

METASTASIS DETECTION AND LOCALIZATION IN LYMPH NODES BY
USING CONVOLUTIONAL NEURAL NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MUSTAFA ÜMİT ÖNER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2016

Approval of the thesis:

**METASTASIS DETECTION AND LOCALIZATION IN LYPMH NODES BY
USING CONVOLUTIONAL NEURAL NETWORKS**

submitted by **MUSTAFA ÜMİT ÖNER** in partial fulfilment of the requirements
for the degree of **Master of Science in Electrical and Electronics Engineering,**
Middle East Technical University by,

Prof. Dr. Gülbin Dural Ünver
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Tolga Çiloğlu
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. Uğur Halıcı
Supervisor, **Electrical and Electronics Engineering Dept., METU** _____

Examining Committee Members:

Prof. Dr. Gözde Bozdağı Akar
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. Uğur Halıcı
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. Aydın Alatan
Electrical and Electronics Engineering Dept., METU _____

Assoc. Prof. Dr. Rengül Çetin Atalay
Graduate School of Informatics - Bioinformatics Dept., METU _____

Assist. Prof. Dr. Mehmet Dikmen
Computer Engineering Dept., Başkent University _____

Date: 05.09.2016

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Mustafa Ümit Öner

Signature :

ABSTRACT

METASTASIS DETECTION AND LOCALIZATION IN LYMPH NODES BY USING CONVOLUTIONAL NEURAL NETWORKS

Öner, Mustafa Ümit

M. Sc., Department of Electrical and Electronics Engineering

Supervisor: Prof. Dr. Uğur Halıcı

September 2016, 120 pages

Breast cancer digital histopathology is a new application area of deep learning. Breast cancer was the leading cause of cancer death among women with 15.1% death rate among all cancer deaths in the world in 2012. Insufficient number of pathologists is one of the key factors in that situation. There were 5.7 pathologists per 100.000 people in USA in 2013 and this value was 1.56 in Turkey in 2011. It is possible to increase the number of slide analysis made by the pathologists within the same period by developing deep learning based systems to assist them.

In this thesis, a convolutional neural networks based system is introduced. This system accepts the whole slide images of lymph node excisions from breast cancer patients as input and detects and localizes metastasis regions on these images automatically. In this system, performance values of 0.9259 and 0.8669 for slide-based evaluation and 0.5349 and 0.4060 values for the lesion based evaluation are achieved on CAMELYON16 training and test sets, respectively.

Keywords: Deep Learning, Convolutional Neural Networks, Metastasis Detection in Lymph Nodes, Breast Cancer Digital Histopathology

ÖZ

EVRIŞİMSEL SİNİR AĞLARI KULLANILARAK LENF DÜĞÜMLERİNDE METASTAZ TESPİTİ VE KONUMLANDIRILMASI

Öner, Mustafa Ümit

Yüksek Lisans, Elektrik Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Prof. Dr. Uğur Halıcı

Eylül 2016, 120 sayfa

Meme kanseri dijital histopatolojisi, derin öğrenme algoritmaları için yeni bir uygulama alanıdır. Dünyada 2012 yılı itibarıyla kadınlar arasında kanser kaynaklı ölümlerin başında, ölümlerin %15'inden sorumlu olan meme kanseri gelmektedir. Yetersiz patoloğ sayıları bu durumda önemli bir rol oynamaktadır. 2013 yılında Amerikada 100.000 kişiye 5.7 patoloğ düşerken bu sayı Türkiye'de 2011 yılında 1.56 olmuştur. Bu noktada, birçok alanda en iyi çözümleri sunan derin öğrenme yöntemleri tabanlı sistemlerle patoloğların birim zamanda inceledikleri slayt görüntülerinin sayısını artırmak mümkün olabilir.

Bu çalışmada evrişimsel sinir ağları tabanlı bir yöntem geliştirilmiştir. Bu yöntem, meme kanseri hastalarının lenf düğümlerinden alınan biyopsi örneklerinin bütün slayt görüntülerini girdi olarak almakta ve bu görüntüler üzerinde metastaz bölgelerini tespit ederek bu bölgelerin konumlarını otomatik olarak belirlemektedir. Bu sistemde slayt seviyesi ve lezyon seviyesi performans ölçüm değerlerinde CAMELYON16 öğretme setinde 0.9259 ve 0.8669, test setinde ise 0.5349 ve 0.4060 değerlerine ulaşılmıştır.

Anahtar Kelimeler: Derin Öğrenme, Evrişimsel Sinir Ağları, Lenf Düğümlerinde Metastaz Tespiti, Meme Kanseri Dijital Histopatolojisi

To My Parents

ACKNOWLEDGEMENTS

I would like to express my gratitude and deep appreciation to my supervisor Prof. Dr. Uğur Halıcı for her guidance, valuable suggestions and contributions.

I would like to express my thanks to Assoc. Prof. Dr. Rengül Çetin Atalay for her contributions in the medical side of this study.

I would like to express my thanks and appreciation to CAMELYON16 organizing team for the database they provided and for their efforts to organize this challenge.

I would also like to thanks to TUBİTAK 2210-A General Domestic Master of Science Scholarship Program (TUBİTAK 2210-A Genel Yurt İçi Yüksek Lisans Burs Programı) for partial scholarship support.

I would like to express my thanks to ODTÜ BAP since this study was partially supported under grant BAP-03-01-2016-002. NVidia TitanX GPU card funded by this

BAP project is used for CUDA GPU implementation of CNN architecture.

Finally, I must express my very profound gratitude to my parents Eda Öner and Necdet Öner for providing me with unfailing support and continuous encouragement throughout my years of study. This accomplishment would not have been possible without them. Thank you.

TABLE OF CONTENTS

ABSTRACT	v
ÖZ	vi
ACKNOWLEDGEMENTS	viii
TABLE OF CONTENTS	ix
LIST OF TABLES	xiii
LIST OF FIGURES	xv
CHAPTERS	
1. INTRODUCTION	1
1.1. Motivation	2
1.2. Scope	2
1.3. Contribution	3
1.4. Organization	3
2. BACKGROUND INFORMATION	5
2.1. Breast Cancer	5
2.1.1. How does breast cancer spread?	6
2.1.2. What are the key statistics about breast cancer?	8
2.1.3. Computer-aided Assisting Services	9
2.2. Breast Cancer Digital Histopathology.....	11
2.2.1. Tissue Preparation and Imaging.....	13
2.2.2. CAMELYON16: ISBI Challenge on Cancer Metastasis Detection in Lymph Node WSI Dataset	14
2.2.3. Related Work on Automated Breast Cancer Assesment.....	19
2.3. Deep Learning and Convolutional Neural Networks.....	23

2.3.1.	Convolutional Neural Networks Architecture Overview	25
2.3.2.	Convolutional Layer	27
2.3.3.	Pooling Layer	42
2.3.4.	Normalization Layer.....	43
2.3.5.	Fully Connected Layer	44
2.3.6.	Convolutional Neural Network Layer Patterns	44
2.3.7.	Training of Convolutional Neural Networks.....	46
2.4.	Performance Measures	48
2.4.1.	True Positive Rate	48
2.4.2.	False Negative Rate	48
2.4.3.	False Positive Rate	49
2.4.4.	True Negative Rate.....	49
2.4.5.	Accuracy.....	49
2.4.6.	Performance Measures Used in CAMELYON16	49
3.	PROPOSED APPROACH	51
3.1.	Block Schema	51
3.2.	Pre-processing	52
3.3.	Classification	55
3.3.1.	Dataset for Supervised Training of Convolutional Neural Network.....	55
3.3.2.	CNN Structure and Training	56
3.4.	Post-processing	57
4.	EXPERIMENTAL RESULTS	63
4.1.	Development Platforms and Environments.....	63
4.2.	Layer Selection.....	64
4.3.	Parameter Selection for Pre-processing Operations	65
4.3.1.	Median Filter Size	65

4.3.2.	Connected Component Elimination Threshold Value	66
4.4.	Classification Stage CNN Architectures	69
4.4.1.	CNN Architecture: Model 0.....	70
4.4.2.	CNN Architecture: Model 1.....	72
4.4.3.	CNN Architecture: Model 2.....	74
4.4.4.	CNN Architecture: Model 3.....	76
4.4.5.	CNN Architecture: Model 4.....	78
4.4.6.	CNN Architecture: Model 5.....	80
4.4.7.	Comparison of CNN Architectures	83
4.5.	Decision Fusion.....	85
4.5.1.	Fusion Filters.....	85
4.5.2.	Performance Comparison of Post-processing Operations.....	87
4.6.	Performance of Proposed Approach	88
5.	CONCLUSION.....	97
	REFERENCES.....	101
	APPENDICES	
	APPENDIX A: PERFORMANCE MEASURE PLOTS OF HARVARD MEDICAL SCHOOL AND MASSACHUSETTS INSTITUTE OF TECHNOLOGY TEAM.....	107
	APPENDIX B: PERFORMANCE MEASURE PLOTS OF EXB RESEARCH AND DEVELOPMENT TEAM.....	108
	APPENDIX C: PERFORMANCE MEASURE PLOTS OF INDIVIDUAL PARTICIPANT QUINCY WONG.....	109
	APPENDIX D: PERFORMANCE MEASURE PLOTS OF METU TEAM	110
	APPENDIX E: PERFORMANCE MEASURE PLOTS OF NLP LOGIX TEAM.....	111
	APPENDIX F: PERFORMANCE MEASURE PLOTS OF RADBOUD UNIVERSITY MEDICAL CENTER TEAM.....	112

APPENDIX G: TRAINING HISTOGRAMS OF CAMELYON16 CHALLENGE	
MODEL.....	113

LIST OF TABLES

TABLES

Table 2.1: Convolutional Layer and Fully-Connected Layer Comparison.....	31
Table 2.2: Convolutional Layer Hyperparameters Relation Summary.....	34
Table 2.3: Convolution Operation to Compute the Output Activation Value of $O[0,0,0]$	37
Table 2.4: Convolution Operation to Compute the Output Activation Value of $O[1,0,0]$	38
Table 2.5: Convolution Operation to Compute the Output Activation Value of $O[2,0,0]$	39
Table 2.6: Convolution Operation to Compute the Output Activation Value of $O[2,2,0]$	40
Table 2.7: Convolution Operation to Compute the Output Activation Value of $O[3,3,1]$	41
Table 2.8: Classification Error Rate and Cross Entropy Error Rate Comparison.....	47
Table 2.9: Basic Statistical Definitions Contingency Table	48
Table 4.1: Workstation Technical Specifications	64
Table 4.2: Desktop Computer Technical Specifications.....	64
Table 4.3: Summary of CNN Architectures of Models	69
Table 4.4: Confusion Matrix for CCN Model 0.....	72
Table 4.5: Confusion Matrix for CNN Model 1	74
Table 4.6: Confusion Matrix for CNN Model 2	76
Table 4.7: Confusion Matrix for CNN Model 3	78
Table 4.8: Confusion Matrix for CNN Model 4	80
Table 4.9: Confusion Matrix for CNN Model 5	82
Table 4.10: Accuracy Values Obtained with Six Different CNN Architectures	83
Table 4.11: First Decision Fusion Filter (FILTER1) Weights	86
Table 4.12: Second Decision Fusion Filter (FILTER2) Weights	86

Table 4.13: Third Decision Fusion Filter (FILTER3) Weights.....	87
Table 4.14: Performance Comparison of Fusion Filters and Erosion Disk Sizes	88
Table 4.15: Confusion Matrix of Final Trained Model over Training Set.....	88
Table 4.16: Color Coding Scheme in Evaluation Images	89
Table 4.17: Performance Comparison Among Top 10 Ranked Teams in Slide Based Category	95

LIST OF FIGURES

FIGURES

Figure 2.1: Breast profile	6
Figure 2.2: Breast Profile with surrounding lymph system components	7
Figure 2.3: Sentinel lymph node biopsy illustration	8
Figure 2.4: Digital histopathology pyramidal image storage: z-stack structure	12
Figure 2.5: Images from different levels of an example WSI.....	12
Figure 2.6: Tumor_110 sub-images from different magnification levels on which metastasis regions are enclosed by blue dotted curves: (a) 2944x2240 image at level 5, (b) 3840x2160 image at level 4, (c) 3840x2160 image at level 3, (d) 3840x2160 image at level 2, (e) 3840x2160 image at level 1, (f) 3840x2160 image at level 0..	16
Figure 2.7: Normal_076 sub-images from different magnification levels: (a) 1390x1770 image at level 5, (b) 1390x1770 image at level 4, (c) 1390x1770 image at level 3, (d) 1390x1770 image at level 2, (e) 1390x1770 image at level 1, (f) 1390x1770 image at level 0	16
Figure 2.8: Tumor_110 whole slide image - 3840x2160 sub-image from layer 0 ...	17
Figure 2.9: Normal_076 whole slide image – 1390x1770 sub-image from layer 0 .	18
Figure 2.10: Top 10 ranked Teams in CAMELYON16 Slide-based Evaluation Category - https://grand-challenge.org/site/camelyon16/results/	20
Figure 2.11: a) Regular Neural Network Structure, b) Convolutional Neural Network Structure	26
Figure 2.12: Neuron Activation Scheme.....	27
Figure 2.13: Weight Sharing in a Depth Slice or Feature Map.....	30
Figure 2.14: Visualization of Convolutional Layer Hyperparameters.....	33
Figure 2.15: Visualization of Elements in Convolution Operation.....	35
Figure 2.16: Non-Overlapping and Overlapping Max Pooling Examples.....	43
Figure 3.1: Block schema of the proposed method.....	51

Figure 3.2: Effects of Preprocessing Operations on Tumor_009 Image: a) Original image, b) Otsu thresholding, c) Median filtering, d) Small connected component elimination (mask), e) Final output of preprocessing stage (masked image), f) Metastasis region boundaries shown on original image and g) Metastasis region boundaries shown on masked image.....	53
Figure 3.3: Example Dataset Images - First row: Samples with label “normal”, Second row: Samples with label “Tumor”	55
Figure 3.4: Convolutional Neural Network Architecture.....	56
Figure 3.5: Calculation of Probability of Belonging to Metastasis Region for a Pixel at (I,J).....	61
Figure 3.6: Post Processing Stages for Tumor_009 Image: a) Binary image showing metastasis regions constructed from CNN output labels, b) Eroded binary image eliminating small regions, c) Probability image obtained after Confidence Filtering (green area), d) Metastasis representative points shown on probability image, e) Metastasis representatives shown on evaluation mask image.....	62
Figure 4.1: The Effect of Median Filter Size on Pre-processed Image: a) Pre-processed image with filter size of 2, b) Pre-processed image with filter size of 5, c) Pre-processed image with filter size of 10	66
Figure 4.2: Result of connected component elimination with 800 pixels threshold value: a) Metastasis boundaries shown on original WSI, b) Metastasis boundaries shown on pre-processed WSI with 800 pixels connected component elimination threshold value	68
Figure 4.3: Learning Rate Update Graph during Training of the Models.....	70
Figure 4.4: First Convolutional Layer Filter Images of Challenge Model.....	72
Figure 4.5: Block Diagram of Model 1 CNN Architecture.....	73
Figure 4.6: First Convolutional Layer Filter Images of Model 1	74
Figure 4.7: Block Diagram of Model 2 CNN Architecture.....	75
Figure 4.8: First Convolutional Layer Filter Images of Model 2.....	76
Figure 4.9: Block Diagram of Model 3 CNN Architecture.....	77
Figure 4.10: First Convolutional Layer Filter Images of Model 3	78
Figure 4.11: Block Diagram of Model 4 CNN Architecture.....	79
Figure 4.12: First Convolutional Layer Filter Images of Model 4.....	79

Figure 4.13: Block Diagram of Model 5 CNN Architecture	81
Figure 4.14: First Convolutional Layer Filter Images of Model 5.....	82
Figure 4.15: Color Coded Evaluation Image of Tumor_089 WSI.....	90
Figure 4.16: False Positive Image Patches from Tumor_089 WSI.....	90
Figure 4.17: Color Coded Evaluation Image of Normal_066 WSI	91
Figure 4.18: False Positive Image Patches from Normal_066 WSI	91
Figure 4.19: ROC Curve of Proposed Approach on Training Set	92
Figure 4.20: FROC Curve of Proposed Approach on Training Set	93
Figure 4.21: ROC Curve of Proposed Approach on Test Set	94
Figure 4.22: FROC Curve of Proposed Approach on Test Set	94
Figure A.1: ROC Curve of Harvard Medical School and Massachusetts Institute of Technology Team.....	107
Figure A.2: FROC Curve of Harvard Medical School and Massachusetts Institute of Technology Team.....	107
Figure B.1: ROC Curve of ExB Research and Development	108
Figure B.2: FROC Curve of ExB Research and Development.....	108
Figure C.1: ROC Curve of Individual Participant Quincy Wong.....	109
Figure C.2: FROC Curve of Individual Participant Quincy Wong.....	109
Figure D.1: ROC Curve of METU Team.....	110
Figure D.2: FROC Curve of METU Team	110
Figure E.1: ROC Curve of NLP LOGIX Team	111
Figure E.2: FROC Curve of NLP LOGIX Team	111
Figure F.1: ROC Curve of Radboud University Medical Center Team	112
Figure F.2: FROC Curve of Radboud University Medical Center Team	112
Figure G.1: Convolutional Layer 1 Sparsity	113
Figure G.2: Convolutional Layer 2 Sparsity	113
Figure G.3: Fully Connected Layer 1 Sparsity	114
Figure G.4: Fully Connected Layer 2 Sparsity	114
Figure G.5: Cross Entropy Error per Image Plot	115
Figure G.6: Total Loss Plot.....	115
Figure G.7: Convolutional Layer 1 Histogram Plots for Biases, Activations and Weights	116

Figure G.8: Convolutional Layer 2 Histogram Plots for Biases, Activations and Weights.....	117
Figure G.9: Fully Connected Layer 1 Histogram Plots for Biases, Activations and Weights.....	118
Figure G.10: Fully Connected Layer 2 Histogram Plots for Biases, Activations and Weights.....	119
Figure G.11: Softmax Layer Histogram Plots for Biases, Activations and Weights	120

CHAPTER 1

INTRODUCTION

Machine learning technology eases our lives in a wide range of application areas. It provides us with the most relevant content through our web searches, recommends goods according to our interests during online shopping, enables us with automatic picture tagging in social media, presents the opportunity of voice control and speech recognition on mobile devices. Moreover, it is used in the fields of object detection and recognition, natural language processing, in medical applications, in driverless - autonomous cars, etc. In most of these technologies, the techniques used belong to the deep learning class and they are growing in terms of application areas (LeCun, Bengio, & Hinton, 2015).

Furthermore, developments in Information Technologies (IT) transform the workflow in all around the world. Almost all of the paper work becomes digital. Medical institutions are also the ones that are affected from these transformations. Management of patient related record keepings, laboratory test results, radiology images, pathology reports and many other medical instruments become digital.

Pathology laboratories, as a part of medical institutions, are also going through these transformations towards fully digital processes. The processes like tissue sample management, tracing pathology orders, and management of reports have already been digitized. What is occurring recently is the transformation of histopathology slides to digital with the help of whole slide image (WSI) scanners and analyzing them through digital screens.

At this point, there is a chance of aggregating the developments in two different sides of the technology. We can use deep learning techniques to process the digital WSIs and assist the pathologist during their daily work routines.

1.1. Motivation

According to the Global Health Estimates 2014 (GHE, 2014) statistics reported by World Health Organization (WHO), breast cancer is the leading cause of cancer death among women. In the same report, it is declared that low and middle income regions are responsible nearly 62% of the breast cancer deaths in the world. Moreover, according to (IARC, 2013) 52.8% of the incidences of breast cancer cases occurred in less developed regions around the world in 2012. Although the incidence rates show up nearly 5% difference, the death rates differ nearly 24% between more developed and less developed countries.

In the paper (Robboy, et al., 2013), it is declared that weekly average working hours of a pathologist was 49.2 hours and there were 5.7 pathologists per 100.000 people in 2010 in the United States. Moreover, this is estimated to drop down to the 3.7 per 100.000 in 2030. In Turkey, there were 1.56 pathologists per 100.000 people in 2011 (Alper, et al., 2011).

These facts shows us that there is a need to create automatic systems to assist the pathologists such that the time of per slide analysis is shortened and the pathologist can serve more patients in the same amount of time. Therefore, an autonomous system based on deep learning will be introduced to detect and localize the metastases regions in lymph nodes in this study.

1.2. Scope

CAMELYON16 challenge is organized in the context of International Symposium on Biomedical Imaging (ISBI). The goal of this challenge is to evaluate new and existing algorithms for automated detection of metatasis in hematoxylin and eosin (H&E) stained whole-slide images of lymph node sections. This is the first challenge using whole-slide images in histopathology and focuses on sentinel lymph nodes of breast cancer patients.

In the context of this study, a fully automated detection and localization system is developed based on the convolutional neural network architecture. We used the dataset provided in the context of CAMELYON16 challenge to train the network.

The input to the system is whole slide image and the outputs are whole slide image probability of containing metastasis in this image and if the whole slide image contains metastasis regions, representatives for metastasis regions and corresponding probability values.

1.3. Contribution

We have participated to the CAMELYON16 challenge and ranked as fourth team in both categories: slide based evaluation and lesion based evaluation. Hence, we are one of the leading teams that develop a fully automated machine learning architecture to detect and localize metastasis regions in sentinel lymph node sections of breast cancer patients first time in the world.

We have showed that convolutional neural networks can be used in metastasis detection and localization in breast cancer, i.e. metastasis detection can be a new application area for deep learning algorithms. In this study, we have studied with whole slide images which have huge sizes in the order of gigapixels. We have achieved to use such big images in our proposed system.

We have achieved to decrease the classification error rate to nearly 15% over the whole training dataset supplied in CAMELYON16. Moreover, the area under the receiver operating characteristics (ROC) curve (AUC), which is the evaluation criteria of slide based evaluation category, value of 0.9259 and 0.8669 are reached in CAMELYON16 challenge training and test sets, respectively. For the lesion based evaluation category, free-response receiver operating characteristics curve is used. The final score that ranks teams in the second leaderboard is defined as the average sensitivity at 6 predefined false positive rates: 1/4, 1/2, 1, 2, 4, and 8 false positives per whole slide image. In this category we have reached the values of 0.5349 and 0.4060 for training and test sets of CAMELYON16 challenge, respectively.

1.4. Organization

This thesis contains five chapters, namely, introduction, background information, proposed approach, experimental results and conclusion.

In Chapter 1, the motivation that thrills us to work on this topic, fully automated metastasis detection in sentinel lymph nodes of breast cancer patients, is presented. Moreover, the scope of the thesis, contributions made by this work and the organization of the thesis is explained.

In Chapter 2, background information related to both medical side and engineering side of the topic is given. In medical side, statistics and necessary information related to breast cancer around the world is presented. The need of such an autonomous system in digital histopathology field is justified with statistics. In engineering side, a comprehensive chronological overview of the developments in deep learning is summarized. Convolutional neural network structure, which is a special form of deep learning, is explained in a very detailed manner with all of its components. Moreover, performance measures to evaluate the developed system are defined in this chapter.

In Chapter 3, the proposed solution is introduced. First, overview of the system is presented as a whole, and then components of the system are explained one by one in a detailed way.

In Chapter 4, different stages of development process are presented with tabulated data. The process of obtaining final proposed solution through iterations on model is documented.

In Chapter 5, the insights gained during this study and the prospective future work in this area is presented.

CHAPTER 2

BACKGROUND INFORMATION

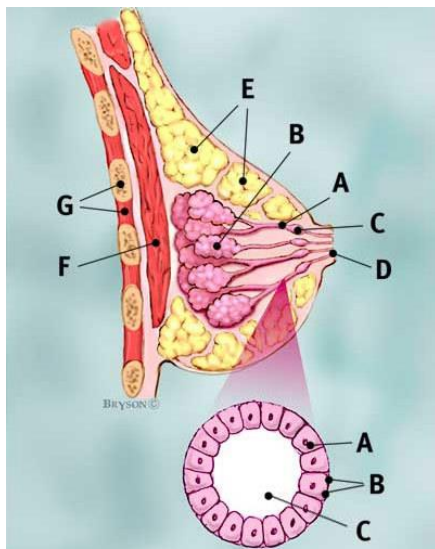
In this chapter, background information related to both medical side and the technical side is presented. In medical side, breast cancer definition, spreading of breast cancer and statistical information in the literature are given. Moreover, developments in digital histopathology are introduced. In technical side, a chronological development of deep learning techniques is presented. More specifically, convolutional neural network architecture with all of its components is explained in a detailed manner.

2.1. Breast Cancer

Breast cancer is an uncontrolled growth of breast cells. The cell division is controlled by the genes that are stored in the nucleus of the cell to sustain the healthy cycle inside the body; however, some mutations may occur inside the nucleus that cause the genes responsible from cell division to be turned off and let some other genes to take control of the division process. At that point, division process goes on in an uncontrolled manner and the same type of cells begins to form tumors (Breastcancer.org, 2015).

Tumors are defined as benign or malignant. Benign tumors are not dangerous to the health and they are not considered as cancerous. They are not so strange in appearance when compared to normal cell structures, growth rate is low and they do not spread beyond the tissues they are originated from. On the other hand, malignant tumors are cancerous. If they are not kept under control, the cells of these types of tumors eventually can spread beyond the original tumor region. This constitutes a risk for whole body. Therefore, breast cancer as a term is used for the malignant tumors that are constituted by the cells in the breast.

In this study, the term breast cancer will be used to refer to female breast cancer. In Figure 2.1, profile of breast is shown: the major parts are the lobules where the milk is produced, ducts through which the milk is transferred to the nipple and the fatty and fibrous connective tissues. Breast cancer mostly starts in the ducts or in the lobules. The cases that start in the other parts of breast are rare issues.



Breast Profile

- A Ducts
- B Lobules
- C Dilated section of duct to hold milk
- D Nipple
- E Fat
- F Pectoralis major muscle
- G Chest wall/rib cage

Enlargement

- A Normal duct cells
- B Basement membrane
- C Lumen (center of duct)

Figure 2.1: Breast profile

2.1.1. How does breast cancer spread?

Malignant tumor cells have the tendency of spreading to nearby healthy breast tissue from where they are originated from, and invade the other parts of the body. The way to travel other parts of the body for the cancer cells go through the lymph system. Since underarm lymph nodes constitute a bridge the other parts of the body, the cancer cells mostly try to reach to the underarm lymph nodes. Therefore, the stage of the breast cancer also refers to the spreading distance of the cancer cells from the original malignant tumor region (Breastcancer.org, 2015).

Lymph system has mainly three components: lymph nodes, lymph vessels and lymph fluid. This system is distributed throughout the whole body and can carry the tissue fluid from one part of the body to the other parts. In this distributed system, the lymph nodes consist of immune system cells and have the shape of beans, and lymph nodes are connected to each other by lymph vessels in which the lymph fluid is carried. Since lymph fluid contains waste products and fluids from tissues, cancer

cells can go into the lymph fluid and locate into the lymph nodes through lymph vessels. Once located, they can grow in the lymph nodes (Cancer.org, 2014). In Figure 2.2, the breast profile with surrounding lymph system components is shown.

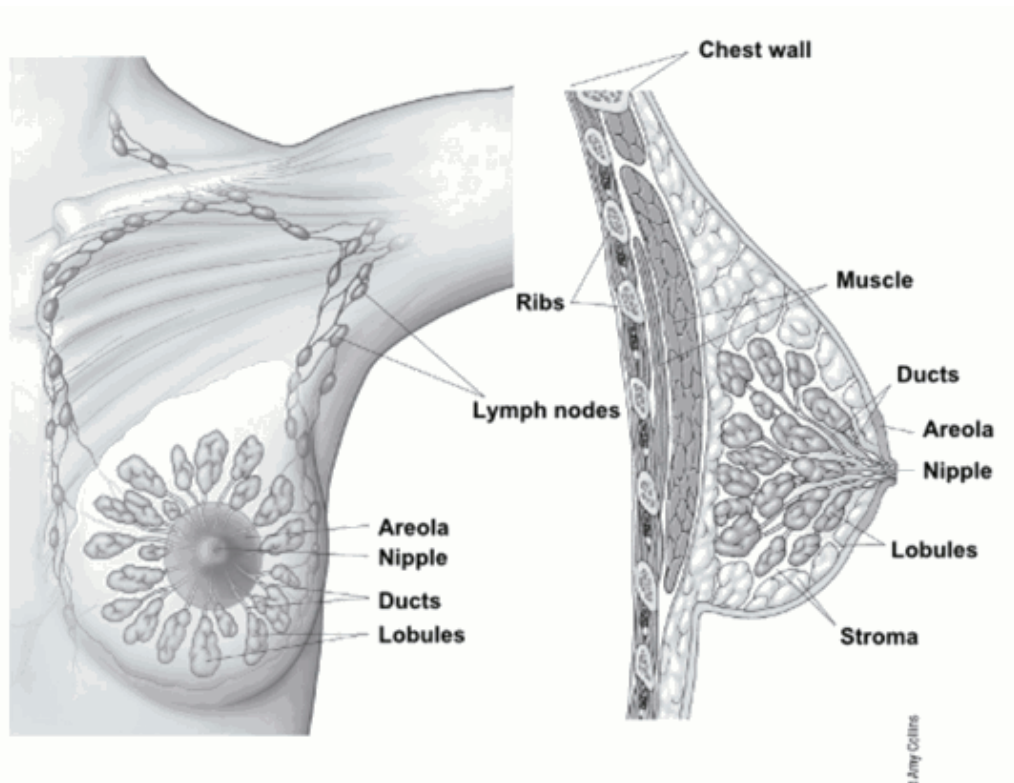


Figure 2.2: Breast Profile with surrounding lymph system components

After reaching to the lymph nodes, the chance of cancer cells to spread (metastasize) to the other part of the body increases. In other words, as the number of affected lymph nodes increases, the probability of the cancer cells spreading to the other parts of the body increases. Therefore, further investigations of the prospective breast cancer patients are done on the samples of breast tumor excisions from lymph nodes.

In this study, the whole slide images (WSIs) of sentinel lymph node biopsy (SLNB) samples are used. “A sentinel lymph node is defined as the first lymph node to which cancer cells are most likely to spread from a primary tumor” (Bejnordi, 2016). Since they are the most probable points that the cancer cells may spread, the excisions are performed on those nodes. In Figure 2.3, sentinel lymph node biopsy is illustrated.

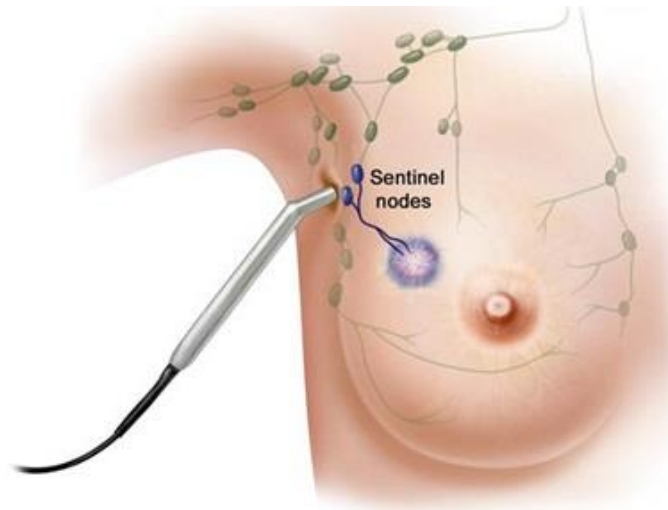


Figure 2.3: Sentinel lymph node biopsy illustration

2.1.2. What are the key statistics about breast cancer?

According to the Global Health Estimates 2014 (GHE 2014) statistics reported by World Health Organization, breast cancer is the leading cause of cancer death among women. In 2012, there were 536.521 women death due to breast cancer in all around the world and a total of 3.544.914 women deaths caused by all cancer types (WHO, 2014). This corresponds to the 15.1% breast cancer death rate among all cancer deaths which makes the breast cancer the top cause in women death due to cancer.

Another important breast cancer statistics is the distribution of the breast cancer deaths among high income (more developed) and low-middle income regions (less developed regions) around the world. According to the GHE 2014, 332.925 of 536.521 breast cancer deaths occurred in low and middle income countries. This constitutes nearly 62% of the breast cancer deaths in the world in 2012, i.e. low and middle income regions are responsible nearly 62% of the breast cancer deaths in the world. Again as it is indicated in the same report, this value was 56.31% in 2000. Moreover, International Agency for Research on Cancer reported estimated cancer incidence, mortality and prevalence worldwide in 2012 in the report of GLOBOCAN 2012 (IARC, 2013). According to this report, there were 1.671.000 breast cancer cases in the world and 883.000 of these cases occurred in less developed regions. This makes 52.8% of the incidences around the world. Although

the incidence rates show up nearly 5% difference, the death rates differ nearly 24% between more developed and less developed countries. Moreover the death rates with respect to the incidences are 25.1% and 36.7% for more developed and less developed regions, respectively.

Both the region based death rates in 2000 and 2012 and region based survival rates among incidence cases show that the more developed regions can cope up with breast cancer better than less developed regions by making use of the comprehensive early detection programs and adequate diagnosis and treatment facilities although the people in more developed regions are more prone to breast cancer due to their lifestyles.

At this point, there is a chance of using the advantages of the technology. With the advents in the digital pathology, it may be possible to increase the survival rates both in more and less developed regions by decreasing the workload of the health professionals and make use of the existing facilities in developed regions by reaching from less developed regions.

2.1.3. Computer-aided Assisting Services

Developments in analog to digital conversion technology have made the slide scanners a promising alternative against the conventional microscopes commonly used by pathologists (Al-Jabani, Huisman, & Van Diest, 2011). By using the slide scanners, whole slide images (WSIs) of glass slides are produced and then can be viewed with the help of the image viewers. WSIs are used for educational purposes, diagnostic purposes and archiving. Since they are constructed at very high physical resolution values (0.50 $\mu\text{m}/\text{pixel}$ and 0.25 $\mu\text{m}/\text{pixel}$ for x20 and x40 magnification values, respectively), the size of the WSIs can be several gigabytes which makes the use of WSIs in daily clinical usages challenging. However, with the advents in compression, storage and transfer technologies, it seems to be commonly used in daily routines.

Digital pathology makes remote pathological meetings, consultations, revisions, diagnosis and computer-aided expert assisting services possible. Although they are

complementary of each other, what is the most important feature among these in the context of this study is the digital structure of WSIs. WSIs' digital structure makes it possible to process them in computers and construct human expert systems to assist the pathologists.

In the paper (Robboy, et al., 2013), it is declared that there were approximately 18.000 pathologists actively working in 2010 in the United States. 75% of those were over 45 years or older and 41% are 55 years or older. Moreover, weekly average working hours of a pathologist is declared as 49.2 hours which is a heavy workload that stealing from their family lives. Another key statistics is that there were 5.7 pathologists per 100.000 people, and this is estimated to drop down to the 3.7 per 100.000 in 2030. When the population growth and the increase in the diseases that require pathological analysis are taken into account, the services given by the medical facilities to their patients re estimated to be negatively affected from these changes.

When compared to the United States, the situation in Turkey in terms of average number of pathologists per 100.000 people is worse. There were 1131 pathologists actively participating to the workload and this correspond to the 1.56 pathologists per 100.000 people (Alper, et al., 2011).

These facts shows us that there is a need to create human expert systems to assist the pathologists such that the time of per slide analysis is shortened and the pathologist can serve more patients in the same amount of time. With the advents of digital pathology and computer technologies, it is possible to create such computer-aided systems to assist the medical experts. Moreover, it may also make contributions to increase the survival rates in less developed regions by the help of both remote diagnosis and consultation.

Based on these observations, the need of such human expert systems is obvious. Therefore, a human expert system based on deep learning will be introduced to detect the metastases regions in lymph nodes in this study.

2.2. Breast Cancer Digital Histopathology

The transformation of histopathology slides to digital with the help of whole slide image (WSI) scanners provides the pathologist with the opportunity of analyzing scanned slides through digital screens. With this transformation, digital histopathology images are aimed to be the primary tool of pathologists instead of optical microscopes (Veta, Pluim, van Diest, & Viergever, 2014).

WSI scanners have been developing rapidly and make the process efficient. They enable to scan whole glass slide at once in 30-60 seconds duration and loading multiple slides in a batch so that they can be scanned consecutively without human intervention. WSI scanners have the ability of scanning at multiple magnification levels up to the resolution of 0.25 $\mu\text{m}/\text{pixel}$ for 40x magnification level. At its highest magnification level a WSI consists of nearly 20 gigapixels and contains 65 gigabyte of raw data. With this ability, it is possible to store images from different magnification levels in a pyramidal structure, which is called as z-stack structure. Z-stack structure provides users (pathologists) with the ability of zoom-in and zoom-out similar to optical microscopes. Z-stack structure is visualized in Figure 2.4. Moreover, example images of a WSI from different magnification levels are shown in Figure 2.5.

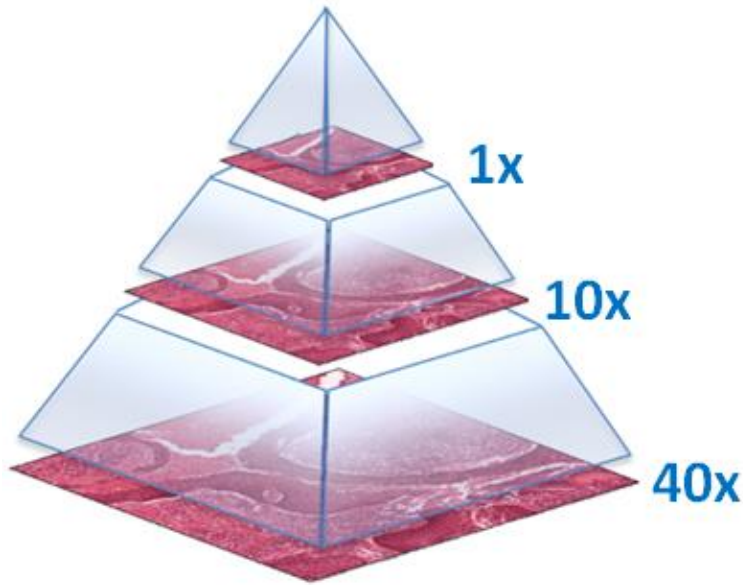


Figure 2.4: Digital histopathology pyramidal image storage: z-stack structure

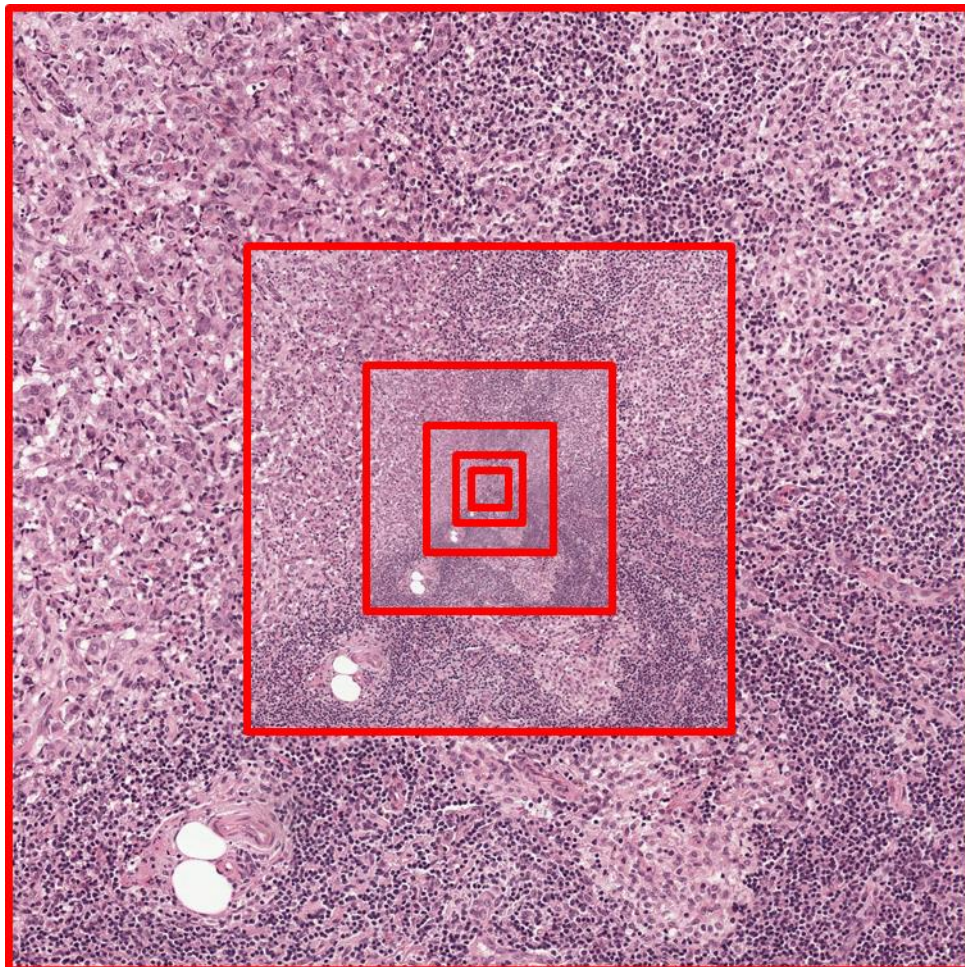


Figure 2.5: Images from different levels of an example WSI

2.2.1. Tissue Preparation and Imaging

Breast cancer histopathology imaging consists of four stages: breast tumor excision, formalin fixation and embedding in paraffin, staining and coverslipping, and slide digitization (Veta, Plum, van Diest, & Viergever, 2014).

i. Breast tumor excisions or biopsies

Breast tumor excisions are performed in operating room and then the sample is sent to the pathology laboratory to be examined. In Figure 2.3, excision operation from a sentinel node is visualized.

ii. Formalin fixation and embedding in paraffin blocks

This is the first step of the tissue preparation. Formalin fixation and embedding in paraffin blocks processes are applied to the tissue samples in order to preserve the structure and cellular details of samples (Kokkat, Patel, McGarvey, LiVolsi, & Baloch, 2013). From the paraffin blocks, nearly 15mm x 15mm sections with a thickness of 3-5 μm are cut and placed on glass slides.

iii. Staining and coverslipping

Mostly, the necessary parts of the tissue for the analysis are nuclei and cytoplasm, but they cannot be viewed easily without staining process. Therefore, the tissue sample is first dyed with stains to highlight nuclei and cytoplasm. Hematoxylin and eosin (H&E) are used in standard staining protocol. While hematoxylin interacts with DNA and dyes nuclei blue/purple, eosin interacts with proteins and dyes cytoplasm pink. After staining process is completed, glass slide is coverslipped and ready to be analyzed by the pathologist in the normal pathology laboratory workflow. However, digital pathology puts one more step onto the standard workflow; slide digitization.

iv. Slide digitization

Slide digitization is the last step of the imaging process. Prepared glass slides are mounted to the WSI scanners and digitized. Currently, commercial WSI scanners make it possible to load multiple slides at once and scan them automatically as a batch. Beyond that by using image processing techniques,

they can differentiate background and tissue sample regions and choose focus points, capture whole image, compress and store data, and register the processed slides to the information system of the laboratory.

2.2.2. CAMELYON16: ISBI Challenge on Cancer Metastasis Detection in Lymph Node WSI Dataset

CAMELYON16 challenge is organized in the context of International Symposium on Biomedical Imaging (ISBI). The goal of this challenge is to evaluate new and existing algorithms for automated detection of metastasis in hematoxylin and eosin (H&E) stained whole-slide images of lymph node sections. This task has a high clinical relevance but requires large amounts of reading time of pathologists. Therefore, a successful solution would hold great promise to reduce the workload of the pathologists while at the same time reduce the subjectivity in diagnosis. This is the first challenge using whole-slide images in histopathology and focuses on sentinel lymph nodes of breast cancer patients.

The data in this challenge contains a total of 400 whole-slide images (WSIs) of sentinel lymph nodes from two independent datasets collected in Radboud University Medical Center (Nijmegen, the Netherlands), and the University Medical Center Utrecht (Utrecht, the Netherlands) (CAMELYON16, 2015). 270 of 400 WSIs is provided as training dataset with ground truth data masks and 130 of them provided as test dataset without ground truth data. In the training dataset, there are 160 normal slides and 110 tumor slides which contain metastasis regions. Naming convention in the provided dataset is as follows: Normal_001 to Normal_160, Tumor_001 to Tumor_110, and Test_001 to Test_130 for normal, tumor and test whole slide images, respectively.

Sub-image samples from normal and tumor WSIs are shown in Figure 2.6, Figure 2.7, Figure 2.8, and Figure 2.9. In Figure 2.6, sub-images of Tumor_110 WSI from different magnification levels are shown with center pixel pointing to the same physical position on the tissue sample. Moreover, the metastasis regions are also marked with blue dotted curves on the sub-images of WSI. Similarly, sub-images of Normal_076 WSI from different magnification levels are shown with center pixel

pointing to the same physical position on the tissue sample in Figure 2.7. In order to see the differences between the normal and tumor WSIs, high resolution sub-images of Tumor_110 and Normal_076 WSIs in a large size are given in Figure 2.8 and Figure 2.9, respectively.

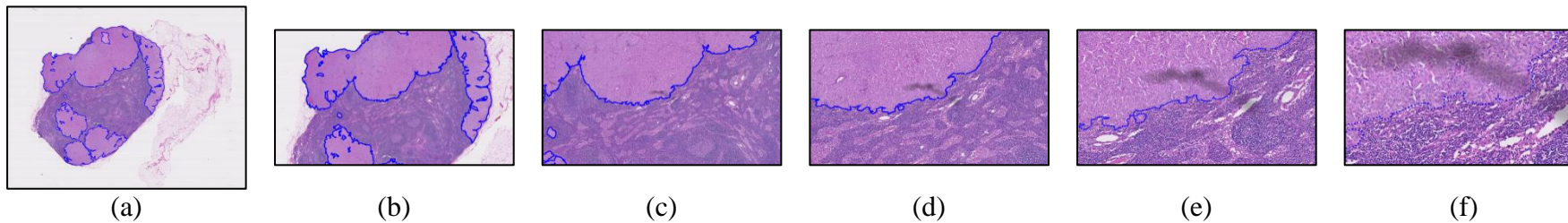


Figure 2.6: Tumor_110 sub-images from different magnification levels on which metastasis regions are enclosed by blue dotted curves: (a) 2944x2240 image at level 5, (b) 3840x2160 image at level 4, (c) 3840x2160 image at level 3, (d) 3840x2160 image at level 2, (e) 3840x2160 image at level 1, (f) 3840x2160 image at level 0

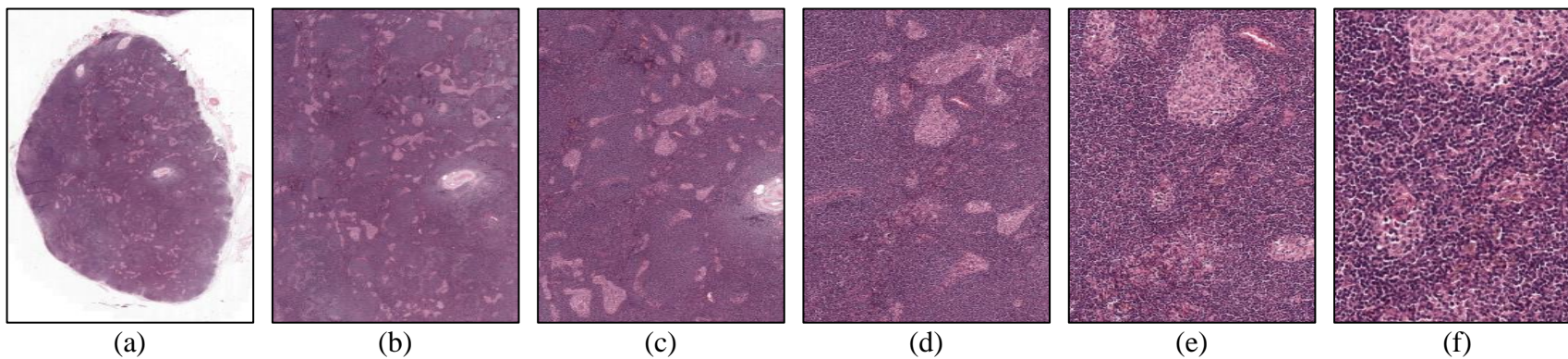


Figure 2.7: Normal_076 sub-images from different magnification levels: (a) 1390x1770 image at level 5, (b) 1390x1770 image at level 4, (c) 1390x1770 image at level 3, (d) 1390x1770 image at level 2, (e) 1390x1770 image at level 1, (f) 1390x1770 image at level 0

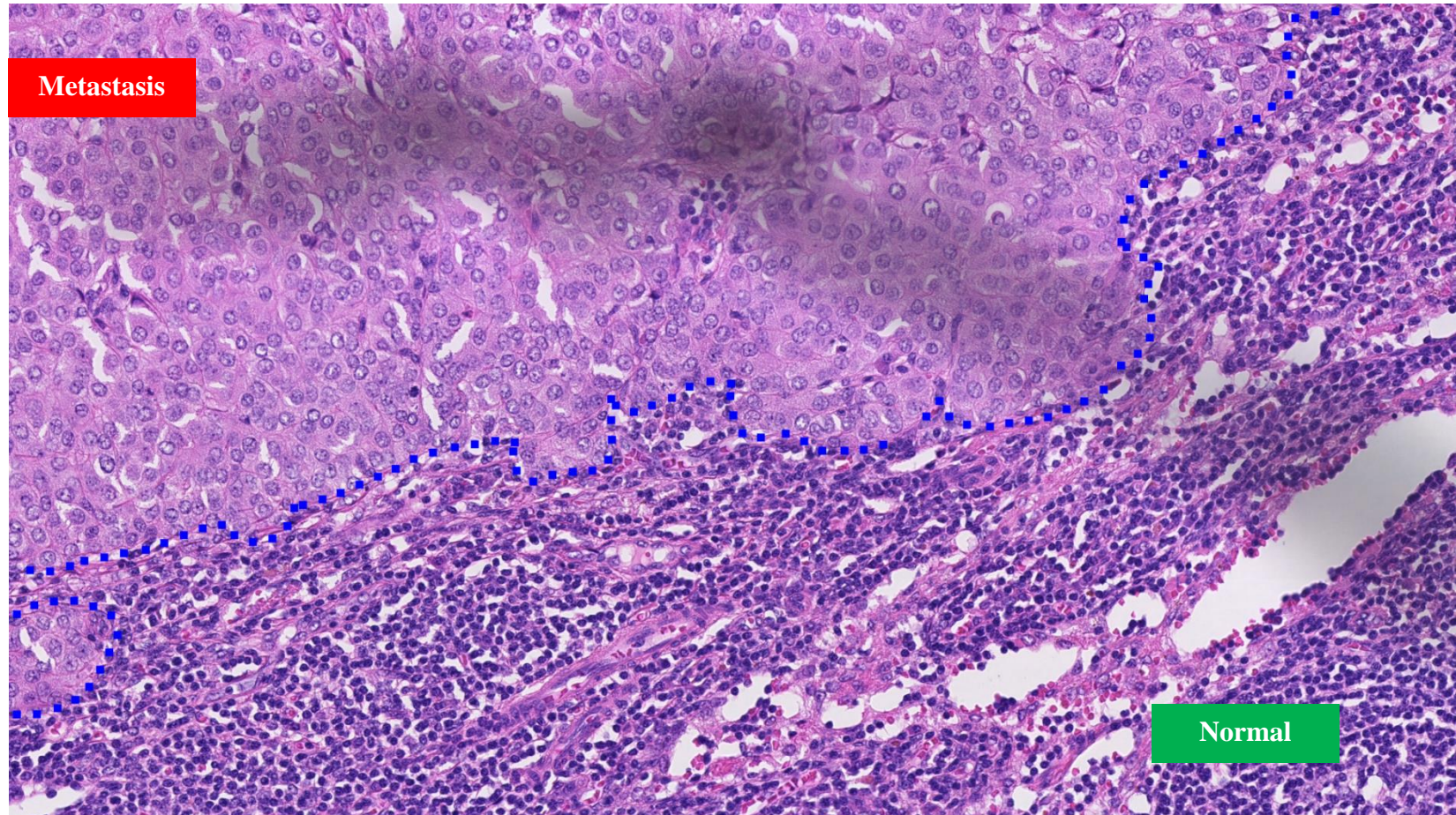


Figure 2.8: Tumor_110 whole slide image - 3840x2160 sub-image from layer 0

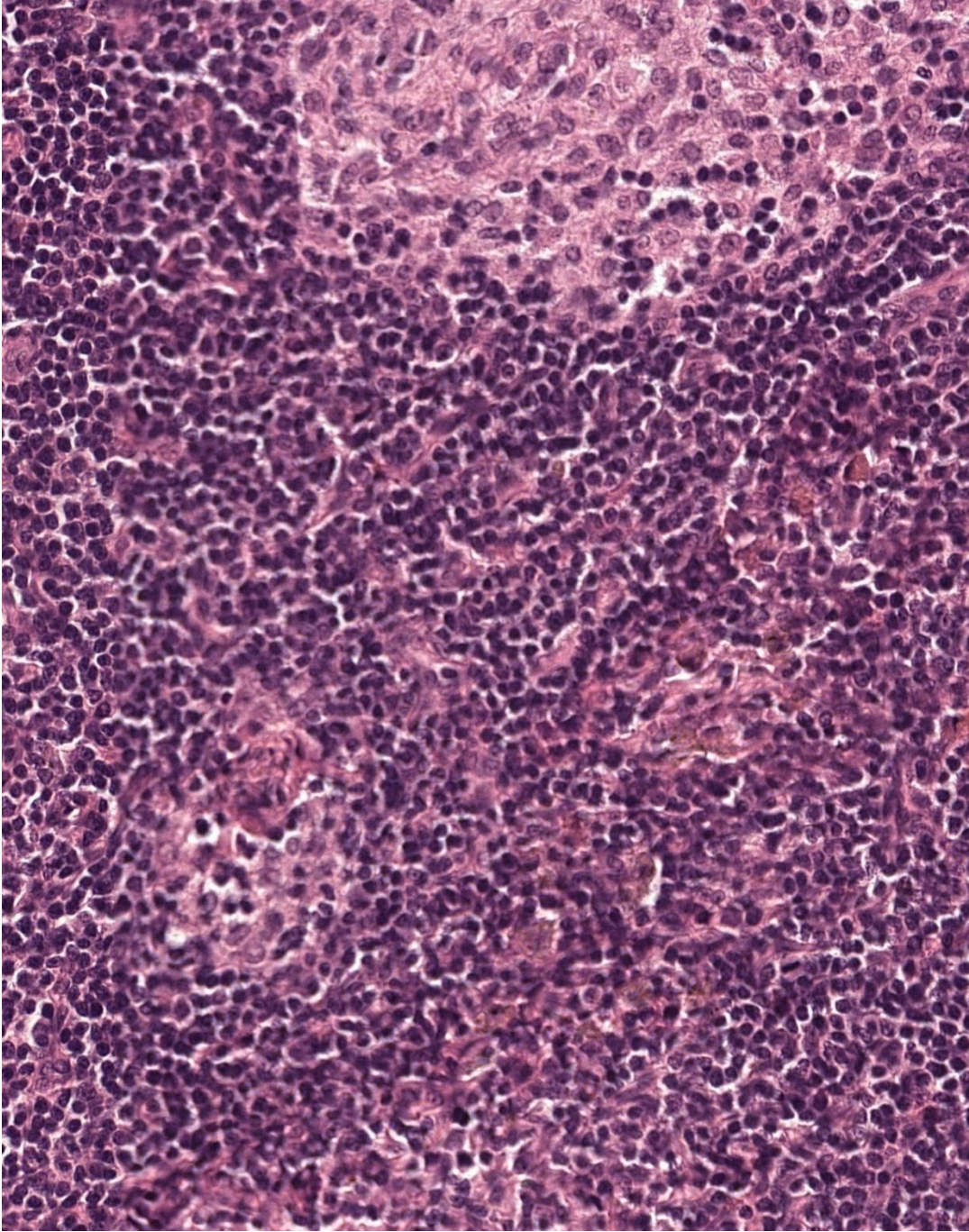


Figure 2.9: Normal_076 whole slide image – 1390x1770 sub-image from layer 0

2.2.3. Related Work on Automated Breast Cancer Assessment

So far, the researchers have concentrated on the individual components of the histological grading of breast cancer separately: assessment of nuclear atypia (nuclei detection and segmentation), tubule formation and mitotic activity. In each of these areas some automated algorithms have been developed and proposed. However, none of them has concentrated on whole slide image level automated analysis of breast cancer assessment yet.

In the field of nuclei detection and segmentation, many different techniques have been proposed. Active contours (Ali & Madabhushi, 2012) and watershed based techniques are popular (Veta, Pluim, van Diest, & Viergever, 2014).

Literature related to the tubule segmentation is not so rich. One of the methods proposed in (Xu, Janowczyk, Chandran, & Madabhushi, 2010) is based on gradient-based geodesic active contour model.

In mitotic figures detection and assessment of proliferation part, many algorithms can be seen in literature. However, the best methods appear to be the convolutional neural network based ones that operate on raw RGB images (Cireşan, Giusti, Gambardella, & Schmidhuber, 2013), (Malon & Cosatto, 2013) and (Malon, et al., 2012). In (Cireşan, Giusti, Gambardella, & Schmidhuber, 2013), there is no initial candidate detection stage, it works on pixel values directly and gives excellent results.

Within the past few years, digital histopathology has been moving towards grand goals with strong potential diagnostic impact: (fully) automated analysis of whole-slide images to detect or grade cancer, to predict prognosis or identify metastases. CAMELYON16 was the first challenge using whole-slide images in histopathology. The list of top ten teams in slide-based evaluation category is shown in Figure 2.10. Proposed approaches of top five ranked teams will be given in this part.















Rank	Team	AUC	Description
01	Harvard Medical School (BIDMC) and Massachusetts Institute of Technology (CSAIL), USA	0.9250	 
02	EXB Research and Development co., Germany	0.9173	 
03	Independent participant, Germany	0.8680	 
04	Middle East Technical University, Departments of EEE, NSNT and HS, Turkey	0.8669	 
05	NLP LOGIX co., USA	0.8332	 
06	University of Toronto, Electrical and Computer Engineering, Canada	0.8181	 
07	The Warwick-QU Team, United Kingdom	0.7999	 
08	Radboud University Medical Center, Diagnostic Image Analysis Group, Netherlands	0.7828	 
09	HTW-BERLIN, Germany	0.7717	
10	University of Toronto, Electrical and Computer Engineering, Canada	0.7666	 

Figure 2.10: Top 10 ranked Teams in CAMELYON16 Slide-based Evaluation Category - <https://grand-challenge.org/site/camelyon16/results/>

Harvard Medical School (BIDMC) and Massachusetts Institute of Technology (CSAIL) joint team from USA was the first ranked team in both of the categories. They tackled the problem by integrating deep learning and traditional machine learning algorithms. They used a deep convolutional neural network with a CNN architecture that was based on the 22-layer GoogLeNet. Then, the trained deep model was applied to partially overlapping patches from each WSI at Layer 0 to create tumor prediction heatmaps. For the slide-based tumor classification task, they extracted a set of geometric features from each tumor probability heatmap in the training set, and trained a random forest classifier to estimate the probability that each slide contained metastatic cancer. They then applied this combined model (CNN and random forest classifier) to the test images to provide a slide-based estimate of the probability of cancer metastases. For the lesion-based tumor region segmentation task, they applied a threshold of 0.90 to the tumor probability heatmaps and predicted tumor location as the center of each predicted tumor region (Wang, Khosla, Gargeya, Irshad, & Beck, 2016). Slide-based and lesion-based performance measure plots and values for the test set are shown in Figure A.1 and Figure A.2 in Appendix A, respectively.

ExB Research and Development team from Germany is the second ranked team in slide-based category and third ranked team in lesion-based category. Their method consists of a tile-wise binary classification based on deep residual networks

(ResNet). Each whole slide image from Layer 0 is splitted into non-overlapping tiles and the tiles are assigned the class tumor if the tile contains tumor pixels, non-tumor otherwise. This setting was used to train a deep neural network on the corresponding binary classification task. Moreover, they normalized input images to have zero mean and unit variance. The architecture of proposed model consists of a 34-layers deep ResNet. They aggregated the results of two different networks as well as applied simple post-processing in order to further increase the performance of their algorithm (Hass, Sanchez, Vasilev, Mey, & Bruni, 2016). Slide-based and lesion-based performance measure plots and values for the test set are shown in Figure B.1 and Figure B.2 in Appendix B, respectively.

The third ranked participant in slide-based category is an individual participant, Quincy Wong, from Germany. VGG-net architecture was used in the proposed solution (Wong, 2016). Used method is not explained in the competition. Slide-based and lesion-based performance measure plots and values for the test set are shown in Figure C.1 and Figure C.2 in Appendix C, respectively.

The forth ranked team in both categories is our team, METU team. The input of our method is whole slide images (WSIs) and outputs are metastasis region representatives together with their probabilities and whole slide probabilities. The method has mainly three steps, which are preprocessing, classification and post processing. Preprocessing was applied on Layer 7 images in order to determine lymph node sections and then only these sections were considered. A training set consisting of 64x64 RGB subimages from Layer 2 of original WSIs was constructed. Negative samples in the training set were selected from slides with label “normal” and positive samples from metastasis regions of slides with label “Tumor”. A CNN having two convolutional layers for feature extraction and two fully connected layer and a softmax for classification was trained with these samples and used for classification of subimages of Layer 2 in the test set. After determining the labels of windowed image regions, we constructed two matrices: one for labels and the other for model outputs, i.e. class probability output of CNN for subimages in windows. In fact, label matrix is a binary image having the same size with Layer 7 images. Then, post-processing operations of elimination of small regions and

confidence filtering were applied and metastasis region representatives were extracted for Evaluation 2 and whole slide probabilities were decided for Evaluation I (Halıcı, Öner, & Çetin Atalay, 2016). Slide-based and lesion-based performance measure plots and values for the test set are shown in Figure D.1 and Figure D.2 in Appendix D, respectively.

The fifth ranked team in slide-based category is NLP LOGIX from USA. To solve the cancer metastasis detection in lymph node challenge, NLP Logix used a seven layer neural network. The first five layers were convolutional and the last two layers were fully connected. A cross entropy cost function was minimized using gradient descent. To train the network, roughly 250,000 image patches were extracted from the labeled slides at Layer 0, each of these image patches were 256 by 256 pixels. The network saw each of the training images approximately ten times. An image patch was labeled as a positive class if more than 50% of the pixels overlapped with the masked region. During training, example patches were rotated and/or flipped randomly. Training and scoring were done using the GPU functionality of Google's TensorFlow machine learning library, running on NVidia graphic cards. The scoring process was distributed across multiple machines. For the localization submission, a combination of watershed and grab-cut algorithms were used to identify pixels that were within likely tumor regions. For the whole slide submission, each of the slides was sliced into eight regions and the Random Forest algorithm was used to assign a probability a slice contained one or more tumor regions. For each of the slides, the slice with the maximum probability was assigned to the slide (Berseth, 2016). Slide-based and lesion-based performance measure plots and values for the test set are shown in Figure E.1 and Figure E.2 in Appendix E, respectively.

Radbound University Medical Center team is the second ranked team in lesion-based category. They implemented two convolutional neural networks to solve the tasks proposed in the ISBI challenge 2016 on cancer metastasis detection in lymph node. For the lesion-based detection task, they proposed a two-step solution. First, a fully convolutional network trained on whole-slide image (WSI) patches from the Challenge dataset was used to generate likelihood maps for tumor areas for each WSI. Second, these likelihood maps were post-processed to locate individual tumor

lesions. For the whole-slide classification task, they applied a second convolutional network to the previously generated likelihood maps, classifying them as tumorous or healthy image samples (Martin, 2016). Slide-based and lesion-based performance measure plots and values for the test set are shown in Figure F.1 and Figure F.2 in Appendix F, respectively.

2.3. Deep Learning and Convolutional Neural Networks

In classical machine learning techniques, to design a machine learning system in order to achieve a specific task, considerable domain expertise and careful engineering study are required. This is because of the fact that the conventional machine learning systems are not capable of processing natural data in their raw format. For example, it may be required to extract features such as edges or blobs for images rather than using the raw pixel values directly. After a hard feature extraction study, it is decided what to use as feature vector. Then, feature vector is supplied to a classifier for pattern recognition on the input.

On the other hand, deep learning methods are representation learning methods in which the machines can be supplied directly with raw data and discover the representations required for detection or classification automatically (LeCun, Bengio, & Hinton, 2015). In deep learning methods, there are a stack of representation layers those are produced by simple non-linear modules that transform the output of one layer at their input to a more abstract level of representation at their outputs. In these methods, raw data is supplied at the input of the system and goes through simple non-linear module transformations. By passing through enough number of those transformations, i.e. stacking more layers on top of each other; it is possible to learn very complex functions.

If we supply an image as an input to such a system, raw pixel values will be the input of the first layer, and the presence or absence of edges at particular orientations and locations will be represented by the features learned by the first layer of representation. In the second layer, motifs constructed by particular positioning of edges are detected regardless of small variations in the edge positions. Motifs may be assembled into different combinations corresponding to the parts of some objects

in the third layer and the objects as combinations of these parts may be detected at upper layers. “The key aspect of deep learning is that these layers of features are not designed by human engineers; they are learned from data using a general purpose learning procedure (LeCun, Bengio, & Hinton, 2015).

From the very beginning of the machine learning/pattern recognition studies in 1957 and 1958, (Rosenblatt, 1957) and (Selfridge, 1958), researchers have been looking for trainable networks in order to replace hand-engineered features. It was the time that it was understood clearly by the statement of backpropagation method in 1986 (Rumelhart, Hinton, & Williams, 1986). The backpropagation is an application of the chain rule for derivatives on the multilayer stack of modules which is called as multi-layer perceptron (MLP), or regular networks. The gradient of an objective function can be calculated with respect to output of a module and it can also be computed with respect to input of the module by using the relation of input and output in this module. Similarly, it is also possible back-propagate the gradients through top of the network (output of the network) to the bottom of the network (input to the network) by using chain rule. Once calculated these gradients, the gradients with respect to weights of the network can also be computed. This constitutes the base of the learning procedure in deep learning architectures.

Although the fundamentals of deep learning methods were constructed in 1980s and 1990s, they could not been so popular and widely used until 2006 (LeCun, et al., 1990), (LeCun, Bottou, Bengio, & Haffner, 1998) and (Hinton, Osindero, & Teh, 2006). In 2006, the researchers introduced and used unsupervised learning procedures to create layers of feature detectors without requiring labeled data for Restricted Boltzmann Machine . By using unsupervised learning procedures, they could initialize the weights of the networks to sensible values and add output layer on top of previously initialized network and fine-tune all network with standard backpropagation method by using limited number of labeled data.

There are two major developments that make deep learning methods applicable and state-of-the-art technologies in the last decade: the availability of powerful computers and the availability of large datasets (LeCun, The AI Arms Race, 2016).

With the development of graphics processing units (GPUs), it became possible to train networks 10x or 20x times faster. This enables researchers to train very large and deep architectures in a limited time. Availability of large datasets comes from the developments in the information technology and infrastructure. It becomes possible to store all the data related to a digital process, for example, the trace of a user on a website, the interests of social media users through their sharings or profiles. With the availability of large datasets it becomes possible to train deep architectures in a supervised manner easily, so leads to state-of-the-art achievements in the literature.

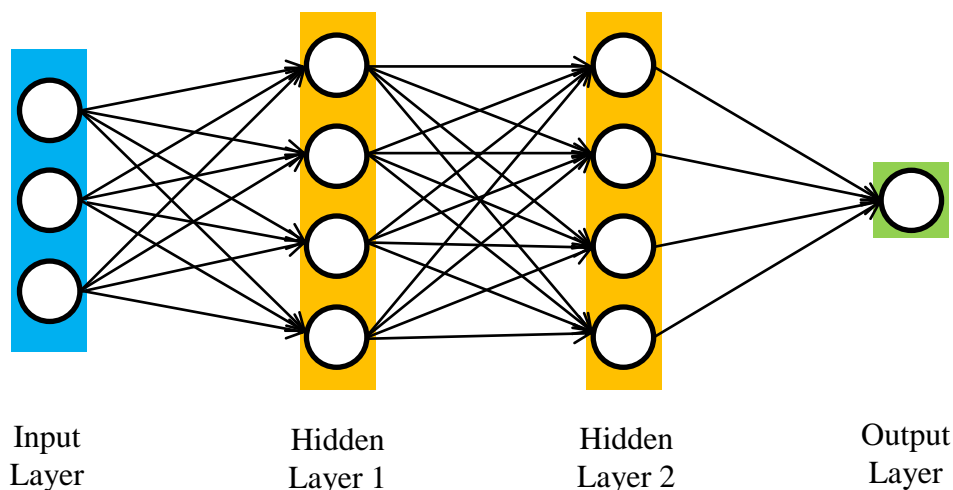
2.3.1. Convolutional Neural Networks Architecture Overview

A specific type of deep feedforward architecture that is much easier to train and has better generalization ability than networks with fully connected layers is the convolutional neural networks (CNNs) (LeCun, et al., 1990), (LeCun, Bottou, Bengio, & Haffner, 1998).

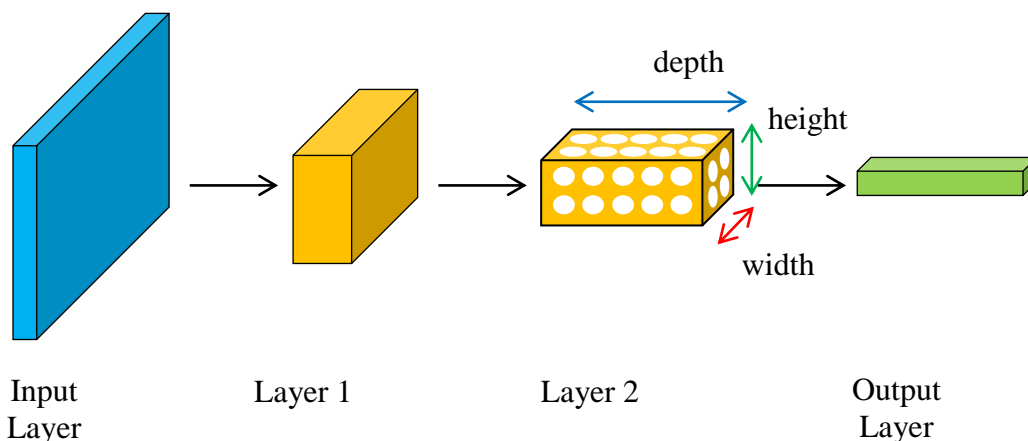
Similar to regular neural networks, CNNs are also made up of layers. Different than regular networks, the form and functions of the layers are different. In regular networks, layers are one dimensional and neurons in these layers are fully connected. On the other hand, the form of layers in CNNs is usually three dimensional: width, height and depth (Karpathy, 2016). Such an input preserves neighborhood relation in data which is important for natural signals like images. In Figure 2.11, general architectures of regular neural networks (MLP) and convolutional neural networks are shown.

Convolutional neural networks are designed to be fed by data that is organized in the form of multiple arrays which provides the CNN structure with special advantages. When we consider an RGB image as input to the network, there are four essential properties of convolutional neural networks that enable them to exploit the properties of the input images: local connections, shared weights, pooling and many layer stack structure (LeCun, Bengio, & Hinton, Deep Learning, 2015). First three of these properties will be explained in the following sections.

Many layer stack structure provides CNNs with that different features from raw data can be extracted autonomously in CNNs' representation layers. If the input is image pixel values to a CNN, most likely the edge properties will be extracted in the first convolutional layer, motifs in the second layer, sub-parts in the third layer, parts in the fourth layer and it can go up to 20 layers in these days. This property of CNNs makes them capable to extract very complex patterns inside the data array.



(a) Regular Neural Network Structure



(b) Convolutional Neural Network Structure

Figure 2.11: a) Regular Neural Network Structure, b) Convolutional Neural Network Structure

Inputs to the CNNs are usually three dimensional RGB images and the output layer gives the class scores of the input images. Layers between the input and output

layers can be convolutional layers, pooling layers, normalization layers and fully connected layers. These layers are stacked on top of each other to construct the CNN structure.

2.3.2. Convolutional Layer

This layer type is the core of CNN structure. A convolutional layer consists of multiple feature maps (or *depth slices*) which consist of the collection of neurons. The neurons in a feature map are similar to the neurons that are used in regular neural networks except that the neurons in a feature map of a convolutional layer are not fully connected to the neurons of the previous layer, i.e. *local connections* exist between the layers. Activation of a neuron is illustrated in Figure 2.12. The activation value of the neuron is go through a non-linear transformation and computes the output value of the neuron. Usually, Rectified linear units (RELU – $\max(0,x)$) are used as activation functions since they are more efficient in terms of training time (Krizhevsky, Sutskever, & Hinton, 2012).

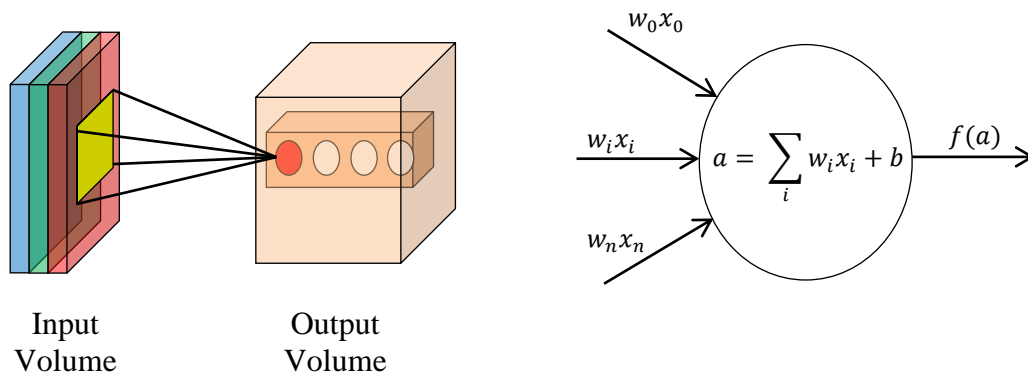


Figure 2.12: Neuron Activation Scheme

Trainable filters of the convolutional layer link the neurons of that layer to the neurons of previous layer. The size of the filter is actually the *receptive field* of a neuron and it determines the connections of the neuron from previous layer. Moreover, a filter is shared through a feature map of a convolutional layer and this is called as *parameter (or weight) sharing* which is one of the advantages of CNNs because such a property decreases the number of parameters.

How to apply convolution operation by using filters of the layer is a matter of *spatial arrangement* in that layer. Size of filters, size of zero padding, size of stride and number of filters are the hyperparameters that shape the output volume of the convolutional layer.

Local connections

When the characteristics of the natural signals are considered, it can be easily seen that the localization is an important aspect. For example, the edges in an image may form a motif at some local parts or the words and the sentences that come one after the other in language make sense for human beings. Hence, localization of features is a distinctive property in natural signal data structures and they can be captured by local connections in many layer stack structure.

In convolutional layer, the units in a feature map are connected to a local region in the previous layer with a set of weights which is called as filter. The localization of connectivity is defined in width and height dimensions, i.e. each filter affects to a local region in width and height dimensions but it uses all the depth slices. Therefore, the size of the filter is defined as receptive field of a neuron. In other words, connection region of a neuron in the input volume (in previous layer) defined by the filter size along width and height and it always goes over all the depth slices.

The three different colored layers (red, green and blue) in Figure 2.12 correspond to the feature maps of the input volume. The highlighted region on these layers is the receptive field of the neuron that is in red color in the output volume. All the neurons in the receptive field of the red neuron in the input volume through all depth slices are connected to red neuron through filter weights. Together with bias term the weighted sum of neuron states in the receptive of input volume determine the state of the red neuron in the output volume. If we have a $48 \times 48 \times 3$ input volume (RGB image) and 5×5 filter size (receptive field), neuron will have $5 \times 5 \times 3$ connection through the filter, i.e. 75 weights and a bias parameter. On the other hand, if this was a fully connected layer, the same neuron would have 6912 ($48 \times 48 \times 3$) connections, so weights, and a bias. CNN structure reduces drastically the number of weights by exploiting the local connectivity structure of inputs.

Parameter (or Weight) sharing

Parameters in a feature map of a representation layer are shared by all units in that feature map. That provides the structure with the ability of revealing similar motifs in a feature map at different local regions, i.e. distinctive local features that emerge at different parts of the input can be detected. Moreover, the weights in different feature maps of a convolutional layer are different in order to detect different motifs over the input.

In Figure 2.13, red and green neurons are in the same depth slice while the purple neuron is in another depth slice. This means that red and green neurons use the same filter, so sharing the weights. However, purple neuron will use different connection weights. There are two highlighted regions on the input volume. Red and green neurons are connected to the corresponding regions, and purple neuron is connected to the same region with the red neuron. The difference of the filters used by red and purple neurons is illustrated by different colored connection lines: yellow and black for purple and red neurons, respectively. Moreover, the connection lines of the green neuron are also drawn in black to emphasize the sharing of weights with red neuron in the same depth slice.

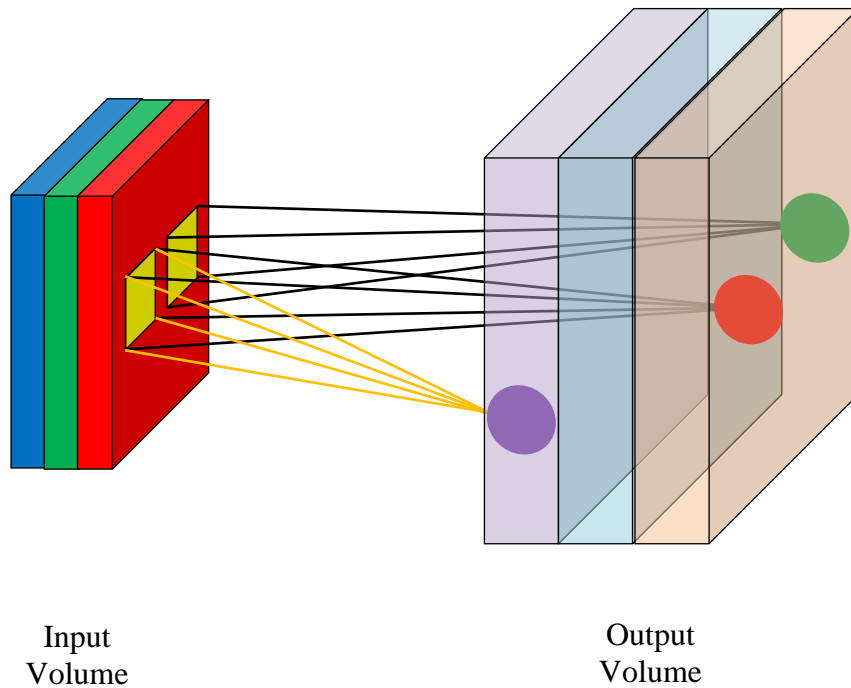


Figure 2.13: Weight Sharing in a Depth Slice or Feature Map

If we continue to explain the benefits of weight sharing on the same example in local connections part, we have $48 \times 48 \times 3$ input volume and $5 \times 5 \times 3$ filters. Moreover, assume that the output volume size is $48 \times 48 \times 64$. In Table 2.1, the comparison of convolutional layer and fully connected layer structure is shown with the assumption that both of the layer types have the same number of neurons in their input and output volumes. There are 6912 neurons in the input volume and 147456 neurons in the output volume for both of the layer types. However, these neurons in the convolutional layer are organized in a stack of 64 feature maps each of which contains 2304 neurons. As we have mentioned previously, convolutional layers employ a local connection scheme rather than using fully-connected scheme. Due to this property of convolutional layer, there are only 76 parameters (75 for connection weights and 1 for bias term) of a neuron in the output volume of convolutional layer. On the other hand, there are 6913 parameters (6912 for connection weights and 1 for bias term) dedicated to single neuron in the output volume of a fully-connected layer. Furthermore, weight sharing property of convolutional layer drastically decreases the number of parameters used by neurons in a feature map when compared with the number of parameters used by equal number of neurons in fully-connected layer. 2304 neurons in a feature map in the convolutional layer share the

parameters and use only 76 parameters; however, 2304 neurons in fully connected layer use nearly 16 million parameters. The number of parameters for entire layers sum up to the values of nearly 5 thousand for convolutional layer and 1 billion for fully-connected layer. In terms of memory usage, there is no difference between convolutional layer and fully-connected layer since in both of them one unit of memory is required to store activation value of a neuron.

Table 2.1: Convolutional Layer and Fully-Connected Layer Comparison

		Convolutional Layer	Fully-Connected Layer
Input Volume (48x48x3)	Number of neurons	6.912	6.912
	Number of neurons in a feature map	48x48 = 2.304	-
Output Volume (48x48x64)	Number of feature maps	64	-
	Total number of neurons	147.456	147.456
	Filter Size	5x5x3	-
Layer Properties	Parameters used by a single neuron (Connection weights + 1 bias)	75+1=76	6.912+1=6.913
	Parameters used by neurons in a feature map	76	6.913 x 2.304=15.927.552 ≈ 16M
	Parameters used by layer	76 x 64=4.864	15.927.552 x 64=1.019.363.328 ≈ 1B
	Memory usage for output	147.456	147.456

We have seen that the neurons in the same depth slice in CNN use the same set of weights; it means that the activation values of the neurons in each depth slice can be computed as a convolution operation and addition of bias term, so this is where the name of convolutional layer derives from. Moreover convolution operation is nothing but the filtering operation, so the set of connection weights is called as filter or kernel.

Spatial Arrangement

We have analyzed the connection and weight properties of a neuron in a feature map of output layer; however, we have not seen yet the parameters that affect the shape of the output volume, i.e. the size of feature maps and the depth of the output volume. There are actually three hyperparameters that affect the size of the output volume: number of filters, stride and zero-padding.

The connections of a neuron is determined by the receptive field of that neuron, i.e. filter size that used in the feature map that the neuron resides on, in width and height dimensions and they go through all the feature map layers in the input volume. Since each neuron in the same feature map of output volume uses the same filter, the number of filters used in a convolutional layer determines the number of feature maps in the output volume, so the depth of the output volume. This structure ensures that the neurons at different depth slices can detect different motifs at the same spatial point of input in the width and height dimensions.

The second hyperparameter is the stride value. Stride value is defined as the number of pixels that the filter is shifted at each step during the convolution operation. This value determines the size of the output volume in width and height dimensions. If the stride value is 1, it means that the filter is shifted 1 pixel at each step of convolution operation. If the stride value is 2, the filter is shifted by 2 pixels and this goes on like that. The importance of the stride value is that as the value increases the size of the output volume decreases.

The last hyperparameter is zero-padding operation at the boundaries of the input volume. This hyperparameter is used for to control the size of the output volume. The value of the zero-padding is chosen in such a way that the size of the input volume is preserved in the output volume in terms of width and height dimensions. If we use 3x3 filters, one pixel zero padding (with the stride value of 1) assures us the sizes of input and output volumes will be the same. Zero-padding value should be 2 for the filter size of 5x5 for that purpose.

In Figure 2.14, hyperparameters of the convolutional layer is visualized to derive some fundamental relations between hyperparameters for the proper convolution operation. Note that the input volume and filter size height and width dimensions are assumed to be equal, i.e. square inputs and filters.

- W : Input volume size (width and height)
- D_1 : Input volume depth size (not shown on the figure)
- D_2 : Output volume depth size (not shown on the figure)
- F : Filter size
- K : Number of filters (not shown on the figure)
- S : Stride
- P : Padding
- N : Number of shift steps in one dimension (must be integer)

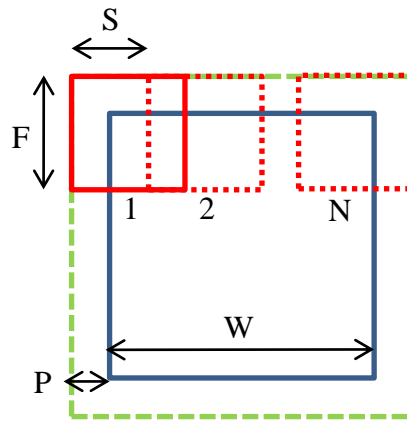


Figure 2.14: Visualization of Convolutional Layer Hyperparameters

We know that the filter is shifted by S units in each step, so the total distance traveled in one row of input volume can be calculated as in Equation (2.1). For proper convolution operation we should choose the hyperparameter values appropriately. For example, if the input volume is an RGB image of size $48 \times 48 \times 3$, filter size is $5 \times 5 \times 3$ and stride value is 1, we can use or not zero-padding for proper operation. P must be chosen as 2 in order to preserve the input volume size of 48×48 at the output. On another configuration, if we decide to use stride value 2 and padding 0, we must shift the filter 22.5 steps which is not possible and this will give erroneous results during learning.

$$(N - 1) * S = W + 2P - F$$

$$N = \frac{W + 2P - F}{S} + 1 \tag{2.1}$$

Most of the times, the general convention is to choose the stride value of 1 and preserving the input volume throughout the convolution operation. In other words, the number of filter shifting steps must be equals to the input volume size. For such a construction padding value must be chosen according to the Equation (2.2).

$$\begin{aligned}
 N &= \frac{W+2P-F}{s} + 1 \\
 W &= W + 2P - F + 1 \\
 P &= \frac{F-1}{2}
 \end{aligned}
 \tag{2.2}$$

Dimensions of the output volume are also determined by the Equation (2.1). N is also the dimension value for the width and height of the output volume. Depth value equals to the value of number of filters which is K. In Table 2.2, the hyperparameters relations in a convolutional layer are summarized.

Table 2.2: Convolutional Layer Hyperparameters Relation Summary

Input volume size	$W \times W \times D_1$
Filter size	$F \times F \times D_1$
Number of filters	K
Stride value	S
Zero-Padding	P
Output volume size	$\left(\frac{W+2P-F}{s} + 1\right) \times \left(\frac{W+2P-F}{s} + 1\right) \times K$
Number of weights per filter	$F \times F \times D_1$
Total number of weights	$(F \times F \times D_1) \times K$
Number of biases	K

The convolution operation can be explained as follows: If we want to compute the activation value of a neuron located at the position of (x_2, y_2) in the d^{th} depth slice of the output volume and the corresponding receptive field center coincides to the location of (x_1, y_1) in the input volume which is visualized in Figure 2.15, the convolution operation will be the dot product (‘*’ symbol is used for dot product) of

input volume and the filter weights and addition of bias term which is shown in Equation (2.3).

$$O[x_2, y_2, d] = X \left[x_1 - \frac{F-1}{2} : x_1 + \frac{F-1}{2}, y_1 - \frac{F-1}{2} : y_1 + \frac{F-1}{2}, : \right] * W_d[:, :, :] + B_d \quad (2.3)$$

- O : Output volume activation values
- X : Input volume values
- W_d : Weights of d^{th} filter
- B_d : d^{th} bias

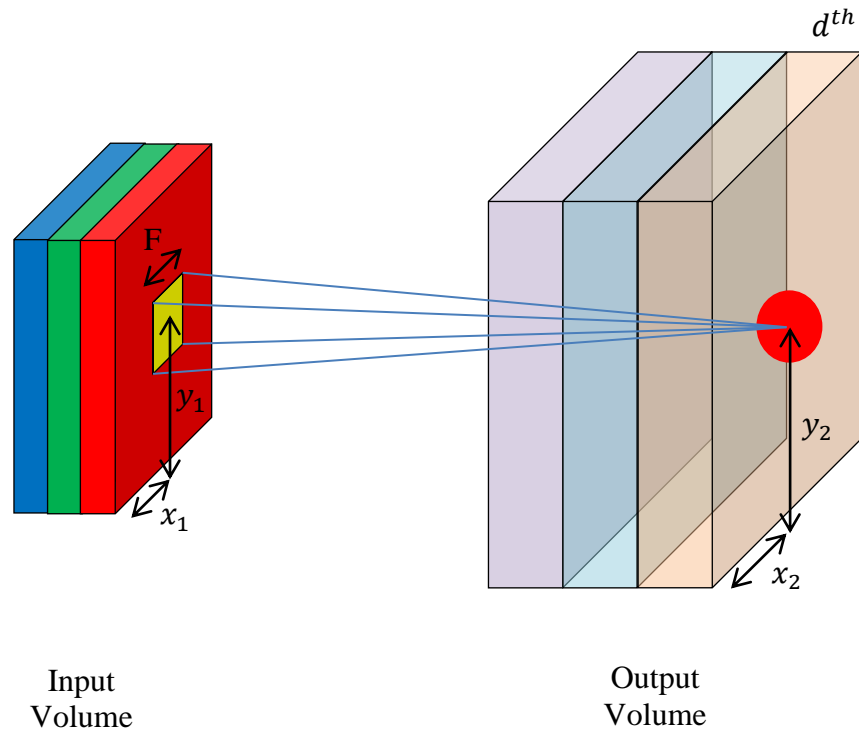


Figure 2.15: Visualization of Elements in Convolution Operation

The application of convolution operation is shown on an example setup that has 7x7x3 input volume size (5x5x3 input + 1 zero-padding), 2 filters with the size of 3x3x3 and stride of 1. Step by step computation of output activation value for $O[0,0,0]$ is shown below. Moreover, the active parts of input volume, filters, biases and output volume for the convolution operations to compute the values of $O[0,0,0]$, $O[1,0,0]$, $O[2,0,0]$, $O[2,2,0]$ and $O[3,3,1]$ are shown in Table 2.3, Table 2.4, Table 2.5, Table 2.6 and Table 2.7, respectively.

$$o[0,0,0] = \mathbf{X}[0:2,0:2,:] * \mathbf{W}_0 + \mathbf{b}_0$$

$$o[0,0,0] = x[0:2,0:2,0] * w_0[:,0] + \\ x[0:2,0:2,1] * w_0[:,1] + \\ x[0:2,0:2,2] * w_0[:,2] + 1$$

$$o[0,0,0] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 2 & 2 \end{bmatrix} * \begin{bmatrix} 1 & 1 & -1 \\ 0 & -1 & 0 \\ 0 & 1 & 0 \end{bmatrix} + \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 2 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} + \\ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 1 & -1 \\ -1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} + 1$$

$$o[0,0,0] = 0 + 0 + 2 + 1$$

$$o[0,0,0] = 3$$

Table 2.3: Convolution Operation to Compute the Output Activation Value of O[0,0,0]

Input Volume + 1 Padding (7x7x3)	Filter W_0 (3x3x3)	Filter W_1 (3x3x3)	Output Volume O (3x3x2)																																																																												
$x[:, :, 0]$ <table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>2</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>2</td><td>2</td><td>2</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>2</td><td>2</td><td>0</td><td>0</td><td>1</td><td>0</td></tr> <tr><td>0</td><td>2</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0	0	0	0	0	2	1	1	1	0	0	0	2	2	2	0	0	0	0	1	1	0	1	0	0	0	2	2	0	0	1	0	0	2	0	1	1	0	0	0	0	0	0	0	0	0	$w_0[:, :, 0]$ <table border="1"> <tr><td>1</td><td>1</td><td>-1</td></tr> <tr><td>0</td><td>-1</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>0</td></tr> </table>	1	1	-1	0	-1	0	0	1	0	$w_1[:, :, 0]$ <table border="1"> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>-1</td></tr> <tr><td>1</td><td>1</td><td>-1</td></tr> </table>	1	0	0	0	0	-1	1	1	-1	$o[:, :, 0]$ <table border="1"> <tr><td>3</td><td>4</td><td>-1</td></tr> <tr><td>6</td><td>4</td><td>2</td></tr> <tr><td>-3</td><td>8</td><td>1</td></tr> </table>	3	4	-1	6	4	2	-3	8	1
0	0	0	0	0	0	0																																																																									
0	2	1	1	1	0	0																																																																									
0	2	2	2	0	0	0																																																																									
0	1	1	0	1	0	0																																																																									
0	2	2	0	0	1	0																																																																									
0	2	0	1	1	0	0																																																																									
0	0	0	0	0	0	0																																																																									
1	1	-1																																																																													
0	-1	0																																																																													
0	1	0																																																																													
1	0	0																																																																													
0	0	-1																																																																													
1	1	-1																																																																													
3	4	-1																																																																													
6	4	2																																																																													
-3	8	1																																																																													
$x[:, :, 1]$ <table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>2</td><td>1</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>2</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>2</td><td>1</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>2</td><td>0</td><td>1</td><td>1</td><td>0</td></tr> <tr><td>0</td><td>2</td><td>0</td><td>1</td><td>2</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0	0	0	0	0	0	0	2	1	0	0	0	2	0	1	1	0	0	0	0	0	2	1	0	0	0	1	2	0	1	1	0	0	2	0	1	2	0	0	0	0	0	0	0	0	0	$w_0[:, :, 1]$ <table border="1"> <tr><td>1</td><td>0</td><td>-1</td></tr> <tr><td>0</td><td>0</td><td>1</td></tr> <tr><td>-1</td><td>0</td><td>1</td></tr> </table>	1	0	-1	0	0	1	-1	0	1	$w_1[:, :, 1]$ <table border="1"> <tr><td>1</td><td>0</td><td>-1</td></tr> <tr><td>0</td><td>-1</td><td>1</td></tr> <tr><td>0</td><td>0</td><td>-1</td></tr> </table>	1	0	-1	0	-1	1	0	0	-1	$o[:, :, 1]$ <table border="1"> <tr><td>-3</td><td>-4</td><td>-1</td></tr> <tr><td>-5</td><td>-3</td><td>-2</td></tr> <tr><td>-4</td><td>-2</td><td>-2</td></tr> </table>	-3	-4	-1	-5	-3	-2	-4	-2	-2
0	0	0	0	0	0	0																																																																									
0	0	0	2	1	0	0																																																																									
0	2	0	1	1	0	0																																																																									
0	0	0	2	1	0	0																																																																									
0	1	2	0	1	1	0																																																																									
0	2	0	1	2	0	0																																																																									
0	0	0	0	0	0	0																																																																									
1	0	-1																																																																													
0	0	1																																																																													
-1	0	1																																																																													
1	0	-1																																																																													
0	-1	1																																																																													
0	0	-1																																																																													
-3	-4	-1																																																																													
-5	-3	-2																																																																													
-4	-2	-2																																																																													
$x[:, :, 2]$ <table border="1"> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>2</td><td>1</td><td>1</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>2</td><td>1</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1</td><td>2</td><td>1</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </table>	0	0	0	0	0	0	0	0	0	2	1	1	0	0	0	0	0	0	1	1	0	0	0	1	1	2	1	0	0	0	1	1	0	0	0	0	0	1	1	2	1	0	0	0	0	0	0	0	0	$w_0[:, :, 2]$ <table border="1"> <tr><td>1</td><td>1</td><td>-1</td></tr> <tr><td>-1</td><td>0</td><td>1</td></tr> <tr><td>0</td><td>0</td><td>1</td></tr> </table>	1	1	-1	-1	0	1	0	0	1	$w_1[:, :, 2]$ <table border="1"> <tr><td>0</td><td>-1</td><td>1</td></tr> <tr><td>-1</td><td>-1</td><td>-1</td></tr> <tr><td>0</td><td>0</td><td>-1</td></tr> </table>	0	-1	1	-1	-1	-1	0	0	-1										
0	0	0	0	0	0	0																																																																									
0	0	2	1	1	0	0																																																																									
0	0	0	0	1	1	0																																																																									
0	0	1	1	2	1	0																																																																									
0	0	1	1	0	0	0																																																																									
0	0	1	1	2	1	0																																																																									
0	0	0	0	0	0	0																																																																									
1	1	-1																																																																													
-1	0	1																																																																													
0	0	1																																																																													
0	-1	1																																																																													
-1	-1	-1																																																																													
0	0	-1																																																																													
	Bias B_0 (1x1x1) $b_0[:, :, 0]$ <table border="1"> <tr><td>1</td></tr> </table>	1	Bias B_1 (1x1x1) $b_1[:, :, 0]$ <table border="1"> <tr><td>0</td></tr> </table>	0																																																																											
1																																																																															
0																																																																															

Table 2.4: Convolution Operation to Compute the Output Activation Value of $O[1,0,0]$

Input Volume + 1 Padding (7x7x3) $x[:, :, 0]$							Filter W_0 (3x3x3) $w_0[:, :, 0]$			Filter W_1 (3x3x3) $w_1[:, :, 0]$			Output Volume O (3x3x2) $o[:, :, 0]$		
0	0	0	0	0	0	0	1	1	-1	1	0	0	3	4	-1
0	2	1	1	1	0	0	0	-1	0	0	0	-1	6	4	2
0	2	2	2	0	0	0	0	1	0	1	1	-1	-3	8	1
0	1	1	0	1	0	0									
0	2	2	0	0	1	0									
0	2	0	1	1	0	0									
0	0	0	0	0	0	0									
$x[:, :, 1]$							$w_0[:, :, 1]$			$w_1[:, :, 1]$			$o[:, :, 1]$		
0	0	0	0	0	0	0	1	0	-1	1	0	-1	-3	-4	-1
0	0	0	2	1	0	0	0	0	1	0	-1	1	-5	-3	-2
0	2	0	1	1	0	0	-1	0	1	0	0	-1	-4	-2	-2
0	0	0	2	1	0	0									
0	1	2	0	1	1	0									
0	2	0	1	2	0	0									
0	0	0	0	0	0	0									
$x[:, :, 2]$							$w_0[:, :, 2]$			$w_1[:, :, 2]$					
0	0	0	0	0	0	0	1	1	-1	0	-1	1			
0	0	2	1	1	0	0	-1	0	1	-1	-1	-1			
0	0	0	0	1	1	0	0	0	1	0	0	-1			
0	0	1	1	2	1	0									
0	0	1	1	0	0	0									
0	0	1	1	2	1	0									
0	0	0	0	0	0	0									
Bias B_0 (1x1x1) $b_0[:, :, 0]$							Bias B_1 (1x1x1) $b_1[:, :, 0]$								
1							0								

Table 2.5: Convolution Operation to Compute the Output Activation Value of O[2,0,0]

Input Volume + 1 Padding (7x7x3)							Filter W_0 (3x3x3)			Filter W_1 (3x3x3)			Output Volume O (3x3x2)		
$x[:, :, 0]$							$w_0[:, :, 0]$			$w_1[:, :, 0]$			$o[:, :, 0]$		
0	0	0	0	0	0	0	1	1	-1	1	0	0	3	4	-1
0	2	1	1	1	0	0	0	-1	0	0	0	-1	6	4	2
0	2	2	2	0	0	0	0	1	0	1	1	-1	-3	8	1
0	1	1	0	1	0	0									
0	2	2	0	0	1	0									
0	2	0	1	1	0	0									
0	0	0	0	0	0	0									
$x[:, :, 1]$							$w_0[:, :, 1]$			$w_1[:, :, 1]$			$o[:, :, 1]$		
0	0	0	0	0	0	0	1	0	-1	1	0	-1	-3	-4	-1
0	0	0	2	1	0	0	0	0	1	0	-1	1	-5	-3	-2
0	2	0	1	1	0	0	-1	0	1	0	0	-1	-4	-2	-2
0	0	0	2	1	0	0									
0	1	2	0	1	1	0									
0	2	0	1	2	0	0									
0	0	0	0	0	0	0									
$x[:, :, 2]$							$w_0[:, :, 2]$			$w_1[:, :, 2]$					
0	0	0	0	0	0	0	1	1	-1	0	-1	1			
0	0	2	1	1	0	0	-1	0	1	-1	-1	-1			
0	0	0	0	1	1	0	0	0	1	0	0	-1			
0	0	1	1	2	1	0									
0	0	1	1	0	0	0									
0	0	1	1	2	1	0									
0	0	0	0	0	0	0									
							Bias B_0 (1x1x1)			Bias B_1 (1x1x1)					
							$b_0[:, :, 0]$			$b_1[:, :, 0]$					
							1			0					

Table 2.6: Convolution Operation to Compute the Output Activation Value of O[2,2,0]

Input Volume + 1 Padding (7x7x3) $x[:, :, 0]$							Filter W_0 (3x3x3) $w_0[:, :, 0]$			Filter W_1 (3x3x3) $w_1[:, :, 0]$			Output Volume O (3x3x2) $o[:, :, 0]$			
0	0	0	0	0	0	0	1	1	-1	1	0	0	3	4	-1	
0	2	1	1	1	0	0	0	-1	0	0	0	-1	6	4	2	
0	2	2	2	0	0	0	0	1	0	1	1	-1	-3	8	1	
0	1	1	0	1	0	0										
0	2	2	0	0	1	0										
0	2	0	1	1	0	0										
0	0	0	0	0	0	0										
$x[:, :, 1]$							$w_0[:, :, 1]$			$w_1[:, :, 1]$			$o[:, :, 1]$			
0	0	0	0	0	0	0	1	0	-1	1	0	-1	-3	-4	-1	
0	0	0	2	1	0	0	0	0	1	0	-1	1	-5	-3	-2	
0	2	0	1	1	0	0	-1	0	1	0	0	-1	-4	-2	-2	
0	0	0	0	2	1	0										
0	0	0	0	1	1	0										
0	0	0	2	0	1	0										
0	1	2	0	1	1	0										
0	2	0	1	2	0	0										
0	0	0	0	0	0	0										
$x[:, :, 2]$							$w_0[:, :, 2]$			$w_1[:, :, 2]$						
0	0	0	0	0	0	0	1	1	-1	0	-1	1				
0	0	2	1	1	0	0	-1	0	1	-1	-1	-1				
0	0	0	0	1	1	0	0	0	1	0	0	-1				
0	0	0	1	1	2	1	0									
0	0	1	1	0	0	0										
0	0	1	1	2	1	0										
0	0	0	0	0	0	0										
							Bias B_0 (1x1x1) $b_0[:, :, 0]$			Bias B_1 (1x1x1) $b_1[:, :, 0]$						
							1			0						

2.3.3. Pooling Layer

Pooling is used in CNN architecture to summarize the outputs of neighboring units of feature map. Usually the most distinctive features are transferred to the next layer in a neighborhood to assure invariance of motifs or sub-elements. Furthermore, by applying stride of more than one row or column during the pooling process reduces the data dimension in the next layer. Data dimension reduction both decreases the computation time and number of parameters in the network, so avoid over-fitting. The pooling operation operates over each depth slice of input independently and it does not change the number of depth slices.

There are two hyperparameters to define the pooling operation: the size or the spatial extent of the pooling window (F) and the stride value (S). Stride value determines downsampling coefficient in width and height dimension. If we use, for example, stride value of 2 and have input volume of size 48x48x64, input volume will be downsampled by 2 along width and height dimensions and produces the output volume of size 24x24x64.

Generally, max pooling is the preferred operation in the pooling layers since it outperforms the other alternatives, such as average pooling, in practice. Moreover, the hyperparameters also have some well-known and commonly used values in practice: one of the F=2, S=2 or F=3, S=2 pairs are preferred by the practitioners. The first one, F=2, S=2 is known as non-overlapping pooling while the second one, F=3, S=2, is called as overlapping-pooling which is claimed to avoid overfitting in (Krizhevsky, Sutskever, & Hinton, 2012). Non-overlapping and overlapping max pooling examples are shown in Figure 2.16. Blue shaded cells and yellow shaded cell in the overlapping pooling example mean that they are shared by two and three pooling windows during pooling operation, respectively. Although larger values for spatial extent (window size or receptive size) and stride value are not so common, they are used in some applications.

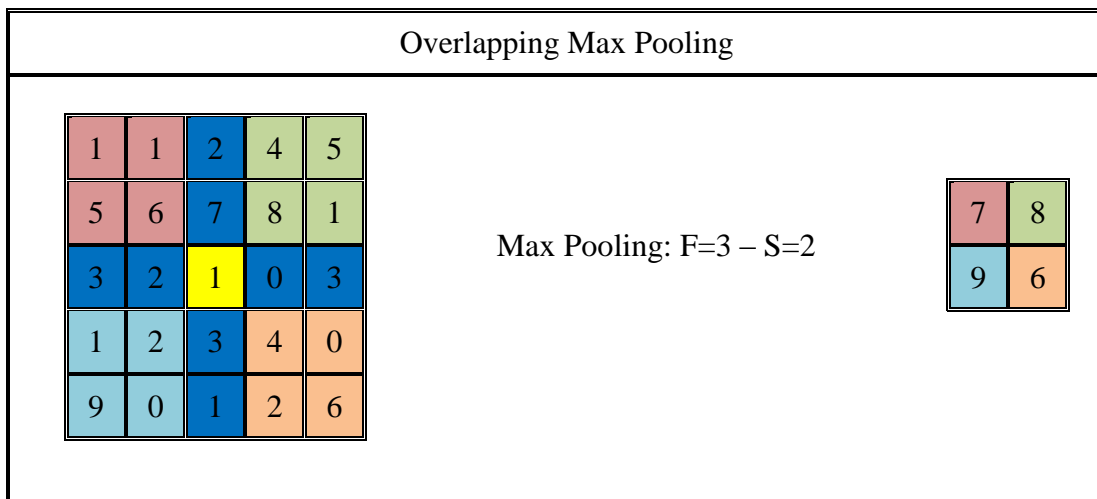
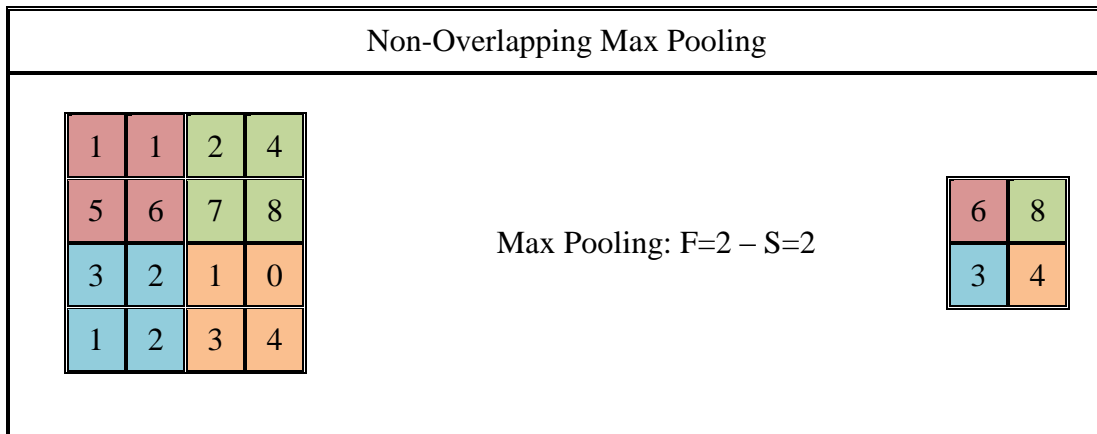


Figure 2.16: Non-Overlapping and Overlapping Max Pooling Examples

Recently, there are some research papers that suggest to eliminating the pooling layers, instead using convolutional layers (Springenberg, Dosovitsky, Brox, & Riedmiller, 2014). They suggest that removing pooling layers does not degrade the performance of the network and it simplifies the network architecture since the network consists of only the stack of convolutional layers. Moreover, they suggest using larger stride convolutional layers to reduce the dimensions.

2.3.4. Normalization Layer

Normalization layers are usually used to bound the output values of neurons so that they do not saturate. However, there exist a common thought that the effect of normalization layer is so small or none, so the usage of normalization layers in state of the art networks are very limited or none, e.g. (Szegedy, et al., 2015).

2.3.5. Fully Connected Layer

Neurons in a fully connected layer are connected to all neurons in the previous layer, as seen in regular neural networks. Their activations can hence be computed with a matrix multiplication followed by a bias offset. Fully connected layers are used to compute the class scores for each class.

2.3.6. Convolutional Neural Network Layer Patterns

General trend is using a structure that multiple number of pairs of convolutional layer with rectified linear unit on top (Conv – ReLU pairs) which is followed by pooling layer. At some point on the network transition to the fully connected layer is made. Fully connected layers also have ReLU on top except the last fully connected layer. At the very end of the network, there is the last fully connected layer that usually produces the class scores. General structure can be formulated as in Equation (2.4). In the structure superscripts indicates repetition of layers. Usually, N value is chosen in between 0 and 3 ($0 \leq N \leq 3$), M value is chosen as positive ($0 \leq M$) and K value is chosen in between 0 and 3 ($0 \leq K \leq 3$). Furthermore, in Equation (2.5) and (2.6), two different convolutional neural network architectures are given as examples.

Conv : Convolutional layer

ReLU : Rectified linear unit

Pool : Optional pooling layer

FC : Fully-connected layer

N : number of (Conv - ReLU) structures

M : number of [(Conv - ReLU)^N + Pool] structures

K : number of (FC - ReLU) structures

$$Input \rightarrow [[Conv \rightarrow ReLU]^N \rightarrow Pool]^M \rightarrow [FC \rightarrow ReLU]^K \rightarrow FC \quad (2.4)$$

$$Input \rightarrow [Conv \rightarrow ReLU \rightarrow Pool]^2 \rightarrow FC \rightarrow ReLU \rightarrow FC \quad (2.5)$$

$$Input \rightarrow [Conv \rightarrow ReLU \rightarrow Conv \rightarrow ReLU \rightarrow Pool]^3 \rightarrow [FC \rightarrow ReLU]^2 \rightarrow FC \quad (2.6)$$

It is recommended to use a stack of small receptive field filters instead of large receptive field single filter. This is mainly due to the fact that when the number of convolutional layers is increased, data goes through more non-linear operations and the features obtained at the output of a stack of layers are richer than the ones at the output of single layer (Karpathy, 2016). If we use, for example 3x3 filters with a stride of 2, we need three convolutional layers to obtain an effective receptive field of 7x7 on input data and the features produced from that structure at the output of the network. Single convolutional layer of 7x7 filters has also the same effective receptive field on input that produces feature vectors at the output, but the number of non-linear transforms that data pass through is different for two structures. In the first one with a stack of layers, feature vector encodes more sophisticated information compared to the second one. Moreover, the number of parameters used in two networks changes drastically. If the number of filters used in two structures is the same and the value is C and the depth of input volume is also C , then $27C^2$ ($3 \times (C \times (3 \times 3 \times C)) = 27C^2$) and $49C^2$ ($C \times (7 \times 7 \times C) = 49C^2$) parameters are used in stack structured network and single layer network, respectively. As a result, we can obtain richer feature vectors by using less number of parameters with stack of convolutional layers with small filter sizes. The drawback is that we need more memory in stack structure. If we have an input volume size of $W \times W$ in width and height dimensions, we need $3 \times C \times W \times W$ units of memory for stack structure and $C \times W \times W$ units of memory for single layer structure.

In practice, the input volume size in width and height dimensions is preferred to be the multiples of 2 and the values of 32, 64, 96, 224, 384 and 512 are used commonly. Moreover, the size of filters (F) is usually chosen as 3x3 and at most 5x5 with a stride (S) of 1 in convolutional layers. Zero-padding is also employed in convolutional layers to preserve the input size at the output and to prevent the data loss at the boundaries of the input volume. In pooling layers, data dimension reduction is achieved by using stride values greater than one. The common form of pooling layer is max pooling with 2x2 receptive size and stride value of 2. This structure achieves a downsampling rate of 2, i.e. 75% of data is discarded at that layer. Pooling layers with 3x3 receptive fields and stride value of 2 is also used but it

is not so common. Larger than 3x3 receptive sizes are hardly preferred since the pooling will be too lossy and aggressive.

2.3.7. Training of Convolutional Neural Networks

Supervised learning is the most common form of the machine learning, whether it is deep or not (LeCun, Bengio, & Hinton, 2015). We can explain *supervised learning* as providing a machine with samples and with their corresponding class labels and adjusting the machine's parameters according to the error between the machine output and the sample class label.

In its simplest form, we want to build a system that can tell us the class of given image sample, i.e. classify the image sample. In order to achieve that it is required to train the machine. During training process, the machine is supplied with image sample and it generates a score vector as an output that has a value for each class label. By comparing the category score values, the class label of the input sample image is decided. The class label of the category with the highest value in the score vector is assigned as the class label of the input image.

Ultimate goal of the system is to classify all input images correctly but this is not the case at the beginning of the training process. Therefore, an objective function (or cost function) to measure the error is required so that we can adjust the parameters of the machine in a way that the machine gives the closest output scores to the desired values for the input samples, i.e. minimizes the error. Cross entropy error is a good alternative to evaluate the training process since it includes information related to the class probability values besides the class assignments (McCaffrey, 2013). Cross entropy is a measure of similarity between two probability distributions. In training process, it can be constructed in such a way that it measures the similarity between true label probability distribution and model output probability distribution. Cross entropy error for single sample in the training set is defined in Equation (2.7) and an example cost function is defined as average cross entropy error for a batch of N samples in Equation (2.8). A comparison of classification error rate and cross entropy error is shown in Table 2.8. There are two models (M1, M2), three labels (L1, L2, L3) and three samples (S1, S2, S3). Both of

the models produces the same classification error rate of 1/3, however, when we analyze the class probability values for two models, we see that model M2 is much better than M1. On the other hand, the average cross entropy values gives the clue that M2 classifies the samples better than M1.

p : true label probability (ground truth)

q : model output probability (network prediction)

$H(p, q)$: cross entropy error

$$H(p, q) = - \sum_i p_i \log(q_i) \quad (2.7)$$

$$L(w) = \frac{1}{N} \sum_1^N H(p_n, q_n) \quad (2.8)$$

Table 2.8: Classification Error Rate and Cross Entropy Error Rate Comparison

		Model Output Probabilities			True Label Probabilities			Correct?	Error Rate	Cross Entropy Error	Avg. Cross Entropy Error
		L0	L1	L2	L0	L1	L2				
M1	S1	0.3	0.3	0.4	0	0	1	Yes	1/3	$-(\ln(0.3)*0)+\ln(0.3)*0+\ln(0.4)*1=-\ln(0.4)$	1.378
	S2	0.3	0.4	0.3	0	1	0	Yes		$-(\ln(0.3)*0)+\ln(0.4)*1+\ln(0.3)*0=-\ln(0.4)$	
	S3	0.1	0.2	0.7	1	0	0	No		$-(\ln(0.1)*1)+\ln(0.2)*0+\ln(0.7)*0=-\ln(0.1)$	
M2	S1	0.1	0.2	0.7	0	0	1	Yes	1/3	$-(\ln(0.1)*0)+\ln(0.2)*0+\ln(0.7)*1=-\ln(0.7)$	0.639
	S2	0.1	0.7	0.2	0	1	0	Yes		$-(\ln(0.1)*0)+\ln(0.7)*1+\ln(0.2)*0=-\ln(0.7)$	
	S3	0.3	0.4	0.3	1	0	0	No		$-(\ln(0.3)*1)+\ln(0.4)*0+\ln(0.3)*0=-\ln(0.3)$	

The method used for training convolutional neural networks is stochastic gradient descent (SGD). SGD is a form of backpropagation such that the weight updates are done after a batch of inputs are fed to the network and average gradient value for those inputs are calculated. Since the weight updates are done over average gradient values of batch inputs and each batch produces noisy estimate of average gradient this process is called as stochastic gradient descent (LeCun, Bengio, & Hinton, Deep Learning, 2015). The process goes over many small data batches even many times until the cost function no longer decreases. Then, model is tested on ‘Test Set’ that the machine has not seen before.

If we think the convolutional layers, single set of weight values, i.e. the same filter, is used in a feature map. The update of filter values are done once for a feature map and this is done by using the gradient averages of neurons in that feature map (Karpathy, 2016).

2.4. Performance Measures

In binary classification systems, performance measures of the system can be defined based on some basic statistical definitions, such as accuracy, sensitivity, specificity, fall-out etc. It is possible to summarize these basic definitions over a contingency table or confusion matrix shown in Table 2.9.

Table 2.9: Basic Statistical Definitions Contingency Table

		Predicted Condition	
		Predicted condition positive	Predicted condition negative
True Condition	Condition positive	True Positive	False Negative
	Condition negative	False Positive	True Negative

2.4.1. True Positive Rate

True positive rate is also called as **sensitivity** or **recall** rate and defined as the ratio of the number of true positives to total number of condition positives.

$$\text{True Positive Rate (TPR)} = \frac{\Sigma \text{True positive}}{\Sigma \text{Condition positive}}$$

2.4.2. False Negative Rate

False negative rate is also called as **miss rate** and defined as the ratio of the number of false negatives to total number of condition positives.

$$\text{False Negative Rate (FNR)} = \frac{\Sigma \text{False negative}}{\Sigma \text{Condition positive}}$$

2.4.3. False Positive Rate

False positive rate is also called as **fall-out** and defined as the ratio of the number of false positives to total number of condition negatives.

$$\text{False Positive Rate (FPR)} = \frac{\Sigma \text{False positive}}{\Sigma \text{Condition negative}}$$

2.4.4. True Negative Rate

True negative rate is also called as **specificity** and defined as the ratio of the number of true negatives to total number of condition negatives.

$$\text{True Negative Rate (TNR)} = \frac{\Sigma \text{True negative}}{\Sigma \text{Condition negative}}$$

2.4.5. Accuracy

Accuracy is defined as the ratio of total number of correct predictions to total number in population.

$$\text{Accuracy (Acc)} = \frac{\Sigma \text{True positive} + \Sigma \text{True Negative}}{\Sigma \text{Total Population}}$$

2.4.6. Performance Measures Used in CAMELYON16

In CAMELYON16 ISBI Challenge, two performance measures are used to evaluate the performance of the algorithms since this challenge evaluates algorithms for both 1) WSI classification and 2) metastasis detection and localization. These performance measures are defined in the following subsections.

Slide-based Evaluation: The merits of the algorithms are assessed for discriminating between slides containing metastasis and normal slides. Receiver operating characteristic (ROC) analysis at the slide level is performed and the area under the ROC curve (AUC) is used as the performance measure for comparing the algorithms in this category. ROC curve is defined as the plot of true positive rate (TPR) versus false positive rate (FPR) for different discriminating threshold values of the binary classifier.

Lesion-based Evaluation: For the lesion-based evaluation, free-response receiver operating characteristic (FROC) curve is used. FROC curve is a graphical plot that is drawn with true positive rate on the y axis and average false positive rate per image on the x axis for different discriminating threshold values of the binary classifier. The final score that ranks teams in the second leaderboard is defined as the average sensitivity at 6 predefined false positive rates: 1/4, 1/2, 1, 2, 4, and 8 false positives per whole slide image. In this category, representative points for metastasis regions with their corresponding probability values are submitted. These points are evaluated according to the ground truth data of the dataset, so true positive rate and false positive rate values are determined. If more than one point inside the same metastasis region is submitted as prospective representative points, the one with the highest probability among those is chosen as representative during the evaluation process and the others are ignored.

CHAPTER 3

PROPOSED APPROACH

In this section, block schema of the proposed solution for CAMELYON16 challenge will be presented, and each step of the solution will be explained in separate sub-sections.

3.1. Block Schema

The block schema of the proposed method is shown in Figure 3.1. The input is whole slide images (WSIs) and outputs are metastasis region representatives together with their probabilities and whole slide probabilities. The method has mainly three steps, which are preprocessing, classification and post processing. Details of these steps are explained in the following sections.

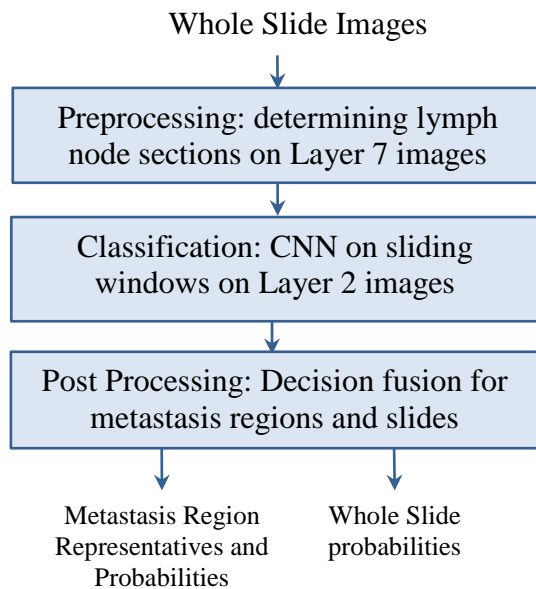


Figure 3.1: Block schema of the proposed method

3.2. Pre-processing

In the training and test sets, WSIs do not contain only the lymph node sections but the majority of the slide area consists of background, which is unnecessary to be included in the analysis of the metastasis detection. In order to speed up the analysis, it is required to apply preprocessing to define the lymph node sections.

Preprocessing operations are conducted in MATLAB. For preprocessing, we used Layer 7 images of the given datasets, which are the 128x down sampled versions of the original images. We applied OTSU thresholding, median filtering and connected component analysis (elimination of small noisy parts) to these images and converted them to binary. As the output of these processes, we obtained the mask of lymph node sections in the WSIs. The effects of these operations are illustrated in Figure 3.2, considering Tumor_009 image of given dataset. From left to right, images are given in the order that original image, OTSU thresholded, median filtered, small connected components eliminated (Mask), and final image (masked image) as output. Moreover, it can be seen in Figure 3.2 that the necessary information related to the lymph node sections in the WSI is not lost.

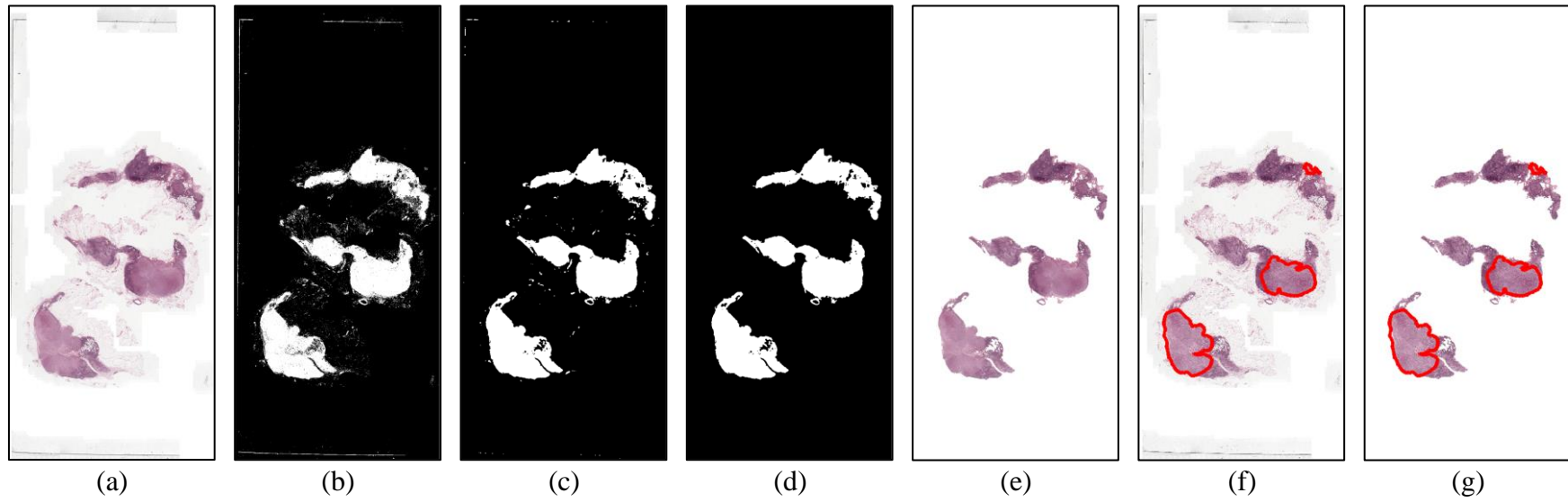


Figure 3.2: Effects of Preprocessing Operations on Tumor_009 Image: a) Original image, b) Otsu thresholding, c) Median filtering, d) Small connected component elimination (mask), e) Final output of preprocessing stage (masked image), f) Metastasis region boundaries shown on original image and g) Metastasis region boundaries shown on masked image.

OTSU method is a statistical, automatic thresholding method to determine the threshold value to separate the pixels of the intensity image into groups such that the variance between the groups is maximized (Otsu, 1979). The two important features of OTSU thresholding method in this application are automatic and unsupervised usage. In (Jassim & Altaani, 2013), OTSU thresholding is hybridized with median filtering. First each channel of RGB images is thresholded by using OTSU method and then all the thresholded channels are merged and median filtered in order to remove the unwanted distortions. Similar to this method, we also hybridize the OTSU method and median filtering. Different from that, we have used OTSU thresholding over all channels by concatenating them side by side to determine threshold value and used this threshold value to obtain binary image from (weighted sum grayscale version of) RGB image.

Lymph node metastases are divided into three categories based on the largest tumor deposit on the tissue: Isolated tumor cells, Micro metastases, Macro metastases (CAMELYON16, 2015).

- **Isolated tumor cells:** Isolated tumor cells are defined as small clusters of cells not greater than 0.2 mm. Many studies have shown that isolated tumor cells do not have negative prognostic value. Therefore this category is not included in the provided dataset's ground truth annotations.
- **Micro-metastases:** Micro-metastatic lymph nodes have cluster of cancer >0.2 mm but no greater than 2.0 mm.
- **Macro-metastases:** Macro-metastatic involvement of the axillary (under-arm lymph nodes) is defined by any tumor cell cluster >2.0 mm.

On the other hand, **normal slides (diagnosis)** means the slide does not contain any tumor cells and the slide is completely free of metastases. So whatever sample you take from normal slides is 100% normal.

The tumor slides in CAMELYON16 challenge contain either micro-metastases or macro-metastases. Removing connected components that have diameter less than 0.2 mm does not contribute to the false positive rate of the system. However, the connected components' sizes are already greater than 0.2mm in diameter in WSIs.

Therefore, dominant factor affecting the connected component elimination threshold value is the necessity of removal of the background at this stage.

3.3. Classification

In this section, dataset preparation for supervised training of the convolutional neural network, structure of the model and its training scheme will be presented.

3.3.1. Dataset for Supervised Training of Convolutional Neural Network

Supervised learning requires a dataset with huge number of labeled samples. Therefore, we constructed a dataset consisting of 64x64x3 RGB images from Layer 2 of original WSIs. Our dataset contains 480000 randomly selected images in total. 244366 of these images are selected from slides with label “normal” and the remaining 235634 are selected from metastasis regions of slides with label “Tumor”.

The mask images obtained by preprocessing are showing the lymph node sections in the WSIs. These masks that we obtained from Layer 7 were used to crop the dataset images from Layer 2. This means that each of 2x2 regions in mask image from Layer 7 corresponds to 64x64 image sections in Layer 2. By cropping these sections we obtained 64x64x3 tensors with R, G, and B layers to construct the dataset. While choosing the cropped dataset images from tumor WSIs, we used the sub-images coming completely from the metastasis regions, that is, we eliminated the image sections coming from normal regions and also from normal-metastasis boundaries. Furthermore, we did not use images that contain background pixels more than 75% in their binary histograms. In Figure 3.3, sample dataset images from “Normal” and “Tumor” classes are shown.

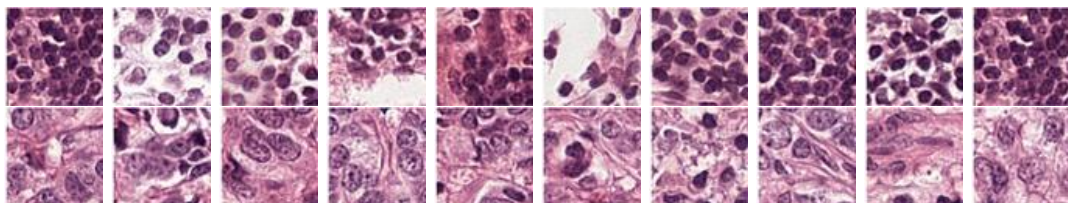


Figure 3.3: Example Dataset Images - First row: Samples with label “normal”, Second row: Samples with label “Tumor”

3.3.2. CNN Structure and Training

In order to classify the subimages of WSIs, we used a CNN structure, which is similar to Google TensorFlow CNN example structure to classify Cifar10 images (TensorFlow). The CNN structure that we used is shown in Figure 3.4.

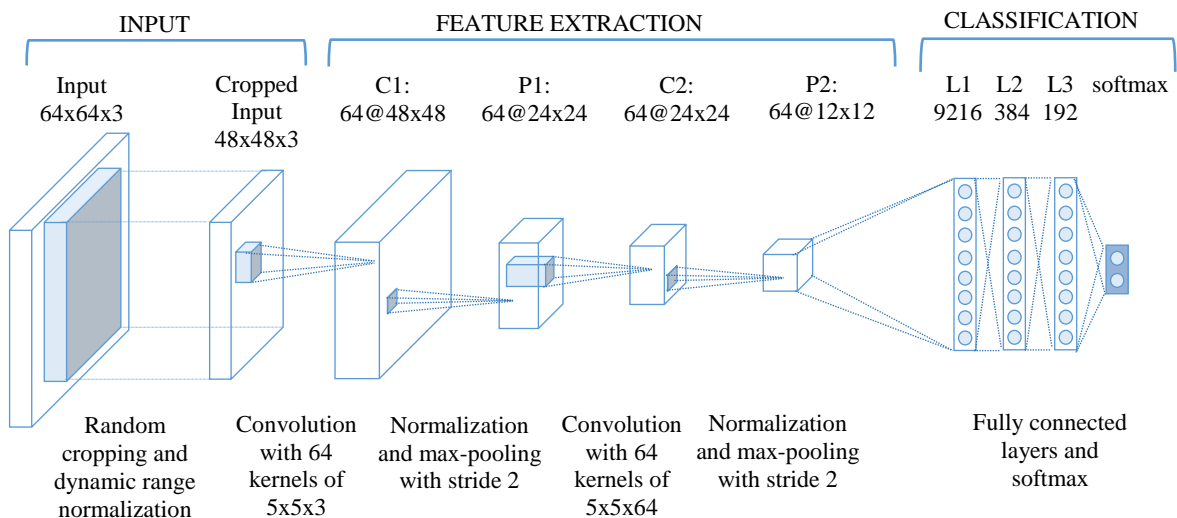


Figure 3.4: Convolutional Neural Network Architecture

The CNN that we used has 2 convolutional layers with RELU (Rectified Linear Units – $\max(0,x)$) outputs since it takes less training time with gradient descent to train the networks with RELU outputs, which is a non-saturating non-linearity, compared to saturating linearities (Krizhevsky, Sutskever, & Hinton, 2012). The output of first convolution layer is max-pooled, normalized, and fed to the input of the second convolution layer. The output of second layer is first normalized, and then max-pooled and fed to the two fully connected layers on the top of convolutional layers. Finally, the output of the last fully connected layer is fed to the 2-way softmax to generate a probability distribution, so classify the input images.

For overlapping and pooling, a filter size of 5x5 and stride value of 2 is applied since it is declared in (Krizhevsky, Sutskever, & Hinton, 2012) that structures with this type of pooling avoids overfitting and gives the same data dimension at the output when compared to non-overlapping techniques.

Dataset images of size $64 \times 64 \times 3$ are not fed directly to the network. First, they are cropped randomly to the size of $48 \times 48 \times 3$ in order to prevent rapid over-fitting of the model and then the means of the images are subtracted from them and they are divided by their variances to normalize the dynamic range of the images.

At the first convolutional layer 64 kernels of size $5 \times 5 \times 3$ are used. At the output of this layer, there exists a normalization and max-pooling layer with a stride of 2, so the tensor at the input of the second convolutional layer has a $24 \times 24 \times 64$ structure. Second layer has 64 kernels of size $5 \times 5 \times 64$ to filter its inputs. Following it, there is a max-pooling and normalization block with a stride of 2 again, and this reduces the tensor dimension to $12 \times 12 \times 64$. This last tensor is converted to a vector having 9216 feature components and fed to the fully connected layers. Fully connected layers have 384 and 192 nodes, respectively. Outputs of the second max-pooling, normalization block, and nodes of fully connected layers construct a classification network of structure $9216 \times 384 \times 192$. At the top of this network there is a softmax classifier generating the probability distributions for class values, which will be used to detect and localize the metastasis regions in WSIs.

After we have trained the model, we have cropped all of the WSIs in the test set $64 \times 64 \times 3$ sub-images in the same way that we have prepared the dataset to train the model. Then, we have fed all sub-images of each WSI to the network and obtain the probability value that the sub-image contains the metastasis. A small patch of CNN model output is illustrated in Figure 3.5.

3.4. Post-processing

To obtain the required performance measures by CAMELYON16, it is required to process the model outputs for sub-images of WSIs further. The outputs of the CNN provide us with the class labels and probabilities of $64 \times 64 \times 3$ WSI regions in Layer 2. By using the labels and the probabilities of these sub-image regions, we have constructed two matrices for Layer 2: a label matrix and a coefficient matrix. The class label of a $64 \times 64 \times 3$ region in Layer 2 is mapped to a 2×2 region in Layer 7 with the same label. Mapping of label assignments constructs a binary image in Layer 7 with white pixels (1) representing metastasis class label and black (0) pixels

representing normal class label. Similarly, we have also constructed a coefficient matrix for each WSI by using the CNN probability output values of the corresponding regions. While constructing this matrix, the model output probability values of the metastasis regions are recorded as positive, but the values of normal regions are recorded as negative. The 0 values in coefficient matrix represent that the regions corresponding to that pixels do not contain lymph node components, i.e. these are background regions. Binary image and coefficient matrix are used together to detect the metastasis regions and compute the corresponding probability of these regions.

The constructed binary image is eroded with a disk shaped structuring element to eliminate the small metastasis regions. We know that each white pixel in the binary image corresponds to a metastasis region. Moreover, we have a coefficient matrix with positive values for metastasis regions and negative values for normal regions. In order to aggregate this information with the ultimate goal of detecting metastasis regions and localizing them, we have employed a confidence filter on the coefficient matrix.

Confidence filter has a size of 7×7 and the values of the filter has a Gaussian-like shape with symmetry axis on the center of the filter and tails decreasing towards the boundaries of the filter which is shown in Figure 3.5. This filter is applied to the coefficient matrix in a sliding window fashion. Let $CNN(I,J)$ be the value of the CNN output at the center of the window where the filter is applied. The filtered result for the center is calculated using Equation (3.1).

$$CNN_{CF}(I,J) = (\sum_i \sum_j CNN(I+i, J+j)CF(i,j))/CF_{sum} \quad (3.1)$$

where CF_{sum} is the sum of the components of Confidence Filter.

If both $CNN(I,J)$ and $CNN_{CF}(I,J)$ are positive, the center pixel is recorded as belonging to metastasis region with a probability $P(I,J)$ that is calculated using Equation (3.2).

$$P(I,J) = \begin{cases} CNN(I,J) * CNN_{CF}(I,J) & \text{if } CNN(I,J) > 0 \text{ and } CNN_{CF}(I,J) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (3.2)$$

An example step by step calculation of probability of belonging to metastasis region for the pixel at the center of a small patch is shown in Figure 3.5. Illustrative model output values for small patch, filter coefficients used in post-processing and other steps are illustrated.

After processing the whole image, we have obtained a probability image with positive values for metastasis pixels and 0 for normal pixels. Over this image, we conducted a connected component analysis to determine the metastasis regions as a whole so that we can provide a single point inside a metastasis region and single probability value for that region. While determining the point that is the representative of the corresponding metastasis region and the probability value that the region contains metastasis, first, the maximum probability value of metastasis pixels inside the connected component is determined. Then, the pixels that have the probability value between $[\text{max value} - 0.2, \text{max value}]$ within the connected component are searched. Among these pixels, the one farthest to the connected component boundaries is chosen as the representative of that region (location of that pixel at Layer 0 is given as the metastasis region location) and the probability value of that pixel as the probability of that region.

After determining and localizing the metastasis regions inside the WSIs, the highest probability value inside a WSI is assigned as the probability that the WSI may contain metastasis for slide level evaluation.

Post processing stages for Tumor_009 image are shown in Figure 3.6. In (a) binary image constructed from the CNN output labels is shown. On this image, an erosion operation is conducted with a disk-shaped structuring element to eliminate the small metastasis regions and image in (b) is obtained as a result. By using the eroded image and the corresponding probability matrix, confidence filtering operation is done and color coded probability image shown in (c) is obtained. Probability distribution is given in the green channel of the image, i.e. the bright green regions are the regions that have the highest probability to contain metastasis and color goes to black as probability decreases. Representative points obtained from this

probability map are shown on that probability image and evaluation mask image of Tumor_009 WSI in (d) and (e), respectively.

Model Output Values Around (I,J): CNN(I+i,J+j)						
0.000	0.800	0.820	0.850	0.800	0.800	0.700
0.000	0.800	0.800	0.850	0.700	0.700	0.750
0.000	0.000	0.900	0.850	0.900	0.900	0.900
-0.900	0.000	0.520	0.820	0.900	0.600	0.900
-0.900	0.000	0.000	0.900	-0.900	0.700	0.900
-0.900	0.000	0.000	0.700	-0.900	0.900	0.000
-0.950	-0.700	0.000	0.000	0.000	0.900	0.000
CNN(I,J) = 0.820						

Value < 0 Normal

Value = 0 No Decision

Value > 0 Metastasis

Filter: CF(i,j)						
0.004	0.004	0.004	0.004	0.004	0.004	0.004
0.004	0.054	0.054	0.054	0.054	0.054	0.004
0.004	0.054	0.242	0.242	0.242	0.054	0.004
0.004	0.054	0.242	0.399	0.242	0.054	0.004
0.004	0.054	0.242	0.242	0.242	0.054	0.004
0.004	0.054	0.054	0.054	0.054	0.054	0.004
0.004	0.004	0.004	0.004	0.004	0.004	0.004
CF_{sum} = 3.295						

Model Output * Filter: CNN(i,j) * CF(i,j)						
0.000	0.003	0.003	0.003	0.003	0.003	0.003
0.000	0.043	0.043	0.046	0.038	0.038	0.003
0.000	0.000	0.218	0.206	0.218	0.049	0.004
-0.004	0.000	0.126	0.327	0.218	0.032	0.004
-0.004	0.000	0.000	0.218	-0.218	0.038	0.004
-0.004	0.000	0.000	0.038	-0.049	0.049	0.000
-0.004	-0.003	0.000	0.000	0.000	0.004	0.000
Σ_iΣ_j CNN(I + i, J + j)CF(i, j) = 1.696						

Calculation for P(I,J)

CNN(I,J) = 0.820

CF_{sum} = 3.295

$$CNN_{CF}(I, J) = \frac{(\sum_i \sum_j CNN(I+i, J+j)CF(i, j))}{CF_{sum}} = \frac{1.696}{3.295} = 0.515$$

P(I, J) = CNN(I, J) * CNN_{CF}(I, J) = 0.820 * 0.515 = 0.423

Figure 3.5: Calculation of Probability of Belonging to Metastasis Region for a Pixel at (I,J)

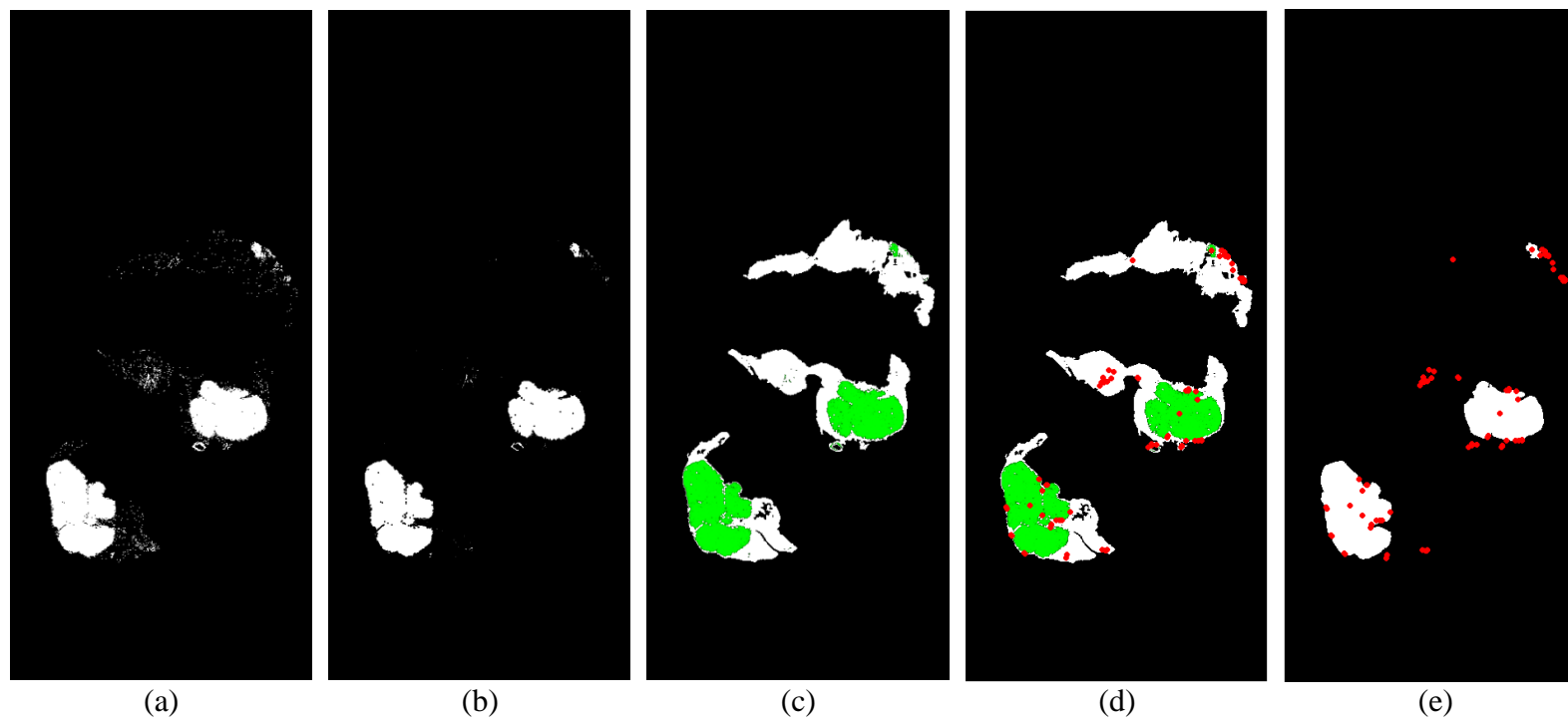


Figure 3.6: Post Processing Stages for Tumor_009 Image: a) Binary image showing metastasis regions constructed from CNN output labels, b) Eroded binary image eliminating small regions, c) Probability image obtained after Confidence Filtering (green area), d) Metastasis representative points shown on probability image, e) Metastasis representatives shown on evaluation mask image

CHAPTER 4

EXPERIMENTAL RESULTS

In this chapter, the experimental work done during the development phase of this thesis will be explained. Development platforms and development environments used during this process will be defined. The justification of selected parameters of methods for each step of the proposed solution will be done based on experimental results. Finally, the results of the experimental studies will be presented.

4.1. Development Platforms and Environments

Different stages of the processing steps are executed on different platforms and in different environments. Two different platforms are used during the experiments of this thesis: a workstation and a desktop computer. The specifications of the platforms are given in Table 4.1 and Table 4.2 for workstation and desktop, respectively.

Pre-processing operations on whole slide images, dataset preparation for supervised training of the convolutional neural network model and post-processing operations to obtain slide based and lesion based performance measures have been done in MATLAB R2013b environment on desktop computer. Classification of whole slide images according to the trained model is also done on desktop computer but in the Tensorflow/Python environment. The only operation conducted on workstation is training of the convolutional neural network in Tensorflow/Python environment on TitanX GPUs in parallel fashion.

Table 4.1: Workstation Technical Specifications

CPU Specifications	CPU Model	Intel Core i7-5820
	CPU frequency	3.3 GHz
	Number of CPUs	12
	Number of cores per CPU	6
	RAM	32 GB
GPU Specifications	GPU Model	NVIDIA TITAN X
	CUDA cores	3584
	Base Clock	1417 MHz
	Boost Clock	1531 MHz
	Memory Speed	10 Gbps
	Standard Memory Configuration	12 GB DDR5X
	Memory Interface Width	384-bit
	Memory Bandwidth	480 GB/sec

Table 4.2: Desktop Computer Technical Specifications

CPU model	Intel Core i7-3770
CPU frequency	3.4 GHz
Number of cores	4
RAM	8 GB

4.2. Layer Selection

Pre-processing and post-processing operations are executed on Layer 7 while dataset images to train machine are extracted from Layer 2 images. Layer 2 is chosen heuristically based on medical knowledge related to metastasis detection. Pathologists are interested in some specific features of the nuclei in hematoxylin and eosin stained slides. The size, shape, texture of nuclei and the special arrangements and organization into tubules are the most valuable feature to pathologists (Veta, Pluim, van Diest, & Viergever, 2014). In other words dataset images must be chosen from a certain layer with a certain size so that the nuclei features can be observed by

the machine during the training. Another concern is that the size of images should be small to train machine in minimum time period and with minimum resources. Therefore, Layer 2 is chosen as source layer to the dataset sample images with a size of 64x64 pixels. In Figure 3.3, the sample images from both normal and metastasis classes are shown and the discrimination of two classes can be done visually. Layer 7 is chosen because of the fact that it is meaningful to choose the highest layer common among the whole slide images to eliminate the background in pre-processing and to localize the metastasis regions in post-processing stages within minimum computation time.

4.3. Parameter Selection for Pre-processing Operations

Selection of median filter size and connected component elimination threshold value is explained in this part.

4.3.1. Median Filter Size

Median filter size of 5-pixel is used while filtering images after OTSU thresholding. The filter size values of 2, 5 and 10 is tried on Tumor_009 image and results after pre-processing are shown in Figure 4.1 (a), (b) and (c), respectively. As it is seen in (a), lines belonging to the background remain around the boundaries of the image after pre-processing. In case of (c), median filtering becomes destructive and some necessary parts begin to vanish around lymph node sections. Therefore, the filter size of 5 is chosen.

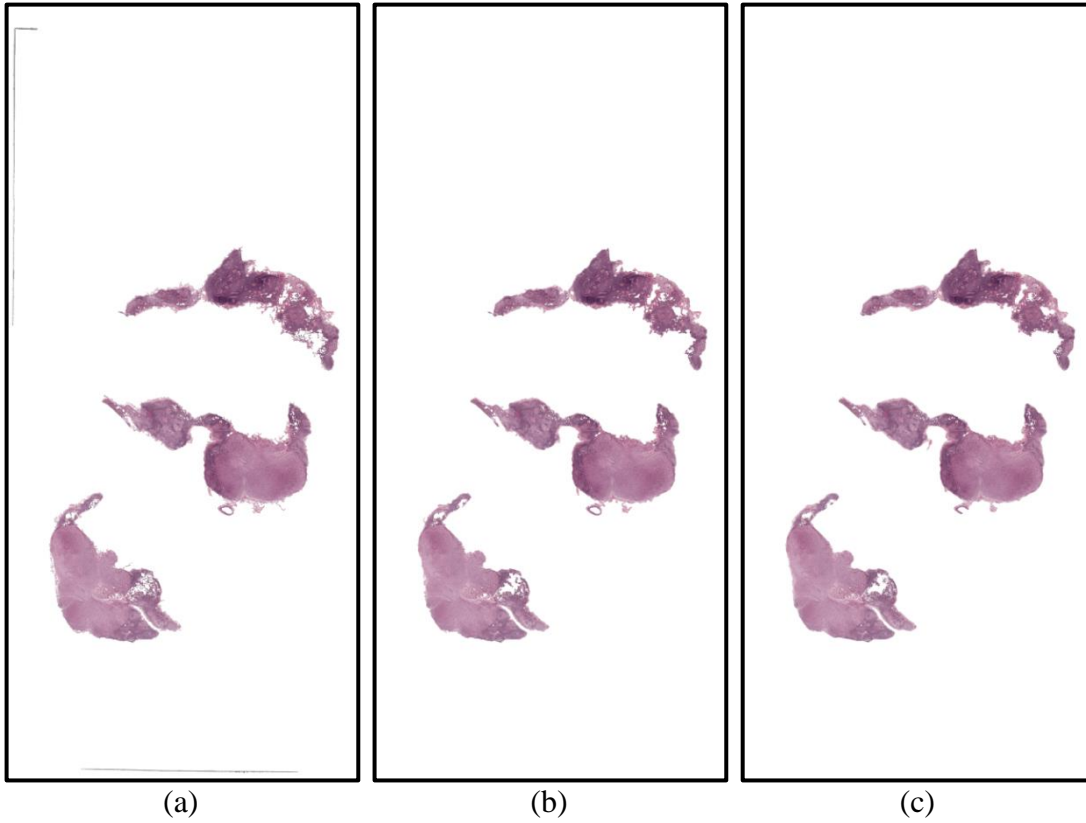


Figure 4.1: The Effect of Median Filter Size on Pre-processed Image: a) Pre-processed image with filter size of 2, b) Pre-processed image with filter size of 5, c) Pre-processed image with filter size of 10

4.3.2. Connected Component Elimination Threshold Value

The connected component elimination threshold value is chosen by considering metastasis region size constraints and the necessity of removing background from whole slide images so as to obtain lymph node section samples while constructing the database.

As it is explained in pre-processing section, micro and macro metastasis regions, i.e. the tumors that have diameter greater than 0.2mm, are taken into consideration while detecting the metastasis regions in this study. The physical size value of metastasis regions corresponds to 64 pixels in Layer 7 images as it is calculated in Equation (4.1).

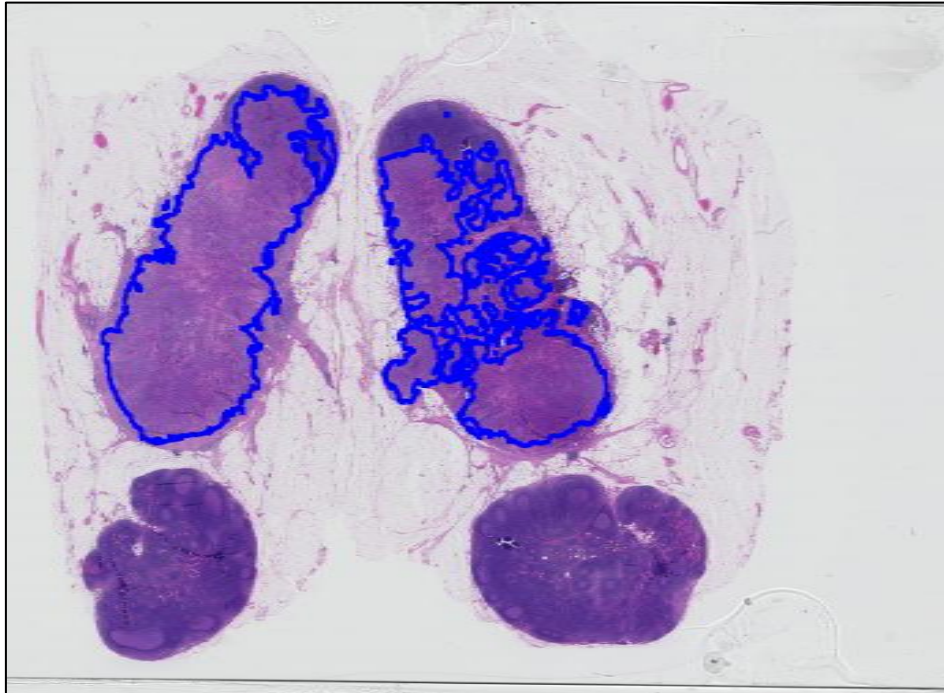
$$\text{Level-0 Physical Spacing} = 0.243 \mu\text{m/pixel}$$

$$\text{Level-7 Physical Spacing} = 0.243 * 2^7 = 31.104 \mu\text{m/pixel}$$

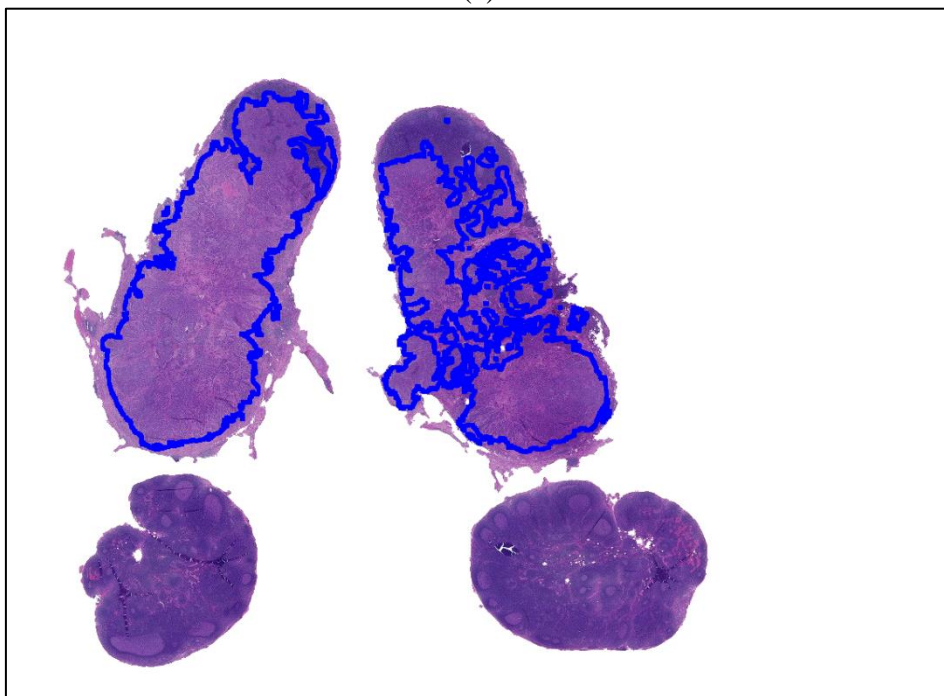
$$\text{Level-7 Physical Spacing Threshold} = (2000 \mu\text{m}) / (31.104 \mu\text{m/pixel}) = 64 \text{ pixels}$$

(4.1)

Since metastasis regions are contained within the lymph node sections and the glass slides contains approximately 15mmx15mm biopsy samples, the connected components' sizes are already greater than 0.2mm in diameter. The dominant factor affecting the connected component elimination threshold value is the necessity of removal of the background. Therefore, the threshold value is chosen as 800 pixels for 4-neighborhood connected component analysis. As a result of this operation, all the necessary parts of the whole slide images are preserved. The result of connected component elimination with a threshold value of 800 pixels is shown for Tumor_089 WSI in Figure 4.2. In (a), metastasis regions' boundaries are drawn on original WSI in Layer 7 while they are drawn on pre-processed WSI with the threshold value of 800 pixels in (b).



(a)



(b)

Figure 4.2: Result of connected component elimination with 800 pixels threshold value: a) Metastasis boundaries shown on original WSI, b) Metastasis boundaries shown on pre-processed WSI with 800 pixels connected component elimination threshold value

4.4. Classification Stage CNN Architectures

Six different convolutional neural network architectures were designed, trained and compared with each other in the context of this dissertation. All of them consist of convolutional and max-pooling layers but no normalization layers except the one that was used in CAMELYON16 challenge. This model contains normalization layers also. Each of the models is explained in the following sub-sections in a detailed manner and the summary of the structures of the models are given in Table 4.3. The models are named from Model 0 to Model 5. The model that we have used for CAMELYON16 Challenge is named as Model 0 and the others are named consecutively. In table Table 4.3, C is used for convolutional layer, P for pooling layer, N for normalization layer, F for fully-connected layer and SM for softmax layer. Superscripted numbers mean the repetition of the corresponding structure and it is represented with xN (N: number of repetitions) in the block diagrams of the models in the following sections.

Table 4.3: Summary of CNN Architectures of Models

Model 0	C – P – N – C – N – P – (F) ² – SM
Model 1	(C – P) ² – (F) ² – SM
Model 2	(C – C – P) ² – (F) ² – SM
Model 3	(C – C – C – P) ² – (F) ² – SM
Model 4	(C – C – C – P) ³ – (F) ² – SM
Model 5	(C – P) ² – (F) ² – SM

All of the models share the same input data structure: 64x64x3 RGB images are randomly cropped to 48x48x3 images and dynamic range normalization is applied on these images. They also share the same fully connected and softmax layers stack at the top of the network to classify the samples. The different part of the networks is the feature extraction parts, i.e. the convolutional layers. Moreover, weights of the models Model 1 to Model 5 are initialized similar to method given in (Glorot & Bengio, 2010). On the other hand, weight initialization of Model 0 is the same with the model given in (TensorFlow).

To train and test the performances of these six models we have constructed training and test set from CAMELYON16 training set WSIs. We have constructed a training set with 40000 samples, 20000 of them normal and 20000 of them metastasis. Training set samples are chosen from Normal_001 - Normal_120 and Tumor_001 - Tumor_079 WSIs. The test set is constructed from the remaining WSIs and it contains 10000 samples with equal number of normal and metastasis samples.

We have trained each model with 288 epochs (90000 steps) and we have updated learning weight at the 200 epochs (62500 steps). Learning rate update graph is given in Figure 4.3.

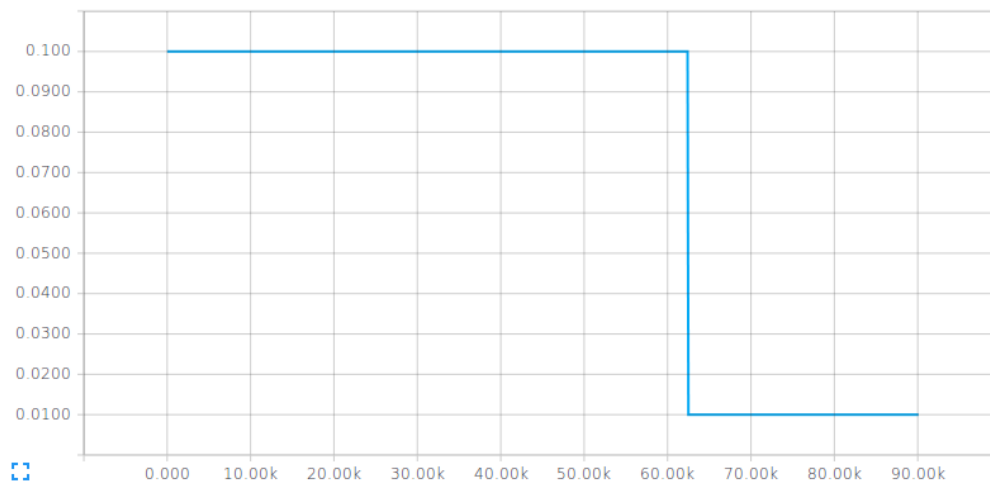


Figure 4.3: Learning Rate Update Graph during Training of the Models

4.4.1. CNN Architecture: Model 0

This is the architecture that we have used to classify the test set in CAMELYON16 challenge and this architecture is also similar to Google TensorFlow CNN example structure to classify Cifar10 images (TensorFlow). Block diagram of this model is shown in Figure 3.4. Input image goes through first convolutional layer of 64 kernels with 5x5 filter size and stride of 1 with zero padding, i.e. the size of the data volume in width and height dimensions does not change at the output of the convolutional layer. After the first convolutional layer, overlapping pooling scheme is applied on data with filter size of 3x3 and stride of 2. This reduces the data dimension by 2 in width and height dimensions. Then, normalization is done. The

output of normalization layer goes through convolutional layer, normalization layer and max-pooling layer one more time and fed to the fully connected and softmax layers to be classified. The important properties of this model are the filter sizes of the convolutional layers and the overlapping pooling scheme in max-pooling layers. 5x5 filter size can be said to be large when compared to contemporary convolutional network models but it enables the model to extract motifs with less number of convolutional layers.

In Figure 4.4, filter images of first convolutional layer are shown. The motifs learned by the filters are not so obvious but if we think that the dark pixels correspond to nucleus and around light pixels to stroma, then the motifs are consistent with the medical insights related to the tubule formation of nuclei. We can see that the filters commonly have dark pixels surrounded by light pixels and dark pixels construct some patterns.

When the model is trained over the training set and evaluated on the test set, confusion matrix shown in Table 4.4 is obtained. Accuracy value of 0.8683 is obtained.

Histograms related to the training of this model are given in APPENDIX G.

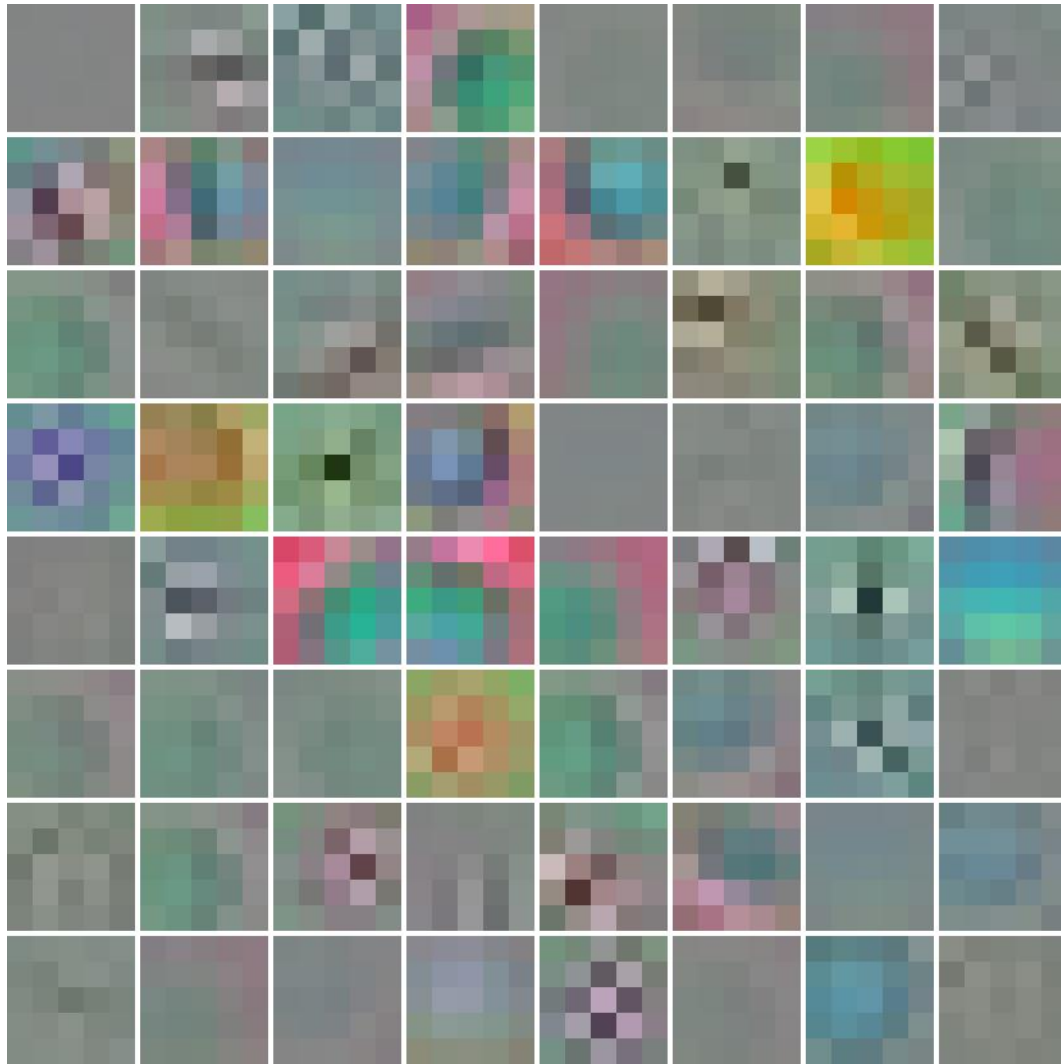


Figure 4.4: First Convolutional Layer Filter Images of Challenge Model

Table 4.4: Confusion Matrix for CCN Model 0

		Estimated Labels	
		NORMAL	METASTASIS
Known Classes	NORMAL	4920	80
	METASTASIS	1237	3763
Accuracy		0.8683	

4.4.2. CNN Architecture: Model 1

In this model, we have used 3x3 filters in convolutional layers and 2x2 filters in max-pooling layers. Stride values of 1 and 2 are used in convolutional layers and

pooling layers, respectively. We have also used zero padding in convolutional layers to preserve the data dimensionality throughout the convolutional layers. In feature extraction section of the CNN, we have only used convolutional layers and max-pooling layers. Base structure of one convolutional layer followed by one max-pooling layer is used and two of these base structures are stacked on top of each other, i.e. two convolutional layers and two max-pooling layers are used in (Conv → Max-pool) fashion. Block diagram of the architecture is presented in Figure 4.5.

The images of filters that have been learned in the first convolutional layer of this model are shown in Figure 4.6. This time it is harder to see the patterns in the filter images. However, the performance of the model is close to the model used in challenge. Confusion matrix of the model is given in Table 4.5. Accuracy value of 0.8562 is achieved.

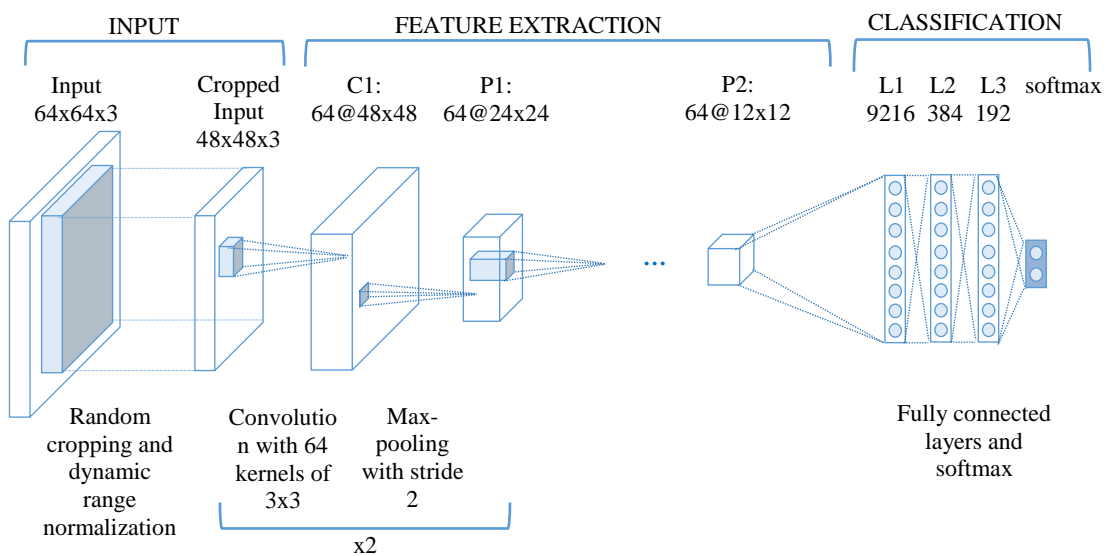


Figure 4.5: Block Diagram of Model 1 CNN Architecture

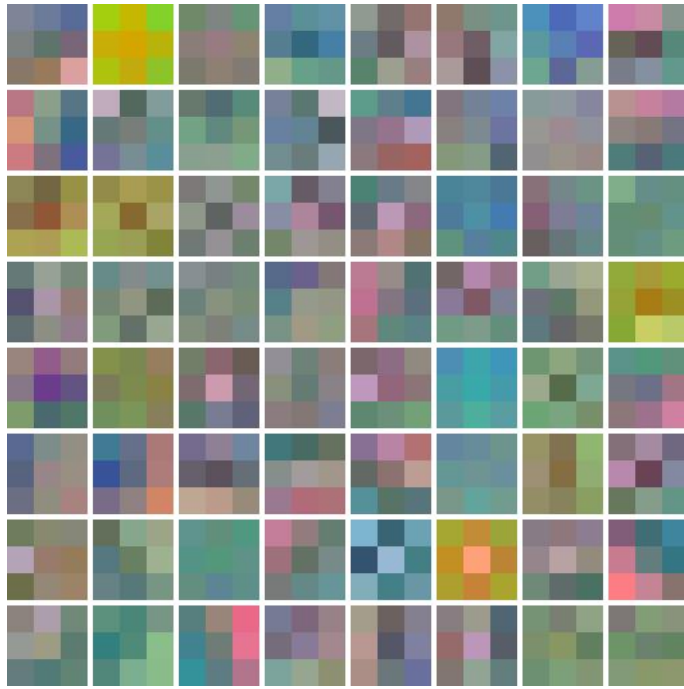


Figure 4.6: First Convolutional Layer Filter Images of Model 1

Table 4.5: Confusion Matrix for CNN Model 1

		Estimated Labels	
		NORMAL	METASTASIS
Known Classes	NORMAL	4894	106
	METASTASIS	1332	3668
Accuracy		0.8562	

4.4.3. CNN Architecture: Model 2

This architecture contains one more convolutional layer in its base structure, so there are two convolutional layer followed by a max-pooling layer and this structure repeated twice in the feature extraction section. The importance of this model is that it has filters with sizes 3x3 in convolutional layers which is smaller than CAMELYON16 challenge model but two convolution layers per max-pooling layer exists. Hence, the effective receptive field obtained on the input image at the end of first max-pooling layer is 5x5. Moreover, the feature maps go through more non-

linear transforms, so the features should be richer than the CAMELYON16 challenge model. The block diagram of Model 2 is given in Figure 4.7.

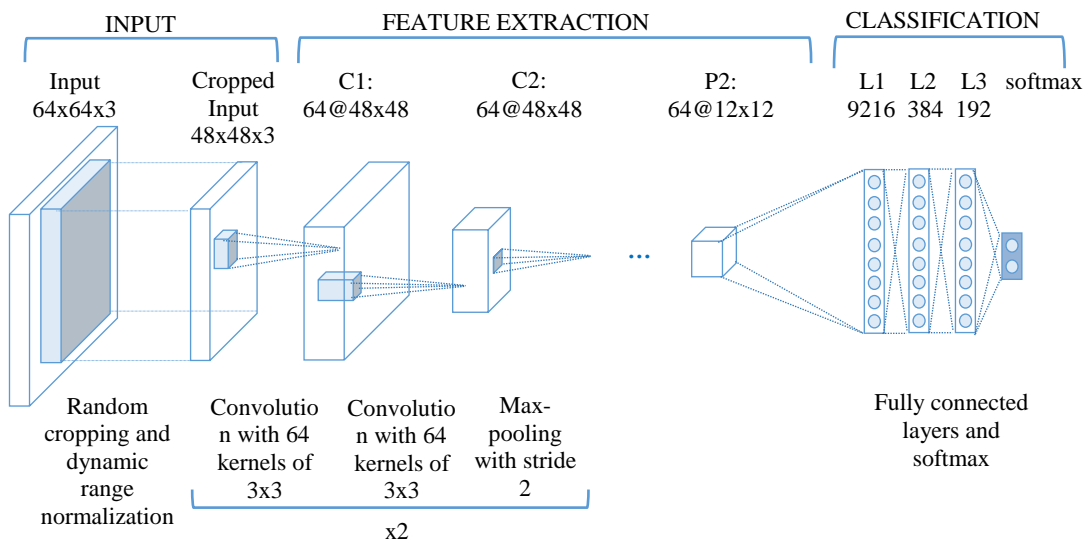


Figure 4.7: Block Diagram of Model 2 CNN Architecture

In Figure 4.8, filter images of first convolutional layer of the second model is shown. Tetragonal patterns again emerge in the filter images but it is hard to match these patterns with some physical observations we have done for the challenge model. It becomes harder and harder as the number of layers increases.



Figure 4.8: First Convolutional Layer Filter Images of Model 2

When we analyzed the confusion matrix table of the model, which is shown in Table 4.6, we have seen that the accuracy of the system has increased as it is expected. Although the difference less than 1%, it shows us that using small filters with more convolutional layers improves the system performance. Furthermore, we can infer that removing the normalization layers does not degrade the system performance. We have attained the accuracy value of 0.8751 in this model.

Table 4.6: Confusion Matrix for CNN Model 2

		Estimated Labels	
		NORMAL	METASTASIS
Known Classes	NORMAL	4905	95
	METASTASIS	1154	3846
Accuracy		0.8751	

4.4.4. CNN Architecture: Model 3

To increase the number of convolutional layers one more in base structure of the feature extraction section, we have constructed Model 4. Three convolutional layers

are followed by a max-pooling layer and this scheme repeated twice. Hence, we have 6 convolutional layers and two max pooling layers. Filter sizes of the convolutional layers are again 3x3 similar to the previous models. With this model, we have again preserved the data dimension supplied to the fully connected layer in the architecture. The block diagram of the model is shown in Figure 4.9. We have constructed this model to test the effect of number of convolutional layers on the performance of the system.

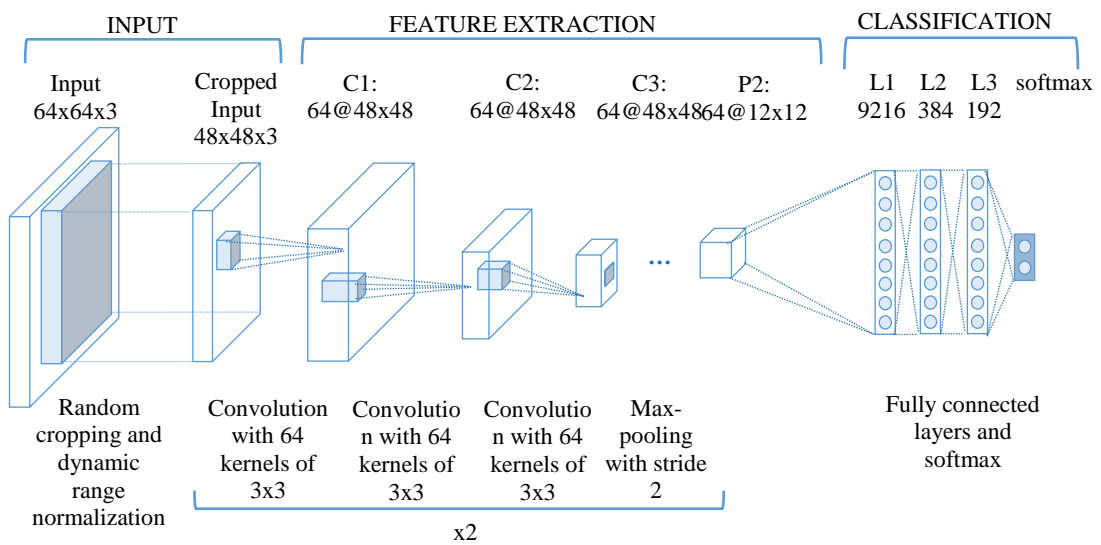


Figure 4.9: Block Diagram of Model 3 CNN Architecture

In Figure 4.10, filter images of the first convolutional layer of Model 3 is given. Although, there are no special patterns different than previous models, we have seen similar filter images in this model also.

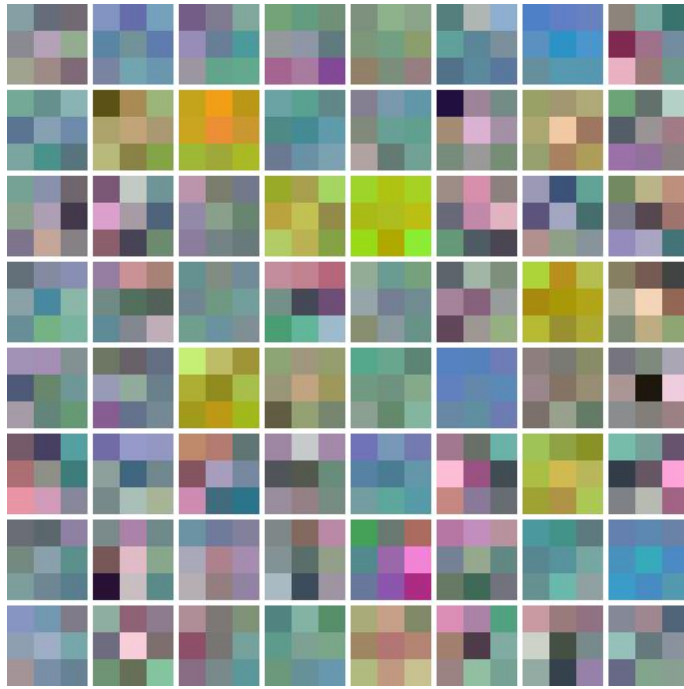


Figure 4.10: First Convolutional Layer Filter Images of Model 3

We have also given the confusion matrix of Model 3 in Table 4.7. We have achieved accuracy value of 0.8728 which is slightly less than the value of Model 2.

Table 4.7: Confusion Matrix for CNN Model 3

		Estimated Labels	
		NORMAL	METASTASIS
Known Classes	NORMAL	4891	109
	METASTASIS	1163	3837
Accuracy		0.8728	

4.4.5. CNN Architecture: Model 4

This model has the same base structure with Model 3 but this has three times repeated version of base structure rather than two in Model 3. Hence, this model has nine convolutional layers each three of which is followed by a max-pooling layer. Since we have three max-pooling layers with stride of 2, data dimension fed to fully connected layer is halved both in width and height dimensions. In other words, while we are increasing the number of non-linearities data goes through network, we are

decreasing size of feature vector fed to the fully connected layer, i.e. lose some of the features. Block diagram of Model 4 is given in Figure 4.11.

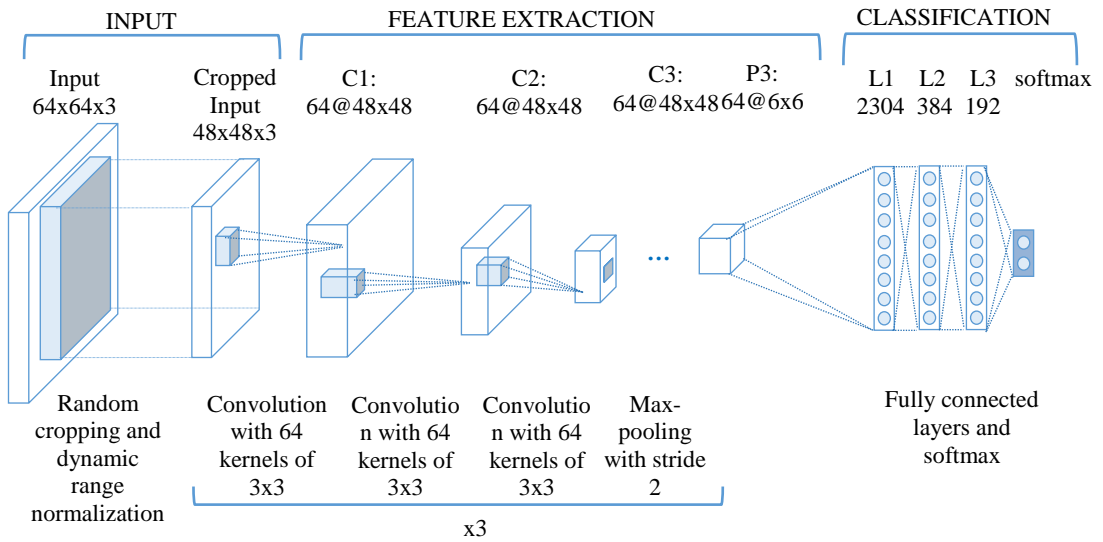


Figure 4.11: Block Diagram of Model 4 CNN Architecture

In Figure 4.12, first convolutional layer filter images of Model 4 is presented. Similar patterns to previous models also exist in this model.

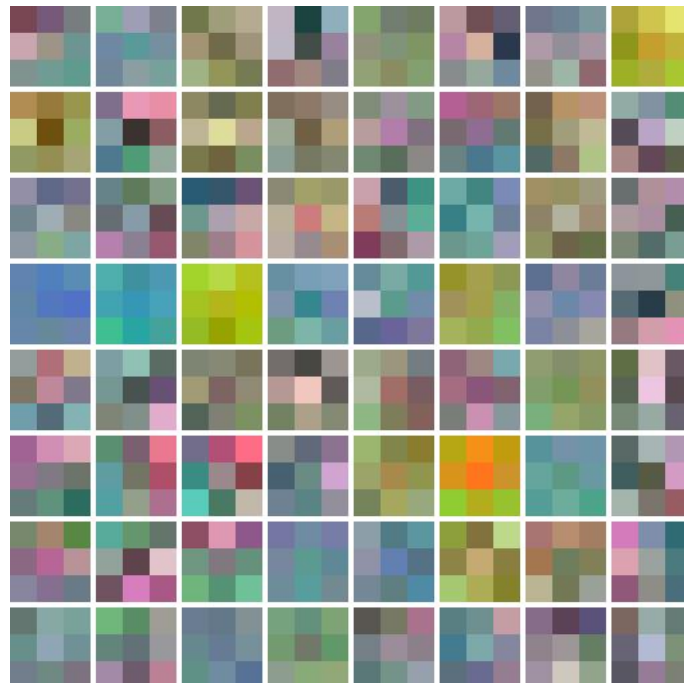


Figure 4.12: First Convolutional Layer Filter Images of Model 4

In Table 4.8, confusion matrix of Model 4 is given. Accuracy value of 0.8624 has been reached with this model. The value again decreases, but this may be due to the reduction in the dimension of the feature vector supplied to the fully convolutional layers of the network.

Table 4.8: Confusion Matrix for CNN Model 4

		Estimated Labels	
		NORMAL	METASTASIS
Known Classes	NORMAL	4905	95
	METASTASIS	1281	3719
Accuracy		0.8624	

4.4.6. CNN Architecture: Model 5

This is the last model we have constructed to test the effect of removing normalization layer in CNN model. This model is the normalization layers removed version of the Model 0 architecture. Block diagram of this model is shown in Figure 4.13. Input image goes through first convolutional layer of 64 kernels with 5x5 filter size and stride of 1 with zero padding. After the first convolutional layer, overlapping pooling scheme is applied on data with filter size of 3x3 and stride of 2. This reduces the data dimension by 2 in width and height dimensions. The output of first pooling layer goes through second convolutional layer and max-pooling layer one more time and fed to the fully connected and softmax layers.

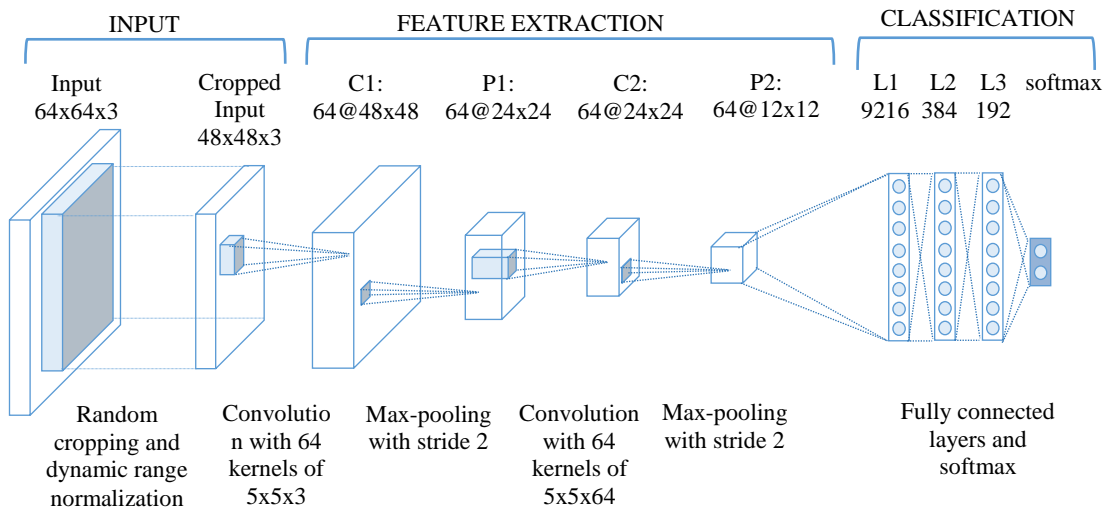


Figure 4.13: Block Diagram of Model 5 CNN Architecture

The images of filters that have been learned in the first convolutional layer of this model are shown in Figure 4.14. Moreover, confusion matrix of the model is given in Table 4.9. Accuracy value of 0.8768 is achieved.

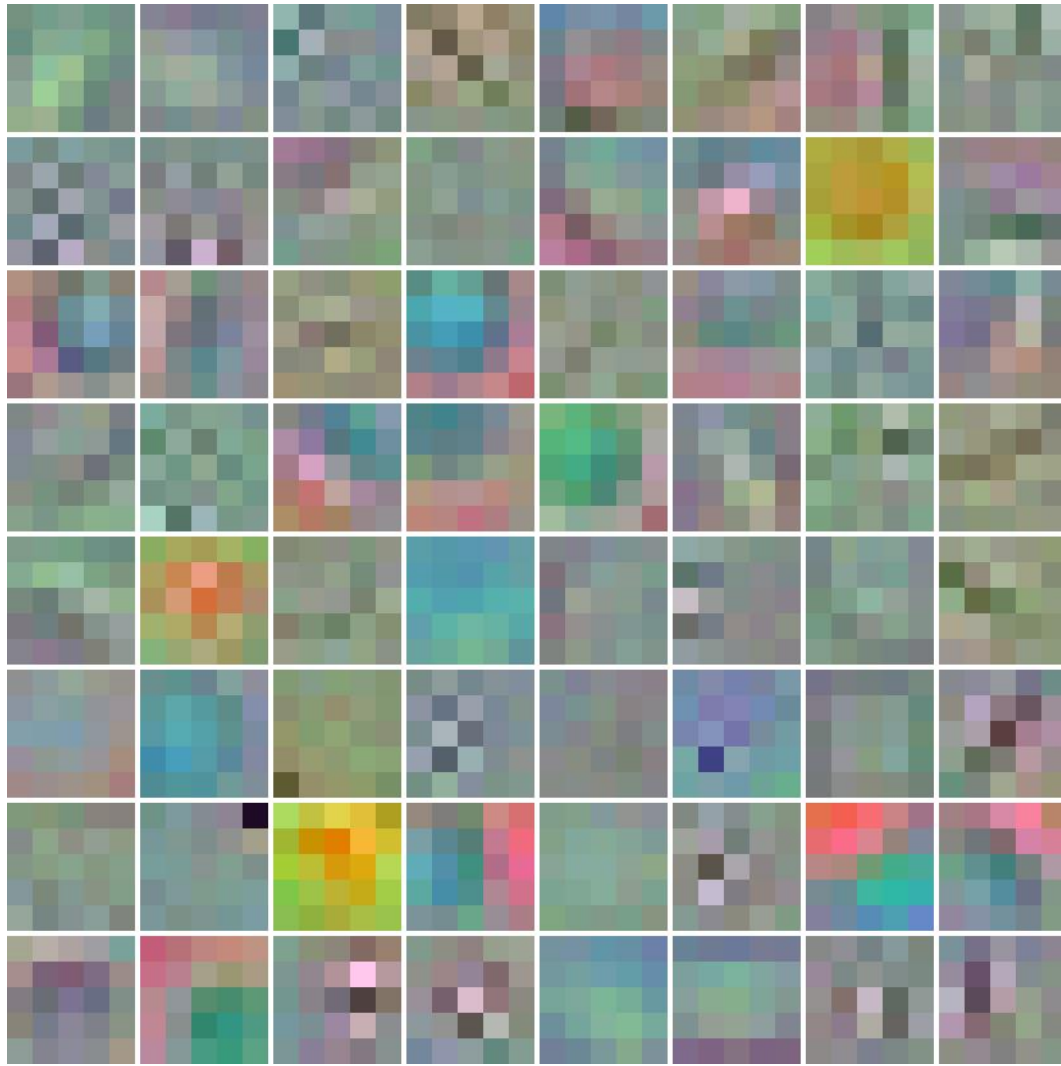


Figure 4.14: First Convolutional Layer Filter Images of Model 5

Table 4.9: Confusion Matrix for CNN Model 5

		Estimated Labels	
		NORMAL	METASTASIS
Known Classes	NORMAL	4906	94
	METASTASIS	1138	3862
Accuracy		0.8768	

4.4.7. Comparison of CNN Architectures

We have constructed six different convolutional neural network structure and we have especially concentrated on the feature extraction stage, i.e. the convolutional layers, max-pooling layers and normalization layers of the network. We have kept fixed the input side and the fully connected layers and softmax layer side in order to make meaningful and consistent inferences. While we are introducing the convolutional neural network structures, we have mentioned about some common approaches in the contemporary CNN architectures. Based on these approaches, we have constructed different architectures and compare them. In Table 4.10, accuracy values obtained with six different CNN architectures are presented. Structures of the models are also given for ease of comparison.

Table 4.10: Accuracy Values Obtained with Six Different CNN Architectures

	Model 0	Model 1	Model 2	Model 3	Model 4	Model 5
Accuracy	0.8683	0.8562	0.8751	0.8728	0.8624	0.8768
Structure	CPNCP(F) ²	(CP) ² (F) ²	(CCP) ² (F) ²	(CCCP) ² (F) ²	(CCCP) ³ (F) ²	(CP) ² (F) ²

In the original model that we have used in CAMELYON16 challenge, the network has all three types of layers in its structure: convolutional layers, max-pooling layers and normalization layers. We have removed the normalization layers from the structures of the remaining five models as it is a common approach recently. Another thing that we have employed in our four models, Model 0 to Model 4, is decreasing the filter sizes to 3x3. Difference among the four models comes from the number of convolutional layers followed by a max-pooling layer and number of repetitions of stacked structure of convolutional layers and max-pooling layer.

When we have compared the CAMELYON16 challenge model (Model 0) and Model 5, we have observed a slight increase in the accuracy of the system. The difference between two models is that Model 5 does not contain normalization layers while the remaining structure is the same for two models. This is consistent with the claim of recent architectures that removing the normalization layers does

not degrade the system performance. On the contrary, it improved the performance in our case.

Performances of Model 1 and Model 5 are close to each other. This shows us that decreasing the filter sizes and increasing the number of convolutional layers gives almost the same accuracy values. Even, it may improve the performance of the system with further training. Although the effective receptive field obtained on the input image at the end of first pooling layer is the same for both of the models, the non-linear transform effect of the ReLU units at the top of the convolutional layers may improve the performance since more non-linear transforms mean richer feature vectors.

Comparison of Model 1 and Model 2 revealed the fact that the number of convolutional layers is important and increasing the number of convolutional layers may improve the performance of the network. The reason of the performance decrease in Model 1 may be that inadequate number of convolutional layers could not reveal the required motifs in the input image as good as the Model 2 did, so the accuracy of the classification decreased.

We have tested the effect of number of convolutional layers one more step further in Model 3. We have increased the convolutional layers per max-pooling layer one more unit and tested the system performance in terms of classification accuracy. We did not observed further increase in the accuracy value of the system. However, we cannot say that increasing the number of convolutional layers further does not improve the system performance since it may contribute to system performance when the model is trained with a larger data set and with more number of epoch values. We can surely say that we need to employ sufficient number of convolutional layers to have a good system performance based on the observations from Model 1 to Model 3. We have observed nearly 2% increase in accuracy from Model 1 to Model 2 and decrease from Model 2 to Model 3 is so small that it can be negligible under these circumstances.

In Model 4, we have repeated three times the three convolutional layer followed by one max-pooling layer structure. In this model, we have increased both the

convolutional layers and the max-pooling layers. This change has two main effects on the network. First, the number of non-linearities through the network has increased so we can extract higher level features. Second, the data dimension fed to the fully connected layers and softmax layer stack has halved in both width and height dimensions, so the feature vector size drops to one fourth of Model 3 at the input of the fully connected layer. When we looked at the accuracy value obtained with this configuration, we have seen that value decreased nearly 1% compared to Model 3. Reason of that drop in accuracy value may be the reduction of the feature vector size at the input of fully connected layer. It may require to increase the number of kernels while increasing the number of max-pooling layers to amortize the feature vector size reduction, so the feature loss.

The observations that we have done throughout the comparison of the models are generally promising in the favor of common approaches in the contemporary convolutional neural network architectures. However, some further analyses are required to validate all of the effects of common approaches used in contemporary CNN architectures.

4.5. Decision Fusion

We have explained the post-processing operations we have conducted on the classification stage outputs. Two important components of the post-processing stage are filters and the erosion disk size. We have checked for three different filters and two different disk sizes the performance of the model on training set of CAMELYON16 Challenge.

4.5.1. Fusion Filters

We have used three different filters to process CNN model outputs in order to obtain slide-based and lesion-based performance measures. All of three filters are chosen in such a way that they give priority to the central pixel and the closest neighborhood of the central pixel. All the filters have common size of 7×7 .

Shape of the first filter with its weights is given in Table 4.11. The filter is constructed according to Equation (4.2).

$$W_i = M - D_i \tag{4.2}$$

$$M = \frac{\text{Filter Size}+1}{2} = \frac{7+1}{2} = 4$$

W_i : Weight of the i^{th} pixel

D_i : Distance of the i^{th} pixel to the center pixel

Table 4.11: First Decision Fusion Filter (FILTER1) Weights

	1	2	3	4	5	6	7
1	1	1	1	1	1	1	1
2	1	2	2	2	2	2	1
3	1	2	3	3	3	2	1
4	1	2	3	4	3	2	1
5	1	2	3	3	3	2	1
6	1	2	2	2	2	2	1
7	1	1	1	1	1	1	1

Second fusion filter is also constructed in a similar way, but this time the number of pixels at the same distance from the central pixel is taken in to account. The filter is constructed according to Equation (4.3) and the filter is shown in Table 4.12.

$$W_i = \frac{1}{N_i} \tag{4.3}$$

N_i : Number of pixels at the same distance D_i from central pixel

W_i : Weight of the i^{th} pixel

D_i : Distance of the i^{th} pixel to the center pixel

Table 4.12: Second Decision Fusion Filter (FILTER2) Weights

	1	2	3	4	5	6	7
1	0.04	0.04	0.04	0.04	0.04	0.04	0.04
2	0.04	0.06	0.06	0.06	0.06	0.06	0.04
3	0.04	0.06	0.13	0.13	0.13	0.06	0.04
4	0.04	0.06	0.13	1.00	0.13	0.06	0.04
5	0.04	0.06	0.13	0.13	0.13	0.06	0.04
6	0.04	0.06	0.06	0.06	0.06	0.06	0.04
7	0.04	0.04	0.04	0.04	0.04	0.04	0.04

Third decision fusion filter is a Gaussian like filter that is sampled at discrete instances of standard normal distribution. Weights of the filter are obtained from Equation (4.4) and the corresponding filter is given in Table 4.13.

$$W_i = \frac{1}{2\pi} e^{-\frac{D_i^2}{2}} \quad (4.4)$$

W_i : Weight of the i^{th} pixel

D_i : Distance of the i^{th} pixel to the center pixel

Table 4.13: Third Decision Fusion Filter (FILTER3) Weights

	1	2	3	4	5	6	7
1	0.004	0.004	0.004	0.004	0.004	0.004	0.004
2	0.004	0.054	0.054	0.054	0.054	0.054	0.004
3	0.004	0.054	0.242	0.242	0.242	0.054	0.004
4	0.004	0.054	0.242	0.399	0.242	0.054	0.004
5	0.004	0.054	0.242	0.242	0.242	0.054	0.004
6	0.004	0.054	0.054	0.054	0.054	0.054	0.004
7	0.004	0.004	0.004	0.004	0.004	0.004	0.004

4.5.2. Performance Comparison of Post-processing Operations

For three different fusion filters and two different erosion disk sizes (DISK1: `strel('disk',1)` and DISK2: `strel('disk',2)` in MATLAB), the post processing operations were conducted over the outputs of the CAMELYON16 Challenge CNN architecture. Performance measures of the challenge were used in comparison, i.e. the area under the ROC curve for slide-based evaluation category and average score value at some pre-defined points on the FROC curve. The performance comparison is given in Table 4.14. Best results were obtained with (FILTER3, DISK1) and (FILTER1, DISK2) combinations in lesion based and slide based evaluation categories, respectively. We have preferred to use (FILTER3, DISK1) combination in challenge for better results in lesion based category.

Table 4.14: Performance Comparison of Fusion Filters and Erosion Disk Sizes

	Area Under ROC Curve	Average Score on FROC Curve
FILTER1, DISK1	0.9359	0.5178
FILTER2, DISK1	0.9282	0.5219
FILTER3, DISK1	0.9259	0.5349
FILTER1, DISK2	0.9514	0.4918
FILTER2, DISK2	0.9402	0.4966
FILTER3, DISK2	0.9380	0.5079

4.6. Performance of Proposed Approach

We have trained the CAMELYON16 Challenge model with a large dataset that was introduced in ‘Proposed Approach’ section and used this model in the classification stage of the proposed approach. Then, we have processed the outputs of that model in post-processing stage with (FILTER3, DISK1) combination to obtain the performance measure values required by the challenge.

We have obtained the confusion matrix given in Table 4.15 with the final model we have trained. Both training and the evaluation were done over the training set while we were obtaining the given confusion matrix because of the lack of the ground truth data for test set in the challenge. Therefore, model has seen some of the samples during the training process. However, when we compared the size of training dataset, 480000 samples in total, with the size of classified samples set, 6039875 in total, two sets can be said to be isolated. We have achieved accuracy value of 0.948 with this model.

Table 4.15: Confusion Matrix of Final Trained Model over Training Set

		Estimated Labels	
		NORMAL	METASTASIS
Known Classes	NORMAL	5498523	305718
	METASTASIS	8041	227593
Accuracy		0.9480	

We have visualized the classification performance of the model with color coded evaluation images of WSIs. Color coding scheme is given in Table 4.16. Moreover,

we have given one example color coded evaluation image of WSIs from both classes. We have also shown some image patches that are classified as ‘False Positive’ for those WSIs. In Figure 4.15 and Figure 4.16, color coded evaluation image and image patches classified as false positive are given for Tumor_089 WSI, respectively. Similarly, two components for Normal_066 WSI are presented in Figure 4.17 and Figure 4.18, respectively.

Table 4.16: Color Coding Scheme in Evaluation Images

		Estimated Labels	
		NORMAL	METASTASIS
Known Classes	NORMAL	Blue	Yellow
	METASTASIS	Red	Green
	BOUNDARY or BACKGROUND	Turquoise Blue	Pink

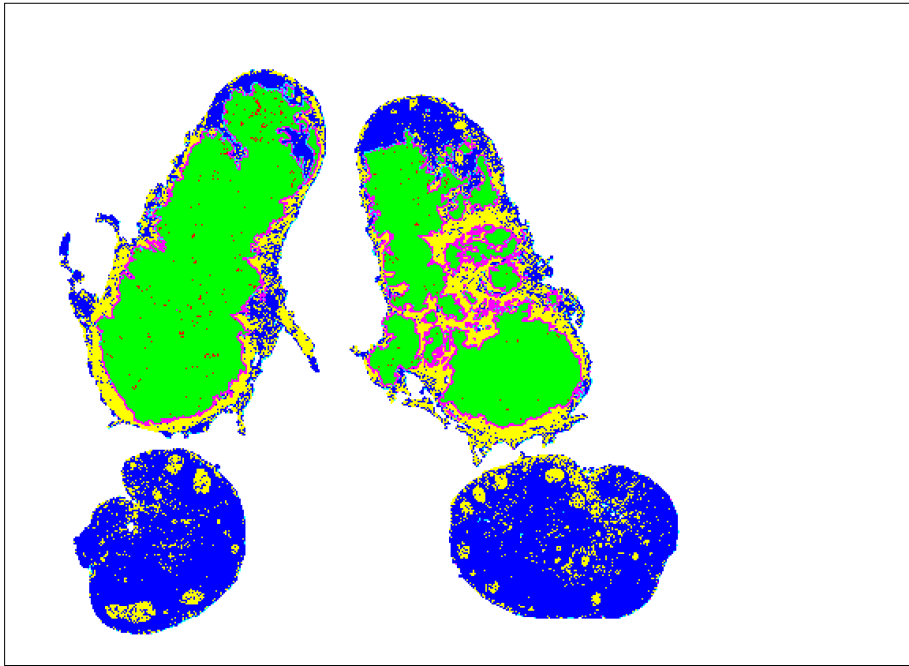


Figure 4.15: Color Coded Evaluation Image of Tumor_089 WSI

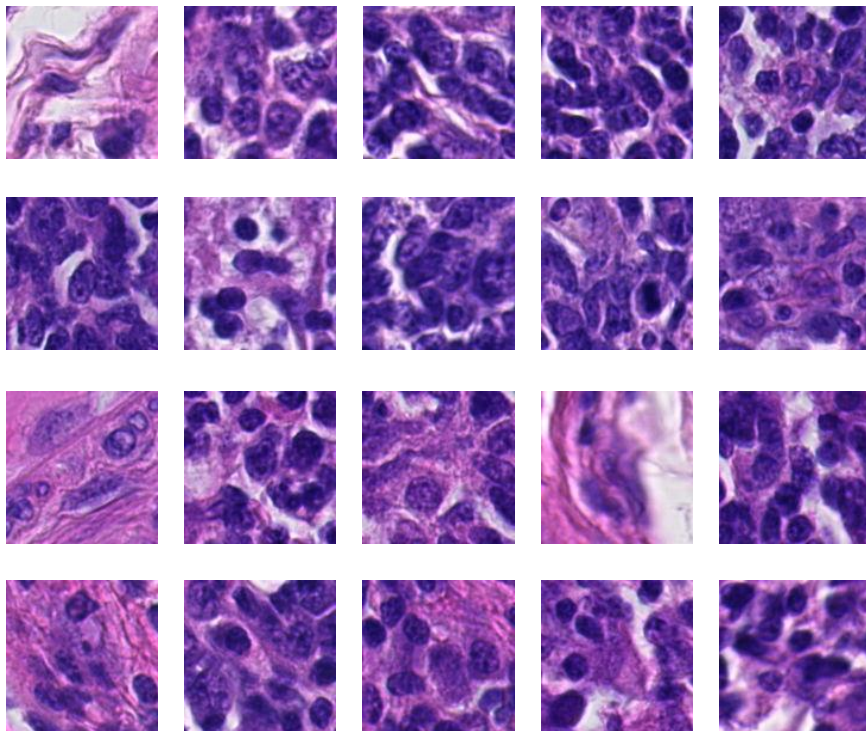


Figure 4.16: False Positive Image Patches from Tumor_089 WSI

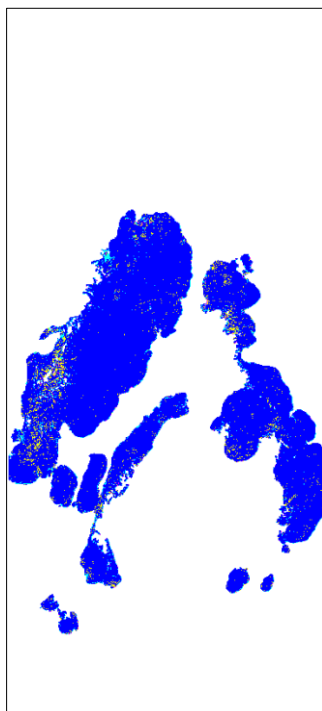


Figure 4.17: Color Coded Evaluation Image of Normal_066 WSI

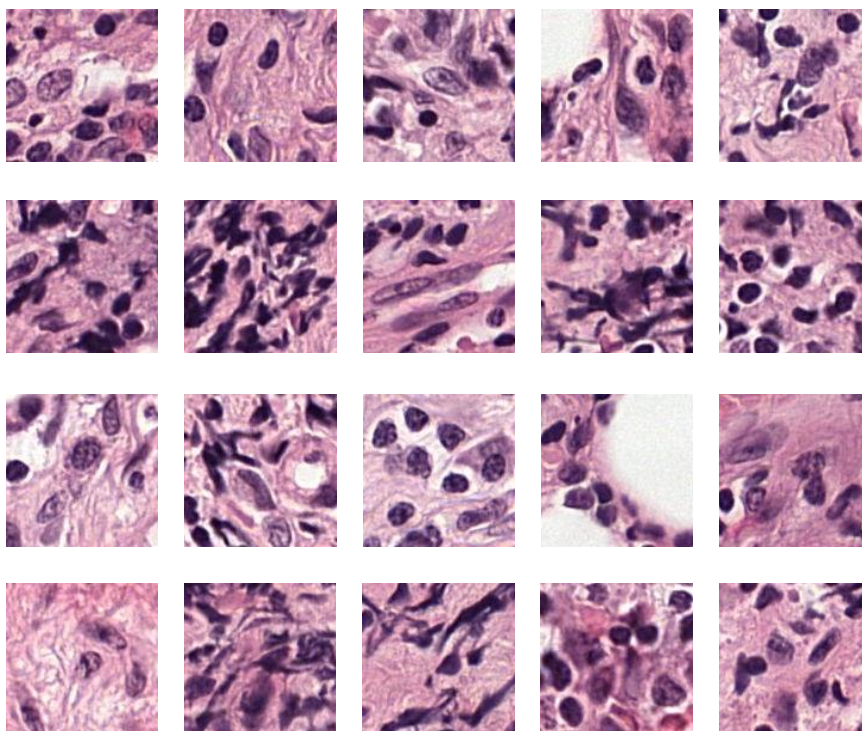


Figure 4.18: False Positive Image Patches from Normal_066 WSI

We have also tested our proposed approach on the training set according to the required performance measures by CAMELYON16 Challenge. We have drawn

ROC and FROC curves of our approach for training set in Figure 4.19 and Figure 4.20, respectively. We have obtained area under the receiver operating characteristic curve (AUC) value of 0.9259 and average score (FROC Score) of 0.5349 at 0.25, 0.5, 1, 2, 4, 8 average numbers of false positive locations on the FROC curve. Moreover, after the results of the challenge were declared, performances of our approach on test set were published in the challenge website. They are also given in Figure 4.21 and Figure 4.22.

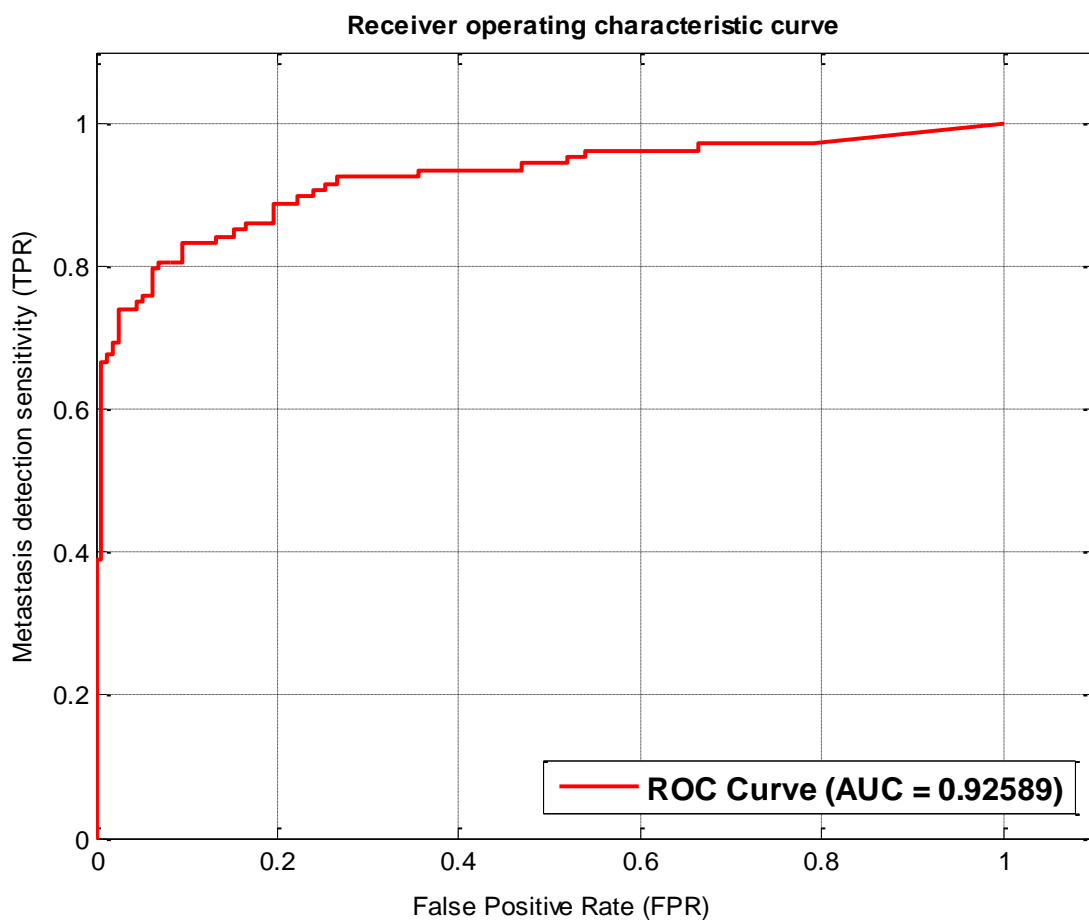


Figure 4.19: ROC Curve of Proposed Approach on Training Set

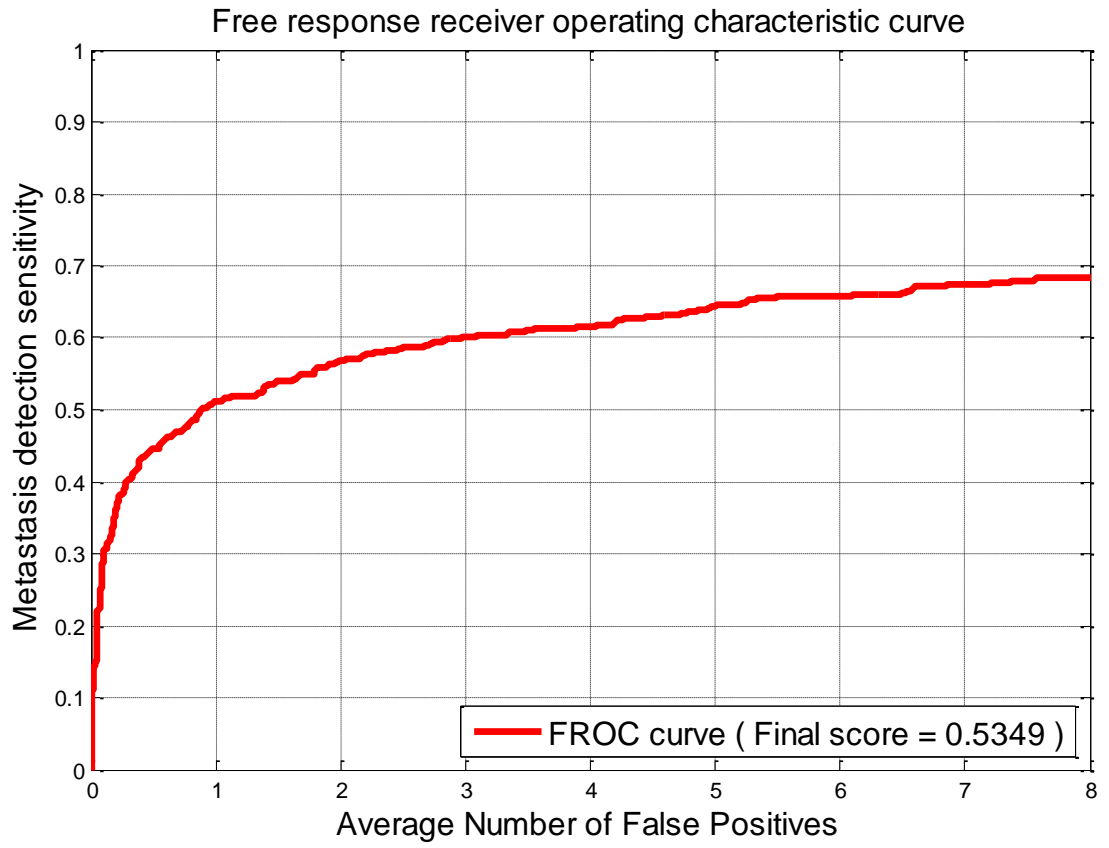


Figure 4.20: FROC Curve of Proposed Approach on Training Set

ROC curve - Health Sciences Middle East Technical University

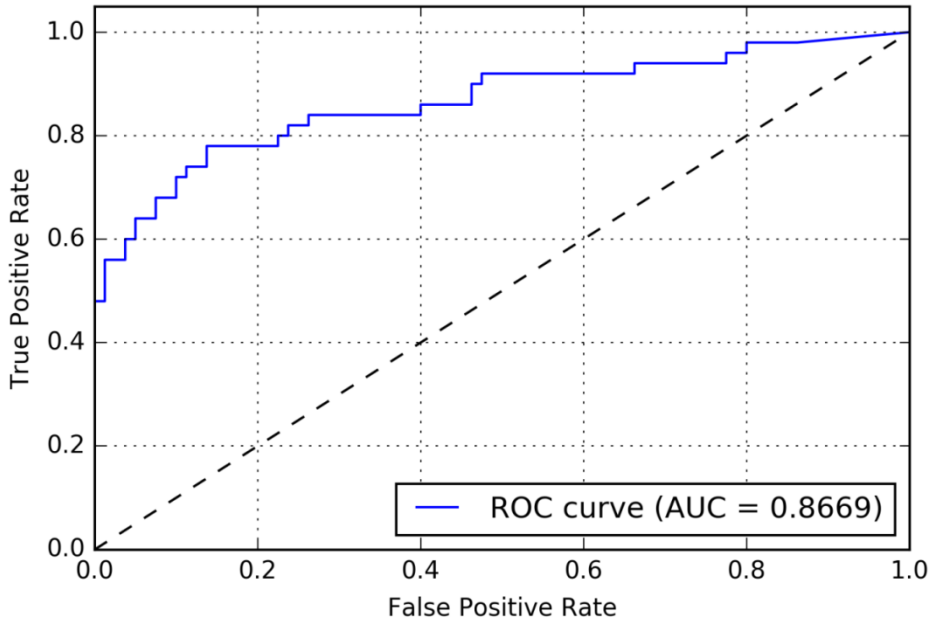


Figure 4.21: ROC Curve of Proposed Approach on Test Set

FROC curve - Health Sciences Middle East Technical University

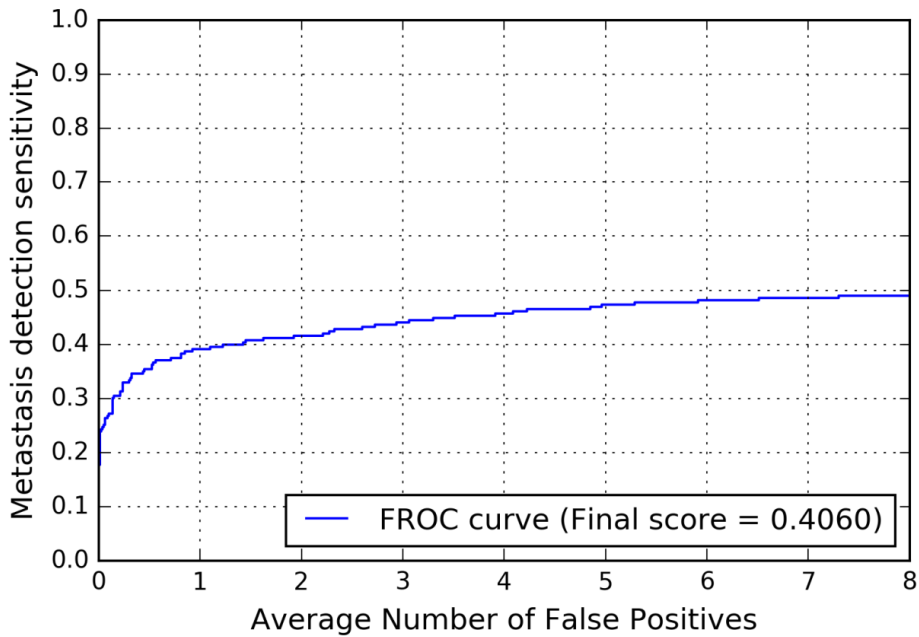


Figure 4.22: FROC Curve of Proposed Approach on Test Set

Finally, the performance comparison of the proposed approach on test set among the top 10 ranked teams' performances in slide based evaluation category is given in

Table 4.17: Performance Comparison Among Top 10 Ranked Teams in Slide Based Category. Moreover, performance measure plots of five leading teams in both categories are given in Appendix A to Appendix F. In slide based category, AUC performance of our proposed approach is not so bad and it is promising for further improvements. However, lesion based performance of our approach (FROC score) is not compatible enough when it is compared with top ranked team's performance. Hence, some radical improvements are required at that stage of the proposed approach. This stage should also be trainable, i.e. it should be included in the training process for better performance.

Table 4.17: Performance Comparison Among Top 10 Ranked Teams in Slide Based Category

Rank	Team	AUC	FROC Score
1	Harvard Medical School and Massachusetts Institute of Technology, USA	0.9250	0.7051
2	EXB Research and Development co., Germany	0.9173	0.5192
3	Independent participant - Quincy Wong, Germany	0.8680	0.3964
4	Middle East Technical University, Departments of EEE, NSNT and HS, Turkey	0.8669	0.4060
5	NLP LOGIX co., USA	0.8332	0.4040
6	University of Toronto, Electrical and Computer Engineering, Canada	0.8181	0.3615
7	The Warwick-QU Team, United Kingdom	0.7999	0.3155
8	Radboud University Medical Center (DIAG), Netherlands	0.7828	0.5761
9	HTW-BERLIN, Germany	0.7717	0.1770
10	University of Toronto, Electrical and Computer Engineering, Canada	0.7666	0.3944

CHAPTER 5

CONCLUSION

In this dissertation, a convolutional neural networks based system is developed to detect and localize the metastasis regions automatically on whole slide images of lymph node excisions taken from breast cancer patients. The proposed approach mainly consists of three steps: pre-processing, classification and post-processing. In the pre-processing stage, the lymph node sections of the whole slide images are extracted from background and prepared to be used in the construction of the supervised training dataset of the classification stage. In order to classify the input images, a convolutional neural network structure is designed and a dataset is constructed, which is used to train the developed network in supervised manner. After design and training of the network, all the patches that are extracted from the lymph node sections of the whole slide image are fed to the network and the class labels are obtained in the classification stage. The last stage is the post-processing stage that is used for decision fusion. We have proposed a sliding window based filtering system to fuse the label data of patches on whole slide image in order to obtain metastasis region level and slide level performance measure values.

Performance analyses of different network architectures are analyzed in this study to be used in the classification stage of the proposed solution. Moreover, different filter structures are used in post-processing stage to obtain better performance values. The comparison of network structures and the filter performances are done throughout this dissertation. Lastly, the performance comparison analysis at system level is done for the best 10 solution submitted to the CAMLEYON16 challenge is presented. The performance of the solution proposed in this thesis is promising for future development when it is compared with other solutions.

In the proposed solution, training dataset of the classification stage CNN architectures was constructed from Normal and Tumor WSIs with nearly equal

number of ‘normal’ and ‘tumor’ samples. However, all normal samples are taken from Normal WSIs, i.e. we have not chosen any normal image samples from the normal regions of Tumor WSIs. Choosing normal samples from Tumor WSIs may also contribute to the system performance, so it should be done in the future work. Furthermore, we have used only the image samples completely coming from the metastasis regions of Tumor WSIs as ‘tumor’ samples. Normal – Tumor boundary samples in Tumor WSIs were not included in the training dataset and this may cause the model to misclassify the boundary samples in the dataset. The solutions proposed by some top ranked teams contain the boundary images as ‘tumor’ samples and it is reasonable since it can contribute to generalization ability of the model, so this is another important issue that must be considered in future work.

We have used Layer 7 and Layer 2 images during the operations of our proposed solution to exploit short training and processing time advantages of working with small size images at the expense of low resolution quality. We have used 64x64 images from Layer 2 in the training of the model; however, teams ranked above our team in the challenge have used 256x256 images from Layer 0. Although corresponding physical sizes in both of the choices are the same, we may lose some of the details due to resolution constraints in Layer 2. Therefore, it is worth to try to use Layer 0 image samples in future work.

Furthermore, training the model in a single pass may not be the best choice. In other words, we have constructed training dataset at the beginning and use this dataset to train the model. Then, we have used the trained model to classify CAMELYON16 Test set. This may not be sufficient to obtain the best performance from the model. We have already separated a set of samples from the training set to validate the performance of the trained model (hold-out cross validation). While classifying this set, some of the samples are misclassified. By adding these samples to the training dataset and training the model further with new enlarged dataset should contribute to performance of the model. This approach is used in the literature and by some of the teams in the challenge also. Therefore, this additional step is important to improve the weak sides of the model and must be used in model training.

In this study, we have trained six different CNN models for the classification stage. We have analyzed the effects of different architecture elements such as filter sizes, number of convolutional layers, effect of normalization layers, etc. We have observed that while we are using two convolutional layers, 5x5 filters give better results than 3x3 filters. This is mainly due to the fact that the large receptive fields in 5x5 filters can capture more distinctive features than smaller receptive field 3x3 filters in two layer structure, which can be said to be shallow. However, when we have employed four convolutional layers with 3x3 filters and obtained the same effective receptive field area on the input image, performance of the system is almost the same. Moreover, it may even be improved with further training, so 3x3 filters (small size filters) in deep architectures are better in CNN structures since they can capture more complex features by using more non-linear transforms by employing a simpler architecture.

We have also observed that increasing the number of convolutional layers up to a point may contribute to the system performance but it gets saturated after some point. We have obtained better results with 4 and 6 convolutional layers and the performances of these two are close to each other. Increasing the number of convolutional layers to 9 has not increased the performance but decreased. This may be caused by additional pooling layer coming with last three convolutional layers. Addition of one more pooling layer with stride 2 decreases the feature vector size fed to the fully connected layer by four, so we may lose some of the features. Additional layers may require some further modifications on the structure of the network, such as increasing the number of filters 4 times to compensate reduction in the feature vector size.

One of the most important insights gained during this study is that initialization of the weights of the networks is an important and critical issue. We have observed that if we have initialized all of the models with the same weights, some of the models learn nothing. Data and structure dependent initialization of weights in the models changes the situation completely. Weights of the models Model 1 to Model 5 are initialized similar to method given in (Glorot & Bengio, 2010). Furthermore, some new publications related to data-dependent weight initialization of networks are

emerging recently. Two of them are given in (Krahenbühl, Doersch, Donahue, & Darrell, 2016) and (Mishkin & Matas, 2016).

Lastly, the classification stage of the proposed solution is based on a convolutional neural network, so further development both in the structure and the training of the network can be done in the future. Moreover, improvement in the post-processing stage to obtain better fusion results, so performances, is also possible. However, what is more promising than improving two stages separately is handling classification and decision fusion both in the same network architecture. In the literature the examples of such architectures emerge recently (Ronneberger, Fischer, & Brox, 2015). Both the classification and the localization of metastasis regions on a whole slide image can be done in one convolutional neural network.

REFERENCES

- Ali, S., & Madabhushi, A. (2012). An Integrated Region-, Boundary-, Shape-Based Active Contour for Multiple Object Overlap Resolution in Histological Imagery. *IEEE Transactions on Medical Imaging*, 31(7), 1448-1460.
- Al-Jabani, S., Huisman, A., & Van Diest, P. (2011). Digital pathology: current status and future perspectives. *Histopathology*, 61(1), 1-9.
- Alper, M., Bayık, P., Demirhan, B., Güler, G., Harorlu, F., Özer, E., et al. (2011). *Ülkemizdeki Sağlık Hizmeti Sunumunda Patoloji Uzmanlık Alanının Durumu ve Öneriler*. Ankara: T.C. Sağlık Bakanlığı Patoloji Bilimsel Komisyonu.
- Bejnordi, B. E. (2016). *CAMELYON16 ISBI Challenge on Cancer Metastases Detection in Lymph Node*. Retrieved June 15, 2016, from CAMELYON16 Website: <http://camelyon16.grand-challenge.org/results/>
- Berseth, M. (2016). *CAMELYON16 ISBI Challenge Results Page*. Retrieved 2016, from <http://camelyon16.grand-challenge.org/PerTeamResult/?id=LOGIX>
- Breastcancer.org. (2015). *What is breast cancer?* Retrieved June 15, 2016, from Breast Cancer Organization Website: http://www.breastcancer.org/symptoms/understand_bc/what_is_bc
- CAMELYON16. (2015). *CAMELYON16 Data*. Retrieved July 24, 2016, from CAMELYON16: ISBI Challenge on Metastasis Detection in Lymph Node: <http://camelyon16.grand-challenge.org/data/>
- Cancer.org. (2014). *What is breast cancer?* Retrieved June 15, 2016, from American Cancer Society Website: <http://www.cancer.org/cancer/breastcancer/detailedguide/breast-cancer-what-is-breast-cancer>

- Cireşan, D. C., Giusti, A., Gambardella, L. M., & Schmidhuber, J. (2013). Mitosis Detection in Breast Cancer Histology Images with Deep Neural Networks. *Proc. Med. Image Comput. Comput.-Assisted Intervention*, 411-418.
- Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *Proceedings of 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*.
- Halıcı, U., Öner, M. Ü., & Çetin Atalay, R. (2016). *CAMELYON16 ISBI Challenge Results Page*. Retrieved 2016, from <http://camelyon16.grand-challenge.org/PerTeamResult/?id=HALICI>
- Hass, C., Sanchez, U., Vasilev, U., Mey, T., & Bruni, E. (2016). *CAMELYON16 ISBI Challenge Results Page*. Retrieved 2016, from <http://camelyon16.grand-challenge.org/PerTeamResult/?id=EXBdev>
- Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation* 18, 1527-1554.
- IARC. (2013). *GLOBOCAN 2012: Estimated Cancer Incidence, Mortality and Prevalance Worldwide in 2012*. Retrieved June 19, 2016, from International Agency for Research on Cancer Website: http://globocan.iarc.fr/Pages/fact_sheets_cancer.aspx
- Jassim, F. A., & Altaani, F. H. (2013, May). Hybridization of OTSU Method and Median Filter for Color Image Segmentation. *International Journal of Soft Computing and Engineering*, 3(2), 69-74.
- Karpathy, A. (2016). *Convolutional Neural Networks (CNNs / ConvNets)*. Retrieved August 7, 2016, from CS231n Convolutional Neural Networks for Visual Recognition: <http://cs231n.github.io/convolutional-networks/>
- Kokkat, T. J., Patel, M. S., McGarvey, D., LiVolsi, V. A., & Baloch, Z. W. (2013). Archived Formalin-Fixed Paraffin-Embedded (FFPE) Blocks: A Valuable

- Underexploited Resource for Extraction of DNA, RNA, and Protein. *Biopreservation and Biobanking*, 11(2), 101-106.
- Krahenbühl, P., Doersch, C., Donahue, J., & Darrell, T. (2016). Data-dependent Initialization of Convolutional Neural Networks. *Conference Paper at ICLR 2016*.
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *NIPS 2012: Neural Information Processing Systems*. Lake Tahoe, Nevada.
- LeCun, Y. (2016). The AI Arms Race. *WIRED Business Conference 2016*. Retrieved July 25, 2016, from <https://www.youtube.com/watch?v=gKF-1N7biSM>
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep Learning. *Nature*, 521(7553), 436-444.
- LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., et al. (1990). Handwritten Digit Recognition with a Back-Propagation Network. *In Proc. Advances in Neural Information Processing Systems*, (pp. 396-404).
- LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. *Proc. IEEE* 86, 2278-2324.
- Malon, C. D., & Cosatto, E. (2013). Classification of mitotic figures with convolutional neural networks and seeded blob features. *Journal of Pathology Informatics*, 4(1), 9.
- Malon, C., Brachtel, E., Cosatto, E., Graf, H. P., Kurata, A., Kuroda, M., et al. (2012). Mitotic Figure Recognition: Agreement Among Pathologists and Computerized Detector. *Analytical Cellular Pathology*, 35(2), 97-100.
- Martin, D. T. (2016). *CAMELYON16 ISBI Challenge Results Page*. Retrieved 2016, from

<http://camelyon16.grand-challenge.org/PerTeamResult/?id=Radboudumc>

McCaffrey, J. D. (2013, November 5). *Why You Should Use Cross-Entropy Error Instead Of Classification Error Or Mean Squared Error For Neural Network Classifier Training*. Retrieved June 19, 2016, from James D. McCaffrey Blog: <https://jamesmccaffrey.wordpress.com/2013/11/05/why-you-should-use-cross-entropy-error-instead-of-classification-error-or-mean-squared-error-for-neural-network-classifier-training/>

Mishkin, D., & Matas, J. (2016). All You Need is a Good Init. *Conference Paper at ICLR 2016*.

Otsu, N. (1979). A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1), 62-66.

Robboy, S., Weintraub, S., Horvath, A., Jensen, B., Alexander, C., Fody, E., et al. (2013). Pathologist Workforce in the United States: I. Development of a Predictive Model to Examine Factors Influencing Supply. *Archives of Pathology & Laboratory Medicine*, 137(12), 1723-1732.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional Neural Networks for Biomedical Image Segmentation. *CoRR*.

Rosenblatt, F. (1957). *The Perceptron - A Perceiving and Recognizing Automaton*. Technical Report, Cornell Aeronautical Laboratory.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323, 533-536.

Selfridge, O. G. (1958). Pandemonium: a paradigm for learning. *Mechanisation of Thought Processes: Proceedings of a Symposium Held at the National Physical Laboratory*, (pp. 513-526). London.

Springenberg, J. T., Dosovitskiy, A., Brox, T., & Riedmiller, M. A. (2014). Striving for Simplicity: The All Convolutional Net. *CoRR*, 1412.6806.

- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., et al. (2015). Going Deeper with Convolutions. Retrieved 2016, from <http://arxiv.org/pdf/1409.4842v1.pdf>
- TensorFlow. (n.d.). *TensorFlow CIFAR-10 Model*. Retrieved March 29, 2016, from TensorFlow.org Website: https://www.tensorflow.org/versions/r0.10/tutorials/deep_cnn/index.html#cif-ar-10-model
- Veta, M., Pluim, J. P., van Diest, P. J., & Viergever, M. A. (2014, May). Breast Cancer Histopathology Image Analysis: A Review. *IEEE Transactions on Biomedical Engineering*, 61(5), 1400-1411.
- Wang, D., Khosla, A., Gargeya, R., Irshad, H., & Beck, A. (2016). *CAMELYON16 ISBI Challenge Results Page*. Retrieved 2016, from http://camelyon16.grand-challenge.org/PerTeamResult/?id=BIDMC_CSAIL
- WHO. (2014). *Global Health Estimates (GHE) 2014: Deaths by age, sex and cause*. Retrieved June 19, 2016, from World Health Organization Website: http://www.who.int/healthinfo/global_burden_disease/estimates/en/index1.html
- Wong, Q. (2016). *CAMLEYON16 ISBI Challenge Results Page*. Retrieved 2016, from <http://camelyon16.grand-challenge.org/PerTeamResult/?id=quincy>
- Xu, J., Janowczyk, A., Chandran, S., & Madabhushi, A. (2010). A Weighted Mean Shift, Normalized Cuts Intitialized Color Gradient Based Geodesic Active Contour Model: Applications to Histopathology Image Segmentation. *Proc. of SPIE Med. Imag.*, 7623.
- Zeiler, M. D., & Fergus, R. (2013). Visualizing and Understanding Convolutional Networks. *CoRR*, 1311.2901.

APPENDIX A

PERFORMANCE MEASURE PLOTS OF HARVARD MEDICAL SCHOOL AND MASSACHUSETTS INSTITUTE OF TECHNOLOGY TEAM

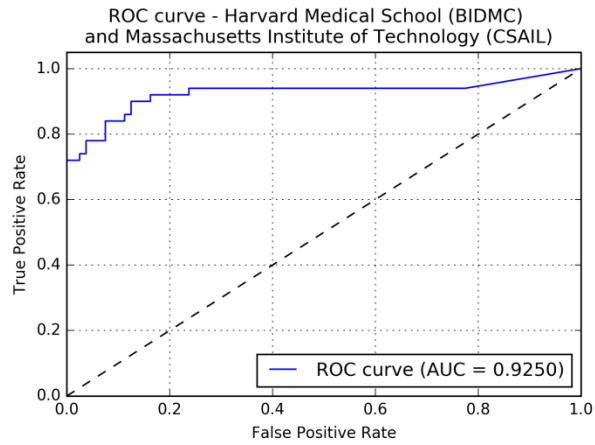


Figure A.1: ROC Curve of Harvard Medical School and Massachusetts Institute of Technology Team

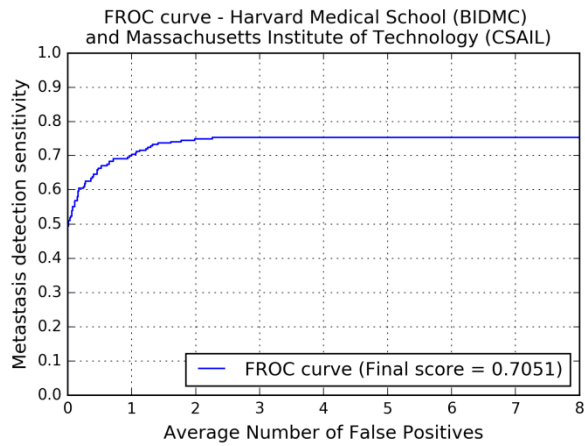


Figure A.2: FROC Curve of Harvard Medical School and Massachusetts Institute of Technology Team

APPENDIX B

PERFORMANCE MEASURE PLOTS OF EXB RESEARCH AND DEVELOPMENT TEAM

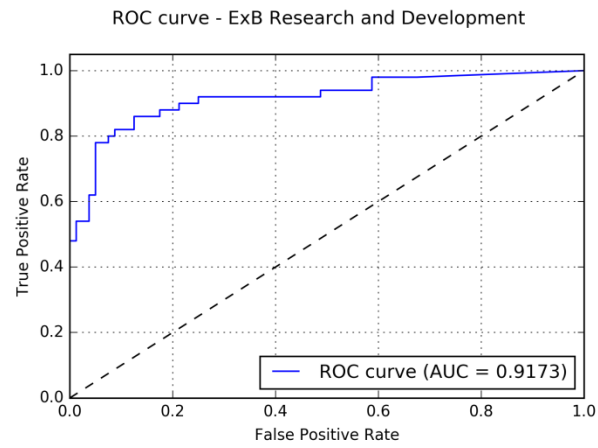


Figure B.1: ROC Curve of ExB Research and Development

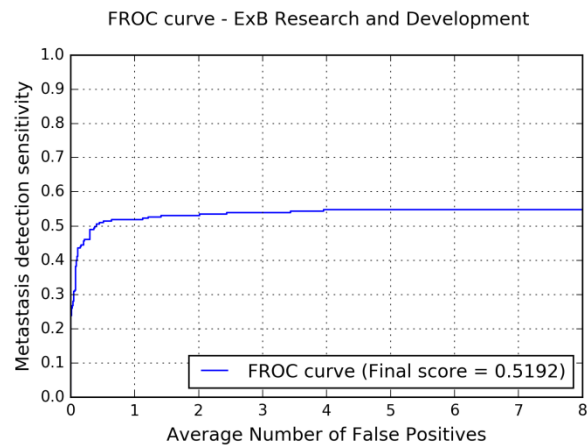


Figure B.2: FROC Curve of ExB Research and Development

APPENDIX C

PERFORMANCE MEASURE PLOTS OF INDIVIDUAL PARTICIPANT QUINCY WONG

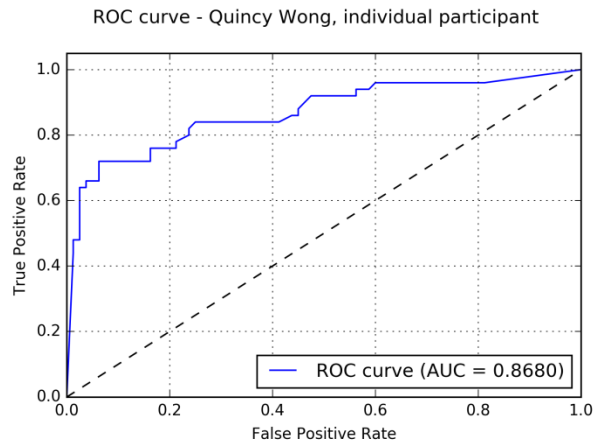


Figure C.1: ROC Curve of Individual Participant Quincy Wong

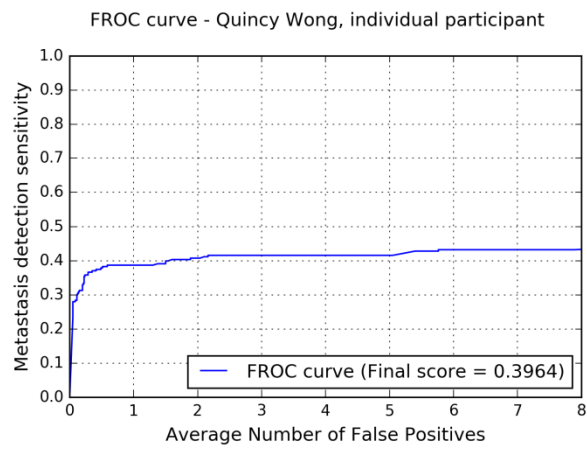


Figure C.2: FROC Curve of Individual Participant Quincy Wong

APPENDIX D

PERFORMANCE MEASURE PLOTS OF METU TEAM

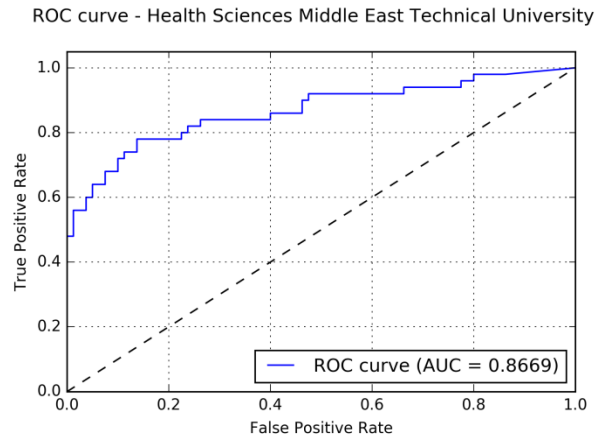


Figure D.1: ROC Curve of METU Team

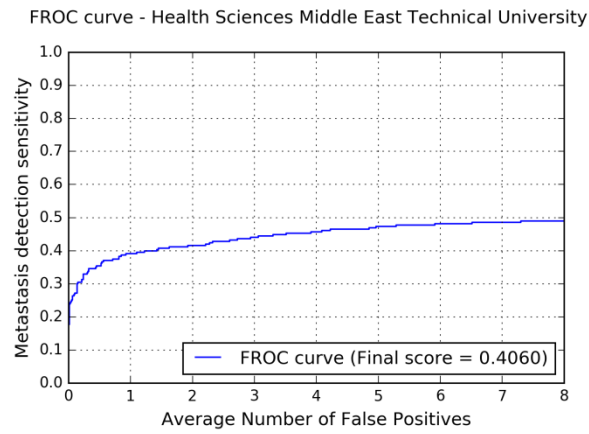


Figure D.2: FROC Curve of METU Team

APPENDIX E

PERFORMANCE MEASURE PLOTS OF NLP LOGIX TEAM

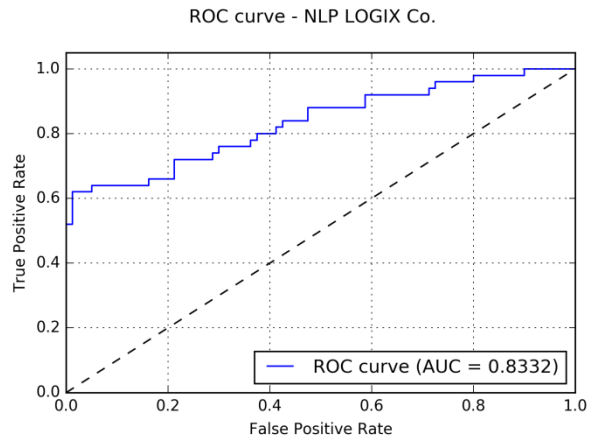


Figure E.1: ROC Curve of NLP LOGIX Team

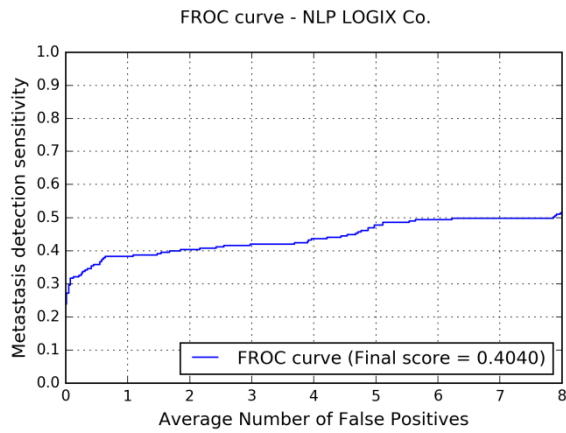


Figure E.2: FROC Curve of NLP LOGIX Team

APPENDIX F

PERFORMANCE MEASURE PLOTS OF RADBOUD UNIVERSITY MEDICAL CENTER TEAM

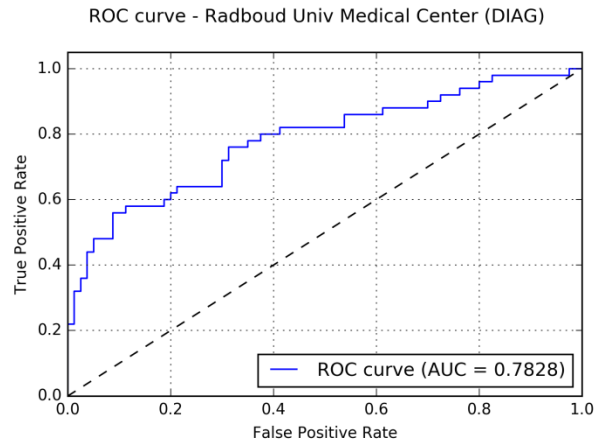


Figure F.1: ROC Curve of Radboud University Medical Center Team

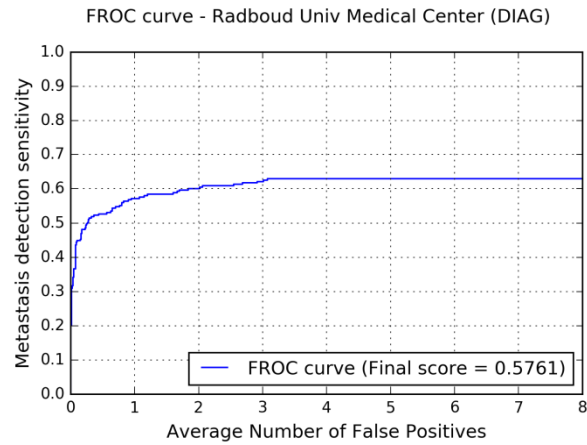


Figure F.2: FROC Curve of Radboud University Medical Center Team

APPENDIX G

TRAINING HISTOGRAMS OF CAMELYON16 CHALLENGE MODEL

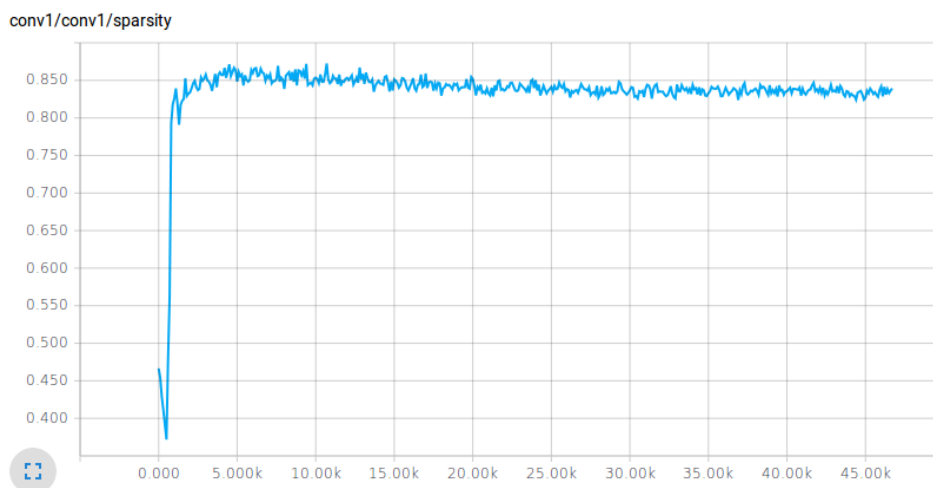


Figure G.1: Convolutional Layer 1 Sparsity

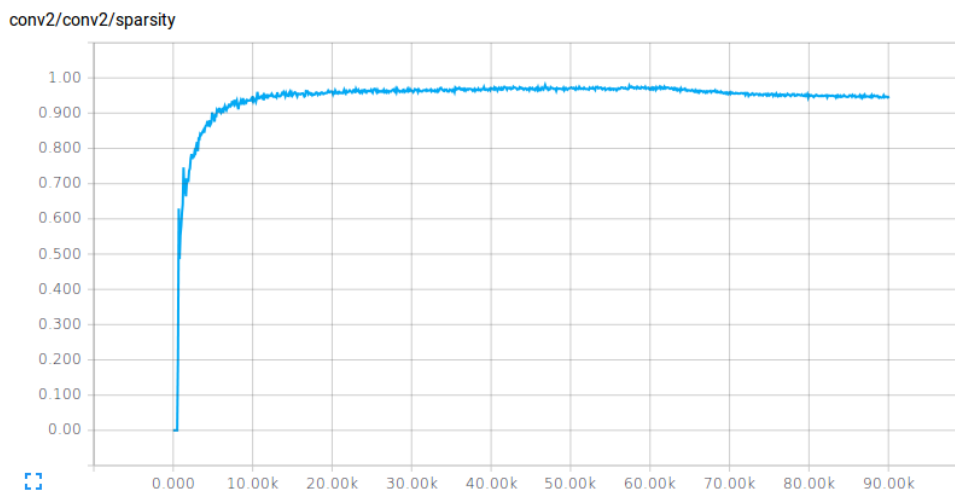


Figure G.2: Convolutional Layer 2 Sparsity

local3/local3/sparsity

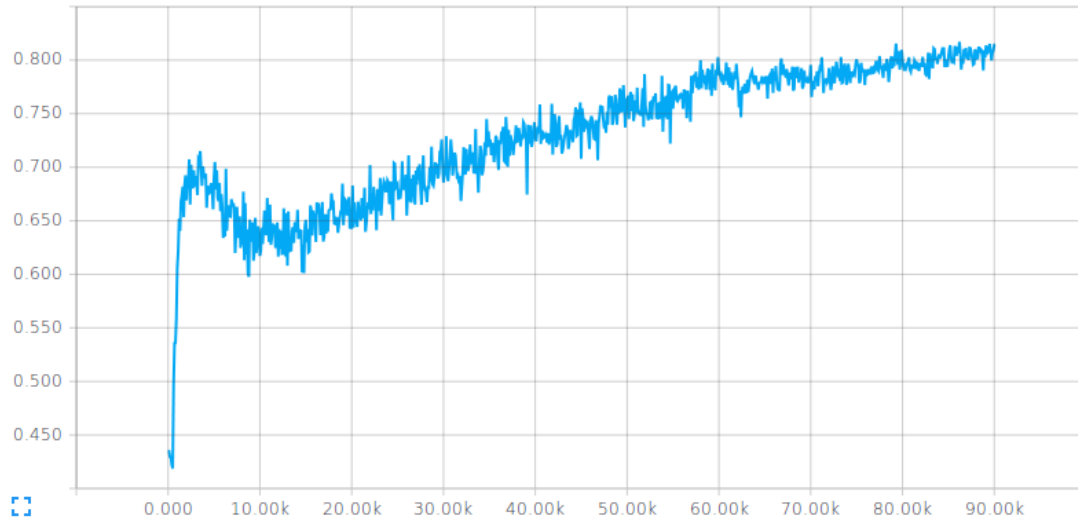


Figure G.3: Fully Connected Layer 1 Sparsity

local4/local4/sparsity

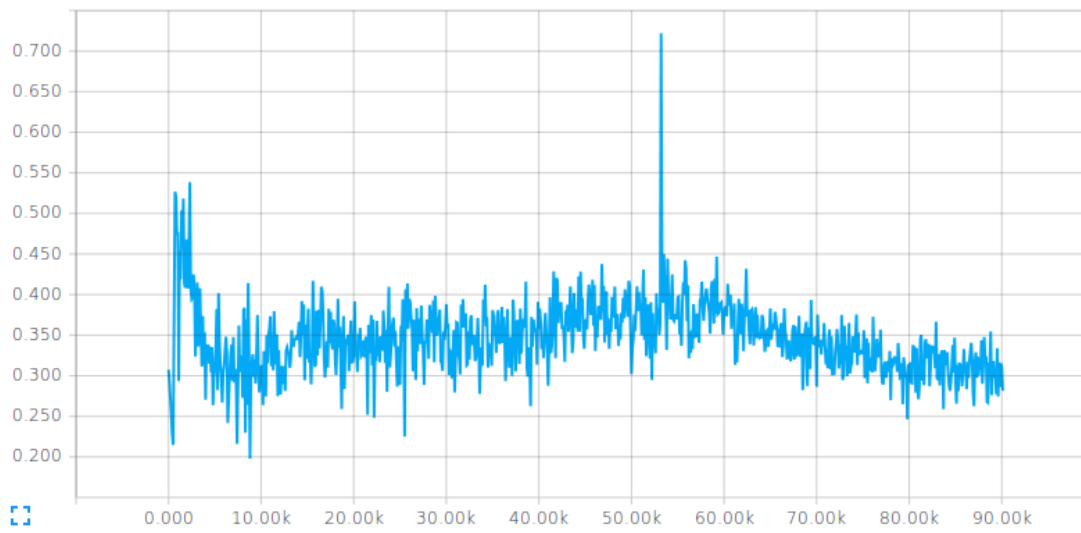


Figure G.4: Fully Connected Layer 2 Sparsity

cross_entropy

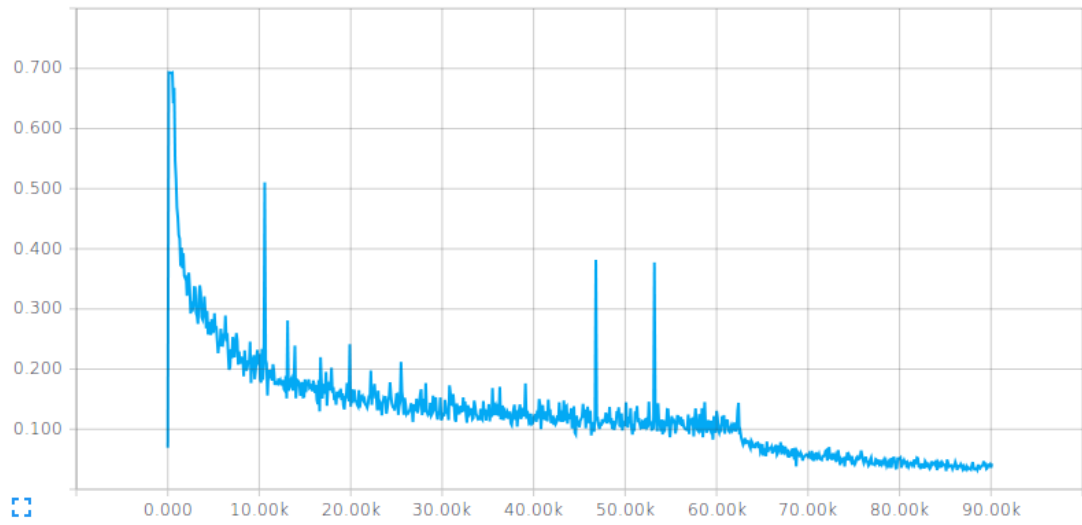


Figure G.5: Cross Entropy Error per Image Plot

total_loss_1

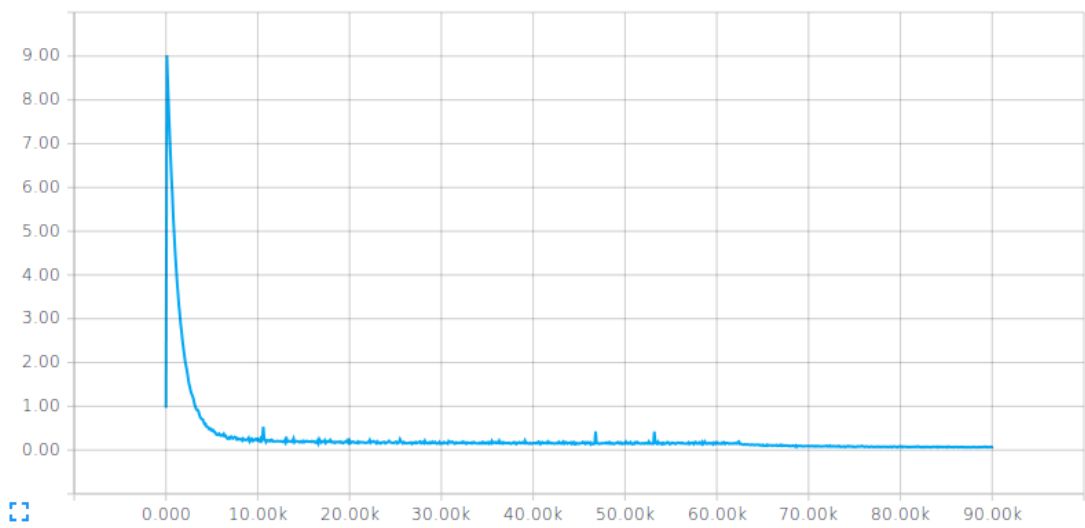
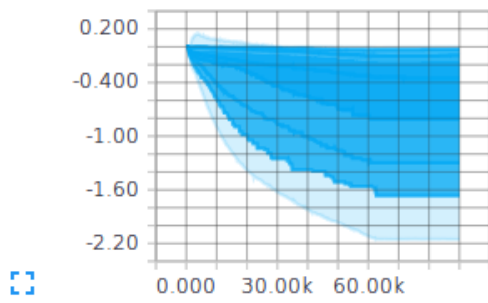
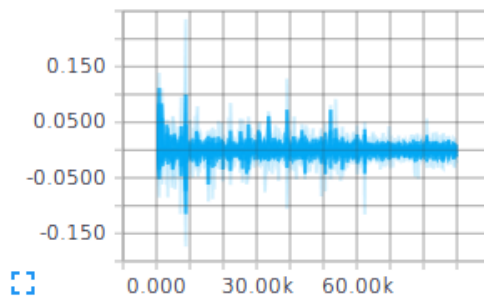


Figure G.6: Total Loss Plot

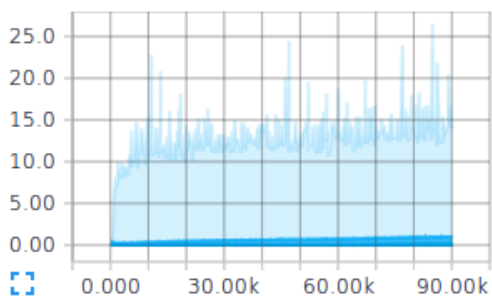
conv1/biases



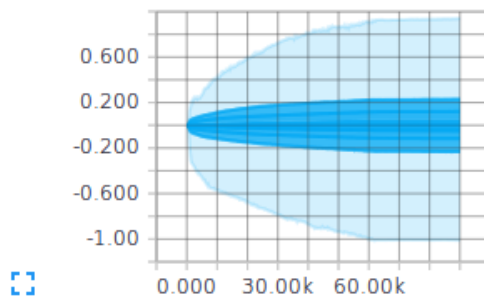
conv1/biases/gradients



conv1/conv1/activations



conv1/weights



conv1/weights/gradients

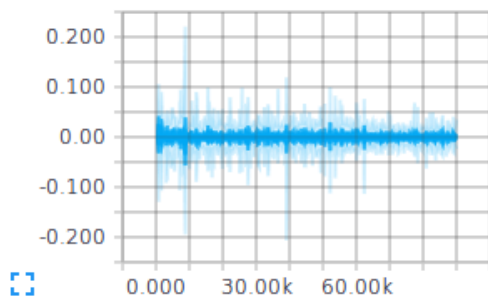


Figure G.7: Convolutional Layer 1 Histogram Plots for Biases, Activations and Weights

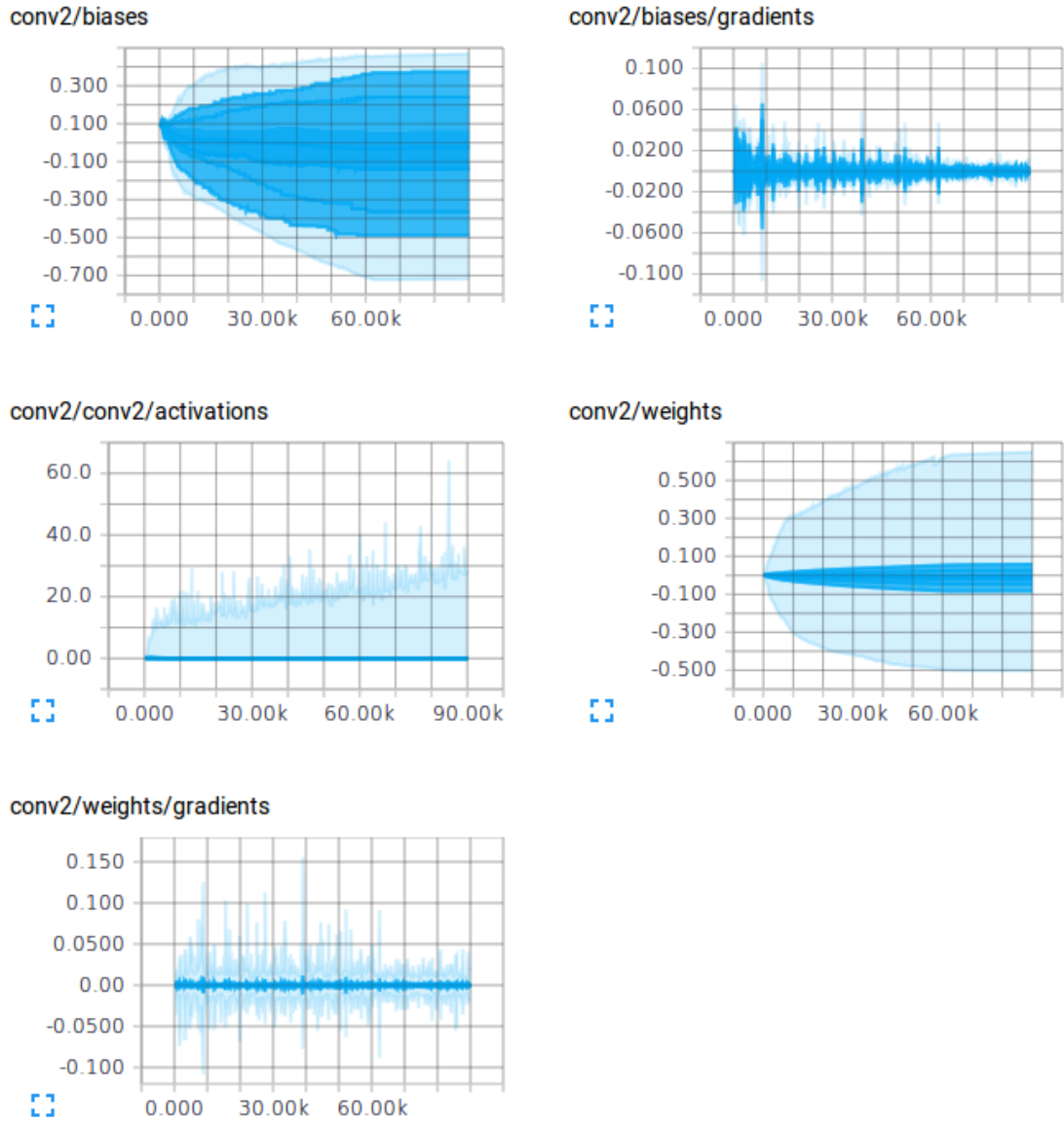
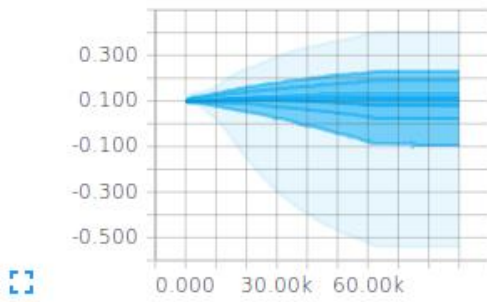
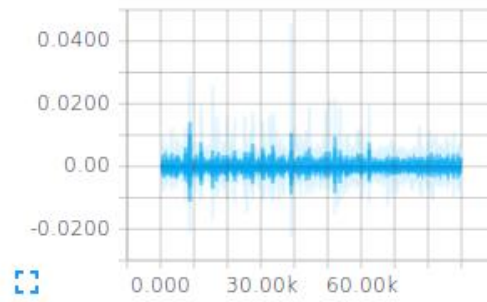


Figure G.8: Convolutional Layer 2 Histogram Plots for Biases, Activations and Weights

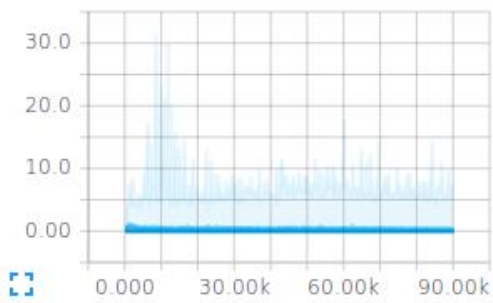
local3/biases



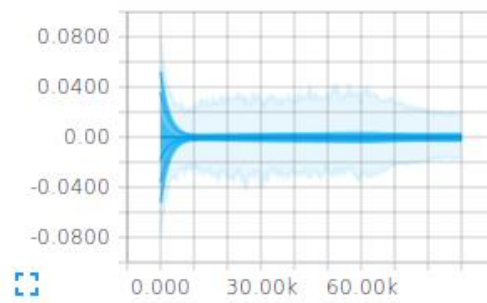
local3/biases/gradients



local3/local3/activations



local3/weights



local3/weights/gradients

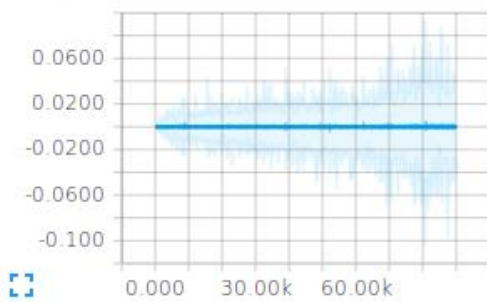


Figure G.9: Fully Connected Layer 1 Histogram Plots for Biases, Activations and Weights

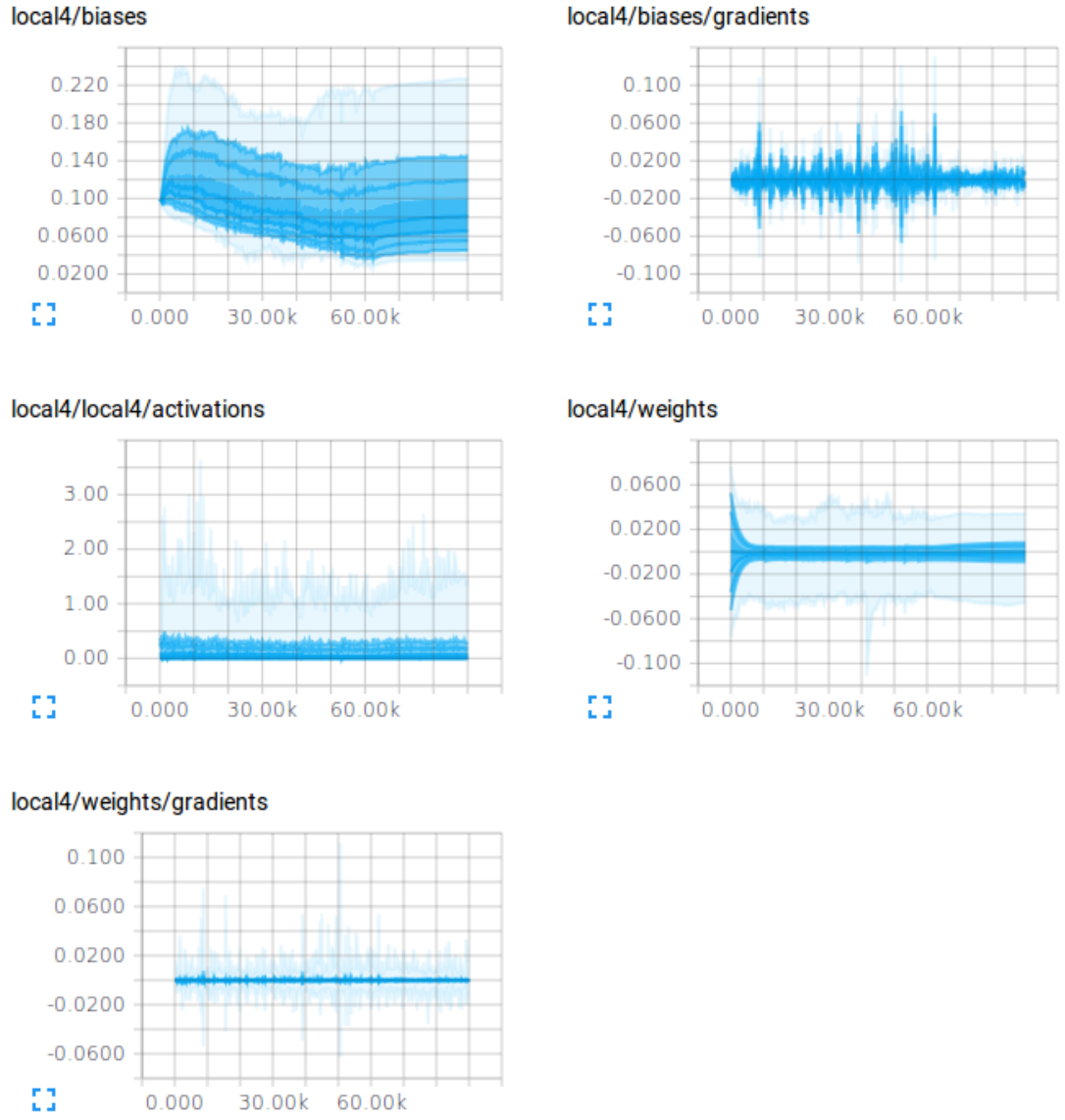
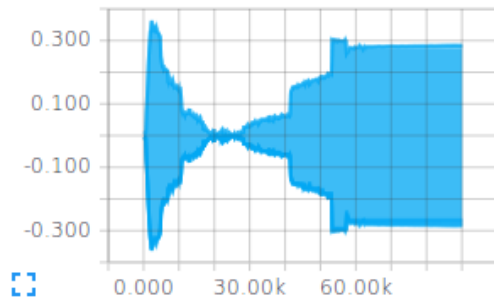
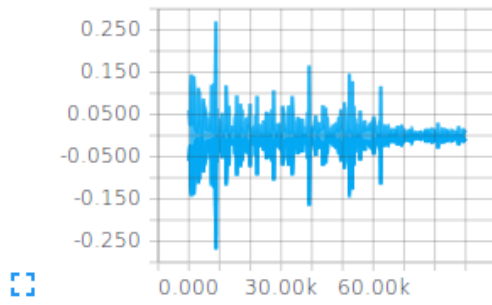


Figure G.10: Fully Connected Layer 2 Histogram Plots for Biases, Activations and Weights

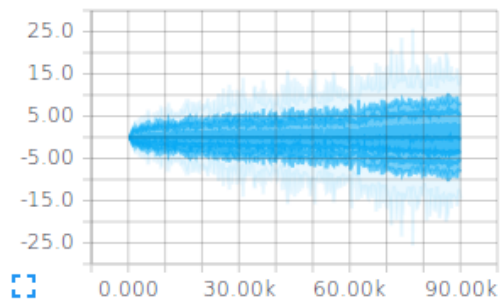
softmax_linear/biases



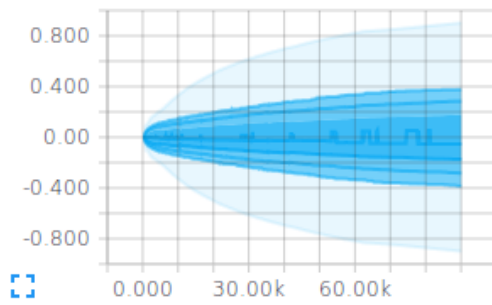
softmax_linear/biases/gradients



softmax_linear/softmax_linear/activations



softmax_linear/weights



softmax_linear/weights/gradients

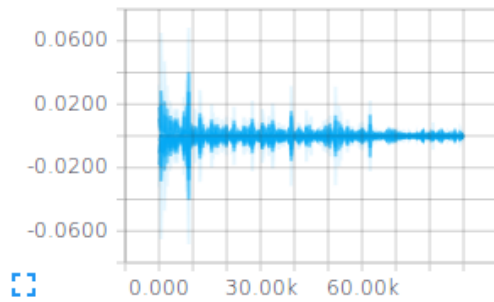


Figure G.11: Softmax Layer Histogram Plots for Biases, Activations and Weights