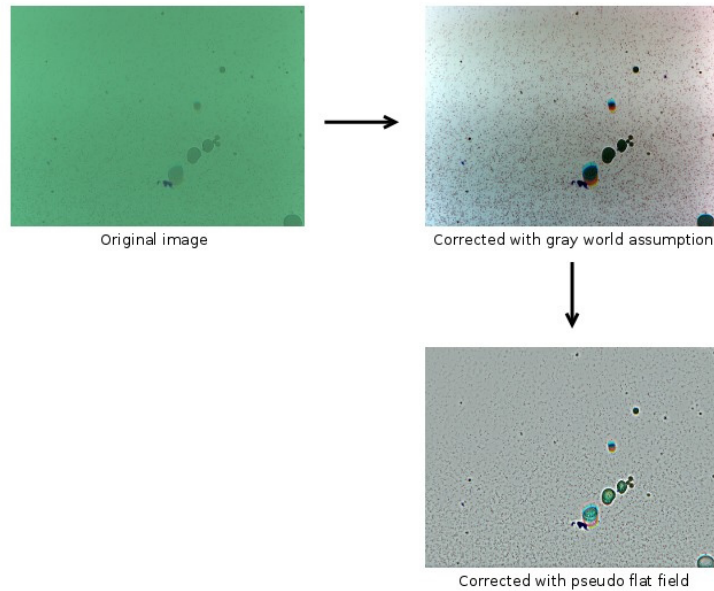


Figure 4.1: Flat field correction

where I_{org} is the original image, I_{FF} is the collected flat field image, g is correction factor equal to mean intensity of the brightest channel, which lifts all the other channels to the same level. The latter called adjusting the image with gray world assumption. It makes sure there is no color bias in direction of any particular channel. I_{FFC} is the output corrected image.

In pseudo FFC instead of acquiring the FF from camera we can create pseudo FF by convolving the image with mean or median filter to obtain approximate background values in each part of the image. The whole process can be seen on Fig.4.1.

4.2.2 Contrast enhancement

Contrast enhancement is usually done by histogram stretching or histogram equalization. The former preserves most of the color information but makes the image more clear for the human eye. The latter usually exposes salient features more than the rest of the image. Following the examples we can see the original image and its value histogram channel on Fig.4.2. The value channel is a pseudo-channel defined as $I_v = \max(I_r, I_g, I_b)$.



Figure 4.2: Original image

To get a stretched histogram we would cut off the chosen fraction (usually between 1% and 5%) of the highest and lowest intensities and treat them respectively as black and white. Then we would distribute the other values evenly across the available space. Fig.4.3 shows the result of slightly adjusted stretching where larger fraction of intensities was cut off on the bright side to leave better visual differentiation of darker cells.

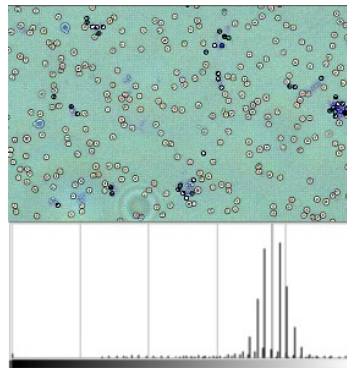


Figure 4.3: Histogram stretch

To get most of the information from the picture we can equalize its intensities which creates the histogram close to the uniform distribution. As the more frequent intensities get more spread across the range it is easier to spot slight hidden differences between them. The result on our image can be seen on Fig. 4.4. To get it we can calculate the intensity probability distribution from histogram and the its cumulative distribution function (*cdf*). Then we apply $I_{x,y} := cdf(I_{x,y}) * 255$ to each pixel.

This gives us very unclear result as most frequent intensities constitute noisy background. What we want is the opposite - suppress the changes in background,

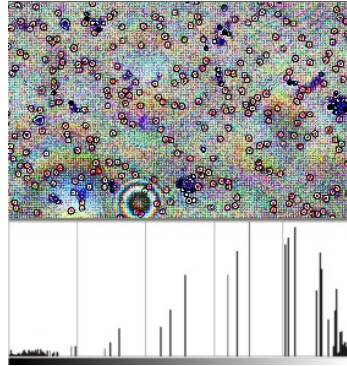


Figure 4.4: Histogram equalization. The histogram is uniform in a sense of having *cdf* representing as even accumulation as possible. The constraint of limited sampling prevents us from splitting the spikes.

thus reducing the noise, and make the more informative pixels more distinguishable. This can be done for example by calculating *cdf* of the probability distribution of the complement histogram - the one which shows the difference between the value and maximum histogram value for each bin, leaving the empty bins as 0 to get the clearer result. The image obtained with this approach can be seen on Fig.4.5.

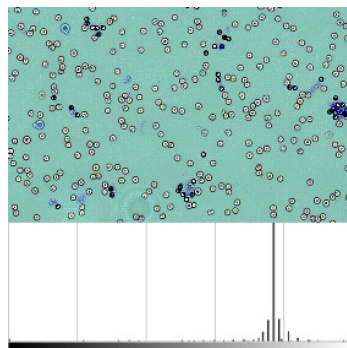


Figure 4.5: Histogram with custom look up table

4.2.3 Denoising

Anisotropic diffusion

Anisotropic diffusion filtering by Perona and Malik (1990) in its numerical form represents itself as an integration of the following PDE in the image space over time:

$$\frac{\partial I(x, y, t)}{\partial t} = \text{div}(g(\|\nabla I\|)\nabla I) \quad (4.2)$$

where $g(x)$ can be an arbitrary function and the whole equation represents constrained heat flow in the image. The algorithm can be best understood via mathematical framework of differential geometry. By treating the image as a 3 dimensional landscape we can describe its properties continuously the same way we describe curves or surfaces.

The derivation and more detailed explanation of Eq.4.2 is presented in Appendix A. The two common choices for $g(x)$, which proved to be useful are $g(\nabla I) = e^{-(\frac{\|\nabla I\|}{\kappa})^2}$ and $g(\nabla I) = \frac{1}{1+(\frac{\|\nabla I\|}{\kappa})^2}$. The former allows for specification of priority of high contrast edges over low contrast and the latter of wide regions over smaller.

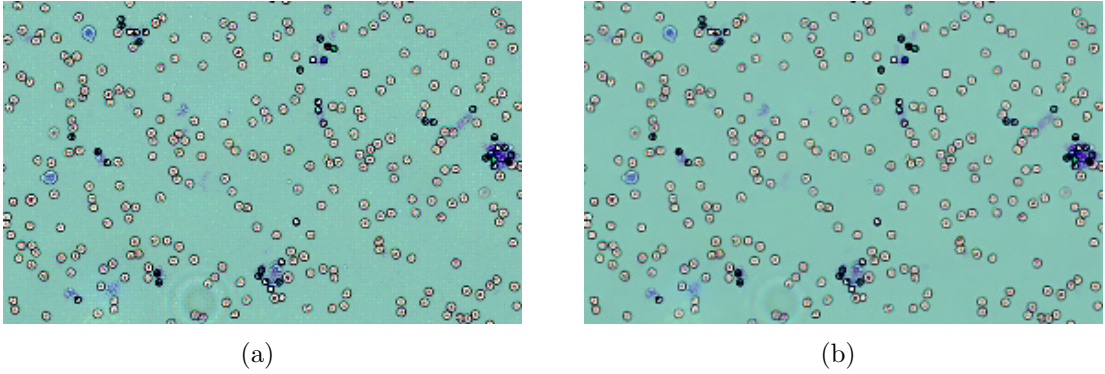


Figure 4.6: Anisotropic diffusion with parameters $\kappa = 20$, $\lambda = 0.25$ and $g(\nabla I) = e^{-(\frac{\|\nabla I\|}{\kappa})^2}$ after 5 iterations. Before (a) and after (b).

To implement this in the computer the following numerical scheme was proposed. First we perform the discretization of the Laplacian operator:

$$\lambda \operatorname{div}(g(\|\nabla I\|)\nabla I) \rightarrow \lambda[c_N \nabla_N I + c_S \nabla_S I + c_E \nabla_E I + c_W \nabla_W I] \quad (4.3)$$

where we calculate the gradient of the image in the four directions. λ in heat equation is just a parameter specifying the speed of thermal diffusion and is usually set for 1 in simulations. As in discrete form we have 4 directions, to keep the system stable we usually set $\lambda = \frac{1}{4}$ or lower for slower and possibly more accurate convergence. Next we perform the integration over time:

$$I_{i,j}^{\tau+1} = I_{i,j}^{\tau} + \lambda[c_N \nabla_N I + c_S \nabla_S I + c_E \nabla_E I + c_W \nabla_W I]_{i,j}^{\tau} \quad (4.4)$$

The gradients in discrete image can be obtained by simple difference:

$$\nabla_{\{N,S,E,W\}} I_{i,j} = I_{a,b} - I_{i,j} \quad (4.5)$$

where $\{a, b\} \in \{(i-1, i, i+1), (j-1, j, j+1)\}$ according to the direction, which gives us 4 gradient images. The conduction coefficients can be computed as follows:

$$c_{\{N,S,E,W\},i,j} = g(\|\nabla I_{a,b}^T\|) \quad (4.6)$$

with $a, b \in \{(i - \frac{1}{2}, i, i + \frac{1}{2}), (j - \frac{1}{2}, j, j + \frac{1}{2})\}$ this time. The κ parameter in either of $g(x)$ function alternatives shown above can be adjusted by hand or automatically. One of the methods for the latter is to adjust it as a specified factor (in %) of the integral of the absolute values of calculated gradients. The implementation used in testing¹ allows only for manual setting empirically suggested to be in the range [20, 100]. This is the most important parameter regulating the spread of diffusion and thus behaviour of the whole system.

The algorithm was used as an alternative median filtering as a denoising approach for the presented method and the example results for low quality image can be seen on Fig.4.6.

4.2.4 Resolution enhancement

As the current pixel size on the sensor is not enough to perform useful holistic analysis we need to project the image onto a higher resolution space. Two popular approaches to facilitate that are interpolation and super-resolution. The interpolation is usually much faster but far less accurate. Below are described two efficient methods exploiting those two approaches.

Lanczos filter

Lanczos filter is the interpolation method used in image transformations, like rotation and scaling, claimed to keep the best compromise between low and high frequencies - between sharpness and smoothness (Wikipedia, 2013a). It is based on convolving the image with so called Lanczos kernel, which in 1D case consists of windowed dilated normalized sinc function defined as:

$$\text{sinc}(x) = \frac{\sin(\pi x)}{\pi x} \quad (4.7)$$

¹By Peter Kovesi from The University of Western Australia

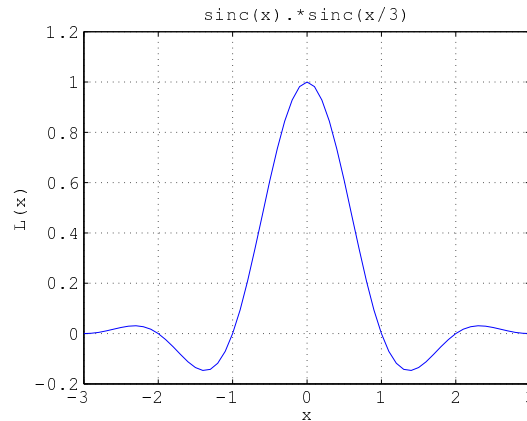


Figure 4.7: Lanczos kernel for window size $a = 3$

The window is centered on the coordinate system's origin and its size is defined by parameter a , specifically:

$$L(x) = \begin{cases} \text{sinc}(\pi x) \text{sinc}(\frac{\pi x}{a}) & \text{if } -a < x < a. \\ 0 & \text{otherwise.} \end{cases} \quad (4.8)$$

as defined on Wikipedia (2013a). The example of a window can be seen on Fig.4.7.

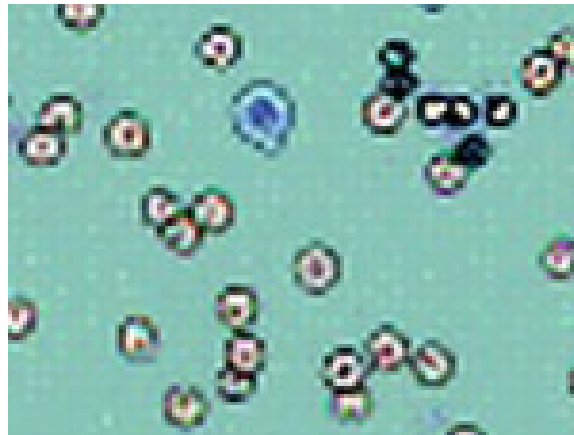


Figure 4.8: Example application of Lanczos filter application

We are able to obtain the value between any discrete samples by applying:

$$S(x) = \sum_{i=\lfloor x \rfloor - a + 1}^{\lfloor x \rfloor + a} s_i L(x - i) \quad (4.9)$$

where s_i stands for sample. The example result can be seen of Fig.4.8.

Super-resolution

Much more accurate method of resolution enhancement can be developed by reconstruction of the image from multiple frames via algorithms used for denoising and deconvolution. One of the method achieving good results out of the box was proposed by Farsiu et al. (2004). In short it can be defined as an optimization of the following equation:

$$\hat{X} = \underset{X}{\operatorname{argmin}} \sum_{k=1}^N \|DF_k HX - Y_k\|_p^p + \sum_{l=-P}^P \sum_{m=0}^P \alpha^{|m|+|l|} \|X - S_x^l S_y^m X\|_1 \quad (4.10)$$

where X is a 1D vector representing the image, N is the number of the images, Y_k is the vector representing input image, H represents assumed distortions, F_k is a motion operator describing repositioning of each element of image X , D is downsampling operation and the whole second term a Total Variation regularization term in which S is an affine shift matrix. H is the model of the Point Spread Function (PSF) defined by the user as matrix containing the filter description for each position in the image. In my approached I used the default Gaussian filter with standard deviation $\sigma = 1$. The parameters and derivation of this equation are explained in more details in Appendix B. The optimization process was further proposed to be split up into two major steps: 1) Noniterative data fusion - finding a blurred HR image $\hat{Z} = H\hat{X}$.

$$\hat{Z} = \underset{Z}{\operatorname{argmin}} \sum_{k=1}^N \|DF_k Z - Y_k\|_p^p \quad (4.11)$$

2) Iterative deblurring-interpolation.

$$\hat{X} = \underset{X}{\operatorname{argmin}} \sum_{k=1}^N \|A(HX - \hat{Z})\|_p^p + \sum_{l=-P}^P \sum_{m=0}^P \alpha^{|m|+|l|} \|X - S_x^l S_y^m X\|_1 \quad (4.12)$$

where $l + m \geq 0$. A is a diagonal matrix containing the square roots of the number of frames that contributed to generation of each element of \hat{Z} .

The first step can be solved analytically by solving a differential equation:

$$\frac{\partial}{\partial Z} \left[\sum_{k=1}^N \|DF_k Z - Y_k\|_p^p \right] = 0 \quad (4.13)$$

which for $p = 2$ takes the form:

$$\sum_{k=1}^N F_k^T D^T (DF_k Z - Y_k) = 0 \quad (4.14)$$

and was proved (according to Farsiu et al. (2004)) to be the pixelwise average of frames after registration.

In the second step the solution is found numerically by gradient descent method with the update step defined as:

$$\begin{aligned} \hat{X}_{n+1} = & \hat{X}_n - \beta \{ H^T A^T \text{sign}(AH\hat{X}_n - A\hat{Z}) + \\ & \lambda \sum_{l=-P}^P \sum_{m=0}^P \alpha^{|m|+|p|} [I - S_y^{-m} S_x^{-l}] \text{sign}(\hat{X}_n - S_x^l S_y^m \hat{X}_n) \} \end{aligned} \quad (4.15)$$

4.2.5 Artifact removal

Frequency domain filtering

While applying super-resolution algorithm to the images with increasing resolution factor, at certain moment we start seeing the artifacts caused by this operation in a form of frequent pattern visible on Fig.4.9. These are singularity points in which no more information from the original images can be added. To still enjoy higher resolution, e.g. to smooth the transitions between pixels for gradient based calculations, we might want to remove them.

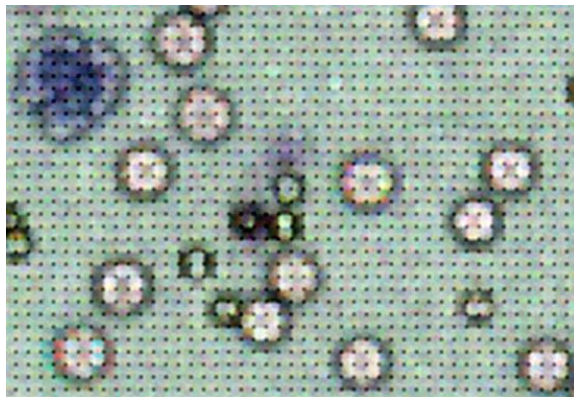


Figure 4.9: Super-resolution image with resolution enhancing factor too large to handle and thus showing places where information cannot be retrieved.

Patterns with constant frequency are easily visible in frequency domain, i.e. after applying Fast Fourier Transform to the image. To visualize the transform it is a common practice to depict its spectrum $|F(u, v)|$, power spectrum $|F(u, v)|^2$ or logarithm of a spectrum $\ln(|F(u, v)|)$ (where $|z| = |x + iy| = \sqrt{x^2 + y^2}$). The last one usually brings up most details. No matter which one we choose, we can use it to easily

design filter manually or automatically and later apply it to the original transform. The example of such filtering with certain frequencies removed by black circles can be seen on Fig.4.10.

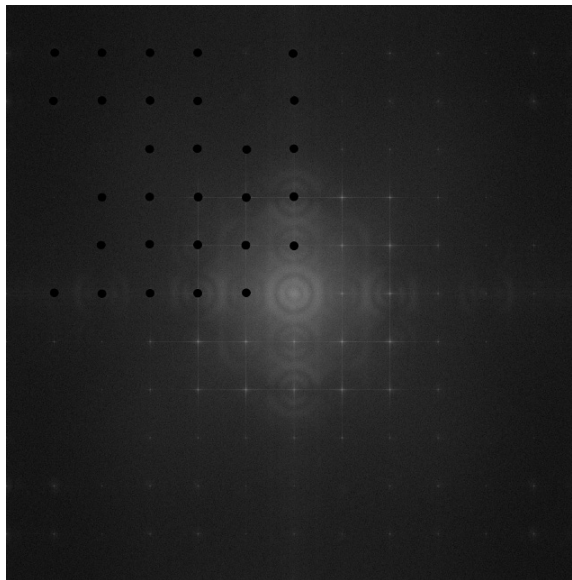


Figure 4.10: Power spectrum of the image on Fig.4.9. The commonly occurring patterns are represented as white dots. In upper left part manual mask was applied to remove them.

To make this automatic its easier to operate on the spectrum parametrizing it in polar coordinates with respect to its center. Following Gonzalez et al. (2004) we can map the defined function $S(r, \theta)$ to two 1D profile functions allowing us to move either by radius with summary of all angles or the other way around:

$$S(r) = \sum_{\theta=0^{\circ}}^{180^{\circ}} S_{\theta}(r) \quad (4.16)$$

$$S(\theta) = \sum_{r=0}^R S_r(\theta) \quad (4.17)$$

where θ is the angle, r is the radius, which can be measured in number pixels, and R is the radius of the largest circle fitting into the image. The latter equation gives us a possibility to look for the angles with high activity. Once we define them - either manually or automatically, we can save them for filtering feature images from the same source. The next step is to define their location of unwanted frequencies. This time we can draw profiles for the specified angles using $S(r, \theta)$ and again look for higher peaks.

To make it automatic it is usually enough to smooth the function with Gaussian filter and collect locations of all of the local maxima (not counting the central one). After getting the coordinates, we can filter them out in spectral image by multiplying with small inhibiting Gaussian filters centered around them. After that we can apply the inverse the FFT image and The final result can be inspected on Fig.4.11.

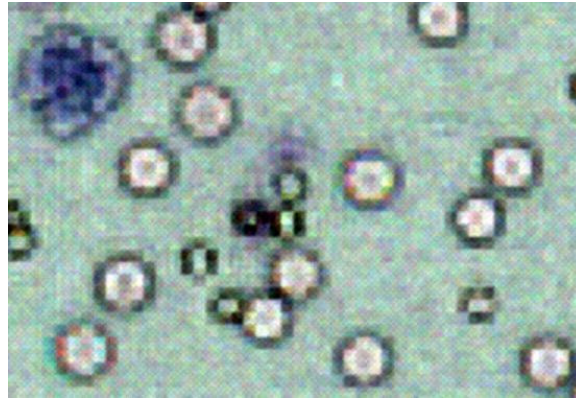


Figure 4.11: Image corrected by FFT filtering

4.2.6 Color space

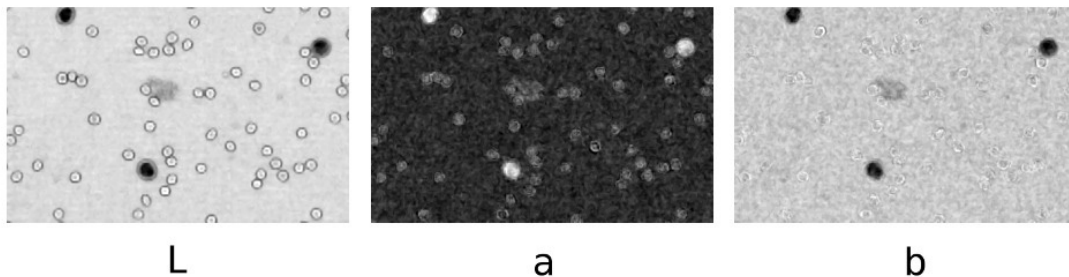


Figure 4.12: L a b color space channels

It is valuable to notice what happen to the Fig.4.12 if we split it into separate channels within Lab (more specifically CIE $L^*a^*b^*$) space, where L stands for lightness and a and b are two color components. Visually those components present themselves as a trade off between blue and magenta (a) and between blue and green (b). This space allow for easy separation of WBCs from the background by describing them as low intensity objects which tend to have color closest to mixing of blue and magenta.

It is useful not only visually though. As we assume those cells to be extremal cases in this color space, we can safely threshold them cutting off certain percentage of histograms, thus not having to worry about lighting or contrast conditions. This also shows that we were lucky in this case as those particular color values mixing gave so good result. We might want to create our own color spaces for this kind of visualization for other objects.

4.3 Segmentation and candidates extraction

After pre-processing stage the next step in object detection is to build segments of connected pixels, where each of them covers maximally the specified object and doesn't cover anything else in the image. To do that we can start with simple thresholding, which works particularly well if we obtain a good representation of the data from the previous step, e.g. mentioned denoised and enhanced image in Lab color space for WBC detection or violet image in case of RBCs. Unfortunately this kind of segmentation is usually quite noisy and results in objects being split in many parts or (especially in case of our cells) ending up connected in clusters labeled as the same object. To correct for that cases utilization of methods presented below is proposed.

4.3.1 Hough transform

Red blood cells often appear in clusters and each cluster will be counted as one object. This produces an unacceptable error in the count and the mean area measure. There are often too many of them on the picture to safely discard them. One of the frequently used approaches to deal with this problem is edge and contour detection. In order to detect edges in the image we can choose among several common gradient based techniques. The most popular ones can be compared on Fig.4.13 where 405nm violet RBC image was used as an example. For further processing we usually stick with Canny detector as it is most general. After doing that in most of the cases we see that edges are broken or still represent clusters. To deal with both issues with help comes the Hough transform, which is a simple contour searching method. It can be described best in terms of finding the specified simple shape based on connected contours in the image. By choosing circle as a model we are able to divide most of the shapes comprising connected circles and join together those which got broken.

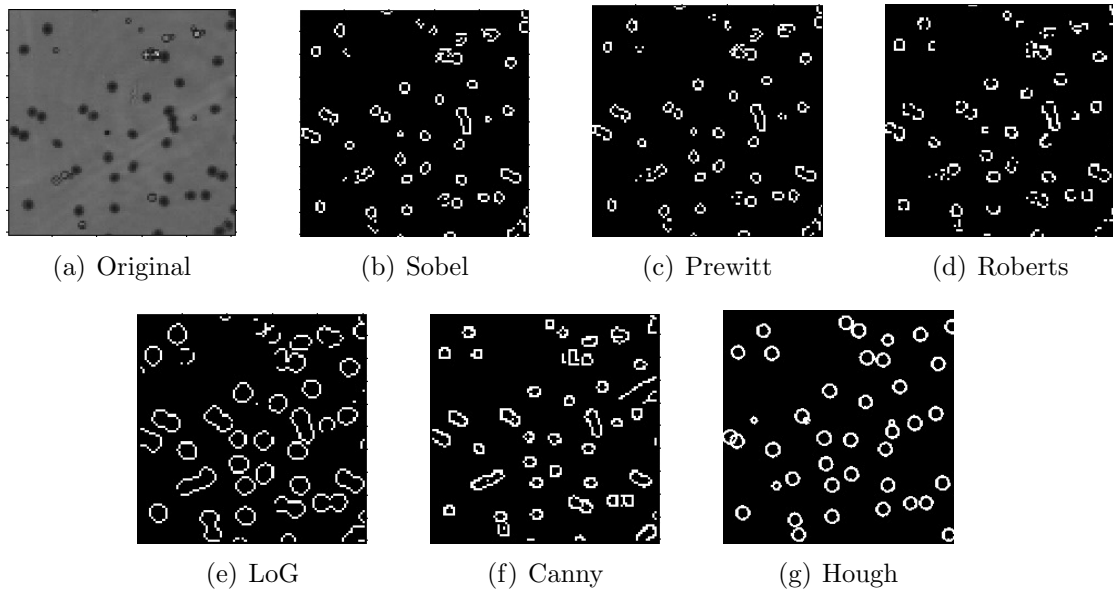


Figure 4.13: Edge detection with different methods.

For visualization purposes it is best to work with the simplest possible contours - the lines. Treating $y = mx + b$ as our model we initialize the algorithm with the chosen range of slopes and biases so to cover the whole image with lines. After that we build a matrix in which rows represent m and columns b settings. We then gradually fill its bins with the number of binary points through which the line with particular (m, b) parameters is passing. There is one obstacle in this approach though. We can't consider the case of $x = const$, thus we can't measure commonly occurring segments described by lines going directly from top to bottom. For that we need to parametrize it in different way and the popular approach is $\rho = x \cos \theta + y \sin \theta$, where ρ is the distance represented by magnitude of the normal vector beginning on the curve and ending on the coordinate system's origin and θ is the angle showing the negative slope of the line (i.e. positive angles direct the line down from the x-axis). The matrix which we would fill up then consists of ρ rows and θ columns and can be visualized as in the example shown on Fig.4.14.

What we are searching for are the lines passing through the highest amount of points in the image and thus for peaks in the shown matrix. Those within threshold specified by us are then projected back to the image as the detected lines. Quite often instead of projecting the parametrized line itself only the points which contributed to its creation are shown, which allows for detecting particular edge segments in the

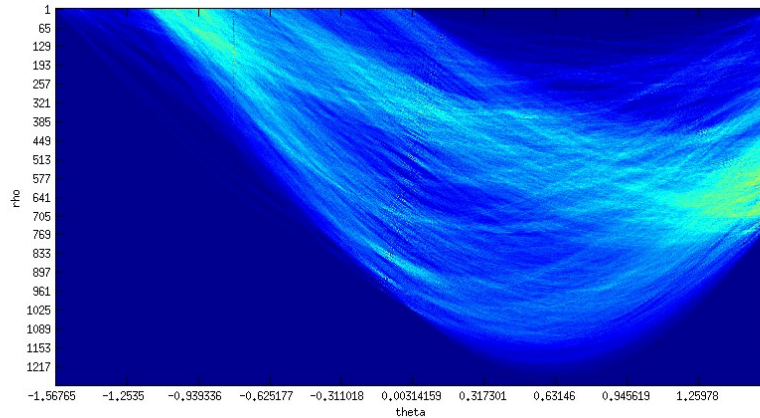


Figure 4.14: Example of Hough transform for line detection

image.

The Hough transform generalizes to any parametric curve. So we can try detecting any shape which can be described by equation formulated as $c(u_1 \dots u_n) = \{x_1(u_1 \dots u_n) \dots x_m(u_1 \dots u_n)\}$. Usually the simpler shapes are chosen though as the computation becomes very expensive with higher number of parameters and simpler shapes usually generalize to many types of objects. For example circle can be expressed as $(x - x_0)^2 + (y - y_0)^2 = r^2$. Here the parameters are x_0 , y_0 and r instead of ρ and θ , which is already a 3D case. It is harder and more expensive to visualize the transform matrix, but the idea remains the same.

4.3.2 Voronoi diagram

Voronoi diagram is the result of applying the equation which can be best described as:

$$Voronoi(q_i) = \{x : \|x - q_i\| \leq \|x - q_j\| \forall j \in S\} \quad (4.18)$$

where $S = \{q_1 \dots q_n\} \subseteq R^2$. So that it is a diagram dividing the space into convex regions each representing all the points x which are closer to the generating point q_i than to any other point q_j . One of the ways we can look at this is that we define a 2D distance function $f_i(x) = \|x - q_i\|$. As in our case each point x and q_i is a 2D point on the plane we can view this result as 3D cones perpendicular to the surface with origin at point q_i defining the closest distance to it from any other point x . At some point the cones will start to intersect each other and, because each of them is exactly

the same, projection of those intersections on the surface will create straight lines, which define our diagram. Formally the regions can be viewed as the lower envelope of the function, i.e. $L_F(x) = \min_i f_i(x)$.

In the project pipeline the generating points were generated based on Hough transform segmentation to decluster red blood cell clumps. The details of utilization of the diagram are described in further sections.

4.4 Feature extraction

4.4.1 Size and nucleus - cytoplasm ratio

One of the important measures when dealing with WBCs are size and nucleus - cytoplasm ratio. Below few methods are presented to deal with the task of boundary tracking. Although examples are presented only for external contour, the attention is put on approaches which can be directly applied to search for nucleus-cytoplasm boundary. The main problems here are posed by cells which touch other objects - especially those with similar color and intensity - like other cells or clumps of floating stain. The other ones include lack of clear border between cytoplasm and the background. The gradient is much more visible between cytoplasm and nucleus in most of the cases.

Exploiting log-polar representation

Log-polar representation allows for easier visualization of the problems and data manipulation. The following sections presents a simple method for contour tracking developed as part of the project. The idea is that for highly symmetrical objects, like our cells, all the edges which need to be detected are some approximations of the straight lines. This is not true when the cells are not perfectly centered of course, so a lot of effort must be put in ensuring this condition is met as well as possible. After choosing the center, the set of equations converting from Cartesian coordinates to log-polar are given by:

$$\begin{cases} \rho = \log \sqrt{x^2 + y^2}, \\ \theta = \arctan \frac{y}{x} \text{ if } x > 0. \end{cases} \quad (4.19)$$

The result of such transformation for WBC can be seen on Fig.4.15.

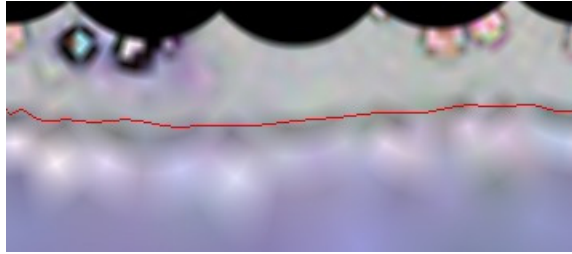


Figure 4.15: Line-fitting to the edge of WBC in log-polar coordinates

The line in the middle is the result of curve fitting - the main part of the algorithm. First the profile s_c , or cross section, of the image is drawn representing its intensity values at each column c and treated as a 1D signal (separate for each channel). It is then smoothed by convolution with Gaussian filter to reduce noise:

$$s'_c = s_c * g(\mu, \sigma) \quad (4.20)$$

where s'_c is the smoothed signal, g is the Gaussian filter and μ and σ are normal distribution parameters defined by the user. Then - just by looking at the image - we can specify that what we are looking for are usually very light regions (cytoplasm) followed by dark regions (boundary, refraction shadow effect not visible as much in higher resolutions). These can be modeled by sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$. After that we can convolve (or correlate - depending on the direction) the profile with this model:

$$l_c = s'_c * \sigma \quad (4.21)$$

to obtain its most probable occurrence locations. We also need to specify the area we are mostly interested in. This is done by multiplying the result with Gaussian function having the mean in the mean boundary distance for WBC, based on previously collected examples.

$$l'_c = l_c g(\mu_b, \sigma_b) \quad (4.22)$$

where μ_b and σ_b are probable locations and spread of the boundary. When we finally obtain our corrected profile l'_c , we look for peaks - which tell us where sigmoid function fitted best (Fig.4.16).

The procedure is repeated for each profile in the image. All the time only the highest peak is collected. This gives us a line similar to the one visible on Fig.4.15 but full of outliers originating from profiles with not as clear boundary. In our case those

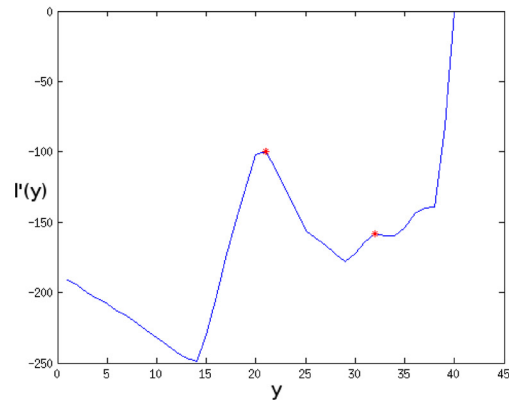


Figure 4.16: Log-polar image profile peaks. y represents the pixel coordinate of particular row of the image and $l'(y)$ is a corrected profile. The profile was inverted to show peaks as local maxima.

can be successfully removed with polynomial fitting of order 5 in most of the cases, although the algorithm could take the advantage of order adaptation to particular case and it is one of the suggestion for future work.

In the end the image is transformed back to Cartesian coordinate system and the final result can be seen on Fig.4.17.

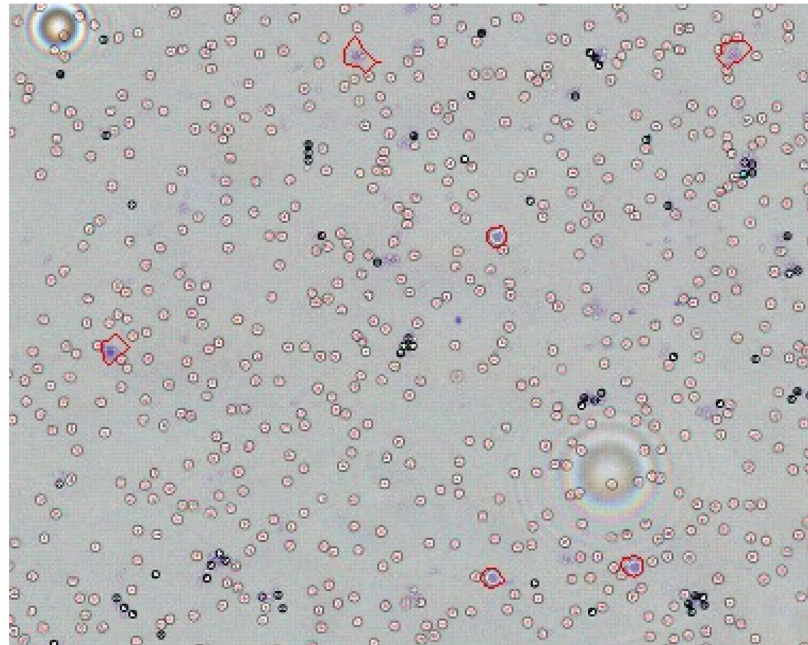


Figure 4.17: Result of WBC contour tracking in log-polar coordinates

As we can see the clearly defined borders are obtained for obvious large WBCs. This

is not the case for the cells of smaller sizes with borders blending into background. Although not as successful in its simplest form, the method might be a good starting point as it is easily expandable and allows for simple incorporation of basic assumptions about symmetrical objects.

Region growing

Slightly more accurate in the tests on leukocytes was the region growing method. It defines the family of different algorithms with a simple idea behind. We start by choosing single pixel on each of the objects of interest. Their locations are usually found by initial segmentation. Those points are further called seeds. In our case the centers of WBCs are chosen and as we keep samples of the centered cells on separate patches, only one seed is chosen per image.

Following from there the algorithm looks at surrounding pixels (either 4 or 8 - depending on chosen connectivity) and compares them with specified threshold values. If pixel meets the condition it is included to the region. Usually the intensity in each channel is used as a similarity measure, but it can be anything. Next we treat each of the newly attached pixels as seeds and keep adding more and more of them to the region until we use up all of the possibilities. It is a common approach to further treat difference between pixel intensity and average region intensity as the measure. This way the algorithm can adapt to small changes within the same object. For multi-seed approach the addition criterion must be specified for joining the regions which come across each other during the development of the process. Again usually the average intensity is compared and the regions are either joined or stay split as separate objects.

For defining detailed boundaries of the cells the single-seed approach is needed. Euclidean difference between 2D vectors representing intensities in L and b channels of Lab color space were used as a metric. To correct for illumination and contrast changes we can adaptively adjust the threshold with step of 0.02 of the normalized difference between regions until no more than 70% of the image is filled. After passing this thresholds the experiment presented sudden jump in activity and filling of all the background in most of the cases. The results of such approach for set of WBCs are presented on Fig.4.18.

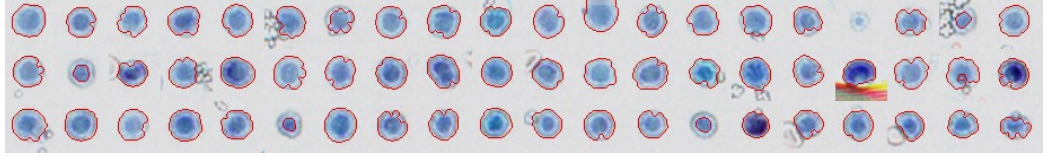


Figure 4.18: Region growing results for set of WBCs

4.4.2 Image moments

Image moments can be thought of moments of 2D probability distribution which is represented by image itself. It can be proved that calculating infinite amount of moments we can obtain the exact reconstruction of the original distribution. Their nice property is that they are sorted by the influence and thus often obtaining just few of them is enough to make correct class discrimination. The properties and full derivation of descriptors used for WBCs classification can be found in Appendix C. The derived Hu moments (Hu, 1962) can be described for each channel by the following equations:

$$\begin{aligned}
 I_1 &= \nu_{20} + \nu_{02} \\
 I_2 &= (\nu_{20} - \nu_{02})^2 + 4\nu_{11}^2 \\
 I_3 &= (\nu_{30} - 3\nu_{12})^2 + (3\nu_{21} - \nu_{03})^2 \\
 I_4 &= (\nu_{30} + \nu_{12})^2 + (\nu_{21} + \nu_{03})^2 \\
 I_5 &= (\nu_{30} - 3\nu_{12})(\nu_{30} + \nu_{12})[(\nu_{30} + \nu_{12})^2 - 3(\nu_{21} + \nu_{03})^2] + \nu_{21} - \nu_{03})(\nu_{21} + \nu_{03}) \\
 &\quad [3(\nu_{30} + \nu_{12})^2 - (\nu_{21} + \nu_{03})^2] \\
 I_6 &= (\nu_{20} - \nu_{02})[(\nu_{30} + \nu_{12})^2 - (\nu_{21} + \nu_{03})^2] + 4\nu_{11}\nu_{30} + \nu_{12})(\nu_{21} + \nu_{03}) \\
 I_7 &= (3\nu_{21} - \nu_{03})(\nu_{30} + \nu_{12})[(\nu_{30} + \nu_{12})^2 - 3(\nu_{21} + \nu_{03})^2] - \nu_{30} - 3\nu_{12})(\nu_{21} + \nu_{03}) \\
 &\quad [3(\nu_{30} + \nu_{12})^2 - (\nu_{21} + \nu_{03})^2]
 \end{aligned} \tag{4.23}$$

where ν_{pq} is a normalized central moment (Appendix C). The Flusser moments (Flusser et al., 2009) can be generated from the following algebraic basis:

$$B = \{\Phi(p, q) \equiv c_{pq}c_{p_0q_0}^{p-q} : p \geq q \wedge p + q \leq r\} \tag{4.24}$$

in which c_{pq} is a complex moment (Appendix C). To obtain the final results the following 11 moments were used per each channel:

$$\begin{aligned}
F_1 &= \Phi(1, 1) \\
F_2 &= \Phi(2, 1) \\
F_3 &= \text{Re}(\Phi(2, 0)) \\
F_4 &= \text{Im}(\Phi(2, 0)) \\
F_5 &= \text{Re}(\Phi(3, 0)) \\
F_6 &= \text{Im}(\Phi(3, 0)) \\
F_7 &= \text{Re}(\Phi(2, 2)) \\
F_8 &= \text{Re}(\Phi(3, 1)) \\
F_9 &= \text{Im}(\Phi(3, 1)) \\
F_{10} &= \text{Re}(\Phi(4, 0)) \\
F_{11} &= \text{Im}(\Phi(4, 0))
\end{aligned} \tag{4.25}$$

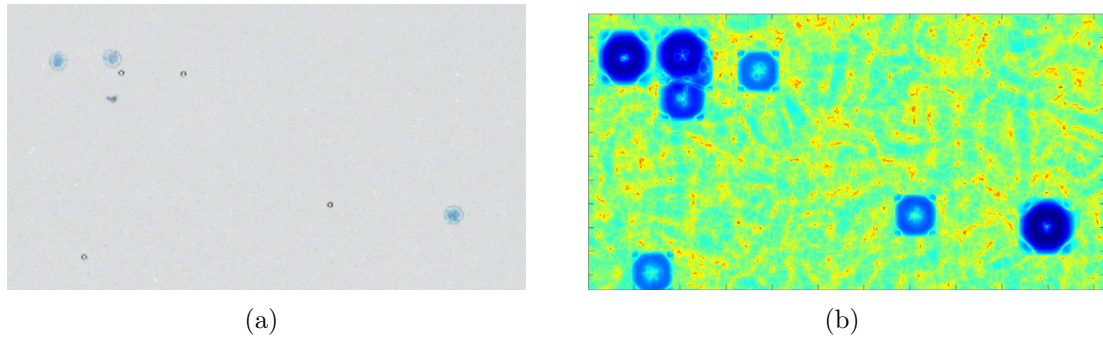


Figure 4.19: (a) Original image. (b) Hu moment invariant map

The last interesting thing is that they can be used also for segmentation. Although it is computationally expensive (~ 5 min on Intel Core i7-2860QM 2.50GHz CPU) it appears to be a very good way of looking for the objects with certain size and shape in the image. Fig.4.19a shows the original image and Fig.4.19b arbitrary chosen (5^{th} Hu) moment invariant map, where for each pixel the value is obtained by calculating the moment for patch (e.g. 100×100 px) centered at that point. As we can see in this space some of the objects are distinguished more clearly. Certainly the small objects represented as contrastive specks with white center and black border, which have similar shape - get closer to the cells. They can be easily filtered out through by

color information. The main advantage however is that of increased distance between cell and the debris or irregularly shaped stained objects, which might cause a lot of trouble in other space (e.g. color space).

4.5 Classification

For classification we chose three different approaches based on the amount of time and effort the trained engineer has to spend on designing and choosing the appropriate features. Rule-based classifier can be any one based directly on manual choice of descriptors easy to understand by humans, like color, shape and texture. Classic classifier is one which automatically adjusts the decision boundaries on those chosen features. The last one - deep learning - usually can operate on raw data and designs features by itself as part of the process.

The whole classification process is focused purely on leukocyte classification. It's quite distinctive problem by itself as detection and initial segmentation of the objects can be done very easily. But it is hard to later split them to different categories.

4.5.1 Rule-based classifier

During the development I tested few popular combinations of input features. Among the others they involved shape analysis in a way similar to one described in the beginning of the previous section. As on our LQ images WBCs have very similar sizes it was important to obtain accurate boundary delineation. We hoped that this would allow us to draw an accurate size histogram and splitting of bimodal distribution would be a main feature discriminating lymphocytes from myeloids.

Our so far best performing rule based algorithm² can be expressed in terms of simple set of rules with thresholds usually organized and adjusted for particular dataset. The ones used to present rule-based results in this thesis are shown below using PROLOG-like description in predicative logic framework:

```
WBC(X) :-
  cytoplasm_size(X) > MIN_LYMPH_SIZE_THRESH AND
  NOT blob(X) AND
  (
    cytoplasm_size(X) > MAX_LYMPH_SIZE_THRESH OR
```

²Thanks to B.Sc. Paul Hollensen for deriving this procedure.

```

    nucleus_circularity(X) > NUC_CIRC_THRESH
).

lymphocyte(X) :-
    WBC(X),
    cytoplasm_size(X) < MAX_LYMPH_SIZE_THRESH OR
    (
        cytoplasm_size(X) < MIN_MYELOID_SIZE_THRESH AND
        nucleus_size(X) / cytoplasm_size(X) > NUC_CYT_RATIO_THRESH AND
        nucleus_circularity(X) > NUC_CIRC_THRESH
    ).

myeloid(X) :-
    WBC(X) AND
    NOT lymphocyte(X).

cytoplasm_size(X) :-
    SUM(pix) FOR_ALL
    (
        contains(X, pix) AND
        mean( [ pix[red], pix[green] ] ) < CYT_RED_GREEN_THRESH AND
        best_fitting_disk(Y) AND
        contains(Y, pix)
    ).

nucleus_size(X) :-
    SUM(pix) FOR_ALL
    (
        contains(X, pix) AND
        (
            mean( [ pix[red], pix[green] ] ) < CYT_RED_GREEN_THRESH AND
            best_fitting_disk(Y) AND
            contains(Y, pix) AND
            dark_blue(pix, DARK_BLUE_THRESH) AND
            nucleus_centered_disk(Z) AND
            NOT contains(Z, pix)
        ) OR
        (
            nucleus_centered_disk(Z) AND
            contains(Z, pix) AND
            dark_blue(X, DARKER_BLUE_THRESH)
        )
    ).

blob(X) :-
    SUM(pix) FOR_ALL
    (
        contains(X, pix) AND
        boundary(Y, X) AND
        contains(Y, pix) AND
        mean( [ pix[red], pix[green] ] ) < DARKER_THAN_RBC_GRAY_THRESH
    ) >
    0.2 * SUM(pix) FOR ALL
    (
        contains(X, pix) AND
        boundary(Y, X) AND
        contains(Y, pix)
    ).

```

Besides adjustable global color thresholds at different stages this procedure is mainly based on fitting two largest possible circles covering most of the area occupied by whole cell (cytoplasm and nucleus) and the other one by nucleus itself. Only

pixels lying within those circles are later considered as important features. Their summarized interrelation constitutes a nucleus-cytoplasm ratio, which is in the end used to draw a final decision boundary. The results of such segmentation for set of cells from one patient can be seen on Fig.4.20.

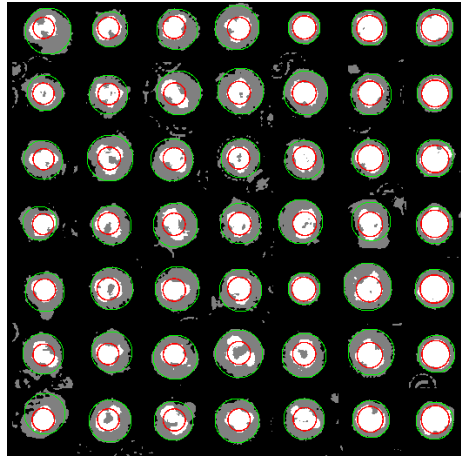


Figure 4.20: Nucleus (white - color threshold; red - fitted circle) to cytoplasm (gray - color threshold; green - fitted circle) ratio segmentation.

4.5.2 Classical analysis

Nearest neighbours classification consists of storing the training data set and then during test phase comparing each sample with its closest, in terms of specified metric, members of the stored data. Then the classification is made based on majority vote. There are many methods for storing and comparing the data. One of the most common data structure, which we believe suits our relatively low dimensional training set, is KD-tree, which divides K dimensional space in binary fashion thus splitting data set based on locations in the space and makes access time much faster. The splitting can be stopped at certain level and all the samples falling within the leaf will be searched using brute force method.

Support Vector Machine (Cortes and Vapnik, 1995) is a linear large margin classifier, which attempts to find a function separating classes by mapping the original feature vector to the space where problem is linearly separable and solving the following

optimization problem:

$$\begin{aligned} \min_{w, \xi, b} & \left\{ \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i \right\} \\ \text{s.t.} & \quad (w^T \phi(x_i) + b) y_i \geq 1 - \xi_i, \quad \xi_i \geq 0 \end{aligned} \quad (4.26)$$

where w is the parameter weight vector, $\phi(x_i)$ is some transform of input data sample x_i , y_i is its label, C is the regularization parameter describing the influence of slack variables ξ_i on the final output. The slack variables measure the distance of each wrongly classified sample from the separating hyperplane and thus provide provide a trade off between maximizing the distance from samples and actually separating the classes. The lower then the parameter C is the less overfitting we will usually get, but making it too low would cause the classifier to learn the wrong model. Using Karush-Kuhn-Tucker conditions and Lagrange multipliers we can look on a problem from different perspective and specify the dual optimization problem in terms of maximizing:

$$\begin{aligned} \tilde{L}(\lambda) &= \sum_{i=1}^n \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \phi(x_i)^T \phi(x_j) \\ \text{s.t.} & \quad 0 \leq \lambda_i \leq C \\ & \quad \sum_{i=1}^n \lambda_i x_k = 0 \quad \forall x_k \end{aligned} \quad (4.27)$$

where λ_i are Lagrange multipliers. The equation unveils the kernel function of the form: $K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j)$. This function is chosen by the user usually as a hypothesis of a form of the separating hyperplane projected onto the original feature space. In further work we make use of radial basis function (RBF) of a form $K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$, where $\gamma = \frac{1}{\sigma^2}$. It is the most general of popularly chosen kernels and one allowing for highly nonlinear solutions in original data space.

SVM is considered a large margin classifier as it attempts to fit a hyperplane for linear class separation, so that it isolates them keeping largest possible margin to all the samples from both sides. When the classes are perfectly separable $(w^T \phi(x_i) + b) y_i \geq 1$ defines the margins, so that no sample can lie closer to the hyperplane than that. The weights w then define the distance between margins as $\frac{2}{\|w\|}$ and to maximize it we need to minimize $\|w\|$ or $\frac{1}{2} \|w\|^2$ without loss of generality. To allow for outliers here the slack variable ξ_i was introduced which gives penalty to incorrectly classified samples

according to how much they exceed "their" margin. As no penalty is given to correct samples this change can also be viewed as a hinge function (compare with softplus on Fig.4.21). The problem specified this way is called soft margin classification.

4.5.3 Convolutional Neural Network

Convolutional Neural Networks emerged from the need of automatic feature extraction. In classical computer vision most of the time is usually spent on finding appropriate filters to exaggerate parts of the image, which seem particularly interesting for human expert. The popular ones include Gaussian based low pass filters, Laplace or Sobel high frequency edge detector, and more spatially localized wavelet filters, like Mexican hat or Gabor. After filtering the processed images constitute a raw input to the classification algorithm or, more often, are subjected to segmentation. The extracted features describe objects in terms of basic symbolic measures like size, shape, color or more advanced, like moment invariants. The other methods of dimensionality reduction, like PCA or ICA, are also often applied. All those methods suffer from need of laborious design and manual adaptation whenever environment changes. They are also very biased towards our believes of what constitutes a set of good features for given objects. The promise of CNNs is to avoid those manipulations and aid researchers with tool capable learning appropriate feature extraction method using example-based optimization.

Popular CNN design (LeCun et al., 1998) involves two types of layers distinguishing it from regular MLP network: convolutional and pooling. The first one is responsible for filters represented by its weights through the operation:

$$z^k = \phi\left(\sum_m W^{k,m} * x^m + b^k\right) \quad (4.28)$$

where k stands for kernel and m for channel index. z^k is the output and x^m the input feature map (i.e. channel of the image). W is a weight matrix and b is the bias. ϕ can be any type of activation function. The other variant of this layer exists in which the weights are not shared across the location. It's dubbed "local" convolutional layer and can be described as follows:

$$z^k = \phi\left(\sum_m \sum_{i,j} \sum_{r,s} \widetilde{W}_{i,j,r,s}^{k,m} x_{i+r-1,j+s-1}^m + b^k\right) \quad (4.29)$$

where i, j are indices of the input feature map and r, s over each kernel. This differs from convolution in that here we have separate W for each i, j . \tilde{A} means $flipplr(flipud(A))$ or $rot180(A)$ operation and therefore the equation represents correlation of rotated matrix with upfollowing indicies, which is just different way of describing convolution.

Usually this layer is immediately followed by pooling layer and therefore it makes sense to have ϕ to be a linear function (i.e. $\phi(x) = x$). The pooling layer is nothing more than specific way of downsampling the image in a way which preserves most of the useful information from the input feature map. It helps to compress its dimensionality and achieve higher translation invariance by extracting only the most important information from the given region. For local neighbourhood of each pixel the most popular methods include max pooling and average pooling. The first one is usually more effective as it carries on only the information the network learned to exaggerate most in given area. Pooling can be described as:

$$z^m = \phi(\beta^m d(x^m) + b^m) \quad (4.30)$$

where d is a downsampling function. The scaling factor of β is usually set up to 1. Both methods are often simplified to create one conv-pooling layer of the form:

$$z^k = \phi(d(\sum_m W^{k,m} * x^m) + b^k) \quad (4.31)$$

The decision was made to use the rectified linear units as a form of activation function:

$$\phi(x) = \max(0, x) \quad (4.32)$$

As they are non-differentiable at 0, they are often approximated with softplus units $\phi(x) = \log(1 + e^x)$ for mathematical analysis, which have an interesting property that its derivative $\frac{\partial \phi}{\partial x} = \frac{1}{1+e^{-x}}$ results in logistic function. Both functions can be compared on Fig.4.21.

Rectified linear units have certain advantages over other popularly used activation functions, like sigmoid/logistic ($\phi(x) = \frac{1}{1+e^{-x}}$) or hyperbolic tangent ($\phi(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$). They enforce sparse activation by constantly switching off subset of output units. They also achieve a type of intensity invariance in a sense that $x \leq 0$ part of a function is constant and linear part gets equally scaled for all the units with changing intensity range. Therefore, even though scaling the data by their standard deviation is the

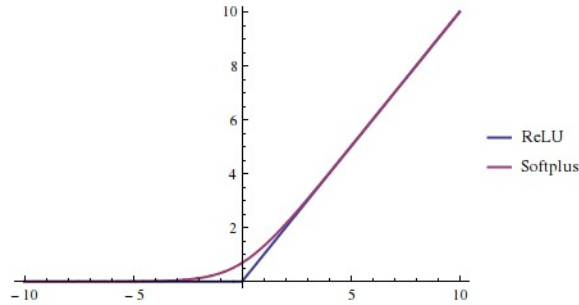


Figure 4.21: Comparison of softplus and rectified linear functions

common practice, here we only center them around mean. Their other very important feature is that they are resistant to the vanishing gradients problem (Maas et al., 2013) and thus allow for easy training of much deeper architectures. This problem occurs in saturating activation functions (like two mentioned before) when units get close to boundaries of their codomains in higher layers thus causing much smaller gradients in layers below. This results in slow convergence and higher probability of getting stuck in local minima. Finally we also chose those kind of units because they resemble more plausibly biological neurons, as mentioned by Glorot et al. (2011).

The weight optimization algorithm of choice is popular stochastic gradient descent backpropagation (Rumelhart et al., 1988, Sutskever et al., 2013). To perform it in general we apply delta rule at each layer in a form: $W^{t+1} \leftarrow W^t + \Delta W^t$ where $\Delta W^t = -\epsilon \frac{\partial E}{\partial W^t}$ and E represents the error function. This can also be written in a form $\Delta W^t = -\epsilon \sum_i \frac{\partial E_i}{\partial W^t}$, which defines batch gradient descent, where i is the batch index. Stochastic version assumes approximation of gradient based on single example and thus allows tracking of progress in more 'on-line' manner, i.e. $\Delta W^t = -\epsilon \frac{\partial E_i}{\partial W^t}$. The mini-batch variation takes advantage of both methods, by treating subsets of full batch as 'samples' for gradient descent. This allows for lower resources utilization and better convergence at the same time.

We view the training of our network as maximum likelihood estimation on stochastic model defined by joint probability distribution over given training dataset:

$$\mathcal{L}(W; X, T) = \prod_i p(Y = t^{(i)} | X = x^{(i)}; W) \quad (4.33)$$

where \mathcal{L} is the likelihood function, T are target labels and X is the input data. Our objective is to find $\hat{W}^{ML} = \underset{W}{\operatorname{argmax}} \mathcal{L}(W; X, T)$. For computational reasons it is a

common practice to use the negative log likelihood, normalized by number of samples, as a loss function to minimize:

$$E(X, T, W) = -\frac{1}{n} \sum_{i=1}^n \log p(Y = t^{(i)} | X = x^{(i)}; W) \quad (4.34)$$

and as $p(Y = t^{(i)} | X = x^{(i)}; W) = \prod_i y^{(i)t^{(i)}}$ the loss function for one sample becomes:

$$E(X, T, W) = -\sum_{i=1}^n t^{(i)} \log y^{(i)} \quad (4.35)$$

and is further accumulated and normalized across number of samples in a batch.

In all our models we use softmax function as our top layer activation function, i.e.:

$$y_j = \phi(W_j X + b) = \frac{e^{W_j X + b_j}}{\sum_k e^{W_k X + b_k}} \quad (4.36)$$

where j is the class index. The choice of this function is motivated by its capabilities to represent categorical probability distribution through normalization.

The gradient derivation of the specified functions used later in backpropagation algorithm is detailed in Appendix D. In all of the following implementations of neural networks in this paper we also use weight decay, parametrized by λ , and momentum (e.g. Fischer and Igel (2012)), parametrized by η so our cost function becomes:

$$\hat{E} = E - \lambda W^2 + \eta \left(\frac{\partial \hat{E}}{\partial W} \right)^2 \quad (4.37)$$

4.5.4 Pre-training and filter initialization

As the convolutional neural network highest accuracy on WBCs test set, in this section we define a potential way of extending the algorithm for speeding up the training process and learning more general filters. I admit though that this help most in the case of larger number of unlabeled data which can be additionally harvested. This is not the case of our dataset and therefore this part should be treated as additional practice helping to improve the results for the reader who wishes to follow our approach.

The most basic filter initialization besides random is to design them manually. I tried this approach and chose a set of those which seemed most sensible to me. I chose different sizes and orientations of Gabor functions, because they are biologically

plausible and this is the most common result in literature of learning neural networks on natural images (Krizhevsky et al., 2012). And then I also chose different sizes of DoG and radial basis functions, because they resemble cells in most of decomposition approaches (like PCA or ICA) or filters learned by CNN in our previous approaches. The set of filters can be seen on Fig. 4.22.

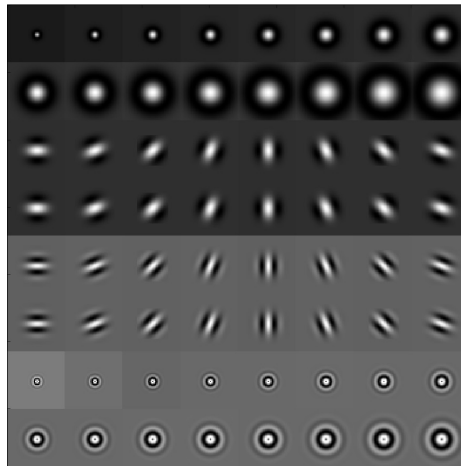


Figure 4.22: Hand crafted filters

The second option is unsupervised pre-training, which should in general help the network to reduce dimensionality of the data more robustly. To get more details on why this is truth, please follow Erhan et al. (2010). Restricted Boltzmann Machine (Fischer and Igel, 2012), beside autoencoders, is the most common and modern method for extracting important common features of objects within set of unlabeled images. It results in a set of components, which describe those objects in similar fashion PCA or ICA does. Recent development of convolutional RBM (Lee et al., 2009) allow us to apply similar techniques to pre-train the filters for our convolutional neural network. We decided to apply that technique only to the first layer and use longer fine tuning after. Description of basics behind RBM is out of the scope of this paper. Please refer to Fischer and Igel (2012) for more information.

The general energy equation of convolutional RBM is defined as follows (Lee et al., 2009):

$$\begin{aligned}
E(v, h) &= - \sum_k \sum_{ij} (h_{ij}^k (\widetilde{W} * v)_{ij} + b_k h_{ij}^k) - c \sum_{ij} v_{ij} \\
s.t. \sum_{ij \in B_\alpha} h_{ij}^k &\leq 1, \forall k, \alpha
\end{aligned} \tag{4.38}$$

Inputs to visible and hidden layers are defined in similar way as in first layer of pure CNN. And thus the same rule applies to the mode of convolution operation.

$$I(h_{ij}^k) = \sum_m v^m * \widetilde{W}^{m,k} \tag{4.39}$$

$$I(v_{ij}^m) = \sum_k h^k * W^{m,k} \tag{4.40}$$

Bernoulli hidden layer sampling uses probabilistic max pooling on top, which constraints number of possible units which are on within certain window B_α to 1 (see the constraint within energy function). Therefore the following conditional distribution of Gibbs sampling can be effectively viewed as simple multinomial distribution enumerating all the possible configurations of nodes within a block (one per each node) and $p(p_\alpha = 0|v)$ defining situation when all of them are off (p_α is here a state of a pooling layer unit, rather than its probability).

$$\begin{aligned}
p(h_{ij}^k = 1|v) &= \frac{p(v, h_{ij \in B_\alpha}^k = 1, h_{rs \in B_\alpha \setminus \{(ij)\}}^k = 0)}{p(v)} = \\
&= \frac{\sum_{h_{rs \in B_\alpha \setminus \{(ij)\}}^k} e^{-E(v, h_{rs \in B_\alpha \setminus \{(ij)\}}^k)} e^{-E(v, h_{ij \in B_\alpha}^k = 1)}}{\sum_{g_{rs \in B_\alpha}} e^{-E(v, g_{rs \in B_\alpha})}} = \\
&= \frac{e^{I(h_{ij \in B_\alpha}^k)} e^{-E(v, h_{rs \in B_\alpha \setminus \{(ij)\}}^k)}}{\sum_{g_{ab \in B_\alpha}} e^{-E(v, g_{ab} = 1, g_{ab \in B_\alpha \setminus \{(ij)\}})} + e^{-E(v, g_{ab \in B_\alpha} = 0)}} = \\
&= \frac{e^{I(h_{ij \in B_\alpha}^k)} e^{b^v \sum_{ij} v_{ij}}}{\sum_{g_{ab \in B_\alpha}} e^{I(h_{ab \in B_\alpha}^k)} e^{b^v \sum_{ij} v_{ij}} + e^{b^v \sum_{ij} v_{ij}}} = \\
&= \frac{e^{I(h_{ij}^k)}}{1 + \sum_{ab \in B_\alpha} e^{I(h_{ab}^k)}}
\end{aligned}$$

$$p(p_\alpha = 0|v) = \frac{1}{1 + \sum_{ab \in B_\alpha} e^{I(h_{ab}^k)}} \tag{4.41}$$

For sampling visible units we need to use Normal distribution (see Krizhevsky and Hinton (2009) for more on that). To make the training easier we normalize the input data and then use constant standard deviation $\sigma = 1$ for sampling.

$$p(v_{ij}^m|h) = N(I(v_{ij}^m); 1) \quad (4.42)$$

We use Contrastive Divergence for training. Positive and negative phases for CD-2 are calculated by:

$$\langle vh \rangle \equiv E[p(h|v) * v] \quad (4.43)$$

The last thing is sparsity constraint which helps a lot with training distinctive filters - each representing different feature - and thus making the model more robust. Here we use the method based on Kullback-Leibler divergence in which we specify target sparsity, as the mean percentage of hidden nodes being on at a given time (usually $\rho = 0.05$) and add the constraint to the cost function.

$$\sum_j D_{KL}(\rho||\hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) \log \left(\frac{1 - \rho}{1 - \hat{\rho}_j} \right) \quad (4.44)$$

The gradient for weights update can then be derived as $\frac{\partial D_{KL}(\rho||\hat{\rho}_j)}{\partial \hat{\rho}_j} = -\frac{\rho}{\hat{\rho}_j} + \frac{(1-\rho)}{(1-\hat{\rho}_j)}$.

4.5.5 Dropout

Dropout (Srivastava et al., 2014) is a simple yet powerful technique which consist of dropping random connections during learning stage so that different ones get updated in each epoch. The main parameter here is the target p specifying probability of weights having non-zero value. It prevents overfitting by providing more general filters. It can be thought of as an ensemble of networks, in which different network gets trained each time we prune connections and we get an average network in the output. For training such network different hyperparameters like learning rate and weights range should be scaled appropriately.

Chapter 5

Experiments and Results

5.1 Image Preparation

All of the pictures were taken from full blood samples of 11 patients and 4 auxiliary samples. The latter constitute of purified blood where all but one cell types were lysed (i.e. broken down and removed). The remaining types of white blood cells in each of those four samples were: lymphocytes, monocytes, myeloids and neutrophils. They were created for training and validation purposes. As we didn't obtain sufficient quality to reliably distinguish monocytes and neutrophils, lymphocyte images were used to fill lymphocyte class and all the other ones to fill the myeloid (i.e. non-lymphocyte) class.

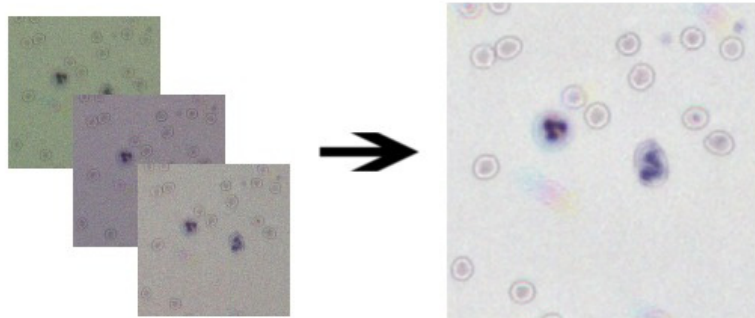


Figure 5.1: Results of super-resolution. The 21 RGB images of low quality (left) are transformed into image of 5 times larger resolution (right) conveying information from all of them. De-blurring deconvolution, median filtering and pseudo flat field correction have been applied as part of the process.

Five 200 Mpix images were collected of each patient's blood. The initial low resolution (LR) images were corrected with pseudo flat field and leveled to common mean intensity (please see section 4.2.1 for method description). The pseudo flat field was created by subsampling the image with step 4 and then filtering it with median filter of size 8x8. The edge effect of filtering was removed by creating a mirror of the edge parts for the time of processing. Often the average level intensity of each channel

is chosen to be the maximum across the channels. Here however we wanted to have the same level for each incoming image. As there was no higher reference point it was set arbitrarily to $I_{FFC} = \frac{I_{org}}{I_{FF}} * 280 - 56$, where 56 was common lower bound of intensity histogram and 280 was chosen higher than intensity spectrum to provide space for larger range. The high resolution was obtained by combining 21 LR noisy images using popular super resolution algorithm (Farsiu et al., 2004) (see Fig.5.1) described in more details in section 4.2.4.

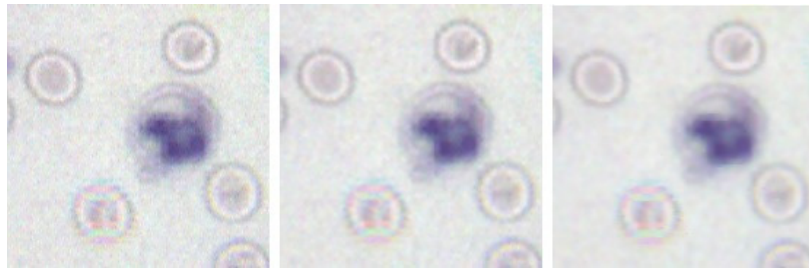


Figure 5.2: Comparison of filtering of the original image (left) with median filter (center) and anisotropic diffusion (right)

After resolution enhancement we perform additional small window 2x2 median filtering to remove noise. Here anisotropic diffusion could be used instead to reduce noise to even higher extend (Fig.5.2), but has a trade off of computation time. Full super-resolution with filtering of HQ images took about 9 minutes on Corei7-2860QM 2.50GHz CPU when using median filtering and about 17 minutes when using anisotropic diffusion. No quantified results were obtained to measure the noisiness of the images, however the background appeared smoother and edges well preserved in case of anisotropic diffusion. The improvement appeared useful within acceptable time for LQ data, but this wasn't the case for HQ. Red blood cells were successfully calculated even in LQ, for white blood cells the classification algorithms were used, which are robust for noise level appearing in HQ, and platelets were represented sometimes by so small speckles, that they couldn't be reliably preserved during the filtering.

The resulting super-resolution, denoised images allow for visual isolation of our three main objects of interest - erythrocytes, thrombocytes and leukocytes. They are presented on Fig.5.3 in the same scale.

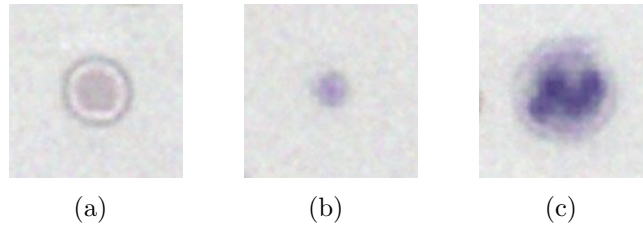


Figure 5.3: Visuals of the main objects of interest cropped out from images after the last step of pre-processing: (a) red blood cell / erythrocyte (b) platelet / thrombocyte (c) white blood cell / leukocyte

5.2 Cell analysis and final results

The most important from the clinical point of view are correlations with state of the art readings. As mentioned before, Coulter Counter was used to measure count factors for each patients. The following sections describes in details experiments conducted for measuring features of each type of blood cell. The counts obtained with computer vision system are plotted against the Coulter Counter values and linear regression is used to obtain the model of relation. Those correlation plots for each measurement achieved with best performing methods are presented on figures below. Subsequently the coefficients of determination (R^2) are computed for each plot as:

$$R^2 = 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \mu_y)^2} \quad (5.1)$$

where y_i are observed relations, μ_y is their mean and f_i are corresponding values of fitted model.

5.2.1 Erythrocyte segmentation and analysis

As mentioned in section 4.3 the red blood cell analysis was based almost purely on segmentation and basic feature extraction. The final pipeline for erythrocytes and thrombocytes was implemented in MATLAB utilizing available image processing and computer vision toolbox functions - mainly Hough transform, filtering, Otsu's method and all the plotting and visualization tools. The images were segmented using only the violet channel (Fig.5.4). The main reason for not using other channels was that the cells were moving during the acquisition process (~ 2 min) and so the objects often didn't overlay exactly across the channels. The smaller the objects were the

more affected they were by the flow. After performing flat field correction in violet channel, the initial segmentation was done by Otsu's method (Fig.5.4b). Next the system performs morphological closing followed by hole filling and obtains Fig.5.4c as a result.

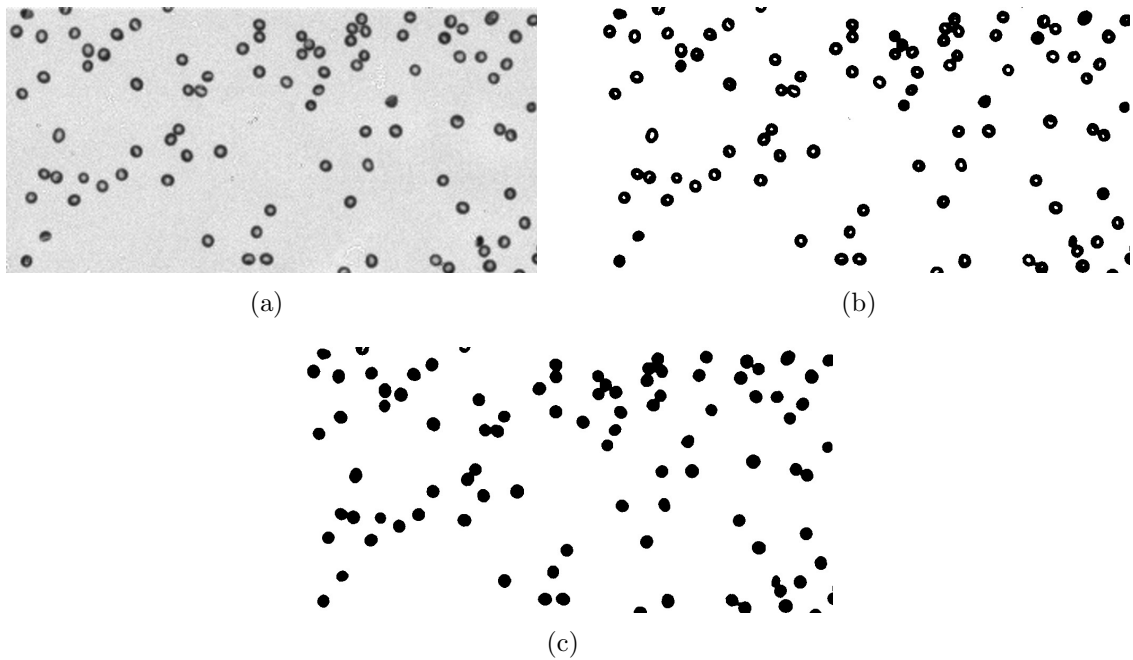


Figure 5.4: (a) Violet channel of full blood image enhanced and prepared for RBC segmentation. (b) Violet image threshold mask using Otsu's method. (c) The same mask after morphological closure and hole filling.

The connected components were later divided by size into singlets and clusters. The results of this operation are shown on Fig.5.5. The histograms showing cell sizes were drawn and the threshold was chosen as an average minimal point between Gaussian representing singlet cells and one of doublets (Fig.5.6a) .

The group of clusters was then subjected to circular Hough transform. As the detected edges were often noisy and were not giving good overall results (please see section 4.3), the other approach was taken instead. The blobs of Fig.5.5 were left without edge tracking and circles of only sizes close to RBCs were fitted. Each cluster was considered separately - to avoid unnecessary overlap. An example of a patch with circles detected is shown on Fig.5.7a. This detection is far from perfect. First of all the cells have different sizes and the segmentation is usually not exactly circular. To avoid location errors smaller circle size was chosen. Also the problem of overlapping

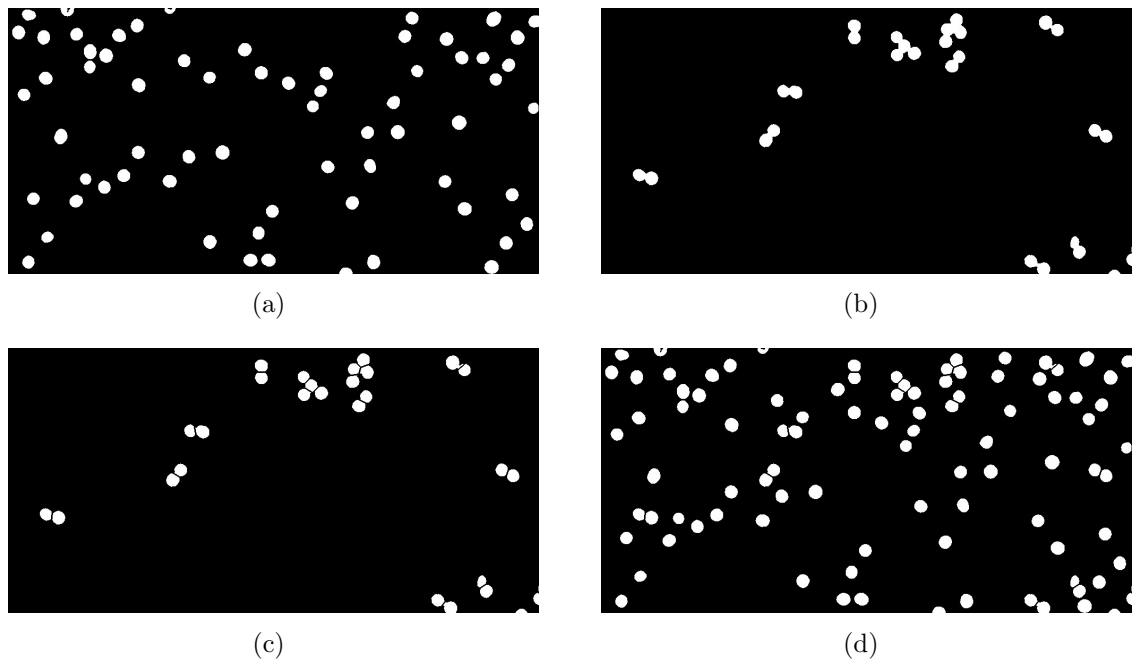


Figure 5.5: (a) Singlet cells (b) Clusters of cells (c) Results of cluster separation (d) Complete result of RBC segmentation. Separated clusters and singlet cells joined in one image mask.

circles often occurs within patch. When the edge detection is skipped it is usually caused by many models being fitted to the same cell. To fix that first the threshold is set on certainty (number of positive pixels under the circle). Second the distances are calculated between all of the circle centers and the neighbours. Then only one circle is left within each neighbourhood and the rest is removed. As the model which stays is chosen randomly, it often occurs that the result circles are not visually positioned in the center of the wanted segment. The whole effort here is then put not on describing the cells by circles - as this is often not accurate enough and causes too much error in mean cell size calculations - but on marking each cell with disjoint pixel (e.g. circle center) to facilitate the subsequent algorithms to divide those cells.

The found objects centers are then segregated and Voronoi diagram is applied for cell separation. The result of circle centroids generating the diagram for cells can be seen on Figs.5.7b and 5.7c. The backprojected results for clusters can be seen on Fig.5.5c. After joining with initially obtained singlet cells the final result is visible on Fig.5.5d and further overlaid on initial violet image on Fig.5.8.

The next step is a measurement of the area of each segment. Two thresholds

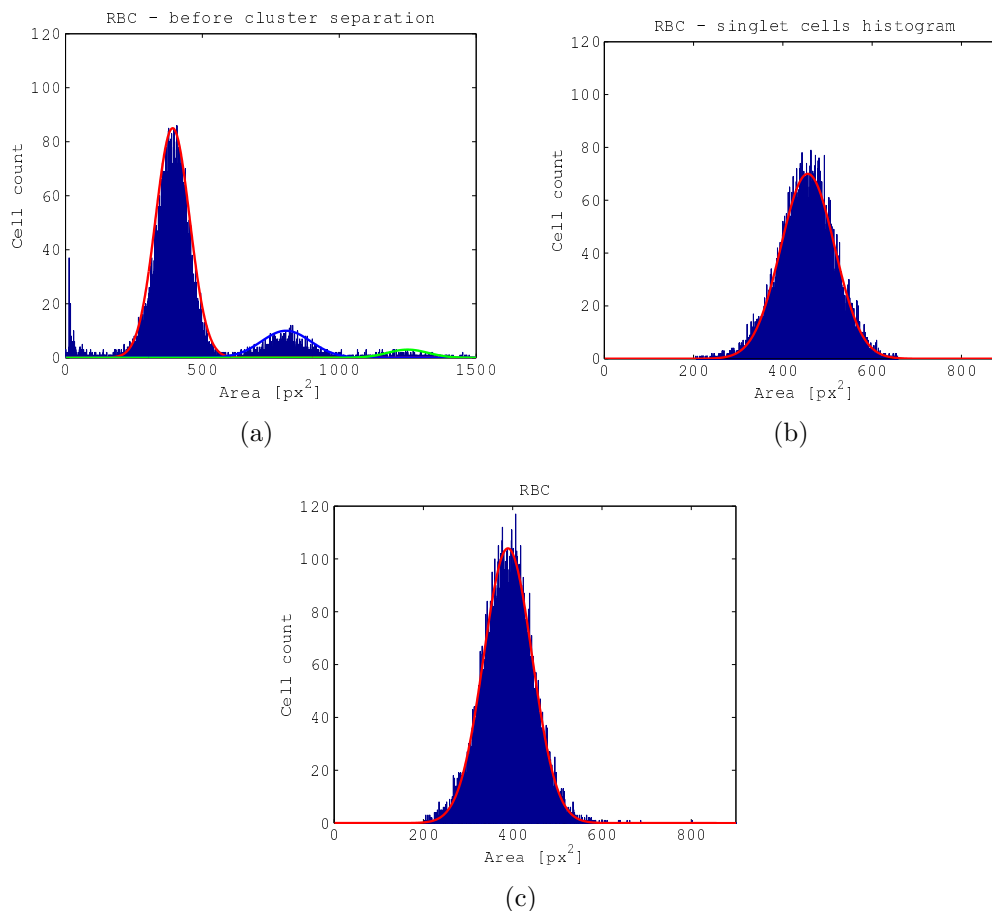


Figure 5.6: Histogram of object sizes before declustering (a), singlet (non-clustered) cells (b) and all red blood cells after declustering (c).

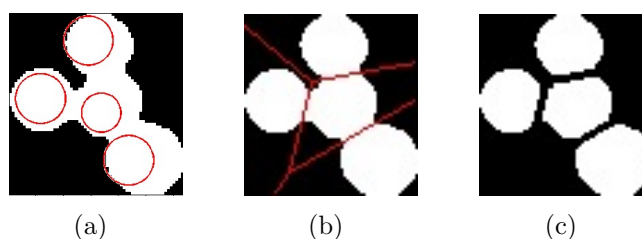


Figure 5.7: (a) Hough transform result (b) Voronoi diagram overlaid with cluster (c) Separated RBC cluster

were set to exclude objects too small and too large to be cells. As the divided cells from clusters are either squeezed or overlapping with other cells, their measures will always be less accurate than those of singlet cells. Taking that into consideration two measures were taken - one for the singlet cells (Fig.5.6a) and one for both singlet and clustered cells (Fig.5.6b). To exploit obtained histogram I fitted Gaussian function by

the same means we perform MLE on normal probabilistic distributions. From that I obtained mean cell size and standard deviation of possible sizes. Just by looking at them it can be seen how much error is introduced by this uneven measures. For further applications the decision was made to calculate the size and other features parameters based on the singlet cells approach and total counts based on all.

Finally CBC measures are obtained for RBC by combination of obtained values and pre-defined constant values based on experiments done by the company. For further description of those please refer to section 2.1.1.

$$\begin{aligned}
 dilution_factor &= 0.25 \\
 chamber_height &= 1.75 \\
 superres &= 3 \\
 area &= \frac{area_covered * (1.1)^2}{superres^2} \\
 RBC\# &= count \\
 RBC_Concentration &= \frac{count * dilution_factor}{area * chamber_height} \\
 MCV &= singlet_cell_mean \\
 RDW &= \frac{singlet_cell_stdev}{singlet_cell_mean} \\
 HCT &= RBC_Concentration * MCV
 \end{aligned} \tag{5.2}$$

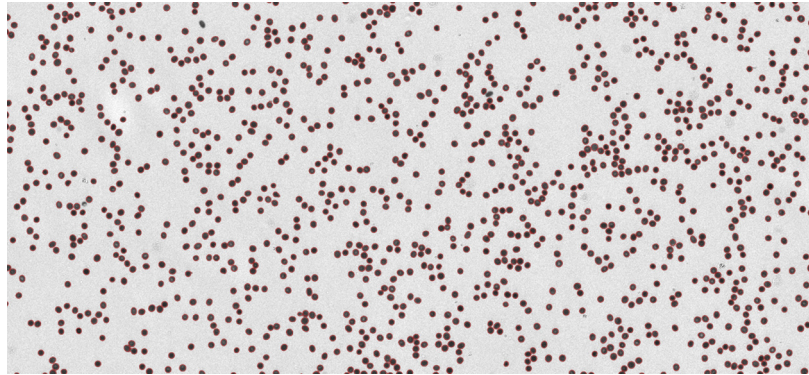


Figure 5.8: RBC segmentation. Violet channel of the image with overlaid borders (red) of the segmented red blood cells.

For each of the RBC measures, median value was used across the images for each

of 11 patients to reduce influence of outliers. Without ranges of standard values and proper documentation, the obtained numbers do not convey enough information to health worker, so they need to be further related to the standard ones (section 2.1.1). This is accomplished by demonstrating the factors of linear dependence between those two types of data. They are obtained by linear regression and the assessment of quality of that fit is done by calculating the coefficient of determination. R^2 values for erythrocytes are presented on Fig.5.9 and further summarized in Tab.5.1. Section 5.3 provides further discussion and potential solution to some of the occurring problems.

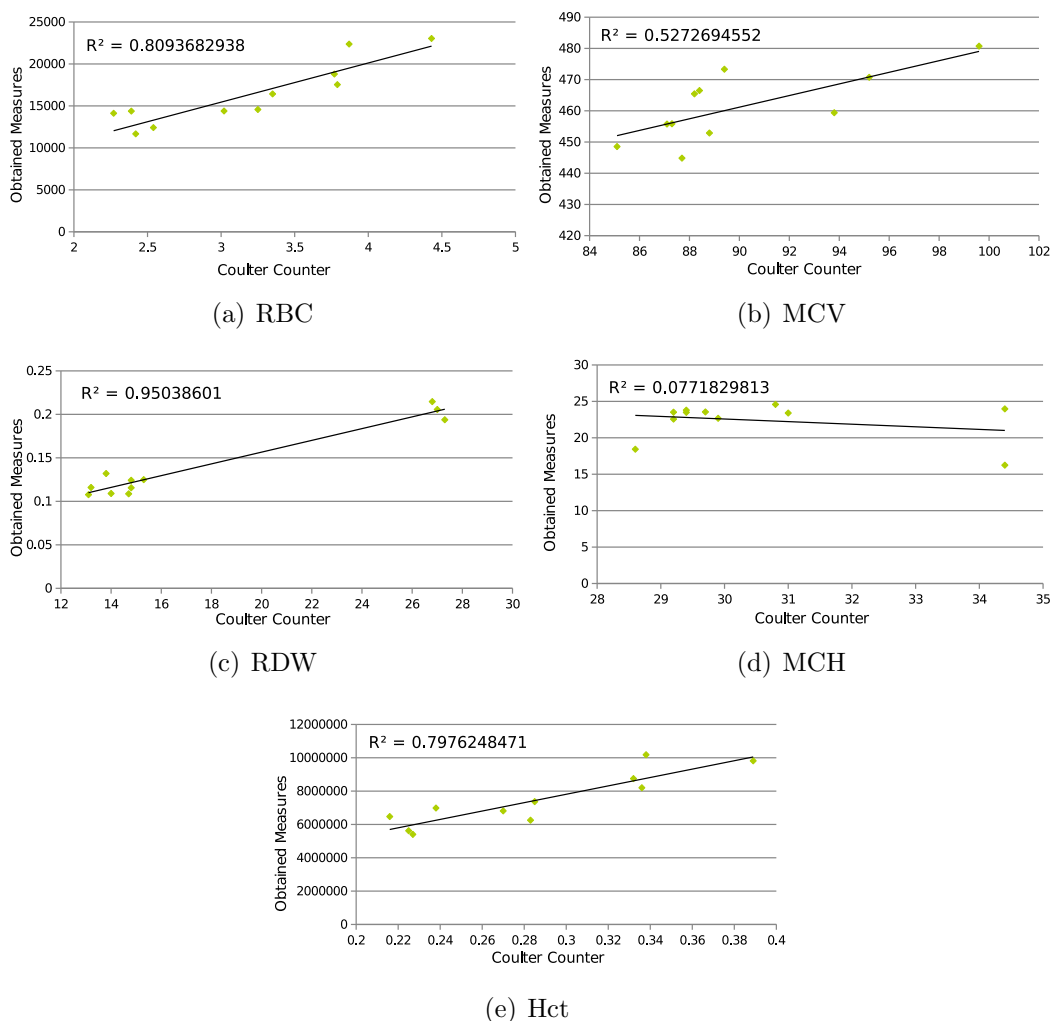


Figure 5.9: Squared Pearson correlation of total (a) RBC (b) MCV (c) RDW (d) MCH (e) Hct count with Coulter Counter readings for given patients

	RBC	MCV	RDW	MCH	HCT
R^2	0.81	0.53	0.95	0.20	0.80

Table 5.1: RBC R^2 summary table. Juxtaposition of the R^2 values of the correlation plots on Fig.5.9.

5.2.2 Thrombocyte segmentation and analysis

A slightly different approach was taken for the case of platelets. The main difference was utilization of probability map based on weighted combination of images filtered or segmented in few popular ways.

First few examples of platelets, with morphology as distinctive as possible, was picked manually from random cropped images. As this step is mostly to exploit color based features, hue channel was calculated for each example and pixel values clustered with k-means. This constituted 5 means (besides background), the values were saved and used to calculate five probability maps for each image based on Euclidean distance from them to each pixel. The additional layer was added based on simple thresholding of violet channel - mainly to exclude examples of unusually looking red blood cells.

The next step was based on Laplacian of Gaussian filtering and then calculation of Hough transform for each of RGB channels - mainly to find all the small circular objects of sizes appropriate for platelets. Before incorporating this layer into the process, found components were smoothed with dilation and Gaussian filtering. Then the next layer was calculated with the same goal of finding small speckles - this time equipped with difference of Gaussians filters. The aim here was similar to basic template matching, where filter, or template, is cross correlated with the image to give most probable positions of objects with similar shape.

In the end the layers were multiplied together (or added in logarithmic space to avoid losing accuracy associated with small floating point numbers). The acquired probability map was normalized, thresholded and subjected to morphological filtering to remove noise and smooth the boundaries. The results can be seen on Fig.5.10.

Here again median count over the pictures for each patient was used and the final Plt parameter results can be seen on the correlation plot presented on Fig.5.11. The R^2 value reached 0.28. The discussion on those results is further presented in section 5.3.

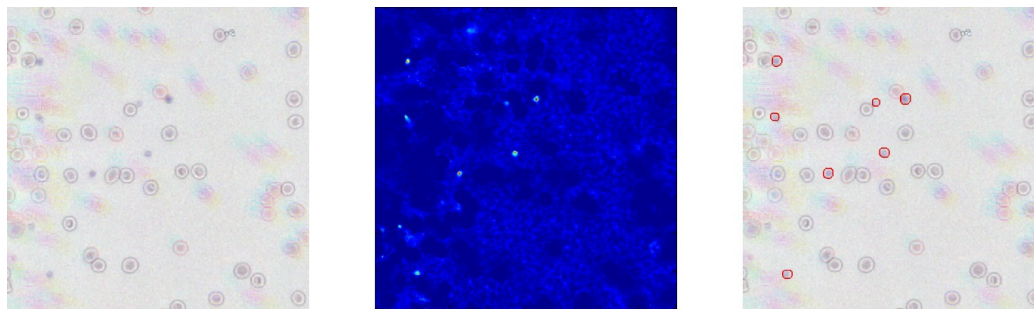


Figure 5.10: Platelet segmentation. Original image (left), platelet probability map (middle), segmented platelets (right).

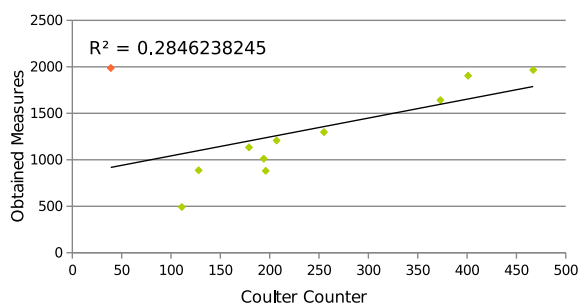


Figure 5.11: Squared Pearson correlation of total platelet count with Coulter Counter readings for given patients

5.2.3 Leukocyte classification

Previously defined classification methods are extensively tested on task of separating three leukocyte classes based on set of cropped 121x121 4-channels color image patches. Most of the original images were partially labeled with rough (non-centered) location and type of the cell. This dataset includes samples of two main leukocyte types - lymphocytes and myeloids, and one additional class of hard case non-leukocyte objects with features similar to both. Example images from the set can be seen on Fig.5.12 and Fig.5.13.

Simple threshold of 10% of lowest values in histogram of b channel from $L^*a^*b^*$ space followed by morphological opening was usually enough to produce good candidates for CNN, but for rule-based approach we wanted to initially exclude as much of anomalies as possible. It wouldn't be fair to compare the classifiers after different segmentation procedures and therefore the following approach was taken. First we perform initial thresholding based on the difference between green and blue channels. After that we remove particles close to each other leaving each time only one from close

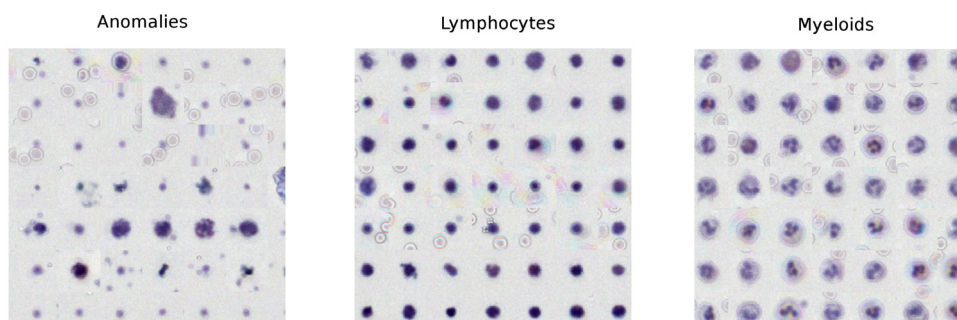
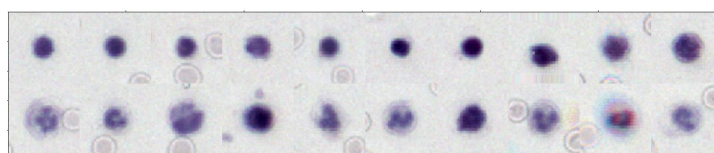


Figure 5.12: Representatives of examples of three classes in the dataset - anomalies, lymphocytes and myeloids



(a)



(b)

Figure 5.13: Random WBCs examples displaying their (a) RGB and (b) fourth 405nm violet channel

neighbourhood. And then we finish by applying size threshold. This produced marker map for the image. This map is then matched with ground truth of human expert (see Fig. 5.14). This is done by first extracting each patch by drawing 121x121 pixel square around centroid of each segmented region. Then the attempt for correction of centroid location is made by applying Otsu's method in $L^*a^*b^*$ space. The dilation is applied to it and if the largest connected component doesn't occupy certain percentage of the patch it is chosen as better candidate, the center point location is corrected and new patch extracted from original image. The samples on the edge of the image are

padding with gray values of mean intensity matching background, so they fit mentioned square shape. After extracting each sample, the simple procedure is applied to check which ground truth label location falls within the box. White blood cells are separated enough on the image samples, so that getting two labels in one patch practically never occurs.

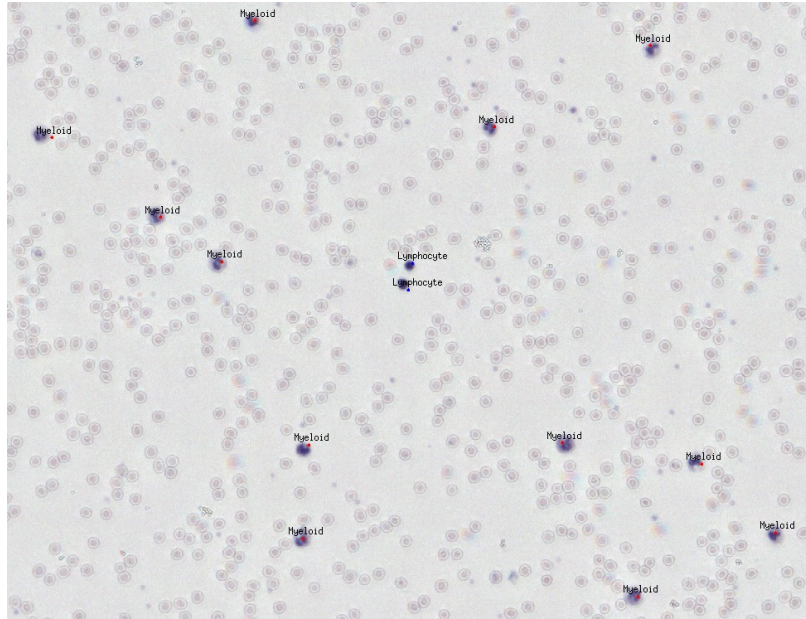


Figure 5.14: Ground truth class and localization markers manually positioned by expert.

Obtained in this way dataset consists of 8106 patches. 2440 are unlabeled, 628 are anomalies, 871 lymphocytes and 4167 myeloids. We further produce augmented samples, mainly to balance it as the number of samples in each class is very uneven. The transformations include 4 90° rotations, 2 mirror operations, 4 shifting operations and as many as needed scaling of eigenvectors in RGB space, effectively changing overall illumination, as described by Krizhevsky et al. (2012). The example result can be seen on Fig. 5.15.

At the feature extraction stage we only extract 7 Hu moments and 11 Flusser moments per channel for each sample. This constitutes input for training SVM and kNN. Even though the TRS moments produce duplicates in augmented dataset, we obtained better results when it was balanced in this crude way. Besides, used moments are not contrast invariant and thus introduce small variety. CNN takes raw images as

input, but rule-based system operates directly on segmented regions.

As the rule-based system was hand tuned for the whole data set, I don't report its accuracy. The correlation with Coulter Counter readings are used instead for comparison with other methods. For all the other classifiers we perform patient based cross validation (CV). The training set comprises of partially marked images of blood from 9 patients. The non-labeled examples include misclassified or missed cells in those images and all the examples from patients 10 and 11. In each classification attempt we leave out patches of the patient and use them for testing. This results in 30-fold CV. Only after the split the training set is subjected to augmentation. The results are then used for Coulter Counter correlation and averaged for accuracy comparison.

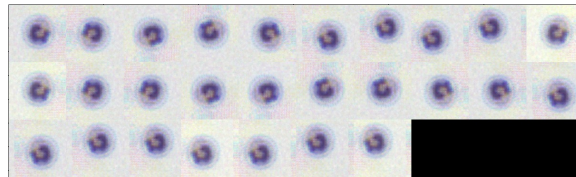


Figure 5.15: Data Augmentation by rotation, flipping, shifting and intensity modification by PCA in RGB space.

Classification with kNN was based on KD tree search with leaf size 30. The Euclidean metric was used for distance comparison. Implementation provided by Pedregosa et al. (2011) was used for testing.

For SVM I chose a kernel to be Radial Basis Function (RBF). The simpler kernels, like linear and polynomial gave worse results consistently in variety of tests during the development process. I first split the whole dataset using 10% random samples as a test set and then optimize hyperparameters C of cost function and γ of RBF kernel. For that I used grid search provided by Chang and Lin (2011) modified in a way that it doesn't perform cross validation (because of time constraints). The optimization is done in the space of $C = [2^{-5}, 2^{15}]$ and $\gamma = [2^3, 2^{-15}]$. To make the search more efficient the coarse grid is first used to find a rough region and the finer one for more accurate tuning. The results are presented on Fig. 5.16 with best accuracy achieved when using $C = 0.125$ and $\gamma = 2$. The accuracy doesn't tell us the real performance here though and thus we later perform patient based cross validation.

The architecture of convolutional neural network used can be seen on Fig. 5.17.

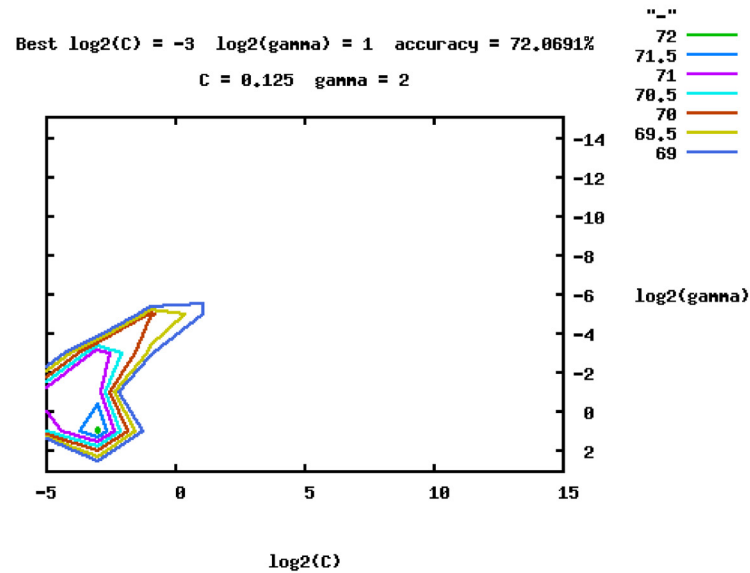


Figure 5.16: SVM grid search in space of regularization parameter C and RBF kernel parameter γ .

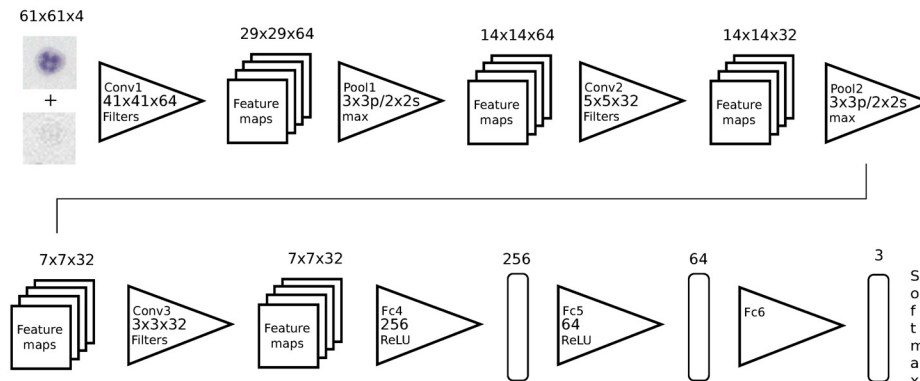


Figure 5.17: Convolutional Neural Network architecture used for WBC recognition

Before feeding it the images, those images were downsampled with step 2. Even though the network tends to learn filters more pleasant to human eye when no padding is added, we always got slightly better accuracy when it was and thus we pad input with zeroes 4 pixels in each direction on the first layer and 2 pixels in each direction on the second one. Pooling was based on max operation with 3x3 window and stride of 2x2. The fully connected layers use ReLU and output is softmax. Momentum was chosen to be $\eta = 0.9$ and weight decay $\lambda_1 = 0.004$ for each convolutional layer and $\lambda_2 = 0.03$ for the higher layers. The cost function was negative log likelihood and the network was trained with stochastic gradient descent algorithm using minibatches

of size 128. Pylearn2 framework (Goodfellow et al., 2013) was used to support the implementation and all the time consuming operations (esp. nD convolution) were performed on GPU. The learning rate was chosen to be 0.0001 and the generalization (test set) learning curves for lymphocyte and myeloid classes are presented on Fig.5.18. Learning curve for anomalies is not representative as it contains too few amount of samples in the test set. Here and in the following result reports we arbitrary chose one of the cross validation classifications as a representative (patient 2, image 4). Fig. 5.18c represents negative log likelihood value (without constraints) during the training averaged over classes. Below on Fig.5.19 we can see 64 first layer filters learned during the training.

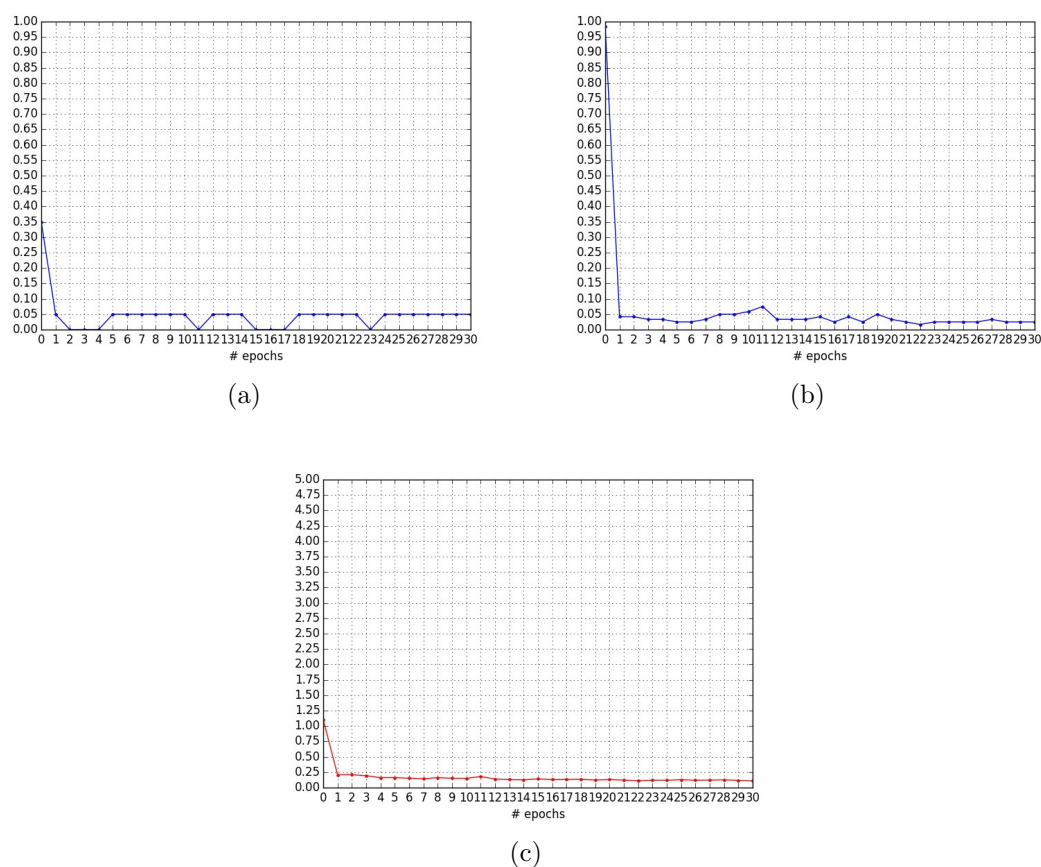


Figure 5.18: (a) Lymphocyte generalization curve. (b) Myeloids generalization curve. (c) Mean negative log likelihood

For dropout setup we removed padding and weight decay. The probability of nodes being on was set to 0.8 for each layer. Although the resulting filters look more

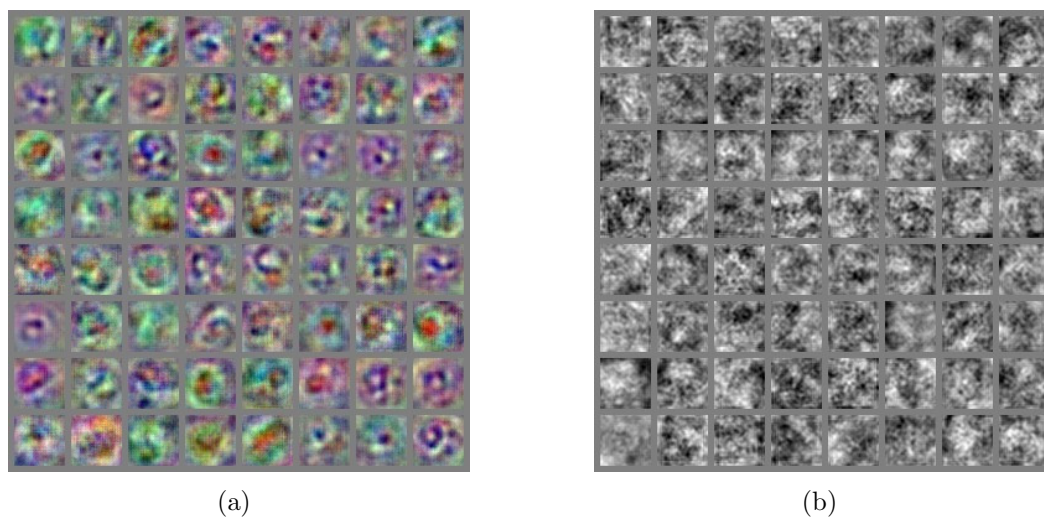


Figure 5.19: Filters trained with regular CNN (a) RGB (b) Violet.

convincing for human (Fig.5.20), the training takes longer time and accuracy achieved by us was always the same or slightly worse than when training without dropout. The cause for that might be less robustness to noise and too general filters, in case of which they should be trained for even longer time. I report these results though as I believe our problem is not complex enough to take advantage of this technique, but readers may take advantage during their system development. The regular network solves the problem with acceptable accuracy and the wrongly classified examples are most of the time incorrectly labeled by human expert.

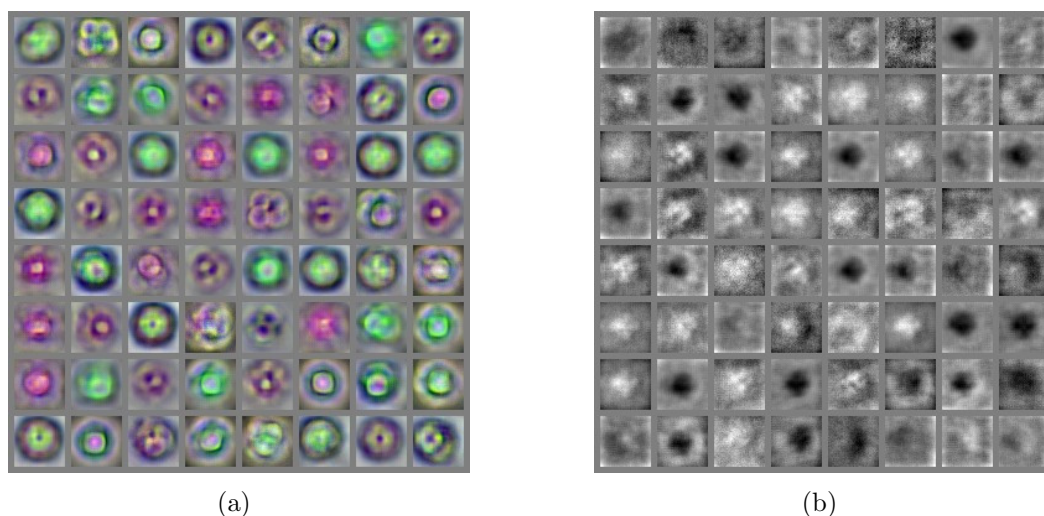


Figure 5.20: Filters trained by CNN with dropout (a) RGB (b) Violet.

Pre-initialization of filters in manual way had no effect on the final classification results. We present filters of convolutional RBM on Fig.5.21. One can see the effect of sparsity constraint, as the filters tend to be more diverse. They also seem to focus more on details which might be of larger importance when classifying cells other than blood, which depend more on morphology. Utilization of those filters to improve the results is part of the future work.

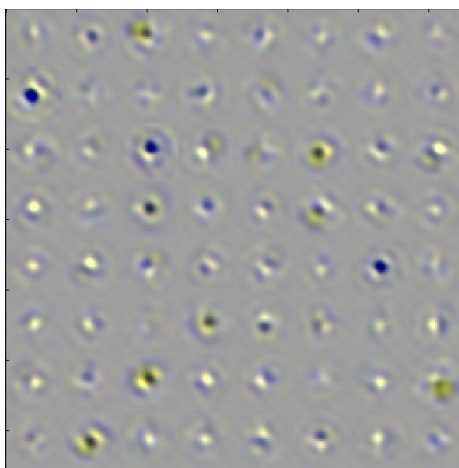


Figure 5.21: RGB filters of convolutional RBM

We further asked what is the optimal number of convolutional layers needed for correct classification and thus what is the hierarchical complexity of our data. In each test we adjusted the number of filters and weights on fully connected layers to nearly match 500000. The results can be seen on Fig. 5.22, where it appears that two convolutional-maxpooling layers is enough. The fully connected layers stayed within the network and we didn't make any attempt to vary their number.

Table 5.2 presents an example confusion matrix (again for patient 2), showing the influence of falsely classified samples on overall accuracy and also the actual unbalance of the classes.

	Anomaly	Lymphocyte	Myeloid
<i>Anomaly</i>	8	0	1
<i>Lymphocyte</i>	0	23	1
<i>Myeloid</i>	1	2	119

Table 5.2: Confusion matrix - predictions (top) vs actual class (left)

The final results of image based cross validation classification are presented on

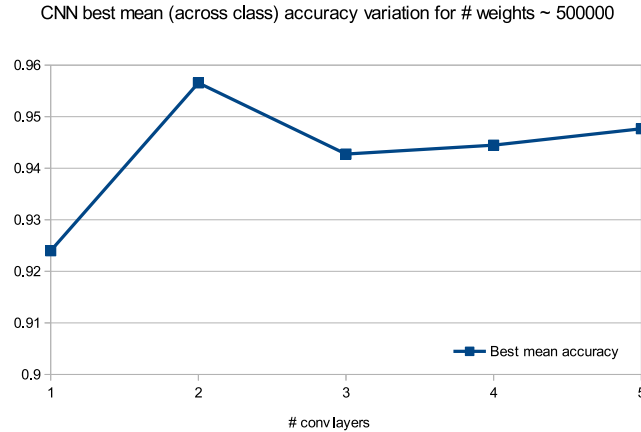


Figure 5.22: Best accuracy achieved on the network with randomly chosen training and testing subsets with varied number of convolutional layers and same number of weights

Fig. 5.23. The mean recall for each class was calculated rather than average accuracy over all classes, as the test dataset was very unbalanced. To show if the results indeed significantly differ from each other, also the variances were shown as error bars. Both measures are given by equations:

$$\mu_{recall}^{(c)} = \frac{\sum_{i=1}^N TP^{(c)}(i)}{\sum_{i=1}^N (TP^{(c)}(i) + FN^{(c)}(i))} \quad (5.3)$$

$$(\sigma_{recall}^{(c)})^2 = \sum_{i=1}^N \left[\left(\frac{TP^{(c)}(i)}{TP^{(c)}(i) + FN^{(c)}(i)} - \mu_{recall}^{(c)} \right)^2 \frac{(TP^{(c)}(i) + FN^{(c)}(i))}{\sum_{i=1}^N (TP^{(c)}(i) + FN^{(c)}(i))} \right]$$

where c is the class, $TP^{(c)}$ are true positives and $FN^{(c)}$ of a given class. The iteration is made over all (N) folds of image based validation.

The distinction was made for the input features. One can see that the hardest class to make decision on correctly was lymphocyte, even though it's not the one with the smallest number of samples. I believe it comes from the fact that it usually lays in between the other two in terms of morphology and visual appearance. Presented convolutional neural network architecture exceeds other classifiers in terms of recall for every category.

CNN results were further tested for statistical significance (Ojala and Garriga,

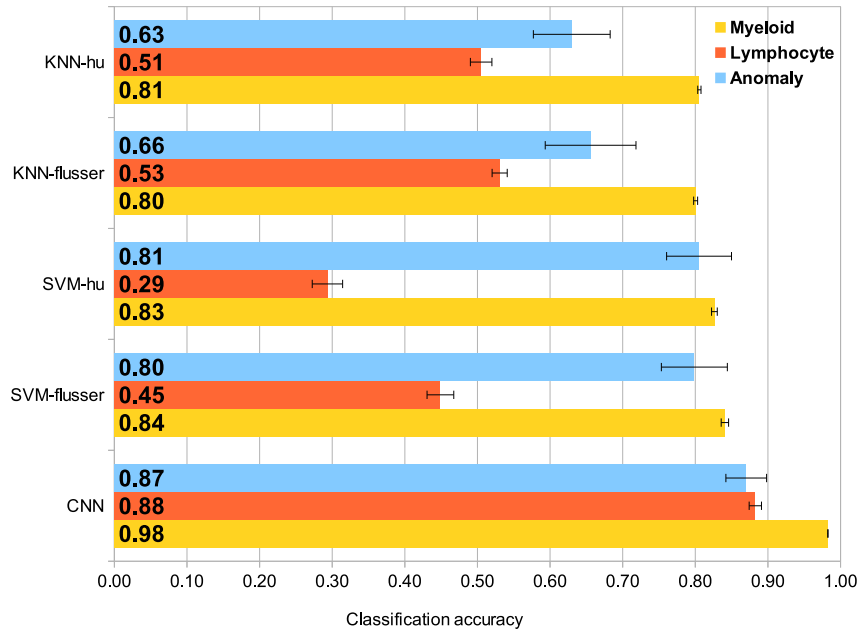


Figure 5.23: Classification results comparison showing per-class recall of classical and deep learning approaches.

2010) and ended up providing the p-value of 0.01. This value can be calculated as:

$$p = \frac{|\{D' \in \hat{D} : e(f, D')\} + 1 \leq e(f, D)\} + 1}{k + 1} \quad (5.4)$$

where D' belongs to a set \hat{D} of data in which labels were randomly permuted, D is the one with original labeling and e computes the misclassification error given cost function f . The idea is that any of random permutation after re-training shouldn't provide a better solution than for the original test. Usually 100 permutations are performed and p-values of ≤ 0.05 are considered statistically significant.

The final counting results for WBC parameters are presented on Fig.5.24 and further summarized in Tab.5.3. The further discussion of those results is done in the next section.

	WBC	Lymphocyte	Myeloid
R^2	0.946	0.894	0.949

Table 5.3: Platelet R^2 summary table (CNN)

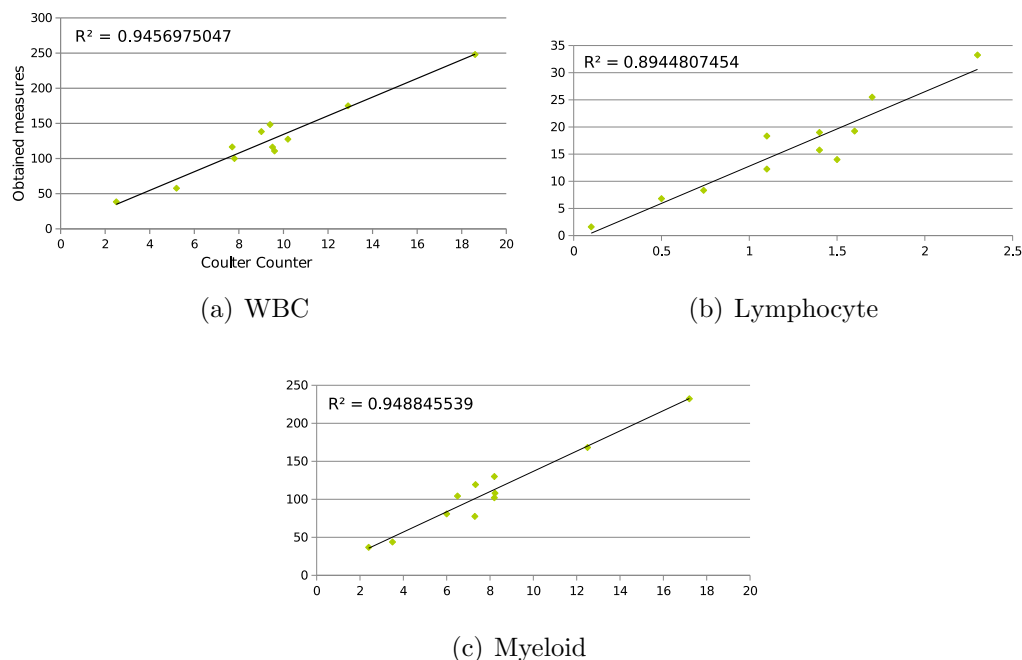


Figure 5.24: Squared Pearson correlation of total (a) WBC (b) Lymphocyte (c) Myeloid count with Coulter Counter readings for given patients

5.3 Discussion on results

5.3.1 Erythrocyte

Treating Coulter Counter readings as a ground truth and comparing those correlations directly with Tab.2.1 specifying acceptable accuracy errors we would see the obtained parameters are not acceptable for clinical treatment at this stage. Visual inspection however (e.g. Fig.5.8) seems to be contradictory to those conclusions. All of the inspected images seem to have similar segmentation accuracy. Unfortunately there was no way to quantify it with available resources and time constraints. The issues however seem to be related mainly to the blood treatment. Many images had very uneven density of cells across the image which suggests bad mixing with diluent or uneven surface of the sensor or the chamber top the sample is covered with. The other issues may be of more physiological nature. The intermediate solutions were proposed to deal with few of them. We could correct the error of total RBC count by calculating the number of other small objects in the solution. Even though they are subjected to the same flow issues, they won't change due to cases of hemoglobin loss, death and disappearance before imaging, which also appear to be a common factor (Fig.5.25).

The total count can be corrected by dividing it by the number of those objects. To calibrate for uneven density we can measure the occurrence of the cells across the image and mask out the regions where it is particularly low. Additionally the pipeline includes an attempt to remove anomalies caused by erroneous data acquisition - e.g. by over-heating the sensor. These are usually well-visible in pre-processed red color channel. This channel is often acquired closest in time to violet, which reduces the problem of object motion across channels. Example of such anomaly is presented on Fig.5.26a. Simple thresholding is usually enough to get rid of those regions (Fig.5.26b). As sometimes they cover a significant part of an image, the total area is excluded from further processing and cover percentage saved for future correction.

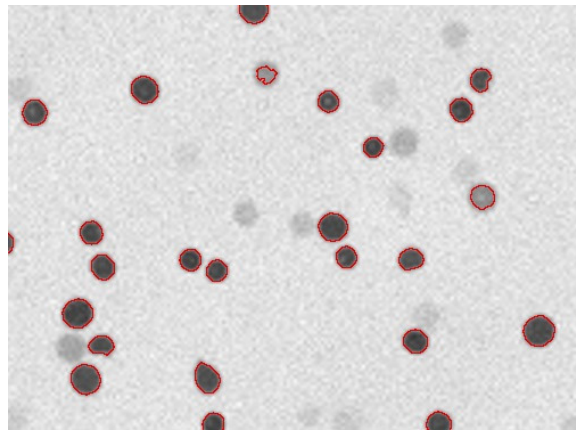
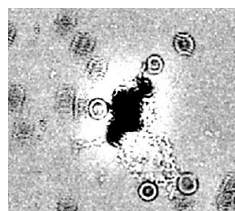


Figure 5.25: Red blood cells fading due to death or loss of hemoglobin.



(a)



(b)

Figure 5.26: (a) Anomaly caused by erroneous data acquisition and mask (b) for removing those and similar areas from further processing.

The mean corpuscular volume (MCV) will need finer resolution and more accurate border tracking in order to catch all the imperfections distinguishing the cells. Also the distance of the cells from the sensor might be an issue and finer body reconstruction

could help. It is interesting though how much this result differs from good result of hematocrit (Hct) and almost perfect one of red cell distribution width (RDW), both of which incorporate scaled MCV.

5.3.2 Thrombocyte

The correlation plot for platelets shows how unstable the R^2 values can be. If we removed the outlier patient 6 from it (marked red on Fig.5.11) the result would change from 0.28 to 0.92. It would then approach the analytic goal (Tab.2.1) very closely. The problem lies in unaccounted changes in platelet appearance in some cases - especially patient 6 (Fig.5.27). The objects appear here more faded and pink instead of violet. This patient had as well a particularly low number of platelets. We could potentially correct for that by including those examples as additional k-means color intensity cluster and adjust thresholds to let them in. This however would cause overfitting and in overall poses a common problem when dealing with this kind of data. We are unable to determine if patient 6 is truly an outlier or it just happened that there are no other examples similar to it. More data from other patients would be needed to verify that. The temporary solution is to normalize the final probability map, so that those faded platelets get as much excitation as normal ones in other images. We cannot however do that directly by dividing the map by maximum excitation as sometimes anomalies (especially pink reflections and moving cells) happen to have even higher probability and adjusting everything for them doesn't solve the problem. So we would have to exclude them first.



Figure 5.27: Typical appearance of platelet (a) and faded platelet commonly occurring within blood samples of patient 6 (b)

5.3.3 Leukocyte

The results for WBCs were particularly successful. Comparison with Tab.2.1 shows that they exceed the analytical goal error (please refer to section 2.1.1), so they could be already used in clinical treatment. Tab.5.4 reports very comparable results for rule-based classification. It shows low complexity of the problem, but doesn't provide a good alternative as it needs much of additional human effort to achieve the same results as CNN. The time for manual adaptation of hyperparameters to each new type of data is prohibitive in quickly changing environment. The parameter space is too large and a structure inefficient to perform automatic optimization. Such classifier is not adaptive - it does not change its structure when new types of objects appear in the image or the environment is varied in other way. The additional comparison of correlation results can be made with CNN trained on randomly selected (instead of patient-based) patches from each patient and thus capturing the whole dataset and more similar information to that available to rule-based classifier (Tab.5.5).

	WBC	Lymphocyte	Myeloid
R^2	0.935	0.93	0.937

Table 5.4: Rule-based classification

	WBC	Lymphocyte	Myeloid
R^2	0.948	0.919	0.952

Table 5.5: CNN after training on validation set

Fig. 5.28 represents small representative subset of examples the network classified perfectly. Each sample has a small horizontal histogram on the left representing probability of affiliation with each of the classes in the same way as in Krizhevsky et al. (2012). The label on top specifies the true label and the legend on the left the affiliated class, i.e. myeloid, lymphocyte or anomaly.

The next figure (Fig. 5.29) depicts few examples which are still recognized correctly, but the network is very uncertain about this result. In some cases we can see why that examples, like the first one, are similar to each classes. It is a contaminant, but a circular blob of roughly $\frac{1}{9}$ of a patch size in its center should create a strong response for a lymphocyte as well. It is also understandable that moving red blood cells around

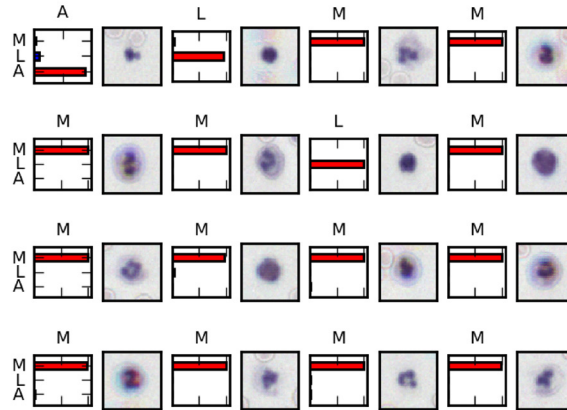


Figure 5.28: Selected positive examples

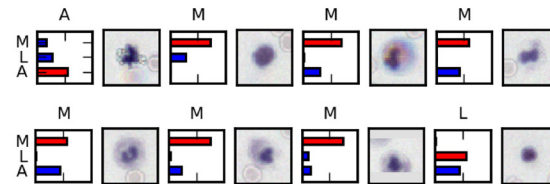
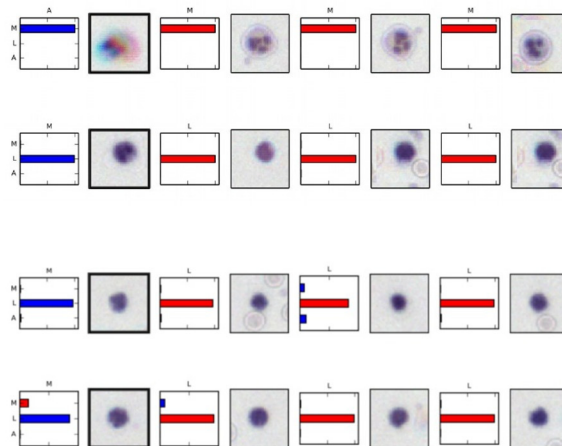
Figure 5.29: Examples of $0.5 \leq p(y|x) < 0.75$ 

Figure 5.30: Most similar examples from other class

and some other objects create a shell around it, which might be to certain degree taken as cytoplasm.

Few examples were chosen from those classified wrongly with very high certainty. To make it clearer why that happened, they were juxtaposed with 3 examples assigned to other classes, but being most similar to them. This similarity was measured in

terms of Euclidean distance in last fully connected layer nodes activations. They are presented in rows on Fig. 5.30. It is well possible that those cells would be wrongly classified by human as well. The first one is actually a myeloid, but because it changed its position during the acquisition, it manifests most of the features common to anomalies. The second one is probably a myeloid, but so small it resembles most lymphocytes. The last two cases are very hard to determine.

Chapter 6

General discussion and future prospects

Given the data and methods a question of more general nature were considered. By comparison of classifiers we asked which approach is faster and more accurate for low resolution image classification - hand-crafted rules or machine learning? The accuracy can be deducted from learning curve, validation and manual inspection of the misclassified cells. Even though rule-based approach in case of leukocyte analysis achieves very comparable results to convolutional neural network, it may indicate a low complexity of the problem. However it doesn't give back endless amount of hours the trained engineer had to spend to obtain such good measures for this particular dataset. As in real life experimental setups the data is subjected to very frequent, sometimes drastic, changes, rule-based system doesn't provide a solution anywhere near practical.

The attempt was made to measure how complex the machine learning algorithm needs to be in order to out-perform the rules created by the expert in terms of accuracy. For that I prepared SVM and few popular network architectures with different amount of layers and other kinds of modifications applied. I ordered them by complexity and for each of them, supported by optimization algorithms (like grid search), I spent similar amount of time attempting to tune the hyper-parameters in order to achieve the best possible performance. The best results were obtained with 2- and 3 - convolutional layers neural networks and only those gave better correlations than rule-based method. The tests presented in results chapter confirmed then that CNN constitutes the state of the art classifier also for the problem of cell analysis with main advantages of development time, minimal human involvement and better accuracy.

Achieving human or over-human performance takes usually a lot of effort in correcting the architectures and tuning the hyper-parameters. In the world where visualizing and understanding the trained models remains an open problem, researchers usually have to spend longer time trying to find the underlying causes of misclassification.

Often this is a matter of trial and error and results with only slight improvements. We already know that human's advantage is flexibility - we can combine simple rules based on completely separate search assumptions - like setting thresholds on object size or eccentricity (major to minor ellipse axis ratio). Adaptation to certain type of features often results in changing the network architecture manually. The advantage of networks though is learning more complex hierarchical recognition models, often based on features which are less general and hard to be picked up by a human being. Sometimes those features are rare within the set and human is likely to neglect them, thus losing this few important percents of accuracy. In the results chapter I provided the visualization of filters and close-by examples especially for that purpose. It can be easily seen that those features are not likely to be picked by human, however they result in better accuracy than when more obvious ones are picked. Comparison of the resulting accuracies of rule-based and deep learning methods suggest that we often don't know what kind of features are most appropriate and most optimal to describe the set of natural images. It is because first of all we still don't fully understand how our brain processes this information, and second, because we learned to generalize to high level features appropriate for all the objects around us, often biased by culture and semantics we learned through our lifetimes.

The lack of explicit knowledge representation poses certain problems for the networks to be picked as default classifiers for certain problems. Information like that may be crucial especially in some medical areas where human life depends on it, where the solution based on neural networks might not be acceptable due to higher prediction uncertainty for unknown data. Usually producing the right examples in sufficient amount for neural network is also costly and can be easily bypassed by flexibility of human mind, which can pick up and implement discrimination based on visible features given only few examples. However there are also many positive aspects causing people to reach for neural networks more and more often. Among others this includes robustness in presence of noise, high adaptability to new problems and the fact that they are universal function approximators - so can be often quickly applied without need of deeper understanding. The last feature is particularly important in quickly changing testing environment of early versions of applications or research studies.

There are few things which might be recommended for future work. The detection and localization of the red blood cells seems appropriate, but correlations with Coulter Counter measures are still not acceptable from clinical point of view. It is important to find the real factors responsible for this fact. First the blood samples are often not spread evenly across the image sample, which causes sometimes huge deviations from the actual counts. The attempt was made to correct for that fact by masking empty regions of the region, but apparently more work has to be done on determining accurate cell concentration in each part of the image. Also there appear to be a lot of problems with clustered cells and including closure of occluded border for each cell - so that it constitutes more circular shape - instead of just raw separation should help in that case. As described in section 5.3 platelet analysis also has flaws in probability map normalization. More research should be done on assuring that weighted values have the same meaning in each picture. All correlation values for each parameter should in general exceed the analytical goal (discussed in the results) in order for the test to become clinically valid.

The future work could also include obtaining images of even better quality so that the ground truth labels for other types of WBCs are obtained and can be incorporated into classification process. RBM pre-training and dropout provide improvements on classification of many types of natural images. I believe they will prove useful also in our case in the very near future when more complex tasks are present and more data are available.

Appendix A

Anisotropic diffusion

The following section specifies some of the background knowledge behind anisotropic diffusion algorithm. Let us take few steps back and recall some of the important aspects of differential geometry (more detailed descriptions can be for example found in Alfred (1998) or Sapiro (2006)). By looking at the problem of smoothing from this perspective we can see how the algorithm preserves the borders by modeling rough contour of each object with strong boundary gradients. Each curve of a standard form (e.g. $ax + by = c$) can be described in a parametric form of $C(t) = \{x(t), y(t)\}$. When we calculate the tangent of that form, $\frac{dC}{dt} = \left\{ \frac{dx}{dt}, \frac{dy}{dt} \right\}$, we get a vector (not a slope as in the standard form), which directly represents the gradient at a particular point in parametric space. This gradient can be normalized by its length $\frac{\frac{dC}{dt}}{\|\frac{dC}{dt}\|}$, which gives us the reparametrization $\frac{dC}{ds}$ for which each gradient is a unit vector. It allows us to travel along the curve with the same "speed" and thus to calculate the arc length between two arbitrary points $s_{a,b} = \int_a^b \|\frac{dC}{dt}\| dt \rightarrow s(\alpha) = \int_0^\alpha \|\frac{dC}{dt}(\tau)\| d\tau$.

For our purpose we are interested in acceleration along the curve $\frac{d^2C}{ds^2} = \kappa N$, where N is the unit vector representing direction of the normal at each point and κ is the normal's magnitude or curvature of the curve. Separating those terms simplifies further calculations. As perpendicular vector for 2D case can be obtained by simple rotation:

$$a^\perp = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} a = \begin{pmatrix} -a_y \\ a_x \end{pmatrix} \quad (\text{A.1})$$

then $N = \left[\frac{\frac{dC}{dt}}{\|\frac{dC}{dt}\|} \right]^\perp$. κ depends on the metric. As we operate in image space we assume the Euclidean metric in which:

$$\kappa = \frac{\left\| \frac{d^2C}{dt^2} \times \left[\frac{dC}{dt} \right]^\perp \right\|}{\left\| \frac{dC}{dt} \right\|^3} \quad (\text{A.2})$$

Having the curvature and the acceleration vector at each point we are able to

perform so called curve evolution and particularly the mean curvature flow in which the curve grows or shrinks across the time on the 2D image based on the strength of the gradients in local neighbourhood. By taking the advantage of strong relation of to physics, we can describe it through a heat equation of the form:

$$\frac{dC}{d\tau} = \nabla^2 C \quad (\text{A.3})$$

The evolution then happens accordingly to temperature, defined by the right side of this equation, across the 3rd dimension and can be visualized as "melting" or "diffusing" the curve. The same equation can be also define as:

$$\frac{dC}{d\tau} = \frac{d^2 C}{ds^2} \quad (\text{A.4})$$

or

$$\frac{dC}{d\tau} = \kappa N \quad (\text{A.5})$$

where τ is the time (not to confuse with the previous term t being a curve parameter).

The same philosophy can be further applied to surfaces in which $C(u, v) = \{x(u, v), y(u, v), z(u, v)\}$ and hence also to image surfaces in which z is the intensity at a particular pixel (x, y) . This allows us to specify the analogous heat equation:

$$\frac{\partial I(x, y, \tau)}{\partial \tau} = \nabla^2 I \quad (\text{A.6})$$

known also as the isotropic diffusion equation. Using definition of Laplacian this can be alternatively represented as:

$$\frac{\partial I(x, y, \tau)}{\partial \tau} = \text{div}(\nabla I) \quad (\text{A.7})$$

Interestingly enough the solution of this equation is equivalent to convolution of the original image with Gaussian kernel G_τ of variance τ :

$$I_\tau = I_0 * G_\tau \quad (\text{A.8})$$

where I_τ is the transformation of the image after time τ .

The anisotropic diffusion algorithm proposed by Perona and Malik (1990) offers the following modification:

$$\frac{\partial I(x, y, \tau)}{\partial \tau} = \text{div}(g(\|\nabla I\|)\nabla I) \quad (\text{A.9})$$

in which the purpose of function $g(x)$ is to stop image diffusion on some specified conditions - usually involving height and consistency of the contours in the image. The edge detection capabilities follow again from the geometry of the landscape and can be easily observed by application of pure Laplacian filter to the image.

Appendix B

Super-resolution

As the task is to find the high resolution (HR) image based on collection of noisy LR images, the standard way of dealing with it is to minimize the difference between those images and our assumption about what happened to the original. The general deconvolution stage can be modeled through filter h , which can represent different artifacts like motion or blur, and the function to minimize would look like this:

$$\min_X \|h * X - y\|_p^p \quad (\text{B.1})$$

where p is the parameter for L_p normalization, \hat{X} is the original scene for which we are looking and y is the distorted input image. To make this mathematical formulation more general we usually represent convolution as a multiplication by matrix H in which each column $H_{:,j}$ represents filter function within specified units shifted by j times the specified discretization constant. Then:

$$\min_X \|HX - y\|_p^p \quad (\text{B.2})$$

allows us to easily extend this approach to different kinds of filtering including e.g. affine transformations.

There is a problem with this formulation though. If we keep minimizing the function, the image we are looking for will become more and more noisy as it is approaching the given ones. This can be alleviated by the Total Variation (TV) regularization term, which, in its simplest form, calculates the normalized gradient at each point of the image. This causes the preservation of large gradients and smooths the rapidly changing parts of the image, like those containing a lot of noise. We further introduce the parameter λ for specifying the trade off between resemblance to filtered version of the original scene and smoothness with detail preservation. Our equation then takes the form:

$$\min_X \|HX - y\|_p^p + \lambda \|\nabla X\|_q \quad (\text{B.3})$$

where q is another norm parameter.

To get the original image we need to perform optimization on the equation involving all of the LR images:

$$\hat{X} = \operatorname{argmin}_X \sum_{k=1}^N \|HX - Y_k\|_p^p + \lambda \|\nabla X\|_q \quad (\text{B.4})$$

Fairly popular approach was described by Farsiu et al. (2004) and can be defined by few modifications. Starting with:

$$\hat{X} = \operatorname{argmin}_X \sum_{k=1}^N \|DF_k HX - Y_k\|_p^p + \lambda \|\nabla X\|_q \quad (\text{B.5})$$

where D is a downsampling operation (by a factor r of resolution enhancement), F_k is a redundant matrix describing the motion vector in terms of repositioning of each pixel from the original image, which shows how much each of the input images must be shifted to register them. H is a Point Spread Function (PSF) of the image projection describing the blur factor. To make the minimization less computationally expensive the specific limited gradient calculation method is presented which result in, so called, Bilateral TV defined as:

$$\Upsilon_{BTV}(X) = \sum_{l=-P}^P \sum_{m=0}^P \alpha^{|m|+|l|} \|X - S_x^l S_y^m X\|_1 \quad (\text{B.6})$$

where $l + m \geq 0$. S_a^b is an affine shift matrix by b pixels in direction of a . This basically calculates the gradient between given pixel and its neighbours with spatially lowered importance factor α . P defines the limits of this discrete calculation and in Farsiu et al. (2004) was usually set up between 1 and 3. Please move to section 4.2.4 for further build up on this background.

Appendix C

Image moments

Moments, in mathematics, capture certain features of the shapes the distributions of points drawn for any random variable form together. The most common are probabilistic distribution moments, where first few can be described as a mean, variance, skewness and kurtosis (or 'peakedness'). All together they can be described with a function:

$$M_p = \int_{-\infty}^{\infty} x^p f(x) dx \quad (\text{C.1})$$

where $f(x)$ describes the underlying distribution. For 2D functions the equivalents are M_{10} , M_{20} , M_{30} , M_{40} for x axis and M_{01} , M_{02} , M_{03} , M_{04} respectively and the general equation is:

$$M_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x, y) dx dy \quad (\text{C.2})$$

where p and q are moment's degrees. The whole idea behind using moments to describe images is by applying the same rules to the whole image as we apply to probability distributions. For example WBC resembles negative 'Mexican hat' function in most of the cases. To see what kind of features are emphasized by each moment it is best to work through examples. The figures below show the 2D Gaussian projected to the discrete image plane, as a representative function, to which different kinds of transformations were applied to show their influence on some of the initial moments. All of them are labeled either 'high', 'negative high', 'low', 'negative low'. This qualitative measure relates directly to the quantitative one of the moment itself. To recall multivariate Gaussian function with correlation factor can be written as:

$$f(x, \mu, \Sigma) = \frac{1}{\sqrt{|\Sigma|(2\pi)^d}} e^{-\frac{1}{2}(x-\mu)\Sigma^{-1}(x-\mu)^T} \quad (\text{C.3})$$

where $d = 2$ is the number of dimensions, x is the variable vector, μ is the mean and σ is the covariance matrix which can be defined as:

$$\Sigma = \begin{pmatrix} \sigma_{11}^2 & -\rho\sigma_{11}\sigma_{22} \\ -\rho\sigma_{11}\sigma_{22} & \sigma_{22}^2 \end{pmatrix} \quad (\text{C.4})$$

where we introduce the correlation factor which extends our set of possible distribution shape transformations. The Gaussian equation can now be written in its direct form as:

$$f'(x, \mu, \rho) = \frac{1}{2\pi\sigma_{11}^2\sigma_{22}^2(1-\rho^2)} e^{-\frac{1}{2\sigma_{11}^2\sigma_{22}^2(1-\rho^2)}((x-\mu_1)^2\sigma_{22}^2+(y-\mu_2)^2\sigma_{11}^2-2(x-\mu_1)(y-\mu_2)\rho\sigma_{11}\sigma_{22})} \quad (\text{C.5})$$

Moment M_{00} is the volume under the function, which is always equal 1 in terms of probability distribution and equal to sum of all the pixels in case of the image. Not to make the redundant examples all the further ones will focus on the x-axis. The first is about changing the mean of our distribution (Fig.C.1), so it is mostly responsible for describing centroid of the object presented on the picture.

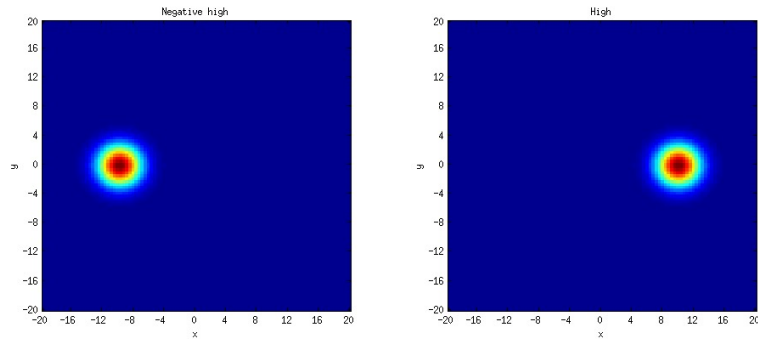


Figure C.1: 2D Gaussian distribution with varied moment $M_{10} = \mu_1$. Negative high (left) and high values(right)

Second example represents the variance (Fig.C.2). Skewness in Gaussian case is just telling us how asymmetrical is the function with respect to point 0 on the chosen axis (Fig.C.3). As the moments are not normalized to the mean at that stage, simple shift from the mean causes skewness to raise. For central moments the density of the distribution must have been spread unevenly in order to obtain similar quantitative result.

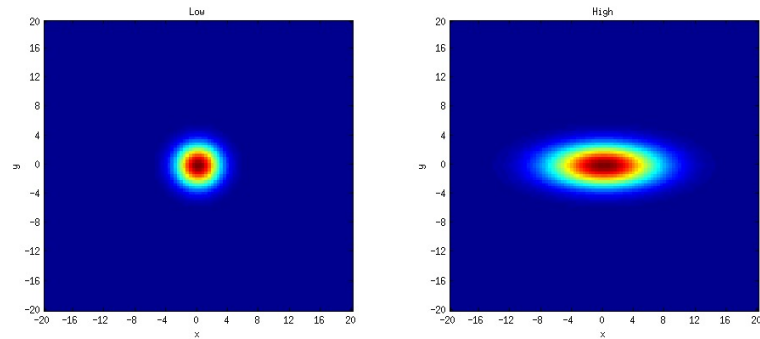


Figure C.2: 2D Gaussian distribution with varied moment M_{20} . Low (left) and high (right) values.

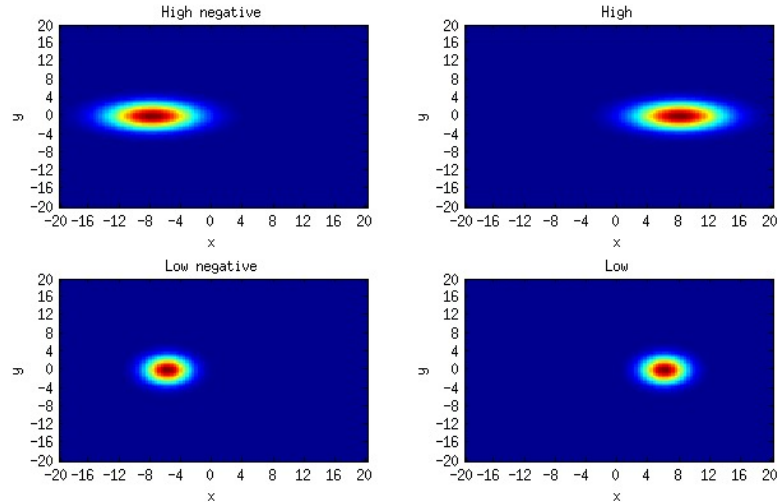


Figure C.3: 2D Gaussian distribution with varied moment M_{30} .

It is interesting to see what happens if we look at some of the moments of mixed order. Fig.C.4 shows us a contrast between high and low correlation. We could think of it as a variance in correlation axis.

Moment M_{21} (Fig.C.5) builds on interrelation of basic moments related to single axes. We can try to describe it as skewness in each of the correlation axes (generated by eigenvectors being more visible when the correlation factor reaches its extrema) summed with correlation along y-axis. The equation defining it can be specified as follows: $M_{21} = \mu_1^2 \mu_2 + \sigma_{11}^2 \mu_2 + 2\rho \mu_1 \sigma_{11} \sigma_{22}$.

The next step after establishing the main functionality of descriptors is to make them invariant to certain transformations. Usually we want the objects to have the

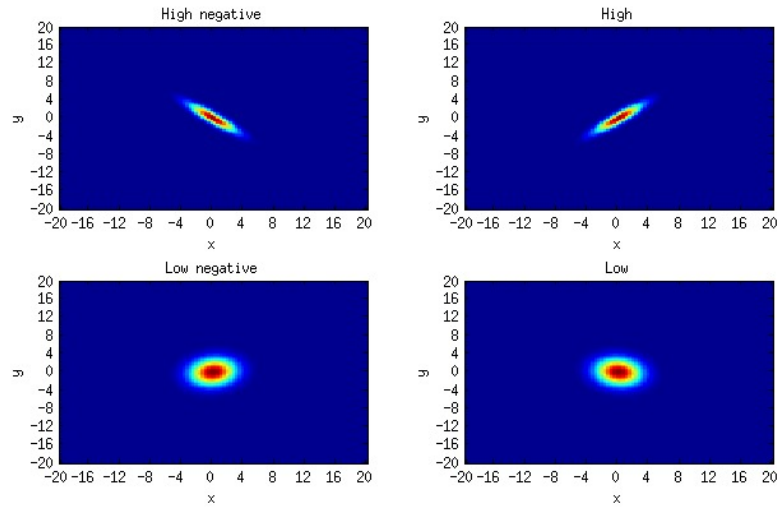


Figure C.4: 2D Gaussian distribution with varied moment M_{11} .

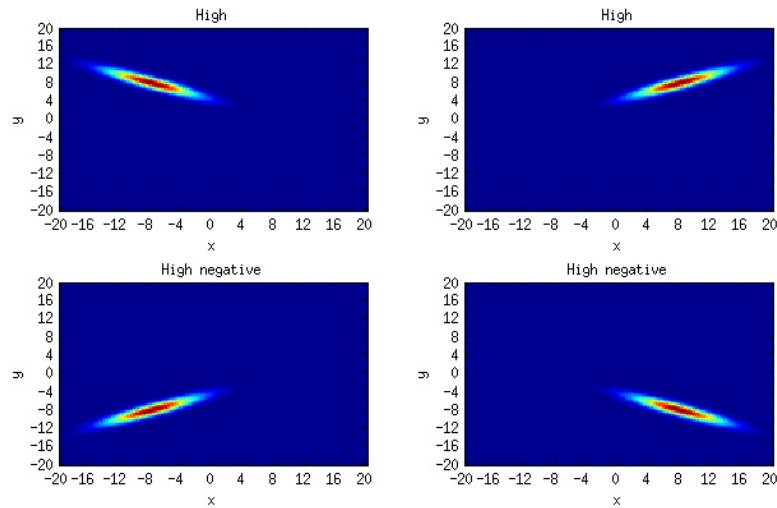


Figure C.5: 2D Gaussian distribution with varied moment M_{21} .

same set of features after operations like translation, rotation and scaling. Moments meeting those conditions are referred to as TRS moments.

Beginning with translation invariance the only thing we have to do is to make sure we do our calculations with respect to centroid of the object, not the origin of the coordinate system. It appears that those kind of moments are also very popular in statistics and are called the central moments, defined as follows:

$$\mu_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x - x_c)^p (y - y_c)^q f(x, y) dx dy \quad (\text{C.6})$$

Transformation	I_1	I_2	I_3	I_4	I_5	I_6	I_7
original	6.6527	17.1070	24.9159	25.6290	51.2049	34.4602	51.3915
half size	6.6533	17.1048	24.8883	25.6236	51.2265	34.4249	51.3220
mirror	6.6527	17.1070	24.9159	25.6290	51.2049	34.4602	51.2954
2° rotation	6.6527	17.1067	24.9156	25.6290	51.2041	34.4602	51.3919

Table C.1: Image moments of examples on Fig.C.6 proving invariance of features under specified transformations

where $x_c = \frac{M_{10}}{M_{00}}$ and $y_c = \frac{M_{01}}{M_{00}}$ are coordinates of the centroid. To further achieve scaling invariance we need to normalize the moments. The most intuitive is to use M_{00} as a factor but any further moment can be used as well. This way we reach the normalized central moment (or TS moment) as:

$$\nu_{pq} = \frac{\mu_{pq}}{\mu_{00}^w} \quad (\text{C.7})$$

where $w = \frac{p+q}{2} + 1$ follows from the properties of algebraic scaling transformation.

The derivation of TRS moments is also based on solution to the system in which original moments and moments after algebraic rotation remain the same, but is far more complex and it won't be presented in this report in details. The most commonly used TRS moments are defined by Hu (1962) as presented in section 4.4.2.

As the original values are usually very small it is common practice to present them by $\ln(I_p)$. For numerical computer calculations we also usually take the absolute of that numbers, so to prevent any potential remains from complex space (imaginary part) due to floating point precision error. On Fig.C.6 we can see an example of calculating those moment invariants from sample image and comparison of the measured values.

Those moments though, even if most commonly used, are proved to be incomplete and dependent on each other by Flusser et al. (2009). The authors propose new set of moments and way to easy generate larger set. They start with distinction of geometric moments (defined as raw moments above) and complex moments of a form:

$$c_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x + iy)^p (x - iy)^q f(x, y) dx dy \quad (\text{C.8})$$

which is just another representation making rotation operations easier to describe. It is worth mentioning that $c_{pq} = c_{qp}^*$ (complex conjugate), which shows that it makes sense to use only one set of those moments - either where $p \geq q$ or $q \geq p$.

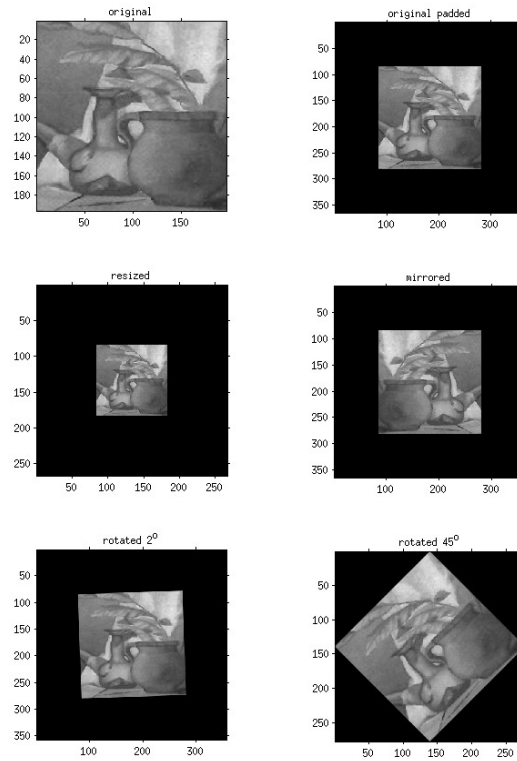


Figure C.6: Examples of an image under transformations of scaling, mirroring, which can be described with exactly the same moment invariants.

We can then define the same moments in polar coordinates as follows:

$$\begin{cases} \rho = \log \sqrt{x^2 + y^2}, \\ \theta = \text{atan2} \frac{y}{x} \text{ if } x > 0. \end{cases} \quad (\text{C.9})$$

$$c_{pq} = \int_0^\infty \int_0^\pi r^{p+q+1} e^{i(p-q)\theta} f(r, \theta) d\theta dr \quad (\text{C.10})$$

where atan2 is a common variation of \arctan (Wikipedia, 2014). If we want to rotate the original image by angle α we get:

$$c'_{pq} = \int_0^\infty \int_0^\pi r^{p+q+1} e^{i(p-q)(\theta-\alpha)} f(r, \theta) dr d\theta = c_{pq} e^{-i(p-q)\alpha} \quad (\text{C.11})$$

so that any rotation in this space causes just a phase shift. To deal with this problems Flusser et al. (2009) propose solution of the form:

$$B = \{\Phi(p, q) \equiv c_{pq} c_{p_0 q_0}^{p-q} : p \geq q \wedge p + q \leq r\} \quad (\text{C.12})$$

where p_0, q_0 are chosen parameters, so that $p_0 + q_0 \leq r$ and $p_0 - q_0 = 1$, and B is the algebraic basis generating functions $\Phi(p, q)$ defining TRS moment invariants.

The authors then show how to present Hu moment invariants in terms of this basis. Setting $p_0 = 2$ and $q_0 = 1$ we have:

$$\begin{aligned}
I_1 &= \Phi(1, 1) \\
I_2 &= \frac{|\Phi(2, 0)|^2}{\Phi(2, 1)^2} \\
I_3 &= \frac{|\Phi(3, 0)|^2}{\Phi(2, 1)^3} \\
I_4 &= \Phi(2, 1) \\
I_5 &= Re(\Phi(3, 0)) \\
I_6 &= Re(\Phi(2, 0)) \\
I_7 &= Im(\Phi(3, 0))
\end{aligned} \tag{C.13}$$

where it can be found that $I_3 = \frac{I_5^2 + I_7^2}{I_4^3}$, so that it depends on the other moments and can be safely removed. We also lack $Im(\Phi(2, 0))$ without which we cannot fully recover some of the raw moments from this set and thus it is incomplete. The following set of 6 moment invariants (for $p_0 = 2, q_0 = 1$) is designed to deal with this issues:

$$\begin{aligned}
F_1 &= \Phi(1, 1) \\
F_2 &= \Phi(2, 1) \\
F_3 &= Re(\Phi(2, 0)) \\
F_4 &= Im(\Phi(2, 0)) \\
F_5 &= Re(\Phi(3, 0)) \\
F_6 &= Im(\Phi(3, 0))
\end{aligned} \tag{C.14}$$

The original Flusser moments can be further normalized to make them invariant to contrast (Flusser et al., 2009). First we need to make them scale invariant and for that we need complex TS moment defined as:

$$\widetilde{c}_{pq} = \frac{c_{pq}}{\mu^{((p+q)/2)+1}} \tag{C.15}$$

then Flusser TRS moment is:

$$\widetilde{\Phi}(p, q) = \widetilde{c}_{pq} \widetilde{c}_{q_0 p_0}^{p-q} \tag{C.16}$$

And finally contrast TRS invariant as:

$$\Psi(p, q) = \frac{\widetilde{\Phi}(p, q)}{|\widetilde{c}_{pq}| \widetilde{\Phi}_{p_0 q_0}^{(p-q)/2}} \quad (\text{C.17})$$

Appendix D

Backpropagation in Convolutional Neural Network

The gradients for backpropagation in convolutional neural network can be calculated as follows. First - to make the notation simpler - it's a common practice to factor out the derivative of the internal activation function $h(x)$. According to the chain rule:

$$\Delta W^l = -\epsilon \frac{\partial E}{\partial W^l} = -\epsilon \frac{\partial E}{\partial h^l} \frac{\partial h^l}{\partial W^l} \quad (\text{D.1})$$

where $h^l = W^l Z^{l-1} + b^l$ for each layer of our network. And thus $\frac{\partial h^l}{\partial W^l} = Z^{l-1}$. As $\frac{\partial h^l}{\partial b^l} = 1$ we gain another view of the gradient, which is often denoted as error function: $\delta^l = \frac{\partial E}{\partial b^l} = \frac{\partial E}{\partial h^l} \frac{\partial h^l}{\partial b^l} = \frac{\partial E}{\partial h^l}$. After that our delta rule takes a simplified form of:

$$\Delta W^l = -\epsilon \delta^l Z^{l-1} \quad (\text{D.2})$$

$$\delta^{out} = \frac{\partial E}{\partial h_j^{out}} = \sum_k \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial h_j^{out}} \quad (\text{D.3})$$

$$\frac{\partial E}{\partial y_k} = -\frac{\partial}{\partial y_k} \sum_j t_j \log y_j = -\frac{\partial}{\partial y} t_k \log y_k = -\frac{t_k}{y_k} \quad (\text{D.4})$$

where k is index of a class.

$$\frac{\partial y_k}{\partial h_j^{out}} = \frac{\partial}{\partial h_j^{out}} \frac{e^{h_k^{out}}}{\sum_n e^{h_n^{out}}} = \frac{\sum_n e^{h_n^{out}} \frac{\partial}{\partial h_j^{out}} e^{h_k^{out}} - e^{h_k^{out}} \frac{\partial}{\partial h_j^{out}} \sum_n e^{h_n^{out}}}{[\sum_n e^{h_n^{out}}]^2} = \frac{e^{h_k^{out}} \hat{\delta}_{jk}}{\sum_n e^{h_n^{out}}} - \frac{e^{h_k^{out}}}{\sum_n e^{h_n^{out}}} \frac{e^{h_j^{out}}}{\sum_n e^{h_n^{out}}} =$$

$$y_k (\hat{\delta}_{jk} - y_j)$$

$$\frac{\partial E}{\partial h_j^{out}} = y_j - t_j \quad (\text{D.5})$$

$$\Delta W_j^l = -\epsilon (y_j - t_j) Z_j^{l-1} \quad (\text{D.6})$$

for $l = out$.

For rectified linear units:

$$\frac{\partial E}{\partial h_j^l} = \frac{\partial \max(0, h_j^l)}{\partial h_j^l} = \begin{cases} 0 & \text{if } x \leq 0, \\ 1 & \text{if } x > 0 \end{cases} \quad (\text{D.7})$$

For conv-pooling layer assume $l + 1$ as pooling and l as current layer. k would be the filter index an m feature map index. Then:

$$\delta^{k,l+1} = \frac{\partial E}{\partial h^{n,l+2}} \frac{\partial h^{n,l+2}}{\partial h^{k,l+1}} = \frac{\partial E}{\partial h^{n,l+2}} \frac{\partial h^{n,l+2}}{\partial Z^{l+1}} \frac{\partial Z^{l+1}}{\partial h^{k,l+1}}. \quad (\text{D.8})$$

As we usually choose sigmoid function as the activation function, this becomes:

$$\delta^{k,l+1} = (\delta^{n,l+2} \circ W^{k,l+2}) Z^{l+1} (1 - Z^{l+1}) \quad (\text{D.9})$$

where \circ is either multiplication or correlation (i.e. convolution with rotated matrix $\widetilde{W}^{k,l+2}$) depending on the layer on top of the pooling one. It's important to notice that convolution would have to be done in 'opposite' fashion than one used for forward propagation, i.e. 'valid' if 'full' was used, 'full' if 'valid' was used and 'same' if 'same' was used.

For the convolutional layer:

$$\delta^{k,l} = \frac{\partial E}{\partial h^{k,l+1}} \frac{\partial h^{n,l+1}}{\partial h^{k,l}} = \delta^{k,l+1} \circ \frac{\partial d(Z^l) + b}{\partial h^{k,l}} \quad (\text{D.10})$$

where \circ is unknown operation at this stage. As the activation function between convolutional and pooling layer is always linear in our models, this becomes:

$$\delta^{k,l} = \delta^{k,l+1} \circ \frac{\partial d(h^{k,l}) + b}{\partial h^{k,l}} = \delta^{k,l+1} \otimes \mathbf{1}_{u \times u} \quad (\text{D.11})$$

where \otimes is Kronecker product and provides efficient implementation of upsampling operation. The output is upsampled to the original size $u \times u$. Following that we will show in detail how to obtain $\frac{\partial E}{\partial W^{l,k,m}}$ and how to implement it.

$$\frac{\partial h^{k,l}}{W^{l,k,m}} = \frac{\partial}{\partial W^{l,k,m}} (W^{l,k,m} * Z^{l-1,m}) \quad (\text{D.12})$$

The indices become more practical for implementation if instead we define $V = \frac{\partial E}{\partial \widetilde{W}^{l,k,m}}$ such that $\frac{\partial E}{\partial W^{l,k,m}} = \widetilde{V}$. Then we have:

$$\frac{\partial}{\partial \widetilde{W}^{l,k,m}} (W^{l,k,m} * Z^{l-1,m}) = \sum_r \sum_s \sum_i \sum_j \widetilde{W}_{r,s}^{l,k,m} Z_{i+r,j+s}^{l-1,m} \quad (\text{D.13})$$

$$\frac{\partial h^{k,l}}{\partial \widetilde{W}_{r,s}^{l,k,m}} = \sum_i \sum_j Z_{i+r,j+s}^{l-1,m} \quad (\text{D.14})$$

$$\frac{\partial E}{\partial \widetilde{W}^{l,k,m}} = \sum_r \sum_s \frac{\partial E}{\partial \widetilde{W}_{r,s}^{l,k,m}} = \frac{\partial E}{\partial h^{k,l}} \frac{\partial h^{k,l}}{\partial \widetilde{W}_{r,s}^{l,k,m}} = \sum_r \sum_s \sum_i \sum_j \delta_{r,s}^{l,k} Z_{i+r,j+s}^{l-1,m} = \widetilde{\delta}^{l,k} * Z^{l-1,m} \quad (\text{D.15})$$

This is always a 'valid' convolution.

Bibliography

- Alfred, G. (1998), *Modern differential geometry of curves and surfaces with Mathematica*, CRC press.
- Bain, B. (2006), *Blood cells: a practical guide*.
- Bankman, I. (2008), *Handbook of medical image processing and analysis*, Academic Press.
- Basavanthally, A. N., Ganesan, S., Agner, S., Monaco, J. P., Feldman, M. D., Tomaszewski, J. E., Bhanot, G. and Madabhushi, A. (2010), ‘Computerized image-based detection and grading of lymphocytic infiltration in her2+ breast cancer histopathology’, *Biomedical Engineering, IEEE Transactions on* **57**(3), 642–653.
- Basu, S., Dahl, K. N. and Rohde, G. K. (2013), ‘Localizing and extracting filament distributions from microscopy images’, *Journal of microscopy* **250**(1), 57–67.
- Bikhet, S. F., Darwish, A. M., Tolba, H. A. and Shaheen, S. I. (2000), Segmentation and classification of white blood cells, in ‘Acoustics, Speech, and Signal Processing, 2000. ICASSP’00. Proceedings. 2000 IEEE International Conference on’, Vol. 6, IEEE, pp. 2259–2261.
- Brunelli, R. (2009), *Template matching techniques in computer vision: Theory and practice*, Wiley.
- Buttarelo, M. and Plebani, M. (2008), ‘Automated blood cell counts: state of the art.’, *American journal of clinical pathology* **130**(1), 104–16.
- Carpenter, A. E., Jones, T. R., Lamprecht, M. R., Clarke, C., Kang, I. H., Friman, O., Guertin, D. A., Chang, J. H., Lindquist, R. A., Moffat, J. et al. (2006), ‘Cellprofiler: image analysis software for identifying and quantifying cell phenotypes’, *Genome biology* **7**(10), R100.
- Chan, T. F. and Vese, L. A. (2001), ‘Active contours without edges’, *Image processing, IEEE transactions on* **10**(2), 266–277.
- Chang, C.-C. and Lin, C.-J. (2011), ‘LIBSVM: A library for support vector machines’, *ACM Transactions on Intelligent Systems and Technology* **2**, 27:1–27:27. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- Chevrefils, C., Cheriet, F., Aubin, C.-E. and Grimard, G. (2009), ‘Texture analysis for automatic segmentation of intervertebral disks of scoliotic spines from mr images’, *Information Technology in Biomedicine, IEEE Transactions on* **13**(4), 608–620.

- Cireşan, D. C., Giusti, A., Gambardella, L. M. and Schmidhuber, J. (2013), Mitosis detection in breast cancer histology images with deep neural networks, *in* ‘Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013’, Springer, pp. 411–418.
- Cortes, C. and Vapnik, V. (1995), ‘Support-vector networks’, *Machine learning* **20**(3), 273–297.
- DAS, D., Chakraborty, C., Mitra, B., Maiti, A. and Ray, A. (2012), ‘Quantitative microscopy approach for shape-based erythrocytes characterization in anaemia’, *Journal of Microscopy* .
- Dundar, M. M., Badve, S., Bilgin, G., Raykar, V., Jain, R., Sertel, O. and Gurcan, M. N. (2011), ‘Computerized classification of intraductal breast lesions using histopathological images’, *Biomedical Engineering, IEEE Transactions on* **58**(7), 1977–1984.
- Dzyubachyk, O., van Cappellen, W. A., Essers, J., Niessen, W. J. and Meijering, E. (2010), ‘Advanced level-set-based cell tracking in time-lapse fluorescence microscopy’, *Medical Imaging, IEEE Transactions on* **29**(3), 852–867.
- Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P. and Bengio, S. (2010), ‘Why does unsupervised pre-training help deep learning?’, *The Journal of Machine Learning Research* **11**, 625–660.
- Farsiu, S., Robinson, M. D., Elad, M. and Milanfar, P. (2004), ‘Fast and robust multiframe super resolution’, *Image processing, IEEE Transactions on* **13**(10), 1327–1344.
- Fischer, A. and Igel, C. (2012), An introduction to restricted boltzmann machines, *in* ‘Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications’, Springer, pp. 14–36.
- Flusser, J. (2006), Moment invariants in image analysis, *in* ‘proceedings of world academy of science, engineering and technology’, Vol. 11, pp. 196–201.
- Flusser, J., Zitova, B. and Suk, T. (2009), *Moments and moment invariants in pattern recognition*, Wiley.
- Fukushima, K. (1988), ‘Neocognitron: A hierarchical neural network capable of visual pattern recognition’, *Neural networks* **1**(2), 119–130.
- Ge, J., Gong, Z., Chen, J., Liu, J., Nguyen, J., Yang, Z., Wang, C. and Sun, Y. (2014), ‘A system for counting fetal and maternal red blood cells.’, *IEEE transactions on bio-medical engineering* .
- George-Gay, B. and Parker, K. (2003), ‘Understanding the complete blood count with differential’, *Journal of PeriAnesthesia Nursing* **18**(2), 96–117.

- Glorot, X., Bordes, A. and Bengio, Y. (2011), Deep sparse rectifier networks, *in* ‘Proceedings of the 14th International Conference on Artificial Intelligence and Statistics. JMLR W&CP Volume’, Vol. 15, pp. 315–323.
- Gonzalez, R., Woods, R. and Eddins, S. (2004), *Digital image processing using MATLAB*, Pearson Education India.
- Goodfellow, I. J., Warde-Farley, D., Lamblin, P., Dumoulin, V., Mirza, M., Pascanu, R., Bergstra, J., Bastien, F. and Bengio, Y. (2013), ‘Pylearn2: a machine learning research library’, *arXiv preprint arXiv:1308.4214* .
- Gurari, D., Theriault, D., Sameki, M. and Betke, M. (2014), How to use level set methods to accurately find boundaries of cells in biomedical images? evaluation of six methods paired with automated and crowdsourced initial contours., *in* ‘Interactive Medical Image Computing Workshop MICCAI’.
- Habibzadeh, M., Krzyżak, A. and Fevens, T. (2013), White blood cell differential counts using convolutional neural networks for low resolution images, *in* ‘Artificial Intelligence and Soft Computing’, Springer, pp. 263–274.
- Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. and Witten, I. H. (2009), ‘The weka data mining software: an update’, *ACM SIGKDD explorations newsletter* **11**(1), 10–18.
- Held, M., Schmitz, M. H., Fischer, B., Walter, T., Neumann, B., Olma, M. H., Peter, M., Ellenberg, J. and Gerlich, D. W. (2010), ‘Cellcognition: time-resolved phenotype annotation in high-throughput live cell imaging’, *Nature methods* **7**(9), 747–754.
- Hiremath, P., Bannigidad, P. and Geeta, S. (2010), ‘Automated identification and classification of white blood cells (leukocytes) in digital microscopic images’, *IJCA special issue on recent trends in image processing and pattern recognition RTIPPR* pp. 59–63.
- Hu, M.-K. (1962), ‘Visual pattern recognition by moment invariants’, *Information Theory, IRE Transactions on* **8**(2), 179–187.
- Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S. and Darrell, T. (2014), Caffe: Convolutional architecture for fast feature embedding, *in* ‘Proceedings of the ACM International Conference on Multimedia’, ACM, pp. 675–678.
- Kapelner, A., Lee, P. P. and Holmes, S. (2007), An interactive statistical image segmentation and visualization system, *in* ‘Medical Information Visualisation-BioMedical Visualisation, 2007. MediVis 2007. International Conference on’, IEEE, pp. 81–86.
- Kozubek, M. and Matula, P. (2000), ‘An efficient algorithm for measurement and correction of chromatic aberrations in fluorescence microscopy’, *Journal of Microscopy* **200**(3), 206–217.

- Krizhevsky, A. and Hinton, G. (2009), ‘Learning multiple layers of features from tiny images’, *Computer Science Department, University of Toronto, Tech. Rep.*
- Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2012), Imagenet classification with deep convolutional neural networks, *in* ‘Advances in neural information processing systems’, pp. 1097–1105.
- LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998), ‘Gradient-based learning applied to document recognition’, *Proceedings of the IEEE* **86**(11), 2278–2324.
- Lee, H., Grosse, R., Ranganath, R. and Ng, A. Y. (2009), Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations, *in* ‘Proceedings of the 26th Annual International Conference on Machine Learning’, ACM, pp. 609–616.
- Maas, A. L., Hannun, A. Y. and Ng, A. Y. (2013), Rectifier nonlinearities improve neural network acoustic models, *in* ‘ICML Workshop on Deep Learning for Audio, Speech, and Language Processing’.
- McClatchey, K. D. (2002), *Clinical laboratory medicine*, Lippincott Williams & Wilkins.
- Möller, M., Burger, M., Dieterich, P. and Schwab, A. (2014), ‘A framework for automated cell tracking in phase contrast microscopic videos based on normal velocities’, *Journal of Visual Communication and Image Representation* **25**(2), 396–409.
- Ojala, M. and Garriga, G. C. (2010), ‘Permutation tests for studying classifier performance’, *The Journal of Machine Learning Research* **11**, 1833–1863.
- Ongun, G., Halici, U., Leblebicioglu, K., Atalay, V., Beksac, M. and Beksac, S. (2001a), An automated differential blood count system, *in* ‘Engineering in Medicine and Biology Society, 2001. Proceedings of the 23rd Annual International Conference of the IEEE’, Vol. 3, IEEE, pp. 2583–2586.
- Ongun, G., Halici, U., Leblebicioglu, K., Atalay, V., Beksac, M. and Beksac, S. (2001b), Feature extraction and classification of blood cells for an automated differential blood count system, *in* ‘Neural Networks, 2001. Proceedings. IJCNN’01. International Joint Conference on’, Vol. 4, IEEE, pp. 2461–2466.
- Online resource* (2012a), www.healthtestingcenters.com.
- Online resource* (2012b), www.idexx.com.
- Paragios, N. and Deriche, R. (2002), ‘Geodesic active regions: A new framework to deal with frame partition problems in computer vision’, *Journal of Visual Communication and Image Representation* **13**(1), 249–268.

- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E. (2011), ‘Scikit-learn: Machine learning in Python’, *Journal of Machine Learning Research* **12**, 2825–2830.
- Perona, P. and Malik, J. (1990), ‘Scale-space and edge detection using anisotropic diffusion’, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **12**(7), 629–639.
- Prewitt, J. and Mendelsohn, M. L. (1966), ‘The analysis of cell images*’, *Annals of the New York Academy of Sciences* **128**(3), 1035–1053.
- Ramoser, H., Laurain, V., Bischof, H. and Ecker, R. (2006), Leukocyte segmentation and classification in blood-smear images, in ‘Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the’, IEEE, pp. 3371–3374.
- Rumelhart, D. E., Hinton, G. E. and Williams, R. J. (1988), ‘Learning representations by back-propagating errors’, *Cognitive modeling* .
- Sapiro, G. (2006), *Geometric partial differential equations and image analysis*, Cambridge university press.
- Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B. et al. (2012), ‘Fiji: an open-source platform for biological-image analysis’, *Nature methods* **9**(7), 676–682.
- Schneider, C. A., Rasband, W. S., Eliceiri, K. W., Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S. et al. (2012), ‘671 nih image to imagej: 25 years of image analysis’, *Nature methods* **9**(7).
- Selinummi, J., Seppala, J., Yli-Harja, O. and Puhakka, J. A. (2005), ‘Software for quantification of labeled bacteria from digital microscope images by automated image analysis’, *Biotechniques* **39**(6), 859.
- Shi, Y., Gao, Y., Yang, Y., Zhang, Y. and Wang, D. (2013), ‘Multimodal sparse representation-based classification for lung needle biopsy images’, *Biomedical Engineering, IEEE Transactions on* **60**(10), 2675–2685.
- Sommer, C., Straehle, C., Koethe, U. and Hamprecht, F. A. (2011), ‘”ilastik: Interactive learning and segmentation toolkit”, in ‘8th IEEE International Symposium on Biomedical Imaging (ISBI 2011)’.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. (2014), ‘Dropout: A simple way to prevent neural networks from overfitting’, *The Journal of Machine Learning Research* **15**(1), 1929–1958.
- Sui, D. and Wang, K. (2013), ‘A counting method for density packed cells based on sliding band filter image enhancement’, *Journal of microscopy* **250**(1), 42–49.

- Sutskever, I., Martens, J., Dahl, G. and Hinton, G. (2013), On the importance of initialization and momentum in deep learning, *in* ‘Proceedings of the 30th International Conference on Machine Learning (ICML-13)’, pp. 1139–1147.
- Thibault, G., Angulo, J. and Meyer, F. (2014), ‘Advanced statistical matrices for texture characterization: Application to cell classification’, *Biomedical Engineering, IEEE Transactions on* **61**(3), 630–637.
- Turgeon, M. L. (2004), *Clinical hematology: theory and procedures*, Vol. 936, Lippincott Williams & Wilkins.
- Vink, J., Laubscher, M., Vlutters, R., Silamut, K., Maude, R., Hasan, M. and HAAN, G. (2013), ‘An automatic vision-based malaria diagnosis system’, *Journal of microscopy* **250**(3), 166–178.
- Wang, Q., You, L. and West, M. (2005), ‘Celltracer: Software for automated image segmentation and line-age mapping for single-cell studies’.
- Wikipedia (2012), ‘Opencv — Wikipedia, the free encyclopedia’, <http://http://en.wikipedia.org/wiki/OpenCV>. [Online; accessed 31-Aug-2012].
- Wikipedia (2013a), ‘Lanczos resampling — Wikipedia, the free encyclopedia’, http://en.wikipedia.org/wiki/Lanczos_resampling. [Online; accessed 10-Aug-2013].
- Wikipedia (2013b), ‘White blood cell — Wikipedia, the free encyclopedia’, http://http://en.wikipedia.org/wiki/White_blood_cell. [Online; accessed 26-Jul-2013].
- Wikipedia (2014), ‘Polar coordinate system — Wikipedia, the free encyclopedia’, http://en.wikipedia.org/wiki/Polar_coordinate_system. [Online; accessed 13-Dec-2014].
- Wu, Q., Merchant, F. and Castleman, K. (2010), *Microscope image processing*, Academic Press.
- Wu, T., Lu, J., Lu, Y., Liu, T. and Yang, J. (2013), ‘Embryo zebrafish segmentation using an improved hybrid method’, *Journal of microscopy* **250**(1), 68–75.
- Xiong, W., Ong, S.-H., Lim, J.-H., Foong, K. W., Liu, J., Racoceanu, D., Chong, A. G. and Tan, K. S. (2010), ‘Automatic area classification in peripheral blood smears’, *Biomedical Engineering, IEEE Transactions on* **57**(8), 1982–1990.
- Yampri, P., Pintavirooj, C., Daochai, S. and Teartulakarn, S. (2006), White blood cell classification based on the combination of eigen cell and parametric feature detection, *in* ‘Industrial Electronics and Applications, 2006 1ST IEEE Conference on’, IEEE, pp. 1–4.
- Young, I. T. (1972), ‘The classification of white blood cells’, *Biomedical Engineering, IEEE Transactions on* (4), 291–298.

Zeiler, M. D. and Fergus, R. (2013), ‘Visualizing and understanding convolutional neural networks’, *arXiv preprint arXiv:1311.2901* .