Deep Learning Assisted Methods for Microscopic Blood Cancer Imaging Analysis

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

INDRAPRASTHA INSTITUTE OF INFORMATION TECHNOLOGY DELHI

NEW DELHI- 110020

Deep Learning Assisted Methods for Microscopic Blood Cancer Imaging Analysis

A THESIS

submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

By

SHIV GEHLOT

PhD16119

ADVISOR

PROF. ANUBHA GUPTA, IIIT DELHI

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

INDRAPRASTHA INSTITUTE OF INFORMATION TECHNOLOGY DELHI

NEW DELHI- 110020

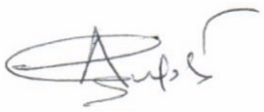
May 23, 2022

Certificate

This is to certify that the thesis titled **"Deep Learning Assisted Methods for Microscopic Blood Cancer Imaging Analysis"**, submitted by **Shiv Gehlot**, to the Indraprastha Institute of Information Technology Delhi, for the award of the degree of **Doctor of Philosophy**, is an original research work carried out by him under my supervision. In my opinion, the thesis has reached the standard fulfilling the requirements of the regulations relating to the degree.

The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

April 03, 2022



Prof. Anubha Gupta
Deptt. of Electronics and Communication Engineering
Indraprastha Institute of Information Technology Delhi-110020, India.

Declaration

This is to certify that the thesis titled **"Deep Learning Assisted Methods for Microscopic Blood Cancer Imaging Analysis"**, being submitted by me to the Indraprastha Institute of Information Technology Delhi, for the award of the degree of **Doctor of Philosophy**, is a bonafide work carried out by me. This research work has been carried out under the supervision of **Prof. Anubha Gupta**. The study pertaining to this thesis has not been submitted in part or in full, to any other University or Institution for the award of any other degree.

April 03, 2022

Shiv Gehlot
PhD Candidate
Deptt. of Electronics and Communication Engineering
Indraprastha Institute of Information Technology Delhi-110020, India.

ABSTRACT

This thesis, "Deep Learning Assisted Methods for Microscopic Blood Cancer Imaging Analysis" aims to develop deep learning-based CAD tools for Acute Lymphoblastic Leukemia (ALL) and Multiple Myeloma (MM). A CAD tool typically follows a hierarchical approach starting with stain normalization followed by cell segmentation and classification. Stain normalization counters the stain variation in the dataset while segmentation extracts the region of interest (ROI) from the histopathology images, which are finally analyzed for diagnosis (classification).

Accordingly, the *first objective* of this thesis is to develop a robust method for *stain normalization*. During the data collection process, a slide is prepared using the smear. A whole slide or microscopic image is then captured using a scanner or a camera mounted on the microscope. The slide preparation also involves staining to highlight the underlying cellular/tissue structure. For example, the application of hematoxylin and eosin (H&E) results in blue staining of the nuclei (due to H) and pink staining of cytoplasm (due to E). The staining leads to stain variation in the images due to multiple factors such as variation in scanners at different collections centers, the stain quantities, or the staining time. A significant prerequisite for deep learning algorithms is: *train and test conditions must match*. However, due to the stain variation, data distribution of the different centers may vary, and the model trained on one center may have a sub-optimal performance at the other. Stain normalization counters the stain variation and matches the chromatic distribution of the two images/centers.

The *second objective* is to develop a *segmentation* algorithm. Segmentation aims to identify the region of interest (ROI) by assigning a label to each pixel. The ROI could be nucleus, cells, or gland in a histopathology image. For example, in ALL and MM diagnoses, ROIs are the cell nuclei of the microscopic images. However, all the nucleus in an image are not corresponding to the cells of interest. Hence, an expert oncologist is required to identify the relevant ones. Semantic and instance segmentation are the two categories of segmentation. While the former assigns the same label to the ROI, the latter assigns a different label to each instance of the ROI. For instance,

semantic segmentation will identify a cluster of the nuclei, while instance segmentation will further segregate the cluster. We target the instance segmentation for ALL and MM diagnosis because the analysis is performed at the nucleus level.

The *final objective* is to develop deep learning-based classification methods for ALL and MM that *discriminate* the segmented structure (nuclei) into different categories, *cancerous or healthy*, based on the underlying pattern. The problem is challenging due to visual similarity between the malignant and normal cells.

To meet the *first objective*, this thesis proposes **Geometry-inspired Chemical-invariant and Tissue Invariant Stain Normalization (GCTI-SN)** method for microscopic medical images. The proposed GCTI-SN method corrects for illumination variation, stain chemical, and stain quantity variation in a unified framework by exploiting the underlying color vector space's geometry. GCTI-SN method is benchmarked against the existing methods via quantitative and qualitative results, validating its robustness for stain chemical and cell/tissue type. Further, the utility and the efficacy of the proposed GCTI-SN stain normalization method are demonstrated diagnostically in the application of breast cancer detection via a CNN-based classifier.

For the *second objective*, the thesis develops an **Encoder-Decoder based Convolutional Neural Network (CNN) with Nested-Feature Concatenation (EDNFC-Net)** for automatic nuclei segmentation. Accurate nuclei identification is an important and yet complex step in the diagnosis due to heterogeneity in structure, color, and texture among the different categories of cells. The problem is further complicated due to overlapped/clustered nuclei. The feature concatenation cell (FCC) of the EDNFC-Net comprises two stacks of convolutional filters combined with nonlinearity, followed by a concatenation of features. Apart from intra-FCC feature concatenation, a mechanism is also provided for inter-FCC feature concatenation. This arrangement leads to better feature flow and feature reusability. Similarly, direct feature flow is provided between the encoder and decoder module that preserves the context information. A new loss function with better-penalizing capability is also proposed that helps in the better background and foreground separation.

For *ALL diagnosis*, this thesis proposes an architecture, namely, **SDCT-AuxNet$^{\theta}$** that is a 2-module framework that utilizes a compact CNN as the primary classifier in one module and a Kernel SVM as the auxiliary classifier in the other one. While the

CNN classifier uses features through bilinear-pooling, spectral-averaged features are used by the auxiliary classifier. Further, this CNN is trained on the stain deconvolved quantity images in the optical density domain instead of the conventional RGB images. A novel test strategy is proposed that couples both the classifiers for decision-making using the confidence scores of their predicted class labels.

For *MM diagnosis*, this work proposes a unified framework that addresses the key challenges of inter-class visual similarity of healthy versus cancer cells and that of the label noise of the dataset. To extract class distinctive features, we propose projection loss to maximize the projection of a sample's activation on the respective class vector besides imposing orthogonality constraints on the class vectors. This projection loss is used along with the cross-entropy loss to design a dual branch architecture that helps achieve improved performance and provides scope for targeting the label noise problem. Based on this architecture, two methodologies have been proposed to correct the noisy labels. A coupling classifier has also been proposed to resolve the conflicts in the dual-branch architecture's predictions.

Next, this thesis presents an integrated framework for Stain Normalization, Classification, and Segmentation. The framework consists of a coupled network composed of two U-Net type architectures that utilize self-supervised learning. The first subnetwork (N1) learns an identity transformation, while the second (N2) learns a transformation to perform stain normalization. We also introduce classification heads in the subnetworks, trained along with the stain normalization task. To the best of our knowledge, the proposed coupling framework, where the information from the encoders of both the subnetworks is utilized by the decoders of both subnetworks as well as trained in a coupled fashion, is introduced in this domain for the first time. Interestingly, the coupling of N1 (for identity transformation) and N2 (for stain normalization) helps N2 learn the stain normalization task while being cognizant of the features essential to reconstruct images. Similarly, N1 learns to extract relevant features for reconstruction invariant to stain color variations due to its coupling with N2. Thus, the two subnetworks help each other, leading to improved performance on the subsequent classification task. Further, it is shown that the proposed architecture can also be used for segmentation, making it applicable for all three applications: stain normalization, classification, and segmentation.

While deep learning-based methods are data-intensive, availability of a large amount of data is a challenge in the medical domain due to difficulties in data capture and data annotation. Consequently, sufficient size imaging datasets are not available in the public domain for ALL and MM diagnosis. Therefore, in-house datasets of more than 100 subjects each were collected by AIIMS, New Delhi, India for ALL and MM cancers. Of these, the curated, annotated, and segmented (cell nucleus) ALL dataset was used in the challenge, *Classification of Normal vs. Malignant Cells in B-ALL White Blood Cancer Microscopic Image*, organized as part of the IEEE International Symposium on Biomedical Imaging (ISBI) 2019. This dataset is available in the public domain.

Keywords: **Deep Learning; AI in Cancer diagnosis; Affordable AI in healthcare; Microscopic images; Stain normalization; Nuclei segmentation; Acute lymphoblastic leukemia; ALL diagnosis; Cell classification; Multiple Myeloma; MM diagnosis; Projection Loss; Label Noise; Self-supervised learning**

Dedicated to the Origin and Sink of the Matter

ACKNOWLEDGEMENTS

I want to take this opportunity to express my sincere gratitude and appreciation to several incredible people whose tremendous guidance, mentoring, help, care, and love lead to my overall development and transformation.

First and foremost, I would like to thank my advisor, Prof. Anubha Gupta, for her consistent support, guidance, motivation, and encouragement over these years. She helped me a lot in nurturing my research capabilities and technical writing skills. I am very thankful to her for providing a very flexible and open research environment and constantly encouraging me to push the limits. I am thankful for the positive transformation that I achieved under her guidance over these years.

I would also like to thank my supervisory committee members: Dr. A V Subramanyam, and Dr. Arun Balaji Buduru, for their valuable advice, support, and constructive feedback through these years. I always benefited from their knowledge and wisdom. My gratitude also extends to the Admin Staff of IIIT-D, especially Priti Patel, for being extremely helpful in fast resolving all admin-related matters.

I am very thankful to my parents and brother, who always trusted me and supported me in my decisions, irrespective of the circumstances.

I want to extend a special thanks to my fiance, Bhawana Verma, for her tremendous support. My life went through challenging situations during my Ph.D., and her support made it possible to sail through such times. I dejected the tribulation and cheered the brilliance with her.

I am fortunate to have friends and labmates, Akanksha Farswan and Ashutosh Vaish. We went through the same cycle, situations, and emotions which bind us together. On many days, we just chattered for hours, and on other days, we did not utter a single word, yet we were in sync. Akanksha has been very supportive, and Ashutosh was always my accomplice in my caffeine addiction.

I want to thank Dr. Jianming Liang, Arizona State University, with whom I spent

six months as a Visting Scholar. I gained a lot from his knowledge and expertise in this limited time.

I am thankful to the Indraprastha Institute of Information Technology, Delhi (IIIT-D), for providing an excellent infrastructure and research environment.

Finally, I am very grateful to the University Grant Commission (UGC), Govt. of India, for the Junior (Senior) Research Fellowship that provided financial support during my Ph.D.

TABLE OF CONTENTS

ABSTRACT i

ACKNOWLEDGEMENTS vi

LIST OF TABLES xv

LIST OF FIGURES xx

ABBREVIATIONS xxi

1 Introduction 1

1.1 Motivation and Related Work 1

1.2 Research Objectives 4

1.3 Thesis Organization and Contributions 6

1.4 List of Publications 8

1.4.1 Journals 8

1.4.2 Conferences 9

2 GCTI-SN: Geometry-Inspired Chemical and Tissue Invariant Stain Normalization of Microscopic Medical Images 10

2.1 Introduction 10

2.2 Materials 14

2.2.1 BRCA-08 14

2.2.2 SN-AM (ALL) 14

2.2.3 SN-AM (MM) 14

2.3 Methods 15

2.3.1 Preliminaries 15

2.3.2 GCTI-SN Method 17

2.4 Results 22

2.4.1 Efficacy of Illumination Correction module 23

2.4.2 Efficacy of Stain Color Correction Module 26
2.4.3 Efficacy of Stain Quantity Correction Module 29
2.5 Application 33
2.5.1 Dataset 35
2.5.2 Augmented Dataset 36
2.5.3 Experiments 37
2.6 Conclusion 40

3 EDNFC-Net: Convolutional Neural Network with Nested Feature Concatenation for Nuclei-Instance Segmentation 42
3.1 Introduction 42
3.2 Material 44
3.3 Methods 44
3.3.1 Proposed Loss Function 46
3.3.2 Training and Testing 47
3.4 Results and Discussion 48
3.5 Conclusion 50

4 SDCT-AuxNet$^{\theta}$: DCT Augmented Stain Deconvolutional CNN with Auxiliary Classifier for Cancer Diagnosis 51
4.1 Introduction 51
4.2 Materials 54
4.3 Methods 55
4.3.1 Base Network 56
4.3.2 SD-Net Architecture 57
4.3.3 SDCT-Net Architecture 59
4.3.4 SDCT-AuxNet$^{\theta}$ 61
4.4 Results 65
4.4.1 System Configuration 66
4.4.2 Performance Metrics 66
4.4.3 Classifiers' Training and Testing Information 67
4.4.4 Experiments with SDCT-AuxNet$^{\theta}$ 68
4.4.5 Comparison with the state-of-the-art Methods 71
4.5 Discussion 73

4.6 Conclusion . . . 75

5 A CNN-based Unified Framework Utilizing Projection Loss in Unison with Label Noise Handling for Multiple Myeloma Cancer Diagnosis 76

5.1 Introduction . . . 76

5.2 Materials . . . 79

5.2.1 Multiple Myeloma (MM) Dataset . . . 79

5.2.2 Camelyon17 (Bandi *et al.*, 2019) . . . 81

5.2.3 TBX11K (Liu *et al.*, 2020) . . . 81

5.3 Methods . . . 81

5.3.1 Projection Loss . . . 82

5.3.2 BaseCE-Net . . . 84

5.3.3 BasePRL Net . . . 84

5.3.4 PRLCE-Net . . . 85

5.3.5 Label Noise Handling . . . 86

5.3.6 Coupling Classifier . . . 89

5.4 Results and Discussion . . . 92

5.4.1 Multiple Myeloma (MM) Dataset . . . 92

5.4.2 Camelyon7 and TBX11K datasets . . . 102

5.5 Conclusion . . . 107

6 Self-Supervision Based Dual-Transformation Learning for Stain Normalization, Classification and Segmentation 108

6.1 Introduction . . . 108

6.2 Methods . . . 109

6.3 Results . . . 112

6.4 Conclusion . . . 116

7 Conclusion and Future Work 118

7.1 GCTI-SN . . . 118

7.2 EDNFC-Net . . . 119

7.3 SDCT-AuxNet$^{\theta}$ . . . 119

7.4 MM Diagnosis . . . 119

7.5 AION . . . 120

7.6 Future Directions 120

A **122**

A.1 Additional Experiments for GCTI-SN 122

A.2 SDCT-Net with Skip Connections (SDCT-Net+S) 124

A.3 Flexibility of SDCT-AuxNet$^{\theta}$ 125

LIST OF TABLES

1.1 A brief summary of some of the methods utilizing CNNs for cell classification and cancer diagnosis. Training from scratch and transfer learning with finetuning are the preferred approaches. T.S.: training from scratch. 3

2.1 Dataset description. Three datasets with three different cell types and two different stains have been utilized in experiments. 15

2.2 Average MSD (with p-value) calculated on background and nucleus patches after illumination correction 24

2.3 Quantitative evaluation of Stain Color Correction module on BRCA-08 (H&E). 27

2.4 Quantitative evaluation of Stain Color Correction module on SN-AM (ALL) and SN-AM (MM). 29

2.5 Quantitative evaluation of Stain Quantity Correction module on BRCA-08 (H&E). 30

2.6 Quantitative evaluation of Stain Quantity Correction module on SN-AM (ALL) and SN-AM (MM). 31

2.7 Comparison of the MSD (p-value) obtained with the proposed and the existing methods 32

2.8 Performance comparison of the classifier trained using all possible six scenarios listed as experiments E_1 to E_6, with the original, normalized, color augmented, and normalized with color augmented images of C3. Testing results are compiled on the images of centers C0, C1, C2, and C4. The bold case numbers represent the highest metric value obtained for the given test dataset among all six experiments. 40

3.1 Dataset description. The training and testing set is obtained by extracting the patches of a given size from the images. 44

3.2 Comparison of U-Net, U-Net$_{\text{BL}}$ and EDNFC-Net on images of different modalities of DSBC18 DSB (2019), on BC19 Naylor *et al.* (2019) and on the mixture of DSBC18 and BC19 48

4.1 Description of training and test data. Training data is split in seven folds. All cells belonging to a subject are placed in the same fold and no two folds have cell images belonging to the same subject. 56

4.2 Comparison on the number of learnable parameters. The number of parameters in the *base network* is significantly less as compared to existing architectures used commonly in transfer learning approach. . 57

4.3 Weighted F_1 score of SDCT-AuxNet$^{\theta}$ for different values of confidence probability (θ). Model1 represents the model trained with fold1 as the validation set and remaining folds as training set. Similar notation applies to Model2 to Model7. The weighted F_1 (WF_1) score improves with increasing θ until $\theta = 0.9$, beyond which the performance starts declining. Results are presented on the test set of 2586 images. Best results are indicated in bold case. 69

4.4 Performance comparison of SDCT-AuxNet^{0}, SDCT-AuxNet^{1}, and SDCT-AuxNet$^{\theta}$ for $\theta = 0.9$. SDCT-Net is is a special case of SDCT-AuxNet$^{\theta}$ with $\theta = 0$ and is represented as SDCT-AuxNet^{0}. Similarly, SDCT-AuxNet^{1} is a spacial case of SDCT-AuxNet$^{\theta}$ with only the auxiliary classifier branch. Model1 represents the model trained with fold1 as the validation set and remaining folds as training set. Similar notation applies to Model2 to Model7. Best results are highlighted in black and equal performance is highlighted in blue color. SDCT-AuxNet$^{\theta}$ performed better than SDCT-AuxNet^{0} or SDCT-AuxNet^{1}. Results are presented on the test set of 2586 images. 71

4.5 Comparison of SDCT-Net and SDCT-AuxNet$^{\theta}$ with the top entry of C-NMC 2019 Challenge. Best results are depicted in bold case. . . . 73

4.6 The number of samples predicted by different classifiers at $\theta = 0.9$. Contribution of each classifier is approximately the same in majority of the cases, indicating significant role of both the classifiers in decision making. 74

5.1 Data Description. Number of subjects and images in the training and test sets. Distribution of subjects and images in different folds of training data is mentioned. The folds have been prepared such that almost 1:1 ratio of the data of healthy and cancer class is maintained, while a subjects' data is present in only one fold. 80

5.2 Dataset description for Camelyon17 (Bandi *et al.*, 2019) and TBX11K (Liu *et al.*, 2020). Test set GT is not available for TBX11K. 82

5.3 Results of Healthy Class F_1 Score and Cancer Class F_1 Score with all the proposed methods. These results are obtained on 40440 test images. Best results are highlighted in bold. Same results are depicted in italics. Network trained with fold-n (n=1,2..5) as the validation set is represented as Model-n. 94

5.4 Results in terms of Weighted F_1 Score and Balanced Accuracy with all the proposed methods. These results are obtained on 40440 test images. Best results are highlighted in bold. Network trained with fold-n (n=1,2..5) as the validation set is represented as Model-n. . . 95

5.5 The number of predictions aided by the coupling classifier. These results are obtained on 40440 test images. Network trained with fold-n (n=1,2,...,5) as the validation set is represented as Model-n. 96

5.6 Comparison of proposed method with some existing architectures in terms of weighted F_1 score, balanced accuracy, number of parameters, and test time. Results are computed on all 40441 test samples including time taken in decision making. Best results are highlighted in bold. $*$: parameters without coupling classifier. †: parameters without auxiliary classifier. 101

5.7 Results on clean Camelyon17 and TBX11K. The best results are highlighted in bold. 103

5.8 Results on Camelyon17 at different noise-levels with PRLCE-Net and incorporation of label flipping (LF) and sample discarding (SD). Best results are highlighted in bold. 104

5.9 Results on TBX11K at different noise-levels with PRLCE-Net and incorporation of sample discarding (SD). Best results are highlighted in bold. 104

5.10 Average number of the samples affected in sample discarding (SD) and label flipping (LF) at different noise levels (p) for TBX11K. 104

5.11 True detection rate (TDR) and false detection rate (FDR) for TBX11K at different noise level with sample discarding (SD) and label flippng (LF) 105

6.1 Description of the four dataset. Highlighted cells for Camelyon17 represent the sum of train and val dataset of the respective columns as the architecture trained on one center's is tested on another center's whole data. 113

6.2 Results with C3 as training data, and other centers as the test data. The shaded cells represent mean results with five various reference images. 115

6.3 AUC with inter-center training-testing 115

6.4 (I) PCam Classification results, (II) CVC and DSB Segmentation results 116

A.1 Effect of introducing very high blur on the performance of the classifier with and without normalization. This blurring is introduced as an additional augmentation. The best value for each row (center) is highlighted in black. 123

A.2 Effect of BANS on the performance of the classifier with and without normalization. Augmented dataset is prepared with lighter color augmentation compared to those used in manuscript R1. The best value for each row (center) is highlighted in black. 124

A.3 Comparison of SDCT-Net and SDCT-Net+S in term of parameters. Both architectures have same number of parameters. 124

A.4 Comparison of SDCT-Net+S and SDCT-Net. SDCT-Net+S is the architecture of SDCT-Net supported with skip connections. Model1 represents the model trained with fold1 as the validation set and remaining folds as the training set. Similar notation applies to Model2 to Model7. Results are tabulated on the test set of 2586 images. Both architectures are providing approximately similar performance. 125

LIST OF FIGURES

1.1 Computer-aided diagnosis pipeline for ALL and MM. A) Slides are prepared from the smear. Microscopic images are captured using the camera mounted on the microscope. B) Stain normalization is performed with **GCTI-SN** to counter the stain variation that may exist in the data. C) Cells of interest, marked by medical experts are segmented from the microscopic images using **EDNFC-Net**. D) Finally, segmented cells are classified as "cancerous" or "healthy": **SDCT-AuxNet**$^{\theta}$ is proposed for ALL diagnosis, and an another methodology is proposed for the MM diagnosis. 5

2.1 Proposed GCTI-SN Stain Normalization Method 17

2.2 Proposed frame alignment where origins are matched via illumination correction 19

2.3 (a) Reference image intensities over two reference basis vectors (b) Query image intensities over two query basis vectors (c) Query image intensities over basis vectors after alignment of query frame with that of reference frame. Note that the color shade in Figure-2.3c is similar to Figure-2.3a. This figure should be seen in color to visualize the significance of the 'proposed frame alignment" shown in Figure-2.2. 20

2.4 Robust stain color vector determination using wedge method (Macenko *et al.*, 2009) 20

2.5 Proposed rotation to every pixel's color according to color wedge basis 21

2.6 (a) Wedge of reference image (b) Wedge of query image (c) Query wedge aligned with reference wedge. This figure should be seen in color to visualize the importance of the "proposed rotation to pixels' color according to wedge basis" shown in Figure-2.5. 21

2.7 MSD after Illumination correction on the background of (a) SN-AM (ALL) (b) SN-AM (MM)
Computed between reference image and (i) original images, (ii) images from NMF based color deconvolution method (Li and Plataniotis, 2015), and (iii) images from the proposed method. 24

2.8 Results with multi-comparison Friedman test followed by Nemenyi posthoc test for Table-1. Numbers in bracket indicate rank averaged over all images of the dataset, where lowest rank indicates best performance with least MSD error. 25

2.9 Background pixel selection on (a) Query Image (b) by the existing method (Li and Plataniotis, 2015) and (c) by the proposed method in GCTI-SN pipeline. 26

2.10 Visual representation of stain color correction on SN-AM (ALL) (a) Reference image (b) Query image (c) Stain correction through SVD basis replacement showing color flipping (d) Stain correction through SVD basis transformation 29

2.11 Results with multi-comparison Friedman test followed by Nemenyi posthoc test for Table-3. Numbers in bracket indicate rank averaged over all images of the dataset, where lowest rank indicates best performance with least MSD error. 29

2.12 Results with multi-comparison Friedman test followed by Nemenyi posthoc test for Table-2.5. Numbers in bracket indicate rank averaged over all images of the dataset, where lowest rank indicates best performance with least MSD error. 31

2.13 Boxplots showing comparison of MSD on (a) SN-AM (ALL) dataset and (b) SN-AM (MM) dataset with original images and images obtained with: Histogram method (Jain, 1989), Color transform (Reinhard *et al.*, 2001), SVD based color deconvolution method (Macenko *et al.*, 2009), NMF based color deconvolution method (Li and Plataniotis, 2015), GCTI-SN (proposed) based on SVD color deconvolution, and GCTI-SN (proposed) based on NMFcolor deconvolution. 32

2.14 Results with multi-comparison Friedman test followed by Nemenyi posthoc test for Table- 2.7. Numbers in bracket indicate rank averaged over all images of the dataset, where lowest rank indicates best performance with least MSD error. 33

2.15 Quantitative comparison on (a) SN-AM (ALL) dataset (b) SN-AM (MM) dataset with original images and images obtained with: Histogram method (Jain, 1989), Color transform (Reinhard *et al.*, 2001), SVD based color deconvolution method (Macenko *et al.*, 2009), NMF based color deconvolution method (Li and Plataniotis, 2015), GCTI-SN (proposed) based on SVD color deconvolution, and GCTI-SN (proposed) based on NMF based color deconvolution. 34

2.16 Stain normalization on three sample images (top to down) of BRCA-08 dataset with (c) color transfer (d) SVD based color denovolution (e) NMF based color denovolution, and (f) proposed GCTI-SN method. 34

2.17 Stain normalization on three sample images (top to down) of SN-AM (ALL) dataset with (c) color transfer (d) SVD color deconvolution (e) NMF based deconvolution, and (f) proposed GCTI-SN method. . . . 35

2.18 Stain normalization on three sample images (top to down) of SN-AM (MM) dataset with (c) color transfer (d) SVD based color denovolution (e) NMF based color denovolution, and (f) proposed GCTI-SN method. 35

2.19 Sample original and normalized images from C0, C1, C2 and C4. The normalization is achieved using the reference image from C3. 36

2.20 Sample original images of C3 (row 1) and corresponding BANS based color augmented images (row 2). For visualization purpose, affine transformations are not introduced in these images. 37

3.1 Some samples from the dataset used. (a) DSBC18 (Flourescence) DSB (2019) (b) DSBC18 (Histology) DSB (2019) (c) DSBC18 (Brightfield) DSB (2019) (d) BC19 Naylor *et al.* (2019) 44

3.2 EDNFC-Net: Encoder-Decoder based Convolutional Neural Network (CNN) with Nested Feature Concatenation 45

3.3 Variation of balanced loss and Dice Loss with false negatives (left) and false positives (right). 47

3.4 Rows 1 to 3: Results on DSBC18 dataset (DSB, 2019) and Row 4: Results on BC19 dataset (Naylor *et al.*, 2019). Results are shown with U-Net, U-Net$_{BL}$, and EDNFC-Net. Results of Row 4 on BC19 are shown on the model trained on the mixture of the dataset DSBC18 and BC19. Pixels in the yellow boxes highlight the performance difference between the different models. These highlighted pixels contrast the difference in cluster segregation and segmentation mask generation with different architectures. 49

4.1 Sample images of different subjects from the cancer class (first row) and normal class (second row). This figure shows that the cell images are not distinguishable across classes or across subjects. 55

4.2 Base network. This network is used as the building block in SD-Net as well as in the proposed architectures. 57

4.3 SD-Net architecture. SD-layer deconvolves stain color vector in the OD space and provides stain quantity images, i.e., images that contain the quantification of the absorbed stain, a characteristic of the tissue. . . 58

4.4 Impact of DCT-layer on the output images of SD-layer. Normal cell images are sparsified more compared to cancer cell images with the DCT-layer. 60

4.5 (a) Effect of DCT-layer on the original RGB images. The output of the DCT-layer to cancer and healthy RGB input images is approximately similar. (b) Effect of DCT-layer on the the output of SD-layer images. DCT-layer when applied on the SD-layer shows differences between the output images of two classes. 61

4.6 Comparison of the effect of DCT-layer in terms of sparsity (a) on original RGB cell images and (b) on SD-layer output images. DCT-layer is not able to induce any noticeable sparsity difference in original images of two classes but a significant dissimilarity is observed when DCT is applied on the output of SD-layer. 62

4.7 Proposed SDCT-Net. DCT layer is appended after SD-layer which helps to learn class discriminative features by inducing different sparsity in cancer and normal cell images. 62

4.8 An overview of the proposed SDCT-AuxNet$^{\theta}$. The SDCT-AuxNet$^{\theta}$ can be interpreted as feature learning sub-network with two different classifiers, one classifier operating on the features obtained after bilinear pooling, and other classifier using the features obtained through spectral averaging. 63

4.9 Two-steps training of SDCT-AuxNet$^{\theta}$. In Step-1, SDCT-Net is trained in an end-to-end fashion on the entire training set containing K training images with indices $i = 1, 2, ..., K$. In Step-2, correctly classified training images by SDCT-Net are used to train an auxiliary classifier. 63

4.10 Testing strategy of the proposed SDCT-AuxNet$^{\theta}$. For each sample, probability of SDCT-Net decision label is compared against the confidence probability. If the probability of the predicted label by SDCT-Net is less than the confidence probability, it is predicted again using the auxiliary classifier. 65

4.11 Training curves of SDCT-Net with three different folds as the validation sets. 68

4.12 Performance of SDCT-AuxNet$^{\theta}$ for different values of confidence probability (θ): (a) F_1 score of normal class, (b) F_1 score of cancer class and (c) Balanced Accuracy. Model1 represents model trained with fold1 as the validation set and remaining folds as training set. Similar notation applies to Model2 to Model7. M.V is the score with majority voting on Model2 to Model7. Performance improves consistently with the increasing value of θ until $\theta = 0.9$, after which a decline in the performance is noted. Results are presented on the test set of 2586 images. 69

4.13 Subject-level performance of SDCT-AuxNet$^{\theta}$ is observed to be consistent across all the subjects indicating that SDCT-AuxNet$^{\theta}$ is robust to subject-level variations. 71

4.14 SDCT-AuxNet$^{\theta}$ as a generic case utilizing two different classifiers in decision making. SDCT-Net is same as SDCT-AuxNet$^{\theta}$ for $\theta = 0$ and utilizes only neural network. SDCT-AuxNet^{1} is SDCT-AuxNet$^{\theta}$ with $\theta = 1$ and use only the auxiliary classifier. SDCT-AuxNet$^{\theta}$ swings between SDCT-AuxNet^{0} (SDCT-Net) and SDCT-AuxNet^{1} depending upon the value of confidence probability θ. 74

4.15 Class discrimination at first (C1) and last convolutional layers (C6). The samples are separated as we progress deeper into the network. At the final layer, majority of the samples can be easily separated even with a linear boundary. 75

5.1 Sample images from the cancer class (first row), and the healthy class (second row). Samples of three subjects of each class have been shown. 80

5.2 Sample images from the Camelyon17 (first row): (a-c) Healthy samples, and (d-f) Tumor samples. TBX11K (second row): (g-h) Healthy samples, (i-j) Sick & Non-TB samples, and (k-l) TB samples. . . . 82

5.3 Projections of the activation on the class vectors. 83

5.4 PRLCE-Net. It uses CE and projection loss in the two branches. . . 85

5.5 The training process of the *PRLCE-Net* with the incorporation of the *sample discarding* as a label noise handling approach. The complete training occurs in two phases: first, the network is trained with all the training data and later, it is fine-tuned with *sample discarding* approach. 87

5.6 Training and Testing with Coupling Classifier. Features to the *coupling classifier* are fed through the spectral averaging layer. 90

5.7 Complete training and testing strategy incorporating label noise handling and coupling classifier. 91

5.8 Receiver Operating Characteristics and Area Under Curve (AUC) with different architectures. These results are obtained on 40440 test images. The network trained with fold-n (n=1,2,...,5) as the validation set is represented as Model-n. 97

5.9 Subject-level performance in terms of accuracy with *PRLCE-Net* and its variants: cancer class subjects (a-c), and healthy class subjects (d) 98

5.10 t-SNE plots with Model-5 on the training data for (a) PRL branch, and (b) CE branch. 101

5.11 t-SNE plots for TBX11K with SD (a-c) and LF (d-f) showing the detected noisy samples and missed noisy samples at noise level 0.3 (first column), 0.4 (second column), and 0.45 (third column). 106

6.1 AION: An architecture with two coupled-networks and classification heads for learning two transformations; one identity and another for stain normalization. AION can also be used for stain-invariant classification and segmentation. 110

6.2 Sample images from the datasets. (Set-1: left to right) Image from C0, C1, C2, C3, and C4 of Camelyon17. (Set-2) augmented images from C3 obtained with transformation $\phi(\cdot)$. (Set-3) PCam. (Set-4) DSB, (Set-5) CVC. 113

6.3 Sample segmentation masks from AION^{--}+H and AION (H2) from CVC. 116

A.1 Sample original images of C3 (row 1) and corresponding BANS based color augmented images (row 2). A large blur is also introduced as an additional augmentation and the resultant images are shown in row 3. Row 4 shows the BANS-based color augmented images. For visualization purpose, affine transformations are not introduced in these images. 123

A.2 SDCT-Net+S. Skip Connections are introduced in the SDCT-Net such that total trainable parameters remains unchanged. 125

ABBREVIATIONS

ALL	Acute Lymphoblastic Leukemia
AI	Artificial Intelligence
AION	All-In-One Network
AUC	Area Under Curve
BAC	Balanced Accuracy
BAN	Brightness-Affine-Noise
BANS	Brightness-Affine-Noise-Saturation
BCE	Binary Cross Entropy
CAD	Computer Aided Diagnosis
CC	Coupling Classifier
CNN	Convolutional Neural Network
DCT	Discrete Cosine Transform
DP	Digital Pathology
DSC	Dice Similarity Score
DL	Deep Learning
DU	Decoder Unit
EDNFC-Net	Encoder-Decoder based CNN with Nested-Feature Concatenation
EU	Encoder Unit
FCC	Feature Concatenation Cell
FCN	Fully Convolutional Network
GAP	Global Averaging Pooling
GCTI-SN	Geometry-Inspired Chemical and Tissue Invariant Stain Normalization of Microscopic Medical Images
H&E	Hematoxylin-Eosin
IoU	Intersection over Union
LMICs	Low- and Middle-Income Countries
LF	Label Flipping
MM	Multiple Myeloma

MSD	Mean Square Distance
MSE	Mean Square Error
NMF	Non-negative Matrix Factorization
OD	Optical Density
PRL	Projection Loss
RBF	Radial Basis Function
RGB	Red-Green-Blue
SD	Sample Discarding
SGD	Stochastic Gradient Descent
SD	Stain Deconvolutional
SDCT-AuxNet$^{\theta}$	DCT Augmented Stain Deconvolutional CNN with Auxiliary Classifier for Cancer Diagnosis
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TL	Transfer Learning
WBC	White Blood Cell
WF$_1$	Weighted F$_1$
WSI	Whole Slide Image

CHAPTER 1

Introduction

"The only thing I know is that I know nothing."

Socrates

1.1 Motivation and Related Work

Abnormalities such as cancer have witnessed a rapid escalation over the last few years. Cancer occurs due to unconstrained cell division and can cause organs' dyscrasias. In 2018, there were 18.1 million estimated new cancer cases and 9.6 million deaths. The cancer death numbers are projected to be approx. 13 million by 2030. The cancer mortality rate is reported to be higher in low- and middle-income countries (LMICs). These countries shared 65% of global cancer deaths in 2012 that is estimated to increase to 75% by 2030. However, the diagnosis infrastructure has not followed the same growth curve, and these statistics can be improved by expanding medical facilities, including expeditious diagnosis, crucial for moderating the mortality rate.

A possible solution exists in computer-aided diagnosis (CAD) or digital pathology (DP). If accurate enough, these tools can be deployed to aid medical professionals and mitigate the requirement of expensive specialized resources. Thus, CAD tools can act as significant enablers in scaling the necessary and affordable diagnostic facilities. While traditional diagnostic tests are the *de facto* standard in the medical domain, CAD has emerged expeditiously in recent years. CAD works at the intersection of computer vision and medical imaging to assess the underlying medical condition; for example, a histopathology image is processed using a computer vision algorithm for medical diagnosis.

Early medical image analysis was based on low-level image processing methods such as edge detection and region growing. Later, machine learning methods were also incorporated in CAD, which utilize the hand-crafted features based on the shape and structure of the ROIs. In the context of blood cancers, Mohapatra *et al.* (2011); Madhukar *et al.* (2012); Joshi *et al.* (2013); Mohapatra *et al.* (2014); Putzu *et al.* (2014);

Chatap and Shibu (2014); Reta *et al.* (2015); Neoh *et al.* (2015); Vincent *et al.* (2015); Kazemi *et al.* (2015); Patel and Mishra (2015); Amin *et al.* (2016*a*); Singhal and Singh (2016); Amin *et al.* (2016*b*); Rawat *et al.* (2017*a*,*b*); Karthikeyan and Poornima (2017); Mishra *et al.* (2017, 2019) have utilized traditional classifiers such as support vector machine, naïve Bayes, decision tree, random forest, etc. Besides the limitation of using hand-crafted features, these studies used very small datasets (19-267 images) for training the classifiers and for evaluating their test performance. The tools designed with such datasets may not be reliable for deployment in practical scenarios due to the large-scale heterogeneity within and across subjects' data in real-life. Also, these manually designed features may limit the performance due to their non-optimal nature for a specific task. An immediate solution is the automatic extraction of the task-dependent features from the dataset. Convolutional neural networks (CNNs) are deep learning architectures consisting of two-dimensional convolutional filters that can overcome the limitation of predefined features by extracting task-specific features from the raw data through the minimization of a specified loss function. Due to the rapid advancement of deep learning methods, computer-assisted medical image analysis has improved significantly. CNNs are frequently used for classification (Rajpurkar *et al.*, 2017), localization (Yan *et al.*, 2015), detection Sermanet *et al.* (2013), segmentation (Ronneberger *et al.*, 2015*a*), and registration (Yang *et al.*, 2017).

However, training of CNNs from scratch requires a large dataset depending upon the depth of the CNN. The availability of a large annotated dataset for supervised learning is a challenge in the medical domain. An alternate solution is to use transfer learning, wherein a network trained on one dataset (pre-trained network) is used on another dataset. In one approach of transfer learning, a pre-trained network is used directly for feature extraction, while in another approach, a pre-trained network is fine-tuned on an available dataset. For example, an overview of the works utilizing CNNs for cell classification and cancer diagnosis with all of the above three approaches is provided in Table 1.1. Training from scratch and transfer learning have been deployed frequently to classify different types of cancers. Although direct feature extraction eliminates the need for a training set, it is a less preferred approach, as seen from Table 1.1, because fine-tuning usually performs better than direct feature extraction.

For blood cancer cell classification, fine-tuning has been performed on AlexNet by Rehman *et al.* (2018) and Shafique and Tehsin (2018), while Vogado *et al.* (2017) and

Table 1.1: A brief summary of some of the methods utilizing CNNs for cell classification and cancer diagnosis. Training from scratch and transfer learning with finetuning are the preferred approaches. T.S.: training from scratch.

Reference	Task	Approach	Architecture
Han *et al.* (2016)	HEp-2 Cell Classification	CNN (T.S.)	CaffeNet
Bayramoglu *et al.* (2016)	Breast Cancer Classification in Histopathology Images	CNN (T.S.)	Custom CNN
Gao *et al.* (2017*a*)	HEp-2 Cell Classification	CNN (T.S.)	LeNet based CNN
Sirinukunwattana *et al.* (2016)	Colon Cancer Histology Images Classification	CNN (T.S.)	Custom CNN
Meng *et al.* (2019)	Cell Classification ATOM images	CNN (T.S.)	Custom CNN
Qin *et al.* (2018)	Leukocyte Classification	CNN (T.S.)	Custom CNN
Chang *et al.* (2017)	Cancer Cell Classification in Pancreas Histological Images	CNN (T.S.)	Custom CNN
Sharma *et al.* (2017)	Classification of Gastric Carcinoma Histopathological Images	CNN (T.S.)	Custom CNN
Gehlot *et al.* (2020*b*)	ALL Classification in Microscopic Images	CNN (T.S.)	Custom CNN
Xu *et al.* (2015)	Brain Tumor Classification in Histopathology Images	Transfer learning (features extraction)	AlexNet
Phan *et al.* (2016)	HEp-2 Cell Classification	Transfer learning (features extraction)	AlexNet based CNN
Bayramoglu and Heikkilä (2016)	Nuclei Classification in Histopathological Images	Transfer learning (fine tuning)	AlexNet, GoogleNet, VGG-16, GenderNet
Tabibu *et al.* (2019)	Renal Cell Carcinoma (RCC) Histopathological Image Classification	Transfer learning (fine tuning)	Resnet-18 & Resnet-34
Han *et al.* (2018*b*)	Classification of Cutaneous Tumors	Transfer learning (fine tuning)	ResNet-152
Harangi (2018)	Skin lesion classification	Transfer learning (fine tuning)	GoogLeNet, AlexNet, ResNet-152, VGGNet
Zhang *et al.* (2017)	Cervical Cell Classification	Transfer learning (fine tuning)	CaffeNet
Jiang *et al.* (2017)	Breast Cancer Classification in Mammograms	Transfer learning (fine tuning)	GoogLeNet, AlexNet
Esteva *et al.* (2017)	Skin Cancer Classification	Transfer learning (fine tuning)	Inception-v3
Hekler *et al.* (2019)	Histopathological Melanoma Image Classification	Transfer learning (fine tuning)	ResNet50
Mazo *et al.* (2018)	Cardiovascular Tissue Classification in Histological Images	Transfer learning (fine tuning)	VGG16, VGG19, ResNet, Inception

Vogado *et al.* (2018) extracted features directly from CNNs and used SVM or other classifiers later. However, very small datasets have been used in these studies, say of 108, 310, 330, and 891 images by Vogado *et al.* (2017), Rehman *et al.* (2018), Shafique and Tehsin (2018), and Vogado *et al.* (2018), respectively. Fine-tuning approach on a large dataset of B-ALL cancer, consisting of 12528 cell images for training and 2586 cell images for testing (Gupta and Gupta, 2019*a*), has been carried out on different architectures (Gupta and Gupta, 2019*c*) by Pan *et al.* (2019); Verma and Singh (2019); Prellberg and Kramer (2019); Xiao *et al.* (2019); Shi *et al.* (2019); Liu and Long (2019);

Shah *et al.* (2019); Ding *et al.* (2019); Xie *et al.* (2019). For example, pretrained ResNet incorporating label correction is employed by Pan *et al.* (2019). Similarly, fine-tuning of ResNetXt50 with a layer-dependent learning rate is used by Prellberg and Kramer (2019). Xiao *et al.* (2019) utilized a pseudo labeling approach with ensembling of pretrained architectures. The ensembling is also used by Shah *et al.* (2019) and Ding *et al.* (2019). Verma and Singh (2019) used MobileNet; Liu and Long (2019) and Xie *et al.* (2019) used Inception ResNet, while Goswami *et al.* (2020) fine-tuned a pretrained Inception-v3 on the above mentioned dataset using a newly defined heterogeneity loss function. Besides using class centers, this loss function assigns a separate center to each subject and attempts to capture the inter-class and inter-subject distinguishable characteristics.

1.2 Research Objectives

In this thesis, we develop deep learning based CAD tools utilizing microscopic images for the blood cancer diagnosis, specifically Acute Lymphoblastic Leukemia (ALL) and Multiple Myeloma (MM), as outlined in Fig 1.1. A computer-assisted cancer diagnosis tool typically follows a hierarchical approach starting with stain normalization followed by segmentation and classification. Stain normalization counters the stain variation in the dataset, and segmentation extracts the region of interest from the input images. For our tasks, the region of interest (ROI) is nuclei in the microscopic images. Finally, ROI is analyzed to make the diagnosis (classification); malignant or benign.

During the data collection process, a slide is prepared using the smear. A whole slide or microscopic image is then captured using a scanner or a camera mounted on the microscope. The slide preparation also involves staining to highlight the underlying cellular/tissue structure. For example, the application of hematoxylin and eosin (H&E) results in blue staining of the nuclei (due to H) and pink staining of cytoplasm (due to E). The staining leads to stain variation in the images due to multiple factors such as variation in scanners at different collections centers, the stain quantities, or the staining time. A significant prerequisite for deep learning algorithms is: *train and test conditions must match*. However, due to the stain variation, data distribution of the different centers may vary, and the model trained on one center may have an incompetent performance at

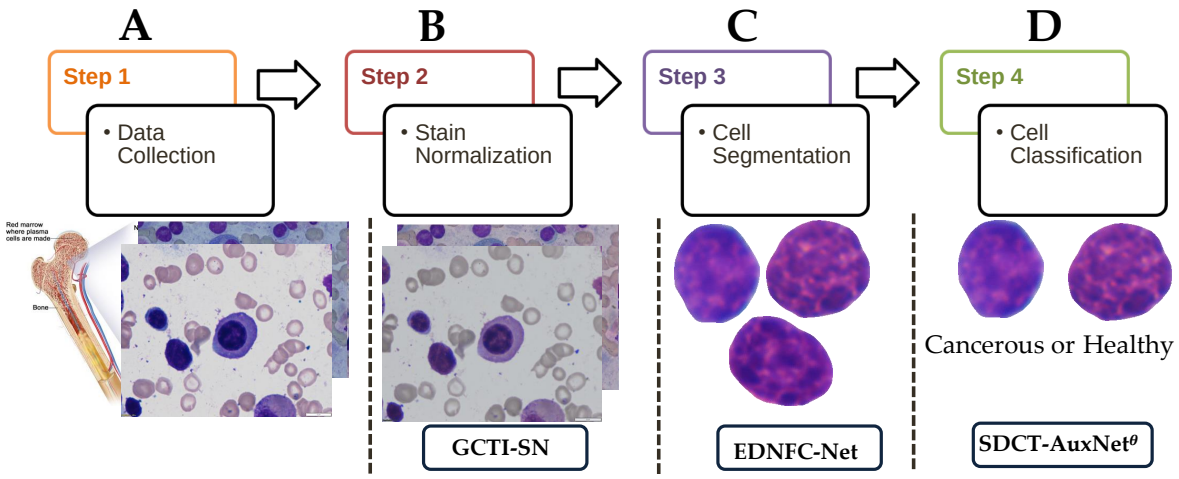


Figure 1.1: Computer-aided diagnosis pipeline for ALL and MM. A) Slides are prepared from the smear. Microscopic images are captured using the camera mounted on the microscope. B) Stain normalization is performed with **GCTI-SN** to counter the stain variation that may exist in the data. C) Cells of interest, marked by medical experts are segmented from the microscopic images using **EDNFC-Net**. D) Finally, segmented cells are classified as "cancerous" or "healthy": **SDCT-AuxNet**$^{\theta}$ is proposed for ALL diagnosis, and an another methodology is proposed for the MM diagnosis.

the other. **Stain normalization** counters the stain variation and matches the chromatic distribution of the two images/centers. Common approaches for stain normalization are based on optimization or generative modeling. The former utilizes a reference image to match the color distribution while the latter involves GANs, requiring an extensive training set.

Segmentation aims to identify the region of interest (ROI) by assigning a label to each pixel. The ROI could be nucleus, cells, or gland in a histopathology image. For example, in ALL and MM diagnoses, ROI is nuclei in the microscopic images. However, not all the nuclei in an image are of interest, and an expert oncologist identifies the relevant one. Semantic and instance segmentation are the two categories of segmentation. While the former assigns the same label to the ROI, the latter assigns a different label to each instance of the ROI. For instance, semantic segmentation will identify a cluster of the nuclei, while instance segmentation will further segregate the cluster. We target the instance segmentation for ALL and MM diagnosis as the analysis is performed at the nuclei level.

Classification discriminates the segmented structure into different categories based on the underlying pattern, and a classifier is trained to learn class discriminative features. For ALL and MM diagnoses, the aim is to classify segmented nuclei as malignant

or benign and is challenging due to higher inter-class visual similarity.

While deep learning-based methods are data-intensive, availability of a large amount of data is a challenge in the medical domain due to difficulties in data capture and data annotation. Consequently, sufficient size imaging datasets are not available in the public domain for ALL and MM diagnosis. Therefore, in-house datasets of more than 100 subjects each were collected by AIIMS, New Delhi, India for ALL and MM cancers. Of these, the curated, annotated, and segmented (cell nucleus) ALL dataset was used in the challenge, *Classification of Normal vs. Malignant Cells in B-ALL White Blood Cancer Microscopic Image*, organized as part of the IEEE International Symposium on Biomedical Imaging (ISBI) 2019. This dataset is available in the public domain.

1.3 Thesis Organization and Contributions

The contributions and organization of this dissertation are as follows:

- *Chapter 2* targets the stain normalization and proposes **Geometry-inspired Chemical - invariant and Tissue Invariant Stain Normalization (GCTI-SN)** method for microscopic medical images. The proposed GCTI-SN method corrects illumination variation, stain chemical, and stain quantity variation in a unified framework by exploiting the underlying color vector space's geometry. GCTI-SN method is benchmarked against the existing methods via quantitative and qualitative results, validating its robustness for stain chemical and cell/tissue type. Further, the utility and the efficacy of the proposed GCTI-SN stain normalization method are demonstrated diagnostically in the application of breast cancer detection via a CNN-based classifier.

- *Chapter 3* is focussed on segmentation and develops **Encoder-Decoder based Convolutional Neural Network (CNN) with Nested-Feature Concatenation (EDNFC-Net)** for nuclei segmentation. Accurate nuclei identification is an important and yet complex step in the diagnosis due to heterogeneity in structure, color, and texture among the different categories of cells. The problem is further complicated due to overlapped/clustered nuclei. The feature concatenation cell (FCC) of the EDNFC-Net comprises two stacks of convolutional filters combined with nonlinearity, followed by a concatenation of features. Apart from

intra-FCC feature concatenation, a mechanism is also provided for inter-FCC feature concatenation. This arrangement leads to better feature flow and feature reusability. Similarly, direct feature flow is provided between the encoder and decoder module that preserves the context information. A new loss function with better-penalizing capability is also proposed that helps in the better background and foreground separation.

- *Chapter 4* deals with ALL diagnosis and presents **SDCT-AuxNet$^{\theta}$: DCT Augmented Stain Deconvolutional CNN with Auxiliary Classifier for Cancer Diagnosis** that is a 2-module framework that utilizes a compact CNN as the primary classifier in one module and a Kernel SVM as the auxiliary classifier in the other one. While the CNN classifier uses features through bilinear-pooling, spectral-averaged features are used by the auxiliary classifier. Further, this CNN is trained on the stain deconvolved quantity images in the optical density domain instead of the conventional RGB images. A novel test strategy is proposed that couples both the classifiers for decision-making using the confidence scores of their predicted class labels.

- *Chapter 5* proposes **A CNN-based Unified Framework Utilizing Projection Loss in Unison with Label Noise Handling for Multiple Myeloma Cancer Diagnosis**. It addresses the key challenges of inter-class visual similarity of healthy versus cancer cells and that of the label noise of the dataset. To extract class distinctive features, we propose projection loss to maximize the projection of a sample's activation on the respective class vector besides imposing orthogonality constraints on the class vectors. This projection loss is used along with the cross-entropy loss to design a dual branch architecture that helps achieve improved performance and provides scope for targeting the label noise problem. Based on this architecture, two methodologies have been proposed to correct the noisy labels. A coupling classifier has also been proposed to resolve the conflicts in the dual-branch architecture's predictions.

- *Chapter 6* presents **AION: an all-in-one integrated framework for Stain Normalization, Classification, and Segmentation**. The framework consists of a coupled network composed of two U-Net type architectures that utilize self-supervised learning. The first subnetwork (N1) learns an identity transformation,

while the second (N2) learns a transformation to perform stain normalization. We also introduce classification heads in the subnetworks, trained along with the stain normalization task. To the best of our knowledge, the proposed coupling framework, where the information from the encoders of both the subnetworks is utilized by the decoders of both subnetworks as well as trained in a coupled fashion, is introduced in this domain for the first time. Interestingly, the coupling of N1 (for identity transformation) and N2 (for stain normalization) helps N2 learn the stain normalization task while being cognizant of the features essential to reconstruct images. Similarly, N1 learns to extract relevant features for reconstruction invariant to stain color variations due to its coupling with N2. Thus, the two subnetworks help each other, leading to improved performance on the subsequent classification task. Further, it is shown that the proposed architecture can also be used for segmentation, making it applicable for all three applications: stain normalization, classification, and segmentation.

1.4 List of Publications

The following publications contributed to this dissertation:

1.4.1 Journals

1. **Shiv Gehlot**, Anubha Gupta, and Ritu Gupta. A CNN-based Unified Framework Utilizing Projection Loss in Unison with Label Noise Handling for Multiple Myeloma Cancer Diagnosis. *Medical Image Analysis*, Volume 72, 102099, 2021.

2. **Shiv Gehlot**, Anubha Gupta, and Ritu Gupta. SDCT-AuxNet$^{\theta}$: DCT Augmented Stain Deconvolutional CNN with Auxiliary Classifier for Cancer Diagnosis. *Medical Image Analysis*, Volume 61, 101661, 2020.

3. Anubha Gupta, Rahul Duggal, **Shiv Gehlot**, Ritu Gupta, Anvit Mangal, Lalit Kumar, Nisarg Thakkar, and Devprakash Satpathy. GCTI-SN: Geometry-Inspired Chemical and Tissue Invariant Stain Normalization of Microscopic Medical Images. *Medical Image Analysis*, Volume 65, 101788, 2020

1.4.2 Conferences

1. **Shiv Gehlot**, Anubha Gupta, and Ritu Gupta. EDNFC-NET: Convolutional Neural Network withNested Feature Concatenation for Nuclei-Instance Segmentation. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, 2020, pp. 1389-1393.

2. **Shiv Gehlot**, and Anubha Gupta. Self-supervision Based Dual-Transformation Learning for Stain Normalization, Classification and Segmentation. In: Lian C., Cao X., Rekik I., Xu X., Yan P. (eds) *Machine Learning in Medical Imaging. MLMI 2021*. Lecture Notes in Computer Science, vol 12966. Springer, Cham.

CHAPTER 2

GCTI-SN: Geometry-Inspired Chemical and Tissue Invariant Stain Normalization of Microscopic Medical Images

2.1 Introduction

Anatomic pathology is the science of detection and identification of diseases from tissue samples. Analysis of a given tissue sample requires staining the tissue with a mixture of dyes (or stains) that highlights the cell or tissue of interest under a microscope. This is due to different affinities of stains for different biological entities in a tissue. Recently, with advances in computer science and increased collaborative efforts of the medical and engineering community, there is an impetus on building automated disease diagnostic tools. Such tools require microscopic images of stained slides as input and involve cell or tissue segmentation followed by classification. Before imaging, microscopic slides are prepared using stain chemicals that bind to the tissues of interest and absorb colors of the visible light spectrum (from violet to red that is sensed by the RGB sensors of the camera) according to its texture. The stained slides are prone to some irregularities with reference to background illumination, camera change, etc. Since the subsequent stages of cell segmentation and classification utilize color information, such tools' performance is susceptible to color variations. This presents a need for color (stain) normalization of stained microscopic images as the first pre-processing step for any computer-assisted automated diagnostic tool.

Stain normalization for microscopic medical images has been widely explored via histogram equalization (Jain, 1989; Tabesh *et al.*, 2007; Kothari *et al.*, 2011; Basavanhally and Madabhushi, 2013), color transfer (Reinhard *et al.*, 2001; Ruderman *et al.*, 1998; Magee *et al.*, 2009), and color deconvolution (Ruifrok and Johnston, 2001; Abe *et al.*, 2005; Macenko *et al.*, 2009; McCann *et al.*, 2014; Khan *et al.*, 2014; Li and Plataniotis, 2015; Newberg and Murphy, 2008; Rabinovich *et al.*, 2003; Bilgin *et al.*, 2012)

methods. In histogram equalization and/or matching methods, normalized histogram is computed and the probability distribution function (PDF) is estimated. Next, the PDF of query image is matched to that of reference image in each of the R,G,B channels (Jain, 1989; Tabesh *et al.*, 2007; Kothari *et al.*, 2011; Basavanhally and Madabhushi, 2013). Color transfer methods also use the PDF or histograms of query and reference images. For example, Reinhard *et al.* (2001) matched mean and standard deviations of reference and query images in the Lab color space. Magee *et al.* (2009) applied Gaussian mixture probabilistic model based automatic segmentation at pixel level for each of the three classes Hematoxylin (H), Eosin (E), and background followed by color transfer on the corresponding classes from the reference to the query image. However, each pixel could not be uniquely classified to H or E classes because stain chemicals react and tissues absorb different components based on their biological content. Thus, this method may alter the histological information of the images.

Both histogram equalization and color transfer methods do not take into account the histological information and local color differences leading to smearing of histological components for overlapping PDFs of regions of interest (ROIs) in color spaces and alteration of nucleus and cytoplasm boundaries and/or their texture. To handle the color smearing of different histological components, manual segmentation (Abe *et al.*, 2005) and automated segmentation (Magee *et al.*, 2009) have also been applied prior to stain normalization via PDF matching. While manual segmentation is not feasible for large datasets, automated segmentation will depend on color information and hence, would be possible only if stain normalization is done apriori.

Color deconvolution is another popular approach for stain normalization (Ruifrok and Johnston, 2001; Macenko *et al.*, 2009; McCann *et al.*, 2014; Khan *et al.*, 2014; Li and Plataniotis, 2015; Newberg and Murphy, 2008; Rabinovich *et al.*, 2003; Bilgin *et al.*, 2012). These methods use mathematical framework for stain color correction by computing stain vectors using singular value decomposition (SVD) or non-negative matrix factorization (NMF) and perform basis replacement and stain quantity correction. Both SVD and NMF involve factorization of optical density (OD) space matrix.

SVD, along with wedge finding, has been used to estimate stain vectors corresponding to H and E components for H&E stained images (Macenko *et al.*, 2009; McCann *et al.*, 2014). The correction for variations in stain quantity in these studies was done by

scaling stain constituent quantities of the query to match the query's 99th percentile to the 99th percentile values of the reference image. However, these stain normalization methods do not implement illumination correction. Also, stain color vector correction is achieved by replacing the query stain vectors with reference stain vectors.

NMF based methods factor OD matrix into stain vector and stain quantity matrices with the constraint that all matrix elements are non-negative because stains cannot absorb a negative amount of light(Li and Plataniotis, 2015; Newberg and Murphy, 2008; Rabinovich *et al.*, 2003; Bilgin *et al.*, 2012). The final solution using NMF based methods depends on initialization and does not yield a closed-form solution. NMF methods work well in multi-spectral images where a lot of information is present but may provide inconsistent convergence when only three spectral components are available (Macenko *et al.*, 2009).

Recently, generative model-based approaches have also been applied for stain normalization. Zanjani *et al.* (2018) have used an unsupervised InfoGAN (Chen *et al.*, 2016) based approach for stain normalization. InfoGAN is supplemented with an auxiliary network, and the overall architecture performs color transformation while preserving the content information. Ghazvinian Zanjani *et al.* (2018) have proposed two stain normalization methods: 1) variational-autoencoder (VAE) based method and 2) deep convolutional Gaussian mixture model (DCGMM). The VAE based method exploits a segmentation-based approach, wherein the encoder transforms the input sample to a latent space that represents the classes according to stain absorption, i.e., hematoxylin, eosin, and no stain. Afterward, the decoder is used for the chromatic channel's reconstruction of the input images from the classes' samples. A similar approach is used in DCGMM, wherein CNN is used for tissue class detection, and a GMM conditioned on these classes learns the color distribution of the pixels. (Shaban *et al.*, 2018) have exploited CycleGAN (Zhu *et al.*, 2017) based approach that achieves color transformation, while keeping the structure information intact. All these methods have a significant limitation in terms of the requirement of an extensive training dataset.

SVD and NMF based color deconvolution methods are noteworthy because they present the mathematical formulation to determine stain vectors from the images instead of deciding the stain constituent vectors experimentally. However, the existing SVD and NMF based methods replace the stain basis of the query image with that of the reference

image. This does not suffice because 1) the origin of the reference image's stain basis frame may be different from that of the query basis frame; 2) the alignment of stain color vector basis of reference and query images may be changed, and 3) the size of the query basis wedge may be different from the reference basis wedge. Thus, although the methods advanced the theory, a few limitations remain to be addressed.

In the present study, we extend the SVD and NMF based methods to develop a solution for robust stain normalization. Overall, the following are the salient features of the proposed work:

1. The proposed GCTI-SN method utilizes the geometry of the underlying color vector space, in a unified framework, to counter all the three putative factors responsible for color variations in microscopic images, namely, (i) illumination variation, (ii) stain color vector variation, and (iii) stain quantity variation.

2. The GCTI-SN method is stable and robust. It is benchmarked on three cancer datasets of three cell/tissue types prepared with two different stain chemicals and is observed to be stain chemical invariant and tissue invariant. This robustness has been achieved via the introduction of a novel but simple illumination correction method besides the two new steps, step 2B and step 2D, in the stain color vector correction module.

3. Quantitative, qualitative, and diagnostic validation of the tool has been carried out. Extensive quantitative comparison with statistical validation of the proposed GCTI-SN pipeline has been carried out with the existing methods of all three classes: histogram methods (Jain, 1989), color transfer methods (Reinhard *et al.*, 2001), and color deconvolution methods (Macenko *et al.*, 2009; Li and Plataniotis, 2015).

4. Diagnostically, the utility and the efficacy of the proposed GCTI-SN stain normalization method is demonstrated in the application of breast cancer detection via a CNN-based classifier.

5. The GCTI-SN method requires only one reference image to implement the solution, which is in contrast to all deep learning (DL) based methods that require a large training dataset to arrive at an optimal solution. Even if transfer learning is used, DL solutions cannot yet be implemented with just one training (here

reference) image. In the case of a small dataset requiring stain color normalization, the proposed method can be applied immediately, while there would be challenges with the implementation of DL based methods. Thus, the proposed work has its merit over DL based solutions.

2.2 Materials

Three publicly available datasets of different cell types prepared with two different staining chemicals are used to validate the proposed GCTI-SN method.

2.2.1 BRCA-08

The first dataset contains publicly available Hematoxylin-Eosin (H&E) stained microscopic images of breast cancer (Gelasca *et al.*, 2008). The dataset consists of 58 images, wherein one image has been used as the reference for the stain normalization of the remaining 57 images.

2.2.2 SN-AM (ALL)

Microscopic images were captured from bone marrow aspirate slides of patients diagnosed with B-lineage Acute Lymphoblastic Leukemia (ALL). Images were captured in raw BMP format with a size of 2560x1920 pixels using the Nikon Eclipse-200 microscope equipped with a digital camera at 1000x magnification. The 30 images of this dataset are considered for experiments. One image has been used as the reference image, and the proposed stain normalization method has been tested on the remaining 29 images.

2.2.3 SN-AM (MM)

The third dataset contains microscopic images captured from slides prepared from bone marrow aspirate collected from patients with Multiple Myeloma (MM). A total of 30 images have been considered, wherein one image has been used as the reference image

Table 2.1: Dataset description. Three datasets with three different cell types and two different stains have been utilized in experiments.

Dataset	Type	Stain	No. of images
BRCA-08 (Gelasca *et al.*, 2008)	Breast cancer	H&E	58
SN-AM (ALL) (Gupta and Gupta, 2019*d*)	B-cell blood cancer	Jenner-Giemsa	30
SN-AM(MM) (Gupta and Gupta, 2019*d*)	Plasma cell blood cancer	Jenner-Giemsa	30

using which the remaining 29 images have been stain normalized.

The SN-AM (ALL) and SN-AM (MM) are prepared with Jenner-Giemsa stain, also known as Romanowsky-Giemsa staining, as Jenner is a Romanowsky stain. Both these dataset are publicly available at the Cancer Imaging Archive (TCIA) (Gupta and Gupta, 2019*d*) maintained according to TCIA protocols (Clark *et al.*, 2013*a*). For both the dataset, a waiver for written informed consent was obtained from the Ethics Committee of the Institute. The patient identifying information was removed from the image dataset before sharing data for building up the color normalization pipeline GCTI-SN proposed in this work. A brief description of these datasets is also provided in Table-2.1.

2.3 Methods

In this section, the proposed method is presented in detail. We will first discuss the required preliminaries.

2.3.1 Preliminaries

While imaging, a specimen slide is exposed to incident light. Assuming a 3-sensor RGB (red (R), green (G), and blue (B) color sensors) camera and assuming three constituents of the given staining chemical, the intensity at a pixel p in the i^{th} color sensor channel is given by Koschan and Abidi (2008) and Li and Plataniotis (2015) as:

$$\mathbf{I}(p, \lambda_i) = \int_{\lambda_i - \delta}^{\lambda_i + \delta} f_i(\lambda) E(\lambda) e^{-\sum_{s=1}^{3} m_i(s,\lambda) d(p,s)} d\lambda, \tag{2.1}$$

where $i = 1, 2, 3$ correspond to R, G, and B channels, $s = 1, 2, 3$ correspond to three constituents of the given staining chemical, $f_i(\lambda) > 0$ for $\lambda \in (\lambda_i - \delta, \lambda_i + \delta)$ rep-

resents i^{th} sensor's response of camera (say for the red color) within $\pm\delta$ of it's color wavelength, $E(\lambda)$ denotes spectral power distribution (SPD) of imaging light, $m_i(s,\lambda)$ denotes the characteristic absorbance of the s^{th} stain constituent in the i^{th} sensor channel, and $d(p,s)$ denotes the quantity of the s^{th} stain constituent bound at pixel position p. The stain chemical binds to the tissue of interest and absorbs colors of visible light spectrum according to its texture. It has been shown by Li and Plataniotis (2015) that if δ is very small, (2.1) can be simplified to

$$\mathbf{I}(p,\lambda_i) = I^b(\lambda_i) e^{-\sum_{s=1}^{3} m_i(s,\lambda_i) d(p,s)}, \tag{2.2}$$

where $I^b(\lambda_i)$ is a scalar quantity that denotes the background intensity in image or the intensity of unstained pixels, i.e., with $d(p,s)=0$. This equation is similar to the Beer-Lambert law and relates image intensity at pixel p to 's' stain's absorbance spectrum $m_i(s,\lambda_i)$ and the quantity of this stain $d(p,s)$ present at that pixel. In the optical density (OD) space, (2.2) is modified as (Ruifrok and Johnston, 2001)

$$\mathbf{I}_{\text{OD}}(p,\lambda_i) = -\log \frac{\mathbf{I}(p,\lambda_i)}{\mathbf{I}^b(\lambda_i)} = -\sum_{s=1}^{3} m_i(s,\lambda_i) d(p,s). \tag{2.3}$$

Since $\mathbf{I}(p,\lambda_i)$ and $\mathbf{I}^b(\lambda_i)$ are known for a microscopic image in each of the i^{th} sensor channel, OD values for an image can be computed in all three channels corresponding to λ_1, λ_2 and λ_3 for R, G, and B channels, respectively. Thus, OD values are obtained in a $3 \times MN$ matrix $\mathbf{I}_{\text{OD}}$.

NMF based color deconvolution methods factorize the OD matrix into stain basis vector and stain quantity matrix as:

$$\mathbf{I}_{\text{OD}} = \mathbf{\Psi A}, \tag{2.4}$$

where $\mathbf{I}_{\text{OD}}$ is a $3 \times MN$ OD matrix, $\mathbf{\Psi}$ is a 3×3 stain basis matrix representing the characteristic absorbance of stains for each of the three channels, and '**A**' is a $3 \times MN$ matrix with each column storing the quantity of each of the staining chemical with both $\mathbf{\Psi}$ and 'A' as unknown. Likewise, SVD based color deconvolution methods factorize the OD matrix as:

$$\mathbf{I}_{\text{OD}} = \mathbf{\Psi \Lambda \Phi}^T = \mathbf{\Psi A}, \tag{2.5}$$

2.3.2 GCTI-SN Method

In the proposed GCTI-SN method, the color normalization of images has been carried out with the understanding of the geometry of the underlying vector space by utilizing SVD, and it consists of three steps: 1) Illumination Correction, 2) Stain Color Vector Correction, and 3) Stain Quantity Correction (Figure-2.1).

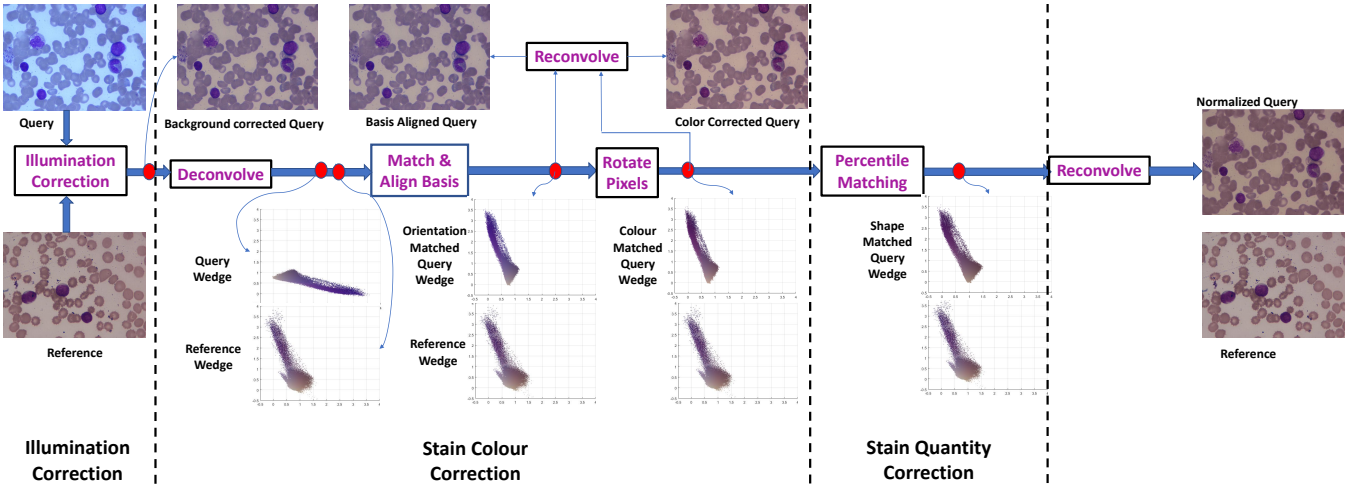


Figure 2.1: Proposed GCTI-SN Stain Normalization Method

Step-1: Illumination Correction

The illumination correction is achieved by default with (2.3) on converting the RGB image to the OD space image. However, it requires robust identification of background pixels' intensity $\mathbf{I}^b(\lambda_i)$ to be used in (2.3). To achieve this, the median filter of size 3×3 was applied on both reference and query images to remove impulse noise. Next, gray intensity images were generated. Background pixels occur in patches that are largely homogeneous with high intensities. Some isolated pixels, although fewer, belonging to stained cells, may also have high intensities. Thus, a low-pass filter (averaging kernel) was used in the neighborhood to robustly identify these background pixels. Any $N \times N$ kernel that can capture some background patches is sufficient. If the kernel size is large for a given image resolution, fewer background pixels would be identified, and correspondingly, kernel size can be decreased to identify more background pixels. A value of $N = 71$ was chosen for SN-AM (ALL) and SN-AM (MM). Next, T_{ref} and T_{Q} were identified as the 99^{th} percentile intensity on the filtered query and the reference image, where this percentile is chosen as the robust rank statistic threshold to eliminate outliers from being selected as background pixels. All pixels with intensities T_{ref} and

T_{Q} were stored in sets S_{ref} and S_{Q}, respectively. Next, R, G, B channel-wise mean of all pixels belonging to S_{Q} and S_{ref} was computed from the median filtered query and reference images to obtain average background intensity $\mathbf{I}^b_{\mathrm{Q}}(\lambda_i)$ and $\mathbf{I}^b_{\mathrm{ref}}(\lambda_i)$ for $i = 1, 2, 3$. These were used to convert the RGB images to the OD space images via (2.6) that led to illumination correction of the query image as below:

$$\mathbf{I}_{\mathrm{ref,OD}}(\lambda_{\mathbf{i}}) = -\log\frac{\mathbf{I}_{\mathrm{ref}}(\lambda_{\mathbf{i}})}{\mathbf{I}^{\mathbf{b}}_{\mathrm{ref}}(\lambda_{\mathbf{i}})}, \qquad \mathbf{I}_{\mathrm{Q,OD}}(\lambda_{\mathbf{i}}) = -\log\frac{\mathbf{I}_{\mathrm{Q}}(\lambda_{\mathbf{i}})}{\mathbf{I}^{\mathbf{b}}_{\mathrm{Q}}(\lambda_{\mathbf{i}})} \tag{2.6}$$

Step-2: Stain Color Vector Correction

The stain chemical variation manifests in the form of a) different directions and ordering of basis vectors, and b) different orientation (right-handed or left-handed) of the frames of reference and query images. SVD was used to extract stain color vectors of both the reference and query images. Since SVD orders basis depending on the singular values' magnitude that will depend on the quantity of cells/tissue in an image, the stain color vector frame of the query may be translated and rotated from the reference image frame in the 3D color vector space. Accordingly, the proposed stain color vector correction method consists of the following steps: A) Finding stain basis vectors using SVD, B) Aligning the color basis frame of the query image to that of the reference image, C) Finding robust stain color vectors using the wedge finding method for both query and reference images, and D) Providing appropriate rotation to every pixel in the OD space that aligns the wedges of both query and reference images.

1. [Step-2A:] **Finding stain basis vectors using SVD:** First, SVD was performed on illumination corrected images to obtain

$$\mathbf{I}_{\mathrm{ref,OD}} = \mathbf{\Psi}_{\mathrm{ref}}\mathbf{\Lambda}_{\mathrm{ref}}\mathbf{\Phi}^{\mathbf{T}}_{\mathrm{ref}} = \mathbf{\Psi}_{\mathrm{ref}}\mathbf{A}_{\mathrm{ref}}, \qquad \mathbf{I}_{\mathrm{Q,OD}} = \mathbf{\Psi}_{\mathrm{Q}}\mathbf{\Lambda}_{\mathrm{Q}}\mathbf{\Phi}^{\mathbf{T}}_{\mathrm{Q}} = \mathbf{\Psi}_{\mathrm{Q}}\mathbf{A}_{\mathrm{Q}}, \tag{2.7}$$

   where $\mathbf{\Psi}_{\mathrm{ref}}$ and $\mathbf{\Psi}_{\mathrm{Q}}$ are 3×3 matrices representing the stain basis vectors of reference image and query image, respectively and, $\mathbf{A}_{\mathrm{ref}}$ and $\mathbf{A}_{\mathrm{Q}}$ are $3 \times MN$ matrices with each column storing the quantity corresponding to each of the three constituent basis vectors of the staining chemical (Li and Plataniotis, 2015; Ruifrok and Johnston, 2001).

Step-2B: **Alignment of color basis:** Frame alignment of reference and query requires (a) translation of query frame to align the query frame origin to the reference frame origin, (b) establishing the correspondence of respective basis vectors (e.g., 1st basis vector of the reference image may mostly capture nucleus color, while 1st vector of the query may correspond to the background color), and (c) matching the frame orientation because it may be right-handed in one and left-handed in the other image. Part (a) of frame origin alignment was taken care of by the illumination correction, as explained earlier. For (b), there are six possible pairings of corresponding reference and query stain vectors for three stain vectors. The pairing with the least distance was chosen, where distance for each pairing, say pairing ((1,2), (2,3), (3,1)), was calculated as:

$$D = \sqrt{||\mathbf{\Psi}_1^{\text{ref}} - \mathbf{\Psi}_2^{\text{Q}}||_2^2 + ||\mathbf{\Psi}_2^{\text{ref}} - \mathbf{\Psi}_3^{\text{Q}}||_2^2 + ||\mathbf{\Psi}_3^{\text{ref}} - \mathbf{\Psi}_1^{\text{Q}}||_2^2}, \tag{2.8}$$

where $||.||_2$ denotes l^2-norm and $\mathbf{\Psi}_i^{\text{ref}}$ refers to the i^{th} basis or column vector of matrix $\mathbf{\Psi}_{\text{ref}}$.

Frame orientation can be checked by considering the distance with the negative directions of all vectors, taken one at a time. Thus, distance for 24 combinations was computed using (2.8), say, for pairing ((1,2), (2,3), (3,1)), for pairing ((1,-2), (2,3), (3,1)), and so on. Accordingly, the ordering and sign of rows of $\mathbf{A}_{\text{Q}}$ (that contain quantity over the corresponding frame basis vectors) was changed to match the pairing with the least distance. This corresponded to the basis vector transformation for query frame alignment. This process is illustrated in Figure-2.2. The impact of this alignment is shown on one of the H&E stained images of SN-AM (MM) in Figure-2.3.

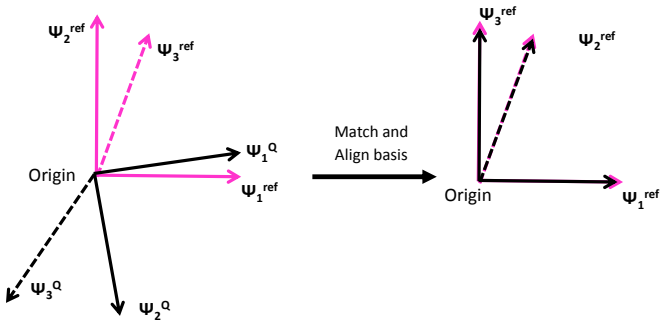


Figure 2.2: Proposed frame alignment where origins are matched via illumination correction

Step-2C: **Stain vector basis determination by wedge finding:** As presented by Macenko *et al.* (2009), pixel intensities were projected on the space (here called

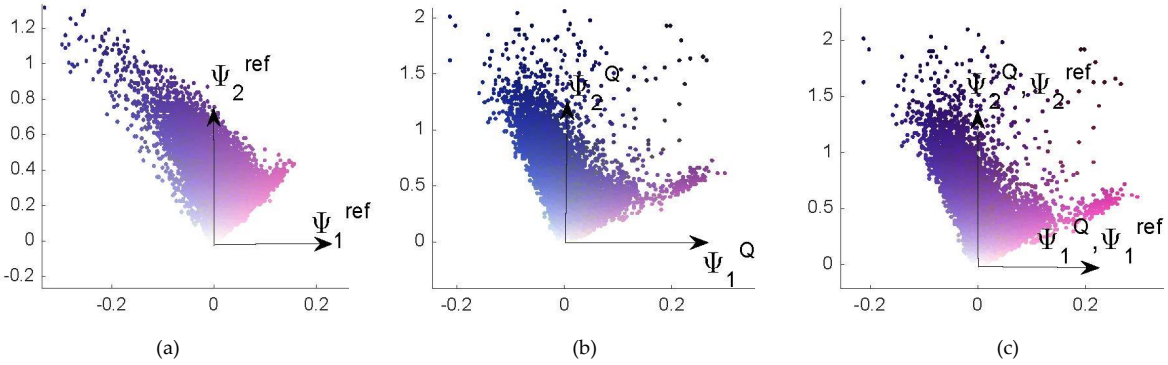


Figure 2.3: (a) Reference image intensities over two reference basis vectors (b) Query image intensities over two query basis vectors (c) Query image intensities over basis vectors after alignment of query frame with that of reference frame. Note that the color shade in Figure-2.3c is similar to Figure-2.3a. This figure should be seen in color to visualize the significance of the 'proposed frame alignment" shown in Figure-2.2.

as color space) spanned by two largest singular vectors $\mathbf{\Psi}_1^{\text{ref}}$ and $\mathbf{\Psi}_2^{\text{ref}}$ of the reference image. Next, we normalized each pixel p defined by coordinates (i, j) in the color space to unit norm such that $p = (i, j)$ changes to $p_n = (i', j') = \left(\frac{i}{\sqrt{i^2+j^2}}, \frac{j}{\sqrt{i^2+j^2}}\right)$. Then, we computed angles of all the pixels as $\theta = tan^{-1}\left(\frac{j'}{i'}\right)$ and ordered all the angles. We set the unit vector in the direction of the 0.1 percentile of all the angles as $\mathbf{\Phi}_1^{\text{Q}}$ and unit vector in the direction of the 99.9th percentile of all the angles as $\mathbf{\Phi}_2^{\text{Q}}$. These two vectors form the wedge defined by the pixels by eliminating outliers (by considering 0.1 percentile and the 99.9th percentile). This process is shown in Figure-2.4 and was completed for both the reference and query images leading to modified stain color vectors $\mathbf{\Phi}_i^{\text{ref}}$ and $\mathbf{\Phi}_i^{\text{Q}}$, respectively, for $i = 1, 2$.

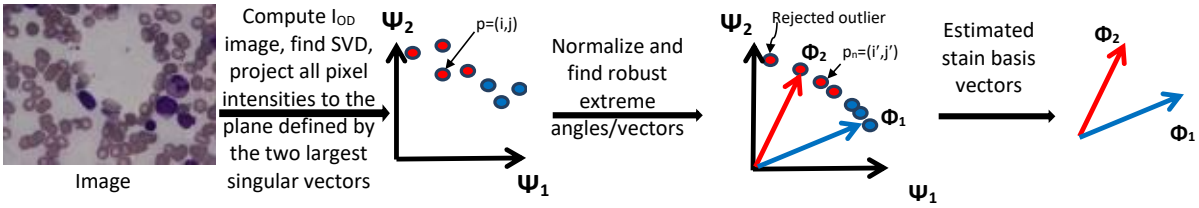


Figure 2.4: Robust stain color vector determination using wedge method (Macenko *et al.*, 2009)

Step-2D: **Rotation of every pixel's color according to color wedge basis:** It is quite likely that the angle between reference stain vectors $\mathbf{\Phi}_1^{\text{ref}}$ and $\mathbf{\Phi}_2^{\text{ref}}$ may be different than that between query stain vectors $\mathbf{\Phi}_1^{\text{Q}}$ and $\mathbf{\Phi}_2^{\text{Q}}$. For example, the first angle is α,

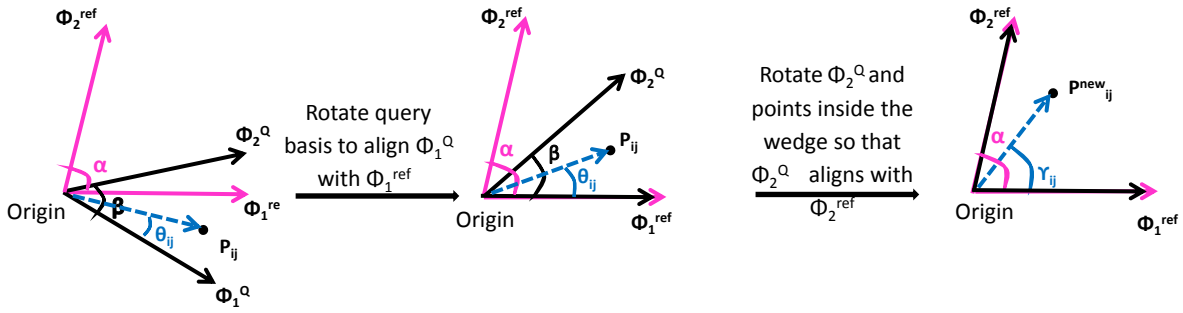


Figure 2.5: Proposed rotation to every pixel's color according to color wedge basis

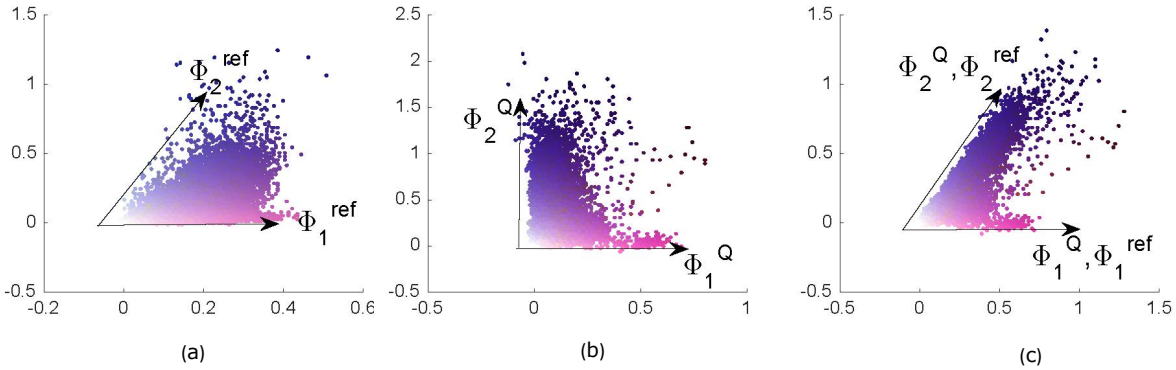


Figure 2.6: (a) Wedge of reference image (b) Wedge of query image (c) Query wedge aligned with reference wedge. This figure should be seen in color to visualize the importance of the "proposed rotation to pixels' color according to wedge basis" shown in Figure-2.5.

while the second angle is β (Figure-2.5). Therefore, there is a need to align the wedge of query to that of reference image and correspondingly, provide transformation to every pixel intensity within the query image wedge. This alignment was achieved as follows. First, $\mathbf{\Phi}_1^{\mathrm{Q}}$ was aligned with $\mathbf{\Phi}_1^{\mathrm{ref}}$. Next, $\mathbf{\Phi}_2^{\mathrm{Q}}$ was rotated counterclockwise by $\alpha - \beta$ such that it aligns with $\mathbf{\Phi}_2^{\mathrm{ref}}$. Correspondingly, rotation was provided to all pixels contained inside the query wedge (refer to pixel P_{ij} in Figure-2.5) by $\gamma_{ij} = \theta_{ij}\frac{\alpha}{\beta}$, where θ_{ij} is computed as $\theta_{ij} = \tan^{-1}\left(\frac{(\mathbf{\Phi}_1^{\mathrm{ref}} \times \mathbf{p}_{ij}).\hat{\boldsymbol{n}}}{\mathbf{\Phi}_1^{\mathrm{ref}}.\mathbf{p}_{ij}}\right)$, where $\mathbf{p}_{ij}$ is the vector associated with the pixel position P_{ij}, '$\times$' denotes the cross product between two vectors, '.' denotes the scalar product between two vectors, and $\hat{\boldsymbol{n}}$ is the unit vector defined as $\hat{\boldsymbol{n}} = \left(\frac{\mathbf{\Phi}_1^{\mathrm{ref}} \times \mathbf{p}_{ij}}{|\mathbf{\Phi}_1^{\mathrm{ref}} \times \mathbf{p}_{ij}|}\right)$ with '$\times$' again denoting the cross product. This is to note that incorporation of unit vector $\hat{\boldsymbol{n}}$ takes care of the sign of θ_{ij} that can be negative for some pixels. The query image wedge was aligned with that of the reference image (Figure-2.6), completing the procedure of stain color vector correction.

Step-3: Stain Quantity Correction

The histogram-based method for stain quantity correction was applied as described by Macenko *et al.* (2009), wherein 99.9^{th} percentile of the query stain quantity was adjusted to the 99.9^{th} percentile of the reference stain quantity. A value of 99.9^{th} percentile was chosen to eliminate random outliers.

Note: The above three steps complete the GCTI-SN method of color normalization for the microscopic images and are presented in Algorithm-1. Step-1, Step 2B, and Step 2D are new in the GCTI-SN method. Researchers have addressed the Step-1 of illumination correction in different ways. GCTI-SN method, although simple, is effective. Step-2B and Step-2D have been introduced newly in the SVD based approach (Step-2A) after understanding the geometry of the problem. Thus, these are new additions in matrix decomposition-based methods. Many researchers have used SVD. However, SVD only provides the order of vectors depending on the content of different components. For example, an image can have ten cells of interest, and another may have only one cell of interest. Hence, the order of stain constituent vectors may be different after SVD in these two images. Step-2B and 2D have augmented the SVD based methods. This is a significant novelty making SVD based approach stable and robust.

2.4 Results

The impact of each GCTI-SN method's stages was evaluated quantitatively and qualitatively, and comparisons were made with the existing methods. The quantitative validation has been done by the root mean squared error (RMSE) on BRCA-08 and by mean squared distance (MSD) computed between reference and query images on SN-AM (ALL) and SN-AM (MM). Since deep learning or machine learning-based diagnostic tools (e.g., cell or tissue segmentation/classification/cancer detection) require color distribution of training and testing data to be the same, stain normalization may be used to bring the color distribution of images collected at different centers/time closer to that of the selected reference image. This similarity in distribution can be best captured in the mean sense via these metrics.

For BRCA-08 which contains H&E images, evaluation of stain normalization method is carried out as suggested by Van Eycke *et al.* (2017). Query and reference images were deconvolved using (2.4). Matrices $\mathbf{\Psi}$ and $\mathbf{A}$ were used to generate two images corresponding to each of the single stain constituent H and E, respectively. This was achieved by setting the row of $\mathbf{A}$ corresponding to the discarded stain constituents to zero. Finally, the H only and E only images obtained from the query, and reference images were converted to grayscale H only and grayscale E only images by adding their respective R, G, B channels. The KS-Statistic provided by the two-sample Kolmogorov-Smirnov test was used as the similarity metric. Also, the Q-Q plot was drawn between each query and reference image. The root means squared error (RMSE) of the plot to the line $X = Y$ was used as another metric of comparison.

For SN-AM (ALL) and SN-AM (MM), the regions of interest correspond to nuclei. Mean Square Distance (MSD) has been computed between pixels sampled from nuclei of reference and query images on a mask containing at least three nuclei per image as below:

$$\text{MSD} = \sqrt{\sum_{i \in \{\text{R,G,B}\}} (\mathbf{I}^{\text{ref}}_{i,\text{avg}} - \mathbf{I}^{\text{Q}}_{i,\text{avg}})^2} \tag{2.9}$$

where $\mathbf{I}^{\text{ref}}_{\text{i,avg}}$ is the average intensity, of all pixels in the mask, in channel i of the reference image. These patches are quite large. For example, each cell image has, on average, around 300×300 pixels. Hence, an image with three cells, the average MSD, might have been calculated on $3 \times 0.5 \times 300 \times 300 = 135000$ no. of pixels, which is quite a large number. Within the mask, pixels with low and high color values are present, preserving the texture of cells that would eventually help in cancer diagnosis. It is only in the mean sense that the distribution of data is brought closer to the entire dataset (training and testing data). In other words, a lower value of MSD with the proposed GCTI-SN indicates the more closeness between the training and testing data in the mean sense. Indeed, the deep learning solution on diagnostics with stain normalization works considerably well, as shown in Section-4.

2.4.1 Efficacy of Illumination Correction module

Illumination correction was achieved by matching the background intensities of the query and reference images. Often, pixels with maximum intensity in a channel are

assumed to be the background pixels (Li and Plataniotis, 2015). Although it worked for the H&E images (BRCA-08) where the background intensities were close to 255, it yielded the selection of wrong pixels as background in the SN-AM (ALL) and SN-AM (MM).

To measure the accuracy of the proposed illumination correction, MSD was calculated using (2.9) on pixels sampled from the background using a mask for each image in the SN-AM (ALL) and SN-AM (MM). The average MSD for these datasets is shown in Table-2.2.

Table 2.2: Average MSD (with p-value) calculated on background and nucleus patches after illumination correction

	Average MSD on Background (p-value)			Average MSD on Nucleus (p-value)		
Dataset	Original	Li et al.(2015)	Proposed	Original	Li et al.(2015)	Proposed
SN-AM (ALL)	41.11	$\mathbf{9.23(7.10^{-6})}$	$\mathbf{3.84(6.10^{-8})^*}$	29.84	$\mathbf{16.23(7e-6)}$	$\mathbf{14.59(7.10^{-6})}$
SN-AM (MM)	24.96	$\mathbf{12.36(7.10^{-6})}$	$\mathbf{6.81(6.10^{-8})^*}$	20.01	$\mathbf{10.87(8.10^{-7})}$	$\mathbf{10.08(5.10^{-5})}$

p-values calculated by a paired, one-sided Sign test between the MSD of the original image patches and the MSD of the Illumination corrected images of the matched samples at a significance value of $p \leq 0.05$ are shown in brackets. Statistically significantly improved performances are shown in bold. Statistically significant best performance, as verified by the multi-comparison Friedman test followed by Nemenyi posthoc test, is indicated with '*' in this Table.

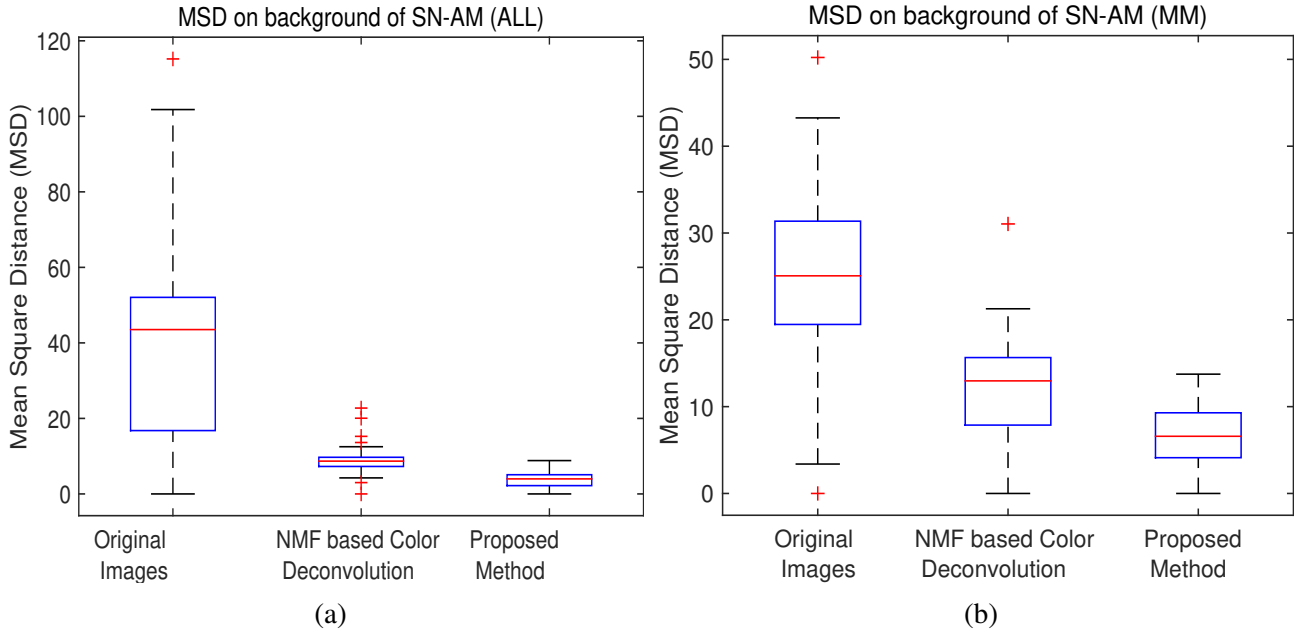


Figure 2.7: MSD after Illumination correction on the background of (a) SN-AM (ALL) (b) SN-AM (MM)
Computed between reference image and (i) original images, (ii) images from NMF based color deconvolution method (Li and Plataniotis, 2015), and (iii) images from the proposed method.

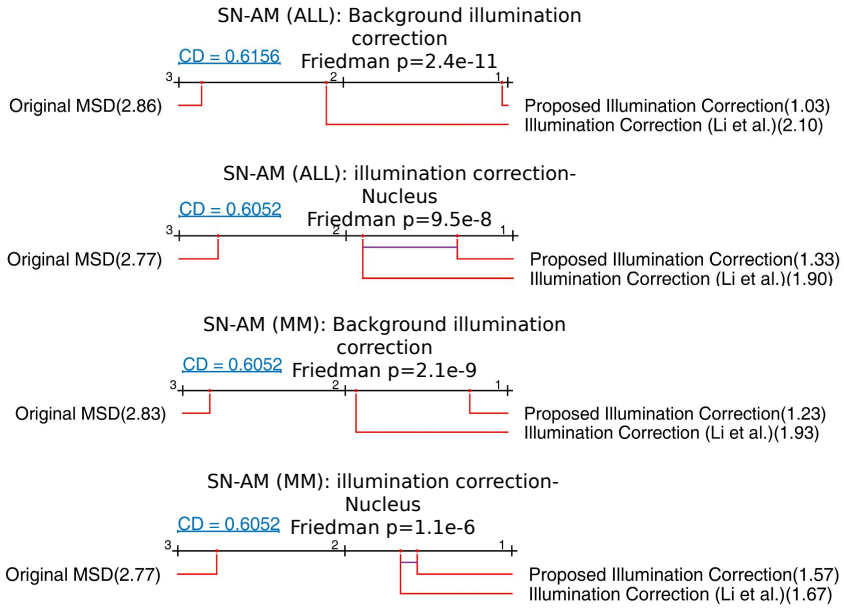


Figure 2.8: Results with multi-comparison Friedman test followed by Nemenyi posthoc test for Table-1. Numbers in bracket indicate rank averaged over all images of the dataset, where lowest rank indicates best performance with least MSD error.

On the SN-AM (ALL), the average MSD of the background reduced from 41.11 to 3.84 ($p << 0.05$) with the proposed illumination correction method as compared to 9.23 achieved with the existing method (Li and Plataniotis, 2015). Similarly, for the SN-AM (MM), the MSD of the background reduced from 24.96 to 6.81 ($p << 0.05$) compared to 12.36 achieved with the existing method (Li and Plataniotis, 2015). Also, the median MSD and the overall MSD of the background were considerably reduced with the proposed method on both the dataset as compared to the existing method (Li and Plataniotis, 2015) (Figure-2.7). Moreover, there was no outlier with a large MSD with the proposed method. Results were further validated by the multiple comparison Friedman test ($p = 2.10^{-11}$ on SN-AM (ALL) and $p = 2.10^{-9}$ on SN-AM (MM)) followed by Nemenyi posthoc test (Figure- 2.8), wherein the proposed method ranked first with lowest MSD on both the dataset with a distance greater than the critical distance (CD) from the next ranked method.

Further, we evaluated if the illumination correction also improved performance on the nucleus, which is the region of interest for these two datasets. The average MSD between the reference image nucleus and the query image nucleus reduced significantly ($p << 0.05$) on both the dataset with the proposed method (14.59 from 29.84 on SN-AM (ALL) and 10.08 from 20.01 on SN-AM (MM)). This was further validated by the multiple comparison Friedman test ($p = 9.10^{-8}$ on SN-AM (ALL) and $p = 10^{-6}$

on SN-AM (MM)). Although the proposed method was ranked first by the Nemenyi posthoc test with the lowest MSD on the nucleus on both the dataset, both the existing method (Li and Plataniotis, 2015) and the proposed method yielded statistically improved results with no statistical difference between them as verified by Nemenyi posthoc test (Figure- 2.8). Further, it was noted that MSD for the nucleus reduced to half compared to the original query images but was still quite large. These results showed that illumination correction is a required step, but is not a sufficient step. Thus, further processing of illumination-corrected query images is needed.

The significant difference in the MSD performance of both the existing and the proposed methods can be assessed by the differences in the selection of background pixels by these methods (Figure-2.9). Different background locations are selected in R, G, and B channels by the existing method (Li and Plataniotis, 2015), where some of these locations do not belong to the background (say red pixels shown in Figure-2.9b). On the other hand, the proposed method chooses background patches robustly owing to the criterion of homogeneity enforced on patch selection in our method (Figure-2.9 c), leading to better performance of the proposed method.

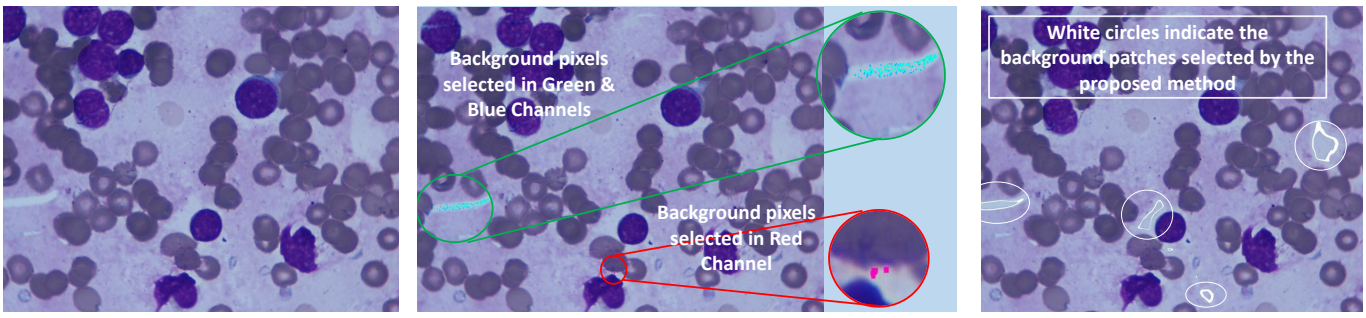


Figure 2.9: Background pixel selection on (a) Query Image (b) by the existing method (Li and Plataniotis, 2015) and (c) by the proposed method in GCTI-SN pipeline.

2.4.2 Efficacy of Stain Color Correction Module

The success of stain color correction depends on the quality of deconvolution achieved through a stain basis matrix determined using matrix factorization. While previous works achieved stain color correction by replacing the query color basis (obtained via SVD or NMF) with the reference color basis, we attempted the same by transforming the query basis to the reference basis.

The quantitative comparisons on BRCA-08 (H&E) were carried out using the RMSE on Q-Q plots and KS-Statistic by a two-sample KS-Test. The RMSE increased from 0.20 to 0.51 ($p < 0.05$) in the Hematoxylin channel and increased from 0.10 to 0.41 ($p = 0.05$) in the Eosin channel with SVD-based basis replacement and hence, the performance deteriorated (Table- 2.3). With the NMF based basis replacement, no improvement in RMSE and KS-statistics were noted.

Table 2.3: Quantitative evaluation of Stain Color Correction module on BRCA-08 (H&E).

	RMSE (p-value)			KS-Statistic (p-value)		
	Original before correction	Correction via replacement	Correction via transformation (Proposed)	Before correction	Correction via replacement	Correction via transformation (Proposed)
			Hematoxylin basis channel			
SVD	0.20	$\mathbf{0.51(0.02)}$	$\mathbf{0.17(0.05)}^*$	0.34	$\mathbf{0.28(6.10^{-6})}$	$\mathbf{0.26(10^{-5})}^*$
NMF	0.23	0.22 (0.64)	0.23 (0.52)	0.27	0.28 (0.008)	0.27 (0.01)
			Eosin basis channel			
SVD	0.10	$0.41(0.05)$	0.11 (0.88)	0.25	$0.29(2.10^{-5})$	$0.26(4.10^{-4})$
NMF	0.19	$0.18(4.10^{-3})$	$\mathbf{0.01(4.10^{-16})}^*$	0.44	$0.41(6.10^{-4})$	$\mathbf{0.23(4.10^{-8})}^*$

p-values, shown in brackets, are calculated by paired, one-sided Sign test comparing the values with the matched samples of "Original before correction" column at a significance value of $p \leq 0.05$. 'Red' colored quantities indicate deterioration in the quality of images as verified statistically, while statistically improved performances are shown in bold. Statistically significant best performance, as confirmed by the multi-comparison Friedman test followed by Nemenyi posthoc test, is indicated with '*' in the Table.

Using SVD based basis transformation, a statistically significant reduction in both RMSE (0.20 to 0.17 with $p < 0.05$) and KS statistic (0.34 to 0.26 with $p << 0.05$) was observed in the Hematoxylin channel, but no significant change was observed in the Eosin channel. With NMF-based basis transformation, a statistically significant reduction in both the RMSE (0.19 to 0.01 with $p << 0.05$) and KS metric (0.44 to 0.23 with $p << 0.05$) was observed in the Eosin channel. These findings were further validated by the multiple comparisons Friedman test ($p < 10^{-9}$) followed by Nemenyi posthoc test, where the basis replacement method ranked first with distance more than the critical distance from the next ranked method. These findings indicate that basis transformation is effective, and it either leads to improved performance or leaves the images unaltered. In contrast, basis replacement methods mostly led to the deterioration in quality.

The quantitative comparisons on SN-AM (ALL) and SN-AM (MM) datasets were

carried out using the average MSD on nuclei. On the SN-AM (ALL) dataset, NMF based stain matrix was initialized using the average saturation values (S channel in HSV color space) of pixels belonging to the nucleus, unstained cells, and the background. In contrast, it was initialized with the average saturation values (S channel in HSV color space) of pixels belonging to the nucleus, cytoplasm, and background for the SN-AM (MM) dataset.

Using SVD basis transformation, MSD in SN-AM (ALL) reduced from 29.84 to 13.51 ($p < 0.05$), while it reduced marginally to 25.79 ($p = 0.94$; Table-2.4) using NMF with basis transformation. With the conventional SVD basis replacement method, MSD did not change from the original value and remained at 29.8 and increased to 33.81 with NMF with basis replacement. In fact, stain basis replacement without aligning the basis led to color flipping on some images (Figure-2.10c). On the other hand, basis transformation consistently performed well, as evident from Table-2.4 and Figure-2.10d. This was further validated by the multiple comparison Friedman test ($p = 10^{-9}$) followed by Nemenyi posthoc test, wherein SVD with transformation ranked first with the distance more than the critical distance from the rank-2 method.

On SN-ALL (MM) dataset, MSD reduced from 20.01 to 8.54 ($p = 7.10^{-6}$ using using SVD basis transformation and it reduced from 20.01 to 11.85 ($p = 0.02$; Table-2.4) using NMF with basis transformation. With the conventional SVD basis replacement method, the MSD reduced to 10.76 ($p < 0.05$) but increased to 42.03 ($p < 0.05$) with NMF with basis replacement. Results were significant with the multiple comparison Friedman test ($p = 3.10^{-7}$). Although SVD with transformation ranked first with Nemenyi posthoc test, the distance was not more than the critical distance from the rank-2 method (Figure- 2.11). In the conventional methods, SVD with basis replacement performed well. But NMF with basis replacement was particularly poor in performance, perhaps because it is critically dependent on the initialization, where the right one is usually difficult to find (Li and Plataniotis, 2015).

Table 2.4: Quantitative evaluation of Stain Color Correction module on SN-AM (ALL) and SN-AM (MM).

Average MSD on Nucleus (p-value)					
Dataset	Original before correction	SVD with basis replacement	SVD with basis transformation (Proposed)	NMF with basis replacement	NMF with basis transformation (Proposed)
SN-AM (ALL)	29.84	29.8 (0.5)	$\mathbf{13.51(7.10^{-6})}^*$	33.81 (0.07)	25.79 (0.94)
SN-AM (MM)	20.01	$\mathbf{10.76(5.10^{-5})}$	$\mathbf{8.54(7.10^{-6})}$	42.03(0.02)	$\mathbf{11.85(0.02)}$

p-values, shown in brackets, are calculated by a paired, one-sided Sign test comparing the values with the matched samples of the "Original before correction" column at a significance value of $p \leq 0.05$. Inferences are made after statistical testing. 'Red' colored quantities indicate deterioration in the quality of images, while improved performances are shown in bold. Best performance, as verified by the multi-comparison Friedman test followed by Nemenyi posthoc test, is indicated with '*' in the Table.

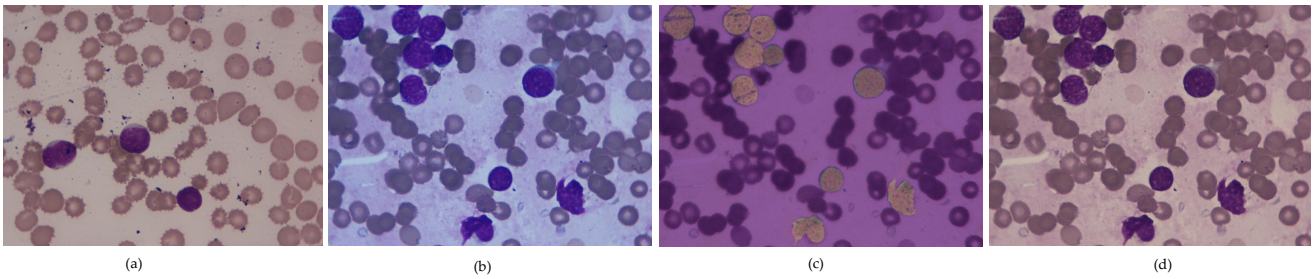


Figure 2.10: Visual representation of stain color correction on SN-AM (ALL) (a) Reference image (b) Query image (c) Stain correction through SVD basis replacement showing color flipping (d) Stain correction through SVD basis transformation

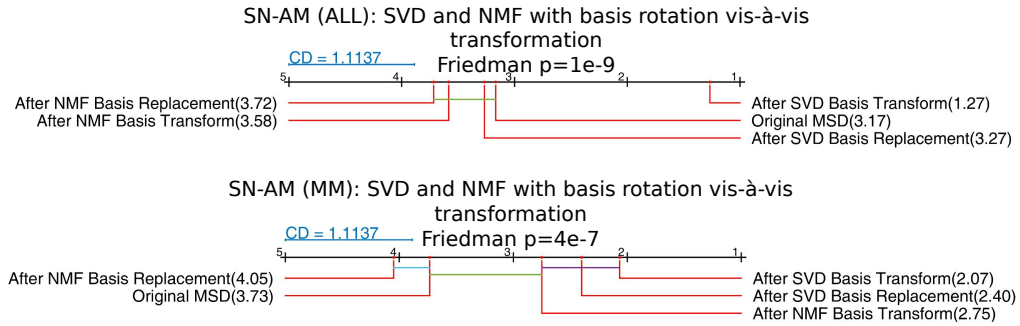


Figure 2.11: Results with multi-comparison Friedman test followed by Nemenyi posthoc test for Table-3. Numbers in bracket indicate rank averaged over all images of the dataset, where lowest rank indicates best performance with least MSD error.

2.4.3 Efficacy of Stain Quantity Correction Module

The performance of stain quantity correction was evaluated on all the three datasets in both SVD and NMF based stain color deconvolution. For BRCA-08 (H&E), SVD based deconvolution with quantity correction provided statistically improved performance in RMSE (0.20 to 0.08; $p << 0.05$) and KS-Statistics (0.34 to 0.26; $p << 0.05$; Table-2.5) in the Hematoxylin channel. Both RMSE and KS-statistics proved significant by

the multiple comparison Friedman test ($p = 9.10^{-15}$ and $p = 8.10^{-11}$) followed by Nemenyi posthoc test, wherein quantity correction ranked first. Similarly, with NMF based deconvolution, improvement in RMSE (0.23 to 0.14; $p << 0.05$) and KS-Statistics (0.27 to 0.23; $p << 0.05$) was observed. Both RMSE and KS-statistics proved significant by the multiple comparison Friedman test ($p = 5.10^{-18}$ and $p = 2.10^{-9}$) followed by Nemenyi posthoc test, wherein quantity correction ranked first. In the Eosin channel, SVD based deconvolution with quantity correction left the images unaltered, while the NMF based deconvolution with quantity correction provided statistically improved performance in RMSE (0.005 from 0.19 at $p << 0.05$) and KS-Statistics (0.23 from 0.44 at $p << 0.05$). Again both RMSE and KS-statistics proved significant by the multiple comparison Friedman test ($p = 7.10^{-21}$ and $p = 3.10^{-10}$).

Table 2.5: Quantitative evaluation of Stain Quantity Correction module on BRCA-08 (H&E).

	RMSE (p-value)			KS-Statistic (p-value)		
	Original before correction	After proposed stain color correction	After stain quantity correction	Original before correction	After proposed stain color correction	After stain quantity correction
			Hematoxylin basis channel			
SVD	0.20	0.17 (0.06)	$\mathbf{0.08(3.10^{-12})}^*$	0.34	$\mathbf{0.26(2.10^{-5})}$	$\mathbf{0.26(2.10^{-10})}^*$
NMF	0.23	0.23 (0.52)	$\mathbf{0.14(7.10^{-18})}^*$	0.27	0.27(0.01)	$\mathbf{0.23(2.10^{-4})}^*$
			Eosin basis channel			
SVD	0.10	0.11 (0.6)	$\mathbf{0.10(0.03)}$	0.25	0.26 (0.12)	0.25 (4e-4)
NMF	0.19	$\mathbf{0.01(4.10^{-16})}$	$\mathbf{0.005(4.10^{-16})}^*$	0.44	$\mathbf{0.23(5.10^{-8})}$	$\mathbf{0.23(8.10^{-9})}^*$

p-values are calculated by a paired, one-sided Sign test comparing the values with the matched samples of the "Original before correction" column at a significance value of $p \leq 0.05$. 'Red' colored quantities indicate deterioration in the quality of images as verified statistically, while statistically improved performances are shown in bold. Statistically significant best performance, as confirmed by the multi-comparison Friedman test followed by Nemenyi posthoc test, is indicated with '*' in the Table.

On the SN-AM (ALL) dataset, the average MSD has reduced considerably from 29.84 to 6.52 at $p << 0.05$ over a set of 30 images after running the complete normalization pipeline, including stain quantity correction (Table-2.6). The SN-AM (MM) dataset showed improvement from 20.014 to 9.86 at $p << 0.05$ with stain quantity correction. Multiple comparison Friedman test indicated results to be statistically significant on SN-AM (ALL) dataset ($p = 2.10^{-9}$) and on SN-AL (MM) dataset ($p = 4.10^{-6}$). The Nemenyi posthoc test reported complete pipeline to be ranked one on SN-AM (ALL) dataset with results from successive stages (illumination correction, stain color vector correction, and stain quantity correction) to be more than critical distance away

from the original MSD (Figure- 2.12). The stages within themselves were far apart on the Nemenyi posthoc test after excluding original MSD in the multiple comparison test on SN-AM (ALL) dataset. In contrast, performance on different stages was not that different from multiple comparison tests with SN-AM (MM) dataset. In addition, we also compared the average MSD obtained from the full stain normalization pipeline, but with and without Step-2D to study the impact of this proposed novelty on both the dataset. Step-2D performed statistically significantly in SN-AM (ALL) dataset ($p < 0.05$), while it did not make any impact on SN-AM (MM) dataset.

Table 2.6: Quantitative evaluation of Stain Quantity Correction module on SN-AM (ALL) and SN-AM (MM).

Average MSD on Nucleus (p-value) with SVD based deconvolution pipeline				
Dataset	Original before correction	After proposed stain illumination correction	After proposed stain color basis correction via SVD	After stain quantity correction
SN-AM (ALL)	29.84	$\mathbf{14.596(7.10^{-6})}$	$\mathbf{13.513(5.10^{-5})}$	$\mathbf{6.52(7.10^{-6})}$
SN-AM (MM)	20.014	$\mathbf{10.082(5.10^{-5})}$	$\mathbf{8.544(7.10^{-6})}$	$\mathbf{9.86(0.001)}$

p-values are calculated by paired, one-sided Sign test comparing the values with the matched samples of "Original before correction" at a significance value of $p \leq 0.05$. Statistically improved performances are shown in bold.

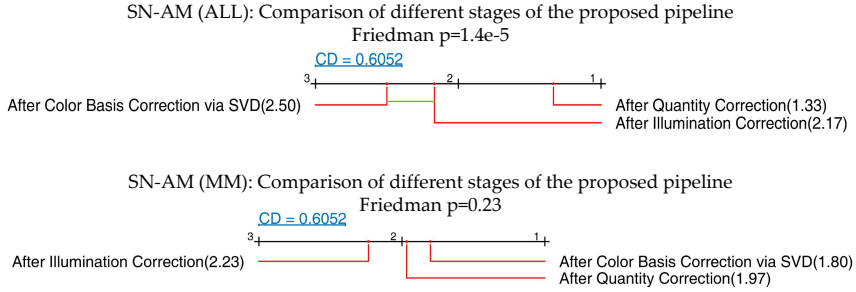


Figure 2.12: Results with multi-comparison Friedman test followed by Nemenyi posthoc test for Table-2.5. Numbers in bracket indicate rank averaged over all images of the dataset, where lowest rank indicates best performance with least MSD error.

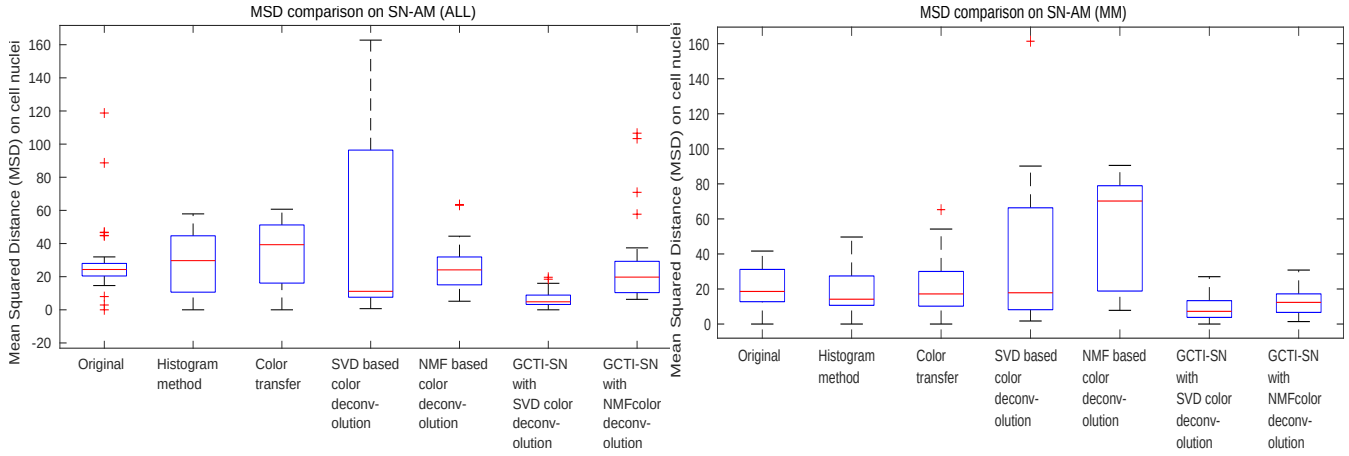
The above results show that of the three dataset considered above, stain quantity correction module improved performance significantly on the two dataset (BRCA-08 and SN-AM (ALL)). The SN-AM (MM) did not show improvement with quantity correction, although the performance was still comparatively better with inclusion of this step compared to the original images. This shows that stain quantity correction is indeed required. A comparison with all methods on SN-AM (ALL) and SN-AM (MM) with MSD on individual images over both the dataset with different methods are shown in Table-2.7 and Figure-2.13.

Proposed GCTI-SN with SVD based deconvolution provides the best performance

Table 2.7: Comparison of the MSD (p-value) obtained with the proposed and the existing methods

Dataset	Original before correction	Histogram (Jain, 1989)	Color Transfer (Reinhard *et al.*, 2001)	SVD based deconvolution (Macenko *et al.*, 2009)	NMF based deconvolution (Li and Plataniotis, 2015)	GCTI-SN with SVD deconvolution	GCTI-SN with NMF deconvolution
SN-AM (ALL)	29.84	28.83(0.9)	36.90(0.01)	47.69(0.3)	26.16(0.6)	$\mathbf{6.52(7.10^{-6})}^{*}$	27.44(0.4)
SN-AM (MM)	20.01	19.24(0.4)	20.97(0.5)	35.60(0.05)	$52.58(7.10^{-4})$	**9.86(0.001)**	**13.20(0.05)**

p-values, shown in brackets, are calculated by paired, one-sided Sign test comparing the values with the matched samples of "Original before correction" column at a significance value of $p \leq 0.05$. 'Red' colored quantities indicate deterioration in the quality of images as verified statistically, while statistically improved performances are shown in bold. Statistically, significant best performances, as confirmed by the multi-comparison Friedman test followed by Nemenyi posthoc test, are indicated with '*' in the Table.



(a) (b)

Figure 2.13: Boxplots showing comparison of MSD on (a) SN-AM (ALL) dataset and (b) SN-AM (MM) dataset with original images and images obtained with: Histogram method (Jain, 1989), Color transform (Reinhard *et al.*, 2001), SVD based color deconvolution method (Macenko *et al.*, 2009), NMF based color deconvolution method (Li and Plataniotis, 2015), GCTI-SN (proposed) based on SVD color deconvolution, and GCTI-SN (proposed) based on NMFcolor deconvolution.

on the SN-AM (ALL) dataset with an average MSD of 6.52 from 29.84 on the original dataset at $p << 0.05$. This finding was further validated by the multiple comparisons Friedman test ($p < 2.10^{-8}$) followed by Nemenyi posthoc test, where the proposed GCTI-SN (with SVD) ranked first with distance more than the critical distance from the next ranked method (GCTI-SN with NMF) (Figure- 2.14). Similarly, GCTI-SN provides the best performance on the SN-AM (MM) dataset with the lowest average MSD of 9.86 from 20.01 at $p << 0.05$. Again, this finding was further validated by the multiple comparisons Friedman test ($p < 3.10^{-9}$) followed by Nemenyi posthoc test, where the proposed GCTI-SN (with SVD) ranked first with distance. The distance between GCTI-SN with SVD and GCTI-SN with NMF was not more than the critical distance, and they performed better than the rest (Figure- 2.14). The overall averaged MSD (over the entire dataset) deteriorated with the color transfer method on SN-AM (ALL) dataset at $p = 0.01$. In contrast, it declined on SN-AM (MM) dataset with exist-

ing color base replacement-based SVD and NMF deconvolution methods at $p < 0.05$. The median MSD error of the GCTI-SN with the SVD color deconvolution method was least with a few or no outliers compared to all other techniques, including proposed GCTI-SN with NMF based color deconvolution method (Figure-2.13). Quantitative comparative results on individual images of SN-AM (ALL) and SN-AM (MM) dataset are shown in Figure-2.15.

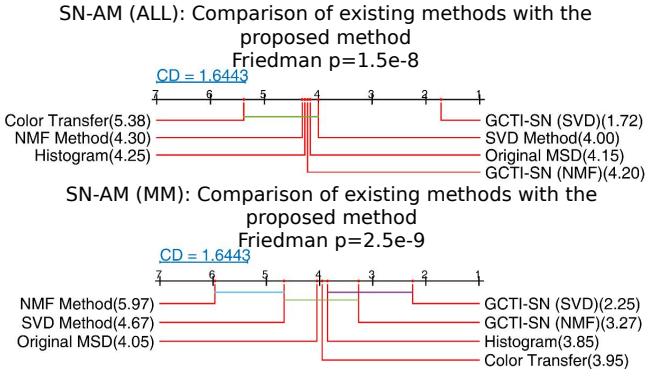
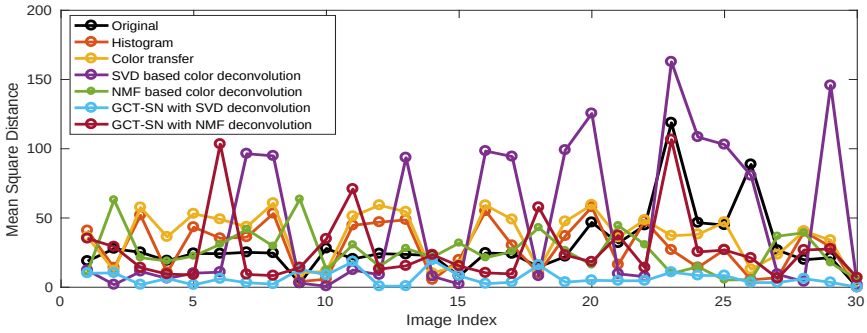


Figure 2.14: Results with multi-comparison Friedman test followed by Nemenyi posthoc test for Table- 2.7. Numbers in bracket indicate rank averaged over all images of the dataset, where lowest rank indicates best performance with least MSD error.

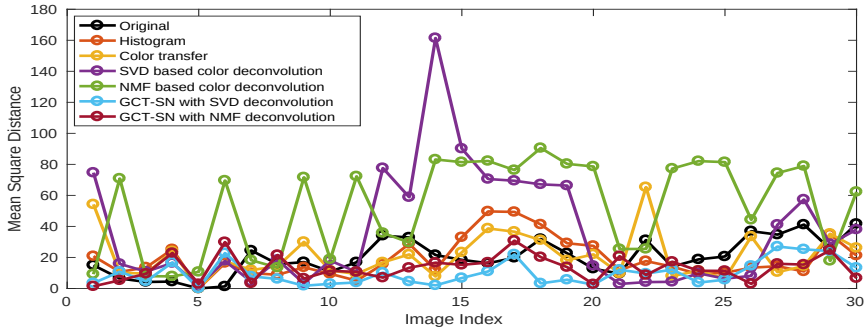
Visual comparisons on three images of each of the datasets are shown in Figures-2.16, 2.17, and 2.18. Of the existing methods, results are significant for color transfer method, SVD based color deconvolution and NMF based color deconvolution, while GCTI-SN with SVD performs better than GCTI-SN with NMF. Hence, visual results are shown with only these methods. GCTI-SN with SVD outperformed all the methods. Since existing SVD and NMF based methods do not align the query basis frame to reference basis frame, flipping of colors was observed in some of the normalized images of SN-AM (ALL) and SN-AM (MM).

2.5 Application

Stain normalization is used to counter the effect of stain color variation/distribution in the captured images that can impact diagnostic performance too adversely. For example, if a diagnostic tool is prepared from the images of one hospital, it should also work on the images captured from another hospital. In other words, in a real-life environment, a computer-aided tool's performance should be task-dependent and not data-dependent. In this Section, we have validated both the need as well as the efficacy of the GCTI-SN



(a)



(b)

Figure 2.15: Quantitative comparison on (a) SN-AM (ALL) dataset (b) SN-AM (MM) dataset with original images and images obtained with: Histogram method (Jain, 1989), Color transform (Reinhard *et al.*, 2001), SVD based color deconvolution method (Macenko *et al.*, 2009), NMF based color deconvolution method (Li and Plataniotis, 2015), GCTI-SN (proposed) based on SVD color deconvolution, and GCTI-SN (proposed) based on NMF based color deconvolution.

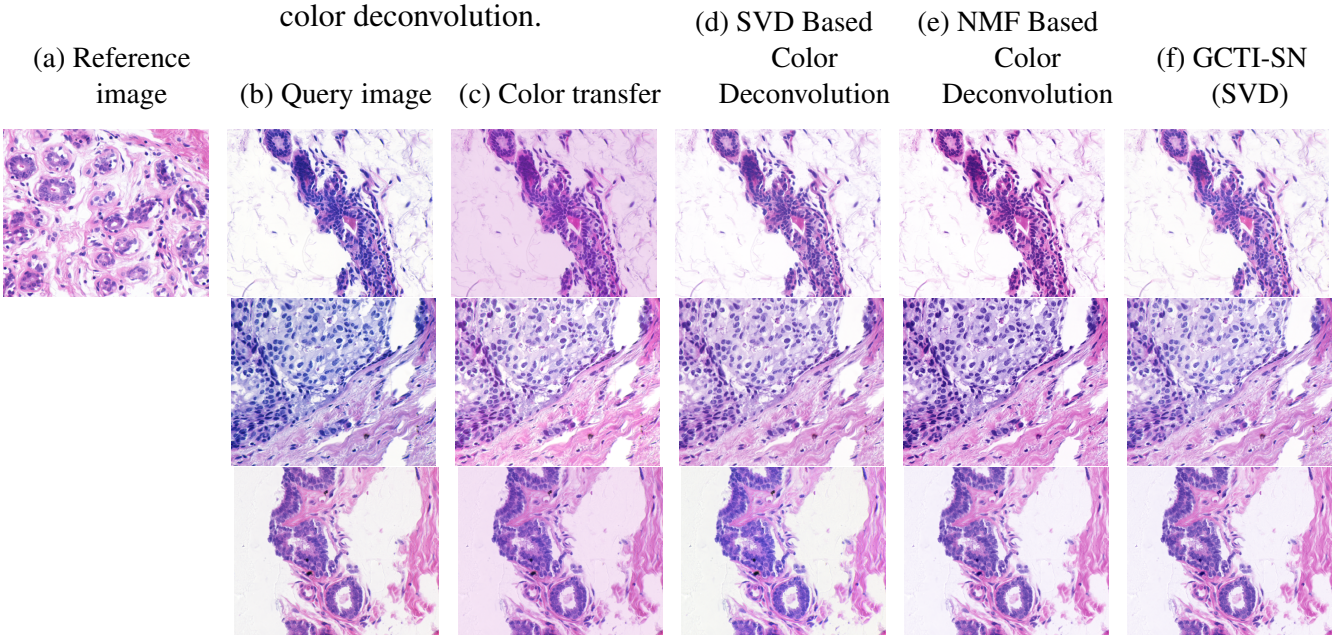


Figure 2.16: Stain normalization on three sample images (top to down) of BRCA-08 dataset with (c) color transfer (d) SVD based color denovolution (e) NMF based color denovolution, and (f) proposed GCTI-SN method.

method on a diagnostic tool developed for the task of breast cancer diagnosis. The tool must perform consistently irrespective of the dataset origin.

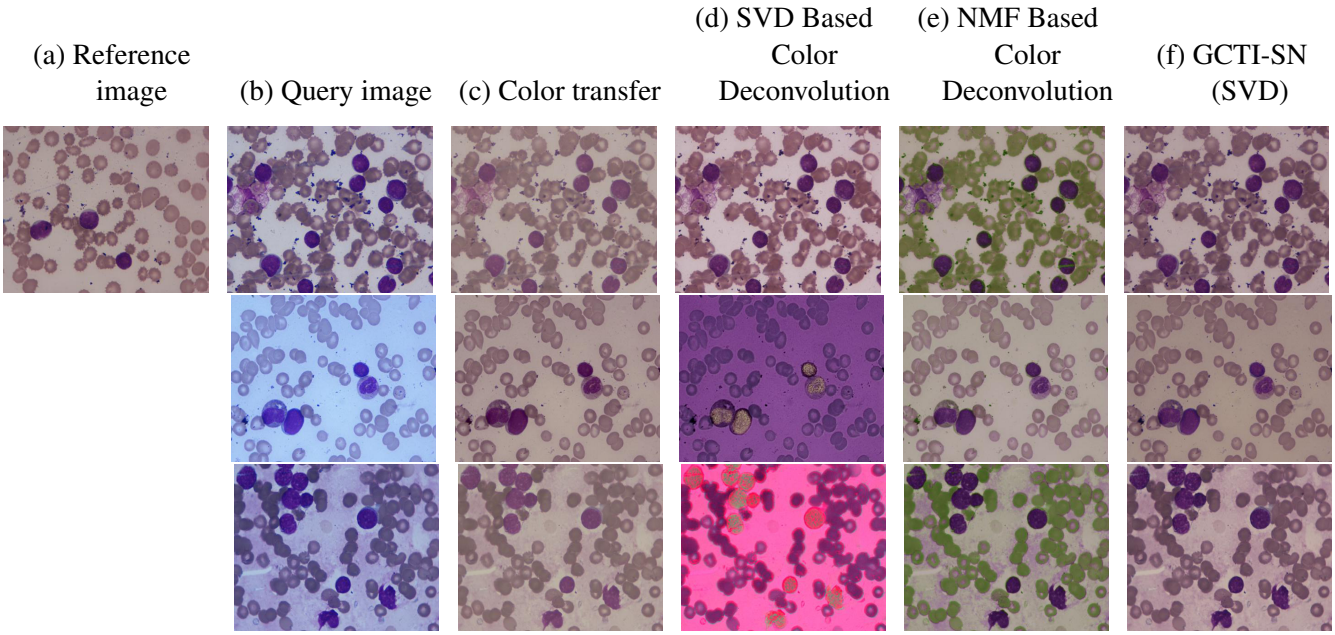


Figure 2.17: Stain normalization on three sample images (top to down) of SN-AM (ALL) dataset with (c) color transfer (d) SVD color deconvolution (e) NMF based deconvolution, and (f) proposed GCTI-SN method.

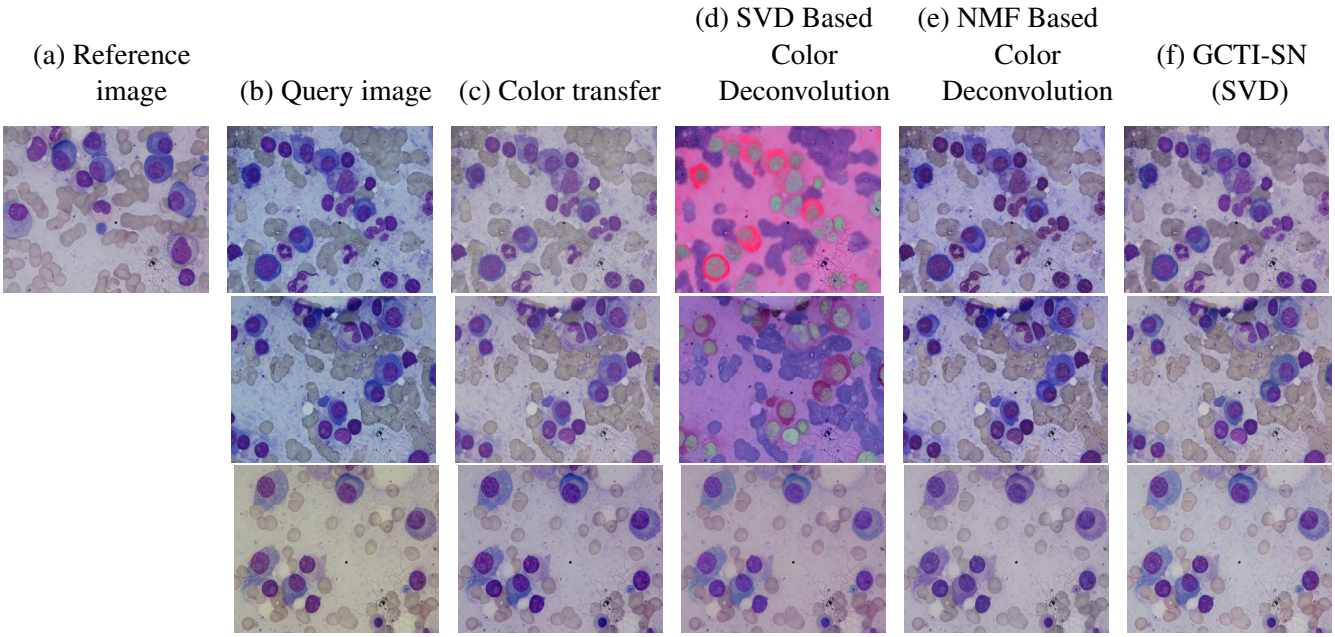
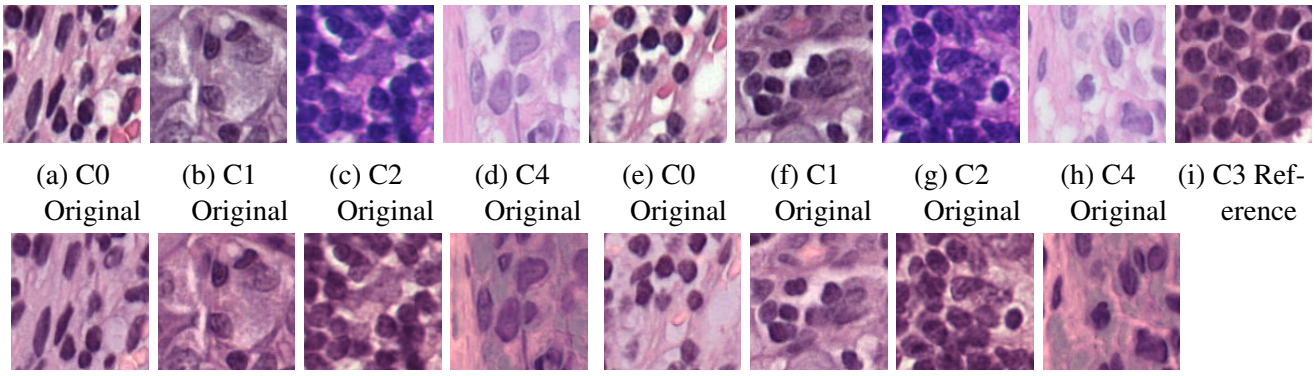


Figure 2.18: Stain normalization on three sample images (top to down) of SN-AM (MM) dataset with (c) color transfer (d) SVD based color denovolution (e) NMF based color denovolution, and (f) proposed GCTI-SN method.

2.5.1 Dataset

We have used a publicly available Camelyon17 Challenge dataset (Bandi *et al.*, 2019) that is a collection of whole slide images (WSIs) from five centers with ten annotated images/center, amounting to a total of 50 annotated images. The images are obtained from H&E stained lymph node slides prepared from the samples of breast cancer pa-

tients. For this dataset, the diagnostic task is to classify each patch as healthy or cancer, where patches with metastatic tumor cells are labeled as cancer patches. The five centers are named as Center-0 (C0) to Center-4 (C4). For training and testing, patches of size 128×128 are extracted from the WSIs of each center. Approximately, 130k (130,000) patches are extracted from the whole slide images of C3, which are subdivided as training (103k), validation (13k), and test set (14k). This data of center C3 is used for training the classifier. Similarly, a total of 60K, 35k, 59k, and 21k patches are extracted from the whole slide images of C0, C1, C2, and C4 centers, respectively. These images are used as a test set. Hence, the classifier is trained on the images of C3 but tested on the images of C0, C1, C2, and C4. We utilized GCTI-SN as the stain normalization method before feeding the images to the classifier, where the reference image is chosen from C3. We built the classifier with and without stain normalization. Fig. 2.19 shows two sample images (original) from every center, along with the corresponding normalized images.



(a) C0 Original (b) C1 Original (c) C2 Original (d) C4 Original (e) C0 Original (f) C1 Original (g) C2 Original (h) C4 Original (i) C3 Reference

(j) C0 Normalized (k) C1 Normalized (l) C2 Normalized (m) C4 Normalized (n) C0 Normalized (o) C1 Normalized (p) C2 Normalized (q) C4 Normalized

Figure 2.19: Sample original and normalized images from C0, C1, C2 and C4. The normalization is achieved using the reference image from C3.

2.5.2 Augmented Dataset

There can be two approaches to handle stain color variations while designing a diagnostic tool, say a classifier on this dataset. The first approach to build a robust diagnostic tool, whose performance is not impacted adversely by stain color variation in the dataset, is by pre-processing the entire training and test dataset via stain normalization. The second approach is by training the classifier in such a fashion that it is invariant to stain color variations. This latter approach requires training the classifier using an

augmented training dataset by generating more images from the training set images. Here, some transformations are applied to the training set images to create new images such that different images exhibit heterogeneous color distribution. The underlying philosophy is that the classifier itself would learn to discard the stain color variations and would learn the class label for classification. Recently, Tellez *et al.* (2019) has shown experiments with and without stain normalization, as well as with and without color augmentations. The augmentations and the set of experiments in this Section are inspired by the work of Tellez *et al.* (2019). We have employed two sets of augmentations:

1. **Brightness-Affine-Noise (BAN):** In this set of augmentation, brightness and contrast are varied uniformly in the range [.65, 1.35] and [.5, 1.5], respectively. Random rotations in the range [0, 360], arbitrary scaling in the range [.8, 1.2], and random shear in the range [-10,10] have also been applied. Also, Gaussian noise with a mean of 0.1 and a standard deviation of 0.2 is applied randomly.

2. **Brightness-Affine-Noise-Saturation (BANS):** Besides the previous set of augmentations, we have also included saturation and hue in this set of augmentations. The saturation and hue are applied uniformly in the range [0,2] and [-0.5, 0.5], respectively. The sample images produced using these augmentations are shown in Fig. 2.20.

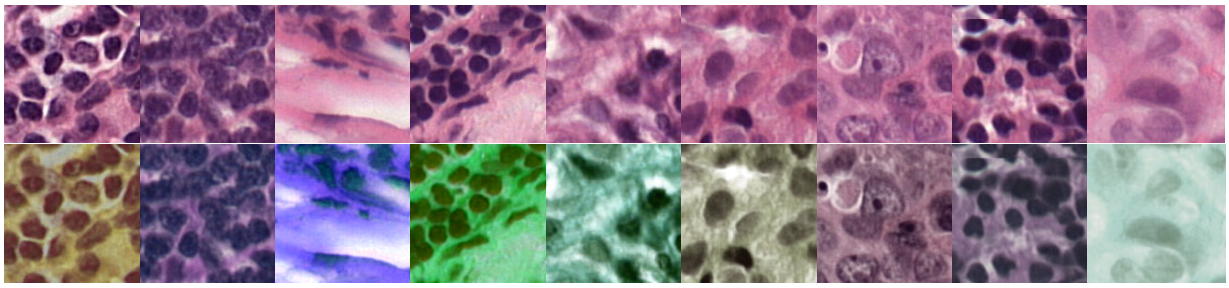


Figure 2.20: Sample original images of C3 (row 1) and corresponding BANS based color augmented images (row 2). For visualization purpose, affine transformations are not introduced in these images.

2.5.3 Experiments

A rigorous analysis has been carried out to explore the impact of stain normalization and color augmentations. We carried out six experiments, listed as below:

(E_1) The classifier is trained on the original (un-normalized) images of C3 and tested on original (un-normalized) images of C0, C1, C2, and C4.

(E_2) The classifier is trained on the original (un-normalized) images, and BAN augmented images of C3. It is tested on original (un-normalized) images of C0, C1, C2, and C4.

(E_3) The classifier is trained on the original (un-normalized) images, and BANS augmented images of C3. It is tested on original (un-normalized) images of C0, C1, C2, and C4.

(E_4) The classifier is trained on the GCTI-SN normalized images of C3 and tested on GCTI-SN normalized images of C0, C1, C2, and C4.

(E_5) The classifier is trained on the GCTI-SN normalized images and BAN augmented images of C3. It is tested on GCTI-SN normalized images of C0, C1, C2, and C4 dataset.

(E_6) The classifier is trained on the GCTI-SN normalized images and BANS augmented images of C3. It is tested on the GCTI-SN normalized images of C0, C1, C2, and C4.

A pretrained SqueezeNet (Iandola *et al.*, 2016), fine-tuned on the images of C3, was used as the classifier to conduct the above experiments. SqueezeNet is a relatively smaller and faster CNN architecture compared to its counterparts: VGGNet, DenseNet, and AlexNet. This architecture was chosen due to its small size and high accuracy on the classification tasks, thus allowing us to train multiple classification models and perform various experiments with limited computational requirements and time. The network was initialized with weights of squeezeNet pretrained on the ImageNet dataset. The last layer was modified to accommodate the number of classes (two classes) of our dataset. Since the input spatial size of the pretrained architecture is 224×224, the input images were resized accordingly. The stochastic gradient descent (SGD) optimizer, with an initial learning rate of 1.10^{-3} and a batch size of 64, was used to minimize the binary cross-entropy loss to train the classifier. The learning rate was decreased by a factor of 0.1 at the 50th and the 70th epoch. The training was stopped at the 80th epoch, which was chosen experimentally.

Results are shown in Table 2.8. The performance is shown in terms of accuracy, precision, sensitivity, and specificity. In addition, the area under the curve (AUC) of receiver operating characteristics (ROC) of the classifier that is a plot of true positive rate versus false-positive rate, was also used as the performance metric. In Table 2.8, bold case numbers represent the highest metric value obtained for the given test dataset among all six experiments. For example, it is noted that best AUC, accuracy, and sensitivity on the dataset of center C0 are obtained with experiment E_6, whereas the best precision and specificity are obtained with experiment E_5. For experiment E_1, the classifier performed worst on the images of centers C2 and C4 in terms of the majority of the metrics. The accuracy is only 28% on the images of C2 and 46% on the images of C4. However, with the introduction of color augmentations– BAN and BANS (E_2 and E_3), the performance improved significantly, although BAN still failed on the images of C2 with only 18.6% accuracy. On classifier trained with BANS augmented images (experiment E_3), accuracy ranged from 75% to 84% on the test dataset. On the other hand, classifiers trained with the GCTI-SN normalized dataset (E_4) shows either the best or comparable to the best performance on the images of almost all the centers. Accuracy with normalized images ranged from 78.9% to 86.8% on the four test centers' images. GCTI-SN normalization combined with the BANS augmentation (E_6) shows better performance on some of the metrics of one or two centers. Thus, classifiers trained with either the stain normalized images (E_4) or trained with stain normalized and BANS color augmented images (E_6) performed better. Additional experiments were performed after introducing blur in BAN and BANS. Blur was obtained with a Gaussian filter of zero mean and a uniform standard deviation between 7 and 8. However, overall results with this set of augmentations were inferior to those with BAN and BANS (Table-2.8), and are summarized in A.1.

On comparing the performance of the classifier trained with color augmentation (experiments E2 and E3) with the classifier trained with un-normalized images (experiment E1), it is observed that former performs better. This shows that compared to a classifier trained on un-normalized images, color augmentations, to some extent, help in improving the quality of the diagnostic tool by making the classifier learn some of the variations of color. Among the augmentations of BAN and BANS, the latter performs better because it introduces a significant difference in the color distribution of images. At the same time, on comparing the performance of classifier trained with

Table 2.8: Performance comparison of the classifier trained using all possible six scenarios listed as experiments E_1 to E_6, with the original, normalized, color augmented, and normalized with color augmented images of C3. Testing results are compiled on the images of centers C0, C1, C2, and C4. The bold case numbers represent the highest metric value obtained for the given test dataset among all six experiments.

E_1: Trained with Original Dataset					
Test Dataset	AUC	Accuracy	Precision	Sensitivity	Specificity
Center 0	0.754	0.670	0.769	0.480	0.858
Center 1	0.873	0.758	0.816	0.651	0.860
Center 2	0.362	0.281	0.160	0.003	0.960
Center 4	0.787	0.460	0.256	0.042	0.877
E_2: Trained with Augmented Dataset (BAN)					
Test Dataset	AUC	Accuracy	Precision	Sensitivity	Specificity
Center 0	0.918	0.828	0.856	0.787	0.870
Center 1	0.876	0.747	0.719	**0.789**	0.706
Center 2	0.508	0.186	0.183	0.043	0.537
Center 4	0.803	0.668	0.698	0.590	0.745
E_3: Trained with Augmented Dataset (BANS)					
Test Dataset	AUC	Accuracy	Precision	Sensitivity	Specificity
Center 0	0.926	0.829	0.878	0.761	0.896
Center 1	**0.916**	**0.816**	0.893	0.708	0.918
Center 2	0.886	0.750	**0.984**	0.658	**0.973**
Center 4	0.913	0.841	0.898	0.770	0.913

E_4: Trained with Normalized Dataset				
AUC	Accuracy	Precision	Sensitivity	Specificity
0.927	0.836	0.879	0.777	0.895
0.90	0.789	**0.921**	0.622	**0.949**
0.914	0.812	0.854	**0.885**	0.631
0.937	0.868	0.858	0.882	0.854
E_5: Trained with Augmented (BAN) + Normalized Dataset				
AUC	Accuracy	Precision	Sensitivity	Specificity
0.938	0.862	**0.891**	0.822	**0.902**
0.895	0.801	0.902	0.666	0.931
0.895	0.784	0.838	0.863	0.592
0.927	0.840	0.810	**0.888**	0.792
E_6: Trained with Augmented (BANS)+ Normalized Dataset				
AUC	Accuracy	Precision	Sensitivity	Specificity
0.945	**0.867**	0.882	**0.844**	0.889
0.902	0.800	0.894	0.668	0.925
0.916	**0.833**	0.891	0.872	0.739
0.927	**0.870**	**0.913**	0.818	**0.923**

color augmentation (experiments E2 and E3) and that trained with stain color normalization (experiment E4), the latter performs better. This shows that color augmentation cannot eliminate the necessity of the need for stain color normalization. On comparing the classifiers trained on BAN augmentation plus stain color normalization to only stain color normalization, results are encouraging with both. Since the only augmentation performs inferior to normalization, it is evident that the proposed GCTI-SN stain normalization method is robust. Further, the performance of DL based stain normalization methods that utilize augmentations is sensitive to the choice of parameters (hue, saturation, etc.) used to generate the augmented images. On the other hand, the performance of GCTI-SN is independent of any such parameter and hence, is more robust (A.1). But indeed, color augmentation supports stain color normalization when used together for building the diagnostic tool.

2.6 Conclusion

From the application point of view, the significance of the proposed GCTI-SN is demonstrated diagnostically in the application of breast cancer detection on a publicly available dataset collected from five different centers/hospitals. Results of this section an-

swer three important questions– Is stain normalization indeed required for building a computer-assisted automated diagnostic tool? What is the performance comparison of stain normalization vis-à-vis color augmentation? Would color augmentation in the classifier design suffice in place of stain color normalization? From Table-2.8, we note that the diagnostic tool fails when only un-normalized images are used as training and testing (Experiment E_1). In contrast, the classifier trained with normalized images consistently has good performance with as high as 94.5% AUC (Experiment E_6). This answers the first question that stain normalization helps build a robust diagnostic tool by mitigating the adverse impact of stain color variations.

CHAPTER 3

EDNFC-Net: Convolutional Neural Network with Nested Feature Concatenation for Nuclei-Instance Segmentation

3.1 Introduction

Nuclei detection, which aids in cell detection, is a starting point in the diagnosis of diseases like cancer. The quick and accurate identification of nuclei under varied conditions can expedite the diagnosis that is necessary for a better survival outcome. Nuclei detection/segmentation is also the preceding step in computer-aided diagnosis, where the aim might be to classify the nuclei as healthy or cancerous. Such automatic tools should not only be resource optimized but also be easily deployable. From all these perspectives, accurate nuclei segmentation is a crucial stage.

Researchers are working on the development of automatic segmentation tools because manual segmentation, apart from being time-intensive, is also vulnerable to human errors. Initial automated segmentation methods were based on clustering, thresholding, and region growing (Meijering, 2012). However, these methods have limited capability in handling complex structures like color/texture variations or clustered cells. Machine learning-based approaches utilizing hand-crafted features have also been applied to this task, but their performance is limited due to low-level features. Deep learning-based methods perform better than the traditional machine learning-based methods as the former learn the task-specific features from the data itself (Caicedo *et al.*, 2019).

The U-Net architecture (O.Ronneberger *et al.*, 2015) is a convolutional neural network (CNN) based encoder-decoder type architecture that has been successfully used in cell segmentation. The architecture, unlike the fully convolutional network (FCN) (Long *et al.*, 2015), involves the concatenation of feature maps from the encoder layers to decoder layers, which helps in context information preservation and generation of

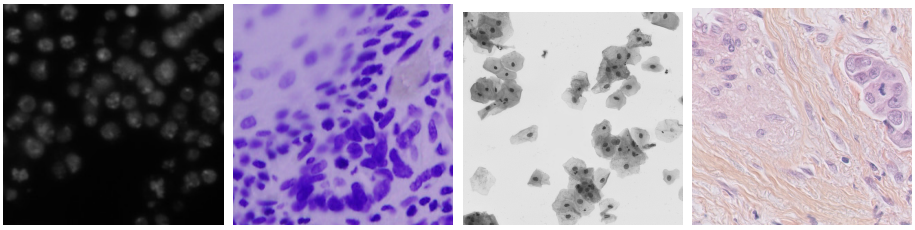
smoother segmentation masks. The U-Net, however, finds the segregation of the clustered nuclei a challenging task. This problem is solved in Chen *et al.* (2017) through the additional prediction of the contour of the region of interest and in Kumar *et al.* (2017), through the formulation of segmentation as a three-class classification, i.e., contour, foreground, and background.

Some approaches use bounding box proposal networks to perform nuclei-instance segmentation (Akram *et al.*, 2016; Johnson, 2018). These networks perform better in separating the overlapped cells but have relatively poor performance in the generation of accurate segmented masks. There have also been to attempts the model segmentation as a regression problem rather than the classification problem (Naylor *et al.*, 2019). Some approaches combine different methodologies to obtain a more robust pipeline. For example, authors in Naylor *et al.* (2017); Wang *et al.* (2019) have utilized CNN in the first step to obtain the approximate segmented masks followed by morphological operations (like watershed) to refine the obtained masks. Similarly, authors in Vuola *et al.* (2019); Xu and et al. (2019) have combined encoder-decoder type architecture with detection networks for better segmentation performance.

We propose a single encoder-decoder based architecture, namely, EDNFC-Net that is shown to have robust performance in segmentation mask generation and cluster segregation with datasets of different modalities. The inter-FCC (Section 3.3) connections are introduced inspired from Huang *et al.* (2017) which is a classification network. However, for segmentation, such connections are non-trivial due to successive downsampling and upsampling in encoder and decoder, respectively, leading to spatial size discrepancy between the different features. For example, in the encoder, the following FCC has feature maps of half spatial size than the previous FCC. To this end, we propose modules that utilize downsampling/upsampling and 1×1 convolution filters for size matching and limiting the features. In contrast to Zhou *et al.* (2018), that redesigned skip connections, the proposed EDNFC-Net is much simpler as the features concatenation is only at the encoder and decoder level. Also, in Zhou *et al.* (2018), convolution blocks for skip connections increase with the depth of the encoder. However, the skip connections in the EDNFC-Net are the same as the original U-Net (Ronneberger *et al.*, 2015*b*), requiring no additional parameters, and only inter-encoder/decoder feature flow is modified. A new loss function is also proposed that induces better pixel classification capability.

Table 3.1: Dataset description. The training and testing set is obtained by extracting the patches of a given size from the images.

Dataset	Total Images	Patch Size	Training	Test
DSBC18 DSB (2019)	660	128×128	10000	2503
BC19 Naylor *et al.* (2019)	50	128×128	2080	297



(a) (b) (c) (d)

Figure 3.1: Some samples from the dataset used. (a) DSBC18 (Flourescence) DSB (2019) (b) DSBC18 (Histology) DSB (2019) (c) DSBC18 (Brightfield) DSB (2019) (d) BC19 Naylor *et al.* (2019)

3.2 Material

Two datasets are used to evaluate the performance of the EDNFC-Net. The first dataset is the 2018 Data Science Bowl challenge (DSBC18) dataset, which has images of three different modalities: i) histology, ii) fluorescence, and iii) brightfield DSB (2019). The second dataset consists of 50 Histology images of breast cancer (BC19) (Naylor *et al.*, 2019). We have also performed experiments by mixing these two datasets, which leads to a more complex dataset. The sample images from these datasets are shown in Fig-3.1. A detailed description of training and test sets is presented in Table-3.1. The 10% of the total images are used for testing, while the training is performed using small-sized patches (refer Table-3.1) of the remaining 90% images, with a 10% subsection used as the validation set.

3.3 Methods

The EDNFC-Net Architecture is designed by arranging the Feature Concatenation Cells (FCCs) in an encoder-decoder type architecture as shown in Fig.3.2.

Feature Concatenation Cell (FCC): The focus of the FCC is to facilitate the learning of better discriminative features and to induce efficient feature flow. The FCC is composed of two stacks of $k \times 3 \times 3$ convolutional filters, where k represents the number of filters. The dropout and batch normalization is introduced after each convolutional layer. The batch normalization, apart from assisting in faster convergence of training,

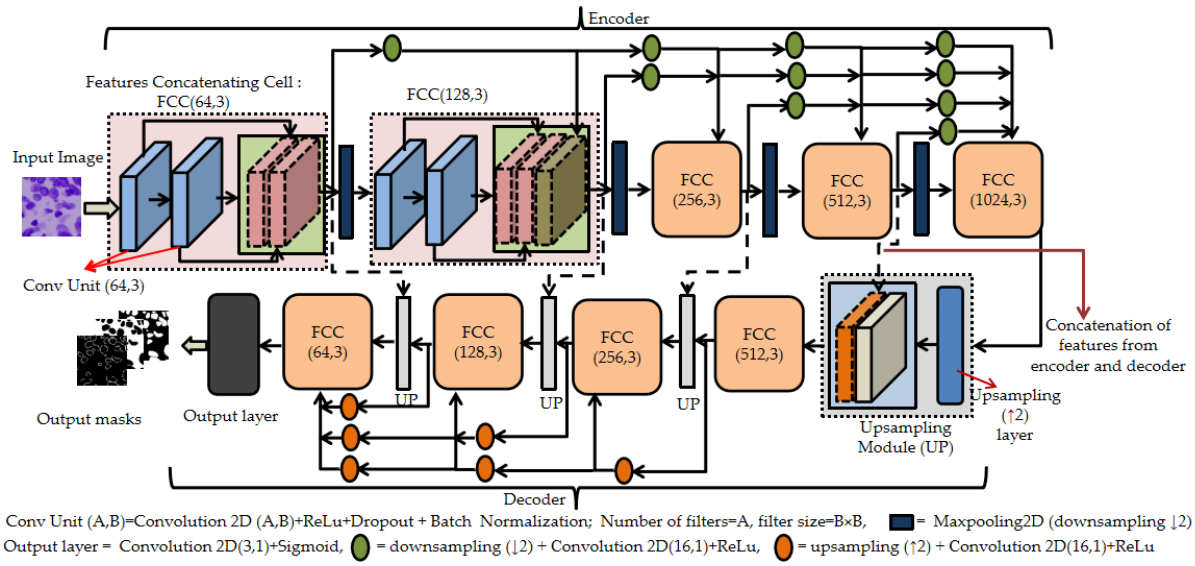


Figure 3.2: EDNFC-Net: Encoder-Decoder based Convolutional Neural Network (CNN) with Nested Feature Concatenation

is also helpful to counter the color variation effects of the input images, which are prevalent due to staining in the microscopic images. Dropout is introduced to check overfitting due to a lack of required training data. Finally, the concatenation of the outputs of both convolutional layers is passed to the next FCC. Such feature concatenation leads to feature reusability and also helps in controlling the vanishing gradient problem through the introduction of the shorter connections Huang *et al.* (2017).

Encoder Module: Encoder module is composed of five FCCs, and each FCC is followed by a downsampling layer, except for the last one. The number of filters consistently increases in a multiple of two, with first FCC and last FCC having 64 and 1024 filters, respectively. This set up leads to a progressive reduction in the feature map size but an increase in the number of feature maps, which helps to capture the intricate structure in the data.

Decoder Module: The decoder module is composed of four FCCs, four upsampling modules, and one output layer. The upsampling module performs upsampling by two on the input features and concatenates the same size feature maps of the encoder module with obtained upsampled feature maps. The direct connections between encoder and decoder are helpful in context information preservation. Upsampling modules progressively increase the feature maps size, which is finally the same as the input image size. The final FCC is followed by an output layer having $c \times 1 \times 1$ filters, with c representing the number of classes. In our experiments, we predict background, and edges along with the foreground, and hence, c is set to three.

Inter-FCC Features Concatenation: Apart from the intra-FCC features concatenation, the network also uses inter-FCC features concatenation. The features of an FCC in the encoder/decoder module are provided to all the following FCCs in the respective modules. However, there is a spatial size discrepancy between the features of the different FCCs, as the following FCC has feature maps of half spatial size than the previous FCC. To resolve this, the features of the FCCs are passed through suitable downsampling followed by convolutional layer with size $16 \times 1 \times 1$, as shown in Fig.3.2. This convolutional layer limits the number of feature maps being passed directly to the following FCCs. This concatenation mechanism leads to a direct flow of features within the whole encoder/decoder module leading to improved segmentation performance. Mathematically, at k^{th} FCC, the inter-FCC input is given as:

$$\mathcal{F}^k = \mathcal{C}\left[\mathcal{G}\left(\mathcal{F}^i\right)\right]_{i=1}^{k-1} \qquad \text{for } k \geq 2, \tag{3.1}$$

where $\mathcal{C}(\cdot)$ represents the recursive features concatenation, and $\mathcal{G}(\cdot)$ matches the features size for the concatenation, apart from squeezing the features to 16. Accordingly, for encoder, $\mathcal{G}(\cdot)$ consists of downsampling, sixteen 1×1 convolutional filters, and ReLU activation. For decoder, the upsampling replaces downsampling in $\mathcal{G}(\cdot)$.

3.3.1 Proposed Loss Function

Consider an input image **A**. We label the foreground with symbol **F**, background with symbol **B**, and edges with symbol **S**. For a given **A**, we predict $\hat{\mathbf{A}}$ with $\hat{\mathbf{F}}$, $\hat{\mathbf{B}}$ and $\hat{\mathbf{S}}$. Accordingly,

$$TP = |\mathbf{F} \cap \hat{\mathbf{F}}|$$
$$TN = |\mathbf{B} \cap \hat{\mathbf{B}}|$$
$$FP = |\mathbf{B} \cap \hat{\mathbf{F}}|$$
$$FN = |\mathbf{F} \cap \hat{\mathbf{B}}|$$
$$F = TP + FN = |\mathbf{F} \cap \hat{\mathbf{F}}| + |\mathbf{F} \cap \hat{\mathbf{B}}|$$

The weighted sum of Dice Loss and binary cross-entropy (BCE) loss is a commonly used loss function in the segmentation problem. Inspired from the metric introduced in Cappabianco *et al.* (2019) we replace the Dice Loss with the following balanced-loss

($\mathcal{L}_{BL}$) term:

$$\mathcal{L}_{BL} = 1 - \left(1 - \frac{|\mathbf{B} \cap \hat{\mathbf{F}}|}{|\mathbf{F} \cap \hat{\mathbf{F}}| + |\mathbf{F} \cap \hat{\mathbf{B}}|}\right)\left(1 - \frac{|\mathbf{F} \cap \hat{\mathbf{B}}|}{|\mathbf{F} \cap \hat{\mathbf{F}}| + |\mathbf{F} \cap \hat{\mathbf{B}}|}\right) \tag{3.2}$$

The last two terms consists of the ratio of $|\mathbf{B} \cap \hat{\mathbf{F}}|$ (false positive) and $|\mathbf{F} \cap \hat{\mathbf{B}}|$ (false negative) with the foreground pixels ($|\mathbf{F} \cap \hat{\mathbf{F}}| + |\mathbf{F} \cap \hat{\mathbf{B}}|$), respectively. The convergence of this loss function demands both of these ratios to be less than one, which in turns will force the minimization of both, false positives (FP) and false negatives (FN). Compared to Dice Loss given as,

$$\mathcal{L}_{Dice} = 1 - 2 * \frac{|\mathbf{A} \cap \hat{\mathbf{A}}| + \delta}{|\mathbf{A}| + |\hat{\mathbf{A}}| + \delta}, \tag{3.3}$$

where δ mitigates the effect of zero denominator, balanced loss introduces a stronger penalization as shown in Fig. 3.3. The curves are shown for the varying values of FP and FN. The balanced loss is always higher than Dice Loss for FP, FN $\in [0, 1]$, particularly in the mid-range.

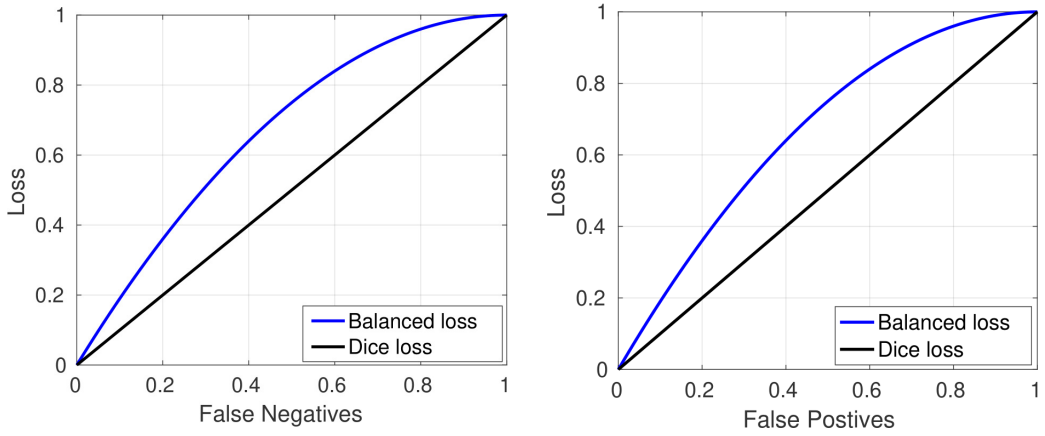


Figure 3.3: Variation of balanced loss and Dice Loss with false negatives (left) and false positives (right).

We use final loss function as a weighted combination of BCE and balanced loss:

$$\mathcal{L} = w_1 \mathcal{L}_{BCE} + w_2 \mathcal{L}_{BL} \tag{3.4}$$

where $\mathcal{L}_{BCE} = BCE(\mathbf{A}, \hat{\mathbf{A}})$. Due to a stronger penalization, this loss function induces better separation of background and foreground.

3.3.2 Training and Testing

We have used Keras library for the implementation and Adam D. P. Kingma (2014) optimizer with batch size of 10 to perform the training using NVIDIA RTX2080 GPU.

The training process converged in 75 epochs with an initial learning rate of 0.0001 that is cumulatively dropped by a factor of 10 after 60 (learning rate=10^{-5}) and 70 (learning rate=10^{-6}) epochs, respectively. For the weighted combination in the loss functions, the value of w_1 and w_2 is set to 0.6 and 0.4 respectively.

3.4 Results and Discussion

To evaluate the quantitative performance we have used dice similarity score (DSC) or F_1 score and intersection over union (IoU) or Jaccard similarity index. DSC and IoU are the frequently used evaluation metrics for medical image segmentation Bertels *et al.* (2019). For an image $\mathbf{I}$ with ground truth segmentation mask $\mathbf{F}$, and predicted segmentation mask $\hat{\mathbf{F}}$, the DSC and IoU are given as:

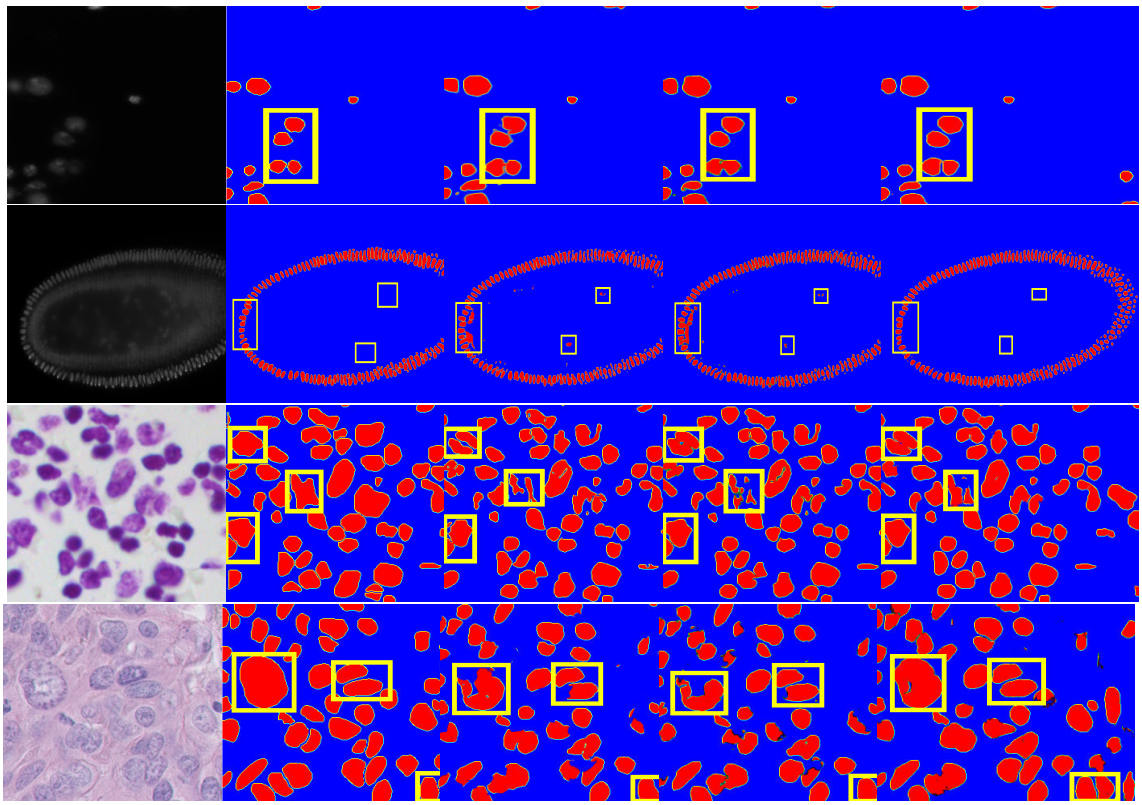
$$\text{DSC} = 2 * \frac{|\mathbf{F} \cap \hat{\mathbf{F}}|}{|\mathbf{F}| + |\hat{\mathbf{F}}|}, \text{ and IoU} = \frac{|\mathbf{F} \cap \hat{\mathbf{F}}|}{|\mathbf{F} \cup \hat{\mathbf{F}}|} \tag{3.5}$$

Both of these metrics have values in the range $[0, 1]$ with higher value denoting the more similarity between prediction and ground truth. Also, the better performing model will have higher values of the DSC and IoU.

Table 3.2: Comparison of U-Net, U-Net$_{\text{BL}}$ and EDNFC-Net on images of different modalities of DSBC18 DSB (2019), on BC19 Naylor *et al.* (2019) and on the mixture of DSBC18 and BC19

	Fluorescence (DSBC18)			Histology (DSBC18)			Brightfield (DSBC18)			BC19 Naylor *et al.* (2019)			Mixed (DSBC18 and BC19)		
	U-Net	U-Net$_{\text{BL}}$	EDNFC-Net	U-Net	U-Net$_{\text{BL}}$	EDNFC-Net	U-Net	U-Net$_{\text{BL}}$	EDNFC-Net	U-Net	U-Net$_{\text{BL}}$	EDNFC-Net	U-Net	U-Net$_{\text{BL}}$	EDNFC-Net
DSC	.901	.904	**.919**	.838	.845	**.858**	.850	.866	**.890**	.762	.742	**.766**	.858	.860	**.874**
IoU	.835	.839	**.858**	.736	.744	**.769**	.760	.782	**.816**	.640	.618	**.648**	.768	.772	**.792**

We have performed the qualitative and quantitative analysis to evaluate the performance of EDNFC-Net and the proposed loss function (Eq. 3.4). The performance of EDNFC-Net is compared with U-Net trained with a weighted sum of Dice Loss and BCE loss, and U-Net$_{\text{BL}}$, the U-Net trained using the proposed loss (Eq. 3.4). Table-3.2 shows the quantitative performance comparison in terms of DSC and IoU. Since the DSBC18 dataset consists of images of three different modalities, we have shown the performance for each modality separately. The EDNFC-Net is performing better for all three types of modalities. Maximum improvement is observed for brightfield modality with 4% and 5.5% gain in DSC and IoU, respectively. Similarly, U-Net$_{\text{BL}}$ is performing better than U-Net, which proves the hypothesis regarding the proposed loss function. Similar trends are also seen for the BC19 dataset with EDNFC-Net leading the



(a) Original (b) Ground Truth (c) U-Net (d) U-Net$_{BL}$ (e) EDNFC-Net

Figure 3.4: Rows 1 to 3: Results on DSBC18 dataset (DSB, 2019) and Row 4: Results on BC19 dataset (Naylor *et al.*, 2019). Results are shown with U-Net, U-Net$_{BL}$, and EDNFC-Net. Results of Row 4 on BC19 are shown on the model trained on the mixture of the dataset DSBC18 and BC19. Pixels in the yellow boxes highlight the performance difference between the different models. These highlighted pixels contrast the difference in cluster segregation and segmentation mask generation with different architectures.

remaining methods. However, U-Net$_{BL}$ is lagging with respect to U-Net in this case. To further evaluate the performance, we mixed the images of both the datasets (DSBC18 and BC19), to form a more challenging dataset. For this dataset also, EDNFC-Net is the leading method, and U-Net$_{BL}$ also has better performance than U-Net.

The qualitative results for all three methods are shown in Fig. 3.4. From the highlighted pixels in row 1, it is observed that EDNFC-Net has better performance in cluster segregation. Similarly, from row 2-row 4, it can be seen that EDNFC-Net is more immune to false positives and false negatives as compared to the other two methods. Also, it can be inferred from Fig. 3.4 that U-Net$_{BL}$ shows better qualitative results with respect to U-Net, which again proves the contribution of the proposed loss function.

3.5 Conclusion

In this work, we have addressed the problem of nuclei-instance segmentation problem on two cancer imaging dataset. We have made two contributions: 1) we have proposed a new EDNFC-Net architecture with intra and multiscale inter-feature concatenation in encoder and decode, 2) EDNFC-Net is trained with a new loss function consisting of a weighted combination of BCE and proposed balanced-loss. On both the dataset, the proposed architecture trained with the proposed performs much better compared to a) U-Net trained with weighted combination of BCE and Dice Loss, and b) U-Net trained with the proposed loss function.

CHAPTER 4

SDCT-AuxNet$^{\theta}$: DCT Augmented Stain Deconvolutional CNN with Auxiliary Classifier for Cancer Diagnosis

Acute lymphoblastic leukemia (ALL) is a pervasive pediatric white blood cell cancer across the globe. With the popularity of convolutional neural networks (CNNs), computer-aided diagnosis of cancer has attracted considerable attention. Such tools are easily deployable and are cost-effective. Hence, these can enable extensive coverage of cancer diagnostic facilities. However, the development of such a tool for ALL cancer was challenging so far due to the non-availability of a large training dataset. The visual similarity between the malignant and normal cells adds to the complexity of the problem. This work discusses the recent release of a large dataset and presents a novel deep learning architecture for the classification of cell images of ALL cancer. The proposed architecture, namely, *SDCT-AuxNet*$^{\theta}$ is a 2-module framework that utilizes a compact CNN as the main classifier in one module and a Kernel SVM as the auxiliary classifier in the other one. While CNN classifier uses features through bilinear-pooling, spectral-averaged features are used by the auxiliary classifier. Further, this CNN is trained on the stain deconvolved quantity images in the optical density domain instead of the conventional RGB images. A novel test strategy is proposed that exploits both the classifiers for decision making using the confidence scores of their predicted class labels. Elaborate experiments have been carried out on our recently released public dataset of 15114 images of ALL cancer and healthy cells to establish the validity of the proposed methodology that is also robust to subject-level variability.

4.1 Introduction

Cancer is a condition of unrestricted cell growth in the human body and is among the deadliest diseases of the world. In 2018, cancer claimed 9.6 million lives globally, 70% of which are from low and middle-income countries (LMIC) (World Health Organization, 2019). Of the different types of cancer, acute lymphoblastic leukemia (ALL)

is pervasive childhood cancer that occurs due to the excessive generation of immature white blood cell (WBC) blasts in the bone marrow. Although prompt diagnosis is a critical factor in cancer survival, LMIC suffer on this account due to the lack of costly infrastructure, trained human resources, and diagnostic tools at the required scale. Hence, these countries experience higher fatality rates. For example, 84% of cases of ALL are reported from LMIC (Asthana *et al.*, 2018). With the popularity of convolutional neural networks (CNNs), computer-aided diagnosis of cancer has attracted considerable attention. Such tools are easily deployable and are cost-effective. Hence, these can enable extensive coverage of cancer diagnostic facilities.

Preliminary diagnostic tests of ALL are based on the count and appearance of blood cells. For example, *complete blood count* (CBC) makes a diagnosis based on cell count, while *peripheral blood smear* test examines the appearance of cells in the blood. Bone marrow based tests such as *bone marrow biopsy* and *bone marrow aspiration* are crucial in ALL diagnosis and analyze bone marrow to evaluate the symptoms of leukemia. Higher accuracy tests for the identification of leukemia include *cytochemistry*, *immunohistochemistry*, and *flow cytometry* that are based on the reaction of staining chemicals with the proteins of blood cells. Imaging tests such as X-rays, computed tomography (CT) scan, magnetic resonance imaging (MRI) scan, and ultrasound are not useful for ALL detection, but they help with the analysis of the overall impact of cancer on the body (American Cancer Society, 2019).

All these methods are time-intensive, require costly medical equipment, and trained medical professionals. Owing to these requirements, these tools cannot be deployed easily at a large scale, especially in rural areas. Computer-aided tools based on microscopic image analysis can overcome these limitations because these can be fully automated and do not require highly trained medical professionals to run the tests.

In this work, a novel deep learning architecture is proposed to differentiate malignant (cancer) immature WBCs from normal (healthy) immature WBCs for computer-aided diagnosis of ALL. Instead of using transfer learning with a CNN architecture pre-trained on natural images, a CNN has been trained from scratch, as used by Bayramoglu *et al.* (2016); Dhungel *et al.* (2017); Gao *et al.* (2017*b*); Zhao *et al.* (2017); Duggal *et al.* (2017). Because cell image features may differ widely from those of the natural images, a network trained from scratch on cell images may converge to a better

solution . Although C-NMC 2019 dataset is quite large compared to the existing ALL datasets, still larger dataset would be required for training a very deep CNN architecture. Hence, to curb this limitation, a compact CNN with fewer parameters is utilized. We have also exploited discrete cosine transform (DCT) to improve the classification performance. DCT finds use in numerous image processing applications including JPEG compression (Raid *et al.*, 2014), brain tumor classification (Sridhar and Murali Krishna, 2013), and face recognition (Zhengjun Pan *et al.*, 2000). We have used DCT to capture inter-class separability in the frequency domain because images of different classes in this dataset are not visually differentiable in the spatial domain, as can be verified from Fig. 4.1.

Overall, the salient points of this work are as follows:

1. A novel deep learning architecture is proposed for the classification of cell images of ALL. The proposed architecture, namely, SDCT-AuxNet$^{\theta}$, is a 2-module framework and is end-to-end trainable. It consists of a compact CNN as the main classifier in one module and a Kernel SVM as the auxiliary classifier in the other one.

2. The input RGB images are first transformed to the OD space, and then stain color is deconvolved using a trainable stain deconvolutional layer (SD-Layer) appended at the front-end of the CNN to obtain stain quantity images. The texture of stain quantity images captures the sub-microscopic cellular features. Hence, a compact CNN trained on stain-deconvolved images in the OD space is envisaged to perform comparable or better than a deeper CNN trained on raw RGB images, overcoming the need of a big training dataset.

3. A DCT layer is used immediately after the SD-layer in the proposed CNN. The inspiration for training on the DCT features is derived from the fact that, conventionally, the spectroscopy-based method, namely flow cytometry, is used for ALL diagnosis.

4. While the CNN classifier utilizes features via bilinear pooling employed before the fully connected layer, averaging across feature maps (spectral averaging) is used to create features for the auxiliary classifier. Bilinear pooling (Lin *et al.*, 2015) helps in capturing fine-grained local feature interactions, whereas aver-

aging across feature maps on the same set of input helps in capturing spectral counterpart that is used in parallel by the auxiliary classifier in the proposed architecture. This strategy allows different features to be used by the two classifiers.

5. A novel test strategy is proposed that exploits both the classifiers for decision making using the confidence scores of their predicted class labels.

Training and testing are carried out on our publicly released dataset of ALL cancer of 118 subjects (Gupta and Gupta, 2019*b*). Benchmark in performance is achieved by comparing the results with the highest performance recorded on the leaderboard of the recently conducted challenge on this dataset at IEEE ISBI conference 2019 (Gupta *et al.*, 2019).

4.2 Materials

Microscopic images are captured from the bone marrow aspirate smears of subjects in the raw BMP format with a size of 2560×1920 pixels. Before imaging, smears are stained using Jenner-Giemsa stain for better visibility of B-type immature white blood cells (WBCs) under the microscope. The images are first normalized for stain color variability across images using the in-house pipeline (Gupta *et al.*, 2017). Next, cells of interest are marked by expert oncologists. These labeled cells are segmented using the in-house segmentation pipeline (Duggal *et al.*, 2016*b*,*a*). Normal cell images are collected from subjects who did not suffer from cancer, while cancer cell images are obtained from subjects who are initially diagnosed with cancer. A dataset of 118 subjects, 49 healthy and 69 cancer subjects, is prepared at Laboratory Oncology, AIIMS, New Delhi, India. A waiver for written informed consent is obtained from the Ethics Committee of AIIMS, New Delhi, on this dataset for research purposes. The subject identifying information is removed entirely from the image dataset before sharing the dataset.

This dataset is publicly available at The Cancer Imaging Archive (TCIA) (Gupta and Gupta, 2019*b*) and displayed according to TCIA standard protocols (Clark *et al.*, 2013*b*). This dataset was also used in the medical imaging challenge, *Classification of Normal vs. Malignant Cells in B-ALL White Blood Cancer Microscopic Images*

(C-NMC) 2019 (Gupta *et al.*, 2019), organized at IEEE International Symposium on Biomedical Imaging (ISBI), 2019 conference.

Data is prepared at the subject-level and is split into the training set and testing set. Training set consists of 8491 cancer cell images collected from 60 cancer subjects and 4037 normal cell images collected from 41 healthy subjects, with a total of 12528 images (Gupta *et al.*, 2019). This training data of 12528 cells is split into seven-folds for cross-validation, where the splits are created such that each fold contains an approximately equal proportion of images of both the classes. Also, the folds are prepared at the subject-level, i.e., all cells belonging to a subject are placed in the same fold. Test data

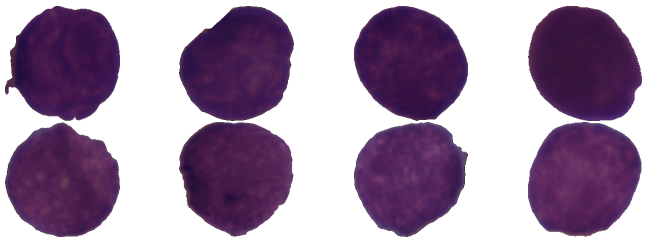


Figure 4.1: Sample images of different subjects from the cancer class (first row) and normal class (second row). This figure shows that the cell images are not distinguishable across classes or across subjects.

contains a total of 2586 cell images, collected from 9 cancer subjects and 8 healthy subjects (Gupta *et al.*, 2019). None of the cell images of these subjects are used for training the classifier and hence, results of the test data are unbiased. A detailed description of the training set and test set is provided in Table 4.1. Since original cell images are of different sizes, all images are zero-padded such that the centroid of every cell is at the center. After zero-padding, all cell images are made to the size of 350×350.

4.3 Methods

In this section, the proposed SDCT-AuxNet$^{\theta}$ architecture and other variants are presented in detail. We start with Base Network, a custom and parameter efficient architecture. Thereafter, modifications are proposed for designing SD-Net, SDCT-Net, and finally SDCT-AUxNet$^{\theta}$.

Table 4.1: Description of training and test data.
Training data is split in seven folds. All cells belonging to a subject are placed in the same fold and no two folds have cell images belonging to the same subject.

Training Data				
Fold	Cancer Class		Normal Class	
	No. of subjects	No. of images	No. of subjects	No. of images
1	13	1234	9	529
2	9	1166	7	546
3	9	1269	6	516
4	6	1275	4	624
5	9	1135	6	618
6	6	1197	4	603
7	8	1215	5	601
Total	60	8491	41	4037
Total no. of subjects		101	Total no. of images	12528
Test Data				
Cancer Subjects			Healthy Subjects	
9			8	
Total no. of subjects			17	
Total no. of cell images in the test data			2586	

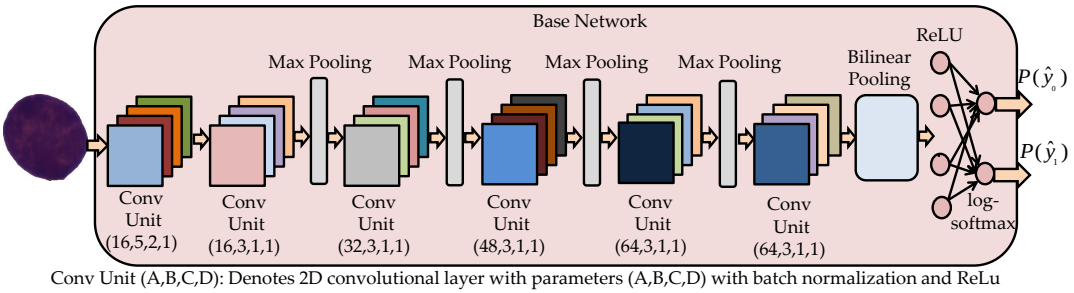
4.3.1 Base Network

The size of the input dataset is a crucial factor in designing a CNN because an architecture with large learnable parameters is prone to overfitting, while a CNN with too few learnable parameters tends to underfit. Hence, it is essential to design an architecture with an adequate number of tunable parameters. Moreover, transfer learning is not appropriate because all well-known architectures are trained on natural images, while this dataset specifically contains blood cell images. In one of our recent works, a compact CNN with fewer parameters was designed on this ALL dataset (Mourya *et al.*, 2018) as shown in Fig. 4.2. This architecture has a convolutional layer with stride two at the front-end followed by five convolutional layers with batch-norm, maxpool, and ReLu. Lin *et al.* (2015) introduced the idea of bilinear pooling, wherein the outer product of the feature vectors obtained through two different CNNs is connected to a common output layer. This idea is adopted in this base network, and the outer product of the feature vector with itself is connected to the classification layer. The output of bilinear pooling is passed through a linear layer followed by the log-softmax layer. This network is referred to as the *base network*. Here, *base network* is used as the building block in SD-Net architecture as well as in the following proposed architectures. A comparison of the number of learnable parameters of the *base network* with some well-known archi-

Table 4.2: Comparison on the number of learnable parameters.
The number of parameters in the *base network* is significantly less as compared to existing architectures used commonly in transfer learning approach.

Architecture	AlexNet	VGG16	ResNet-18	DenseNet-161	Inception-v3	Base Network
Parameters	61100840	138357544	11689512	28681000	27161264	**95362**

Note: Least number of parameters are specified in bold.



Conv Unit (A,B,C,D): Denotes 2D convolutional layer with parameters (A,B,C,D) with batch normalization and ReLu activation, where A denotes the number of filters, B denotes kernel size, C indicates size of stride and D denotes padding size

Figure 4.2: Base network. This network is used as the building block in SD-Net as well as in the proposed architectures.

tectures used in the transfer learning approach is shown in Table 4.2. It is evident that the *base Network* has fewer parameters compared to those in the existing architectures. Hence, base network is suitable for this dataset that is, although very large compared to existing public ALL datasets, is still relatively small compared to the ImageNet dataset on which often-used architectures are trained.

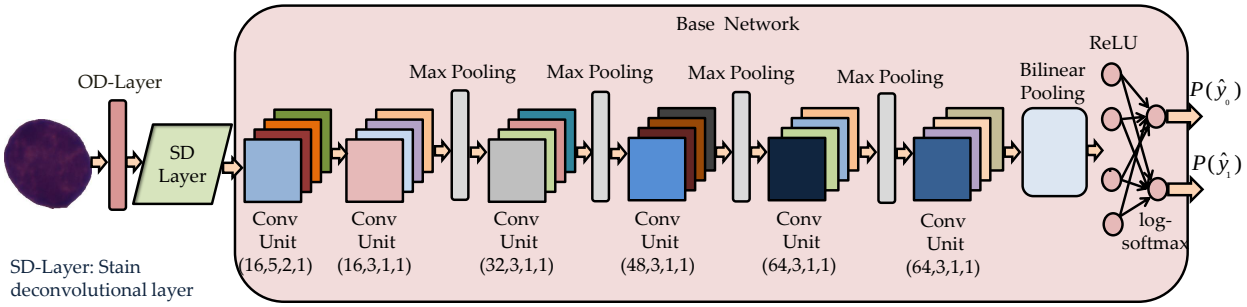
4.3.2 SD-Net Architecture

In this architecture, SD-layer (Duggal *et al.*, 2017) is stacked at the front-end of the *base network*. It is, hereby, named as SD-Net (refer to Fig. 4.3). For better understanding of the architecture we provide a brief review of the related background.

Beer-Lambert Law: Assuming light with intensity $I_{o,k}$ with $k = 0, 1, 2$ representing red, blue, or green channels, illuminates a stained slide, then according to the Beer-Lambert law:

$$I(i,k) = I_{o,k} \exp^{-\sum_{j=0}^{N-1} D(i,j)(\lambda(j,k)}, \tag{4.1}$$

where $I(i,k)$ is the light intensity reflected by the i^{th} pixel in the k^{th} channel, N represents the total number of stain chemical's components, $D(i,j)$ is the quantity of the j^{th} stain component absorbed by the i^{th} pixel, and $\lambda(j,k)$ denotes j^{th} stain component's characteristic absorbance in the k^{th} color channel. It is noted that $I(i,k)$ (reflected pixel intensity) and D (quantity of absorbed stain) follow a non-linear relation. However, a



Conv Unit (A,B,C,D): Denotes 2D convolutional layer with parameters (A,B,C,D) with batch normalization and ReLu activation, where A denotes the number of filters, B denotes kernel size, C indicates size of stride and D denotes padding size

Figure 4.3: SD-Net architecture. SD-layer deconvolves stain color vector in the OD space and provides stain quantity images, i.e., images that contain the quantification of the absorbed stain, a characteristic of the tissue.

linear relationship can be established in the optical density (OD) space as:

$$O(i,k) = -\log_{10}\frac{I(i,k)}{I_{o,k}} = \sum_{j=0}^{2} D(i,j)(\lambda(j,k), \tag{4.2}$$

or in matrix notation,

$$\mathbf{O}_{RC\times 3} = \mathbf{D}_{RC\times 3}\boldsymbol{\lambda}_{3\times 3} \tag{4.3}$$

where we have assumed $N = 3$ constituents of staining chemical, each row of $\boldsymbol{\lambda}$ represents one basis vector for this stain constituent, and the quantity of these stain vectors available at different pixels is represented by the columns of $\mathbf{D}$. Also, R and C represent the number of rows and columns, respectively, in each channel. Given Eq. 4.3, $\mathbf{O}$ can be factored using non negative matrix factorization (NMF) (Li and Plataniotis, 2015) or Singular value decomposition (SVD)(Macenko *et al.*, 2009) to obtain $\mathbf{D}$ and $\boldsymbol{\lambda}$. Hence, OD-space is helpful in quantifying absorbed stain that is, otherwise, not possible in the RGB space.

Stain-Deconvolution Layer (SD-layer): Since the texture of the cell is representative of the quantity of stain absorbed, it represents cell characteristic that can provide better class distinguishing features as compared to RGB pixel intensities. To accommodate this fact, SD-layer is pre-appended to a CNN architecture in one of our earlier works (Duggal *et al.*, 2017). Given OD-space representation $\mathbf{O}$, the absorbed stain quantities ($\mathbf{D}$) can be calculated through deconvolution of $\mathbf{O}$ and $\boldsymbol{\lambda}$ as:

$$\mathbf{D} = \mathbf{O}\boldsymbol{\lambda}^{-1} \tag{4.4}$$

Initially, $\boldsymbol{\lambda}$ is calculated from a reference image through matrix factorization (Eq.4.3) and is then updated through backpropagation. This process enables the network to see tissue-specific stain absorption quantities. It has been shown by Duggal et al. (2017) that the introduction of the SD-layer as the first layer leads to better performance of the classifier. Motivated by this, SD-Net utilizes SD-layer at the front-end and hence, works in the OD-space rather than the RGB space. SD-Net is also used as the baseline classifier in experiments.

4.3.3 SDCT-Net Architecture

We build upon the SD-Net to achieve better class discrimination capability. As seen in Fig. 4.1, images of the two classes are visually indistinguishable. In such a scenario, it is more intuitive to work in another domain that is more helpful in capturing class-specific characteristics. The assumption is that each class would have some distinct properties that can be highlighted by applying a specific transform. For images, one such property is sparsity. A signal $\mathbf{x}$ is $r-$ sparse if only r elements are non-zero. In general, compressibility, which is less strict condition than sparsity, is used frequently. The magnitude of sorted coefficients of a compressible signal decay quickly. Images are generally compressible in the discrete cosine transform (DCT) domain. The famous JPEG compression uses a compression technique based on DCT. This property of compressibility of DCT is used in many other applications, including denoising and compressive sensing based reconstruction (Ansari and Gupta, 2017).

In general, images have different sparsification levels, depending upon their internal structure. This understanding can be extended to images of different classes as well, i.e., images of different classes will be sparsified differently. To analyze this effect, we appended a DCT layer after the SD-layer. The output of the DCT-layer is given as:

$$\mathbf{D}^j = sign(\mathbf{Z}^j) \circ \log_{10}\big(\mathbf{I} + |\psi(\mathbf{Z}^j)|\big), \qquad j = 0, 1, 2 \tag{4.5}$$

where, $\circ$ denotes the Hadamard (elementwise) product,$\psi(.)$ denotes the 2D-DCT operator, $\mathbf{Z}^j$ denotes the j^{th} feature map input to the DCT layer, and $\mathbf{I}$ denotes a matrix with '1' at each position of the matrix. This '1' is added to the DCT output at every pixel before logarithm so that the resulting image $\mathbf{D}$ contains only positive quantities. Since

the dynamic range of the DCT output image is broad, “log" acts as the normalization function and allows the contribution from all pixels to be exploited by the following layers. Thereafter, each obtained quantity is multiplied with its sign to avoid any loss of information.

The feature maps/images obtained at the output of the SD-layer are passed to the above explained DCT-layer. The impact of DCT-layer on these feature maps is shown in Fig. 4.4. Interestingly, it is observed that the normal cell images are sparsified more by DCT-layer compared to the cancer cell images. The dynamic range of DCT coefficients of the two classes is also different. This observation can be exploited by the following *base network* to enhance its classification performance. This architecture with DCT layer between the SD-layer and the *base network*, inspired by our earlier work (Mourya *et al.*, 2018), is shown in Fig. 4.7 and is, hereby, named as SDCT-Net. This architecture is a sub-part of the proposed architecture SDCT-AuxNet$^{\theta}$, described next.

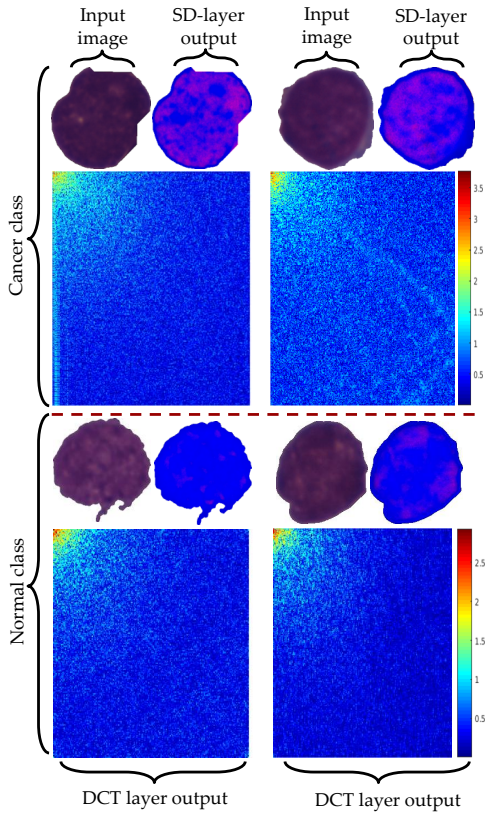
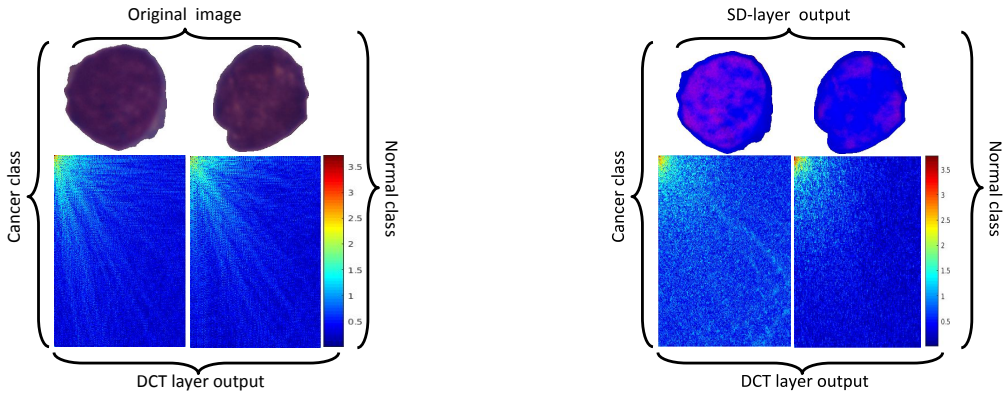


Figure 4.4: Impact of DCT-layer on the output images of SD-layer. Normal cell images are sparsified more compared to cancer cell images with the DCT-layer.

Effectiveness of DCT-layer

We have shown the output of DCT-layer (Fig.4.5(a) below) on the RGB cell images of cancer class and normal (healthy) class. We do not see any characteristic difference between the DCT-layer output of cancer and healthy images in terms of induced sparsity/output pattern (Fig.4.5(a) and Fig.4.6(a)). This observation implies that DCT-layer is not effective when applied directly to the original RGB images. However, difference is indeed observed when DCT-layer is added on the output of SD-layer as shown in (Fig.4.5(b) and Fig.4.6(b) below). From these figures, it is observed that the DCT-layer induces significant sparsity and distinction in the two classes, when added on the output of the SD-layer. Hence, the cumulative effect of DCT-layer and SD-layer can be used to enhance dissimilarity in microscopic images that otherwise appear similar in RGB space.

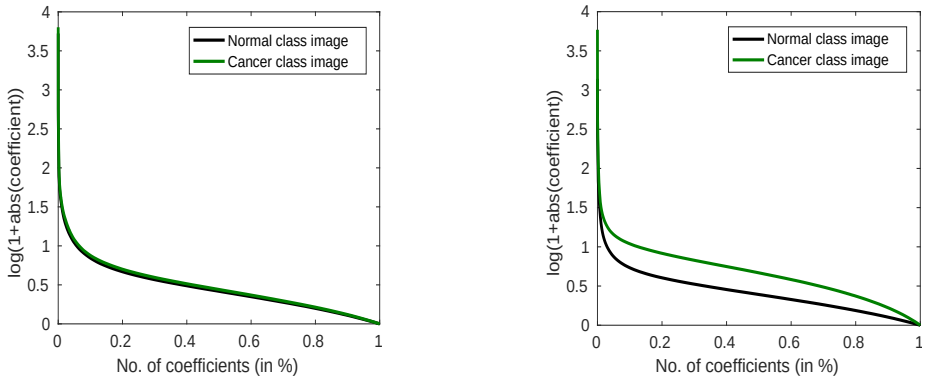


(a) (b)

Figure 4.5: (a) Effect of DCT-layer on the original RGB images. The output of the DCT-layer to cancer and healthy RGB input images is approximately similar. (b) Effect of DCT-layer on the the output of SD-layer images. DCT-layer when applied on the SD-layer shows differences between the output images of two classes.

4.3.4 SDCT-AuxNet$^{\theta}$

Next, we propose a novel framework that consists of SDCT-Net and an additional auxiliary classifier. An overview of this architecture is shown via a block diagram in Fig. 4.8. SDCT-AuxNet$^{\theta}$ has two sub-networks: i) features learning sub-network and ii) classification sub-network. In general, learned features are applied to a neural network after applying some transformations. But, SDCT-AuxNet$^{\theta}$ processes the feature maps obtained from the feature learning sub-network through two different transformations: 1)



(a) (b)

Figure 4.6: Comparison of the effect of DCT-layer in terms of sparsity (a) on original RGB cell images and (b) on SD-layer output images. DCT-layer is not able to induce any noticeable sparsity difference in original images of two classes but a significant dissimilarity is observed when DCT is applied on the output of SD-layer.

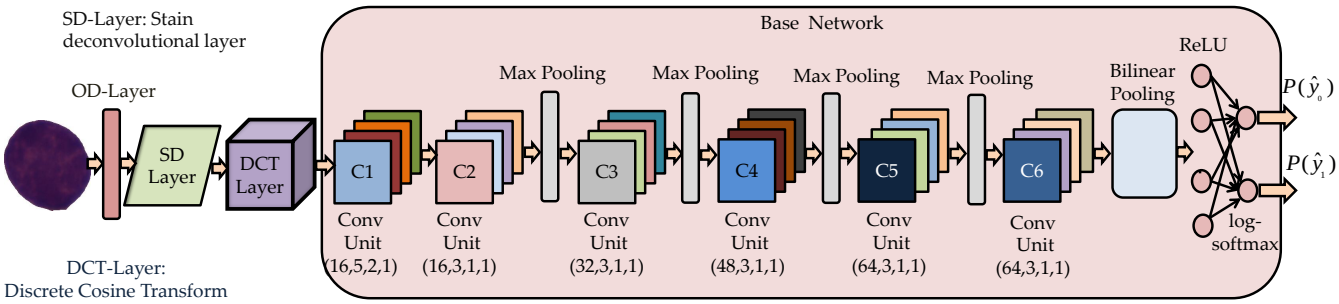


Figure 4.7: Proposed SDCT-Net. DCT layer is appended after SD-layer which helps to learn class discriminative features by inducing different sparsity in cancer and normal cell images.

bilinear pooling that feeds to the neural network for decision making and 2) spectral averaging that carries out averaging across all feature maps at every pixel position and feeds the output to the auxiliary classifier.

Training Methodology for the proposed SDCT-AuxNet$^{\theta}$

The training procedure of SDCT-AuxNet$^{\theta}$ is depicted with more clarity in Fig. 4.9. Although two different classifiers are used (main CNN classifier and auxiliary classifier), they are not trained simultaneously. Let $\{\mathbf{X}_i^{train}\}_{i=1,2,...,K}$ with size $m \times n \times 3$ represents a set of K input images and $\{y_i^{train}\}_{i=1,2,...,K}$ represents true labels of the respective images. In Step-1, SDCT-Net (feature learning sub-network + neural network) is trained in an end-to-end fashion by minimizing the negative log likelihood loss function. Let

$P(\hat{y}_{i,0}^{train})$ and $P(\hat{y}_{i,1}^{train})$ denote the probabilities that the i^{th} training image belongs to normal class (label 0) and

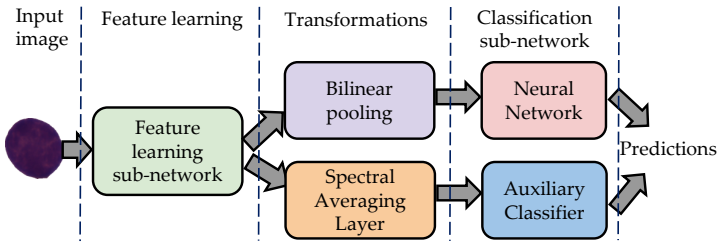


Figure 4.8: An overview of the proposed SDCT-AuxNet$^{\theta}$. The SDCT-AuxNet$^{\theta}$ can be interpreted as feature learning sub-network with two different classifiers, one classifier operating on the features obtained after bilinear pooling, and other classifier using the features obtained through spectral averaging.

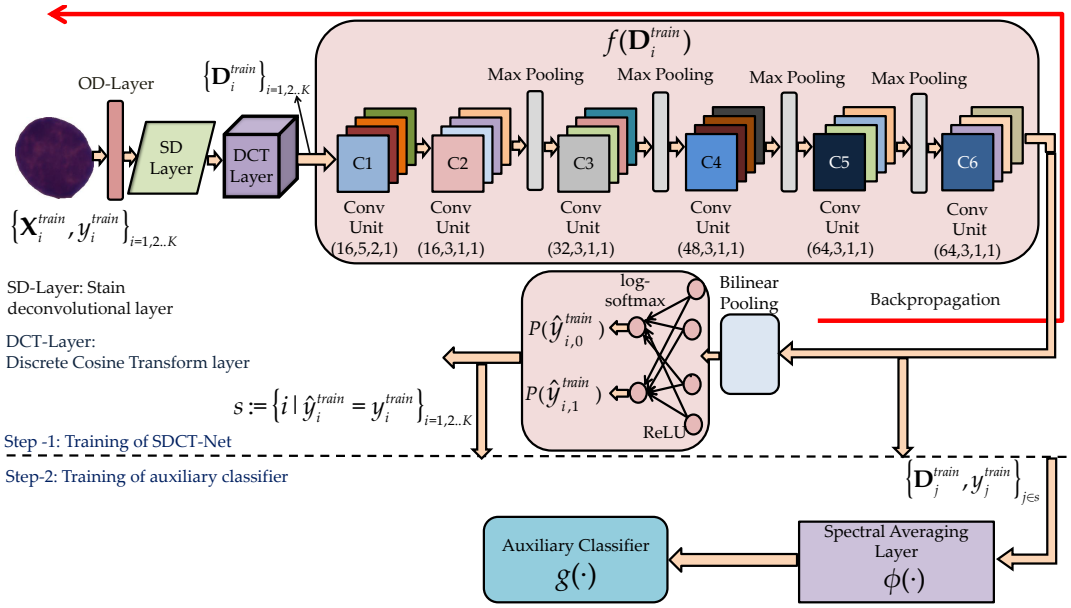


Figure 4.9: Two-steps training of SDCT-AuxNet$^{\theta}$. In Step-1, SDCT-Net is trained in an end-to-end fashion on the entire training set containing K training images with indices $i = 1, 2, ..., K$. In Step-2, correctly classified training images by SDCT-Net are used to train an auxiliary classifier.

cancer class (label 1), respectively. The predicted label is given as

$$\begin{aligned} \hat{y}_i^{train} &= 1 \qquad \text{if} \quad P(\hat{y}_{i,1}^{train}) > P(\hat{y}_{i,0}^{train}), \\ &= 0 \qquad \text{Otherwise} \end{aligned} \tag{4.6}$$

Once SDCT-Net is trained, we predict labels of all the training images using this trained SDCT-Net. We define s as a set of indices of training images for which SDCT-Net made

the correct predictions, i.e.,

$$s := \{i | \hat{y}_i^{train} = y_i^{train}\}, \quad \text{for} \quad i = 1, 2, ..., K. \tag{4.7}$$

In Step-2, images $\{\mathbf{X}_j^{train}\}_{j \in s}$ are utilized to train the auxiliary classifier. These images are first processed by the feature learning sub-network followed by the spectral averaging layer ($\phi(\cdot)$).

This is to note that only the images, whose labels are correctly predicted by the main CNN classifier (SDCT-Net), are used to train the auxiliary classifier. Also, the auxiliary classifier is not used to update the parameters of the feature learning sub-network. The strategy is to exploit two different classifiers and yet train the architecture in an end-to-end fashion. The intuition behind this approach is simple. Different transformations applied to learned features will capture different information and hence, two different but coupled classifiers trained on two distinct sets of features of the same images may perform better.

Testing Methodology for proposed SDCT-AuxNet$^\theta$

Once the SDCT-AuxNet$^\theta$ is trained, we use the following strategy to predict the label of an input test image $\mathbf{X}^{test}$. First, a prediction is made using the main CNN classifier of the SDCT-AuxNet$^\theta$ via (4.6). Let $P(\hat{y}_0^{test})$ and $P(\hat{y}_1^{test})$ denote the prediction probabilities that the test image belongs to normal class (label 0) and cancer class (label 1), respectively. These prediction probabilities on the test image represent the confidence of the main CNN classifier in making the predictions for the respective labels. We introduce a confidence probability $\theta \in [0, 1]$ as a user-defined variable. We assume the prediction from the CNN classifier to be correct if its predicted label's probability is higher than θ. For more clarity, let us define

$$\tau = max\left(P(\hat{y}_0^{test}), P(\hat{y}_1^{test})\right). \tag{4.8}$$

This parameter τ is compared with the user-defined confidence probability θ and the decision is made as below:

$$\hat{y}^{test} = c \quad \text{if} \quad \tau > \theta \quad \text{and} \quad P\left(\hat{y}_c^{test}\right) > P\left(\hat{y}_{1-c}^{test}\right), \quad \text{where} \quad c = 0, 1. \tag{4.9}$$

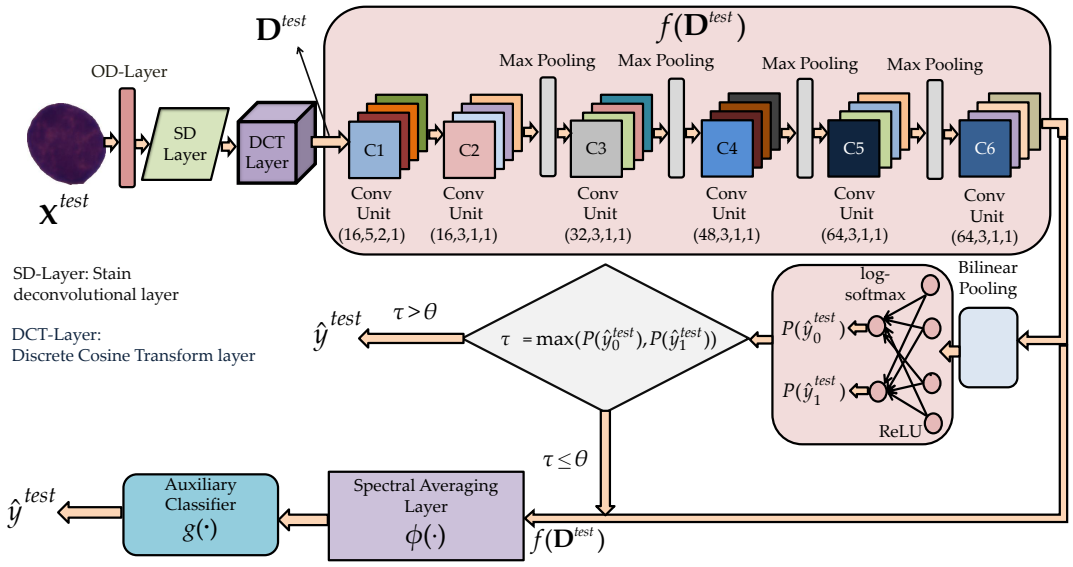


Figure 4.10: Testing strategy of the proposed SDCT-AuxNet$^{\theta}$. For each sample, probability of SDCT-Net decision label is compared against the confidence probability. If the probability of the predicted label by SDCT-Net is less than the confidence probability, it is predicted again using the auxiliary classifier.

However, if τ less than θ, we use the auxiliary classifier to predict the label of that image and consider that to be the final prediction.

To predict a label using the auxiliary classifier, features of the input image obtained from *feature learning sub-network* are given to the spectral averaging layer, and the output of this layer is fed to the auxiliary classifier. The prediction of the auxiliary classifier is given as:

$$\hat{y}^{test} = g(\phi(f(\mathbf{D}^{test}))), \tag{4.10}$$

where $\phi(f(\mathbf{D}^{test}))$ is the output of spectral averaging layer and $f(\mathbf{D}^{test})$ are the feature maps obtained from $f(\cdot)$. This process is demonstrated in Fig. 4.10. Using this approach, we take into consideration the impact of two different classifiers, a compact CNN classifier and an auxiliary classifier. Since we are making predictions with two different classifiers using different set of features, we hypothesize that the incorrect prediction of one classifier can be corrected by the other classifier.

4.4 Results

In this section, we analyze all the above discussed architectures and also compare their performances.

4.4.1 System Configuration

A forty core system with 126 GB RAM is used to train and test all the networks. We have also used an Nvidia GTX 1080 Ti GPU to boost up the training and testing process. Also, PyTorch deep learning library is used to implement all the architectures. We have used stochastic gradient descent (SGD) optimizer with an initial learning rate of 0.001 and a batch size of 64. A total of 130 epochs are used for training the networks, during which the learning rate is dropped regularly by a factor of 10.

4.4.2 Performance Metrics

We have used F_1 score, weighted F_1 score and balanced accuracy to evaluate the performance of classifiers. Weighted F_1 score and balanced accuracy are reported to take into account the class imbalance problem of the data. Further, we report F_1 score on each class to show class-wise performance. F_1 score is defined mathematically as

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall}. \tag{4.11}$$

For binary data, weighted F_1 score is defined as

$$WF_1 = \frac{\sum_{j=0}^{1} n(c_j) F_1(c_j)}{N}, \tag{4.12}$$

where $F_1(c_j)$ is the F_1 score of j^{th} class, $n(c_j)$ is the number of samples in j^{th} class, and N is total number of samples. Finally, balanced accuracy is defined as

$$BAC = \frac{recall + specificity}{2}. \tag{4.13}$$

Precision is the ratio of correctly detected positive samples to all predicted positive samples. Recall is defined as the ratio of correctly predicted positive samples to all positive samples in the ground truth. Specificity denotes the ratio of correctly classified negative samples to all negative samples in the ground truth. Recall reflects the classifier's ability to correctly diagnose the disease, while specificity shows the classifier's effectiveness in discarding the healthy subjects. For an ideal classifier, the value of precision, recall, and specificity is 1.

4.4.3 Classifiers' Training and Testing Information

For training and testing the networks, we used seven-fold cross-validation strategy. The training set is divided into seven-folds. At any time, six folds are used for training and the remaining one fold is used for validation. Therefore, for every architecture (SD-Net, SDCT-Net, or SDCT-AuxNet$^{\theta}$), we have a total of seven trained models: one for each different validation fold. To make the final prediction on the new test sample $\mathbf{X}_i^{test}$, we note the individual prediction from each model as

$$\hat{y}_i^{test,m} = \gamma_m(\mathbf{X}_i^{test}) \qquad m = 1, 2, ..., 7, \text{ and } i = 1, 2, ..., N, \tag{4.14}$$

where $\gamma_m(.)$ represents the output of the model trained using the m^{th} fold as the validation set and N denotes the total number of test samples.

The final prediction for each i^{th} test sample is given by majority vote as:

$$\hat{y}_i^{test} = mode\{\hat{y}_i^{test,1}, \hat{y}_i^{test,2}, ..., \hat{y}_i^{test,7}\}. \tag{4.15}$$

To summarize, the label predicted by the majority of the models for an architecture (SD-Net, SDCT-Net, or SDCT-AuxNet$^{\theta}$) is chosen as its final prediction.

Another issue that needs to be addressed during the training is class imbalance because training using an imbalanced dataset may result in a biased classifier. From Table 4.1, it is noted that the training set is skewed towards the cancer class as the number of cancer cell images is approximately twice the total number of normal cell images. To resolve this, we have used the *oversampling* strategy in which images of the minority class are duplicated, such that the total sample count of both the classes is the same.

Further, to enhance the performance, we have used augmentation techniques that simulate the conditions of real-life data preparation. In general, three types of augmentations are natural for these cell images. First, it is observed that 3-dimensional cells get sheared to some extent on the 2-dimensional slide when cells are pushed between two glass slides during the microscopic slide preparation. Secondly, cells would fall on the slide at varying orientations and hence, would be captured with different rotated angles. Thirdly, there can be variations in the sharpness of the images because the manual focusing of the microscope lens was carried out while capturing the images. Thus,

to mimic these effects during the training process, shear augmentation is used within a range of [-20, 20]; augmentation of random rotation, horizontal flip, and vertical flip is used; and Gaussian blur augmentation is used within a range of [0.0, 0.75]. The same augmentations are used with all architectures: SD-Net, SDCT-Net, and SDCT-AuxNet$^{\theta}$.

Training curves of SDCT-Net are shown in Fig. 4.11. Graphs show the accuracy and loss of SDCT-Net trained with different folds as the validation sets. It is observed from the graphs that SDCT-Net converged in 130 epochs. Also, SDCT-Net is able to generalize well because the performance on training and validation sets is similar. Similar observation is noted for SDCT-AuxNet$^{\theta}$.

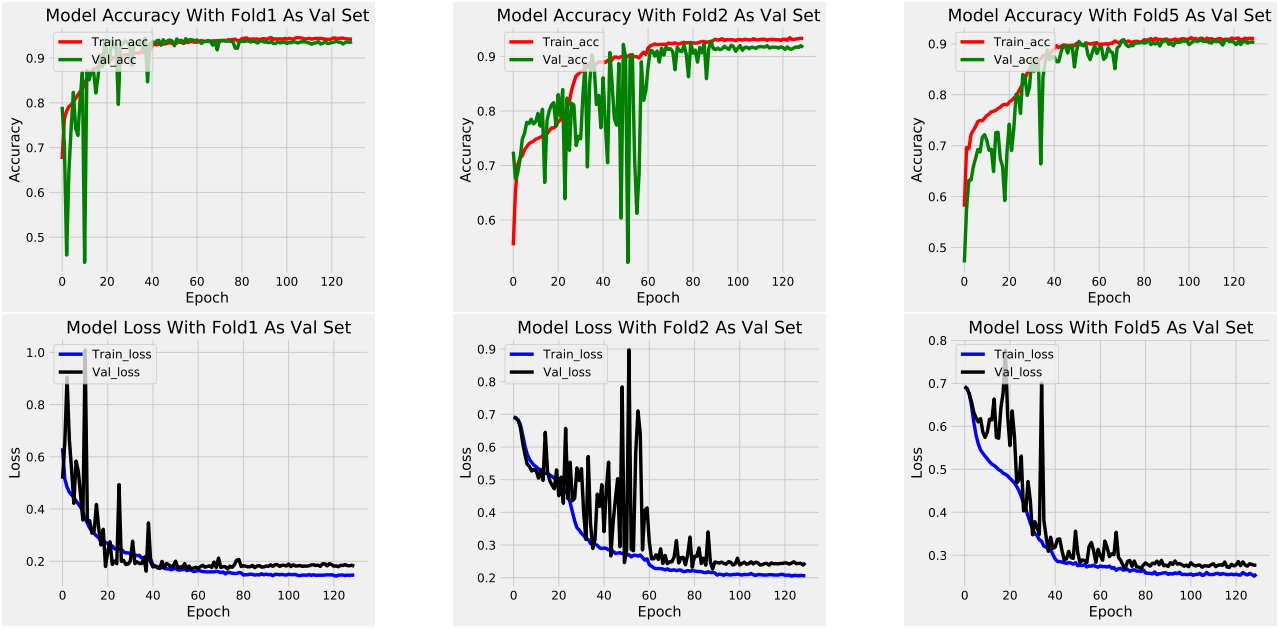


Figure 4.11: Training curves of SDCT-Net with three different folds as the validation sets.

4.4.4 Experiments with SDCT-AuxNet$^{\theta}$

In this section, we analyze the performance of SDCT-AuxNet$^{\theta}$ architecture according to the testing strategy as discussed in Section 4.3.4. Support Vector Machine (SVM) with radial basis function (RBF) kernel is used as the auxiliary classifier. We analyze SDCT-AuxNet$^{\theta}$ using different experiments: 1) by studying the impact of confidence probability θ on the performance of SDCT-AuxNet$^{\theta}$, 2) by studying the performance comparison of SDCT-AuxNet$^{\theta}$, SDCT-Net^{0}, and SDCT-Net^{1}, 3) by studying the subject-level performance of SDCT-AuxNet$^{\theta}$.

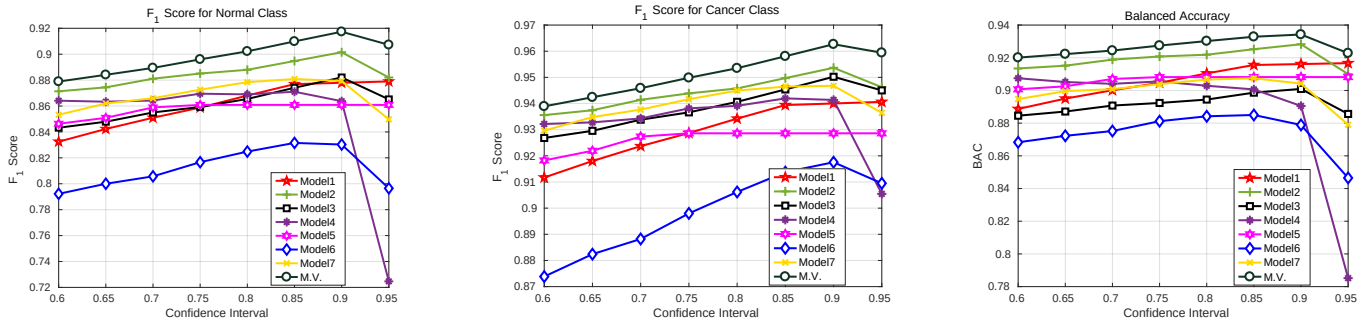


Figure 4.12: Performance of SDCT-AuxNet$^{\theta}$ for different values of confidence probability (θ): (a) F_1 score of normal class, (b) F_1 score of cancer class and (c) Balanced Accuracy. Model1 represents model trained with fold1 as the validation set and remaining folds as training set. Similar notation applies to Model2 to Model7. M.V is the score with majority voting on Model2 to Model7. Performance improves consistently with the increasing value of θ until $\theta = 0.9$, after which a decline in the performance is noted. Results are presented on the test set of 2586 images.

Table 4.3: Weighted F_1 score of SDCT-AuxNet$^{\theta}$ for different values of confidence probability (θ). Model1 represents the model trained with fold1 as the validation set and remaining folds as training set. Similar notation applies to Model2 to Model7. The weighted F_1 (WF_1) score improves with increasing θ until $\theta = 0.9$, beyond which the performance starts declining. Results are presented on the test set of 2586 images. Best results are indicated in bold case.

Weighted F_1 (WF_1) Score								
Confidence Probability (θ) \Models	Model1	Model2	Model3	Model4	Model5	Model6	Model7	Majority Voting
0.6	0.886	0.915	0.900	0.910	0.895	0.848	0.905	0.920
0.65	0.894	0.917	0.904	0.911	0.899	0.856	0.912	0.924
0.7	0.900	0.922	0.909	0.912	0.906	0.862	0.915	0.928
0.75	0.906	0.925	0.912	0.916	0.907	0.872	0.920	0.933
0.8	0.913	0.927	0.917	0.917	0.907	0.880	0.924	0.937
0.85	0.919	0.932	0.923	0.919	0.907	0.888	0.926	0.943
0.9	0.920	0.937	0.928	0.917	0.907	0.890	0.925	**0.948**
0.95	0.921	0.926	0.920	0.848	0.907	0.873	0.909	0.943

Impact of Confidence Probability

We experiment with different values of the confidence probability (θ) to analyze its effect on the performance of the model. Results are summarized in Fig. 4.12 and Table 4.3. Fig. 4.12 shows the effect of confidence probability on individual class F_1 score and balanced accuracy. It is observed that for the most of the models, performance improves with an increase of θ until $\theta = 0.9$ after which a decline in performance is observed. The final prediction, i.e., prediction with majority voting also improves until

$\theta = 0.9$ and decreases thereafter.

Comparison of SDCT-AuxNet$^{\theta}$, SDCT-Net^{0} and SDCT-Net^{1}

We have compared the performance of SDCT-AuxNet$^{\theta}$ for $\theta = 0.90$, $\theta = 1$, and $\theta = 0$ in Table 4.4. SDCT-AuxNet^{0} and SDCT-AuxNet^{1} correspond to the results with $\theta = 0$ and $\theta = 1$, respectively, in SDCT-AuxNet$^{\theta}$. This is to note that SDCT-AuxNet^{0} is SDCT-AuxNet$^{\theta}$ without auxiliary classifier and hence, SDCT-AuxNet^{0} is same as SDCT-Net. Similarly, SDCT-AuxNet^{1} is SDCT-AuxNet$^{\theta}$ with only auxiliary classifier. Table 4.4 shows the performance of SDCT-AuxNet$^{\theta}$ that exploits two different classifiers (SDCT-AuxNet^{0} and SDCT-AuxNet^{1} with different architectures) in a coupled fashion. From the performance of individual models (Model1-Model7) of all these architectures, it can be inferred that SDCT-AuxNet$^{\theta}$ has achieved a significant gain over SDCT-AuxNet^{0} and SDCT-AuxNet^{1} in terms of all three performance metrics. Although majority voting is able to bridge the performance difference between SDCT-AuxNet$^{\theta}$ and SDCT-AuxNet^{1}, the former is leading the later for most of the models (Model1-Model7), while the performance coincides in some cases. Compared to SDCT-AuxNet^{0} (SDCT-Net), SDCT-AuxNet$^{\theta}$ provides a gain of 5.5% for F_1 score of normal class, 3.5% for F_1 score of cancer class, 4.0% for weighted F_1 score, and 2.3% for balanced accuracy. The highest gain is observed for F_1 score of normal class, which is 0.917 with SDCT-AuxNet$^{\theta}$ as compared to 0.862 with SDCT-AuxNet^{0}. To summarize, SDCT-AuxNet$^{\theta}$ ensembles the performance of SDCT-AuxNet^{0} and SDCT-AuxNet^{1} that results in significant performance improvement. The results of SDCT-Net with skip connections are also summarized in A.2

Subject-Level Performance of SDCT-AuxNet$^{\theta}$

We have also evaluated the performance of SDCT-AuxNet$^{\theta}$ for individual subjects in terms of accuracy. The subject-level accuracy and error rate for both the classes is depicted in Fig. 4.13. SDCT-AuxNet$^{\theta}$ performs consistently good for all the subjects. These results are encouraging since it shows that the model is robust to subject-level variability.

Table 4.4: Performance comparison of SDCT-AuxNet^0, SDCT-AuxNet^1, and SDCT-AuxNet$^\theta$ for $\theta = 0.9$. SDCT-Net is is a special case of SDCT-AuxNet$^\theta$ with $\theta = 0$ and is represented as SDCT-AuxNet^0. Similarly, SDCT-AuxNet^1 is a spacial case of SDCT-AuxNet$^\theta$ with only the auxiliary classifier branch. Model1 represents the model trained with fold1 as the validation set and remaining folds as training set. Similar notation applies to Model2 to Model7. Best results are highlighted in black and equal performance is highlighted in blue color. SDCT-AuxNet$^\theta$ performed better than SDCT-AuxNet^0 or SDCT-AuxNet^1. Results are presented on the test set of 2586 images.

F_1 Score for Normal Class								
Architecture\Model	Model1	Model2	Model3	Model4	Model5	Model6	Model7	Majority Voting
SDCT-AuxNet^0	0.813	0.855	0.836	0.856	0.822	0.774	0.825	0.862
SDCT-AuxNet^1	0.878	0.882	0.865	**0.868**	0.860	0.794	0.849	0.915
SDCT-AuxNet$^\theta$	0.878	**0.902**	**0.882**	0.864	**0.861**	**0.830**	**0.879**	**0.917**
F_1 Score for Cancer Class								
Architecture\Model	Model1	Model2	Model3	Model4	Model5	Model6	Model7	Majority Voting
SDCT-AuxNet^0	0.900	0.926	0.921	0.926	0.900	0.855	0.912	0.928
SDCT-AuxNet^1	0.940	0.946	0.945	0.936	0.928	0.908	0.936	0.961
SDCT-AuxNet$^\theta$	0.940	**0.954**	**0.950**	**0.941**	**0.929**	**0.918**	**0.947**	**0.963**
Weighted F_1 (WF_1) Score								
Architecture\Model	Model1	Model2	Model3	Model4	Model5	Model6	Model7	Majority Voting
SDCT-AuxNet^0	0.873	0.904	0.894	0.904	0.875	0.829	0.885	0.908
SDCT-AuxNet^1	0.920	0.925	0.919	0.915	0.907	0.872	0.908	0.947
SDCT-AuxNet$^\theta$	0.920	**0.937**	**0.928**	**0.917**	0.907	**0.89**	**0.925**	**0.948**
Balanced Accuracy (BAC)								
Architecture\Model	Model1	Model2	Model3	Model4	Model5	Model6	Model7	Majority Voting
SDCT-AuxNet^0	0.875	0.903	0.882	0.905	0.887	0.856	0.879	0.911
SDCT-AuxNet^1	0.916	0.910	0.885	**0.906**	0.908	0.845	0.879	0.932
SDCT-AuxNet$^\theta$	0.916	**0.928**	**0.901**	0.891	0.908	**0.879**	**0.904**	**0.934**

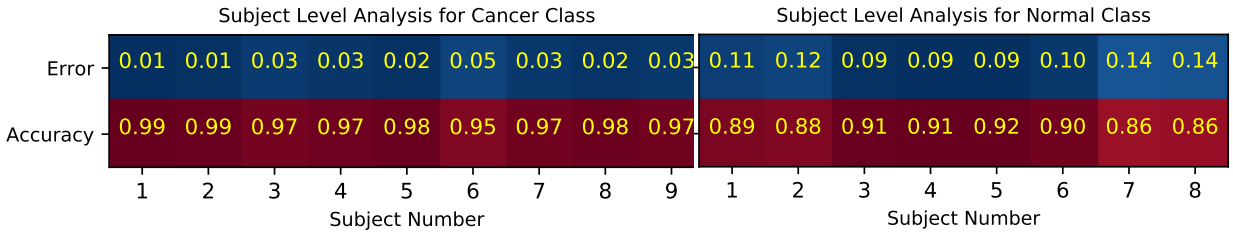


Figure 4.13: Subject-level performance of SDCT-AuxNet$^\theta$ is observed to be consistent across all the subjects indicating that SDCT-AuxNet$^\theta$ is robust to subject-level variations.

4.4.5 Comparison with the state-of-the-art Methods

This C-NMC 2019 dataset has recently been used by the participants of the ISBI-2019 challenge (Gupta *et al.*, 2019). Many top-performing methods in the challenge have used ResNet and other related convolutional neural networks as the backbone architectures. Here, we provide a summary of these methods along with the performance com-

parison with our proposed method. Pan *et al.* (2019) used a neighborhood-correction algorithm (NCA) with fine-tuned ResNets as the backbone architectures. The feature maps produced from fine-tuned ResNets are used to construct Fisher vectors, which are then used for label correction based on the neighbors' majority. An ensemble of this strategy with ResNet50, ResNet101, and ResNet150 is used to make the predictions on the final test set. This method achieved a weighted F_1 score of 91.04% and secured the first position in the challenge. Prellberg and Kramer (2019) utilized a pretrained ResNeXt50 and fine-tuned it with different learning rates for different layers and used augmentations during testing. This method achieved a final weighted F_1 score of $.889$. Xiao *et al.* (2019) used ensembling of DenseNet121, ResNet101, SENet154, VGG16, DeepTEN, and InceptionV4 along with pseudo labeling approach, achieving a final weighted F_1 score of $.885$. Shi *et al.* (2019) utilized PNASNet with fusion and voting approach to achieve a weighted F_1 score of $.879$. Liu and Long (2019) have divided the larger class samples into two subsets, which are then combined with other class samples to train two different models initialized with pretrained Inception ResNets weights. These two models are then fine-tuned jointly on whole training dataset to obtain the final predictions. This method is able to achieve a weighted F_1 score of 87.6%. Similarly, other methodologies based on Inception ResNet(Xie *et al.*, 2019), MobileNet (Verma and Singh, 2019), and ensembling (Shah *et al.*, 2019; Ding *et al.*, 2019) were utilized by other researchers.

A comparison of the proposed SDCT-AuxNet$^{\theta}$ with these top-performing methods is provided in Table-4.5. It is observed that SDCT-Net is approximately equal to the top-performing method (Pan *et al.*, 2019), whereas the proposed SDCT-AuxNet$^{\theta}$ has achieved a gain of 3.8% over the top ranked team of the IEEE ISBI-2019 Challenge.

From all the above discussed results and from Fig. 4.8, 4.9, it is evident that the proposed SDCT-AuxNet$^{\theta}$ architecture with shared feature learning sub-network ensures 1) end-to-end training, 2) a strong main classifier, 3) extraction of efficient features that can be used by auxiliary classifier, and 4) an architecture where the performance of a strong classifier is further enhanced via improving confidence probability of its decision in a systematic fashion. Further, owing to the feature shareability, the auxiliary classifier need not be end-to-end trainable yielding flexibility in the choice of auxiliary classifier.

Table 4.5: Comparison of SDCT-Net and SDCT-AuxNet$^{\theta}$ with the top entry of C-NMC 2019 Challenge. Best results are depicted in bold case.

Method	Weighted F_1
Pan *et al.* (2019)	0.910
Verma and Singh (2019)	0.894
Prellberg and Kramer (2019)	0.889
Xiao *et al.* (2019)	0.885
Shi *et al.* (2019)	0.879
Liu and Long (2019)	0.876
Shah *et al.* (2019)	0.866
Ding *et al.* (2019)	0.855
Xie *et al.* (2019)	0.848
SDCT-Net	**0.908**
SDCT-AuxNet$^{\theta}$	**0.948**

4.5 Discussion

In this section, we have analyzed SDCT-AuxNet$^{\theta}$ from different perspectives. First, we analyze SDCT-AuxNet$^{\theta}$ as a generic architecture that can be modified to two different architectures through the setting of confidence probability (θ). As discussed in above sections, confidence probability (θ) that lies in the range $[0, 1]$, plays a crucial role during the testing phase of SDCT-AuxNet$^{\theta}$. By setting the value of θ, we are deciding the contribution of two different classifiers in decision making. However, with two extreme values of θ, i.e. $\theta = 0$ or $\theta = 1$, we can also choose only one classifier. This scenario is depicted in Fig. 4.14. By setting $\theta = 0$, SDCT-AuxNet$^{\theta}$ is reduced to SDCT-AuxNet^{0} (SDCT-Net), which is the feature learning sub-network along with neural network as the classifier that utilizes bilinear pooling. For $\theta = 1$, SDCT-AuxNet$^{\theta}$ becomes SDCT-AuxNet^{1}, which utilizes feature learning sub-network along with spectral averaging and auxiliary classifier. Between these two extremes, SDCT-AuxNet$^{\theta}$ exploits bilinear pooling with neural network and spectral averaging with auxiliary classifier. This methodology is an advantage of SDCT-AuxNet$^{\theta}$ because it provides flexibility to decide the contribution of the classifiers by simply setting the value of θ. This flexibility is also the reason for the better performance of SDCT-AuxNet$^{\theta}$ over SDCT-AuxNet^{0} (SDCT-Net) or SDCT-AuxNet^{1}. In fact, the coupling of SDCT-AuxNet^{0} (SDCT-Net) and SDCT-AuxNet^{1} in SDCT-AuxNet$^{\theta}$ can also be interpreted as an ensemble of two classifiers, with each classifier utilizing a different set of features. Results of Table 4.4 validates our hypothesis that an appropriate coupling of two different classifiers (working on different sets of features) may lead to better performance as compared to an

Table 4.6: The number of samples predicted by different classifiers at $\theta = 0.9$. Contribution of each classifier is approximately the same in majority of the cases, indicating significant role of both the classifiers in decision making.

Classifier/ Models	Model1	Model2	Model3	Model4	Model5	Model6	Model7
Neural Network	1634	1396	428	1885	1199	1544	1056
Auxiliary Classifier	952	1190	2158	701	1387	1042	1530
Total Samples	2586						

individual classifier utilizing a single set of features.

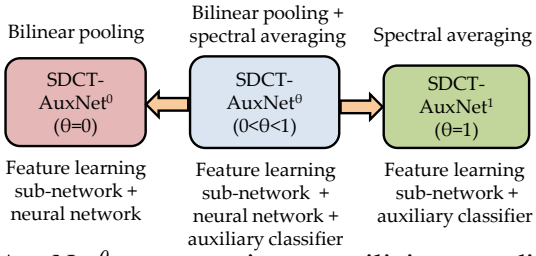


Figure 4.14: SDCT-AuxNet$^\theta$ as a generic case utilizing two different classifiers in decision making. SDCT-Net is same as SDCT-AuxNet$^\theta$ for $\theta = 0$ and utilizes only neural network. SDCT-AuxNet^1 is SDCT-AuxNet$^\theta$ with $\theta = 1$ and use only the auxiliary classifier. SDCT-AuxNet$^\theta$ swings between SDCT-AuxNet^0 (SDCT-Net) and SDCT-AuxNet^1 depending upon the value of confidence probability θ.

We have also analyzed the contribution of each classifier in the final prediction (Table 4.6). It is observed that both the classifiers contribute approximately equally in most of the cases. These results prove the hypothesis that both the classifiers are involved in decision making, and the final outcome is not due to only one of the classifiers.

We have also visualized class discrimination at different layers in Fig. 4.15. At the first convolutional layer, majority of the samples of the two classes are overlapping. However, as we go deeper into the network, the two classes start to diverge. Finally, at the last convolutional layer, the two classes can be linearly separated. This points to an intuitive understanding that difference in sparsity in SDCT-AuxNet$^\theta$ may be one of the significant causes of separability of the two classes. Although some samples still lie on the opposite side of the boundary, majority of the samples can be classified correctly. These observations establish the validity of DCT-layer in ALL classification.

Finally, samples in the medical domain are collected from different subjects, which may induce subject-level variations. Robustness to subject-level variability is a crucial

(a) First Convolutional Layer (C1) (b) Sixth Convolutional Layer (C6)

Figure 4.15: Class discrimination at first (C1) and last convolutional layers (C6). The samples are separated as we progress deeper into the network. At the final layer, majority of the samples can be easily separated even with a linear boundary.

factor for the practical implementation of the classifier. To mimic this effect, we trained and tested the architectures on folds separated at subject level. Results summarized in Fig. 4.13 prove the robustness of SDCT-AuxNet$^{\theta}$ to subject-level variability. However, error rate for normal class is high as compared to cancer class. These results imply that classifier has relatively lower specificity as compared to recall. Hence, classifier yields excellent performance in predicting the disease in unhealthy subjects, but relatively poor performance in discarding the disease in healthy subjects.

4.6 Conclusion

We proposed a novel framework, SDCT-AuxNet$^{\theta}$, for the challenging problem of ALL classification. The framework utilizes two different classifiers operating on two different set of features. SDCT-AuxNet$^{\theta}$ use sparsity in DCT domain to learn class discriminative features. In ALL classification, it is important to pool the data at subject-level due to inter-subject variability in cell images. We have used this approach to make SDCT-AuxNet$^{\theta}$ robust to subject-level variations.

CHAPTER 5

A CNN-based Unified Framework Utilizing Projection Loss in Unison with Label Noise Handling for Multiple Myeloma Cancer Diagnosis

Multiple Myeloma (MM) is a malignancy of plasma cells. Similar to other forms of cancer, it demands prompt diagnosis for successful treatment. The traditional diagnostic tools are resource-intense and hence, these solutions are not easily scalable for extending their reach to the masses. Advancements in deep learning have led to rapid developments in affordable, resource optimized, easily deployable computer-assisted solutions. This work proposes a unified framework for MM diagnosis using microscopic blood cell imaging data that addresses the key challenges of inter-class visual similarity of healthy versus cancer cells and that of the label noise of the dataset. To extract class distinctive features, we propose projection loss to maximize the projection of a sample's activation on the respective class vector besides imposing orthogonality constraints on the class vectors. This projection loss is used along with the cross-entropy loss to design a dual branch architecture that helps achieve improved performance and provides scope for targeting the label noise problem. Based on this architecture, two methodologies have been proposed to correct the noisy labels.

5.1 Introduction

Lymphocytes are a type of white blood cells that are further categorized as T lymphocytes (T cells) and B lymphocytes (B cells). B cells counter the invaders' antigens and are transformed into the plasma cells in the process. These plasma cells usually reside in the bone marrow. Multiple myeloma (MM), a type of white blood cancer, occurs due to the unrestricted growth of these plasma cells (MM, 2019). Some conventional and reliable tests for MM diagnosis include quantitative immunoglobulins, electrophoresis, and bone marrow biopsy (immunohistochemistry, flow cytometry, cytogenetics, and

fluorescent in situ hybridization) (MM, 2019). Such tests require costly medical infrastructure and a trained workforce, limiting the expansion of diagnostic facilities at the required scale in rural and urban areas.

The use of CNNs in the medical domain has seen a rapid surge in recent years (Deng *et al.*, 2020; Litjens *et al.*, 2017). However, training of CNNs from scratch requires a large dataset depending upon the depth of the CNN. The availability of a large annotated dataset for supervised learning is a challenge in the medical domain. An alternate solution is to use transfer learning, wherein a network trained on one dataset (pre-trained network) is used on another dataset. In one approach of transfer learning, a pre-trained network is used directly for feature extraction, while in another approach, a pre-trained network is fine-tuned on an available dataset. Also, training from scratch and transfer learning have been deployed frequently to classify different types of cancers. Although direct feature extraction eliminates the need for a training set, it is a less preferred approach because fine-tuning usually performs better than direct feature extraction.

The classification networks used for transfer learning are generally pre-trained on the ImageNet, a large 1000 class non-medical images' dataset. For transfer learning, the target medical images are required to match these pre-trained networks' input image size. This requires a suitable scaling of input medical images. This scaling may change the morphology of the medical constituents of images, say cells, and hence, can hurt the classifier's overall performance. Moreover, these networks may be undesirably massive for medical applications because the medical domain, in general, does not encounter these many classes as are present in the ImageNet dataset (Wong *et al.*, 2018).

Duggal *et al.* (2017) trained AlexNet and T-CNN from scratch for leukemia diagnosis. The trained architectures were then fine-tuned after including the trainable stained deconvolutional (SD) layer. However, a significant downside of this method is the aggregation of the dataset of all the subjects. This leads to the training and testing on the same subjects' data that is not the case in practical deployment, wherein a tool developed using a set of the subjects is to be tested on the prospective unseen subjects. Gehlot *et al.* (2020*b*) has addressed this issue by segregating the dataset at the subject-level such that there is no common subject between training and test datasets. The method includes utilizing a combination of SD and DCT layers pre-appended to a

compact CNN architecture to aid the extraction of distinctive features from the visually similar classes. The resultant architecture is then trained from scratch. Also, an ensembling approach utilizing an auxiliary classifier has been used to boost the classifier's overall performance.

In this work, we have employed the technique of training a custom CNN architecture from scratch to target the problem of multiple myeloma (MM) cancer diagnosis. The proposed approach includes a novel loss function, a label noise handling method, and an ensembling approach. Also, we have used a large dataset for training and testing purposes. Specifically, we have used a total of 72 subjects' data (26 healthy subjects and 46 MM cancer patients) divided into 34555 training cell images of 46 subjects and 40441 test cell images of 26 subjects. To the best of our knowledge, no other work has utilized such a large dataset for any blood cancer diagnosis. The salient contributions of this works are listed as below:

1. A novel projection loss utilizing class-specific vectors has been proposed to achieve inter-class separation by maximizing the projection between the activation and the respective class vectors. The proposed loss also constrains the class vectors to be orthogonal to each other.

2. A dual-branch architecture is used to accommodate projection loss in combination with cross-entropy loss to achieve enhanced performance. The architecture also employs two different feature pooling to capture the discerning features in multiple ways.

3. Two label noise handling approaches utilizing the dual-branch architecture has been proposed to address the training samples' label noise. The proposed approaches are unsupervised from the perspective that no label information is required.

4. A coupling classifier is proposed that resolves the ambiguity and predicts a unique label from the dual-branch architecture. This coupling classifier uses distinct sets of features from the two branches.

5. The test performance has been evaluated using a large multiple myeloma (MM) dataset of 40441 images. An ablation study highlighting the proposed contri-

butions along with the subject-level analysis has also been provided on the test dataset.

6. Two other datasets (Camelyon7 and TBX11K) have also been used to validate the proposed label noise handling approach.

5.2 Materials

We have used three datasets for the experiments. In this section, we provide a detailed description of all the three datasets.

5.2.1 Multiple Myeloma (MM) Dataset

The dataset is collected using the slides prepared from the bone marrow aspirate of the healthy and cancer subjects using the standard procedure, including staining slides with the Jenner-Giemsa stain. The stain is used for highlighting the bone marrow cells, including the plasma cells, i.e., the cells of interest. Subsequently, the slides are imaged in the .bmp format using the camera mounted on the microscope. The captured microscopic images are of size 2040×1536 pixels and contain the cells of interest annotated by the expert oncologists. These cells are segmented from the images using an in-house deep learning-based segmentation tool (Gehlot *et al.*, 2020*a*). Each segmented image contains only a single cell centered at the origin. The segmented cell images are also zero-padded to achieve a fixed spatial size of 350×350, ensuring the containment of cells of varying sizes. The cancer class samples are collected from the patients diagnosed with multiple myeloma, whereas samples of the healthy class are from subjects not suffering from cancer of plasma cells.

The dataset is collected from 72 subjects (26 healthy subjects and 46 MM cancer patients), out of which 46 subjects' data is used for training, and 26 subjects' data is used for testing. Out of the 46 subjects in the training set, 26 subjects belong to the MM cancer class, and the remaining 20 are healthy. These subjects are divided into five-folds, such that the entire data of one subject is present in one fold only. In total, there are 34555 images in the train set and 40441 images in the test set. The detailed description of the dataset is provided in Table 5.1. Also, the sample images from both classes

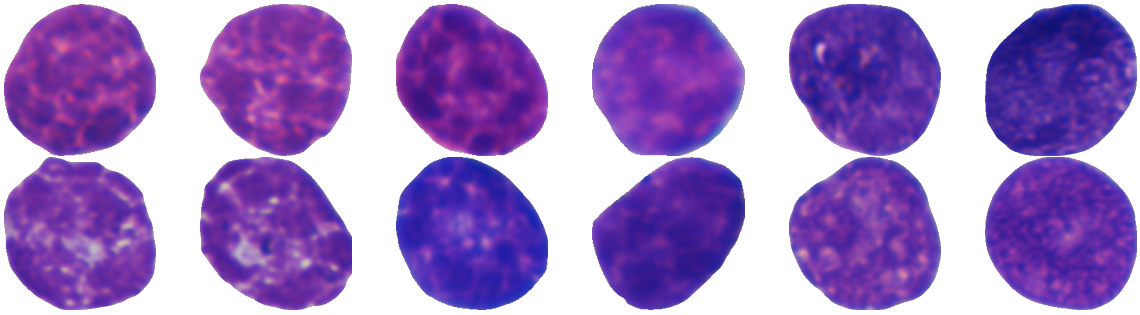


Figure 5.1: Sample images from the cancer class (first row), and the healthy class (second row). Samples of three subjects of each class have been shown.

are shown in Fig. 5.1. The test set size is sufficiently large for the fair observation of the classifier's performance. The dataset was collected at the Laboratory Oncology, AIIMS, New Delhi, India, after the Ethics Committee's approval. The subjects' confidentiality was maintained during the data collection process. The subject-specific information was entirely removed before sharing the data for experiments.

Table 5.1: Data Description.
Number of subjects and images in the training and test sets. Distribution of subjects and images in different folds of training data is mentioned. The folds have been prepared such that almost 1:1 ratio of the data of healthy and cancer class is maintained, while a subjects' data is present in only one fold.

Training set						
	Folds	1	2	3	4	5
No. of subjects	Cancer Class	5	5	5	5	6
	Healthy Class	4	4	4	3	5
	Total no. of subjects in fold	9	9	9	8	11
No. of Images	Cancer Class	3456	3134	3679	3673	3471
	Healthy Class	3150	3428	3606	3870	3088
	Total no. of images in fold	6606	6562	7285	7543	6559
Total no. of subjects		46				
Total no. of images		34555				
Test set						
No. of subjects	Cancer Class	20				
	Healthy Class	6				
	Total no. of subjects	26				
No. of Images	Cancer Class	19366				
	Healthy Class	21075				
	Total no. of images	40441				

Challenges of the dataset From Fig. 5.1, it is evident that the images of both the classes are visually similar. Moreover, there is a stain variation in the cell images of data of different subjects. In general, stain color variation occurs owing to multiple

reasons (Gupta *et al.*, 2020*a*). Since this data is collected over a period of two years, the significant reason for stain variation is the use of different manufactured batches of the staining chemical to prepare the slides. Although stain normalization can be used before cell segmentation/classification (Gupta *et al.*, 2020*a*), we have eliminated this step to make the problem more challenging.

5.2.2 Camelyon17 (Bandi *et al.*, 2019)

It is a publicly available dataset and consists of 50 annotated whole slide images (WSIs) collected from five different centers (C0-C4). The WSIs have captured H&E stained slides prepared from the lymph node sections. We have extracted 128×128 pixels patches from WSIs of center-3 (C3) for experiments. From the collected patches, 74,633 are used for training, 15000 for validation, and the remaining 40005 for testing. We have performed a classification task on the resultant dataset wherein the patches containing metastatic tumor cells are annotated as cancer and healthy otherwise.

5.2.3 TBX11K (Liu *et al.*, 2020)

It is a Tuberculosis X-ray dataset having images of 512×512 pixels. Each X-ray image is annotated as healthy or sick & non-TB or TB. The dataset also contains the images from Chauhan *et al.* (2014) and Jaeger *et al.* (2014). This results in the training set of 6889 images, a validation set of 2087 images, and a test set of 3302 images. The GT is available only for training and validation sets and not for the test set. There is a challenge (TBX11K Tuberculosis Classification and Detection Challenge, 2020) available on this dataset, and the test set performance can be evaluated through the challenge portal. The challenge consists of two tasks; TB detection and classification. We have performed analysis only on the classification task.

5.3 Methods

In this section, we will discuss the proposed projection loss, architecture, and noisy label handling method.

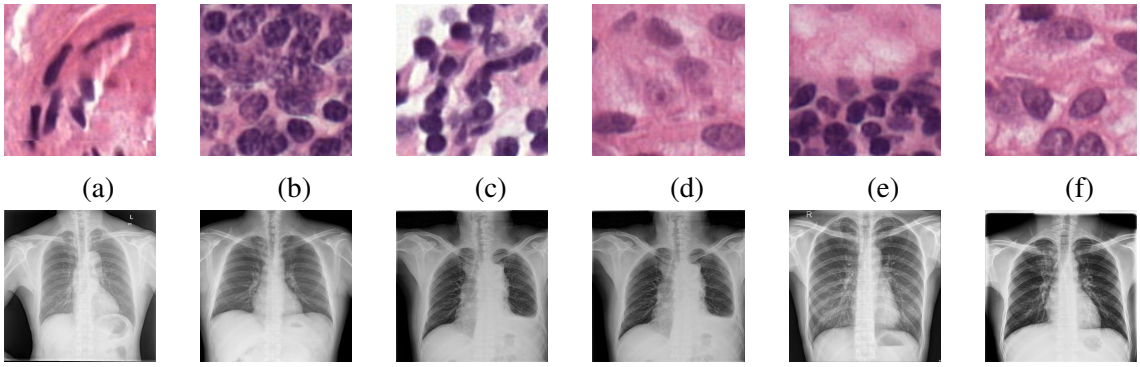


Figure 5.2: Sample images from the Camelyon17 (first row): (a-c) Healthy samples, and (d-f) Tumor samples. TBX11K (second row): (g-h) Healthy samples, (i-j) Sick & Non-TB samples, and (k-l) TB samples.

Table 5.2: Dataset description for Camelyon17 (Bandi *et al.*, 2019) and TBX11K (Liu *et al.*, 2020). Test set GT is not available for TBX11K.

	Camelyon17			**TBX11K**			
Splits/Classes	Tumor	Healthy	Total	TB	Sick & Non-TB	Healthy	Total
Train	37135	37498	74633	759	3001	3129	6889
Val	7500	7500	15000	355	799	933	2087
Test	20004	20001	40005	-	-	-	3302

5.3.1 Projection Loss

We denote the d-dimension class vectors as $\mathbf{c}_i$ such that $i \in [0, C-1]$, where C is the total number of classes. Also, let y_i represents the label of class i. Let $\boldsymbol{a}_k$ be the activation obtained from the last layer of a CNN for the sample $\mathbf{X}_k$. We define the projection of $\boldsymbol{a}_k$ on $\mathbf{c}_i$ as:

$$p_{k,i} = \frac{\boldsymbol{a}_k^T \mathbf{c}_i}{||\mathbf{c}_i||_2}. \tag{5.1}$$

In an ideal scenario, if $\mathbf{X}_k \in y_i$, then $p_{k,i} = 1$ and if $\mathbf{X}_k \in y_{j \neq i}$, then $p_{k,i} = 0$. The above interpretation can be modeled through a distance d as:

$$d(\boldsymbol{a}_k, \mathbf{c}_i) = \left|\left| \frac{\boldsymbol{a}_k^T \mathbf{c}_i}{||\mathbf{c}_i||_2} - 1 \right|\right|_2^2, \tag{5.2}$$

$$= ||p_{k,i} - 1||_2^2 . \tag{5.3}$$

The distance $d(\boldsymbol{a}_k, \mathbf{c}_i)$ should be ideally equal to 0, if $\mathbf{X}_k \in y_i$ and should be 1 otherwise. Hence, $d(\boldsymbol{a}_k, \mathbf{c}_i)$ should be minimum for $\mathbf{X}_k \in y_i$ and maximum otherwise. The conditional probability $P(y_i|\mathbf{X}_k; \boldsymbol{w}, \mathbf{c}_i)$ is represented in terms of the distance $d(\boldsymbol{a}_k, \mathbf{c}_i)$

as:

$$P(y_i|\mathbf{X}_k;\boldsymbol{w},\mathbf{c}_i)=\frac{e^{-d(\boldsymbol{a}_k,\mathbf{c}_i)}}{\sum_{i=0}^{C-1}e^{-d(\boldsymbol{a}_k,\mathbf{c}_i)}}, \tag{5.4}$$

where the activation vectors $\boldsymbol{a}_k$s are functions of weights $\boldsymbol{w}$. The maximization of $P(y_i|\mathbf{X}_k;\boldsymbol{w},\mathbf{c}_i)$ demands the maximization of the numerator in (5.4), which results in the minimization of $d(\boldsymbol{a}_k,\mathbf{c}_i)$. Without loss of generality, we introduce a variable β in (5.4), resulting in:

$$P(y_i|\mathbf{X}_k;\boldsymbol{w},\mathbf{c}_i)=\frac{e^{-\beta d(\boldsymbol{a}_k,\mathbf{c}_i)}}{\sum_{i=0}^{C-1}e^{-\beta d(\boldsymbol{a}_k,\mathbf{c}_i)}}, \tag{5.5}$$

where β (Yang *et al.*, 2018) acts as a scaling factor. The loss can then be formulated as

$$\mathcal{L}'(\boldsymbol{w},\mathbf{c})=-\sum_{i=0}^{C-1}y_i\left[log(P(y_i|\mathbf{X}_k;\boldsymbol{w},\boldsymbol{c}_i)\right] \tag{5.6}$$

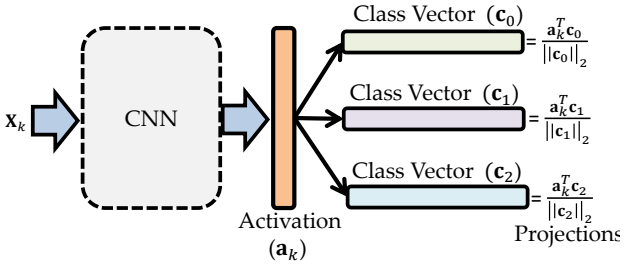


Figure 5.3: Projections of the activation on the class vectors.

Orthogonality of the class vectors $\mathbf{c}_i$

Apart from maximizing the projections of the sample on the respective class vectors, we also induce a orthogonality constraint on the class vectors, i.e., $\mathbf{c}_i^T\mathbf{c}_{j\neq i}=0$. For this, we introduce a regularization term $\mathcal{L}_{orth}(\cdot)$ given by

$$\mathcal{L}_{orth}(\mathbf{c})=\lambda\sum_{j=i+1}^{C-1}\sum_{i=0}^{C-2}||\mathbf{c}_i^T\mathbf{c}_j||_2^2. \tag{5.7}$$

This constraint helps in a better separation of classes. Combining (5.6) and (5.7), we obtain the overall loss function given by

$$\begin{aligned}\mathcal{L}_{PRL}(\boldsymbol{w},\mathbf{c})&=-\sum_{i=0}^{C-1}y_i\left[log(P(y_i|\mathbf{X}_k;\boldsymbol{w},\boldsymbol{c}_i)\right]+\lambda\sum_{j=i+1}^{C-1}\sum_{i=0}^{C-2}||\mathbf{c}_i^T\mathbf{c}_j||_2^2,\\ \mathcal{L}_{PRL}(\boldsymbol{w},\mathbf{c})&=\mathcal{L}'(\boldsymbol{w},\mathbf{c})+\lambda\mathcal{L}_{orth}(\mathbf{c}).\end{aligned} \tag{5.8}$$

Intuitively, the projection loss helps to maximize the projection of the learned activations on the learnable class vectors and also attempts to induce the orthogonality among the class vectors. The update equations for $\boldsymbol{w}$ and $\mathbf{c}$ are obtained from (5.8) as follows:

$$\boldsymbol{w} = \boldsymbol{w} - \alpha \frac{\partial \mathcal{L}'}{\partial \boldsymbol{a}} \frac{\partial \boldsymbol{a}}{\partial \boldsymbol{w}}, \tag{5.9}$$

$$\mathbf{c}_i = \mathbf{c}_i - \alpha \left(\frac{\partial \mathcal{L}'}{\partial \mathbf{c}_i} + \lambda \frac{\partial \mathcal{L}_{orth}}{\partial \mathbf{c}_i} \right), \tag{5.10}$$

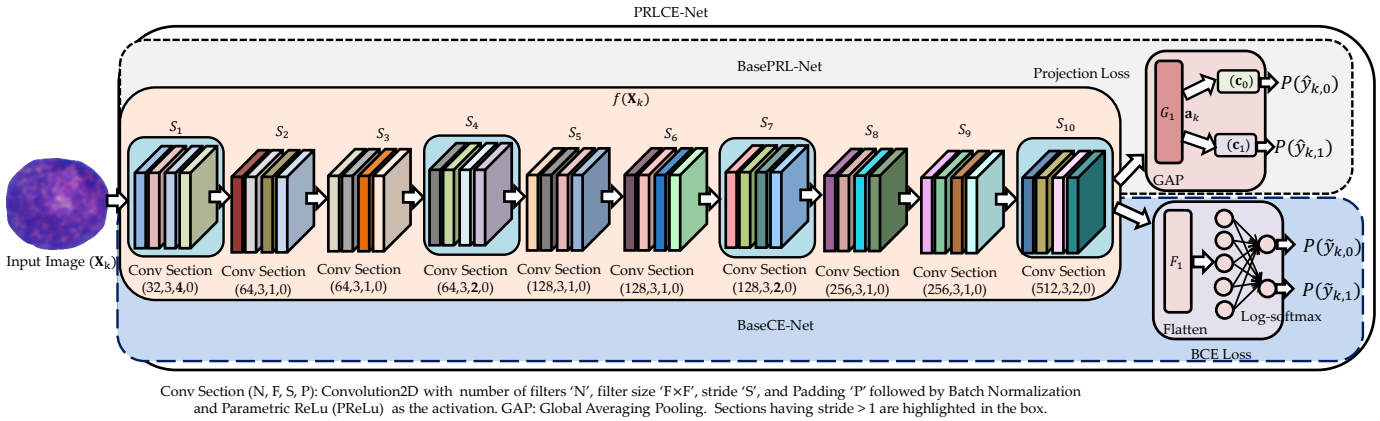
where α is the learning rate. Hence, the class vectors ($\mathbf{c}_i$) are updated through both the terms.

5.3.2 BaseCE-Net

As a starting point, we design a custom CNN classification network instead of using any existing pretrained architecture. The network consists of ten *Conv Sections*, where each *Conv Section* consists of a combination of 2D convolution filters, batch normalization, and parametric ReLu (PReLu) as an activation function. There is no max-pooling in the network; instead, stride$\geq$2 is used to achieve spatial size reduction. This also helps the network to learn the required pooling operation. After the last *Conv Section*, the output features are given to a global averaging pooling (GAP) layer and finally passed to the output softmax layer. We name this architecture *BaseCE-Net Network* because we use binary cross-entropy loss function to train it.

5.3.3 BasePRL Net

Next, we replace the BCE loss with the proposed projection loss ($\mathcal{L}_{PRL}$) in the BaseCE-Net network and name this architecture as *BasePRL-Net*. As we have a binary class dataset, we initialize the two-class vectors $\mathbf{c}_0$ and $\mathbf{c}_1$ for class 0 (healthy) and 1 (cancer), respectively. The output of GAP is projected on the class vectors $\mathbf{c}_0$ and $\mathbf{c}_1$, where (5.8) is used as the objective function. The output of GAP and the class vectors are of the same dimension. During the training, $\mathbf{c}_0$ and $\mathbf{c}_1$ are also updated along with the network parameters according to (5.10).



Conv Section (N, F, S, P): Convolution2D with number of filters 'N', filter size 'F×F', stride 'S', and Padding 'P' followed by Batch Normalization and Parametric ReLu (PReLu) as the activation. GAP: Global Averaging Pooling. Sections having stride > 1 are highlighted in the box.

Figure 5.4: PRLCE-Net.
It uses CE and projection loss in the two branches.

5.3.4 PRLCE-Net

We design a hybrid architecture that uses both BCE loss and Projection Loss (PRL). However, instead of directly adding BCE loss and PRL, we combine the BaseCE-Net and BasePRL-Net such that the new architecture has shared convolution filters (*conv sections)* for both the objectives. After that, the network is fragmented into two branches. One branch is flattening (reshaping) the input features to use with the BCE loss, while the other branch is applying GAP on the incoming features and then using the PRL on the resultant output. Different pooling layers help capture different structures of the data that helps to utilize different information by each branch. The *PRLCE-Net* is shown in Fig. 5.4 and its loss function is given by

$$\mathcal{L}_{PR-CE} = \beta_1 \mathcal{L}_{PR} + \beta_2 \mathcal{L}_{CE} \tag{5.11}$$

As observed from Fig. 5.4, the predictions for the two loss functions, $\mathcal{L}_{PR}$ and $\mathcal{L}_{CE}$, are obtained differently. During the backpropagation, shared convolutional filters (feature extraction filters) will be updated with two different objective functions. Since both the loss functions are attempting to perform a common task, albeit with different approaches, the resultant weights will lead to more robust feature extraction and better final performance. This claim has been verified in Sec 5.4.1. One of the advantages of *PRLCE-Net* is that it provides scope for inducing robustness. As discussed in Sec 5.3.5 and Sec 5.3.6, we introduce ensembling and noise handling capability in *PRLCE-Net Network* to enhance its performance.

5.3.5 Label Noise Handling

Medical datasets often suffer from the problem of noisy labels due to several reasons such as decision ambiguity, variations in the acquisition process, etc. Unlike the natural images, samples with noisy labels in medical datasets can not be identified manually, even for small datasets, due to the inter-class visual similarity. This makes the problem more challenging in medical data. At the same time, handling the noisy labels appropriately may improve the model's performance. The label noise in the medical images have been addressed through label cleaning, noise layer, loss functions, data re-weighting, and training procedures (Karimi *et al.*, 2020). Among the label cleaning based approaches, Veit *et al.* (2017) trained two CNNs, one using the clean data for learning to denoise data to be used by the another CNN. However, this approach also requires clean data, which may not be available in some scenarios. Another approach that requires clean data is discussed in Lee *et al.* (2018). In this work attention utilizing encoder is used to generate the embedding vector of each class. In parallel, another encoder is used to generate the query image's embedding vector. The similarity between query and reference embedding vector is used to predict the samples with the label noise. Han *et al.* (2018*a*) eliminates the necessity of clean data. It maintains two CNNs, and the clean labels identified by one network based on loss criteria are used to update the peer CNN parameters. The proposed approach also does not require a clean dataset. In contrast to Han *et al.* (2018*a*), our approach utilizes only a single CNN and two different loss functions. We have also used different label noise identification criteria that are not based on loss function but on the predictions of two different branches. The training procedure is also different. While the training of Han *et al.* (2018*a*) consists of only one stage, our approach is based on two-steps training. The proposed technique does not require any prior knowledge of class labels and is unsupervised from this perspective. Consider the *PRLCE-Net* trained with $\{\mathbf{X}_k, y_k\}_{k=0}^{N-1}$ samples for T epochs. At any given epoch, the label predicted by PRL branch for $\mathbf{X}_k$ is given by

$$\hat{y}_k = \underset{l \in \{0,1\}}{\arg\max}\, P(\hat{y}_{k,l}). \tag{5.12}$$

This can be easily extended to C classes, wherein there will be C class vectors with $l \in \{0, 1, ..C-1\}$. Similarly, for the CE branch, the prediction for $\mathbf{X}_k$ is

$$\tilde{y}_k = \underset{l \in \{0,1\}}{\arg\max}\, P(\tilde{y}_{k,l}). \tag{5.13}$$

Let $P(\hat{y}_k)$ and $P(\tilde{y}_k)$ denote the probability scores of the predicted labels. As the training of a CNN progresses, i.e., as more numbers of epochs are completed, the networks' performance on the training set improves. Accordingly, the performance on the validation set will improve if there is no overfitting. Let the total epochs T be divided into two sets: $\{0, 1, ..P-1\}$ and $\{P, P+1..T-1\}$. The number P is chosen to be sufficiently large for the convergence of the training process. This parameters will also be updated such that both the PRL branch and CE branch yield an optimal performance. We hypothesize that even though the two branches use different loss functions, both will give approximately the same performance, at least on the non-noisy training data. After the $(P-1)^{th}$ epoch, there are three possible scenarios for a training sample $\mathbf{X}_k$:

- (S_1) $\hat{y}_k = \tilde{y}_k$ i.e., predictions of the branches are same.
- (S_2) $\hat{y}_k \neq \tilde{y}_k$ with $P(\hat{y}_k) \geq \tau$ **and/or** $P(\tilde{y}_k) \geq \tau$ i.e both branches are predicting a different class but at least one of them has the prediction probability greater than or equal to the predefined threshold τ.
- (S_3) $\hat{y}_k \neq \tilde{y}_k$ with $P(\hat{y}_k) < \tau$ **and** $P(\tilde{y}_k) < \tau$ i.e., both the branches have different predictions and their prediction probabilities are also lower than the predefined threshold τ.

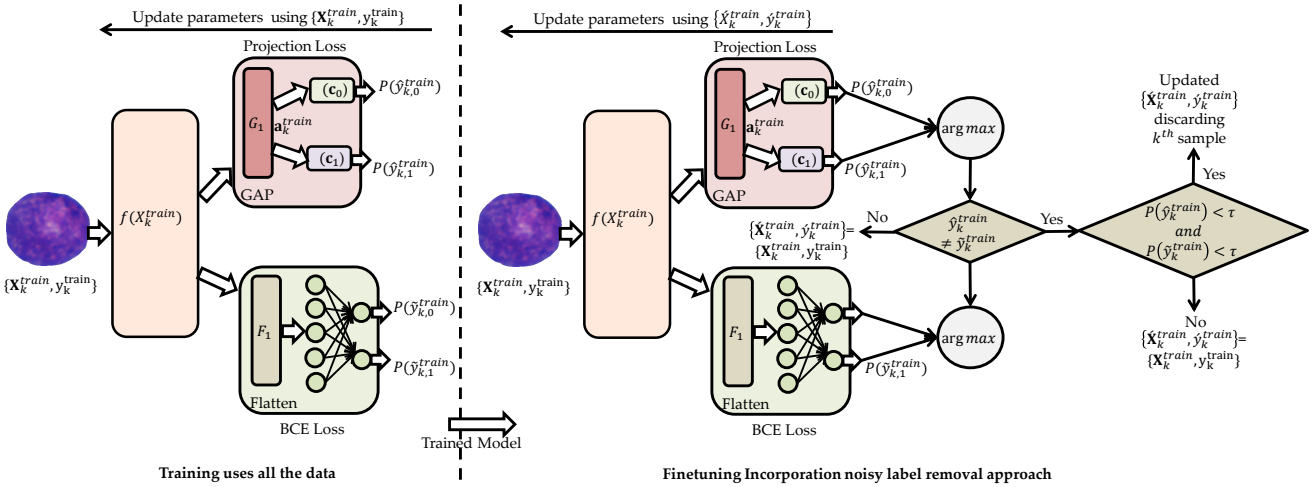


Figure 5.5: The training process of the *PRLCE-Net* with the incorporation of the *sample discarding* as a label noise handling approach. The complete training occurs in two phases: first, the network is trained with all the training data and later, it is fine-tuned with *sample discarding* approach.

For (S_1), both the branches are in agreement and we assume that the label is noise-free. For (S_2), although the predictions are different, either one of the two branches is confident about its prediction. This difference may arise due to different loss functions in each branch. However, since one of the branches is certain about its decision, we assume that the sample's label is correct. For (S_3), both the branches are in disagreement, and also none of them is confident about its prediction. If we choose $\tau = 0.60$, then the prediction probabilities will be in the range of $(0.50, 0.60]$, which is very low. This shows a low confidence of both the branches along with the disagreement. We assume that such samples have noisy labels. Samples that meet (S_3) and hence, have noisy labels, can be handled with the below two approaches.

(A_1) **Label Flipping:** If we have a binary class dataset, each sample will belong to either class 0 or class 1. Therefore, when we suspect a sample to be having an incorrect label, we flip its label to that of the other class, i.e.,

$$y_k = 1 - y_k \quad \text{if } \hat{y}_k \neq \tilde{y}_k \text{ with } P(\hat{y}_k) < \tau \text{ and } P(\tilde{y}_k) < \tau. \tag{5.14}$$

Hence, true label (y_k) is changed if (S_3) is satisfied by the predicted labels ($\hat{y}_k$ and $\tilde{y}_k$). For a C class dataset, there are $C - 1$ possibilities with which k^{th} true label can be flipped, out of which only one scenario is correct. Hence, we require to take a computationally-expensive iterative approach.

(A_2) **Sample Discarding:** Another approach to handle (S_3) is sample discarding. If any sample satisfies (S_3), we remove that sample from the training set. In this way, we may have a clean dataset with no samples having noisy labels. Also, since no flipping is involved, this approach is independent of the number of classes and has the same computational cost irrespective of the classes.

Sample discarding reduces the number of training samples, which is not the case with *label flipping*. Once we decide to opt for either of the two approaches to tackle (S_3), we fine-tune the model during $\{P, P+1..T-1\}$ epochs with the modified set $\{\mathbf{X}_k, y_k\}$. This fine-tuning with possibly clean data will try to adjust the decision boundary that was obtained earlier until P epochs with the noisy data. With *label flipping*, we have $C - 1$ final models for a C class dataset. This has a high computational complexity, especially, if C is very large. However, for $C = 2$, we have only one final model. On the

other hand, there will always be a single final model with *sample discarding* approach, irrespective of the number of classes.

5.3.6 Coupling Classifier

Consider *PRLCE-Net* (Fig. 5.4). As the network contains the two branches, there will be two possible scenarios for any test sample $\mathbf{X}_k^{test}$:

1. Both branches have same predictions i.e $\hat{y}_k^{test} = \tilde{y}_k^{test}$.
2. Both branches yield different outcomes, i.e., $\hat{y}_k^{test} \neq \tilde{y}_k^{test}$.

For case (1), we consider the prediction of either branch to be the final prediction ($\acute{y}_k^{test}$) or $\acute{y}_k^{test} = \hat{y}_k^{test} = \tilde{y}_k^{test}$. However, in case of a conflict, the prediction of any branch could be correct. Hence, choosing the label of any one branch will make us biased towards that particular branch. It also gives uncertainty in choosing the right prediction. Hence, it is ideal to output a single label instead of two different predictions. We propose a solution to this problem in the form of a *coupling classifier.* This couples the two branches and helps to obtain a unique prediction from the *PRLCE-Net*.

Training of the Coupling Classifier Once the training of the *PRLCE-Net* is completed, we find all the training samples on which both the branches are yielding correct and same predictions.

$$\acute{k} = \{k | \hat{y}_k^{train} = \tilde{y}_k^{train} = y_k^{train}, \text{ and } 0 \leq k \leq N-1\} \tag{5.15}$$

We use $\{\mathbf{X}_{\acute{k}}^{train}, y_{\acute{k}}^{train}\}$ to train the coupling classifier. However, instead of using flattening or GAP, $\{\mathbf{X}_{\acute{k}}^{train}\}$ is passed through a spectral averaging layer. Hence, $\{S\left(f\left(\mathbf{X}_{\acute{k}}^{train}\right)\right), y_{\acute{k}}^{train}\}$ are used for training the *coupling classifier*. The use of a different type of pooling provides a different set of features to the *coupling classifier* as compared to the ones obtained from the other two branches. This helps the classifier in making a better decision.

Testing of the Coupling Classifier During the testing, test samples having distinct predictions by both the classifiers are predicted by the *coupling classifier*. Let $\mathbf{X}_k^{test}$ be the sample for which we have a conflicting decision. The $f(\mathbf{X}_k^{test})$ is then passed to

the spectral averaging layer and the resultant output is given to the *coupling classifier* to predict its label. Again, there are two possibilities to use the labels predicted by the *coupling classifier*.

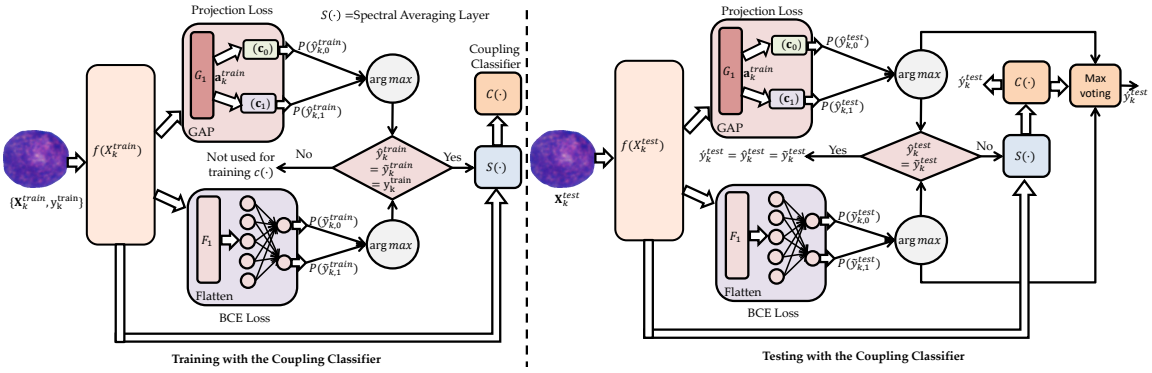


Figure 5.6: Training and Testing with Coupling Classifier. Features to the *coupling classifier* are fed through the spectral averaging layer.

(D_1) **Stand-Alone Decision:** In this case, we consider the label of the *coupling classifier* to be the final prediction, i.e., $\acute{y}_k^{test} = C\left(S(f(\mathbf{X}_k^{test})\right)$. Through this, an independent decision is obtained because we ignore the predictions of both the branches.

(D_2) **Ensemble Decision:** Another possibility is to consider the decisions of both the branches along with the *coupling classifier*. For example, we can obtain the final decision as:

$$\acute{y}_k^{test} = g\left(\hat{y}_k^{test}, \tilde{y}_k^{test}, C\left(S(f(\mathbf{X}_k^{test}))\right)\right), \tag{5.16}$$

where $g(\cdot)$ is some ensembling function. If we consider $g(\cdot)$ to be the majority voting, the final label is same as that predicted by two or more classifiers. This criterion will always work for the binary class datasets but fail in the case of multi-class datasets if each classifier's prediction is different.

The training and testing of the *custom classifier* are also summarized in Fig. 5.6. Also, for the case of binary class datasets, both the testing approaches will lead to the same results.

To summarize, we start with *PRLCE-Net* and include *label flipping* or *sample discarding* to address the label noise problem. Finally, we include the *coupling classifier* to make the final decision without any conflict. This process is also summarized in

Fig. 5.7. Specifically, the training and testing process of *PRLCE-Net* with *sample discarding* and *coupling classifier* is elaborated in the below algorithm.

Algorithm: Training and Testing of *PRLCE-Net+SD+CC* (PRLCE-Net with sample discarding and coupling classifier)

Input: *PRLCE-Net network*, Epoch sets: $\{0, 1, ..P-1\}$ and

$\{P, P+1..T-1\}$, and coupling classifier: $C(\cdot)$

Output: Final Predictions: $\{\acute{y}_k^{test}\}$

Data: Train set: $\{\mathbf{X}_k^{train}, y_k^{train}\}$, validation set: $\{\mathbf{X}_k^{val}, y_k^{val}\}$, test set: $\{\mathbf{X}_k^{test}\}$

1 Initial Training

while *epoch* $(e) \in \{0, 1, ..P-1\}$ **do**

  Update the parameters of *PRLCE-Net network* (Fig. 5.4) using (5.11)

2 Finetuning incorporating Noisy Labels Handling Approach

while *epoch* $(e) \in \{P, P+1..T-1\}$ **do**

  Identify noisy labels' samples using (S_3)

  Use (A_2) to update the training set to $\{\acute{\mathbf{X}}_k^{train}, \acute{y}_k^{train}\}$

  Update the parameters of *PRLCE-Net network* using the $\{\acute{\mathbf{X}}_k^{train}, \acute{y}_k^{train}\}$

3 Train the Coupling Classifier $C(\cdot)$

for $\{\boldsymbol{X}_k^{train}\}$ **do**

$$\acute{k} = \{k | \hat{y}_k^{train} = \hat{y}_k^{train} = y_k^{train}\}$$

$$C = \phi\left(S\left(f\{\mathbf{X}_{\acute{k}}^{train}, y_{\acute{k}}^{train}\}\right)\right)$$

4 Testing on $\{\mathbf{X}_k^{test}\}$

Predict $\hat{y}_k^{test}$ and $\tilde{y}_k^{test}$ using *PRLCE-Net network*

if $\hat{y}_k^{test} \neq \tilde{y}_k^{test}$ **then**

$$\acute{y}_k^{test} = C\left(S(f(\mathbf{X}_k^{test})\right)$$

else

$$\acute{y}_k^{test} = \hat{y}_k^{test} = \tilde{y}_k^{test}$$

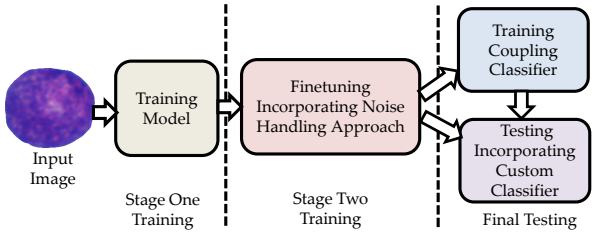


Figure 5.7: Complete training and testing strategy incorporating label noise handling and coupling classifier.

5.4 Results and Discussion

5.4.1 Multiple Myeloma (MM) Dataset

In this section, we will validate all the proposed methodologies. We will also compare the proposed architectures with the existing networks. First, we will discuss the results on MM dataset.

Training and Testing details

Stochastic gradient descent with a momentum of 0.9 is used as an optimizer. We have also used a weight decay of 0.01 and a batch size of 64. The training is carried out for a total of 150 epochs, starting from a learning rate of 0.001. The learning rate is reduced to one-tenth of its present value after 80^{th}, 120^{th}, and 140^{th} epoch. The parameters β and γ in (5.8) are set to 2 and 1, respectively, and τ is set to 0.6 in (S_2) and (S_3). Also, the values of β_1 and β_2 in (5.11) are 1. PyTorch deep learning library is used for the implementation and GeForce RTX 2080 Ti is used to accelerate the training and testing processes. This strategy is used for the training of *BaseCE-Net*, *BasePRL-Net*, and *PRLCE-Net*. For training *PRLCE-Net+SD* or *PRLCE-Net+LF*, we have fine-tuned *PRLCE-Net* for another 35 epochs with an initial learning rate of 0.00001 using the Adam optimizer after incorporating *sample discarding* or *label flipping*. The current value of the learning rate is multiplied by 0.1 after 10^{th} and 30^{th} epoch. Results of the fine-tuning of *PRLCE-Net* with Adam optimizer and without including sample discarding or label flipping is provided in the supplementary. We have also used kernel SVM with radial basis function (RBF) as the coupling classifier's kernel.

We used five-fold cross-validation for the training and testing of the architectures. One fold is used for validation at a given instance and the remaining four folds are used for training. Thus, we obtain five trained models for any particular architecture. We use Model-n to denote the architecture trained using the n^{th} fold as the validation set and remaining folds as the training sets. The best model achieved according to the validation performance is used for final analysis. We used *majority voting* on the predictions obtained from the five models (Model-1 to Model-5) to arrive at the final decision on test samples.

To augment the training data, we used random rotation in $[0, 360)$, and horizontal and vertical flips (Huang *et al.*, 2017; He *et al.*, 2016; Springenberg *et al.*, 2015). We also normalized the dataset through mean and standard deviation before feeding it to the architectures. Oversampling is also used to handle class imbalance, if any.

For performance comparison, we used weighted F_1 score as well as the individual F_1 scores for each class. We also used balanced accuracy, which is the average of recall and specificity. Class-wise F_1 score helps highlight the performance on the individual classes, whereas weighted F_1 score and balanced accuracy take the class imbalance into consideration. We also computed the area under the curve (AUC) to highlight different thresholding effects. We utilized the accuracy metric to analyze the subject-level performance.

Ablation Study

We perform an ablation study to highlight the significance of all the proposed techniques. We start with the *BaseCE-Net* and subsequently, analyze the contribution of each additional component. As a notation, *PRLCE-Net+SD (CE)* denotes the *CE* branch of *PRLCE-Net* trained with incorporating *Sample Discarding*. Similarly, *PRLCE-Net+SD (PRL)* represents the *PRL* branch of the same network. Also, *PRLCE-Net+SD+CC* denotes *PRLCE-Net* augmented by *Sample Discarding* and *Coupling Classifier*. Other notations can be followed on similar lines.

As compared to *BaseCE-Net*, *BasePRL-Net* performs better on three models (Model-1, Model-4, and Model-5), while the latter leads on the remaining two models. Overall, with ensembling (majority voting), *BasePRL-Net* leads *BaseCE-Net*. This trend is seen on all the four metrics. Specifically, *BasePRL-Net* performs better than *BaseCE-Net* with a margin of 0.19% on the healthy class F_1 score (NF_1 score), 0.35% on the cancer class F_1 score (NF_1 score), 0.29% on the weighted F_1 score (NF_1 score), and 0.27% on the balanced accuracy, which shows the contribution of the projection loss.

PRLCE-Net consists of CE loss and projection loss in different branches. This combination helps boost the performance of each branch as compared to the individual networks (*BaseCE-Net* and *BasePRL-Net*). For example, with ensembling, as compared to *BaseCE-Net*, *PRLCE-Net (CE)* gains by 0.55%, 0.76%, 0.66%, 0.66% on NF_1 score,

Table 5.3: Results of Healthy Class F_1 Score and Cancer Class F_1 Score with all the proposed methods. These results are obtained on 40440 test images. Best results are highlighted in bold. Same results are depicted in italics. Network trained with fold-n (n=1,2..5) as the validation set is represented as Model-n.

Healthy Class F_1 Score						
Architectures/Models	Model-1	Model-2	Model-3	Model-4	Model-5	Majority Voting
BaseCE-Net	0.9212	0.9391	0.9275	0.9321	0.9309	0.9374
BasePRL-Net	0.9285	0.9310	0.9253	0.9378	0.9316	0.9393
PRLCE-Net (CE)	0.9312	0.9435	0.9304	0.9388	0.9368	0.9429
PRLCE-Net (PRL)	0.9314	0.9424	0.9307	0.9389	0.9371	0.9424
PRLCE-Net+CC	0.9317	0.9439	0.9308	0.9394	0.9378	0.9435
PRLCE-Net+LF (CE)	0.9356	0.9360	0.9276	0.9458	0.9408	0.9440
PRLCE-Net+LF (PRL)	0.9309	0.9325	0.9253	0.9443	0.9372	0.9404
PRLCE-Net+LF (CC)	0.9357	0.9361	0.9278	0.9463	0.9410	0.9443
PRLCE-Net+SD (CE)	**0.9404**	0.9435	0.9359	0.9437	0.9414	0.9481
PRLCE-Net+SD (PRL)	0.9367	0.9409	0.9356	0.9423	0.9384	0.9457
PRLCE-Net+SD+CC	0.9403	**0.9436**	**0.9360**	**0.9442**	**0.9416**	**0.9482**
Cancer Class F_1 Score						
Architectures/Models	Model-1	Model-2	Model-3	Model-4	Model-5	Majority Voting
BaseCE-Net	0.9006	0.9274	0.9094	0.9156	0.9143	0.9233
BasePRL-Net	0.9121	0.9187	0.9073	0.9251	0.9167	0.9268
PRLCE-Net (CE)	0.9160	0.9320	0.9140	0.9249	0.9238	0.9309
PRLCE-Net (PRL)	0.9161	0.9305	0.9145	0.9251	0.9240	0.9303
PRLCE-Net+CC	0.9167	0.9327	0.9148	0.9259	0.9252	0.9318
PRLCE-Net+LF (CE)	0.9223	0.9224	0.9104	0.9351	0.9291	0.9327
PRLCE-Net+LF (PRL)	0.9150	0.9171	0.9068	0.9329	0.9238	0.9275
PRLCE-Net+LF (CC)	0.9224	0.9225	0.9107	0.9357	0.9295	0.9331
PRLCE-Net+SD (CE)	**0.9291**	*0.9321*	0.9223	0.9320	0.9301	0.9381
PRLCE-Net+SD (PRL)	0.9235	0.9283	0.9217	0.9300	0.9258	0.9346
PRLCE-Net+SD+CC	0.9289	*0.9321*	**0.9225**	**0.9328**	**0.9304**	**0.9383**

CF_1 score, WF_1 score and balanced accuracy, respectively. Similar trends are seen on individual models (Model-1 to Model-5). Similarly, *PRLCE-Net (PRL)* performs better than *BasePRL-Net* on each model (Model-1 to Model-5), as well as with ensembling. Overall, the former is performing better than the latter by a margin of 0.31% (NF_1 score), 0.35% (CF_1 score), 0.33% (WF_1 score), and 0.32% (balanced accuracy).

Effect of Label Flipping

The effect of label flipping ((A_1)) in handling label noise is shown in Table 5.3 and Table 5.4. Comparing *PRLCE-Net+LF (CE)* with *PRLCE-Net (CE)*, an improvement is seen on three models, and finally, after majority voting, the gain is 0.11%, 0.18%, 0.14%, 0.15% for NF_1 score, CF_1 score, WF_1 score, and balanced accuracy, respec-

Table 5.4: Results in terms of Weighted F_1 Score and Balanced Accuracy with all the proposed methods. These results are obtained on 40440 test images. Best results are highlighted in bold. Network trained with fold-n (n=1,2..5) as the validation set is represented as Model-n.

Weighted F_1 Score						
Architectures/Models	Model-1	Model-2	Model-3	Model-4	Model-5	Majority Voting
BaseCE-Net	0.9113	0.9335	0.9188	0.9242	0.9230	0.9306
BasePRL-Net	0.9206	0.9251	0.9167	0.9317	0.9244	0.9333
PRLCE-Net (CE)	0.9239	0.9380	0.9226	0.9321	0.9306	0.9372
PRLCE-Net (PRL)	0.9241	0.9367	0.9229	0.9323	0.9308	0.9366
PRLCE-Net+CC	0.9245	0.9385	0.9231	0.9329	0.9318	0.9379
PRLCE-Net+LF (CE)	0.9293	0.9295	0.9193	0.9407	0.9352	0.9386
PRLCE-Net+LF (PRL)	0.9233	0.9251	0.9165	0.9389	0.9308	0.9342
PRLCE-Net+LF (CC)	0.9293	0.9296	0.9196	0.9413	0.9355	0.9390
PRLCE-Net+SD (CE)	**0.9350**	0.9380	0.9294	0.9381	0.9360	0.9433
PRLCE-Net+SD (PRL)	0.9304	0.9348	0.9289	0.9364	0.9323	0.9404
PRLCE-Net+SD+CC	0.9348	**0.9381**	**0.9295**	**0.9388**	**0.9362**	**0.9435**
Balanced Accuracy						
Architectures/Models	Model-1	Model-2	Model-3	Model-4	Model-5	Majority Voting
BaseCE-Net	0.9088	0.9317	0.9164	0.9218	0.9206	0.9285
BasePRL-Net	0.9184	0.9235	0.9144	0.9298	0.9223	0.9314
PRLCE-Net (CE)	0.9218	0.9360	0.9203	0.9299	0.9286	0.9351
PRLCE-Net (PRL)	0.9220	0.9347	0.9206	0.9301	0.9289	0.9346
PRLCE-Net+CC	0.9224	0.9366	0.9209	0.9307	0.9298	0.9359
PRLCE-Net+LF (CE)	0.9273	0.9274	0.9170	0.9388	0.9333	0.9366
PRLCE-Net+LF (PRL)	0.9211	0.9229	0.9141	0.9369	0.9287	0.9321
PRLCE-Net+LF (CC)	0.9274	0.9275	0.9173	0.9394	0.9337	0.9371
PRLCE-Net+SD (CE)	**0.9332**	0.9361	0.9273	0.9361	0.9342	0.9415
PRLCE-Net+SD (PRL)	0.9284	0.9328	0.9269	0.9343	0.9304	0.9384
PRLCE-Net+SD+CC	0.9331	**0.9362**	**0.9275**	**0.9368**	**0.9345**	**0.9417**

tively. However, an improvement is observed with *PRLCE-Net+LF (PRL)* as compared to *PRLCE-Net (CE)* only for one model (Model-4). On the remaining models or with majority voting, there is no gain on either of the metrics. Overall, there is a reduced performance with the inclusion of label flipping.

Effect of Sample Discarding

Next, we analyze the contribution of sample discarding as a noise label handling approach. We compare the performance of *PRLCE-Net+SD* with other methods in Table 5.3 and Table 5.4. On comparing *PRLCE-Net+SD (PRL)* with *PRLCE-Net (PRL)* or *PRLCE-Net+SD (CE)* with *PRLCE-Net (CE)*, we see an enhanced performance on each model as well as with ensembling. For NF_1 score, *PRLCE-Net+SD (CE)* leads

PRLCE-Net (CE) by 0.52%, whereas with *BaseCE-Net* the margin is 1.07%. Similarly, margins, after including sampling discarding, are 0.72% for CF_1 score, 0.61% for WF_1 score, and 0.64% on balanced accuracy in the CE branch. For the PRL branch, the introduction of sample discarding led to a gain of 0.33%, 0.43%, 0.38%, 0.38% in terms of NF_1 score, CF_1 score, WF_1 score, and balanced accuracy, respectively. These values are compared to the results of *PRLCE-Net (PRL)*. The gain is higher in comparison to the *BasePRL-Net*.

Table 5.5: The number of predictions aided by the coupling classifier. These results are obtained on 40440 test images. Network trained with fold-n (n=1,2,...,5) as the validation set is represented as Model-n.

Architectures/Models	Model-1	Model-2	Model-3	Model-4	Model-5
PRLCE-Net+CC	153	145	120	120	154
PRLCE-Net+SD+CC	334	193	135	174	259

Role of Coupling Classifier

It is observed from Tables 5.3-5.4 that the networks' predictions involving the CE and PRL branches are not the same for both the branches, and that there is a conflict in the decision of the two classifiers. Table 5.5 shows the exact number of samples for *PRLCE-Net* and *PRLCE-Net+SD* on which there is a disagreement between the two branches. Without a coupling classifier, there will be an uncertainty in choosing the prediction of either branch. Apart from providing certainty in the final predictions, the coupling classifier also yields somewhat improved performance. Although the improvement is not that significant, the coupling classifier serves its primary purpose of removing the discrepancy.

Summary of the Ablation Study

These results indicate that each proposed component aids in performance enhancement. *PRLCE-Net+SD+CC* is obtained after modifying *BaseCE-Net* by including projection loss, label noise handling, and the coupling classifier. On comparing *BaseCE-Net* with *PRLCE-Net+SD+CC*, we observe an improvement of 1.5% on CF_1 score. This improvement is 1.08% in terms of NF_1 score. Similarly, we see an increment of 1.29%, and 1.32% on WF_1 score, and balanced accuracy, respectively. We also observe the

significance of *PRLCE-Net* because it provides the scope for introducing label noise handling strategy like *sample discarding* and *label flipping*, and ensembling with the *coupling classifier.*

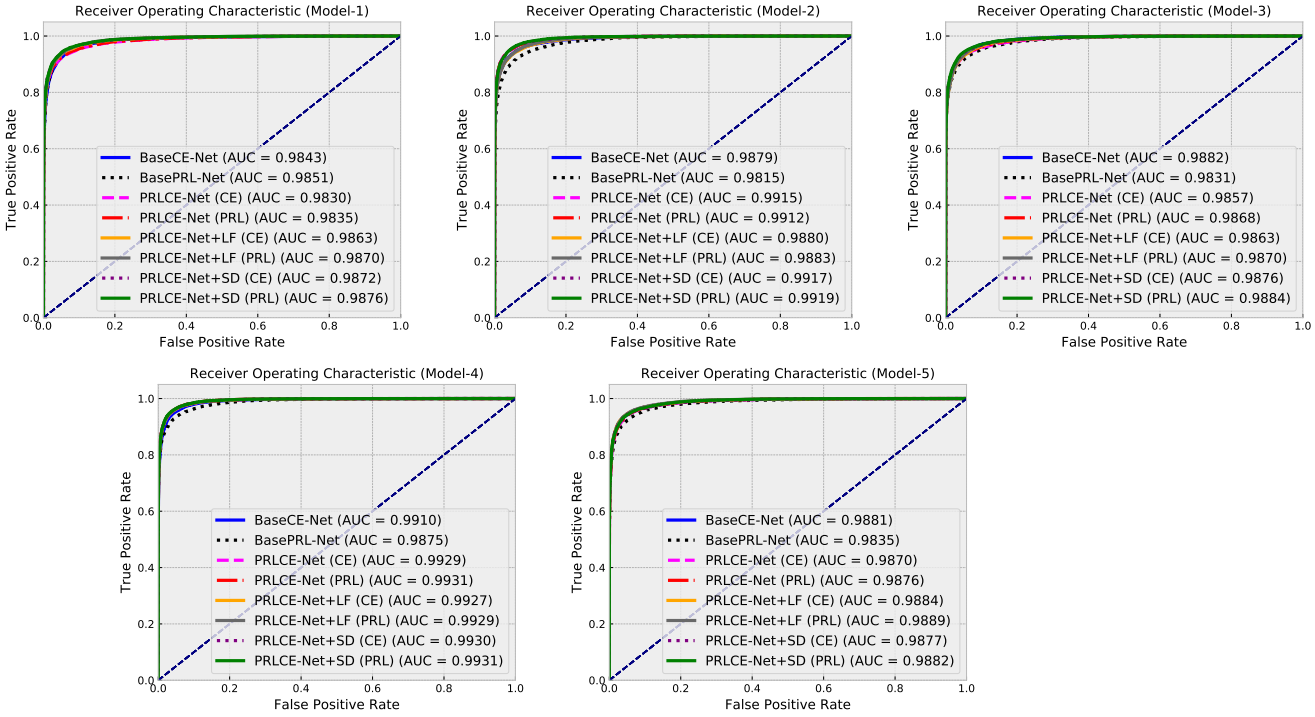
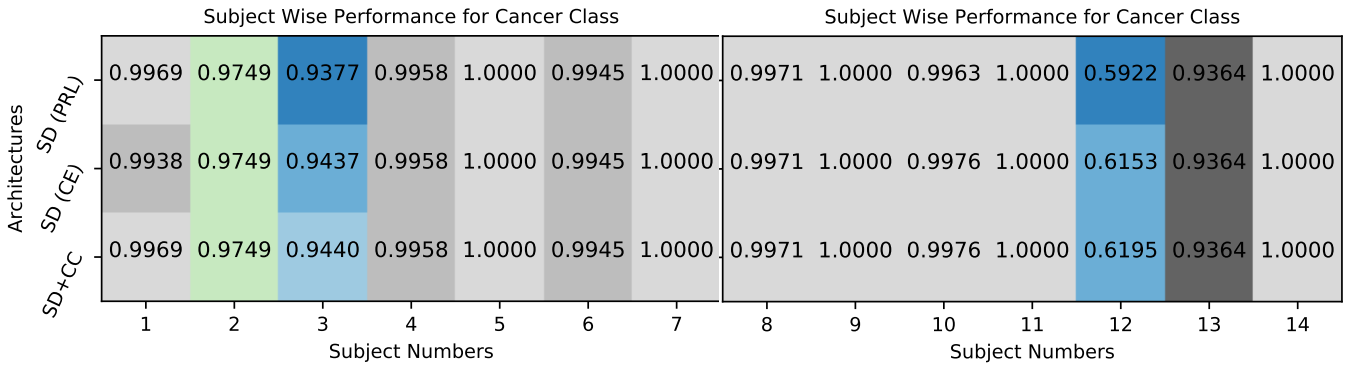


Figure 5.8: Receiver Operating Characteristics and Area Under Curve (AUC) with different architectures. These results are obtained on 40440 test images. The network trained with fold-n (n=1,2,...,5) as the validation set is represented as Model-n.

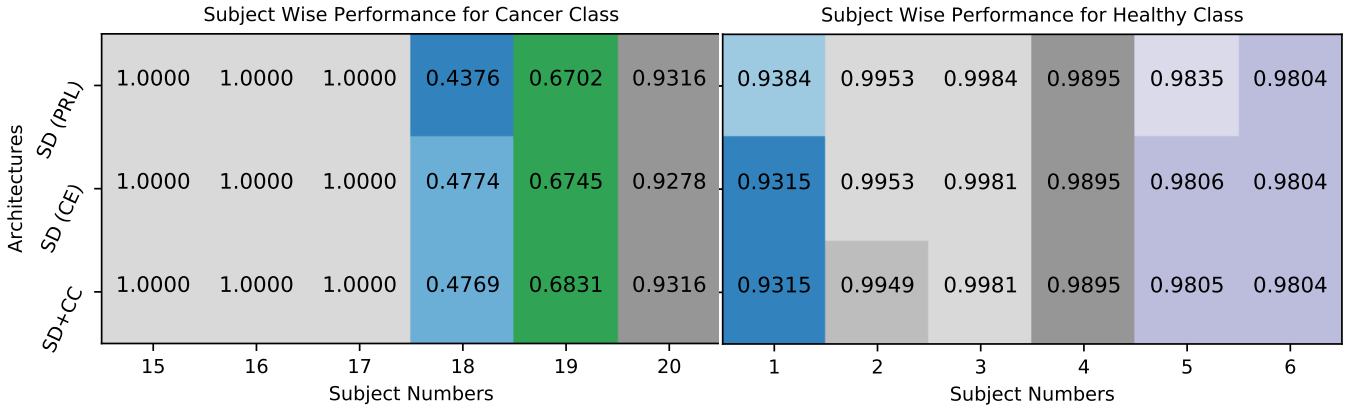
Receiver Operating Characteristics (ROC) and Area Under Curve (AUC)

Next, we analyze ROC and AUC for all the proposed architectures. Results of different models (Model-1 to Model-7) are shown in Fig. 5.8. There is an incremental trend in the AUC from *BaseCE-Net* to *BasePRL-Net*. On all models, except for Model-5, *PRLCE-Net+SD (PRL)* has the highest value of AUC. Same values are obtained for *PRLCE-Net+SD (PRL)* and *PRLCE-Net(PRL)* in Model-4. Also, among the PRL and CE branches, the former has the dominant AUC. Again, on majority of the models, sample discarding yields better AUC compared to label flipping. Also, on some of the models (Model-2 and Model-4), addition of label flipping does not provide improved performance. All architectures are observed to have high AUC that is close to one in some cases.



(a) Cancer class subjects

(b) Cancer class subjects



(c) Cancer class subjects

(d) Healthy class subjects

Figure 5.9: Subject-level performance in terms of accuracy with *PRLCE-Net* and its variants: cancer class subjects (a-c), and healthy class subjects (d)

Subject Level Performance Analysis

Apart from the overall performance on all the test images, another essential aspect in the computer-aided diagnosis is subject-level performance. This is because it will be tested on new test subjects in a realistic environment once the model is deployed. As there might be subject-level variability, the performance may vary at the subject-level. The variations could be due to some noise in the image capturing process or some variations in the slide preparation pipeline, or other related issues. To highlight this issue, we have carried out a subject-level analysis, as shown in Fig. 5.9. The analysis is performed on twenty subjects of the cancer class and five subjects belonging to the healthy class. With *PRLCE-Net+SD+CC*, the network performance is very high on the seventeen subjects of the cancer class, with 100% accuracy on some subjects (subject no. 5,7,9,11,14-17). For the rest of them, the accuracy is close to 1.00, while for some, it varies from $93.13\% - 97.49\%$. On three subjects, the performance is poor. The accuracy on subject numbers 12 and 19 is 61.95% and 68.31%, respectively. On the last remaining subject

(no. 18), the accuracy is only 47.69%. Hence, the classifier performs well on most subjects, but its performance is non-optimal on some of them. This difference in the performance highlights the impact of subject-level variations. This also implies that overall performance is impacted only due to some subjects. Overall, we are able to design a classifier that performs well on most of the subjects (17 out of 20).

On the healthy subjects, the performance is consistently good with minimum performance being 93.15% and a maximum being 99.81%. There is not a significant decline in the performance on any subject in the healthy class as was observed in the cancer class. The standard deviation of the accuracy between the subjects is 0.0244 that again indicates stable inter-subject performance.

Again, a difference is observed in the performance of *PRLCE-Net+SD (CE)* and *PRLCE-Net+SD (PRL)*. The coupling classifier helps in deciding with certainty. It either deflects the outcomes towards one branch or leads to better performance compared to both the branches. Hence, the coupling classifier is also helpful in improving the subject-level performance.

Comparison With Existing Architectures

We have also compared the performance of the proposed methodology with some existing architectures, and some of them have achieved state-of-the-art results on the classification task. These results are depicted in Table 5.6. For training, these architectures are initialized with pre-trained weights on the ImageNet dataset. The input image size is resized to 224×224 (299×299 for Inception-v3) as required by these networks. Apart from WF_1 score and balanced accuracy, we also analyzed the number of parameters and the test compute time of each network. As seen from Table 5.6, SqueezeNet (Iandola *et al.*, 2016) has the least WF_1 score and balanced accuracy. The SDCT-Net (Gehlot *et al.*, 2020*b*) is a very compact architecture with least number of parameters and testing time, and yet it is performing better than SqueezeNet. The SDCT-AuxNet$^{\theta}$ (Gehlot *et al.*, 2020*b*) is performing better than SDCT-Net, highlighting the impact of auxiliary classifier. Also, both of these architectures are trained from scratch with original image size (350×350). ShuffleNet-V2 (Ma *et al.*, 2018) performs better than SqueezeNet by 4.38% and 4.14% for WF$_1$ score and balanced accuracy, respectively. Also, the number of parameters (and test time) has increased significantly. GoogleNet (Szegedy *et al.*,

2015) performs with a marginal increase of 4.68% and 4.48% over SqueezeNet in terms of WF_1 score and balanced accuracy. DenseNet121 (Huang *et al.*, 2017) is performing almost similar to GoogleNet, but has a higher number of parameters and greater test time. ResNet34 (He *et al.*, 2016) and ResNeXT (Xie *et al.*, 2017) have better performance compared to ResNet18 (He *et al.*, 2016), but the former's number of parameters and test time are also relatively high. All of these three networks have residual connections and have better performance than DenseNet121 and a higher number of the parameters. MobileNet-V2 (Sandler *et al.*, 2018) has the least number of parameters (and test time) than all these architectures (except for SqueezeNet and ShuffleNet), but has better performance. Results with distance based cross entropy loss (DCE loss) are calculated by replacing the loss function in *BasePRL-Net* and training the network from scratch. The performance with DCE loss (Yang *et al.*, 2018) is better than all the existing architectures discussed to this point. Inception-V3 (Szegedy *et al.*, 2016) has the highest number of parameters and test time, but it also has better performance than these architectures. From Table 5.4, we observe that *BaseCE-Net* has better performance than all these architectures, which are also trained with BCE loss.

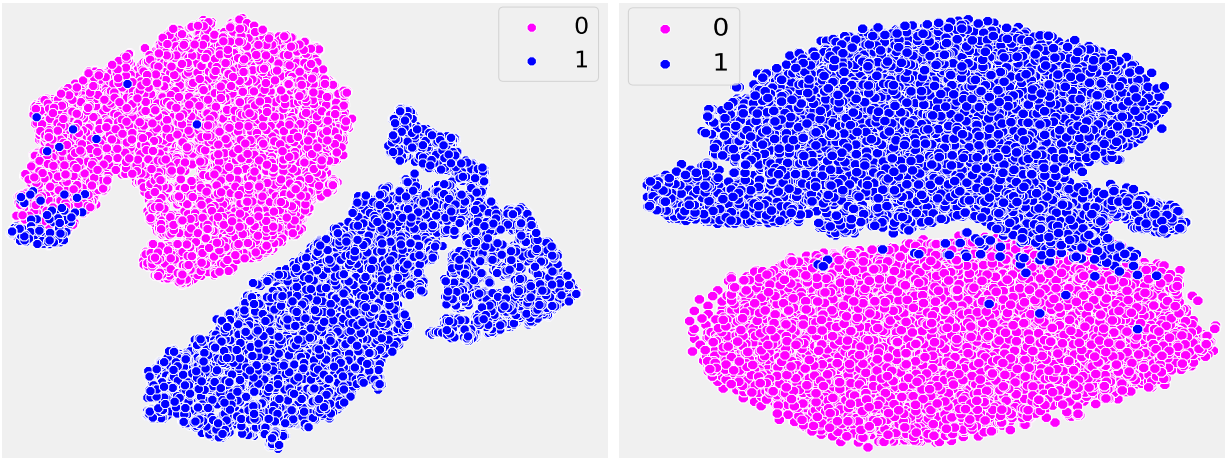
Further, projection loss performs better than the DCE loss. The DCE loss (Yang *et al.*, 2018) minimizes the euclidean distance between the class centers and respective activations, with no constraints on the centers. In contrast, the projection loss is minimizing the projection of the features on respective class vectors. It is also constraining class vectors to be orthogonal to each other. These results highlight the importance of the network's depth used in this proposed work and the contribution of projection loss over DCE loss. Finally, *PRLCE-Net+SD+CC* has a leading performance than the rest of the network in terms of both the metrics.

In conclusion, we observe that a larger number of parameters does not lead to significantly higher performance. This may be due to the relatively smaller size of the dataset, although the dataset is very large looking from the perspective of medical domain. Also, fewer parameters are not sufficient for the satisfactory performance either. Hence, it is necessary to design an optimal depth network that has, perhaps, been achieved with our custom network. Also, architectures trained from scratch have better performance than the ones initialized with pre-trained weights.

We also visualized the t-SNE plots for *PRLCE-Net*. Since this network has two

Table 5.6: Comparison of proposed method with some existing architectures in terms of weighted F_1 score, balanced accuracy, number of parameters, and test time. Results are computed on all 40441 test samples including time taken in decision making. Best results are highlighted in bold. $*$: parameters without coupling classifier. †: parameters without auxiliary classifier.

Weighted F_1 Score								
Models/ Architectures	Model-1	Model-2	Model-3	Model-4	Model-5	Majority Voting	Parameters	Time (in sec)
SqueezeNet	0.8161	0.8414	0.8356	0.8449	0.8447	0.8514	736450	2469.5092
SDCT-Net	0.8493	0.8849	0.8626	0.8354	0.8605	0.8700	95937	2064.1083
SDCT-AuxNet$^{\theta}$	0.8547	0.8819	0.8661	0.8447	0.8677	0.8811	95937†	2999.2237
ShuffleNet-V2	0.8539	0.8772	0.8928	0.8823	0.9072	0.8952	1255654	9259.4499
GoogleNet	0.8730	0.8943	0.9012	0.8702	0.9001	0.8982	5601954	12419.9610
DenseNet121	0.9041	0.8980	0.8890	0.8631	0.8942	0.8984	6955906	19884.8681
ResNet18	0.8892	0.8924	0.9018	0.8442	0.8994	0.9052	11177538	9360.4598
ResNeXt	0.9114	0.8931	0.9119	0.8581	0.8940	0.9058	22984002	18348.7165
ResNet34	0.8866	0.9211	0.9054	0.8372	0.9057	0.9094	21285698	15202.9181
MobileNet-V2	0.9098	0.8664	0.9131	0.9057	0.9093	0.9120	2226434	9505.9263
DCE Loss	0.9005	0.8813	0.9045	0.9090	0.9093	0.9142	2531979	3105.1951
Inception-V3	0.9210	0.9331	0.9043	0.8877	0.8944	0.9171	24348900	23590.9895
PRLCE-Net+SD+CC	0.9348	0.9381	0.9295	0.9388	0.9362	**0.9435**	2569871*	3694.9547
Balanced Accuracy								
Models/ Architectures	Model-1	Model-2	Model-3	Model-4	Model-5	Majority Voting	Parameters	Time (in sec)
Squeezenet	0.8142	0.8414	0.8381	0.8462	0.8435	0.8512	736450	2469.5092
SDCT-Net	0.8472	0.8832	0.8606	0.8332	0.8580	0.8677	95937	2064.1083
SDCT-AuxNet$^{\theta}$	0.8552	0.8799	0.8672	0.8429	0.8653	0.8796	95937†	2999.2237
Shufflenet-V2	0.8516	0.8746	0.8903	0.8796	0.9048	0.8926	1255654	9259.4499
GoogleNet	0.8709	0.8920	0.8993	0.8679	0.8979	0.8959	5601954	12419.9610
DenseNet121	0.9017	0.8958	0.8865	0.8604	0.8915	0.8957	6955906	19884.8681
ResNet18	0.8869	0.8898	0.8997	0.8432	0.8967	0.9028	11177538	9360.4598
ResNeXt	0.9093	0.8908	0.9094	0.8555	0.8912	0.9031	22984002	18348.7165
ResNet34	0.8844	0.9188	0.9028	0.8352	0.9031	0.9068	21285698	15202.9181
MobileNet-V2	0.8637	0.9108	0.9033	0.9065	0.9094	0.9071	2226434	9505.9263
DCE Loss	0.8990	0.8793	0.9030	0.9072	0.9069	0.9124	2531979	3105.1951
Inception-V3	0.9190	0.9320	0.9019	0.8868	0.8917	0.9149	24348900	23590.9895
PRLCE-Net+SD+CC	0.9330	0.9361	0.9275	0.9368	0.9344	**0.9417**	2569871*	3694.9547



(a) PRL branch (b) CE branch

Figure 5.10: t-SNE plots with Model-5 on the training data for (a) PRL branch, and (b) CE branch.

branches containing CE loss and PR loss, the scatter plots are depicted for both the losses. The t-SNE is used to reduce the feature dimension to 2 from 18432 and 512 for the CE branch and PRL branch, respectively. Resulting plots in Fig. 5.10 show the class separation with both of these branches on the training data with Model-5. Perfect classification is not achieved with either loss, as some samples lie on the opposite side of the boundary. In both of the cases, some cancer class samples are on the opposite side. However, with PR loss, there is a clear separation (with some error) of the two classes, which is not the case with the CE loss.

5.4.2 Camelyon7 and TBX11K datasets

We use these two datasets for the validation of the label noise handling with the proposed methodology.

Experiments Set-Up

The label noise is introduced in both the datasets and experiments are carried out with the proposed methodology to analyze its impact in handling the introduced label-noise. We have used pair flipping (Han *et al.*, 2018*a*) to introduce the label-noise. In pair flipping, for a noise level p, and the number of classes C, the transition matrix $\mathbf{A}_{C\times C}$ is given as:

$$\mathbf{A}_{C\times C} = \begin{bmatrix} 1-p & p & 0 & \dots & 0 \\ 0 & 1-p & p & \dots & 0 \\ & & \dots\dots\dots\dots & & \\ p & 0 & 0 & \dots & 1-p \end{bmatrix}, \tag{5.17}$$

where $\mathbf{A}_{m,k} = P(\hat{y} = k|y = m)$. As an example, for three classes ($C = 3$), and noise level(p) of 0.4:

$$\mathbf{A}_{C\times C} = \begin{bmatrix} 0.6 & 0.4 & 0 \\ 0 & 0.6 & 0.4 \\ 0.4 & 0 & 0.6 \end{bmatrix} \tag{5.18}$$

Hence, the label is flipped only to the next class with a given probability.

The PRLCE-Net architecture is used for both the datasets except for TBX11K in which a stride of 2 is used in the second last Conv Section also, and the number of class

Table 5.7: Results on clean Camelyon17 and TBX11K. The best results are highlighted in bold.

Camelyon17			**TBX11K**	
Architecture	WF1	BAC	Architecture	Accuracy
PRLCE-Net (CE)	0.9751	0.9751	PRLCE-Net (CE)	0.9255
PRLCE-Net (PRL)	0.9741	0.9741	PRLCE-Net (PRL)	**0.9364**
PRLCE-Net+CC	**0.9755**	**0.9755**	PRLCE-Net+CC	0.9313

vectors is three. Since there is a class imbalance in TBX11K, oversampling is used during the training to balance the classes. The training details for both datasets are the same as for the MM dataset.

Results on Camelyon17

Results on clean (noise-free) Camelyon17 are summarized in Table-5.7 in terms of weighted F1 score (WF1) and balanced accuracy (BAC). Next, the noise is introduced through pair flipping, and results are reported with PRLCE-Net in combination with label flipping (LF) and sample discarding (SD) in Table-5.8. From Table-5.7 & 5.8, the best results are obtained with clean data, and performance is decremented with increment in noise level. Further, LF and SD are able to increase the performance with the noisy data. For example, the best performance gain in terms of WF1 at noise levels $0.1, 0.2, 0.3, 0.4$ with inclusion of coupling classifier is $1.3\%, 1.08\%, 1.23\%,$ and 3.7%, respectively. At 0.45 the gain is 6.76%. Similar gains are observed for BAC. Also, both LF and SD provide approximately similar incremental performance, with LF giving the best performance at $p = 0.1$ and 0.4 while SD gives maximum performance on the remaining noise-levels. Another observation is higher gain at higher noise levels. Hence, incorporating noise handling approaches can provide significant gains, particularly at the higher noise.

Results on TBX11K

Results on the clean dataset are provided in Table-5.7. For coupling-classifier, results are reported with an ensembling approach. The best accuracy of 0.9364 is obtained with PRLCE-Net. These results are obtained through submission on the challenge portal. At the time of writing, these are the best classification results available on the leaderboard (TBX11K Tuberculosis Classification and Detection Challenge, 2020). As compared to

Table 5.8: Results on Camelyon17 at different noise-levels with PRLCE-Net and incorporation of label flipping (LF) and sample discarding (SD). Best results are highlighted in bold.

	WF1					BAC				
Noise-level (p)/ Architectures	0.1	0.2	0.3	0.4	0.45	0.1	0.2	0.3	0.4	0.45
PRLCE-Net (CE)	0.9600	0.9557	0.9421	0.9210	0.8832	0.9601	0.9557	0.9422	0.9214	0.8839
PRLCE-Net (PRL)	0.9596	0.9555	0.9424	0.9171	0.8792	0.9555	0.9555	0.9425	0.9176	0.8801
PRLCE-Net+CC	0.9609	0.9559	0.9424	0.9210	0.8832	0.9610	0.9560	0.9425	0.9214	0.8839
PRLCE-Net+LF (CE)	0.9732	0.9646	0.9505	0.9573	0.9460	0.9733	0.9646	0.9506	0.9574	0.9461
PRLCE-Net+LF (PRL	0.9732	0.9652	0.9524	**0.9580**	0.9507	0.9733	0.9653	0.9525	**0.9580**	0.9507
PRLCE-Net+LF (CC)	**0.9739**	0.9653	0.9523	**0.9580**	0.9507	**0.9739**	0.9653	0.9524	**0.9580**	0.9507
PRLCE-Net+SD (CE)	0.9686	0.9664	0.9534	0.9546	0.9506	0.9686	0.9664	0.9535	0.9547	0.9507
PRLCE-Net+SD (PRL)	0.9689	0.9665	**0.9547**	0.9551	**0.9508**	0.9689	0.9665	**0.9548**	0.9552	**0.9508**
PRLCE-Net+SD+CC	0.9696	**0.9667**	**0.9547**	0.9554	**0.9508**	0.9696	**0.9667**	**0.9548**	0.9554	**0.9508**

Table 5.9: Results on TBX11K at different noise-levels with PRLCE-Net and incorporation of sample discarding (SD). Best results are highlighted in bold.

Accuracy					
Noise-level (p)/ Architecture	0.1	0.2	0.3	0.4	0.45
PRLCE-Net (CE)	0.9122	0.8898	0.8486	0.8144	0.7020
PRLCE-Net (PRL)	0.9237	0.9079	0.8937	0.8640	0.7753
PRLCE-Net+CC	0.9204	0.9043	0.8937	0.8680	0.8071
PRLCE-Net+LF (CE)	0.9131	0.8934	0.8695	0.8531	0.8038
PRLCE-Net+LF (PRL)	**0.9249**	0.9064	**0.8943**	**0.8886**	0.8625
PRLCE-Net+LF+CC	0.9216	0.9037	0.8898	0.8852	**0.8692**
PRLCE-Net+SD (CE)	0.9146	0.8886	0.8422	0.7789	0.6814
PRLCE-Net+SD (PRL)	0.9237	**0.9085**	0.8904	0.8755	0.8107
PRLCE-Net+SD+CC	0.9194	0.9046	0.8901	0.8767	0.8295

Table 5.10: Average number of the samples affected in sample discarding (SD) and label flipping (LF) at different noise levels (p) for TBX11K.

Average Number of Samples					
Method/Noise level	0.1	0.2	0.3	0.4	0.45
SD	163	221	424	1350	1772
LF	162	213	354	740	954

the second-best results, our method is providing a gain of 2.85%. This performance is achieved with a very light architecture (only ten convolutional layers) and no other data augmentation apart from random rotation and oversampling.

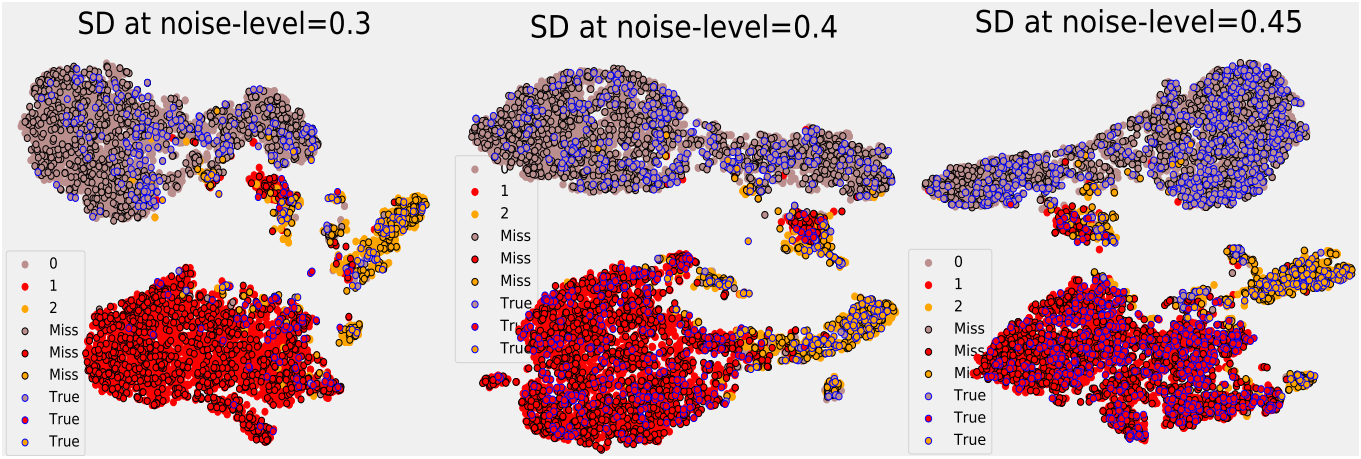
Results with noise addition are provided in Table-5.9. For brevity, we have used the knowledge of pair flipping in experiments with LF. Although this knowledge is not available in the practical scenario, we have used it for concept validation. There are $C-1$ possible scenarios in LF in the absence of such knowledge.

Table 5.11: True detection rate (TDR) and false detection rate (FDR) for TBX11K at different noise level with sample discarding (SD) and label flippng (LF)

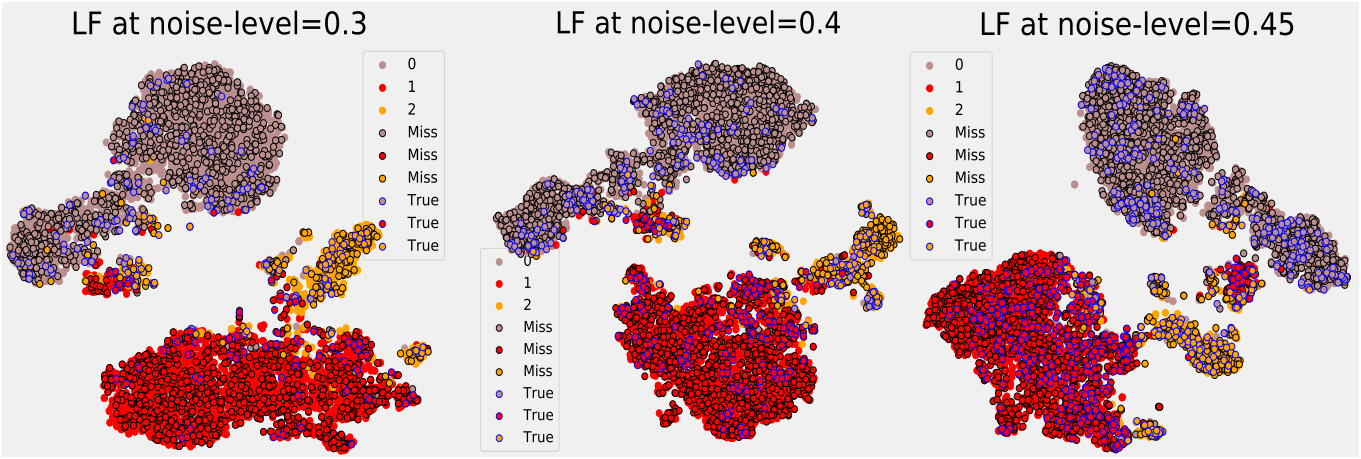
Method	Noise (p)/ Detection Rate	0.1	0.2	0.3	0.4	0.45
LF	TDR (in %)	4.95	6.25	9.09	11.66	16.73
	FDR (in %)	1.53	2.60	4.62	6.51	9.82
SD	TDR (in %)	5.81	7.82	14.22	20.23	28.62
	FDR (in %)	1.53	2.78	7.26	12.51	19.73

As expected, increased noise label results in decreased performance. Again there is an improvement with LF and SD. At 0.1, there no gain with SD, and with LF, the maximum gain is 0.12%. At 0.2, the maximum gain with SD is 0.06%. At $p = 0.3$ and 0.4, the maximum gain with LF is 0.06% and 2.46%, respectively. Also, there is a maximum gain of 0.87% at $p = 0.4$ with SD. Hence, there is no significant gain up to $p = 0.3$, but at $p = 0.4$, there is a significant improvement. Similarly, at 0.45, the maximum gain is 8.72% and 3.54% with LF and SD, respectively. For Camelyon7 also, the gain is very significant at $p = 0.4$ and 0.45.

Hence, the approach is very effective at the higher noise levels. Another aspect of TBX11K is higher LF performance than SD, especially at the higher noise levels. This is because a more number of the samples affected at the higher noise levels. Hence, in SD, the samples are dropped in a similar proportion making the dataset smaller. It is crucial if the dataset already has fewer samples. It is the case with TBX11K, as it has only 6889 training samples. Whereas in LF, the samples are retained and are some of them are potentially assigned correct labels leading to much more improved performance. To highlight this issue, the average number of the samples affected at different noise levels in SD and LF are reported in Table-5.10. On average, 1772 samples have been discarded in SD, which is why there is a large margin between SD and LF performance. We have also used the true detection rate (TDR) to represent the percentage of the noisy samples detected correctly. Similarly, the false detection rate (FDR) represents the fraction of the clean samples detected as the noisy samples. We report these two metrics for dataset TBX11K in Table-5.11 with sample discarding and label flipping. The two main observations from Table-5.11 are as follows: i) both TDR and FDR are increasing with the increasing noise level. Also, the increment is sharp for each additional noise level. For example, as the noise level increased from 0.4 to 0.45, TDR and FDR in LF increased by 5.07%, and 3.31% respectively. Similarly, the increment



(a) (b) (c)



(d) (e) (f)

Figure 5.11: t-SNE plots for TBX11K with SD (a-c) and LF (d-f) showing the detected noisy samples and missed noisy samples at noise level 0.3 (first column), 0.4 (second column), and 0.45 (third column).

in TDR and FDR with SD is 8.39%, and 7.22%, respectively. Hence, the approach is becoming aggressive with increment in the noise level. ii) The TDR and FDR are higher with SD as compared to LF. For example, at $p = 0.45$, SD has an additional 11.89% TDR over LF.

The gap between PRLCE-Net (CE) and PRLCE-Net (PRL) is also covered with the coupling classifier. The t-SNE plots with SD and LF at noise-levels 0.3, 0.4, and 0.45 for dataset TBX11K are shown in Fig-5.11. The plots show the detected and missed noisy samples. The noisy samples have been detected for all three classes. As observed from Table-5.11, detection increases with the increased noise level. Also, SD is more aggressive than LF.

5.5 Conclusion

We have proposed a CNN based unified framework for the diagnosis of multiple myeloma (MM). The problem is challenging due to inter-class visual homogeneity. We have addressed this classification problem through a methodology involving cross-entropy loss, novel projection loss, label noise handling, and coupling classifier. The resultant approach provides a final weighted F_1 score of 94.35%, and a balanced accuracy of 94.17% on a large test set of 40441 images. The method has a good subject-level performance, which is essential. However, the performance lacks on some of the subjects of the cancer class. The low performance on some subjects may be due to subject-level variability in the data. An approach to improve the performance may be to add more subjects to the training dataset. However, the collection of annotated medical datasets is a challenging task. Generative models such as GANs can be used to generate the new subjects' data. However, there could always be cases where testing subject distribution is relatively different from the training subjects. Hence, a robust approach may be to make the architecture immune to such variability.

We have also shown the application of the proposed methodology on the two other datasets (one binary and one three classes dataset). The proposed approach can work on both the datasets in the presence of noise also. Also, on a multi-class dataset (TBX11K), we achieve state-of-the-art classification results with the proposed architecture.

CHAPTER 6

Self-Supervision Based Dual-Transformation Learning for Stain Normalization, Classification and Segmentation

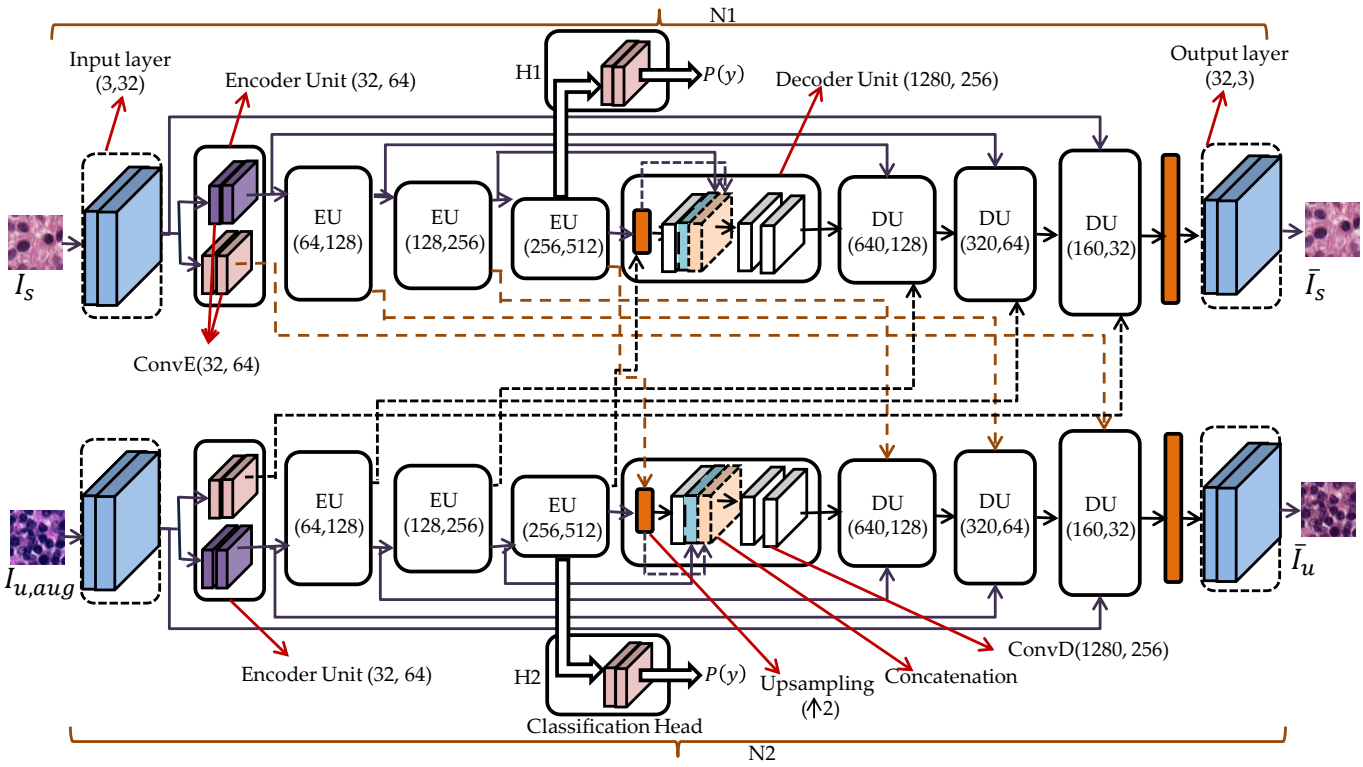
6.1 Introduction

Staining is a step in microscopic slide preparation that involves staining chemicals to highlight the regions of interest in the tissue/blood sample or bone marrow smear. Stain color variations in the images across datasets collected from various hospitals/centers may arise due to varying illumination conditions, stain chemicals, and staining time. Thus, the diagnostic tool trained on one center's dataset may perform non-optimally on another center's dataset. To counter this problem, stain color normalization is employed. Some of the widely used stain normalization approaches are based on histogram equalization Jain (1989); Tabesh *et al.* (2007); Kothari *et al.* (2011), color deconvolution Ruifrok and Johnston (2001); Abe *et al.* (2005); Macenko *et al.* (2009); McCann *et al.* (2014), and color transfer methods Reinhard *et al.* (2001); Ruderman *et al.* (1998); Magee *et al.* (2009). In histogram equalization methods, the probability density function of the reference image and source image colors are matched for each R, G, and B channel. In Magee *et al.* (2009), color transfer between the two images is used to achieve stain normalization, while in Reinhard *et al.* (2001), mean and standard deviation of the source and reference images are matched. However, these methods do not utilize histological information. Color deconvolution methods are based on negative matrix factorization (NMF) or singular value decomposition (SVD) to find the stain vectors. In Gupta *et al.* (2020*b*) SVD, along with the robust aligning of the reference and query images' Cartesian frames is utilized to counter stain color variation. A limitation of all these methods is that the performance depends on the reference image's choice and may change considerably with the change of reference image. Of late, deep learning (DL) methods are gaining significance in this area. Some generative

model-based DL methods for stain normalization are discussed in Zanjani *et al.* (2018); Ghazvinian Zanjani *et al.* (2018); Shaban *et al.* (2018). In Zanjani *et al.* (2018), Info-GAN is utilized, while in Ghazvinian Zanjani *et al.* (2018), variational-autoencoder and deep convolutional Gaussian mixture model are used for stain normalization. In Shaban *et al.* (2018), a CycleGAN based method is used to preserve the structure information while applying the color transformation. However, this method requires reference center's data that may not always be available. As a remedy, self-supervised learning can be employed that does not require GT Jing and Tian (2019). It is based on first training the network on some pretext tasks and later utilizing for the other downstream tasks. In Tellez *et al.* (2019), a network is trained to carry out stain normalization by learning to map the color (stain) augmented images to the corresponding original images. This can be considered as self-supervised learning, wherein the pretext task is to learn to transform the color augmented images to the original images. Color transformation of test images to that of training data can be considered as the downstream task. This work utilizes this approach but uses two coupled subnetworks (N1 and N2) instead of a single network and learns dual transformation (pretext tasks). In addition, classification heads are added to the encoders of both the subnetworks Mehta *et al.* (2018); Haghighi *et al.* (2020). The first subnetwork (N1) learns an identity transformation, while the second (N2) learns to perform stain normalization. N1 helps N2 learn the context-aware stain normalization task, while N2 helps N1 learn the stain-invariant features for the reconstruction task. Thus, both N1 and N2 assist each other, leading to improved stain-invariant performance on the subsequent classification task. We also experimented with segmentation as a downstream task. Since the proposed architecture can be utilized for stain normalization, classification, and segmentation, we name it an all-in-one network (AION). Elaborate experiments are presented on four datasets for each of the three tasks.

6.2 Methods

Consider the source image dataset and and the corresponding augmented dataset with color variations. The aim is to perform stain normalization such that the stain color profile of the augmented dataset is as close as possible to the source dataset. If trained successfully, the trained model can map one Center's images to those of another Centre



ConvE(A,B)=Conv2D(A,B,F=3,S=2,P=1)+BatchNorm2D+LeakyReLu; **ConvD(A,B)**=Conv2D(A,B,F=3,S=1,P=1)+BatchNorm2D+LeakyReLu
Decoder Unit (DU)=Upsampling by 2 + concatenation + ConvD(A,B); **Input Layer (A,B)**=Conv2D(A,B,F=3, S=2,P=1) +BatchNorm2D+LeakyReLu
Output layer(A,B)=Layer=Conv2D(A,B,F=3, S=1,P=1) +tanh; **Classification Head**=Conv2D(A=512, B=512,F=3,S=2,P=1) + BatchNorm2D + GAP + Linear+Softmax**; A**=no. of input filters, **B**=no. of output filters, **F**=filters of size F × F, **S**=stride, **P**=padding;

Figure 6.1: AION: An architecture with two coupled-networks and classification heads for learning two transformations; one identity and another for stain normalization. AION can also be used for stain-invariant classification and segmentation.

with a matching stain color profile. To accomplish this, we modify the architecture in Tellez *et al.* (2019) (here named as AION^{--}) to arrive at AION, presented in Fig.6.1.

Normalization by the AION architecture: The AION architecture can be seen as a self-supervised learning architecture consisting of two parallel U-Net types AION^{--} coupled through cross-connections running from the encoder units of one to the decoder units of the other. The first subnetwork, N1 is trained to learn the identity transformation on the source dataset. In contrast, the second subnetwork, N2 is supposed to map the augmented image dataset's stain color profile to that of the source dataset. Each subnetwork consists of an input layer, four encoder units (EUs), four decoder units (DUs), and one output layer. Input layer and encoder units constitute the encoder, while decoder units and output layer constitute the decoder. The input layer consists of 32 filters, while the successive EUs consist of twice the filters than the preceding layer/unit. Hence, the first EU consists of 64 filters, and the fourth EU consists of 512

filters. The size of each filter is 3×3. Each convolutional operation in the encoder is performed with a stride of two and a padding of one. This leads to a reduction of output size by two after each convolutional operation. Each EU consists of two parallel convolutions layers and provides a coupling mechanism for N1 and N2. These parallel layers receive the same input from the preceding EU. However, the output from one goes to the succeeding EU of the same subnetwork and the output of the other goes to the decoder unit of another subnetwork. Similar cross-connections are followed for both the subnetworks (Fig.6.1).

Each DU of N1 concatenates the input from the preceding layer/unit, the output from the corresponding EU of N1, and the output from the corresponding EU of N2. DUs also involve upsampling by two by the nearest neighbors approach to match the spatial size for concatenation. The concatenated output is given to the convolutional layers within DUs. The final output is forwarded to the next DU. Similar is the decoder structure of N2. Each DU contains half the number of filters of the preceding DU, with the first DU containing 256 filters and the fourth containing 32 filters. Finally, the output layer provides a three-channel output. Each convolutional operation in the network is followed by batch normalization (BatchNorm) and leaky ReLu except for the output layer, which has tanh activation and no BatchNorm.

Classification and Segmentation: To introduce the classification capability, we add two classification heads (CHs), one in each subnetwork on top of the last EU. Each CH consists of a single convolutional layer with 512 filters followed by BatchNorm, ReLu activation, global averaging pooling, and a classification layer. The CH of N1 predicts the class of images of the source dataset, while CH of N2 predicts the class of color augmented dataset. The AION can also be used for the segmentation. For this case, both N1 and N2 are fed with the source dataset, and the classification heads are not trained.

Training Methodology: To train the AION for stain normalization, an image I_s from the source dataset is given as input to N1. Another image $I_{u,aug}$, obtained through some transformation $\phi(\cdot)$ on image I_u from the source dataset, is given as input to N2, where indices u and s may or may not be equal. N1 aims to reconstruct the input image, while N2 attempts to match I_u. If we denote the output of N1 as $\bar{I}_s$ and of N2 as $\bar{I}_u$, the loss

function is given as:

$$\mathcal{L}_1 = MSE(\bar{I}_s, I_s) + MSE(\bar{I}_u, I_u), \tag{6.1}$$

where $MSE(\cdot)$ is the mean square error. Hence, N1 is learning an identity transformation, and N2 is learning ϕ^{-1}, assuming such inverse exists. Once trained, N2 should transform the stain (color) of the input data to that of the data input to N1. The classification heads are also trained simultaneously to predict the labels of the input images. Hence, the complete loss function is given as:

$$\mathcal{L} = MSE(\hat{I}_s, I_s) + MSE(\bar{I}_u, I_u) + NLL(y_1, \hat{y}_1) + NLL(y_2, \hat{y}_2), \tag{6.2}$$

where NLL is the negative log-likelihood, y is the true label, and $\hat{y}$ is the predicted label by the classification head. The subscripts 1 and 2 denote the classification head of Network-1 and Network-2, respectively. *It should be noted that classification Head-1 is trained on the source image data, while classification Head-2 is trained on the augmented data.*

For segmentation, the tanh activations at the output layers are replaced with sigmoid activations. The input of both the networks is the same, and the classification heads are not trained. Also, the original (not augmented) data is used for training. The respective segmented mask is predicted from both the networks, and both the networks are trained simultaneously using the binary cross-entropy loss.

6.3 Results

In this work, we use four publicly available datasets. Camelyon17 challenge dataset Bandi *et al.* (2019) is used for stain normalization, PatchCamelyon (PCam) Veeling *et al.* (2018) for classification, and Data Science Bowl (DSB) & CVC-ClinicDB (CVC) Bernal *et al.* (2015) for segmentation. Camelyon17 consists of hematoxylin and eosin (H&E) stained whole slide images (WSIs) of lymph node sections collected from five centers labeled as C_0 to C_4. For training and testing, the patches of size 128×128 are extracted from the level-0 of the WSIs. PCam have 96×96 patches having a binary label indicating the presence or absence of metastatic tissue. There is stain-variation among the patches, which makes the classification task challenging. The DSB is a cell segmentation dataset having images of varying sizes. We have extracted 128×128

Table 6.1: Description of the four dataset. Highlighted cells for Camelyon17 represent the sum of train and val dataset of the respective columns as the architecture trained on one center's is tested on another center's whole data.

Application	Stain Normalization and Classification						Segmentation	
Datasets	Camelyon17 (Patches)					PCam	Databowl	CVCDB
	C0	C1	C2	C3	C4	(Patches)	(Patches)	(Images)
Train	53538	31169	54834	94635	16776	262146	8640	488
Val	6000	4000	4000	15002	4000	32770	1360	62
Test	59538	35169	58834	109637	20776	32770	2503	62

Figure 6.2: Sample images from the datasets. (Set-1: left to right) Image from C0, C1, C2, C3, and C4 of Camelyon17. (Set-2) augmented images from C3 obtained with transformation $\phi(\cdot)$. (Set-3) PCam. (Set-4) DSB, (Set-5) CVC.

patches for training and testing. CVC is a polyps segmentation dataset with 612 images of size 384×288 pixels. For training with CVC, we have augmented the training set using random rotations. The distribution of the patches/images for all the datasets is provided in Table 6.1, and the sample images are shown in Fig. 6.2.

Stain-Normalization: For Camelyon17, there are variations in the colors of stained images of the five centers (Fig.6.2). Intuitively, a classifier trained on one center's data may perform unsatisfactorily on the remaining centers' data. We train a classifier on a particular center's data and test on the remaining centers' data for stain-normalization. The stain normalization quality can then be assessed through improvement in the classification performance over the unnormalized data. We begin our analysis with C3 as the training data. For comparison, we chose Macenko Macenko *et al.* (2009) and ReinhardReinhard *et al.* (2001) as non-DL methods, and StainGANShaban *et al.* (2018) and AION^{--} Tellez *et al.* (2019) as DL methods. StainGAN Shaban *et al.* (2018) is trained using C3 and C0 because it requires two training datasets. Results are also reported by adding a classification head to AION^{--}. We name this architecture as AION^{--}+H. Both AION and AION^{--}+H are trained with the same training methodology as discussed in Section-6.2. To obtain the augmented images, $\phi(\cdot)$ consists of saturation in the range [0,2.5] and hue in the range [–0.5,0.5] with a probability of 0.95. The sample augmented images are shown in Fig.6.2. Some results are shown with an external classifier (EC) that consists of 10 convolutional layers, batch normalization and Leaky ReLU as activation, and a classification layer.

All the models are implemented using PyTorch 1.1.0 and trained using RTX2080 GPU. Except for classifiers, all the models are trained using Adam optimizer for 80 epochs, batch size of 32, and initial learning rate of 0.001, which is reduced to one-tenth of the present value if there is no change in the validation performance for seven epochs. For classifiers, an SGD optimizer and a batch size of 64 are used for 150 epochs. The initial learning learning rate of 0.001 is reduced to one-tenth at 80th, 120th, and 140th epochs. The following results are reported with AION^{--}+H and AION:

1. **AION^{--}+H (H):** Results from the classification head that is added to AION^{--}
2. **AION (H1):** Results with N1's classification head of AION
3. **AION (H2):** Results with N2's classification Head of AION
4. **AION^{--}+H (EC):** Results with an external classifier trained on a center, and tested on normalized images obtained from AION^{--}+H
5. **AION (N2+EC):** Results with an external classifier trained on a center, and tested on normalized images obtained from N2 of AION

Results with C3 as training data are shown in Table-6.2 in terms of balanced accuracy (BAC), the area under the curve (AUC), and the weighted F1-score (WF1). The mean performance of the methods requiring reference images is inferior to other methods, showing their dependency on the reference image's choice. It can be seen that stain normalization leads to improved classification performance over original images for all the centers. For C0, AION (N2+EC) gives the top performance with a significant margin in terms of BAC and WF1. For C1 and C4, AION^{--}+H (EC) and AION (N2+EC) are having approximately similar performances, and for C2, AION^{--} is giving the best results. Apart from C2, both AION^{--}+H (EC) and AION (N2+EC) are performing better than AION^{--}. This shows the contributions of the added components to the AION^{--}. Another significant observation is the better performance of AION (H1) than original images. This shows that AION (H1) has also become stain-invariant, even though it is trained with the unnormalized images. This capability may have been introduced due to the coupling in the AION. AION^{--}+H (H) and AION (H2) are trained with augmented images and have seen varying stain colors. Hence, the performance improvement with these architectures is apparent. Table-6.3 shows the inter-center train-test results in terms of AUC. For all the centers, there is performance enhancement after stain normalization. Also, for most cases, results with AION are better than AION^{--}+H, which are better than AION^{--}.

Table 6.2: Results with C3 as training data, and other centers as the test data. The shaded cells represent mean results with five various reference images.

Dataset	C0			C1			C2			C4		
Metric/ Method	AUC	BAC	WF1	AUC	BAC	WF1	AUC	BAC	WF1	AUC	BAC	WF1
Original	0.7835	0.5960	0.5422	0.7581	0.7013	0.6948	0.4026	0.3338	0.3187	0.8628	0.7826	0.7824
Macenko Macenko *et al.* (2009)	0.5530	0.5025	0.4539	0.6117	0.5410	0.4767	0.5499	0.5428	0.5203	0.6096	0.5907	0.5490
Reinhard Reinhard *et al.* (2001)	0.5465	0.4898	0.3738	0.5833	0.5087	0.3791	0.4699	0.4759	0.2521	0.4353	0.4867	0.3532
StainGAN Shaban *et al.* (2018)	0.7926	0.5714	0.5023	0.8097	0.7217	0.7175	0.6882	0.6072	0.5241	0.5887	0.5097	0.365
AION^{--} Tellez *et al.* (2019)	**0.8611**	0.7192	0.7042	0.8257	0.7577	0.7553	**0.9390**	**0.8067**	**0.7535**	0.8115	0.7206	0.7137
AION^{--}+H (H)	0.7975	0.7199	0.7172	0.8327	0.7433	0.7398	0.7457	0.6751	0.6046	0.8739	0.8040	0.8039
AION^{--}+H (EC)	0.8438	0.7538	0.7497	0.8600	**0.7805**	**0.7794**	0.8174	0.7240	0.6567	**0.9289**	**0.8661**	**0.8659**
AION (H1)	0.7767	0.6941	0.6915	0.8040	0.7279	0.7248	0.6419	0.6072	0.6958	0.9286	0.8562	0.8562
AION (H2)	0.8019	0.7186	0.7161	0.8370	0.7395	0.7349	0.7833	0.6828	0.6008	0.8851	0.8129	0.8125
AION (N2+EC)	**0.8661**	**0.7733**	**0.7712**	**0.8605**	0.7781	0.7769	0.8579	0.7534	0.6937	0.9218	0.8627	0.8625

Table 6.3: AUC with inter-center training-testing

AUC								
Train	C0				C1			
Test	C1	C2	C3	C4	C0	C2	C3	C4
Original	0.8607	0.5076	0.7157	0.5080	0.8065	0.1852	0.6490	0.1052
AION^{--} Tellez *et al.* (2019)	**0.8714**	**0.9341**	0.9344	0.7714	**0.7169**	**•.8880**	0.7305	0.4088
AION^{--}+H (H)	0.8368	0.8915	0.9114	0.8973	0.7093	0.8448	0.8761	0.7907
AION^{--}+H (EC)	0.8183	0.9157	0.9304	0.9137	0.6674	0.8870	0.7787	0.4132
AION (H2)	0.8412	0.8956	0.9141	0.9029	0.7128	0.8465	**0.8774**	**0.8221**
AION (N2+EC)	0.8359	0.9130	**0.9362**	**0.9206**	0.6551	0.8680	0.8088	0.4920
Train	C2				C4			
Test	C0	C1	C3	C4	C0	C1	C2	C3
Original	0.6055	0.5722	0.3577	0.3617	0.4506	0.6532	0.7667	0.4012
AION^{--} Tellez *et al.* (2019)	0.6884	**0.8269**	0.7648	0.4211	0.5983	0.5797	0.8019	0.7130
AION^{--}+H (H)	0.6706	0.7722	0.7732	0.7732	0.5708	0.6920	0.7699	0.7466
AION^{--}+H (EC)	0.6772	0.8005	0.8455	0.5670	**0.6125**	0.6269	**0.8264**	**0.7817**
AION (H2)	**0.6995**	0.7800	0.7967	**0.7940**	0.5869	**0.7121**	0.7761	0.7529
AION (N2+EC)	0.6926	0.8034	**0.8508**	0.6623	0.6080	0.6418	0.7821	0.7576

Classification: To highlight AION's classification capability, we use the binary class dataset of PCam with inter-image stain variations. We have also compared the results with AION^{--}+H to show the proposed architecture's contribution. Results are summarized in Table-6.4.I. A methodology similar to stain normalization is used for training with PCam. First, we trained EC on the original PCam training set. This classifier provided an AUC of 0.8448 on the original PCam test set. The AION^{--}+H (H), AION (H1), and AION (H2) provide a gain of 4%, 4.61%, and 3.66%, respectively. Even though these classifiers are shallow (6 layers) than EC (10 layers), they provided a significant gain. This gain could be more with deeper heads. Hence, the proposed architecture can produce stain-invariant classifiers as a byproduct. To explore further, we stain-normalized PCam using AION^{--}+H and AION trained on C3 of Camelyon17. Training and testing with the resultant dataset provided an AUC of 0.8948 and 0.9105 with AION^{--}+H and AION (N2), respectively, which again proves AION's usefulness. We also replaced the custom classifier with ResNet-24 and DenseNet-161. The best AUC is achieved with AION (N2) for both classifiers, which is 0.9356 with ResNet-34, and 0.9275 with DenseNet-161.

Table 6.4: (I) PCam Classification results, (II) CVC and DSB Segmentation results

(I) PCam Classification Results					
Classifier	Method	AUC	Classifier	Method	AUC
AION^{--}+H (H)	AION^{--}+H	0.8848	Resnet-34	Original	0.8948
AION (H1)	AION	**0.8909**		AION^{--} Tellez *et al.* (2019)	0.9059
AION (H2)	AION	0.8814		AION^{--}+H	**0.9181**
				AION (N2)	**0.9356**
EC	Original	0.8448	DensNet-161	Original	0.8691
	AION^{--} Tellez *et al.* (2019)	0.9054		AION^{--} Tellez *et al.* (2019)	0.9137
	AION^{--}+H	0.8948		AION^{--}+H	0.9091
	AION (N2)	**0.9105**		AION (N2)	**0.9275**

(II) Segmentation results on CVC and DSB with AION^{--}+H and AION						
Dice Similarity Coefficient (DSC)						
Dataset	CVC			DSB		
Initialization/Method	AION^{--}+H	AION (N1)	AION (N2)	AION^{--}+H	AION (N1)	AION (N2)
C3-Pretrained	0.6897	**0.7453**	0.6896	0.8670	0.8533	**0.8682**
Random	0.8243	0.8198	**0.8328**	0.8712	**0.8723**	0.8705
Intersection over Union (IoU)						
Dataset	CVC			DSB		
Initialization/Method	AION^{--}+H	AION (N1)	AION (N2)	AION^{--}+H	AION (N1)	AION (N2)
C3-Pretrained	0.7406	**0.7765**	0.7407	0.8527	0.8378	**0.8542**
Random	0.8361	0.8327	**0.8432**	0.8568	**0.8579**	0.8560

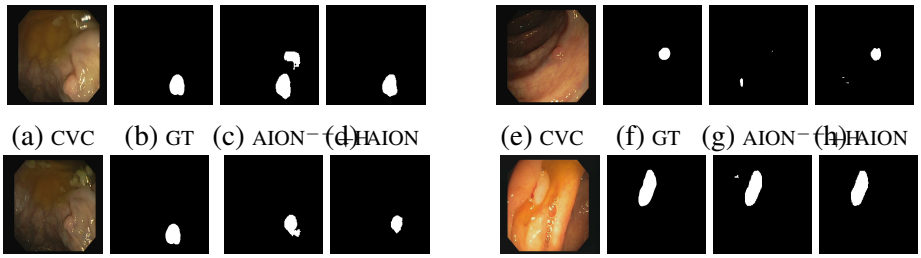


Figure 6.3: Sample segmentation masks from AION^{--}+H and AION (H2) from CVC.

Segmentation: In one set of experiments, we considered segmentation as the downstream task and initialized AION^{--}+H and AION with the network's weights trained on C3 (C3-Pretrained). In contrast, in the other experiment, they are initialized randomly. Also, as the classification head is not trained, AION^{--} and AION^{--}+H are the same for segmentation. Both AION^{--}+H and AION gave an inferior performance with C3-pretrained networks as compared to random initialization (Table-6.4.II). This shows that stain-normalization-specific features are, perhaps, not helpful in the segmentation task on this dataset. Also, the performance difference between the two initializations is large for CVC as compared to DSB. This is due to the similarity in the imaging modality of DSB and C3. Also, AION has better performance than AION^{--}+H for both datasets. The qualitative results are also presented in Fig. 6.3.

6.4 Conclusion

We proposed a novel self-supervision based dual-transformation coupled-network architecture for stain normalization. The architecture is also equipped with classification heads that can achieve stain-invariant classification. The utility is also shown for the

downstream task of segmentation. The architecture can perform well on all three applications of stain normalization, classification, and segmentation.

CHAPTER 7

Conclusion and Future Work

This thesis systematically develops the CAD pipeline for blood cancer diagnosis (ALL and MM). The dissertation proposes module-specific methodologies for stain normalization, segmentation, and classification for a specific CAD tool. This section summarizes the main contribution and future directions of this dissertation.

7.1 GCTI-SN

Stain variation is a generic problem in histopathology image analysis that arises due to staining during slide preparation. This variation poses challenges in the deployment of DL based models in an *inter-center training-testing* setting. The mismatch between the stain basis vectors of the source and target image gives rises to such a variation.

Geometry-Inspired Chemical and Tissue Invariant Stain Normalization (GCTI-SN) attempts to align these stain basis vectors through illumination correction, stain color correction, and stain quantity correction. In effect, these steps redefine and align the color vectors in the OD space. GCTI-SN offers advantages over other SVD based approaches that do not consider the alignment of the vectors. GCTI-SN performs better than other methods both qualitatively and quantitatively. From the application's perspective, the GCTI-SN leads to significant performance enhancement in inter-center training-testing on Camelyon17, a publicly available dataset containing WSIs from five different centers/hospitals, leading to drastic stain variation. We also analyze the effect of augmentation during the training in handling the stain variation. During training, the color-based augmentations induce the stain invariance in the network, improving the test performance on the target data. However, the result shows that the best performance is achieved with a combination of color augmentations and stain normalization. A significant performance deterioration is observed in the absence of these modifications.

7.2 EDNFC-Net

EDNFC-Net targets the nuclei instance segmentation and follows an encoder-decoder hierarchy. It uses a feature concatenation cell (FCC) as the basic building unit. Features reusability and better gradient flow are encouraged through intra/inter FCC features concatenation. The latter also promotes the multiscale feature flow throughout the encoder and decoder. Additionally, EDNFC-Net is trained using a balanced loss that minimizes false positives (FP) and false negatives (FN) for its convergences. Compared to Dice Loss, balanced loss forces more penalty for similar FP and FN. The EDNFC-Net performs better than U-Net on the images of four different modalities.

7.3 SDCT-AuxNet$^{\theta}$

The ALL diagnosis is performed using SDCT-AuxNet$^{\theta}$ that systematically addresses the challenges of the dataset that contains nucleus microscopic images. The nucleus of the malignant and healthy cells are visually similar that is a significant challenge. Hence, the SD layer followed by the DCT layer is used as the initial layer in the network. The combined effect of these two layers is to induce different sparsity levels in the images of the two classes that become a significant discriminating characteristic. This architecture named SDCT-Net is further improved by inducing an auxiliary classifier used in the coupling fashion with SDCT-Net leading to SDCT-AuxNet$^{\theta}$. The confidence score of the SDCT-Net acts as the coupling parameter in SDCT-AuxNet$^{\theta}$. In effect, SDCT-AuxNet$^{\theta}$ is an ensemble of two classifiers coupled through a coupling variable. The SDCT-AuxNet has a weighted F1 of 0.9480 on the test set and also consistently good subject-level performance.

7.4 MM Diagnosis

A dual-head architecture with a common backbone is used for the MM diagnosis. One head uses the BCE loss while the other utilizes the proposed projection loss (PRL). The PRL is focused on maximizing the projection of the activations on the respective class vectors while simultaneously enforcing the orthogonality between the inter-class

vectors. This architecture is trained to bring both heads to an agreement that leads to performance improvement and provides scope for label correction. After initial training, the disagreement between the two heads is used for identifying the potential samples with label noise. Next, fine-tuning is performed by applying label flipping and sample discarding on the identified samples. An additional coupling classifier is induced in the network during the testing to address the disagreement between the two heads. Apart from the MM, the proposed methodology improves significantly on the other two public datasets of different modalities.

7.5 AION

All-In-One-Network for stain normalization, classification, and segmentation is a coupled encoder-decoder type architecture that utilizes self-supervision for stain normalization, classifier heads at encoders' end for classification, and decoders for segmentation as the downstream task. The proposed architecture also leads to stain-invariant classification heads.

7.6 Future Directions

A significant issue about GCTI-SN and generic stain normalization is the dependency on the reference image. The performance may vary significantly with reference images. This is due to the inability of the single image to capture the underlying color distribution. A remedy may be to take into account multiple images instead of a single image from the source data. A very naive approach to achieve this may be to use an eigen image created from the multiple images of the source data as the reference image. However, the formal methodology and analysis of such an approach are left as a future direction. Underlying structure preservation during stain normalization is another issue to be considered to design a robust algorithm.

EDNFC-Net performs the nuclei instance segmentation. However, the complete cell consists of nucleus and cytoplasm, but EDNFC-Net is focussed on the former only, completely discarding the latter, which may also contain class-specific information. A future step in this direction may be to focus on complete cell segmentation (nucleus and

cytoplasm).

For ALL and MM diagnosis, the classification of the nuclei is extracted using the segmentation network. As discussed above, utilizing the complete cell instead of just the nucleus may lead to improved performance. Secondly, as observed, the classification performance depends on the quality of the segmented cells that may be a limiting factor. A solution to resolve this is to eliminate the intermediate segmentation and perform classification using the whole microscopic images. Also, the current approach requires the annotation of the individual cell, which is expensive. Multiple instance learning (MIL) may be a potential step in this direction. MIL eliminates the necessity to classify the individual cells in an image and assigns a single label to a collection of cells/images. The stain normalization may also effect the performance as the it may alter the underlying structure. The alternative may be to utilize stain adaptability instead of stain normalization.

APPENDIX A

A.1 Additional Experiments for GCTI-SN

We have conducted the following two additional set of experiments to explore these ideas.

Experiment-1: In the first set of experiments, we included a very high blur in the existing BANS augmentations and name the new set of augmentations as BANSB. The blur is introduced through a Gaussian filter with mean zero and a standard deviation between 7 and 8. The resultant images are shown in Fig. A.1 (row 3). As observed from the figure, the details are hard to interpret after the application of blur. Next, we have used BANSB with original (augmented dataset) and GCTSI-SN normalized images (augmented + normalized dataset) to train the classifier. The performance of the classifier with these settings is tabulated in Table-A.1. For the augmented dataset, the performance of the classifier has reduced on all the metrics except for specificity. However, this improvement in specificity is obtained at the cost of large reduction in sensitivity. The specificity and sensitivity measure the ratio of correctly predicted negative and positive samples, respectively. This implies that the classifier is biased towards the negative (healthy) class.

A decrease in performance is also observed with the combination of the normalized and augmented dataset. However, in this case, the classifier is not significantly biased towards a single class because there is no significant increase in the sensitivity or specificity at the expense of the other. But the performance is still inferior to the experiment in the manuscript without adding a high blur. In conclusion, the performance with BANS (without high blurring) is significantly better than with BANSB (with high blurring).

Experiment-2: In the second set of the experiments, we modified the range of brightness, contrast, saturation, and hue in BANS to obtain the lighter color variations. We call these set of augmentations as BANS^{l}. Numerically, the brightness and contrast are in the range $[.9, 1.1]$, saturation is in the range $[.5, 1.5]$, and hue is in the range

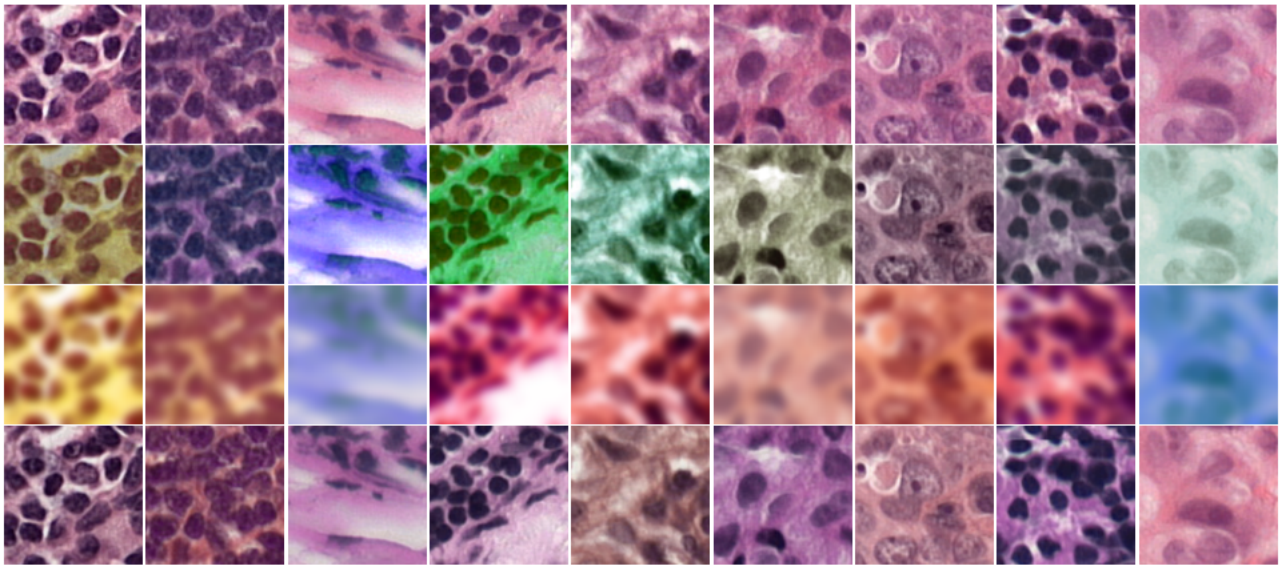


Figure A.1: Sample original images of C3 (row 1) and corresponding BANS based color augmented images (row 2). A large blur is also introduced as an additional augmentation and the resultant images are shown in row 3. Row 4 shows the BANS-based color augmented images. For visualization purpose, affine transformations are not introduced in these images.

Table A.1: Effect of introducing very high blur on the performance of the classifier with and without normalization. This blurring is introduced as an additional augmentation. The best value for each row (center) is highlighted in black.

Trained with Augmented Dataset (BANS)						Trained with Augmented Dataset (BANSB)				
Test Dataset	Accuracy	F1 score	AUC	Sensitivity	Specificity	Accuracy	F1 score	AUC	Sensitivity	Specificity
Center 0	**0.829**	**0.816**	**0.927**	**0.761**	0.896	0.651	0.477	0.853	0.320	**0.976**
Center 1	**0.816**	**0.790**	**0.916**	**0.708**	0.919	0.688	0.547	0.897	0.386	**0.976**
Center 2	**0.745**	**0.789**	**0.886**	**0.658**	0.973	0.375	0.215	0.797	0.121	**0.996**
Center 4	**0.841**	**0.829**	**0.913**	**0.770**	0.913	0.834	0.812	0.905	0.721	**0.947**
Trained with Augmented Dataset (BANS) + Normalized Dataset						Trained with Augmented Dataset (BANSB) + Normalized Dataset				
Test Dataset	Accuracy	F1 score	AUC	Sensitivity	Specificity	Accuracy	F1 score	AUC	Sensitivity	Specificity
Center 0	**0.867**	**0.862**	**0.944**	**0.844**	**0.889**	0.6847	0.641	0.661	0.570	0.798
Center 1	**0.800**	**0.7645**	**0.902**	**0.668**	**0.925**	0.660	0.557	0.782	0.437	0.872
Center 2	**0.833**	**0.881**	**0.916**	**0.872**	**0.739**	0.655	0.736	0.645	0.677	0.603
Center 4	**0.870**	**0.863**	**0.927**	0.818	**0.922**	0.757	0.771	0.849	**0.821**	0.693

$[-0.1, 0.1]$. The range of the remaining augmentations (affine and noise) are kept unchanged. The resultant augmented images are shown in Fig. A.1 (row 4). As observed from the figure, color variations with BANS^l is more natural as compared with BANS. We have applied BANS^l to the original (augmented dataset) and normalized (augmented+normalized) images of C3 to train the classifier. The test performance on the remaining centers is shown in Table-A.2. For the augmented dataset, BANS is performing better than BANS^l on most of the performance metrics. The same observation is made for the combination of normalized and augmented dataset. This difference in performance with BANS and BANS^l can be explained as follows: Training with widely color varied images leads to a classifier that is robust to stain variations. BANS introduces much stronger color variations (even if they are not natural) that helps to achieve

a better stain-invariant classifier as compared to BANS^l. The idea is to use the augmentations such that the classifier is stain-invariant. Augmentations in a larger range of color variations introduces this invariance and hence, makes the method robust.

Table A.2: Effect of BANS on the performance of the classifier with and without normalization. Augmented dataset is prepared with lighter color augmentation compared to those used in manuscript R1. The best value for each row (center) is highlighted in black.

Trained with Augmented Dataset (BANS)					
Test Dataset	Accuracy	F1 score	AUC	Sensitivity	Specificity
Center 0	**0.829**	**0.816**	**0.927**	**0.761**	**0.896**
Center 1	**0.816**	**0.790**	**0.916**	**0.708**	0.919
Center 2	**0.745**	**0.789**	0.886	**0.658**	0.973
Center 4	**0.841**	**0.829**	**0.913**	0.770	**0.913**
Trained with Augmented Dataset(BANS) + Normalized Dataset					
Test Dataset	Accuracy	F1 score	AUC	Sensitivity	Specificity
Center 0	**0.867**	**0.862**	0.944	**0.844**	0.889
Center 1	**0.800**	**0.764**	0.902	**0.668**	0.925
Center 2	**0.833**	**0.881**	**0.916**	**0.872**	0.739
Center 4	0.870	0.863	0.927	0.818	**0.922**

Trained with Augmented Dataset (BANS^l)				
Accuracy	F1 score	AUC	Sensitivity	Specificity
0.805	0.786	0.902	0.720	0.889
0.797	0.762	0.907	0.665	**0.923**
0.674	0.703	**0.908**	0.545	**0.988**
0.785	0.785	0.863	**0.786**	0.784
Trained with Augmented Dataset (BANS^l)+ Normalized Dataset				
Accuracy	F1 score	AUC	Sensitivity	Specificity
0.864	0.855	**0.946**	0.808	**0.919**
0.796	0.754	**0.908**	0.642	**0.943**
0.825	0.874	0.907	0.854	**0.754**
0.873	**0.868**	**0.933**	**0.837**	0.909

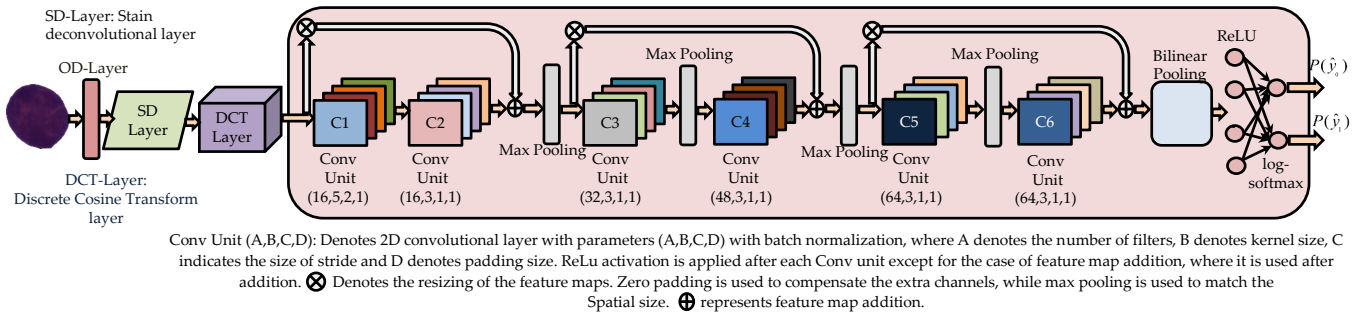
A.2 SDCT-Net with Skip Connections (SDCT-Net+S)

In order to further analyze the effect of skip connections as suggested by the reviewer, we introduced skip connections in the SDCT-Net, such that total trainable parameters remained unchanged. The modified architecture is shown in Fig. A.2 and the comparison in terms of trainable parameters are shown in Table-A.3. Skip connections are used to add the feature maps at a gap of two convolutional layers. In case, two feature maps are of different sizes, zero padding is used to increase the depth and max-pooling is used for downsampling the feature maps. Batch normalization and ReLU activation are applied after each convolutional layer. However, in the case of feature map addition, activation is applied after addition. This strategy is inspired by He *et al.* (2016) and introduces no extra parameters.

Table A.3: Comparison of SDCT-Net and SDCT-Net+S in term of parameters. Both architectures have same number of parameters.

Architecture	SDCT-Net	SDCT-Net+S
Parameters	95362	95362

The comparison of SDCT-Net and SDCT-Net+S in terms of performance metrics is provided in Table-A.4. It is observed that the performance of both the architectures is approximately the same. We believe that the lesser depth of the architecture may be the



Conv Unit (A,B,C,D): Denotes 2D convolutional layer with parameters (A,B,C,D) with batch normalization, where A denotes the number of filters, B denotes kernel size, C indicates the size of stride and D denotes padding size. ReLu activation is applied after each Conv unit except for the case of feature map addition, where it is used after addition. ⊗ Denotes the resizing of the feature maps. Zero padding is used to compensate the extra channels, while max pooling is used to match the Spatial size. ⊕ represents feature map addition.

Figure A.2: SDCT-Net+S. Skip Connections are introduced in the SDCT-Net such that total trainable parameters remains unchanged.

reason for this outcome. While skip connections are helpful to improve the training and performance of very deep architectures, the SDCT-Net has only six layers (He *et al.*, 2016). Due to this, there is no significant gain with shortcut connections.

Table A.4: Comparison of SDCT-Net+S and SDCT-Net. SDCT-Net+S is the architecture of SDCT-Net supported with skip connections. Model1 represents the model trained with fold1 as the validation set and remaining folds as the training set. Similar notation applies to Model2 to Model7. Results are tabulated on the test set of 2586 images. Both architectures are providing approximately similar performance.

F_1 Score for Normal Class								
Architecture\Model	Model1	Model2	Model3	Model4	Model5	Model6	Model7	Majority Voting
SDCT-Net+S	0.830	0.838	0.793	0.793	0.813	0.792	0.806	0.860
SDCT-Net	0.813	0.855	0.836	0.856	0.822	0.774	0.825	0.862
F_1 Score for Cancer Class								
Architecture\Model	Model1	Model2	Model3	Model4	Model5	Model6	Model7	Majority Voting
SDCT-Net+S	0.917	0.920	0.894	0.921	0.917	0.870	0.900	0.933
SDCT-Net	0.900	0.926	0.921	0.926	0.900	0.855	0.912	0.928
Weighted F_1 (WF_1) Score								
Architecture\Model	Model1	Model2	Model3	Model4	Model5	Model6	Model7	Majority Voting
SDCT-Net+S	0.889	0.894	0.862	0.880	0.884	0.844	0.870	0.910
SDCT-Net	0.873	0.904	0.894	0.904	0.875	0.829	0.885	0.908
Balanced Accuracy (*BAC*)								
Architecture\Model	Model1	Model2	Model3	Model4	Model5	Model6	Model7	Majority Voting
SDCT-Net+S	0.879	0.885	0.855	0.833	0.859	0.871	0.867	0.898
SDCT-Net	0.875	0.903	0.882	0.905	0.887	0.856	0.879	0.911

A.3 Flexibility of SDCT-AuxNet$^{\theta}$

It is evident from the architecture of SDCT-AuxNet$^{\theta}$ and its training process that the proposed architecture uses two different classifiers. The main CNN classifier acts on raw cell images followed by feature extraction CNN layers (called feature learning sub-

network), bilinear pooling layer and traditional NN layers. On the other hand, the auxiliary classifier uses a different transformation of spectral averaging (in contrast to the bilinear pooling used in the main CNN classifier) on the output of the shared feature learning subnetwork. Hence, both the classifiers (main and auxiliary) share weights of feature learning subnetwork. The advantages of this weight sharing are two folds: i) end-to-end training from raw images to the main classifier output, and ii) the flexibility in choosing the auxiliary classifier. If different feature learning subnetworks are used for each classifier, then it is not able to exploit the main classifier. Secondly, the choice of this auxiliary classifier will be limited, which can be understood as follow. For example, if we would like the auxiliary classifier to work on raw images with end-to-end training, classifiers such as Kernel SVM, random forest, etc. cannot be employed as auxiliary classifiers. And, if we want to have the choice of auxiliary classifiers as wide open, then classifiers like SVM will work on some handcrafted features that may be sub-optimal compared to the features borrowed from the shared feature learning subnetwork proposed in the current setup.

REFERENCES

1. **Abe, T.**, **Y. Murakami**, **M. Yamaguchi**, **N. Ohyama**, and **Y. Yagi** (2005). Color correction of pathological images based on dye amount quantification. *Optical review*, **12**(4), 293–300.

2. **Akram, S. U.**, **J. Kannala**, **L. Eklund**, and **J. Heikkilä**, Cell segmentation proposal network for microscopy image analysis. *In* **G. Carneiro**, **D. Mateus**, **L. Peter**, **A. Bradley**, **J. M. R. S. Tavares**, **V. Belagiannis**, **J. P. Papa**, **J. C. Nascimento**, **M. Loog**, **Z. Lu**, **J. S. Cardoso**, and **J. Cornebise** (eds.), *Deep Learning and Data Labeling for Medical Applications*. 2016.

3. **American Cancer Society** (2019). `https://www.cancer.org/`. Online; accessed 12 June 2019.

4. **Amin, M. M.**, **S. Kermani**, **A. Talebi**, and **M. Ghelich Oghli** (2016*a*). Recognition of acute lymphoblastic leukemia cells in microscopic images using k-means clustering and support vector machine classifier. *Journal of Medical Signals and Sensors*, **6**(3), 183–93.

5. **Amin, M. M.**, **A. Memari**, **N. Samadzadehaghdam**, **S. Kermani**, *et al.* (2016*b*). Computer aided detection and classification of acute lymphoblastic leukemia cell subtypes based on microscopic image analysis. *Microscopy Research and Technique*, **79**(10), 908–916.

6. **Ansari, N.** and **A. Gupta** (2017). Image reconstruction using matched wavelet estimated from data sensed compressively using partial canonical identity matrix. *IEEE Transactions on Image Processing*, **26**(8), 3680–3695. ISSN 1057-7149.

7. **Asthana, S.**, **S. Labani**, **S. Mehrana**, and **S. Bakhshi** (2018). Incidence of childhood leukemia and lymphoma in india. *Pediatric Hematology Oncology Journal*, **3**(4), 115 – 120. ISSN 2468-1245.

8. **Bandi, P.**, **O. Geessink**, **Q. Manson**, **M. Van Dijk**, **M. Balkenhol**, **M. Hermsen**, **B. Ehteshami Bejnordi**, **B. Lee**, **K. Paeng**, **A. Zhong**, **Q. Li**, **F. G. Zanjani**, **S. Zinger**, **K. Fukuta**, **D. Komura**, **V. Ovtcharov**, **S. Cheng**, **S. Zeng**, **J. Thagaard**, **A. B. Dahl**, **H. Lin**, **H. Chen**, **L. Jacobsson**, **M. Hedlund**, **M. Cetin**, **E. Halici**, **H. Jackson**, **R. Chen**, **F. Both**, **J. Franke**, **H. Kusters-Vandevelde**, **W. Vreuls**, **P. Bult**, **B. van Ginneken**, **J. van der Laak**, and **G. Litjens** (2019). From detection of individual metastases to classification of lymph node status at the patient level: The camelyon17 challenge. *IEEE Transactions on Medical Imaging*, **38**(2), 550–560.

9. **Basavanhally, A.** and **A. Madabhushi**, EM-based segmentation-driven color standardization of digitized histopathology. *In SPIE Medical Imaging*. International Society for Optics and Photonics, 2013.

10. **Bayramoglu, N.** and **J. Heikkilä**, Transfer learning for cell nuclei classification in histopathology images. *In Computer Vision – ECCV 2016 Workshops*. 2016.

11. **Bayramoglu, N.**, **J. Kannala**, and **J. Heikkilä**, Deep learning for magnification independent breast cancer histopathology image classification. *In 2016 23rd International Conference on Pattern Recognition (ICPR)*. 2016.

12. **Bernal, J.**, **F. J. Sánchez**, **G. Fernández-Esparrach**, **D. Gil**, **C. Rodríguez**, and **F. Vilariño** (2015). Wm-dova maps for accurate polyp highlighting in colonoscopy: Validation vs. saliency maps from physicians. *Computerized Medical Imaging and Graphics*, **43**, 99–111.

13. **Bertels, J.**, **T. Eelbode**, **M. Berman**, **D. Vandermeulen**, **F. Maes**, **R. Bisschops**, and **M. B. Blaschko**, Optimizing the dice score and jaccard index for medical image segmentation: Theory and practice. *In* **D. Shen**, **T. Liu**, **T. M. Peters**, **L. H. Staib**, **C. Essert**, **S. Zhou**, **P.-T. Yap**, and **A. Khan** (eds.), *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*. 2019.

14. **Bilgin, C. C.**, **J. Rittscher**, **R. Filkins**, and **A. Can** (2012). Digitally adjusting chromogenic dye proportions in brightfield microscopy images. *Journal of microscopy*, **245**(3), 319–330.

15. **Caicedo, J. C.**, **J. Roth**, **A. Goodman**, **T. Becker**, **K. W. Karhohs**, **M. Broisin**, **C. Molnar**, **C. McQuin**, **S. Singh**, **F. J. Theis**, and **A. E. Carpenter** (2019). Evaluation of deep learning strategies for nucleus segmentation in fluorescence images. *Cytometry Part A*, **95**(9), 952–965.

16. **Cappabianco, F. A. M.**, **P. F. O. Ribeiro**, **P. A. V. de Miranda**, and **J. K. Udupa**, A general and balanced region-based metric for evaluating medical image segmentation algorithms. *In 2019 IEEE International Conference on Image Processing (ICIP)*. 2019.

17. **Chang, Y. H.**, **G. Thibault**, **O. Madin**, **V. Azimi**, **C. Meyers**, **B. Johnson**, **J. Link**, **A. Margolin**, and **J. W. Gray**, Deep learning based Nucleus Classification in pancreas histological images. *In 2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. 2017.

18. **Chatap, N.** and **S. Shibu** (2014). Analysis of blood samples for counting leukemia cells using support vector machine and nearest neighbour. *IOSR Journal of Computer Engineering*, **16**(5), 79–87.

19. **Chauhan, A.**, **D. Chauhan**, and **C. Rout** (2014). Role of gist and phog features in computer-aided diagnosis of tuberculosis without segmentation. *PloS One*, **9**(11), e112980.

20. **Chen, H.**, **X. Qi**, **L. Yu**, **Q. Dou**, **J. Qin**, and **P.-A. Heng** (2017). Dcan: Deep contour-aware networks for object instance segmentation from histology images. *Medical Image Analysis*, **36**, 135 – 146. ISSN 1361-8415.

21. **Chen, X.**, **Y. Duan**, **R. Houthooft**, **J. Schulman**, **I. Sutskever**, and **P. Abbeel**, InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. *In Proceedings of the 30th International Conference on Neural Information Processing Systems*. 2016.

22. **Clark, K.**, **B. Vendt**, **K. Smith**, **J. Freymann**, **J. Kirby**, **P. Koppel**, **S. Moore**, **S. Phillips**, **D. Maffitt**, **M. Pringle**, *et al.* (2013*a*). The Cancer Imaging Archive (TCIA): maintaining and operating a public information repository. *Journal of digital imaging*, **26**(6), 1045–1057.

23. **Clark, K.**, **B. Vendt**, **K. Smith**, **J. Freymann**, *et al.* (2013*b*). The cancer imaging archive (TCIA): Maintaining and operating a public information repository. *Journal of Digital Imaging*, **26**(6), 1045–1057.

24. **D. P. Kingma, J. B.** (2014). Adam: A Method for Stochastic Optimization. *arXiv:1412.6980*.

25. **Deng, S.**, **X. Zhang**, **W. Yan**, **E. I. C. Chang**, **Y. Fan**, **M. Lai**, and **Y. Xu** (2020). Deep learning in digital pathology image analysis: a survey. *IEEE Transactions on Biomedical Engineering*, **14**(7), 470–487.

26. **Dhungel, N.**, **G. Carneiro**, and **A. P. Bradleyc** (2017). A deep learning approach for the analysis of masses in mammograms with minimal user intervention. *Medical Image Analysis*, **37**, 114 – 128. ISSN 1361-8415.

27. **Ding, Y.**, **Y. Yang**, and **Y. Cui**, Deep learning for classifying of white blood cancer. *In* **A. Gupta** and **R. Gupta** (eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*. Springer Singapore, 2019.

28. **DSB** (2019). Data Science Bowl. `https://www.kaggle.com/c/data-science-bowl-2018`. Accessed: 2019-10-02.

29. **Duggal, R.**, **A. Gupta**, and **R. Gupta**, Segmentation of overlapping/touching white blood cell nuclei using artificial neural networks. *In CME Series on Hemato-Oncopathology, All India Institute of Medical Sciences (AIIMS), New Delhi, India*. 2016*a*.

30. **Duggal, R.**, **A. Gupta**, **R. Gupta**, and **P. Mallick**, SD-Layer: Stain deconvolutional layer for cnns in medical microscopic imaging. *In* **M. Descoteaux**, **L. Maier-Hein**, **A. Franz**, **P. Jannin**, **D. L. Collins**, and **S. Duchesne** (eds.), *Medical Image Computing and Computer Assisted Intervention – MICCAI 2017*. Springer International Publishing, 2017.

31. **Duggal, R.**, **A. Gupta**, **R. Gupta**, **M. Wadhwa**, *et al.*, Overlapping cell nuclei segmentation in microscopic images using deep belief networks. *In Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing*. 2016*b*.

32. **Esteva, A.**, **B. Kuprel**, **R. A. Novoa**, **J. Ko**, *et al.* (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, **542**, 115–118. ISSN 0028-0836.

33. **Gao, Z.**, **L. Wang**, **L. Zhou**, and **J. Zhang** (2017*a*). Hep-2 cell image classification with deep convolutional neural networks. *IEEE Journal of Biomedical and Health Informatics*, **21**(2), 416–428.

34. **Gao, Z.**, **L. Wang**, **L. Zhou**, and **J. Zhang** (2017*b*). HEp-2 cell image classification with deep convolutional neural networks. *IEEE Journal of Biomedical and Health Informatics*, **21**(2), 416–428. ISSN 2168-2194.

35. **Gehlot, S.**, **A. Gupta**, and **R. Gupta**, EDNFC-Net: Convolutional Neural Network with Nested Feature Concatenation for Nuclei-Instance Segmentation. *In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020*a*.

36. **Gehlot, S.**, **A. Gupta**, and **R. Gupta** (2020*b*). SDCT-AuxNet$^{\theta}$: DCT augmented stain deconvolutional CNN with auxiliary classifier for cancer diagnosis. *Medical Image Analysis*, **61**, 101661. ISSN 1361-8415.

37. **Gelasca, E. D.**, **J. Byun**, **B. Obara**, and **B. Manjunath**, Evaluation and benchmark for biological image segmentation. *In IEEE International Conference on Image Processing*. 2008. URL `http://vision.ece.ucsb.edu/publications/elisa_ICIP08.pdf`.

38. **Ghazvinian Zanjani, F.**, **S. Zinger**, **P. de With**, **B. E. Bejnordi**, and **J. van der Laak**, Histopathology stain-color normalization using deep generative models. *In 1st Conference on Medical Imaging with Deep Learning (MIDL 2018)*. 2018.

39. **Goswami, S.**, **S. Mehta**, **D. Sahrawat**, **A. Gupta**, and **R. Gupta** (2020). Heterogeneity loss to handle intersubject and intrasubject variability in cancer.

40. **Gupta, A.**, **R. Duggal**, **S. Gehlot**, **R. Gupta**, **A. Mangal**, **L. Kumar**, **N. Thakkar**, and **D. Satpathy** (2020*a*). GCTI-SN: geometry-inspired chemical and tissue invariant stain normalization of microscopic medical images. *Medical Image Analysis*, **65**, 101788.

41. **Gupta, A.**, **R. Duggal**, **S. Gehlot**, **R. Gupta**, **A. Mangal**, **L. Kumar**, **N. Thakkar**, and **D. Satpathy** (2020*b*). GCTI-SN: Geometry-inspired chemical and tissue invariant stain normalization of microscopic medical images. *Medical Image Analysis*, **65**, 101788.

42. **Gupta, A.** and **R. Gupta** (2019*a*). ALL challenge dataset of ISBI 2019 [Dataset]. *The Cancer Imaging Archive*. URL `https://doi.org/10.7937/tcia.2019.dc64i46r`.

43. **Gupta, A.** and **R. Gupta** (2019*b*). ALL challenge dataset of ISBI 2019 [dataset]. *The Cancer Imaging Archive*. URL `https://doi.org/10.7937/tcia.2019.dc64i46r`.

44. **Gupta, A.** and **R. Gupta**, *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*. Springer, Singapore., 2019*c*. ISBN 978-981-15-0797-7.

45. **Gupta, A.** and **R. Gupta** (2019*d*). SN-AM Dataset: White Blood Cancer Dataset of B-ALL and MM for Stain Normalization [Data set]. *The Cancer Imaging Archive (TCIA)*. URL `https://doi.org/10.7937/tcia.2019.of2w8lxr`.

46. **Gupta, A.**, **R. Gupta**, **S. Gehlot**, and **S. Mourya** (2019). Classification of normal vs malignant cells in B-ALL white blood cancer microscopic images. *IEEE International Symposium on Biomedical Imaging (ISBI)-2019 challenges*. URL `https://competitions.codalab.org/competitions/20395#learn_the_details-top-entries-of-the-challenge`.

47. **Gupta, R.**, **P. Mallick**, **R. Duggal**, **A. Gupta**, *et al.* (2017). Stain color normalization and segmentation of plasma cells in microscopic images as a prelude to development of computer assisted automated disease diagnostic tool in multiple myeloma. *Clinical Lymphoma, Myeloma and Leukemia*, **17**(1), e99.

48. **Haghighi, F.**, **M. R. Hosseinzadeh Taher**, **Z. Zhou**, **M. B. Gotway**, and **J. Liang**, Learning semantics-enriched representation via self-discovery, self-classification, and self-restoration. *In* **A. L. Martel**, **P. Abolmaesumi**, **D. Stoyanov**, **D. Mateus**, **M. A. Zuluaga**, **S. K. Zhou**, **D. Racoceanu**, and **L. Joskowicz** (eds.), *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*. 2020.

49. **Han, B.**, **Q. Yao**, **X. Yu**, **G. Niu**, **M. Xu**, **W. Hu**, **I. Tsang**, and **M. Sugiyama**, Co-teaching: Robust training of deep neural networks with extremely noisy labels. *In Advances in Neural Information Processing Systems 31*. 2018*a*.

50. **Han, S. S.**, **M. S. Kim**, **W. Lim**, **G. H. Park**, **I. Park**, and **S. E. Chang** (2018*b*). Classification of the Clinical Images for Benign and Malignant Cutaneous Tumors Using a Deep Learning Algorithm. *Journal of Investigative Dermatology*, **138**(7), 1529 – 1538.

51. **Han, X.-H.**, **J. Lei**, and **Y.-W. Chen**, HEp-2 Cell Classification Using K-Support Spatial Pooling in Deep CNNs. *In* **G. Carneiro**, **D. Mateus**, **L. Peter**, **A. Bradley**, **J. M. R. S. Tavares**, **V. Belagiannis**, **J. P. Papa**, **J. C. Nascimento**, **M. Loog**, **Z. Lu**, **J. S. Cardoso**, and **J. Cornebise** (eds.), *Deep Learning and Data Labeling for Medical Applications*. 2016.

52. **Harangi, B.** (2018). Skin lesion classification with ensembles of deep convolutional neural networks. *Journal of Biomedical Informatics*, **86**, 25 – 32.

53. **He, K.**, **X. Zhang**, **S. Ren**, and **J. Sun**, Deep residual learning for image recognition. *In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

54. **Hekler, A.**, **J. S. Utikal**, **A. H. Enk**, **W. Solass**, **M. Schmitt**, **J. Klode**, **D. Schadendorf**, **W. Sondermann**, **C. Franklin**, **F. Bestvater**, **M. J. Flaig**, **D. Krahl**, **C. von Kalle**, **S. Fröhling**, and **T. J. Brinker** (2019). Deep learning outperformed 11 pathologists in the classification of histopathological melanoma images. *European Journal of Cancer*, **118**, 91 – 96.

55. **Huang, G.**, **Z. Liu**, **L. Van Der Maaten**, and **K. Q. Weinberger**, Densely connected convolutional networks. *In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

56. **Iandola, F. N.**, **S. Han**, **M. W. Moskewicz**, **K. Ashraf**, **W. J. Dally**, and **K. Keutzer** (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. *arXiv:1602.07360*.

57. **Jaeger, S.**, **S. Candemir**, **S. Antani**, **Y.-X. J. Wáng**, **P.-X. Lu**, and **G. Thoma** (2014). Two public chest x-ray datasets for computer-aided screening of pulmonary diseases. *Quantitative Imaging in Medicine and Surgery*, **4**(6), 475.

58. **Jain, A. K.**, *Fundamentals of digital image processing*. Prentice-Hall, Inc., 1989.

59. **Jiang, F.**, **H. Liu**, **S. Yu**, and **Y. Xie**, Breast mass lesion classification in mammograms by transfer learning. *In Proceedings of the 5th International Conference on Bioinformatics and Computational Biology*. New York, NY, USA, 2017.

60. **Jing, L.** and **Y. Tian** (2019). Self-supervised visual feature learning with deep neural networks: A survey.

61. **Johnson, J. W.** (2018). Adapting Mask-RCNN for automatic nucleus segmentation. *arXiv:1805.00500*.

62. **Joshi, M. D.**, **A. H. Karode**, and **S. Suralkar** (2013). White blood cells segmentation and classification to detect acute leukemia. *International Journal of Emerging Trends and Technology in Computer Science (IJETTCS)*, **2**(3), 147–151.

63. **Karimi, D.**, **H. Dou**, **S. K. Warfield**, and **A. Gholipour** (2020). Deep learning with noisy labels: Exploring techniques and remedies in medical image analysis. *Medical Image Analysis*, **65**, 101759.

64. **Karthikeyan, D. R.** and **N. Poornima** (2017). Micros-copic image segmentation using fuzzy c means for leukemia diagnosis. *International Journal of Advanced Research in Science, Engineering and Technology*, **4**(1).

65. **Kazemi, F.**, **T. Abbasian Najafabadi**, and **B. Nadjar Araabi** (2015). Automatic Recognition of Acute Myelogenous Leukemia in Blood Microscopic Images Using K-means Clustering and Support Vector Machine. *Journal of Medical Signals and Sensors*, **5**(1), 49–58.

66. **Khan, A. M.**, **N. Rajpoot**, **D. Treanor**, and **D. Magee** (2014). A nonlinear mapping approach to stain normalization in digital histopathology images using image-specific color deconvolution. *IEEE Transactions on Biomedical Engineering*, **61**(6), 1729–1738.

67. **Koschan, A.** and **M. Abidi**, *Digital color image processing*. John Wiley and Sons, 2008.

68. **Kothari, S.**, **J. H. Phan**, **R. A. Moffitt**, **T. H. Stokes**, **S. E. Hassberger**, **Q. Chaudry**, **A. N. Young**, and **M. D. Wang**, Automatic batch-invariant color segmentation of histological cancer images. *In From Nano to Macro, 2011 IEEE International Symposium on Biomedical Imaging*. 2011.

69. **Kumar, N.**, **R. Verma**, **S. Sharma**, **S. Bhargava**, **A. Vahadane**, and **A. Sethi** (2017). A dataset and a technique for generalized nuclear segmentation for computational pathology. *IEEE Transactions on Medical Imaging*, **36**(7), 1550–1560.

70. **Lee, K.**, **X. He**, **L. Zhang**, and **L. Yang**, Cleannet: Transfer learning for scalable image classifier training with label noise. *In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018.

71. **Li, X.** and **K. N. Plataniotis** (2015). A complete color normalization approach to histopathology images using color cues computed from saturation-weighted statistics. *IEEE Transactions on Biomedical Engineering*, **62**(7), 1862–1873.

72. **Li, X.** and **K. N. Plataniotis** (2015). A complete color normalization approach to histopathology images using color cues computed from saturation-weighted statistics. *IEEE Transactions on Biomedical Engineering*, **62**(7), 1862–1873.

73. **Lin, T.-Y.**, **A. RoyChowdhury**, and **S. Maji**, Bilinear CNN models for fine-grained visual recognition. *In Proceedings of ICCV*. 2015.

74. **Litjens, G.**, **T. Kooi**, **B. E. Bejnordi**, **A. A. A. Setio**, **F. Ciompi**, **M. Ghafoorian**, **J. A. van der Laak**, **B. van Ginneken**, and **C. I. Sánchez** (2017). A survey on deep learning in medical image analysis. *Medical Image Analysis*, **42**, 60 – 88. ISSN 1361-8415.

75. **Liu, Y.** and **F. Long**, Acute lymphoblastic leukemia cells image analysis with deep bagging ensemble learning. *In* **A. Gupta** and **R. Gupta** (eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*. Springer Singapore, 2019.

76. **Liu, Y.**, **Y.-H. Wu**, **Y. Ban**, **H. Wang**, and **M.-M. Cheng**, Rethinking computer-aided tuberculosis diagnosis. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

77. **Long, J.**, **E. Shelhamer**, and **T. Darrell**, Fully convolutional networks for semantic segmentation. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

78. **Ma, N.**, **X. Zhang**, **H.-T. Zheng**, and **J. Sun**, Shufflenet v2: Practical guidelines for efficient cnn architecture design. *In* **V. Ferrari**, **M. Hebert**, **C. Sminchisescu**, and **Y. Weiss** (eds.), *Computer Vision – ECCV 2018*. Cham, 2018.

79. **Macenko, M.**, **M. Niethammer**, **J. Marron**, **D. Borland**, **J. T. Woosley**, **X. Guan**, **C. Schmitt**, and **N. E. Thomas**, A method for normalizing histology slides for quantitative analysis. *In ISBI*. 2009.

80. **Macenko, M.**, **M. Niethammer**, **J. S. Marron**, **D. Borland**, **J. T. Woosley**, **Xiaojun Guan**, **C. Schmitt**, and **N. E. Thomas**, A method for normalizing histology slides for quantitative analysis. *In 2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*. 2009.

81. **Madhukar, M.**, **S. Agaian**, and **A. T. Chronopoulos**, New decision support tool for acute lymphoblastic leukemia classification. *In Proc. SPIE 8295, Image Processing: Algorithms and Systems X; and Parallel Processing for Imaging Applications II, 829518*. 2012.

82. **Magee, D.**, **D. Treanor**, **D. Crellin**, **M. Shires**, **K. Smith**, **K. Mohee**, and **P. Quirke**, Colour normalisation in digital histopathology images. *In Proc Optical Tissue Image analysis in Microscopy, Histopathology and Endoscopy (MICCAI Workshop)*, volume 100. 2009.

83. **Mazo, C.**, **J. Bernal**, **M. Trujillo**, and **E. Alegre** (2018). Transfer learning for classification of cardiovascular tissues in histological images. *Computer Methods and Programs in Biomedicine*, **165**, 69 – 76.

84. **McCann, M. T.**, **J. Majumdar**, **C. Peng**, **C. A. Castro**, and **J. Kovacevic**, Algorithm and benchmark dataset for stain separation in histology images. *In 2014 IEEE International Conference on Image Processing (ICIP)*. 2014.

85. **Mehta, S.**, **E. Mercan**, **J. Bartlett**, **D. Weaver**, **J. Elmore**, and **L. Shapiro**, Y-Net: Joint Segmentation and Classification for Diagnosis of Breast Biopsy Images. *In International Conference on Medical image computing and computer-assisted intervention*. Springer, 2018.

86. **Meijering, E.** (2012). Cell segmentation: 50 years down the road [life sciences]. *IEEE Signal Processing Magazine*, **29**(5), 140–145.

87. **Meng, N.**, **E. Y. Lam**, **K. K. Tsia**, and **H. K. So** (2019). Large-Scale Multi-Class Image-Based Cell Classification With Deep Learning. *IEEE Journal of Biomedical and Health Informatics*, **23**(5), 2091–2098.

88. **Mishra, S.**, **B. Majhi**, and **P. K. Sa** (2019). Texture feature based classification on microscopic blood smear for acute lymphoblastic leukemia detection. *Biomedical Signal Processing and Control*, **47**, 303 – 311. ISSN 1746-8094.

89. **Mishra, S.**, **B. Majhi**, **P. K. Sa**, and **L. Sharma** (2017). Gray level co-occurrence matrix and random forest based acute lymphoblastic leukemia detection. *Biomedical Signal Processing and Control*, **33**, 272–280.

90. **MM** (2019). Multiple Myeloma. `https://www.cancer.org/cancer/multiple-myeloma`. Accessed: 2019-10-02.

91. **Mohapatra, S.**, **D. Patra**, and **S. Satpathy** (2014). An ensemble classifier system for early diagnosis of acute lymphoblastic leukemia in blood microscopic images. *Neural Computing and Applications*, **24**(7), 1887–1904.

92. **Mohapatra, S.**, **S. S. Samanta**, **D. Patra**, and **S. Satpathi**, Fuzzy based blood image segmentation for automated leukemia detection. *In 2011 International Conference on Devices and Communications (ICDeCom)*. 2011.

93. **Mourya, S.**, **S. Kant**, **P. Kumar**, **A. Gupta**, *et al.*, LeukoNet: DCT-based CNN architecture for the classification of normal versus Leukemic blasts in B-ALL Cancer. *In arXiv:1810.07961*. 2018.

94. **Naylor, P.**, **M. Laé**, **F. Reyal**, and **T. Walter**, Nuclei segmentation in histopathology images using deep neural networks. *In 2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*. 2017.

95. **Naylor, P.**, **M. Laé**, **F. Reyal**, and **T. Walter** (2019). Segmentation of nuclei in histopathology images by deep regression of the distance map. *IEEE Transactions on Medical Imaging*, **38**(2), 448–459.

96. **Neoh, S. C.**, **W. Srisukkham**, **L. Zhang**, **S. Todryk**, *et al.* (2015). An intelligent decision support system for leukaemia diagnosis using microscopic blood images. *Scientific Reports*, **5**, 1–14.

97. **Newberg, J.** and **R. F. Murphy** (2008). A framework for the automated analysis of subcellular patterns in human protein atlas images. *Journal of proteome research*, **7**(6), 2300–2308.

98. **O.Ronneberger**, **P. Fischer**, and **T. Brox**, U-Net: Convolutional networks for biomedical image segmentation. *In International Conference on Medical Image Computing and Computer Assisted Intervention*. 2015.

99. **Pan, Y.**, **M. Liu**, **Y. Xia**, and **D. Shen**, Neighborhood-correction algorithm for classification of normal and malignant cells. *In* **A. Gupta** and **R. Gupta** (eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*. Springer Singapore, 2019.

100. **Patel, N.** and **A. Mishra** (2015). Automated leukaemia detection using microscopic images. *Procedia Computer Science*, **58**, 635 – 642. ISSN 1877-0509.

101. **Phan, H. T. H.**, **A. Kumar**, **J. Kim**, and **D. Feng**, Transfer learning of a convolutional neural network for HEp-2 cell image classification. *In 2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*. 2016.

102. **Prellberg, J.** and **O. Kramer**, Acute lymphoblastic leukemia classification from microscopic images using convolutional neural networks. *In* **A. Gupta** and **R. Gupta** (eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*. Springer Singapore, 2019.

103. **Putzu, L.**, **G. Caocci**, and **C. D. Ruberto** (2014). Leukocyte classification for leukaemia detection using image processing techniques. *Artificial Intelligence in Medicine*, **62**(3), 179 – 191. ISSN 0933-3657.

104. **Qin, F.**, **N. Gao**, **Y. Peng**, **Z. Wu**, **S. Shen**, and **A. Grudtsin** (2018). Fine-grained leukocyte classification with deep residual learning for microscopic images. *Computer Methods and Programs in Biomedicine*, **162**, 243 – 252.

105. **Rabinovich, A.**, **S. Agarwal**, **C. Laris**, **J. H. Price**, and **S. J. Belongie**, Unsupervised color decomposition of histologically stained tissue samples. *In Advances in neural information processing systems*. 2003.

106. **Raid, A. M.**, **W. M. Khedr**, and **W. Ahmed**, JPEG image compression using discrete cosine transform - a survey. *In arXiv:1405.6147*. 2014.

107. **Rajpurkar, P.**, **J. Irvin**, **K. Zhu**, *et al.*, CheXNet: Radiologist-level pneumonia detection on chest x-rays with deep learning. *In arXiv:1711.05225*. 2017.

108. **Rawat, J.**, **A. Singh**, **H. S. Bhadauria**, **J. Virmani**, *et al.* (2017*a*). Classification of acute lymphoblastic leukaemia using hybrid hierarchical classifiers. *Multimedia Tools and Applications*, **76**(18), 19057–19085. ISSN 1573-7721.

109. **Rawat, J.**, **A. Singh**, **B. HS**, **J. Virmani**, *et al.* (2017*b*). Computer assisted classification framework for prediction of acute lymphoblastic and acute myeloblastic leukemia. *Biocybernetics and Biomedical Engineering*, **37**(4), 637 – 654. ISSN 0208-5216.

110. **Rehman, A.**, **N. Abbas**, **T. Saba**, **S. I. u. Rahman**, *et al.* (2018). Classification of acute lymphoblastic leukemia using deep learning. *Microscopy Research and Technique*, **81**(11), 1310–1317.

111. **Reinhard, E.**, **M. Ashikhmin**, **B. Gooch**, and **P. Shirley** (2001). Color transfer between images. *IEEE Computer graphics and applications*, **21**(5), 34–41.

112. **Reta, C.**, **L. Altamirano**, **J. A. Gonzalez**, **R. Diaz-Hernandez**, *et al.* (2015). Segmentation and classification of bone marrow cells images using contextual information for medical diagnosis of acute leukemias. *PLOS ONE*, **10**, 1–18.

113. **Ronneberger, O.**, **P. Fischer**, and **T. Brox**, U-Net: Convolutional networks for biomedical image segmentation. *In Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. 2015*a*.

114. **Ronneberger, O.**, **P. Fischer**, and **T. Brox**, U-Net: Convolutional networks for biomedical image segmentation. *In Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. 2015*b*.

115. **Ruderman, D. L.**, **T. W. Cronin**, and **C.-C. Chiao** (1998). Statistics of cone responses to natural images: Implications for visual coding. *JOSA A*, **15**(8), 2036–2045.

116. **Ruifrok, A. C.** and **D. A. Johnston** (2001). Quantification of histochemical staining by color deconvolution. *Analytical and quantitative cytology and histology/the International Academy of Cytology [and] American Society of Cytology*, **23**(4), 291–299.

117. **Sandler, M.**, **A. Howard**, **M. Zhu**, **A. Zhmoginov**, and **L. Chen**, Mobilenetv2: Inverted residuals and linear bottlenecks. *In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018.

118. **Sermanet, P.**, **D. Eigen**, **X. Zhang**, **M. Mathieu**, *et al.*, OverFeat: Integrated recognition, localization and detection using convolutional networks. *In arXiv:1312.6229*. 2013.

119. **Shaban, M. T.**, **C. Baur**, **N. Navab**, and **S. Albarqouni** (2018). StainGAN: Stain style transfer for digital histological images. *arXiv preprint arXiv:1804.01601*.

120. **Shafique, S.** and **S. Tehsin** (2018). Acute lymphoblastic leukemia detection and classification of its subtypes using pretrained deep convolutional neural networks. *Technology in Cancer Research & Treatment*, **17**.

121. **Shah, S.**, **W. Nawaz**, **B. Jalil**, and **H. A. Khan**, Classification of normal and leukemic blast cells in b-all cancer using a combination of convolutional and recurrent neural networks. *In* **A. Gupta** and **R. Gupta** (eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*. Springer Singapore, 2019.

122. **Sharma, H.**, **N. Zerbe**, **I. Klempert**, **O. Hellwich**, and **P. Hufnagl** (2017). Deep convolutional neural networks for automatic classification of gastric carcinoma using whole slide images in digital histopathology. *Computerized Medical Imaging and Graphics*, **61**, 2 – 13. Selected papers from the 13th European Congress on Digital Pathology.

123. **Shi, T.**, **L. Wu**, **C. Zhong**, **R. Wang**, and **W. Zheng**, Ensemble convolutional neural networks for cell classification in microscopic images. *In* **A. Gupta** and **R. Gupta** (eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*. Springer Singapore, 2019.

124. **Singhal, V.** and **P. Singh**, Texture features for the detection of acute lymphoblastic leukemia. *In Proceedings of International Conference on ICT for Sustainable Development*. 2016. ISBN 978-981-10-0135-2.

125. **Sirinukunwattana, K.**, **S. E. A. Raza**, **Y. Tsang**, **D. R. J. Snead**, **I. A. Cree**, and **N. M. Rajpoot** (2016). Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images. *IEEE Transactions on Medical Imaging*, **35**(5), 1196–1206.

126. **Springenberg, J.**, **A. Dosovitskiy**, **T. Brox**, and **M. Riedmiller**, Striving for simplicity: The all convolutional net. *In ICLR (workshop track)*. 2015.

127. **Sridhar, D.** and **I. Murali Krishna**, Brain tumor classification using discrete cosine transform and probabilistic neural network. *In 2013 International Conference on Signal Processing , Image Processing Pattern Recognition*. 2013.

128. **Szegedy, C.**, **V. Vanhoucke**, **S. Ioffe**, **J. Shlens**, and **Z. Wojna**, Rethinking the inception architecture for computer vision. *In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

129. **Szegedy, C.**, **Wei Liu**, **Yangqing Jia**, **P. Sermanet**, **S. Reed**, **D. Anguelov**, **D. Erhan**, **V. Vanhoucke**, and **A. Rabinovich**, Going deeper with convolutions. *In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

130. **Tabesh, A.**, **M. Teverovskiy**, **H.-Y. Pang**, **V. P. Kumar**, **D. Verbel**, **A. Kotsianti**, and **O. Saidi** (2007). Multifeature prostate cancer diagnosis and gleason grading of histological images. *IEEE Transactions on Medical Imaging*, **26**(10), 1366–1378.

131. **Tabibu, S.**, **P. Vinod**, and **C. Jawahar** (2019). Pan-Renal Cell Carcinoma classification and survival prediction from histopathology images using deep learning. *Sci Rep*, **9**, 10509.

132. **TBX11K Tuberculosis Classification and Detection Challenge** (2020). `https://competitions.codalab.org/competitions/25848`. Online; accessed 02 Feb 2020.

133. **Tellez, D.**, **G. Litjens**, **P. Bándi**, **W. Bulten**, **J.-M. Bokhorst**, **F. Ciompi**, and **J. van der Laak** (2019). Quantifying the effects of data augmentation and stain color normalization in convolutional neural networks for computational pathology. *Medical Image Analysis*, **58**, 101544. ISSN 1361-8415.

134. **Van Eycke, Y.-R.**, **J. Allard**, **I. Salmon**, **O. Debeir**, and **C. Decaestecker** (2017). Image processing in digital pathology: an opportunity to solve inter-batch variability of immunohistochemical staining. *Scientific Reports*, **7**.

135. **Veeling, B. S.**, **J. Linmans**, **J. Winkens**, **T. Cohen**, and **M. Welling** (2018). Rotation equivariant cnns for digital pathology.

136. **Veit, A.**, **N. Alldrin**, **G. Chechik**, **I. Krasin**, **A. Gupta**, and **S. Belongie**, Learning from noisy large-scale datasets with minimal supervision. *In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

137. **Verma, E.** and **V. Singh**, ISBI challenge 2019: Convolution neural networks for b-all cell classification. *In* **A. Gupta** and **R. Gupta** (eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*. Springer Singapore, 2019.

138. **Vincent, I.**, **K. Kwon**, **S. Lee**, and **K. Moon**, Acute lymphoid leukemia classification using two-step neural network classifier. *In 2015 21st Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV)*. 2015.

139. **Vogado, L. H.**, **R. M. Veras**, **F. H. Araujo**, **R. R. Silva**, *et al.* (2018). Leukemia diagnosis in blood slides using transfer learning in cnns and svm for classification. *Engineering Applications of Artificial Intelligence*, **72**, 415 – 422. ISSN 0952-1976.

140. **Vogado, L. H. S.**, **R. D. M. S. Veras**, **A. R. Andrade**, **F. H. D. D. Araujo**, *et al.*, Diagnosing leukemia in blood smear images using an ensemble of classifiers and pre-trained convolutional neural networks. *In 2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. 2017. ISSN 2377-5416.

141. **Vuola, A. O.**, **S. U. Akram**, and **J. Kannala**, Mask-RCNN and U-Net ensembled for nuclei segmentation. *In IEEE International Symposium on Biomedical Imaging*. 2019.

142. **Wang, W.**, **D. A. Taft**, **Y.-J. Chen**, **J. Zhang**, **C. T. Wallace**, **M. Xu**, **S. C. Watkins**, and **J. Xing** (2019). Learn to segment single cells with deep distance estimator and deep cell detector. *Computers in Biology and Medicine*, **108**, 133 – 141.

143. **Wong, K. C.**, **T. Syeda-Mahmood**, and **M. Moradi** (2018). Building medical image classifiers with very limited data using segmentation networks. *Medical Image Analysis*, **49**, 105 – 116. ISSN 1361-8415.

144. **World Health Organization** (2019). `https://www.who.int/news-room/fact-sheets/detail/cancer#`. Online; accessed 12 June 2019.

145. **Xiao, F.**, **R. Kuang**, **Z. Ou**, and **B. Xiong**, DeepMEN: Multi-model ensemble network for b-lymphoblast cell classification. *In* **A. Gupta** and **R. Gupta** (eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*. Springer Singapore, 2019.

146. **Xie, S.**, **R. Girshick**, **P. Dollár**, **Z. Tu**, and **K. He**, Aggregated residual transformations for deep neural networks. *In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

147. **Xie, X.**, **Y. Li**, **M. Zhang**, **Y. Wu**, and **L. Shen**, Multi-streams and multi-features for cell classification. *In* **A. Gupta** and **R. Gupta** (eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*. Springer Singapore, 2019.

148. **Xu, Y.**, **Z. Jia**, **Y. Ai**, **F. Zhang**, *et al.*, Deep convolutional activation features for large scale brain tumor histopathology image classification and segmentation. *In 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2015. ISSN 1520-6149.

149. **Xu, Z.** and **et al.**, US-Net for robust and efficient nuclei instance segmentation. *In IEEE International Symposium on Biomedical Imaging*. 2019.

150. **Yan, Z.**, **Y. Zhan**, **Z. Peng**, **S. Liao**, *et al.*, Bodypart recognition using multi-stage deep learning. *In Information Processing in Medical Imaging*. 2015. ISBN 978-3-319-19992-4.

151. **Yang, H.**, **X. Zhang**, **F. Yin**, and **C. Liu**, Robust classification with convolutional prototype learning. *In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018.

152. **Yang, X.**, **R. Kwitt**, **M. Styner**, and **M. Niethammer** (2017). Quicksilver: Fast predictive image registration – a deep learning approach. *NeuroImage*, **158**, 378 – 396. ISSN 1053-8119.

153. **Zanjani, F. G.**, **S. Zinger**, **B. E. Bejnordi**, **J. A. W. M. van der Laak**, and **P. H. N. de With**, Stain normalization of histopathology images using generative adversarial networks. *In 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. 2018. ISSN 1945-8452.

154. **Zhang, L.**, **Le Lu**, **I. Nogues**, **R. M. Summers**, **S. Liu**, and **J. Yao** (2017). DeepPap: Deep Convolutional Networks for Cervical Cell Classification. *IEEE Journal of Biomedical and Health Informatics*, **21**(6), 1633–1643.

155. **Zhao, J.**, **M. Zhang**, **Z. Zhou**, **J. Chu**, *et al.* (2017). Automatic detection and classification of leukocytes using convolutional neural networks. *Medical & Biological Engineering & Computing*, **55**(8), 1287–1301. ISSN 1741-0444.

156. **Zhengjun Pan**, **A. G. Rust**, and **H. Bolouri**, Image redundancy reduction for neural network classification using discrete cosine transforms. *In International Joint Conference on Neural Networks*, volume 3. 2000. ISSN 1098-7576.

157. **Zhou, Z.**, **M. M. R. Siddiquee**, **N. Tajbakhsh**, and **J. Liang** (2018). Unet++: A nested u-net architecture for medical image segmentation. *CoRR*, **abs/1807.10165**. URL `http://arxiv.org/abs/1807.10165`.

158. **Zhu, J.**, **T. Park**, **P. Isola**, and **A. A. Efros**, Unpaired image-to-image translation using cycle-consistent adversarial networks. *In 2017 IEEE International Conference on Computer Vision (ICCV)*. 2017.