Computational Science and Engineering
(International Master's Program)

Technische Universität München

Master's Thesis

Multi-Domain Learning for Multi-Output Classification of Red Blood Cells

Madhan Ravuru

Computational Science and Engineering
(International Master's Program)

Technische Universität München

Master's Thesis

Multi-Domain Learning for Multi-Output Classification of Red Blood Cells

Domänenübergreifendes Lernen für die multi-output Klassifizierung von roten Blutkörperchen

Author:	Madhan Ravuru
Examiner:	Prof. Dr. Slobodan Ilic
Submission Date:	July 18th, 2021

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

July 18th, 2021 Madhan Ravuru

Acknowledgments

Throughout many months of working on this thesis, I have received great amount of support and assistance from many people. I would like to take this opportunity to express my sincere appreciation and gratitude.

First of all, I would like to thank my supervisor Prof. Dr. Slobodan Ilic for letting me work on this project and his smart advises. My warmest thanks to my project manager Dr. Gaby Marquardt for all her support. My heartfelt thanks to my project team members, Dr. Frank Forster, Dr. Thomas Engel, Agnieszka Tomczak and Jens-Peter Brock for weekly meeting discussions and guidance. With their integral and critical thinking along with technical discussions, this thesis proved to be a great learning experience. A big thanks to Siemens Healthineers for providing me a productive work environment.

Finally, my grateful thanks to my family and friends for their constant motivation and encouragement throughout my years of study.

Abstract

Hematologists use their experience to identify Red Blood Cell (RBC) disorders. Manual examination of cells are time-consuming, labour-intensive and subjective. RBCs are highly deformable and vary in size, shape, hemoglobin distribution and inclusion, making manual classification tedious. Classification of unstained cells become even more problematic since the hematologists recognize features enhanced by the staining process.

This thesis investigates the use of Machine Learning models - particularly Deep Learning algorithms - to address above mentioned problem. Firstly, RBCs are detected from blood smear images and cropped. Then, the center cell is extracted from each crop using Segmentation, with special handling of touching and overlapping cells. The hand-crafted features are derived from extracted center cell crops using Feature Engineering and analyzed. Later, Multi-Output Classification is performed on the cell to predict its properties. Comparison is made between extracted and learnt features using Support Vector Machine (SVM) classifier. Finally, Domain Adaptation is used in supervised and unsupervised fashion to learn from stained and unstained images for cell classification. The proposed Image Recognition Artificial Intelligence (AI) can revolutionize medical diagnostics, thereby making it cheaper and more accessible.

Contents

1 Introduction and Background

Blood is a body fluid with four main components, namely, plasma, red blood cells (RBCs), white blood cells (WBCs) and platelets. RBCs, also called erythrocytes, are the most abundant cells in vertebrate blood. These contain hemoglobin, an iron-containing protein, which transports oxygen and nutrients to the cells and carries away carbon dioxide and other waste products from those same cells. WBCs, also known as leukocytes, protect the body from infection. Platelets, or thrombocytes, help in blood clotting process, to prevent excess blood loss. In this thesis, we will focus only on analysis of RBCs.

1.1 Motivation

Diagnosis of RBCs helps in early detection of diseases such as malaria and anaemia [1]. Traditional complete blood count (CBC) device only gives blood cells count and not morphology of cells. In case of blood cell disorders, only a subset of all cells are morphologically altered and relevant for diagnosis. Detecting abnormality in RBCs needs microscopic examination of peripheral blood smear by hematologist. It takes several years of medical training to correctly distinguish blood cells. It is often arduous and time-consuming process besides that human error risk is high. Analysis of cells is also subjective for difficult cases. The demand for highly qualified experts far exceeds the available supply and in turn, delays life-saving patient diagnostics. Commercial systems, like CellaVision®, can detect RBCs from blood smear and do pre-classification of cell types. However, classification is human-assisted, as it requires confirmation by a skilled operator [2].

1.2 Problem Statement

Automatic classification of blood cells is very important in the field of medical diagnostics. It is achieved by employing Machine Learning algorithms. These algorithms try to mimic the way hematologists examine cells. Examination of cells boils down to determining its properties, namely, size, shape, hemoglobin distribution and inclusions, which give insights on blood cell disorder. A major challenge in constructing reliable and robust deep learning model is the lack of labelled data. Also, the annotated data is subjective, which will confuse the model.

1.3 Preliminaries

1.3.1 Biological Background

1.3.1.1 Blood film preparation and Staining

A blood film, also called peripheral blood smear, is a thin layer of blood smeared on a glass microscope slide. It is prepared by placing a drop of blood on one end of slide. Spreader slide is advanced with a smooth, steady motion so that a thin film of blood is spread over the slide. This ensures that the cells are well separated and get a region, called a monolayer. Then, the slide is left to air dry and later blood film is fixed by immersing in methanol. Later, the slide is stained using May–Grünwald–Giemsa stain to distinguish the cells from each other and to highlight important cellular features. There are many protocols for staining, which are different across hospitals. Therefore, there is little consistency between laboratories in the staining process [1]. Staining is necessary as the unstained smears have fewer details than stained smears and in turn, hematologists find it difficult to analyze unstained films. After staining, the blood smear is viewed under a microscope using magnification up to 1000x. The cells are examined individually and their morphology is characterized and recorded.

1.3.1.2 Microscopy types

Two different types of microscopy, viz. Bright Field (BF) microscopy and Differential Interference Contrast (DIC) microscopy are of interest [3]. BF microscopy is one of the simplest optical microscopy. The specimen is transmitted by illumination light and the contrast is generated by the absorption of light in dense areas of the sample. In our case, the specimen is stained blood smear. On the other hand, DIC microscopy is a light microscopic technique based on an interference principle involving two coherent beams of light. The gradients in optical path help in enhancing the contrast in unstained biological samples. The unstained blood smear is used as specimen to produce high resolution images using DIC microscopy.

1.3.2 Basics of Machine Learning

1.3.2.1 Feature Engineering Process

Feature engineering is the traditional process of transforming raw data into physical features. A feature is a individual measurable property on which analysis or prediction is done. Since the range of these hand-crafted features vary widely, they are normalized using feature scaling. The dimensionality of feature vector can be reduced by transforming data from a high-dimensional space into a low-dimensional space, which retains some meaningful properties of the original data. Dimensionality reduction methods are commonly divided into linear, viz. Principal Component Analysis (PCA) [4] and non-linear,

viz. T-distributed Stochastic Neighbor Embedding (t-SNE) [5] approaches. They are helpful in data visualization and clustering. Clustering is unsupervised machine learning algorithm, which learns patterns from unlabelled data. It works by grouping set of data points, such that points belonging to same cluster are more similar to each other than to those in other cluster. Many clustering algorithms [6, 7] exist depending on definition of a cluster. Supervised learning method such as Support Vector Machines (SVMs) [8] can be applied on extracted features to perform classification. The objective of algorithm is to find a hyperplane with maximum margin in an N-dimensional space (N - the number of features) that distinctly classifies the data points.

1.3.2.2 Artificial Neural Network

Deep Learning is a subfield of Machine Learning. It began as an attempt to perform tasks that conventional algorithms had little success with. It is based on Artificial Neural Networks (ANNs), a computing paradigm inspired by the functioning of the human brain. ANN comprises a number of neurons connected together to form a directed and weighted computational graph. The network is trained to adjust its weights in response to the error between output and target data, using gradient-based optimization. It involves supervised learning as target labelled data is required for a neural network to learn the correlation between labels and data. The feature computation is an inherent part of training, unlike feature engineering.

1.3.2.3 Convolutional Neural Network

A Convolutional Neural Network (CNN or ConvNet) is a type of feed-forward ANN in which the connectivity patterns between its neurons is inspired by the organization of the human visual cortex [9]. They are also known as space invariant ANNs. CNNs consist of multiple layers of small neuron collections that process portions of the input image, called receptive fields. The shared-weight architecture of the convolution filters are slided along input features to produce feature maps. The network is trained to optimize the filters using backpropagation. With the increased processing power of modern GPUs as well as the availability of large-scale and diverse datasets, CNNs are capable of producing state-of-the-art results. The main applications of CNNs are related with image recognition tasks such as object detection, image segmentation and classification.

1.3.2.4 Object Detection

It is the task to detect instances of objects of a certain class within an image. It can be classified into two main types: one-stage methods (YOLO [10], SSD [11]) and two-stage methods (Faster R-CNN [12], Mask R-CNN [13]). One-stage methods prioritize inference speed, while Two-stage methods prioritize detection accuracy.

1.3.2.5 Image Segmentation

It involves localizing parts of an image together which belong to the same object class. It is a form of pixel-level prediction on image because each pixel is classified according to a particular category. The objective of segmentation is to change the representation of an image by partitioning to sets of pixels, that is more meaningful and easier to analyze. It is divided into two groups: semantic segmentation and instance segmentation. Semantic segmentation treats multiple objects of the same class as a single entity. The U-Net [14] architecture can be used for semantic segmentation. In contrast, Instance segmentation treats multiple objects of the same class as different individual instances. The Mask R-CNN [13] architecture is used for instance segmentation. There are many different loss formulations for segmentation problems. Generalized Dice Loss (GDL) [15] is a robust and accurate deep-learning loss function for highly unbalanced segmentations.

Generalized Dice Loss: The loss formulation as expressed in Equation 1.1 has been analyzed under a binary classification (foreground vs. background). Let R be the reference foreground segmentation with voxel values r_n, and P the predicted probabilistic map for the foreground label over N image elements p_n, with the background class probability being $1 - P$. The contribution of each label is corrected by the inverse of its volume, thus reducing the well known correlation between region size and dice coefficient (see Equation 4.8).

$$\begin{aligned} \mathrm{GDL} &= 1 - 2\frac{\sum_{l=1}^{2} w_l \sum_n r_{ln} p_{ln}}{\sum_{l=1}^{2} w_l \sum_n r_{ln} + p_{ln}} \\ &\text{where,} \\ w_l &= \frac{1}{\left(\sum_{n=1}^{N} r_{ln}\right)^2} \end{aligned} \tag{1.1}$$

1.3.2.6 Image Classification

The goal is to classify the image by assigning it to a specific class label. Typically, image classification refers to images in which only one object appears and is analyzed. But, object detection can involve both localization and classification tasks, and can analyze cases where multiple objects exist in an image. Some prominent network architectures for classification are ResNet [16], DenseNet [17], EfficientNet [18]. Functionality related to multi-learning classification problems [19], including multi-class, multi-label and multi-output classification are discussed in types of classification.

Types of Classification

- **Multi-Class Classification:** Traditional single-output learning paradigm, where each sample is assigned to only one label from set of target labels with more than two classes.
- **Multi-Label Classification:** Each sample is labelled with n labels from a set of possible target labels, where n can be 0 to possible target labels set inclusive. This can be thought of as predicting properties of a sample that are not mutually exclusive. There is no a priori knowledge regarding the output size.
- **Multi-Output Classification:** Each sample is predicted with a fixed-length set of labels that measure different concepts. It determines several properties of an object simultaneously, but the number of outputs is known a priori. It involves solving two or more separate (although related) classification problems concurrently within the same model.

Binary Cross Entropy Loss: Binary Cross Entropy (BCE) loss [20] is defined for binary class classification problem. It is expressed in Equation 1.2. The target variable, $t_n \in \{0, 1\}$ and y_n is predicted probability for a sample. The predicted probabilty is obtained by passing network output to *logistic sigmoid* function (refer Equation 1.3).

$$BCE(t, y) = -\frac{1}{N}\sum_{n=1}^{N}\{t_n \log(y_n) + (1 - t_n)\log(1 - y_n)\} \tag{1.2}$$

Logistic Sigmoid function:

$$\sigma(a) = \frac{1}{1 + \exp(a)} \tag{1.3}$$

Cross Entropy Loss: Cross Entropy (CE) loss [20] for Multi-Class classification problem is shown in Equation 1.4. Here, N is batch size and K is number of classes. There exists a one-hot encoded $N \times K$ matrix of target variables with elements t_{nk} and y_{nk} represents predicted probability of class k for a sample n. The predicted probability is obtained by passing network output through *softmax* function as indicated in Equation 1.5.

$$CE(t, y) = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log(y_{nk}) \tag{1.4}$$

Softmax function:

$$y_k = \frac{\exp(a_k)}{\sum_{j=1}^{K} \exp(a_j)} \tag{1.5}$$

In case of class imbalance, *weight* argument is specified. Now, the loss becomes weighted average across observations. Median frequency balancing [21] method can be used to calculate class weights. The weight assigned to a particular class is the ratio of the median of class frequencies calculated from the entire training set divided by the class frequency [22]. This ensures that classes with high frequency have weight less than 1 and low frequency classes have bigger weight. The Weighted Cross Entropy (WCE) loss is formulated as given in Equation 1.6.

$$CE_{\text{weighted}}(t, y) = -\frac{\sum_{n=1}^{N}\sum_{k=1}^{K} w_k\, t_{nk} \log(y_{nk})}{\sum_{n=1}^{N}\sum_{k=1}^{K} w_k\, t_{nk}} \tag{1.6}$$

1.3.2.7 Domain Adaptation

Machine Learning algorithms are widely used in medical image analysis, and typically assume that the training dataset (source/reference domain) and test dataset (target domain) share the same data distribution. However, this assumption is too strong and may not hold true in real-world practice. This is referred to as the *domain shift* problem [23]. Domain adaptation aims to minimize distribution differences among different but related domains. It can be viewed as a special case of Transfer Learning. We have both source and target samples (possibly unlabelled) available during training. The goal is to learn from labelled data in a source domain that performs well on a different but related target domain. There are several contexts of domain adaptation. In supervised domain adaptation, both source and target labelled data are accessible. Unsupervised domain adaptation addresses the situation in which labelled source data and unlabelled target data are available [24]. The distance between source and target distributions can be calculated using standard distribution distance metric, Maximum Mean Discrepancy (MMD) [25].

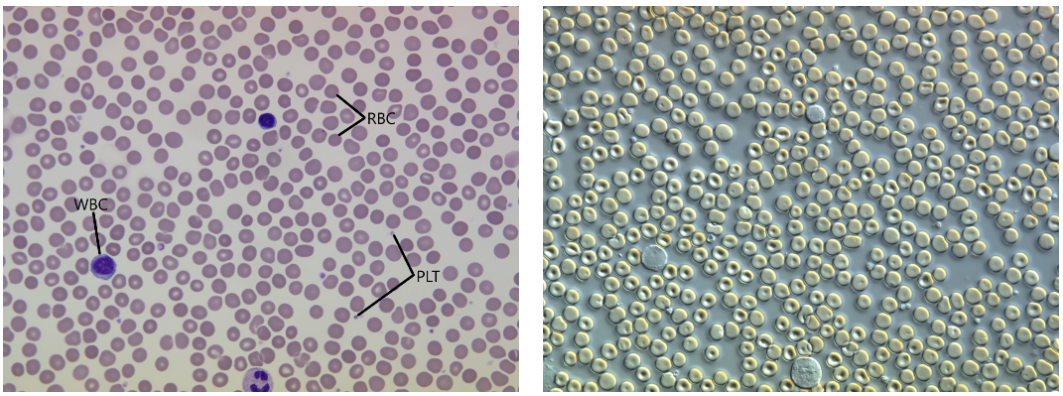
MMD Loss: Let us define a representation, $\phi(\cdot)$, which operates on source data points, $x_s \in X_S$, and target data points, $x_t \in X_T$. MMD distance is computed with respect to $\phi(\cdot)$. The empirical approximation to this distance [26] is given in Equation 1.7. Loss is formulated by minimizing this distance.

$$\text{MMD}(X_S, X_T) = \left\| \frac{1}{|X_S|} \sum_{x_s \in X_S} \phi(x_s) - \frac{1}{|X_T|} \sum_{x_t \in X_T} \phi(x_t) \right\| \tag{1.7}$$

1.4 Dataset Overview

The data consists of digitalized blood smear images of 17 patients obtained from three different hospitals. Blood smears are digitalized using the optical demonstrator of our future Integrated Smear Review (ISR) system. During conversion, the slide image is magnified upto 100x, in order to properly identify individual cells. The whole slide image is divided

into portions and each portion is analyzed. Figure 1.1 shows portion of slide images obtained from BF and DIC microscopy. Stained and Unstained images are obtained from BF and DIC microscopy respectively. From the shown microscopic view of a normal adult blood film, we can clearly see that cellular features are more visible in stained images than unstained images. The images are not aligned and slightly displaced. WBC, RBC and Platelet (PLT) are marked in the Figure 1.1a. As there are multitude of cells, the different cell types must be localized before they can be classified.



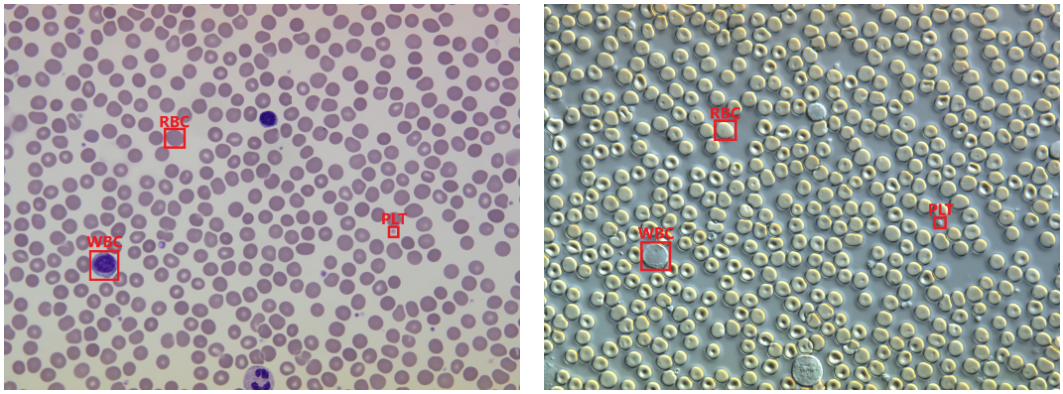
(a) Stained (BF) image (b) Unstained (DIC) image

Figure 1.1: Slide image

1.4.1 Cell Detection and Alignment

Detection of cells from slide images is important, as they are provided to either automated classifiers or hematology experts for cell classification. Deep Learning detector (YOLO [10]) is trained to detect relevant cellular structures (RBCs, WBCs, Platelets) from BF microscopic images and thus, tight bounding box is obtained enclosing each cell. We can see the cells with bounding boxes, each for WBC, RBC and platelet from Figure 1.2a. The size of bounding box will vary depending on cell size. Then, the crop is made from the middle of each bounding box. For the classification of WBCs, crop size is fixed at 256x256 pixels. The bigger crop is necessary, as WBC classification does not solely depend upon cell structure, but also on contextual information surrounding the cell. For RBCs, crop of common size 120x120 pixels is used to properly fit the cell.

Manual labelling of ground truth bounding boxes is a tedious task. It becomes even more difficult to do it for both domains. Also, because of absence of colour in unstained slide image, manual annotation is problematic. The detected bounding boxes from stained images are transferred to unstained images after image alignment. Both the images are aligned using Enhanced Correlation Coefficient (ECC) maximization [27] based on their gradients. The matching bounding boxes from both images can be seen in Figure 1.2. As the thesis



(a) Stained (BF) image (b) Unstained (DIC) image

Figure 1.2: Slide image with cell detection

focus is on RBC classification, only detection and matching of RBCs is of interest. The crop derived from each bounding box is named in a unique way based on its center inside particular slide image. The classification labels are prepared by hematologist upon analysis of stained crops. Now, we can use RBC classification cell labels of stained crops for unstained crop classification, as we have matching between the crops names. If not for matching, the preparation of ground truth classification labels for DIC crops would be very hard. Figure 1.3 shows matching RBC crops between BF and DIC.

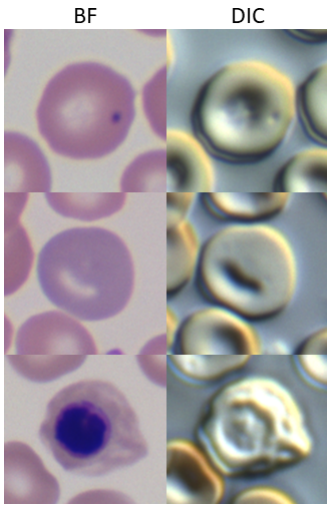


Figure 1.3: Aligned RBC crops

1.4.2 Classification Labels

RBCs have many variations in terms of size, shape, hemoglobin distribution and inclusions. Figure 1.4 shows the red blood cell morphology in terms of its property. A team of medical experts at Siemens Healthineers in Erlangen are responsible for preparing ground truth classification labels. These labels are prepared by looking at stained crops. As we have alignment between stained and unstained crops, we can safely assign the same labels to DIC crops as well. The displayed labels for cell properties in Figure 1.4 are obtained from our dataset.

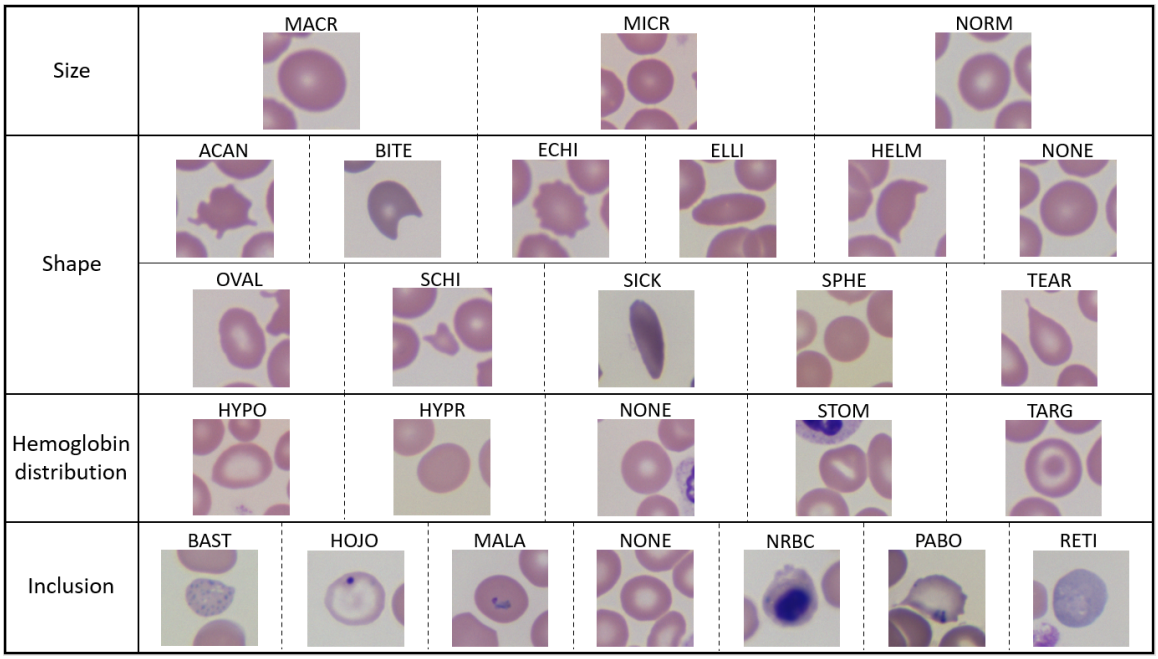


Figure 1.4: RBC Morphology

Cell size is categorized into 3 labels.

- MACR - Macrocyte
- MICR - Microcyte
- NORM - Normal

Cell shape is classified into 11 labels.

- ACAN - Acanthocyte
- BITE - Bite cell
- ECHI - Echinocyte
- ELLI - Elliptocyte

- HELM - Helmet cell
- NONE - Proper
- OVAL - Ovalocyte
- SCHI - Schistocyte
- SICK - Sickle cell
- SPHE - Spherocyte
- TEAR - Teardrop

Cell hemoglobin distribution is categorized into 5 labels.

- HYPO - Hypochromia
- HYPR - Hyperchromia
- NONE - Proper
- STOM - Stomatocyte
- TARG - Target cell

Inclusion inside cell is classified into 7 labels.

- BAST - Basophilic Stippling
- HOJO - Howell-Jolly-Body
- MALA - Malaria
- NONE - Proper
- NRBC - Nucleated Red Blood Cell
- PABO - Pappenheim Body
- RETI - Reticulocyte

The detailed description of each label can be referred from [1]. Please note that there can be two or more inclusion labels possible for the same crop.

1.5 Objectives and Contribution

Hematologists perform microscopic examination of size, shape and colouration of red blood cells. Machine Learning algorithms can learn to see patterns similar to the way medical experts see them. Firstly, the features are extracted using feature engineering and analyzed. Secondly, the features are learnt using deep learning algorithms. Finally, comparison is made between extracted and learnt features. The quantitative features from extracted center cell in the crop are derived. The information thus obtained convey biophysical meaning. Then, clustering and SVM classification of cell types using these features are performed.

Recently, Deep Learning algorithms have gained increasing attention from researchers for its potential to build automated diagnostic system for identification of blood cell disorders [28, 29, 30]. But, these supervised algorithms need a lot of concrete examples – many

thousands – in order to learn. We already know that manually assigning labels to cell is a labourious process. Also, supervised classifiers lack valuable quantitative information obtained as part of the feature extraction process. However, the performance of deep learning methods are much higher than traditional methods. The objective is to train multi-output classification model, which is capable to predict cell morphology in terms of size, shape, hemoglobin distribution and inclusion. The workflow for the approach is shown in Figure 1.5. Pre-Classification is performed to remove crops with bad cell extraction and artifacts.

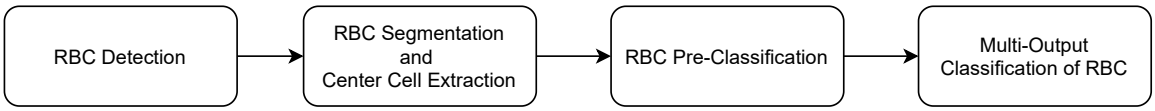


Figure 1.5: Workflow for Multi-Output classification

Matching BF and DIC crops helps in getting labels for DIC crops. Otherwise, labelling such crops are very hard. Usually, the classification performance with BF crops is more than DIC crops because of rich colour features produced by staining. As the matching crops are available, supervised domain adaptation is performed using feature alignment to improve DIC performance. Nevertheless, we can't always rely on matching the crops, to get ground truth DIC classification labels for training model. The data from stained and unstained domains come from different distributions but are related. So, unsupervised domain adaptation is used to get DIC labels by training with labelled BF and unlabelled DIC crops. BF and DIC images become our source and target data respectively.

The contributions of this thesis work are summarized as follows:

1. Derivation of quantitative features using feature engineering and analysis.
2. Performing multi-output classification on BF and DIC crops.
3. Supervised and unsupervised domain adaptation to learn from stained and unstained images for cell classification.

1.6 Thesis Outline

This thesis proposes multi-domain learning approach for multi-output classification of stained and unstained RBC crop images. It is structured into 6 chapters. Chapter 2 summarizes the related literature for different concepts involved. In Chapter 3, all methods which have been used in order to produce the results are discussed. Chapter 4 goes more in detail towards data preparation, data pre-processing and various experiments performed. Chapter 5 present the results from different experiments and analysis on them. Finally, Chapter 6 discuss about conclusions and future work.

2 Related Work

In this chapter, we address the related literature for different concepts: feature engineering, segmentation and cell extraction, classification and domain adaptation.

1. Feature Engineering

J. Bacus *et al.* [31] used quantitative RBC differential analysis to study blood cell disorders. Geometrical attributes extracted from RBCs were used to classify cell as normal and abnormal [32]. L. L. Wheeless *et al.* [33] selected single image analysis feature, *form factor* by recursive partitioning analysis to segregate cells. However, it lacks sufficient differentiation. C. L. Chen *et al.* [34] captures quantitative optical phase and intensity images and extract multiple biophysical features of individual cells for label-free cell classification. B. T. Grys *et al.* extract morphology and texture measurements using phenotypic profiling. [35]. Quantitative descriptors that characterize cell morphology are computed for label-free cell shape classification [36]. In this thesis, properties like cell size, shape and hemoglobin distribution are studied using respective quantitative feature extraction.

2. Segmentation and Cell Extraction

Most RBC segmentation methods use traditional image processing algorithms, such as thresholding [37], mathematical morphology [38, 39] and watershed [40, 41]. F. Al-Hafiz *et al.* [42] used thresholding and canny edge detector to segment RBCs. As there is little consistency in staining process among laboratories, varying colouration of the cells is possible. Threshold-based techniques typically fail for such cases. The segmented whole slide images can contain overlapping RBCs, which results in inaccurate classification. Hough Circle transform [43] was applied to detect disk-like overlapping RBCs and separate cells [44, 28]. K. Naruenatthanase *et al.* [45] used directed ellipse fitting using the detected concave points to separate overlapping cells. But, ellipse fitting can't be applied to cells with complex shapes. Deep neural networks were used for semantic segmentation of cells in blood smear images [46, 29]. These approaches assured new state-of-the-results, comparing with traditional algorithms. In their work, they extracted only cell of interest by cropping and masking, which removes background noise and is effective for further analysis. But, touching or overlapping RBCs are simply neglected and not analyzed. M. Zhang *et al.* [47] applied deformable convolution layers to the classic U-Net [14], to separate touching cells, discriminate the background noise and predict correct cell shapes without any shape priors. This thesis focus on deep learning based segmentation and post-processing to extract center cell.

3. Classification

C. Patgiri *et al.* [48] used naive bayes classifier and k-nearest neighbour classifier for RBC classification. Image processing and support vector machine was used to classify red blood cell morphology [49, 50]. R. Tomari *et al.* [32] extracted geometrical properties from RBCs and used neural networks to classify as normal and abnormal cells. Deep learning based approaches have been successfully applied to erythrocytes classification. Deep convolutional neural network (CNN) was used for detection of malaria [28, 29] and anaemia [44, 30] in RBCs. D. Mundhra *et al.* [37] presented automated peripheral blood smear analysis system that can classify RBC subtypes using CNN. T. J. Durant *et al.* [51] used DenseNet [17] as classification network to classify 10 red blood cell classes. Contrary to our work, none of the previous methods utilizes multi-output classification of erythrocytes, based on its properties.

4. Domain Adaptation

In the field of medical image analysis, unsupervised deep domain adaptation has gained prominence [24], as it does not require any labelled target data. Most works try to align source and target distributions by minimizing divergence that measures distance between distributions [52, 26, 53, 54]. Instead of minimizing divergence, M. Ghifary *et al.* [55] and K. Bousmalis *et al.* [56] used reconstruction network to align the distributions. Many adversarial domain adaptation methods have also been employed in literature. They learn a representation, which is simultaneously discriminative of source labels and not being able to distinguish between domains, using adversarial loss to minimize domain shift [57, 58, 59, 60]. In our work, the performance on target data is improved by feature alignment of source and target distributions similar to [26], but in a supervised setting. Then, gradient reversal algorithm is used as proposed in [59] for unsupervised domain adaptation, which directly maximizes the loss of the domain classifier by reversing its gradients.

3 Methodology

In this chapter, quantitative features extracted from feature engineering are studied. Then, multi-output classification is presented for prediction of cell properties. Finally, domain adaptation is used to learn from stained and unstained domain for cell classification.

3.1 Feature Engineering and Analysis

This section deals with extraction of physical features from center cell extracted crops and investigating them. Quantitative descriptors that characterize cell morphology in terms of size, shape and hemoglobin distribution are computed. Although the domain (whether the cell is stained or unstained) does not matter to study features from size and shape, it is of great importance for features extracted from hemoglobin distribution. As the colour features are more visible in BF crops, they are used for analysis. The extracted features contain information that can be easily interpreted by hematologists. On the other hand, deep learning features lack biophysical meaning and are hard to interpret. Also, deep learning methods require lot of annotated data for good model performance. They get better and better with increase in training data. In context of cell classification, getting labelled data for training requires significant time and effort. This is the major drawback of supervised methods.

The hand-picked features are visualized using dimensionality reduction. Then, clustering of cell types for each cell variation is performed separately. Unsupervised learning algorithms are more challenging to design and implement, but have the advantage that they are self-contained and don't require expert curated training data. They exploit the patterns hidden in the quantitative features under the premise that similar data should correspond to similar outcomes. However, performance of unsupervised methods is low as compared to supervised learning methods. This is due to lack of consensus in class delineation, limitations in feature extraction, sensitivity to algorithm parameters and lack of robustness in cluster identification [61]. So, Support Vector Machines (SVMs), a classical machine learning method is used to perform classification based on extracted physical features in a supervised manner. The classification is performed separately for each variation using selective features. They are selected, keeping in mind how hematologists analyze cell properties. The results using SVM are compared to deep learning results in the end.

3.1.1 Hand-Crafting Features

In this section, relevant features are derived to estimate cell size, shape and hemoglobin distribution. The features are quite sensitive to successful segmentation and extraction of center cell. So, features are extracted only for the good center cell crops, obtained from the pre-classification of images (refer Section 4.2.1).

3.1.1.1 Size Attributes

The features that maintain homogeneity in cell size within clusters is of interest. In general, cell size can be normal, macrocyte or microcyte. The average diameter of normal RBC is about 7.5 µm [1]. Geometrical features like area and perimeter are good choice to separate cells in terms of size. Ellipse, circle and rotated rectangle are fitted to cell boundary, which are universally applicable to all planar objects [36]. Ellipse and circle fit is performed using method of least squares, such that the distance between closest points on the ellipse and the cell contour points are minimized. Figure 3.1 shows geometrical shapes fitted to cell boundary.

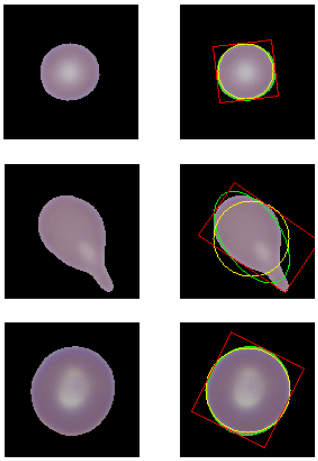


Figure 3.1: Ellipse (in green), Circle (in yellow) and Oriented Rectangle (in red) fitted to cell outline

Now, features are extracted from the fitted geometrical shapes. The derived attributes are translation and rotation invariant, which is required for characterization of cell size. Altogether, 10 dimensional feature vector is designed as listed below.

- **Region Area:** The area of center cell
- **Contour Area:** The area of center cell contour
- **Ellipse Major Axis:** The length of major axis of fitted ellipse

- **Ellipse Minor Axis:** The length of minor axis of fitted ellipse
- **Ellipse Area:** The area of fitted ellipse
- **Ellipse Perimeter:** The perimeter of fitted ellipse
- **Circle Radius:** The radius of circle fit
- **Circle Area:** The area of circle fit
- **Maximum Feret Diameter:** Longest distance between two parallel tangents on cell boundary obtained from oriented rectangle fit
- **Minimum Feret Diameter:** Shortest distance between two parallel tangents on cell boundary obtained from oriented rectangle fit

3.1.1.2 Shape Attributes

RBCs vary in different shapes as shown in Figure 1.4. The features describing the shape should be scale invariant, so that we can have more heterogeneity in size within clusters. So, shape factors are considered as they are invariant to scale [36]. Shape matching features, such as Hu Moments [62] are widely used in pattern recognition. These moments are calculated from binary cell image to get features describing shape. Some geometrical features, such as ellipse eccentricity and goodness of fit measure, Circle goodness of fit measure are also considered. The features, which can quantify angles, corners and curves along the cell boundary give valuable information about cell shape. So, boundary based features are derived by cubic spline interpolation along the cell outline [36]. Overall, 22 dimensional feature descriptor is obtained. The discussed features are explained below.

Shape Factors: These factors include extent, solidity, compactness, elongation, circularity, and convexity. They are scale invariant non-dimensional quantities, which can be used to represent cell shape. Figure 3.2 shows the description regarding the used shape factors.

Feature	Range	Equation	Description
Extent	$[0, 1]$	$\frac{A_{\text{cell}}}{A_{\text{bounding box}}}$	Ratio of pixels belonging to segmented cell to pixels in the bounding box.
Solidity	$[0, 1]$	$\frac{A_{\text{cell}}}{A_{\text{convex}}}$	Ratio of pixels belonging to segmented cell to pixels in the convex hull.
Compactness	$[0, 1]$	$\frac{\sqrt{\frac{4(A_{\text{cell}})}{\pi}}}{\text{Max. Diameter}}$	Ratio of circular equivalent diameter to maximum Feret diameter.
Elongation	$[0, 1]$	$1 - \frac{\text{Min. Diameter}}{\text{Max. Diameter}}$	1 - Aspect Ratio. Close to 1 for elongated cells and close to 0 for circular cells.
Circularity	$[0, 1]$	$\sqrt{\frac{4\pi A_{\text{cell}}}{P_{\text{cell}}^2}}$	Degree of resemblance to a circle.
Convexity	$[0, 1]$	$\frac{P_{\text{convex hull}}}{P_{\text{cell}}}$	Ratio of the convex hull perimeter to the cell perimeter.

Figure 3.2: Shape factors [36]

Hu Moments (Binary Image): A set of seven moments that are invariant to translation, scale, and rotation are derived by Hu. These moments are obtained from the binary image of center cell. Binary image is used because the interior composition of cell is irrelevant to describe its shape. Upon variation of scale and rotation for image, sign changes in some moments are observed. However, magnitude is comparable and hence, absolute value of these moments are considered.

Geometric Features: The ellipse and circle fit to cell boundary is shown in Figure 3.1. Features such as eccentricity, goodness of fit from ellipse fit and goodness of fit measure from circle fit are obtained. The goodness of fit tells us how good the shape fits to the cell boundary. Normalized value is considered with lower value indicating better fit. These features help in describing cell shape, as they are scale invariant.

Boundary Features: These are the most important features in determining cell shape. A cubic spline interpolation from contour points along the cell boundary is computed. Curvature values along the cell outline are calculated from the first and second order derivative of the spline at 500 uniformly spaced points. Mean, standard deviation, global maximum and global minimum of curvature values are included in the feature vector. The number of local maxima and local minima (with values above 0.2 and below -0.2 respectively) is also recorded to find the number of protrusions and indentations in the cell boundary. To avoid small arbitrary fluctuations in curvature computation, an additional constraint that no two maxima or minima can be present within a neighbourhood of 10 pixels is imposed. The features explained are represented in Figure 3.3.

Feature	Description
Mean	Mean curvature along the boundary of the cell.
Standard deviation	Indicates variation in curvature along the boundary.
Number of protrusions	Number of local maxima in curvature function, with peak values above the 0.2 per pixel threshold.
Number of indentations	Number of local minima in curvature function, computed by calculating maxima of the negative.
Maximum curvature	Global maximum of curvature on cell boundary.
Minimum curvature	Global minimum of curvature on cell boundary.

Figure 3.3: Boundary features derived from cubic spline interpolation [36]

Figure 3.4 shows cubic spline fit to cell boundary colour coded by curvature. The colour bar showing curvature values is also displayed. The first image is Spherical shaped cell and has positive curvature along its boundary with zero protrusions or indentations. In second image, we can see Teardrop cell with one single drawout at one side of cell membrane. Thus, it has one protrusion, which is global maximum of curvature along cell boundary. Elliptical cell can be seen in third image, with two local maxima of curvature at the extremities. However, the observed local maxima values are less than threshold value of 0.2. Consequently, we will have no protrusions. The choice of threshold values play important role in determining number of protrusions and indentations. The fourth image is Echinocyte, with thorns evenly distributed along the edge of cell. As a result, the curvature along its cell boundary varies a lot, which can be seen from Figure 3.5. It has 4 protrusions and 1 indentation. We can clearly see that the extracted boundary features convey important information describing shape of cell.

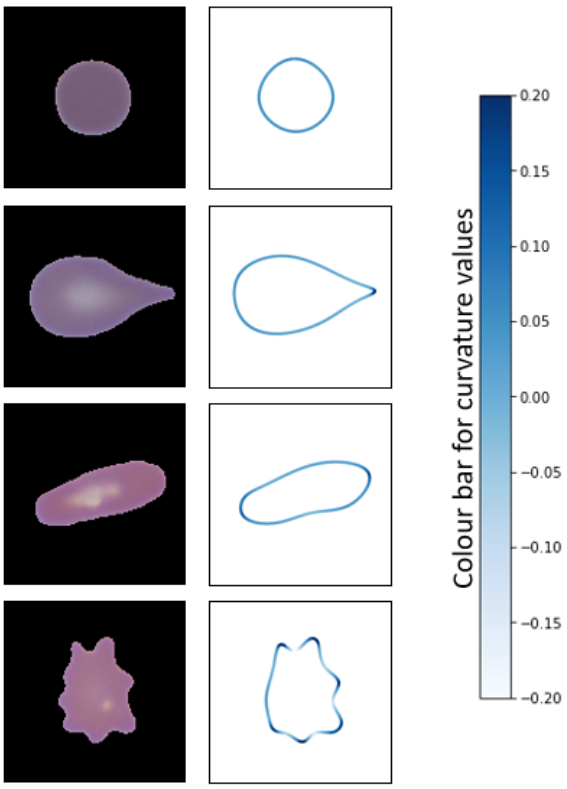


Figure 3.4: Cubic spline fit along the cell boundary colour coded by curvature

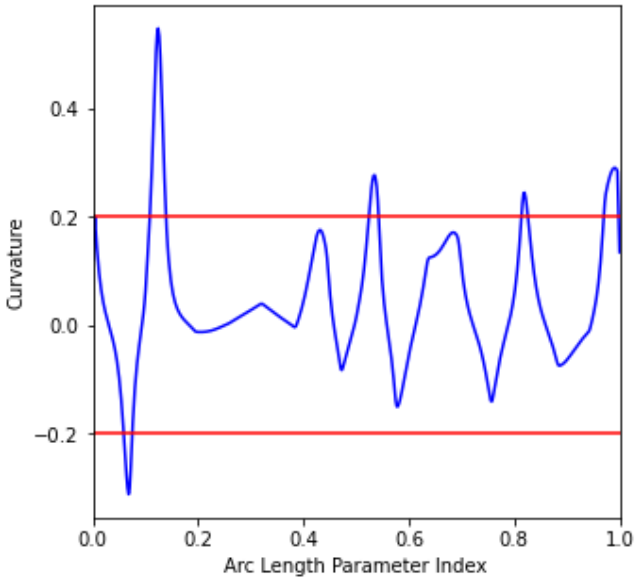


Figure 3.5: Curvature along the cell boundary (shown in blue) of Echinocyte obtained from cubic spline fit. The red lines indicate the per pixel threshold values

3.1.1.3 Hemoglobin Distribution Attributes

The features that can represent interior composition of cell is of importance to determine cell hemoglobin distribution. The ratio between whitening and hemoglobin inside cell distinguishes Hypochromia, Hyperchromia and Proper cell. The appearance of whitening within cell helps to identify Stomatocyte and Target cell (See Figure 1.4). So, Hu Moments on grayscale image are considered. The cell can be proper with respect to hemoglobin distribution in spite of inclusions within it. This resulted in including colour features in our feature vector. Altogether, 21 dimensional feature vector is constructed.

Hu Moments (Grayscale Image): These are set of seven moments derived by Hu. They are derived from grayscale image of center cell, as internal cell composition is of interest. Absolute value of these moments are taken into account. Moments can vary based on location of different pixel intensity. This is needed to uniquely distinguish different hemoglobin distribution composition inside cell.

Colour Features: Mean and standard deviation of Grayscale, RGB and HSV image of center cell are included as part of colour features. For each channel of image, mean and standard deviation can be calculated. As a result, 14 features describing colour of cell are obtained. These features don't convey information regarding the placement of pixel intensities, unlike Hu Moments.

3.1.2 Clustering

The extracted feature vectors contain information that describe the characteristics of cells. The task is to group cells based on feature vector such that cells present in the same group are more similar to each other than to those in other groups. Once the hand-crafted features are obtained for each cell variation, they are standardized by removing the mean and scaling to unit variance to get feature values in same range. Standardization prevents variables with larger scales from dominating how clusters are defined. The obtained feature vectors for size, shape and hemoglobin distribution have 10, 22 and 21 dimensions accordingly. To visualize the extracted features, dimensionality reduction can be performed separately for each cell variation. Then, clustering algorithms are applied on different feature set corresponding to each cell property. It is performed on low-dimensional data set obtained after dimensionality reduction, since in high-dimensional space, distances between pairs of points tend to converge to similar values. The clustering algorithm used is HDBSCAN (Hierarchical Density-based Spatial Clustering of Applications with Noise) [7]. It takes only MinPts (minimum points in a cluster) as input. The steps involved [63] in HDBSCAN algorithm are:

1. Computation of core-distances (and mutual reachability distances)
2. Computation of a minimum spanning tree (MST) used for single linkage computation
3. Tree condensing

It performs DBSCAN [64] over varying epsilon values and integrates the result to find a clustering that gives the best stability over epsilon. Epsilon is a parameter specifying the radius of a neighborhood with respect to some point. The main advantage of this method in comparison to famous k-means algorithm [6] is that, it has the ability to distinguish outliers inside clusters.

3.1.3 SVM Classification

Unsupervised classification is hard because of lack of agreement in class delineation and failure to identify clusters [61]. So, SVM is used to perform cell classification for each variation based on respective extracted features. The computed feature vector is concise and computationally inexpensive, in order to achieve high-throughput cell classification. It doesn't require lot of training data as compared to deep learning methods.

Let us discuss about SVM by defining two-class classification problem using linear models of the form [20]

$$y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}) + b \tag{3.1}$$

where $\phi(\mathbf{x})$ denotes a fixed feature-space transformation, $\mathbf{w}$ is normal vector to the hyperplane decision boundary and b is bias parameter. The training data set comprises N input

vectors $\mathbf{x}_1, ..., \mathbf{x}_N$, with corresponding target values $t_1, ..., t_N$ where $t_n \in \{-1, 1\}$, and new data points $\mathbf{x}$ are classified according to the sign of $y(\mathbf{x})$. Margin is defined as the smallest distance between the decision boundary and any of the samples. In SVM, the objective is to find a decision boundary such that the margin is maximized. This leads to the optimization problem [20] as defined below in Equation 3.2. Although the training involves nonlinear optimization, the objective function is convex, and so any local solution is also a global optimum.

$$\begin{aligned} &\arg\min_{\mathbf{w},b} \ \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to the constraints,} \\ &\qquad t_n(\mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n) + b) \geq 1, \qquad n = 1, ..., N \end{aligned} \tag{3.2}$$

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using kernel trick, which implicitly maps their inputs into high-dimensional feature spaces. The support vector machine is fundamentally a two-class classifier. But, more than two class labels are present for each cell property in our case. Multi-class classifiers can be easily obtained by using many binary classifiers. Various methods are proposed for combining multiple two-class SVMs in order to build a multiclass classifier. There are two well-known methods [20]: *one-versus-the-rest* and *one-versus-one*.

- *One-versus-the-rest* uses K classifiers if there are K different classes. Each of these classifiers is trained to separate one class from all the others. For prediction with test input, class label is obtained from the classifier with maximum prediction.
- *One-versus-one* trains a classifier for each pair of class. $K(K-1)/2$ different 2-class SVMs on all possible pairs of classes are trained. To classify test points, the class with highest number of votes is selected.

One-versus-the-rest approach suffers from the problem that different classifiers are trained on different tasks, and there is no guarantee that predictions from different classifiers will have appropriate scales. Also, the training sets become imbalanced. On the other hand, *one-versus-one* approach can lead to ambiguities in the resulting classification. For large K, more training time is required than the *one-versus-the-rest* approach. Similarly, to evaluate test points, significantly more computation is needed [20].

3.2 Multi-Output Classification

Typically, the performance with traditional methods using feature engineering are low in comparison to deep learning methods. So, deep learning based multi-output classification model is presented for cell classification. We have seen that many variations of RBCs exist (see Figure 1.4). Predicting these properties of cell is our classification task. Single multi-class classification cannot be performed in this case. The interest lies in multi-output classification (refer Section 1.3.2.6), in order to predict cell morphology. When compared to training different models for predicting cell properties separately, this can result in improved learning efficiency and accuracy [65]. In recent times, CNN based simultaneous localization and classification networks have gained prominence [13, 12]. This method can be used to detect and classify cell directly from blood smear image. However, such techniques are difficult to apply for our problem as a pretrained classification model is employed, typically one trained on the ImageNet dataset [66]. Different classifier heads are attached, depending on number of cell properties to the feature extractor of the pretrained model. The goal is to predict cell size, shape, hemoglobin distribution and inclusions simultaneously within the same model. The described multi-output classification network is depicted in Figure 3.6. The network consists of feature extractor, 3 multi-class classifier heads and 1 multi-label classifier head. All classifier heads are connected to feature extractor network. As multiple inclusions are possible for crop, we have multi-label classifier for predicting inclusion.

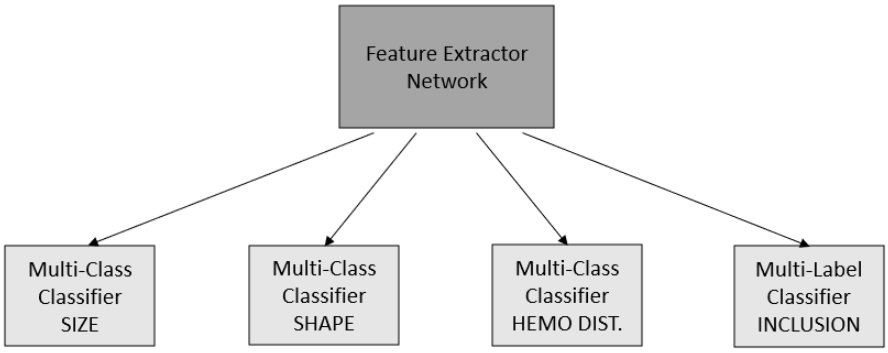


Figure 3.6: Multi-Output Classification network

3.2.1 Multi-Output Classification of BF crops

The objective is to predict several properties of cell for stained images within same network. The number of possible outputs for size, shape, hemoglobin distribution and inclusion are 3, 11, 5 and 7 respectively (see Figure 1.4). All these cell properties excluding

inclusion will have only one label as prediction. As there are multiple inclusions possible within cell, more than one cell label is possible here. Three different types of input, viz. full crop, center cell and merged to perform cell classification and results are compared. The center cell and full crop input along with ground truth labels for BF crops are shown in Figure 3.7. To get merged type input, concatenation of center cell and full crop is made along the channels and fed as input to network.

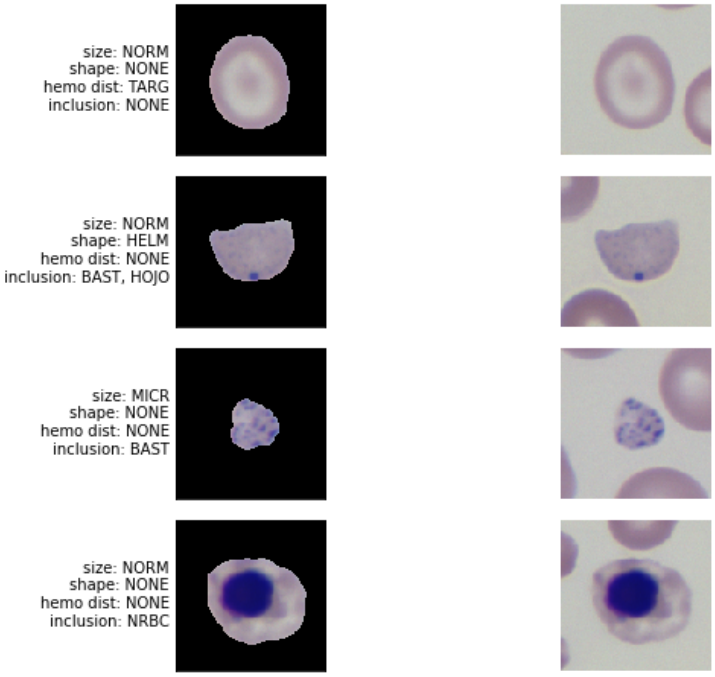


Figure 3.7: Stained crops with respective ground truth labels. Center cell crops and corresponding Full crops are shown on left and right side respectively.

The ground truth labels from Figure 3.7 clearly depict multi-output predictions for crop. Usually, there exists single prediction for each cell variation. But, there can be multiple inclusion labels possible for the crop. This is shown in second image of Figure 3.7. Prediction of inclusion can be thought as multiple binary classification subtasks. To train the presented complex network, linear combination of losses from different classifiers are used as total loss $\mathcal{L}_{\text{total}}$ (as shown in Equation 3.3). From the classification data preparation in Figure 4.10, class imbalance between labels inside each variation of train processed data can be seen. So, weighted classification losses, with weights for each classifier labels determined using median frequency balancing [21] are obtained. For size, shape and hemoglobin distribution, weighted cross entropy (WCE) loss is used and for inclusion, weighted binary cross entropy (WBCE) loss is used (refer Equation 1.6 and 1.2). Each classifier tries to optimize parameters in backbone network θ_E and parameters within it. As

the loss due to binary cross entropy is small in magnitude compared with cross entropy and to give more importance to inclusion variation, inclusion loss is increased to α times its value.

$$\mathcal{L}_{\text{total}}(\theta_E, \theta_{C_{\text{size}}}, \theta_{C_{\text{shape}}}, \theta_{C_{\text{hemo}}}, \theta_{C_{\text{inc}}}) = \mathcal{L}_{WCE(\text{size})}(\theta_E, \theta_{C_{\text{size}}}) + \mathcal{L}_{WCE(\text{shape})}(\theta_E, \theta_{C_{\text{shape}}}) + \mathcal{L}_{WCE(\text{hemo})}(\theta_E, \theta_{C_{\text{hemo}}}) + \alpha\mathcal{L}_{WBCE(\text{inc})}(\theta_E, \theta_{C_{\text{inc}}}) \tag{3.3}$$

3.2.2 Multi-Output Classification of DIC crops

The aim is to predict the cell properties for unstained images. As unstained images contain fewer details in comparison with stained images, it becomes challenging for network to achieve good performance. Figure 3.8 shows unstained crops with labels. The DIC crop labels are obtained by matching crop names with stained crops. The classification network is trained with 3 different input types similar to BF crops and the results are compared. The total loss function for training is same as shown in Equation 3.3.

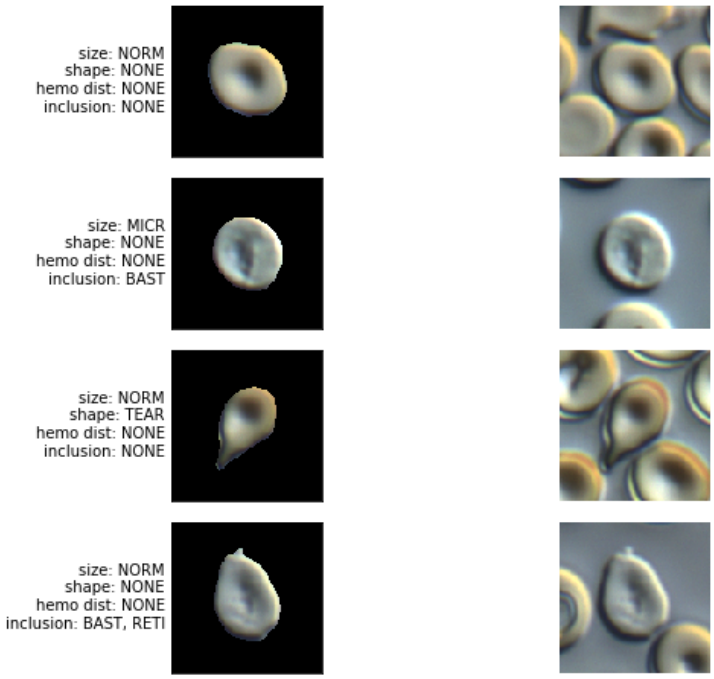


Figure 3.8: Unstained crops with respective ground truth labels. Center cell crops and respective full crops are displayed on left and right side accordingly.

3.3 Domain Adaptation

In this section, two different contexts of domain adaptation are discussed. Firstly, supervised domain adaptation is performed using feature alignment of source and target distributions in latent space. Source and target data are stained and unstained crops accordingly. The performance on DIC data is less in comparison to BF data, in terms of hemoglobin distribution and inclusion (see Table 5.1 and 5.4). In contrast, size and shape don't change a lot, as domain type doesn't matter for their prediction. Especially the F1 scores of inclusion labels drop significantly in case of DIC crops. This can be seen from comparison of Table 5.3 and 5.5. It is majorly due to absence of more details in cell composition of unstained images. As matching between the crops are available, the labels are present for both domains. The objective of this task is to improve DIC performance using network additional information from BF domain.

Secondly, unsupervised domain adaptation is carried out to predict DIC labels in presence of labelled BF data and unlabelled DIC data. The matching between crops should not always be the solution to get DIC labels. In this task, model trained on BF data is used in the context of a DIC data. With the progress in training, discriminative and domain-invariant deep features are emerged. The success in adaptation depends on level of relatedness between the source and target domains.

3.3.1 Supervised Domain Adaptation

The proposed network architecture can be seen from Figure 3.9. BF and DIC crops become source and target data respectively. It is similar to model proposed by E. Tzeng *et al.* [26], but used in supervised setting.

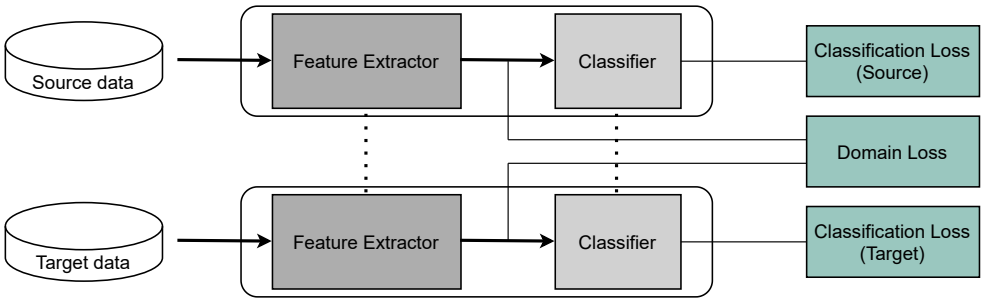


Figure 3.9: Classification network using feature alignment. It includes *feature extractor* and *classifier*. The dotted lines indicate that the network parameters are shared. Two classification loss terms are minimized for source and target samples and domain loss term directly minimizes the distance between source and target representations.

The shared network is fed with same sampled batch of crops from two different domains separately. This can be done as matching crops between domains are available. Then, the distance between the two feature representations in latent space is minimized using domain loss. The total loss of network is linear combination of classification losses from BF and DIC input and domain loss as shown in Equation 3.4. Standard distribution distance metric, MMD (refer Equation 1.7) is used as distance measure between the representations, z_{BF} and z_{DIC}. In Equation 3.4, y_{BF} and y_{DIC} are the predicted logits with BF and DIC input and t represent target labels. WCE loss (see Equation 1.6) is used for label classification, with weights determined from median frequency balancing [21] method. MMD loss is weighted by a factor α, which helps to find balance between minimizing feature distance and classification errors. The focus will be towards minimizing feature distance for α greater than 1. For α less than 1, minimization of classification losses will be given importance.

$$\mathcal{L}_{\mathrm{total}}(t, y_{\mathrm{BF}}, y_{\mathrm{DIC}}, z_{\mathrm{BF}}, z_{\mathrm{DIC}}) = \mathcal{L}_{WCE_{\mathrm{BF}}}(t, y_{\mathrm{BF}}) \; + \; \mathcal{L}_{WCE_{\mathrm{DIC}}}(t, y_{\mathrm{DIC}}) \; + \; \alpha \mathcal{L}_{MMD}(z_{\mathrm{BF}}, z_{\mathrm{DIC}}) \tag{3.4}$$

The model is trained to predict the class labels for both domains while simultaneously finding a representation that makes them indistinguishable. By minimizing the distance in latent space, the network is forced to learn domain-invariant feature representation. The input features are more visible in BF images than DIC images. The main task is to make DIC performance better for hemoglobin distribution and inclusion prediction using feature alignment. However, this works only if the features are present in DIC crops. It can be tested using Gradient Class Activation Map (Grad-CAM) [67], which helps in visualization of heatmap. This tells us what particular parts of the image the network is interested to assign a specific label. To find the importance of this label, the gradient of the corresponding logit with respect to the final convolutional layer of network is calculated. Then, the gradients are pooled channel-wise and weighed against the output of this convolutional layer. By inspecting these weighted activation channels, the regions in input image that played the most significant role in the class decision can be visualized.

3.3.2 Unsupervised Domain Adaptation

Domain-Adversarial Neural Network (DANN) [59] is used for unsupervised domain adaptation by backpropagation. The network consists of *feature extractor* and two classifier heads, namely, *label predictor* and *domain classifier*. The *label predictor* predicts class labels and *domain classifier* discriminates between source and target domains. Adaptation behaviour is achieved by placing *gradient reversal layer* between *feature extractor* and *domain classifier* that doesn't change the input during forward propagation, but reverses the gradient by multiplying it by a negative scalar during the backpropagation. Gradient reversal tries to match feature distributions over the two domains and make them as indistinguishable as possible for the domain classifier, thus resulting in emergence of domain-invariant

features. The DANN network architecture is shown in Figure 3.10.

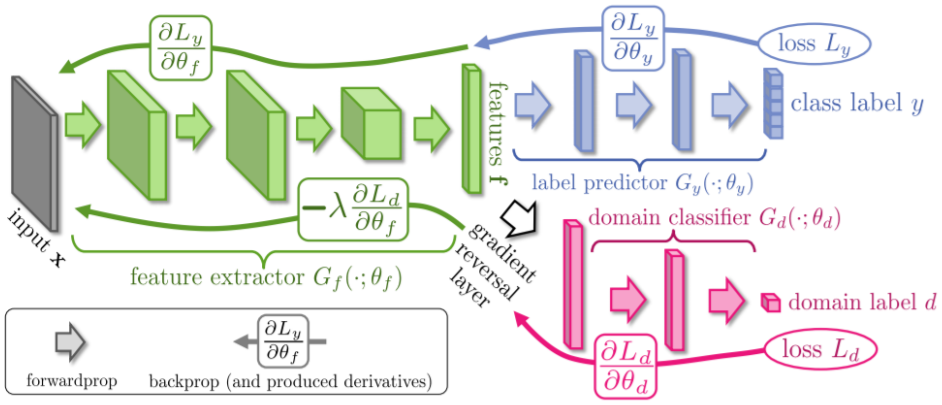


Figure 3.10: DANN network architecture [58]. It includes a deep *feature extractor* (green), a deep *label predictor* (blue) and a *domain classifier* (red) connected to the *feature extractor* through a *gradient reversal layer*.

This network consists of two losses, the classification loss ($\mathcal{L}_y$) and the domain confusion loss ($\mathcal{L}_d$). The classification loss is minimized for source samples and the domain confusion loss is minimized for all samples (while the domain confusion loss is maximized for feature extraction), ensures that the samples are made mutually indistinguishable for the domain classifier. The stochastic updates of parameters in *feature extractor* θ_f, *label predictor* θ_y and *domain classifier* θ_d are given by,

$$\theta_f \longleftarrow \theta_f - \mu\left(\frac{\partial \mathcal{L}_y^i}{\partial \theta_f} - \lambda\frac{\partial \mathcal{L}_d^i}{\partial \theta_f}\right) \tag{3.5}$$

$$\theta_y \longleftarrow \theta_y - \mu\frac{\partial \mathcal{L}_y^i}{\partial \theta_y} \tag{3.6}$$

$$\theta_d \longleftarrow \theta_d - \mu\frac{\partial \mathcal{L}_d^i}{\partial \theta_d} \tag{3.7}$$

where μ is the learning rate and λ is the adaptation factor. The latent feature space for multi-output scenario is very complex and to get domain-invariant features in that space is hard. The complex multi-output classification problem is broken down to multi-class problems. Three different DANN networks are trained for prediction of cell property individually. As inclusion prediction is multi-label, the network is not trained for that problem. The requirements for the latent feature space manifold are more demanding for multi-label than in the multi-class case due to more number of possible predictions [68]. The objective is to predict DIC labels for cell size, shape and hemoglobin distribution separately, using labelled BF and unlabelled DIC data.

4 Experiments

Firstly, this chapter deals with preparation of center cell crops using segmentation and cell extraction. Then, data preprocessing is performed to get the classification dataset for training. Finally, the implementation details for feature analysis, multi-output classification and domain adaptation are discussed. The deep learning models are trained using Python 3.8.5, PyTorch 1.7.1 [69] and Cudatoolkit 10.2.89 on Titan X (Pascal) GPU with 12GB RAM.

4.1 Center Cell Data Preparation

The crops are obtained from blood smear images using cell detection. But, the crops have noisy background information apart from cell of interest. Our goal is to extract only center cell from the crop. This has many advantages. Firstly, the contextual information other than cell of interest is irrelevant for cell classification [29]. This information is only redundant for classification network. Classification of cell can be better, in absence of noisy information other than cell of interest. Secondly, derivation of quantitative biophysical features from the cell using feature engineering is of interest. These features can only be obtained, only if cell of interest is inside the crop. Thirdly, feature alignment is performed to improve the classification performance on DIC crops. This can be made better, if the distributions of only center cell from BF and DIC crops are matched.

As cell of interest is exactly at the crop center, semantic segmentation can be performed and center contour mask can be obtained. This mask can be placed on input image to extract the cell of interest. But, this is only possible, if we have well separated cells inside crop. Unfortunately, touching or overlapping cells are present inside crops and extraction of center cell is not possible in this case. Most literature works neglect these type of crops and proceed further, as RBCs are abuntant. These crops might have disorders associated with them and neglecting such crops is not a good idea. So, an approach to extract center cell, even in case of touching or overlapping cells is proposed.

Two different U-Nets, namely, semantic segmentation and border segmentation network are trained. Semantic segmentation network does segmentation of cells in crop. Border segmentation network is trained to segment borders between the cells inside crop. Post-processing is performed using the results of the two networks in a unique way, to extract only center cell. The proposed approach is also experimented with MultiResUNet [70], modification of U-Net architecture. The observation is U-Net performed better than MultiResUNet.

4.1.1 Cell Segmentation

In biomedical image processing, the interest often lies in distinguishing interesting areas of the image. This can be done by localization to get desired output, i.e., assigning a class label to each pixel. The classical U-Net architecture [14] is used for this task, which is most prominent in this regard. Figure 4.1 shows the model, adapted from U-Net architecture. The model comprises of an encoder and a decoder pathway, with skip connections between the corresponding layers.

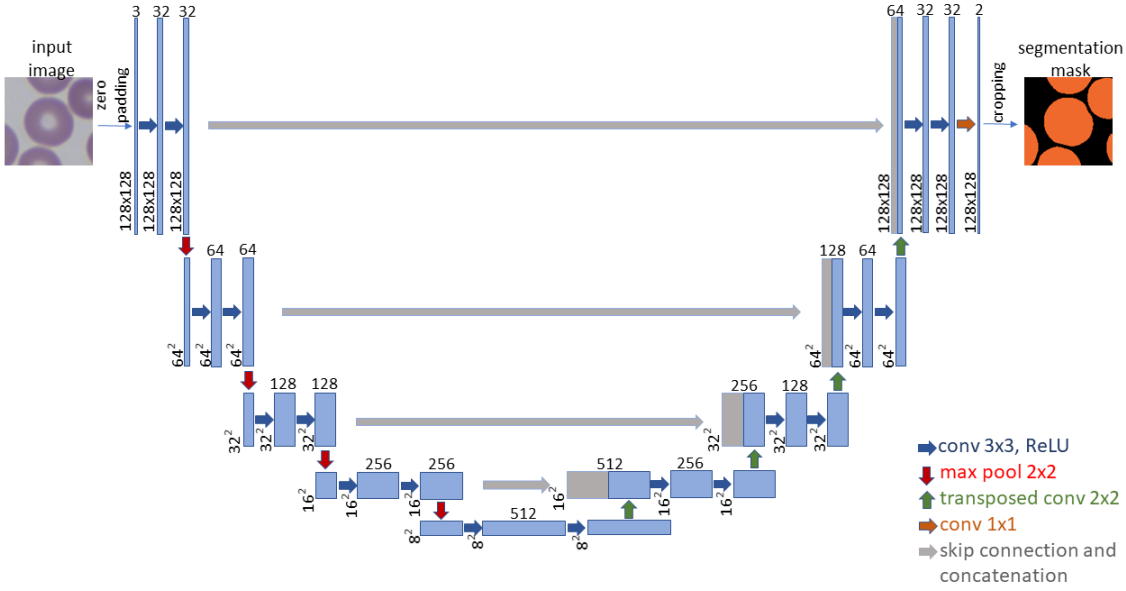


Figure 4.1: U-Net architecture. Multi-channel feature map is represented by blue box, with number of channels denoted on top. The spatial size is mentioned at lower left edge of the box. Grey boxes represent copied feature maps. The arrows denote different operations.

Model is trained to get output segmentation map for corresponding input image. The input data is stained RBC crop image detected from BF microscopy (see Figure 1.3). As we have 120x120 spatial input, skip connection and concatenation will be problematic in the high dimensional latent space, i.e, final layer of encoder. So, zero padding is applied along top, bottom, left and right of input image to get spatial size of 128x128 (closest power of 2). As a result, the output of decoder will have 128x128 spatial size. Cropping of output to 120x120 is necessary to match ground truth segmentation labels. This operation is differentiable and does not affect backpropagation.

4.1.1.1 Semantic Segmentation of cells

Goal The goal is to segment cells in the crop. The annotation data is prepared using pixel annotation tool. Initially the idea is to segment cells along with its composition (whitening, hemoglobin, inclusions, etc). Analysis can be drawn, if cell composition in the form of segmentation label predictions can be obtained. Different colours for neighbouring cells are present in annotated data (see Figure 4.2), which is helpful to get border between cells, to be discussed in Section 4.1.1.2. These colours are mapped to cell labels, which vary in terms of only cell composition. The performance is not accurate in predicting these labels. So, task is changed to binary segmentation. This means analysis regarding the cell composition will be part of classification network rather than segmentation network. Now, task consists of only two labels (cell mask and background). The mapped ground truth labels along with input can be seen in Figure 4.2. We can see that cell label is assigned to platelet as well as shown in second image. Both input image and ground truth are of spatial size 120x120 pixels.

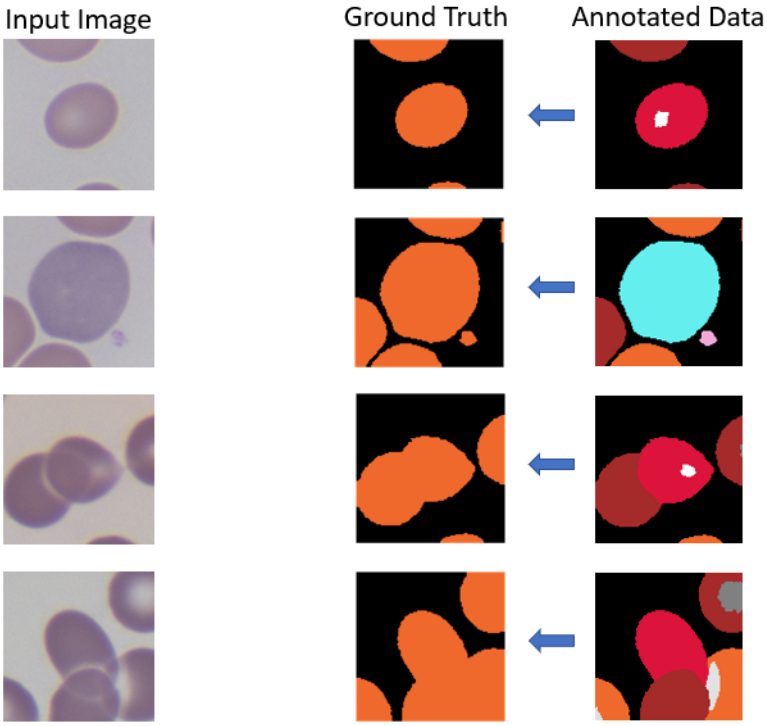


Figure 4.2: Training data for semantic segmentation network

Implementation The network architecture for training is the same as shown in Figure 4.1. The number of output channels are 2, as binary segmentation is performed. The network weights are initialized with Kaiming Initialization [71]. The training data is split into training and validation (80/20 split) to get 1584 and 396 samples respectively. The labelled classification data to perform multi-output classification will be test data. Training data is augmented with vertical flip, horizontal flip, random rotate by 90°, transpose and random brightness contrast all with probability of 0.5 to prevent over-fitting. Train data is standardized with its mean and standard deviation across RGB channels. Adam optimizer [72, 73] is used with learning rate and weight decay of 10^{-4} and 10^{-3} respectively. To get more robust and optimal segmentation model, linear combination of WCE (see Equation 1.6) loss and GDL [15] (see Equation 1.1) is used as loss function as shown in Equation 4.1. The model is trained to minimize total loss and get optimized parameter (θ) values of network. As class-imbalance between the two segmentation labels is present, weighted cross entropy loss instead of simple cross entropy loss is used.

$$\mathcal{L}_{\text{total}}(\theta) = \mathcal{L}_{WCE}(\theta) + \mathcal{L}_{GDL}(\theta) \tag{4.1}$$

Median frequency balancing [21] is employed to calculate class weights. For segmentation problems, class frequency is the number of pixels of class *c* divided by the total number of pixels in images where *c* is present. The batch size and number of epochs are set at 16 and 50 respectively. Dice Coefficient (as defined in Equation 4.8) on validation data is used as metric to check model performance during training.

4.1.1.2 Segmentation of border

Goal The goal here is to predict border between cells. The ground truth border masks are prepared using the annotation data. To distinguish two different cells, the cells are labelled with different colours in annotation data, particularly *R channel* value is different. But, the annotation data has so many colours, including different composition whitening colours for cell, as seen in Figure 4.3. Border between cells are taken into account, even if the whitening between two cells are close to each other. The good thing about having different colours in annotation data is, mapping can be made according to the need.

At first, only *R channel* masks from annotation data are used and dilation is performed with kernel of size 5. This dilation produce borders even if the cells are close to each other and not exactly touching, which results in over-estimation of border and doesn't cause any harm. This is better than not having border, if the cells are actually touching. Then, sliding window of 5x5 pixels is moved pixel-by-pixel, all through the annotation mask. The *R channel* values for different cells and their composition are known. If the unique value of pixels from sliding window contain different *R channel* values, the presence of border is indicated at the center pixel of sliding window. As a result, border pixels are obtained when the cells interact or present close-by. Finally, dilation is performed on these border pixels to get continuous border mask. The input image along with generated ground truths

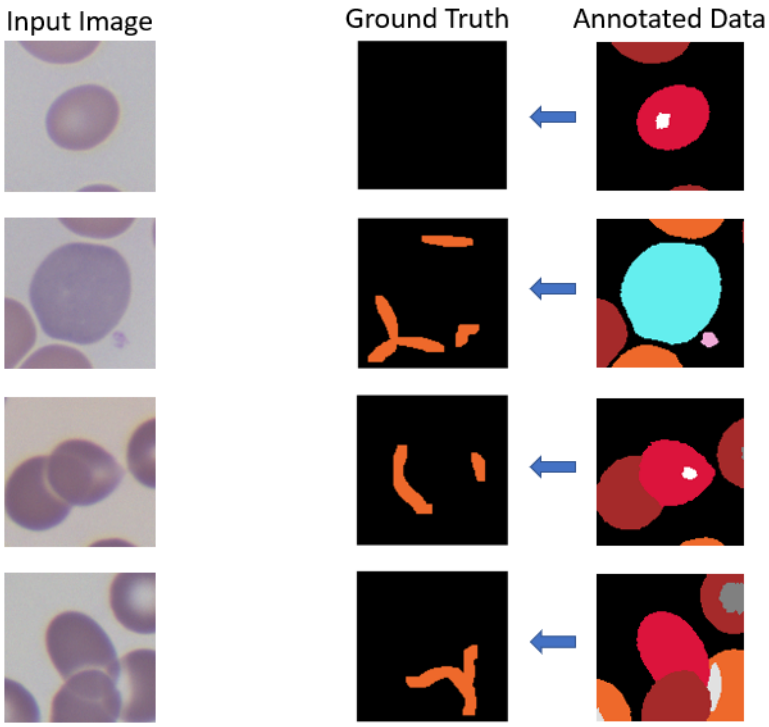


Figure 4.3: Training data for border segmentation network

border masks are shown in Figure 4.3. For the first image, we can see that there is no border mask present, as the cells are far apart. For the second image, though the cells are close to each other and actually not touching, border is marked. If the results of semantic segmentation network predicts non-separated masks for cells which are close-by, this over-estimation of border comes in handy, to separate the masks. For the third and fourth image, overlap exists between cells and the cell border is indicated in ground truth border mask.

Implementation The implementation of border segmentation network is exactly the same as semantic segmentation network, but only ground truth image is changed. The number of output channels are 2 (border mask, background). A different U-Net is trained from scratch with the same setting as described earlier in Section 4.1.1.1. Here, high class-imbalance between the segmentation class labels exists. Having the robust loss function, along with median frequency weight balancing, as discussed in Equation 4.1 makes more sense in this case.

4.1.2 Center Cell Extraction

The aim is to extract only center cell from the crop. Thus, noisy information present in the crop can be removed. This can be obtained by post-processing predicted semantic segmented image and border image. As matching pairs between BF and DIC crops are present, the center cell masks obtained from stained crops can be transferred to DIC crops. Hence, center cells from DIC crops can be obtained as well.

4.1.2.1 Splitting cells inside crop

Splitting of cells is necessary in case of touching and overlapping cells inside crop. To begin with, polynomial is fit to each contour present in border grayscale image. Then, these polynomials are used as inverse mask on grayscale segmented image. It results in splitting of cells inside segmented image crop. Algorithm 1 explains the adopted procedure to split cells. This algorithm is implemented by one of our project team member. It takes predicted border contour, degree of polynomial fit to contour, predicted border and segmented grayscale images as input. The output is to have segmented image with split cells inside it. This algorithm has to run as many times as number of predicted border contours. Finally, we will get segmented image with cells separated without any interaction.

Algorithm 1 Fit polynomial to contour

Require: c: contour, n: degree of polynomial, border image, segmented image
Ensure: Inverse masks on segmented image
1: Fit and transform contour points of c to new basis using 2 - component PCA
2: Get eigen values of PCA
3: Fit polynomial of degree n to transformed points
4: Find maximum error between transformed points and fitted polynomial
5: **if** second largest eigen value >50 **and** maximum error >15 **then**
6: Skeletonize the contour and dilate
7: Apply it as inverse mask on border and segmented image
8: **else**
9: Extrapolate the transformed points using fitted polynomial
10: Do inverse transform to get points on old basis
11: Apply it as inverse mask on border and segmented image
12: **end if**

For crops with well-separated cells, there will be no presence of border. We don't need to do anything in this case. The border segmentation network is trained for over-prediction of border, i.e, the border shows up, even if the cells are so close to each other but not actually touching. Figure 4.4 shows division of cells for touching and overlapping cases. Now, let's look into each case separately. The first segmented image shows cells, which are touching. There is border indicated as well, as a result of this interaction. If border contour area is

less than 500 pixels, polynomial of degree 2 is fit, else polynomial of degree 7. In this case, 2^{nd} degree polynomial is fit with extrapolation. Extrapolation is needed to ensure proper division between cells. It is then used as inverse mask on segmented grayscale image to split the two cells.

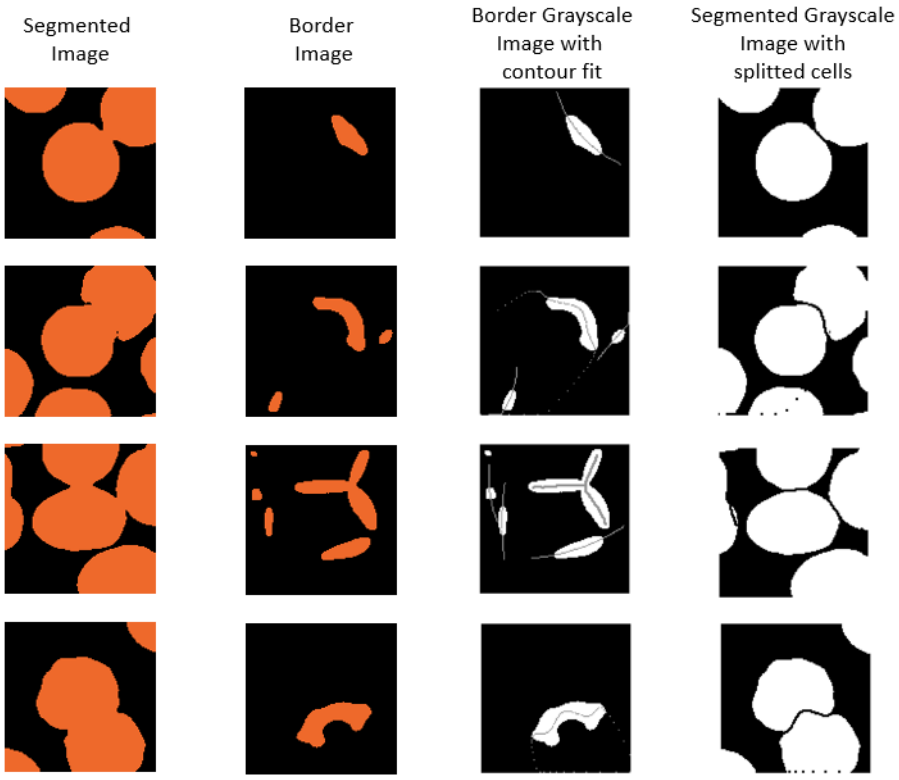


Figure 4.4: Splitting of cells inside crop

For the second image, we see there is a clear overlap between cells. As a result, a bigger border is shown. It is better to fit high degree polynomial for bigger contour borders in this case, such as 7. For the third image, we see multiple cells interaction with center cell. We can't properly fit any polynomial to multiple cell interaction border. So, better way to split is to skeletonize the contour and use it as inverse mask. The decision, whether to fit polynomial or skeletonize the contour is made based on eigen values from the principal component analysis (PCA) of contour points and maximum error between contour points and polynomial fit (see Algorithm 1). The fourth case illustrates splitting of cells depends on border prediction. Basically, overlap occurs for bad monolayer crops. In case of overlap, the center cell can be on top or bottom of other cell. Here, it is present under the neighbouring cell. So, when inverse mask is applied, part of center cell is lost.

4.1.2.2 Getting center cell

After splitting the cells, extraction of center cell mask from segmented grayscale image with splitted cells is performed using the famous point in polygon test [74]. Figure 4.5 shows the extracted center cells from input images.

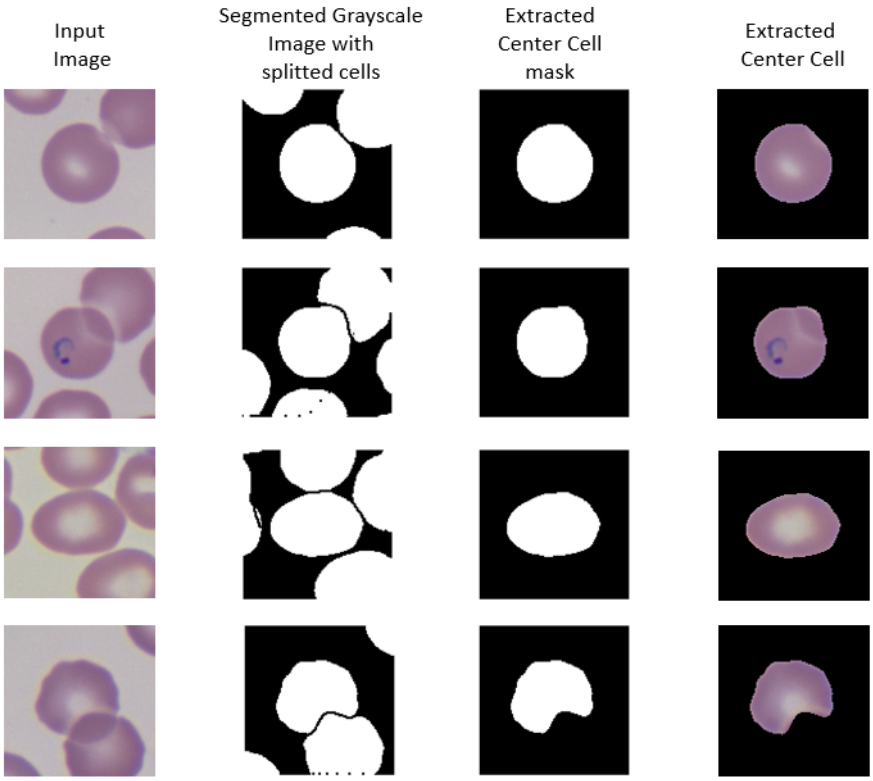


Figure 4.5: Extraction of center cell from crop

The crop center is checked whether it is inside, outside, or lies on an edge (or coincides with a vertex) of cell contour. The test returns positive (inside), negative (outside), or zero (on an edge) value, correspondingly as signed distance between the point and the nearest contour edge. As a result, the contour mask closest to crop center can be acquired. The inverse mask can sometimes run through the center cell mask. To fill the mask completely, dilation and erosion is carried out. The mask is placed on input image, to extract only center cell from it. The second image from Figure 4.5, shows malaria infected cell. Most literature works would not have used this crop further, as there is presence of overlap. Neglecting these type of critical crops would alter the diagnosis. But, this method is able to extract center cell successfully. In the fourth image from Figure 4.5, we can clearly see that part of center cell is lost. This is because the center cell is below neighbouring cell. The cell

extraction process heavily depends on border prediction. In some cases, the border might not be predicted as overlap is not so evident. As a result, the center cell cannot be obtained from crop. Also, artifacts in crops make it difficult to extract center cell. These are some failure cases for cell extraction using segmentation.

4.1.2.3 Combining the whole pipeline

Firstly, the results from semantic segmentation of cells (described in Section 4.1.1.1) and border segmentation (described in Section 4.1.1.2) are shown. The dice coefficient (refer Equation 4.8) for semantic segmentation and border segmentation on validation data is 0.9916 and 0.986 respectively. The predicted semantic and border masks for the input image are displayed in Figure 4.6. Secondly, the extracted center cells using the predicted segmentation masks are also shown.

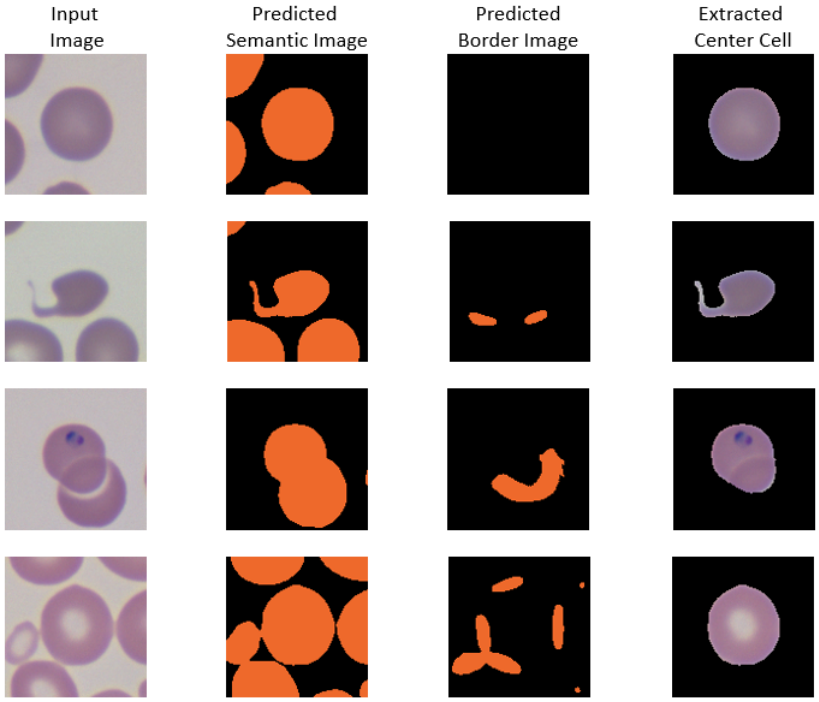


Figure 4.6: Segmentation and center cell extraction

In the first image crop from Figure 4.6, we observe that the cells are well apart. Extracting center cell will be easy for this case. From the second image, we can see that the cell structure is well preserved by predicted semantic mask. As the neighbouring cells are close to center cell, border is predicted by network. The obtained center cell looks very good, despite of complex cell structure. Third image clearly shows overlap between cells. In spite of overlap, center cell is extracted successfully. Most works won't consider such

overlapping cells for classification analysis. This is crucial, as the center cell is infected by malaria and neglecting such crops will affect diagnosis. In the fourth image, we can see that the cell of interest is touching other cells and center cell is obtained without any difficulty.

The center cell masks obtained from stained crops can be used on unstained crops because of matching between crops (See Figure 1.3). Thus, center cells from DIC crops can be obtained as well. Figure 4.7 shows the center cells extracted from DIC input images using center cells masks of stained crops.

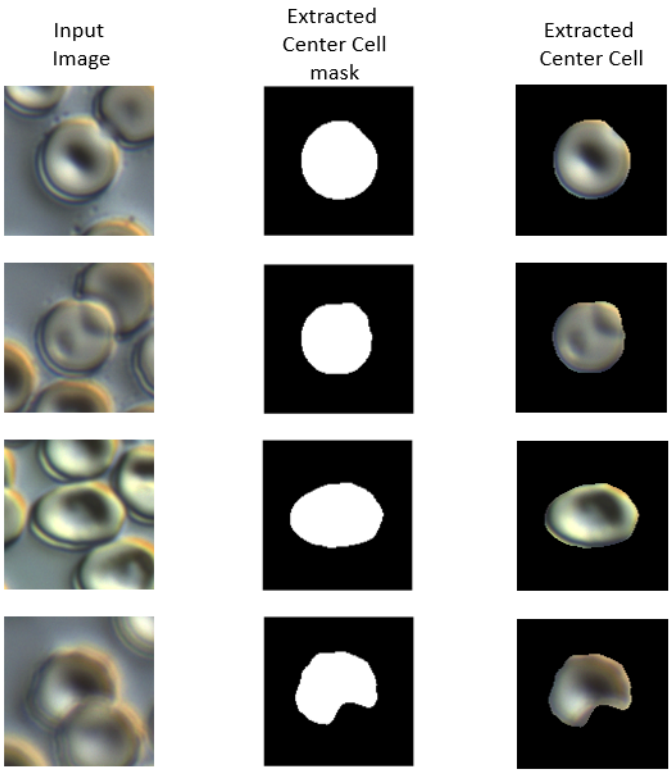


Figure 4.7: Extraction of center cell from DIC crop

4.2 Data Preprocessing

4.2.1 Pre-Classification

4.2.1.1 Importance

The center cell crops obtained from data preparation are not always proper. Also, the detected crops might have platelets on top of RBCs, artifacts and sometimes even WBC is detected as RBC. These crops are irrelevant for cell classification and need to be removed. So, pre-classification is performed to decide whether crop is good or bad and only good crops will be used later for classification. Also, the traditional systems based on feature engineering need to ensure successful extraction of center cell to derive quantitative features. Only the pre-classified good crops are used to obtain hand-crafted features. Some examples of good and bad crops are shown in Figure 4.8 and Figure 4.9 respectively. In Figure 4.8, we can see touching and overlapping cells inside crop. However, the extracted center cell is proper without losing its structure. So, these crops are labelled as good crops. In the overlapping case, the center cell is on top of neighbouring cell and thus, center cell is acquired without alteration of its shape.

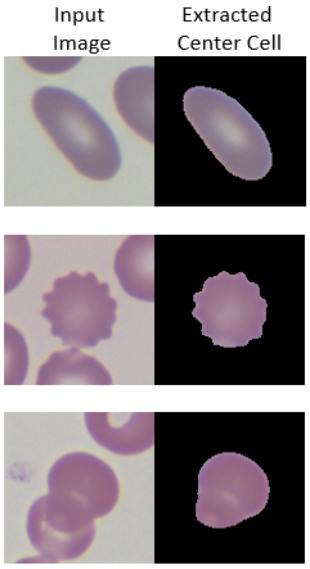


Figure 4.8: Examples of good crops. First input image show well separated cells. Second and third input image shows touching and overlapping cells. For all cases, center cell is extracted properly.

On the other hand, the cell extraction process might not be accurate in some crops with bad monolayer. In addition, the localized crops can have platelets on top of interested cell, artifacts and WBC. In case of overlap, the border might not be so evident and extraction of

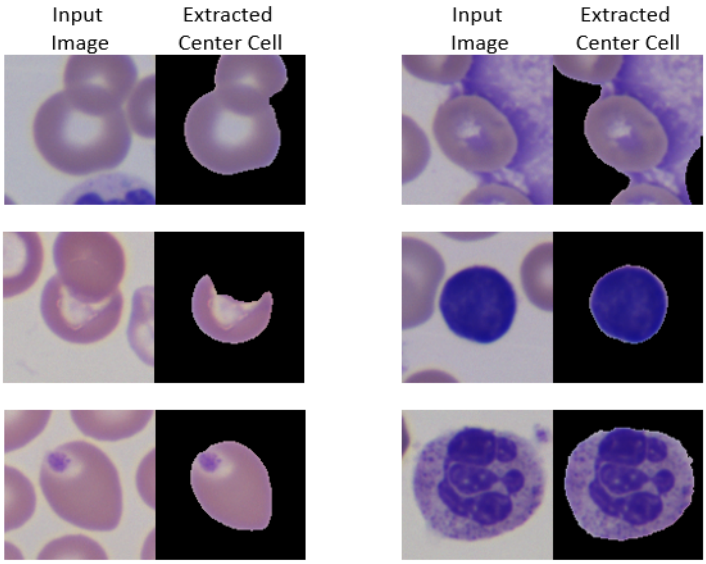


Figure 4.9: Examples of bad crops. First row on left show image with center cell not extracted. Second row on left show image where part of cell is lost. Cell with platelet on top is shown by third row on left image. First and second row on right shows image with artifact. Third row on right display WBC.

center cell is not feasible. Or, when center cell is below neighbouring cell, the morphology of cell will be changed in extraction process. These type of crops are illustrations of bad crops and are shown in Figure 4.9.

4.2.1.2 Implementation

The objective is to design binary classifier, which does pre-classification on crops. The pretrained EfficientNet-B7 [18] on ImageNet dataset [66] is used and classifier head is attached with number of output classes as 2 (good/bad). The input to network is concatenated with center cell and full crop along the channels. This input is called as merged type input. This is necessary because for overlapping cells with center cell at bottom, part of cell is lost during extraction. It is difficult to say whether crop is good or bad from center cell crop for such cases. The network can learn better, if the input is of merged type. Dataset is mostly comprised of unlabelled crops and some challenging labelled crops (such as Malaria, Nucleated RBCs, etc). These challenging crops should be included in training the model because they look like bad crops, but are actually not. The training data is divided into training and validation (80/20 split) to get 13657 and 3415 samples respectively. For each sample, full crop and its corresponding center cell crop is present. The ground truth label is prepared by manual inspection of crops as good (0) or bad (1). Augmenta-

tions are performed on training data with vertical flip, horizontal flip, random rotate by 90°, transpose, random brightness contrast all with probability of 0.5 and blur with 0.1 probability to prevent over-fitting. The center cell input is normalized in the range [-1,1] and full crop input is standardized with its mean and standard deviation. Adam optimizer [72, 73] is used with learning rate and weight decay of 10^{-4} and 10^{-3} respectively. To encounter class-imbalance, WBCE loss is used with median frequency balancing [21] to calculate class weights. The batch size and number of epochs are set at 16 and 30 respectively. The model is trained on BF merged type input. It is tested on BF dataset with multi-output labels. DIC good crops are obtained, by directly matching with the good classified BF crops. Now, multi-output classification can be performed for both BF and DIC domains using the good classified crops.

4.2.2 Classification Data Preparation

Medical experts labelled 13729 stained crops including artifacts. As some labelled crops are used to train pre-classification model, only 13648 labelled crops are present. After pre-classification, 12064 crops are classified as good. The artifacts in good crops are removed manually, and finally 11891 crops are left. These are divided into 9512 train and 2379 validation samples. As very few labelled crops are available, test data is not present. The train and validation data with label distribution are shown in Figure 4.10. Four different types of cell variations, namely size, shape, hemoglobin distribution and inclusion exist. Refer to Section 1.4.2 for description regarding labels. The inclusion variation can have multiple labels within it. The crop is given one label from each variation. From train data of Figure 4.10, we can see that there is heavy class imbalance among labels belonging to each variation. Random undersampling and oversampling of crops are carried out to get processed train data as seen in Figure 4.10. There is still imbalance between labels in train processed data. This imbalance can't be removed completely, as the crops have multiple labels and oversampling labels of one variation affect the labels in other variation. The train processed data is obtained from train data as described below. Almost half of train data is with normal crops. By normal, it means cell size is *NORM* and all other variations are of *NONE* type. Firstly, random undersampling is done to remove 50 % of normal crops from train data. Secondly, the minority labels under each variation are randonly oversampled to 200. The minimum labels in inclusion variation are kept at 400. Thus, 11528 train processed data are obtained from train data. The label distribution within 2379 validation data crops is also shown in Figure 4.10. For all variations, the crops are shown in decreasing order of their count.

Train Data

Size	count
NORM	8369
MICR	986
MACR	157

Shape	count
NONE	7655
ECHI	561
OVAL	460
TEAR	355
SCHI	108
HELM	103
ACAN	92
ELLI	84
SPHE	48
SICK	35
BITE	11

Hemo dist.	count
NONE	7953
HYPO	818
HYPR	341
TARG	305
STOM	95

Inclusion	count
NONE	8945
RETI	223
MALA	139
HOJO	81
BAST, RETI	49
BAST	28
PABO	25
NRBC	16
BAST, HOJO	2
HOJO, RETI	2
BAST, HOJO, RETI	2

Train Processed Data

Size	count
NORM	9597
MICR	1680
MACR	251

Shape	count
NONE	8051
OVAL	861
ECHI	575
HELM	474
TEAR	356
SICK	211
ACAN	200
ELLI	200
SPHE	200
SCHI	200
BITE	200

Hemo dist.	count
NONE	9642
HYPO	953
HYPR	398
TARG	334
STOM	201

Inclusion	count
NONE	7528
RETI	400
MALA	400
HOJO	400
BAST, RETI	400
BAST	400
PABO	400
NRBC	400
BAST, HOJO	400
HOJO, RETI	400
BAST, HOJO, RETI	400

Validation Data

Size	count
NORM	2091
MICR	249
MACR	39

Shape	count
NONE	1885
ECHI	150
OVAL	127
TEAR	87
SCHI	33
ACAN	30
ELLI	23
HELM	23
SPHE	11
SICK	7
BITE	3

Hemo dist.	count
NONE	1960
HYPO	222
HYPR	106
TARG	75
STOM	16

Inclusion	count
NONE	2252
RETI	50
MALA	25
BAST, RETI	18
HOJO	12
NRBC	7
PABO	6
BAST	5
PABO, RETI	1
HOJO, RETI	1
BAST, HOJO	1
BAST, HOJO, RETI	1

Figure 4.10: Classification labels distribution for BF dataset

Similar process is followed for DIC data preparation as well. Firstly, DIC crop labels are made available by matching the crop names with BF crops. The matching DIC crop is not always present for every BF crop. There are some border crops from slide images, which needs to be removed eventhough matching BF label is available for the crop. Instead of performing pre-classification on available DIC data, the good samples are directly obtained from matching the pre-classification results of BF crops. As a result, 11071 good DIC crops with labels are obtained. The artifacts in good crops are removed, finally 10915 crops are present. These crops are divided into train and validation crops to get 8732 and 2183 crops accordingly. Secondly, undersampling of normal crops and oversampling of minority labels is followed to get processed train crops. Finally, 11544 train processed and 2183 validation DIC crops are available. The label distributions inside each cell variation for train, train processed and validation DIC crops are shown in Figure 4.11

Train Data

Size	count
NORM	7690
MICR	893
MACR	149

Shape	count
NONE	6912
ECHI	570
OVAL	447
TEAR	347
SCHI	105
HELM	96
ACAN	95
ELLI	80
SPHE	41
SICK	30
BITE	9

Hemo dist.	count
NONE	7199
HYPO	809
HYPR	355
TARG	297
STOM	72

Inclusion	count
NONE	8208
RETI	202
MALA	121
HOJO	74
BAST, RETI	55
PABO	26
BAST	21
NRBC	18
BAST, HOJO, RETI	3
BAST, HOJO	2
HOJO, RETI	1
PABO, RETI	1

Train Processed Data

Size	count
NORM	9854
MICR	1414
MACR	276

Shape	count
NONE	8153
OVAL	1060
ECHI	582
TEAR	349
SCHI	200
HELM	200
ACAN	200
ELLI	200
SPHE	200
SICK	200
BITE	200

Hemo dist.	count
NONE	9672
HYPO	936
HYPR	432
TARG	299
STOM	205

Inclusion	count
NONE	7144
RETI	400
MALA	400
HOJO	400
BAST, RETI	400
PABO	400
BAST	400
NRBC	400
BAST, HOJO, RETI	400
BAST, HOJO	400
HOJO, RETI	400
PABO, RETI	400

Validation Data

Size	count
NORM	1927
MICR	226
MACR	30

Shape	count
NONE	1719
ECHI	126
OVAL	125
TEAR	76
SCHI	35
HELM	30
ACAN	25
ELLI	25
SPHE	13
SICK	7
BITE	2

Hemo dist.	count
NONE	1822
HYPO	186
TARG	78
HYPR	74
STOM	23

Inclusion	count
NONE	2053
RETI	54
MALA	30
HOJO	13
BAST	11
BAST, RETI	10
NRBC	5
PABO	4
HOJO, RETI	2
BAST, HOJO	1

Figure 4.11: Classification labels distribution for DIC dataset

4.3 Clustering and SVM Classification

The feature vectors for each cell property are calculated from the validation data from BF dataset with 2379 center cell crops (See Figure 4.10). Only validation data is considered because HDBSCAN algorithm used for clustering has $O(N^2)$ run-time on N data points [7]. Dimensionality reduction is performed on standardized feature vectors to obtain two components using PCA [4] and t-SNE [5]. The perplexity value in t-SNE algorithm is varied as 50, 5 and 20 for size, shape and hemoglobin distribution respectively. The perplexity is effective number of neighbours of a data point providing a weighting between the importance of local and global properties. The ground truth labels for each property are imposed on data points for better visualization. Then, HDBSCAN [7] algorithm is used on low dimensional feature set with *MinPts* parameter as 50 and 20 for size and shape accordingly.

For SVM classification, feature vectors are extracted from train and validation data of BF dataset (refer Figure 4.10) for each cell variation. Then, the whole feature vectors are standardized to bring feature values to same scale. The *one-versus-one* SVM classifier is trained with ground truth labels from size, shape and hemoglobin distribution individually. Polynomial kernel is employed with balanced class weights. Prediction is made on validation data and results of classification are recorded.

4.4 Multi-Output Classification

4.4.1 Implementation

The network for multi-output classification is shown in Figure 3.6. Pretrained EfficientNet-B7 [18] is used as backbone network and 4 different classifier heads for predicting cell variations are attached. Simple single class prediction is needed for size, shape and hemoglobin distribution. So, softmax layer (refer Equation 1.5) is added at the end of each classifier to get predicted probability vector. But for inclusion, sigmoid activation (see Equation 1.3) is added to get multiple predictions. For training with BF crops, the train processed and validation BF crops used are 11528 and 2379 respectively (see Figure 4.10). The train processed data have some redundant data in minority label crops due to oversampling. Augmentations (same as described in Section 4.2.1.2) are performed on train processed data, which augments repeated data to some extent and prevents over-fitting. The center cell input is normalized in the range [0,1] and full crop input is standardized with its mean and standard deviation. The merged type input is obtained by concatenating these inputs. The network is trained separately with all three types of input. The loss function to be optimized is shown in Equation 3.3. The weight factor for inclusion loss, α is used as 5. The initial learning rate and weight decay of 10^{-4} and 10^{-3} respectively are used with Adam optimizer [72, 73]. For training, batch size and number of epochs are set as 16 and 35 correspondingly. The learning rate is decreased to 0.1 times its initial value for last 5 epochs. Validation loss is used to ensure that the model doesn't overfit.

For training with DIC crops, the processed train and validation samples are 11544 and 2183 respectively (see Figure 4.11). The weights for each classifier labels in loss function are re-calculated using median frequency balancing [21]. The performance for different multi-output models trained with full crop, center cell and merged type DIC input are compared. Slightly low performance for hemoglobin and inclusion predictions is expected in comparison to stained data. This is because of absence of relevant details in DIC crops.

4.4.2 Comparison with Hand-crafted features

This section shows comparison between hand-crafted features and deep learning features using SVM classification. To make comparison fair, multi-output classification results from center cell input of BF crops are chosen. The logit space of size, shape and hemoglobin

distribution classifier form the feature vectors accordingly. SVM classifier is trained individually for each cell property on these feature vectors. The deep learning features are also visualized using 2-component t-SNE [5] embedding.

4.5 Domain Adaptation

4.5.1 Supervised Domain Adaptation Implementation

The network (as shown in Figure 3.9) is trained for prediction of hemoglobin distribution and inclusion individually. The *feature extractor* part is taken from pretrained EfficientNet-B7 [18]. Multi-Class classifier for hemoglobin distribution prediction and Multi-Label classifier for inclusion prediction are used. Dropout is not considered between *feature extractor* and *classifier*. Dataset is comprised of 11544 training crops and 2183 validation DIC crops (see Figure 4.11). Matching BF crop for every DIC crop is selected. Normalized *center cell input* in range [0,1] is considered to allow only center cell features into account. Same augmentations as described in Section 4.2.1.2 are applied to BF and DIC crops simultaneously. During training, each sampled batch with same input from two domains is passed through network individually. The total loss is calculated as given in Equation 3.4. But, for inclusion prediction, binary cross entropy loss is used instead of cross entropy loss. Adam optimizer with learning rate and weight decay of 10^{-4} and 10^{-3} accordingly is used. For the last 5 epochs, learning rate is decreased to 0.1 times its initial value. The number of epochs and batch size are fixed as 20 and 16 respectively. Model with good weighted F1 score on DIC validation data is selected for hemoglobin distribution prediction. For inclusion, model with good DIC validation accuracy is chosen.

The baseline model with only DIC input is trained separately for hemoglobin distribution and inclusion prediction with the same training procedure as mentioned above. Only difference is that dropout of 0.5 is considered between *feature extractor* and *classifier*. Now, comparison can be made from three different models for hemoglobin distribution and inclusion prediction. The models are the multi-output classification model of DIC crops (from Section 4.4.1), the baseline model and the model trained using feature alignment. This comparison tells us clearly whether DIC performance can be improved using feature alignment or not. Also, experiment is carried out to find effect of weighting factor α with values 0.2, 1 and 5.

For visualization of heatmap using Grad-CAM, pretrained Resnet50 [16] on ImageNet dataset [66] is used as feature extractor. The training procedure is exactly same as above. The last convolutional layer of network will have spatial size of 4x4 with our center cell input (120x120 spatial size). Thus, only heatmap of 4x4 grid can be obtained. Such low spatial heatmap mostly activates only one particular grid at center, as we have center cell input. So, the center cell crop is resized to 224x224 to get 7x7 spatial size in final convolutional layer. As a result, the produced heatmap will be of grid size 7x7 with possibly many

activations. Better visualization of important regions can be observed, if this heatmap is interpolated and visualized on input image.

4.5.2 Unsupervised Domain Adaptation Implementation

The network architecture is shown in Figure 3.10. Pretrained EfficientNet-B7 [18] is used as *feature extractor* network. *Label predictor* and *domain classifier* are with 1 fully connected and 2 fully connected layers respectively. No dropout is considered between *feature extractor* and *classifiers*. The network is trained for prediction of cell size, shape and hemoglobin distribution separately with 3, 11 and 5 class labels correspondingly. Training is performed using 11528 and 11544 train processed BF and DIC crops. Please note that there are no matching crops available between the domains. Standardized *full crop input* of BF and DIC data is used as source and target samples accordingly. Augmentations are same as described in Section 4.2.1.2, but random brightness contrast and blur are not considered. During training, cross entropy losses for label and domain prediction are calculated from each sampled batch of BF input, while only domain cross entropy loss is included from DIC input. As class-imbalance exists, WCE loss is used for label prediction with weights calculated from median frequency balancing [21] method. SGD with momentum [75] is used to optimize network parameters with momentum and weight decay of 0.9 and 5×10^{-4} accordingly. The initial learning rate (μ_0) for feature extractor and classifiers are 10^{-3} and 10^{-2} respectively. Low initial learning rate is used because feature extractor network is taken from pretrained model. Learning rate is decayed during training as shown in Equation 4.2, where p is the training progress linearly changing from 0 to 1, α and β are 10 and 0.75 correspondingly.

$$\mu_p = \frac{\mu_0}{(1 + \alpha \cdot p)^\beta} \tag{4.2}$$

Instead of fixing the adaptation factor λ, it is gradually changed from 0 to 1 as given in Equation 4.3, to suppress noisy signal from the domain classifier at the early stages of the training procedure. The γ value is fixed as 10.

$$\lambda_p = \frac{2}{1 + \exp(-\gamma \cdot p)} - 1 \tag{4.3}$$

Batch size and number of epochs are set as 16 and 30 respectively. After each epoch, validation accuracy using DIC input of 2183 crops is calculated. The model with good DIC validation accuracy is chosen as best performing model.

4.6 Evaluation Metrics

1. **Confusion Matrix:**
   Confusion Matrix is specific table layout that reports True Positives (TP), False Positives (FP), False Negatives (FN) and True Negatives (TN). It allows visualization of the performance of classification task. Example of confusion matrix for binary class classification is depicted in Figure 4.12. Each row of the matrix represents the instances of actual class while each column represents the instances of predicted class.

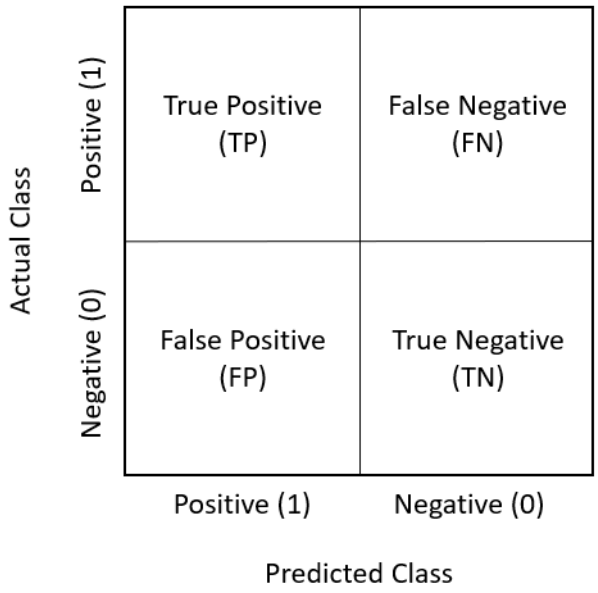


Figure 4.12: Confusion matrix

2. **Accuracy:**
   Accuracy is the ratio of correct predictions (both true positives and true negatives) among the total number of examined samples. For unbalanced dataset, accuracy will yield misleading results. Accuracy varies from 0 to 1, with 1 as best value.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{4.4}$$

3. **Precision:**
   Precision is the ratio of number of true positives among the total number of elements labelled as belonging to the positive class. It tells us the proportion of positive identifications which are actually correct. It is also referred to as positive predictive value.

It ranges between 0 and 1, with 0 as low and 1 as high value.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4.5}$$

4. **Recall:**
   Recall is the ratio of number of true positives among the total number of elements which actually belong to the positive class. It gives the proportion of actual positives that are identified correctly. It is also known as true positive rate or sensitivity. It varies from 0 to 1, with 0 being the lowest and 1 being the highest value.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{4.6}$$

5. **F1 Score:**
   F1 score is harmonic mean of precision and recall. It ranges between 0 and 1, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal. It is better metric to measure classification performance than accuracy for imbalanced data.

$$\text{F1 Score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{\text{TP}}{\text{TP} + \frac{1}{2}(\text{FP} + \text{FN})} \tag{4.7}$$

6. **Dice Coefficient:**
   Dice Coefficient is a statistic matrix used to measure the similarity of two samples. It is a performance metric for image segmentation problems. Given two sets, A and B, it is defined as,

$$\text{Dice Coefficient} = \frac{2\,|A \cap B|}{|A| + |B|} \tag{4.8}$$

   where $|A|$ and $|B|$ are cardinalities of the two sets. When applied on boolean data, we get same formulation as in Equation 4.7. It varies from 0 to 1, with 0 as lowest value.

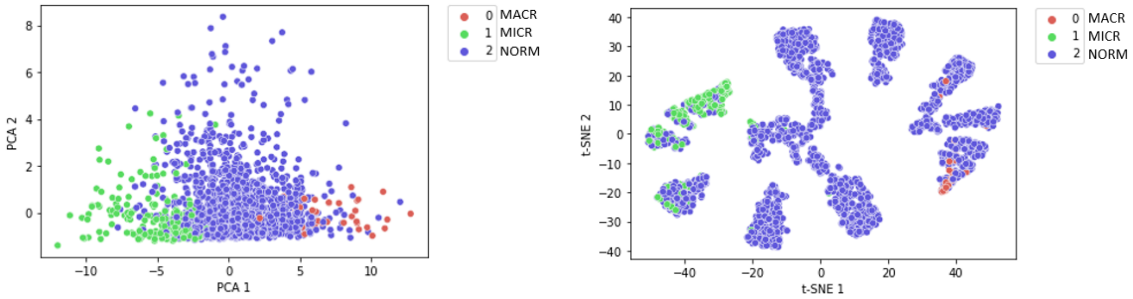
5 Results

In this chapter, the results obtained from various experiments are presented and anaylzed.

5.1 Clustering and SVM Classification

5.1.1 Size

Figure 5.1a and Figure 5.1b shows the visualization of size features on low dimensional space using PCA and t-SNE algorithms respectively. The ground truth labels for size (see Section 1.4.2) are marked on data points for better understanding. Clearly, the clusters are not evident on PCA plot, though 99% of variance is retained. So, clustering is performed on 2 component t-SNE, where clusters are visible.



(a) Size PCA

(b) Size t-SNE

Figure 5.1: Dimensionality reduction (Size)

Figure 5.2 show the results from HDBSCAN algorithm. Six clusters have been obtained by algorithm and they are grouped into 3 size clusters. The normal sized cells can exist in different shapes. So, many clusters exist within the *NORM* size cluster. The samples belonging to each cluster are displayed in Figure 5.3. The results from SVM classifier are shown in Figure 5.4 in the form of confusion matrix. On Y-axis, true labels are present and on X-axis, predicted labels are available. The matrix is normalized along the rows. The classifier performed very good in classification of cell sizes with accuracy of 87.43%.

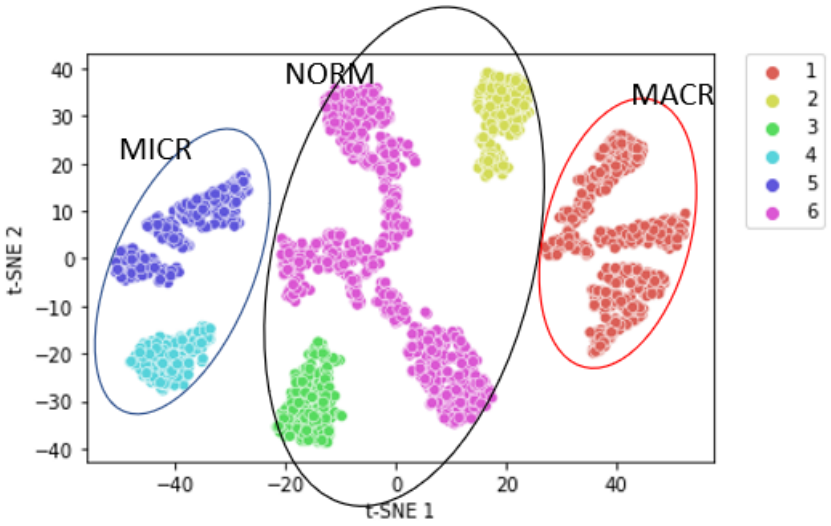


Figure 5.2: HDBSCAN clustering (Size)

Figure 5.3: Visualization of samples in size clusters

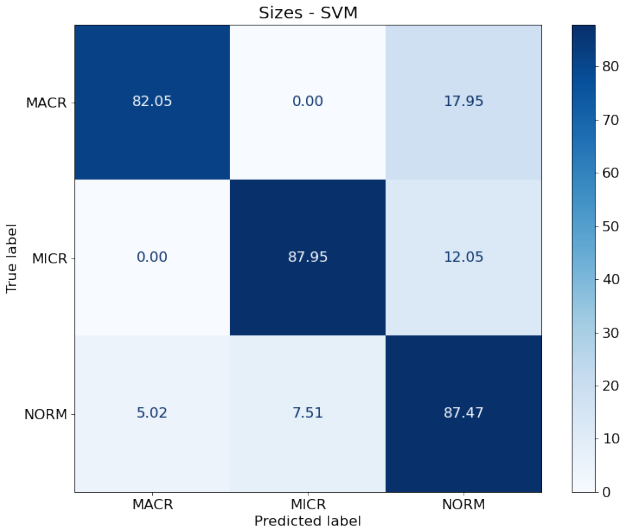
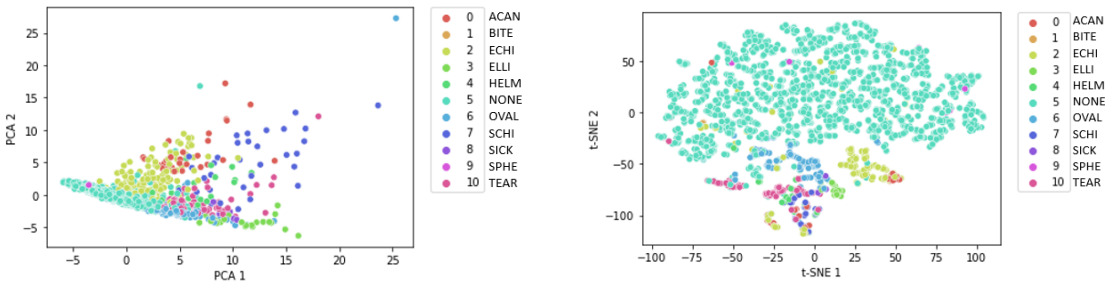


Figure 5.4: SVM classification (Size)

5.1.2 Shape

Figure 5.5a and Figure 5.5b shows the visualization of shape features on low dimensional space using PCA and t-SNE algorithms respectively. There are no clusters visible in PCA plot, even with 72% of captured variance. But, from t-SNE plot, some clusters can be seen.



(a) Shape PCA

(b) Shape t-SNE

Figure 5.5: Dimensionality reduction (Shape)

Figure 5.6 show the results from HDBSCAN clustering on 2 component t-SNE. Cluster 0 indicate outliers. Excluding outlier cluster, 7 clusters are obtained and they are named from inspection of samples within clusters. Acanthocytes, Schistocytes and Helmet cells can't be separated with defined feature vector. Bite cells are very few in number, to be exact 3 cells (see validation data in Figure 4.10) and are unable to visualize in clusters. *NONE*

and *SPHE* cells are mostly circular but vary in size. But, features selected are scale invariant and so, they are present in same cluster. *TEAR*, *ECHI* and *OVAL* cells have distinguishable shapes. *ELLI* and *SICK* look similar in shape, but *SICK* cells have pointed ends. So, they are placed in same cluster. Figure 5.7 display the samples present in different clusters.

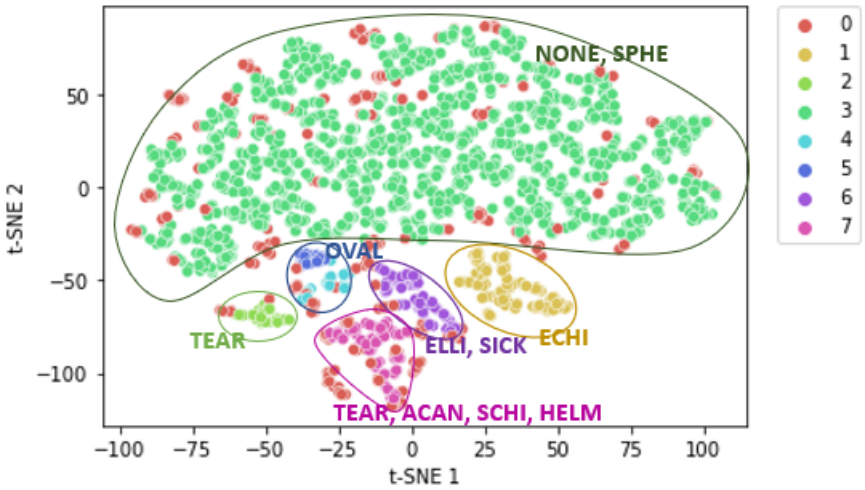


Figure 5.6: HDBSCAN clustering (Shape)

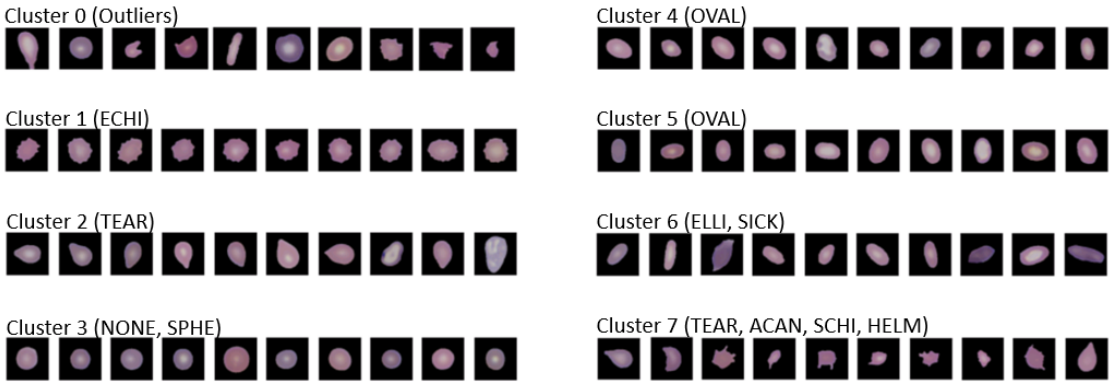


Figure 5.7: Visualization of samples in shape clusters

The result from SVM classification of shape features is shown in Figure 5.8. The accuracy of classifier is 86.21%. As dataset does not have enough *BITE* cells, the result is poor for them. There is confusion between *SPHE* and *NONE* cells because of scale invariant features selected.

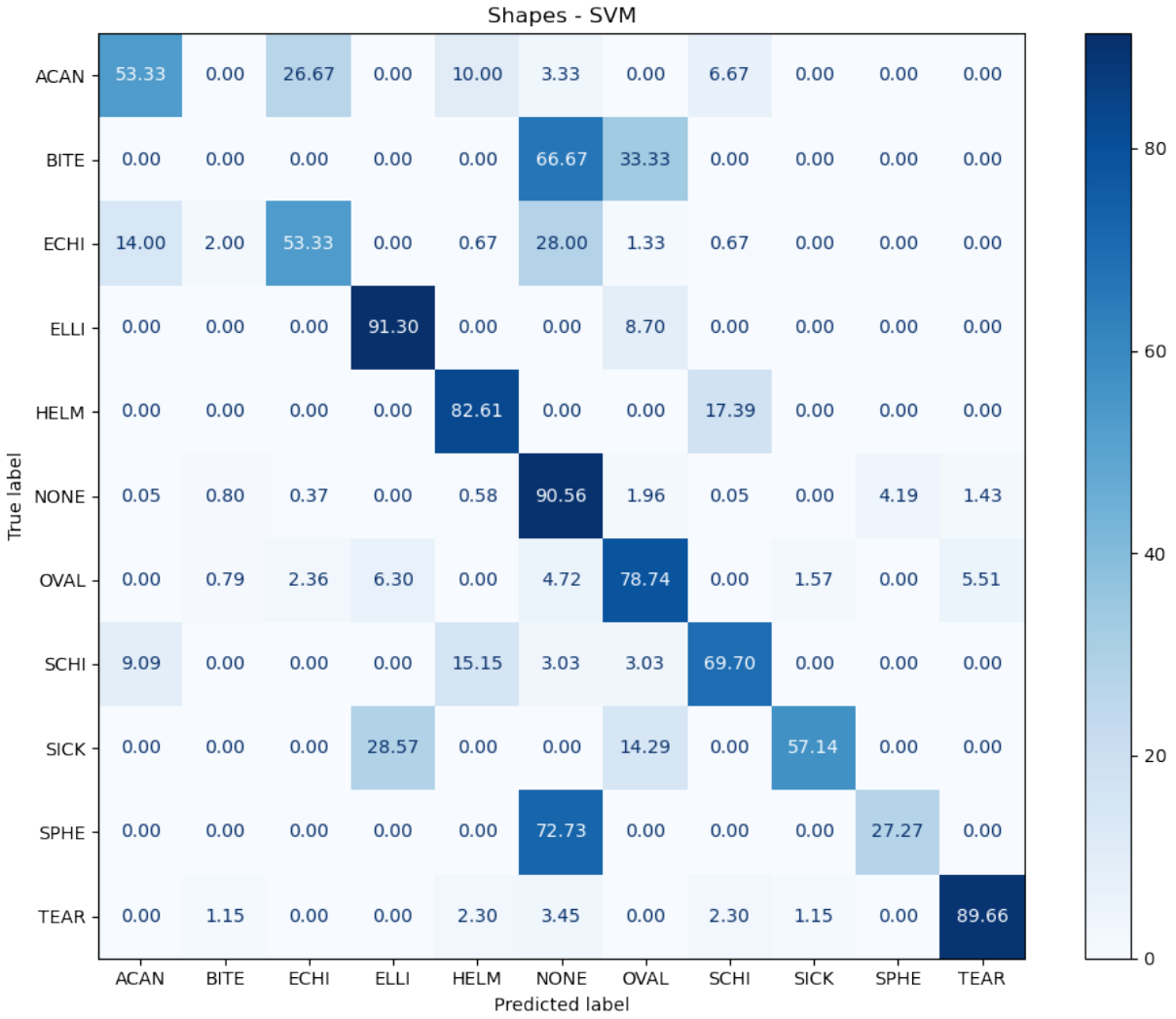
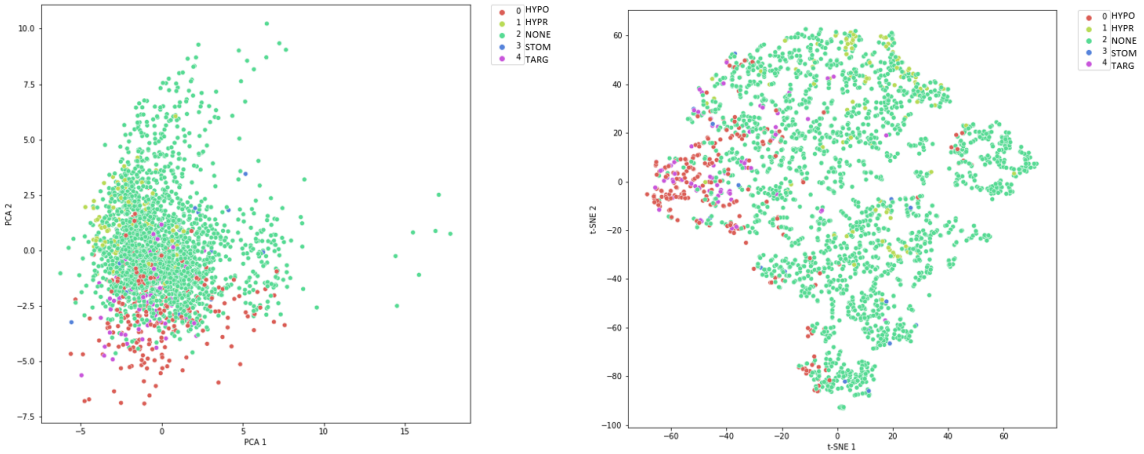


Figure 5.8: SVM classification (Shape)

5.1.3 Hemoglobin distribution

PCA and t-SNE plot for hemoglobin distribution features are shown in Figure 5.9a and Figure 5.9b accordingly. There are no clusters visible in both the plots. This indicates the features selected are not good enough to distinguish different cell types. Because of staining inconsistency, the hemoglobin colour will vary between crops and no good clusters can be seen. Figure 5.10 show SVM classification results for hemoglobin distribution. The accuracy of classifier is found to be 71.46%. The Stomatocytes are not classified properly. More false positives as *NONE* are predicted for rest of the classes.



(a) Hemoglobin distribution PCA (b) Hemoglobin distribution t-SNE

Figure 5.9: Dimensionality reduction (Hemoglobin distribution)

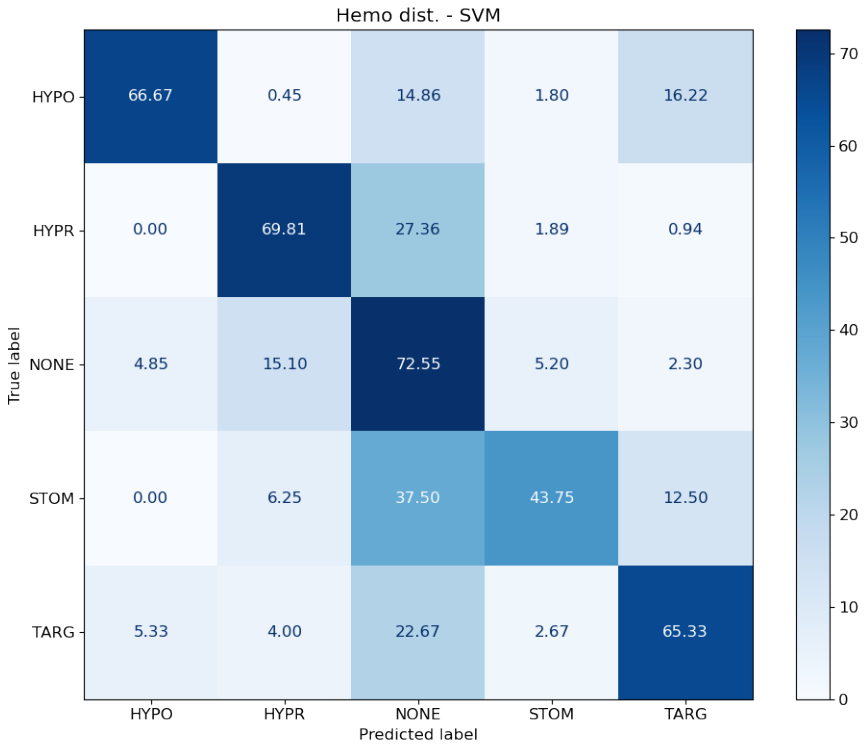


Figure 5.10: SVM classification (Hemoglobin distribution)

5.2 Multi-Output Classification

5.2.1 Stained crops

This Section shows the results obtained from multi-output classification model on BF validation data. Table 5.1 shows the accuracy of different cell properties achieved using different input types. We see that results from the full crop input is better than center cell input for all variations. This is because pretrained model was used on ImageNet dataset [66]. The data is similar to full crop as it has gradients all over the image. To verify it, model is trained without using pretrained weights for full crop and center cell input. This is shown in Table 5.2. Now, the center cell results are competent compared to full crop. However, the obtained accuracies are lower in comparison to that of pretrained model. So, we can go ahead with merged type input, whose results are promising as displayed in Table 5.1. Advantage of having this concatenated input is, we can have advantages of both full crop and center cell input and the trained model will be capable enough to pick relevant features. The performance of each classifier is analyzed in detail for merged type input.

Input type (BF)	Size	Shape	Hemo dist.	Inclusion
Full crop	**92.52**	**93.15**	84.40	97.65
Center cell	91.13	92.51	84.16	97.61
Merged	92.10	93.07	**85.49**	**97.94**

Table 5.1: Accuracy (in %) using different BF input types

Input type (BF)	Size	Shape	Hemo dist.	Inclusion
Full crop	**91.68**	87.09	81.43	**96.09**
Center cell	89.79	**89.57**	**83.19**	95.84

Table 5.2: Accuracy (in %) for model trained from scratch

Refer to Section 1.4.2, for description regarding the cell labels for all variations. Accuracy is not good metric for evaluation of data with imbalanced classes. Table 5.3 shows the F1 scores of various inclusion labels for different input types. Clearly, results from merged type look good in comparison. Remember that inclusion classifier is multi-label classifier and very few non-mutually exclusive labels are present as seen in Figure 4.10. The results can be better, if the model is trained on bigger dataset with more such labels.

The confusion matrices for size, shape and hemoglobin distribution are shown in Figures 5.11, 5.12 and 5.13 respectively. In Figure 5.11, we can see there is high confusion between Macrocytes and Normal sized cells. Also, small confusion exists between Microcytes and Normal cells. There is small degree of uncertainty in size during preparation of

Input type (BF)	BAST	HOJO	MALA	NONE	NRBC	PABO	RETI
Full crop	0.89	0.62	**1**	**0.99**	0.93	0.63	0.78
Center cell	**0.93**	0.59	0.98	**0.99**	**1**	0.4	0.79
Merged	0.87	**0.69**	0.98	**0.99**	**1**	**0.67**	**0.81**

Table 5.3: F1 scores for Inclusion labels using different BF input types

ground truth labels. So, the decision boundary learnt by model is not strict, which results in confusion.

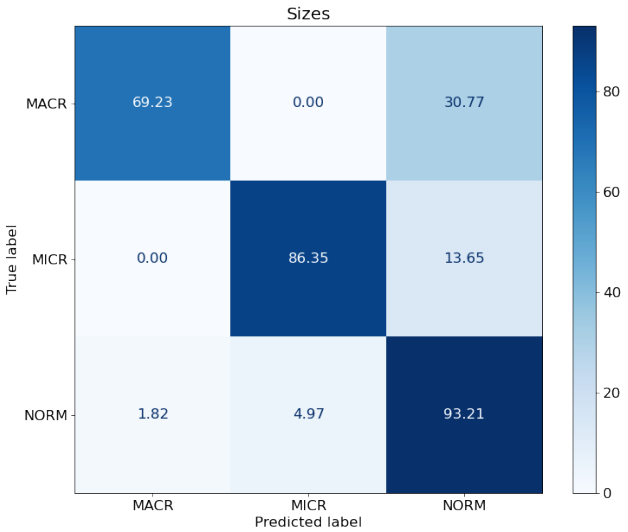


Figure 5.11: Size confusion matrix (BF)

Figure 5.12 shows confusion matrix for shape variation. Overall, the performance of shape classifier is good, except for Acanthocytes and Bite cells. This is because of few such crops in the training dataset. We can see confusion between *HYPO* and *NONE*, *HYPR* and *NONE* in the Figure 5.13 . This is expected because the ground truth labels might not be accurate for some crops in distinguishing these labels and the model's decision boundary is not so precise. Figure 5.14 illustrates the results from multi-output classification model. On Y-axis, we have Ground Truth (GT) labels and on X-axis, Predicted labels are shown. The order of displayed labels are size, shape, hemoglobin distribution and inclusion accordingly. The results are displayed for merged type input. For displaying the crop, center cell is chosen.

Figure 5.12: Shape confusion matrix (BF)

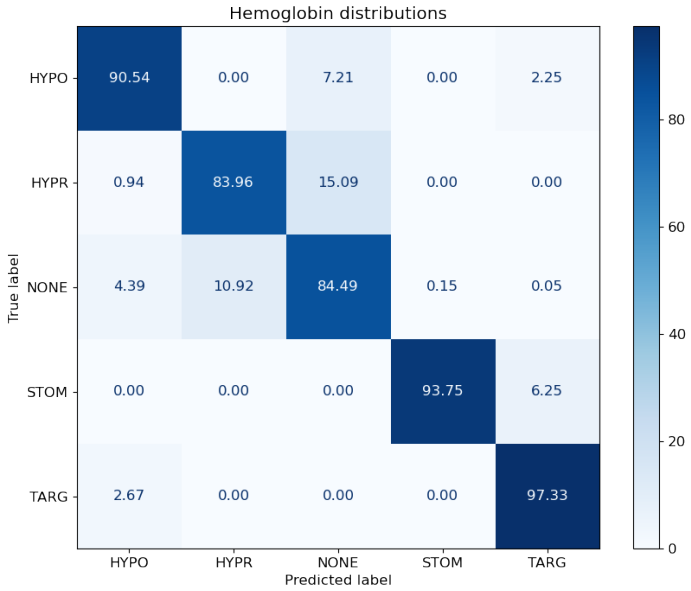


Figure 5.13: Hemoglobin distribution confusion matrix (BF)

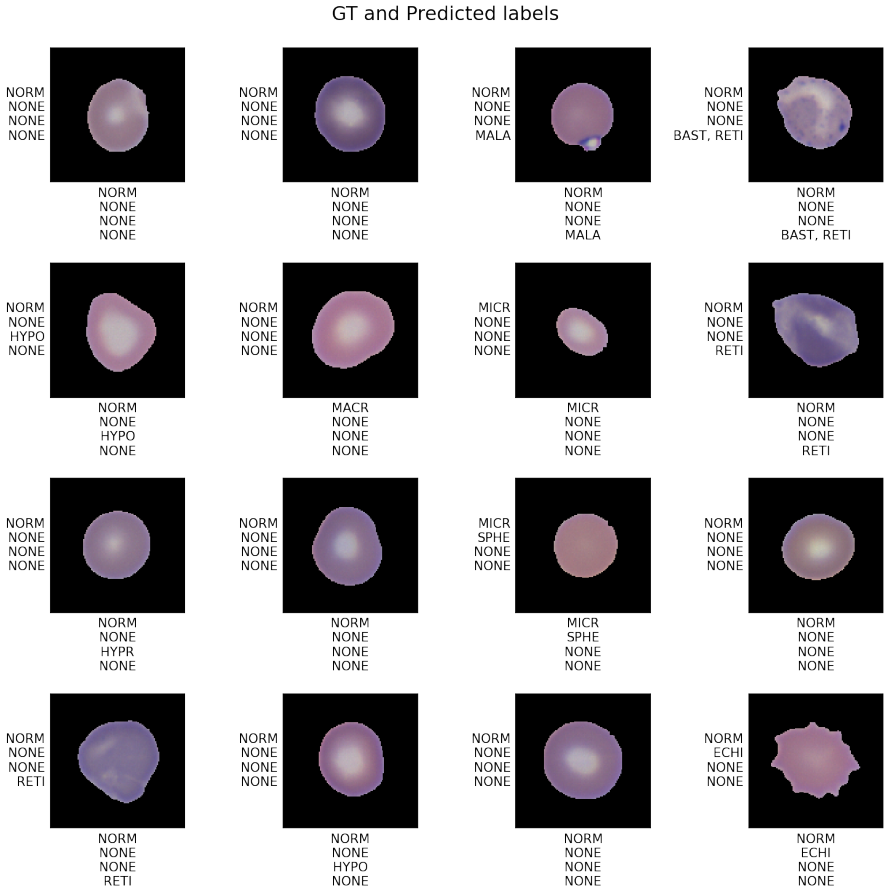


Figure 5.14: Classification results (BF)

5.2.2 Comparison results

5.2.2.1 Size

The SVM classification results from logits of size classifier is shown in Figure 5.15. The accuracy of classifier is 92.06%. For size, we already have results from SVM classification using hand-picked features (see Figure 5.4). These results are comparable to results with deep learning features. The accuracy using deep learning features is higher than with hand-crafted features. This is due to majority class, *NONE* which is classified correctly. But, the minority class, *MACR* has good classification performance using hand-crafted features. There are no precise size clusters formed with deep learning features. This can be visualized from Figure 5.16. The ground truth labels might vary for some border cases and so, the learnt model could not generalize well.

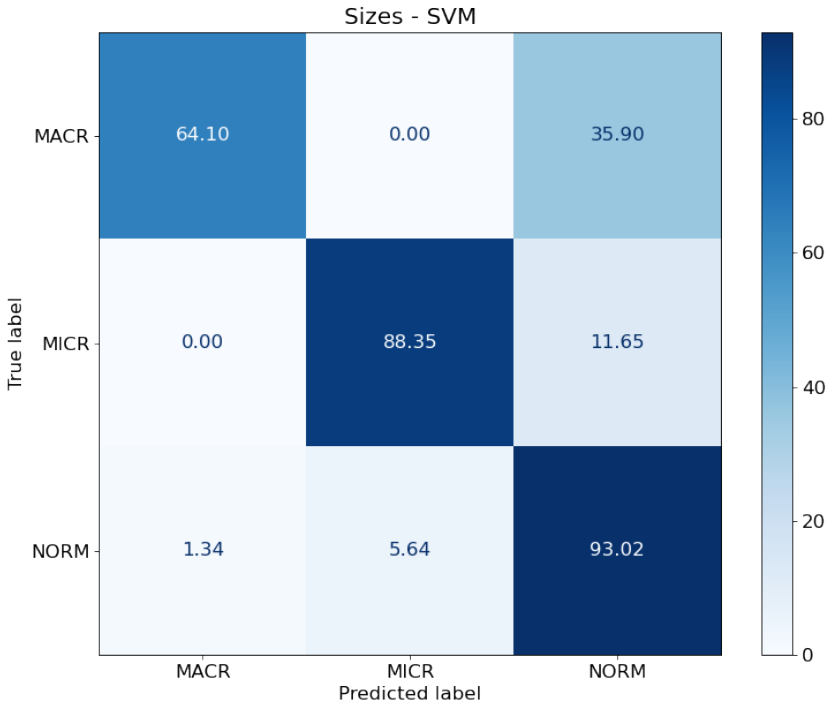


Figure 5.15: SVM classification (Size) on deep learning features

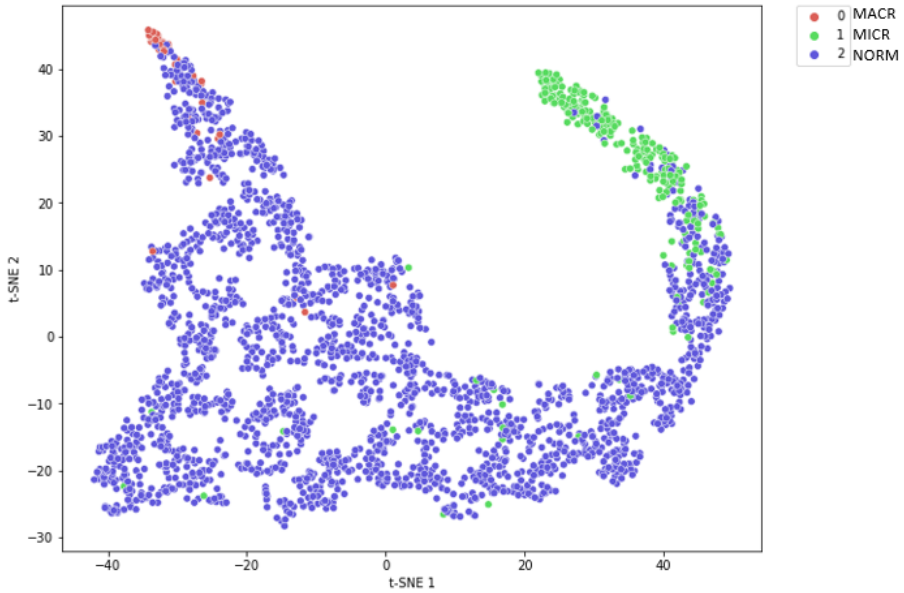


Figure 5.16: Visualization of Size classifier logit space using t-SNE

5.2.2.2 Shape

Figures 5.17 shows the SVM classification results from the output of shape classifier. The accuracy is found to be 93.78%. On comparing it with Figure 5.8, we can deduce that deep learning shape features are very good in comparison to quantitative extracted shape features. As *BITE* cells are few in number, the performance is bad. The visualization of learnt features is displayed in Figure 5.18. It can be seen that shape classes form separate clusters.

Figure 5.17: SVM classification (Shape) on deep learning features

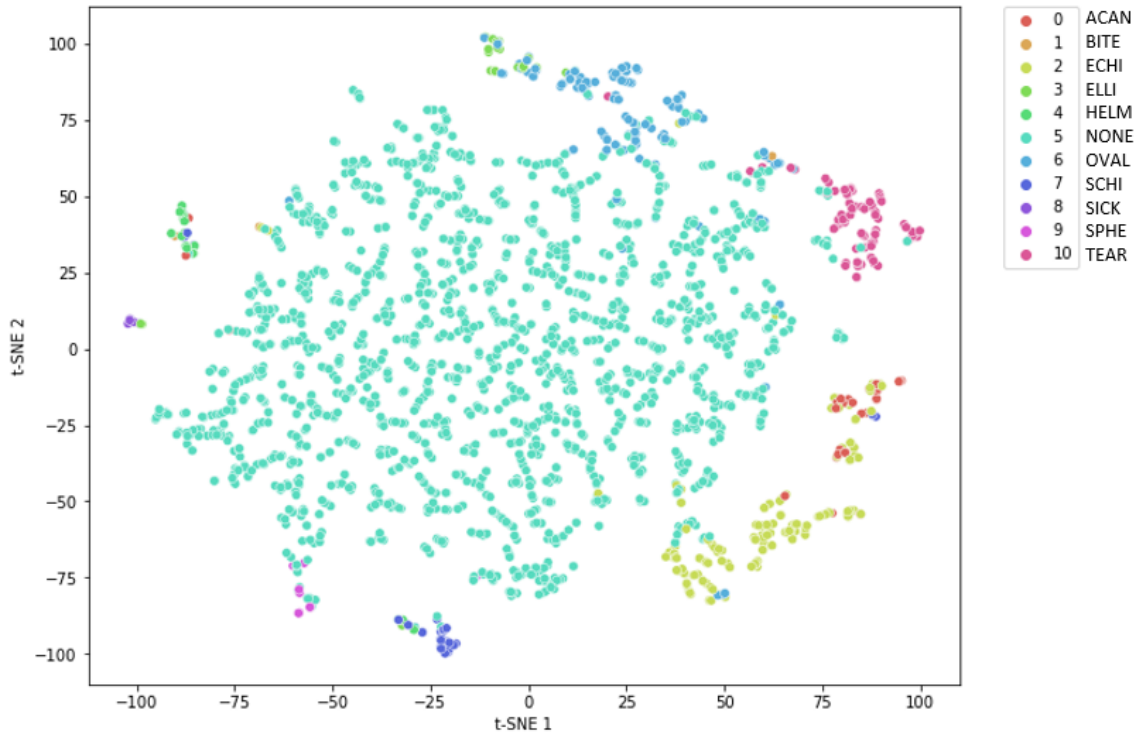


Figure 5.18: Visualization of Shape classifier logit space using t-SNE

5.2.2.3 Hemoglobin Distribution

SVM classification results from the output of hemoglobin distribution classifier can be seen from Figure 5.19. The accuracy of classifier is 86.17%. Big boost in performance is observed on contrary to results using extracted physical features from Figure 5.10. It can be clearly seen that deep learning model learnt abstract features yielding good classification performance. The extraction of similar biophysical features to describe hemoglobin distribution are hard. The visualization of deep learning features from output of hemoglobin distribution classifier is displayed in 5.20. We can see that *HYPO* and *HYPO* class labels are far apart in embedding space, which makes sense as they vary based on ratio of hemoglobin distribution to whitening inside cell. On the other hand, *TARG* cells are close to *HYPO* cells. This is because the cell composition is similar between cells, except that colouring is also present in center of *TARG* cell. Between *NONE* and *HYPO* class, *NONE* and *HYPR* class, there is no clear separation visible because of confusion between classes.

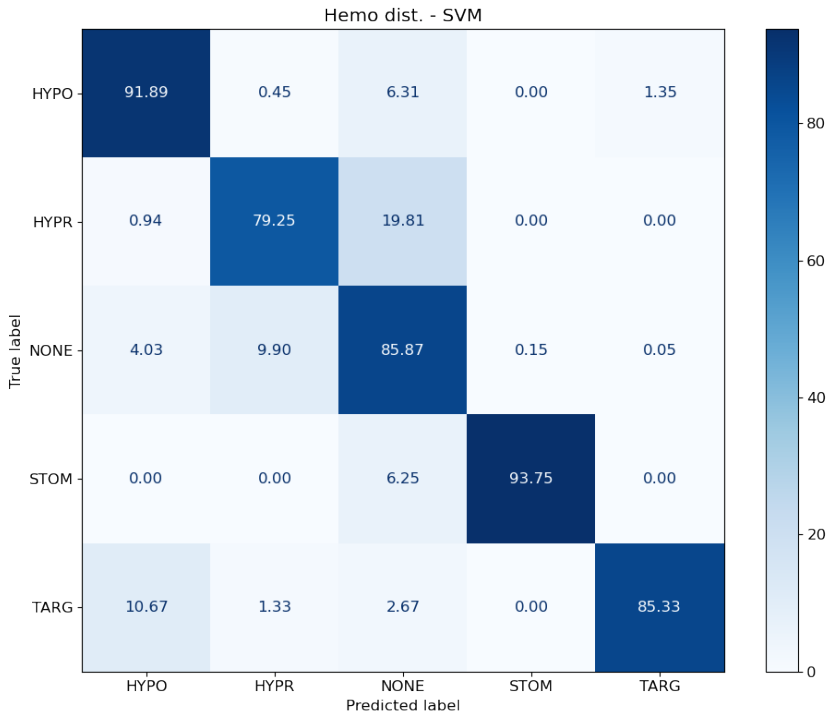


Figure 5.19: SVM classification (Hemoglobin distribution) on deep learning features

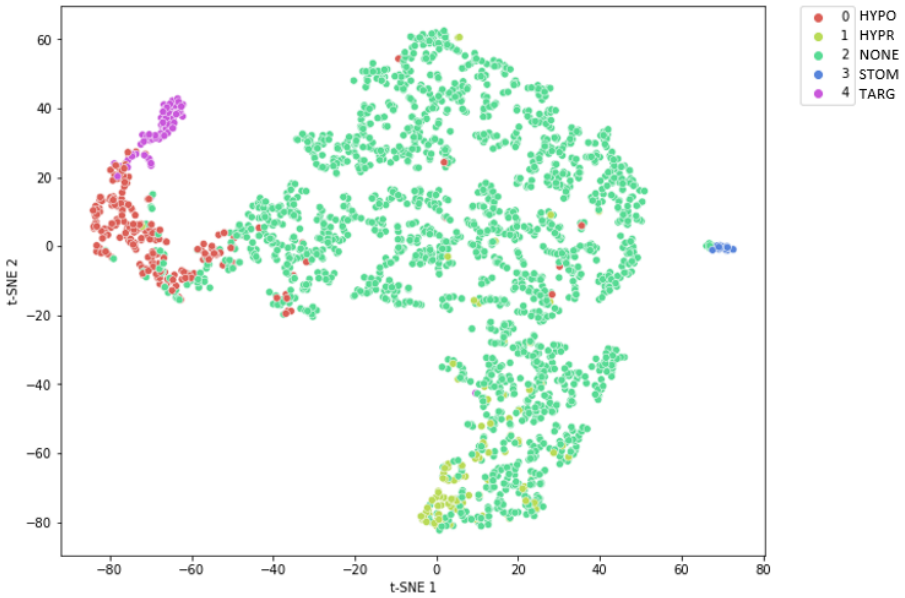


Figure 5.20: Visualization of Hemoglobin distribution classifier logit space using t-SNE

5.2.3 Unstained crops

This section deals with results from multi-output classification on DIC validation data. Accuracy of different cell variations using different input types are tabulated in Table 5.4. There is considerable drop in hemoglobin distribution and inclusion accuracy, in comparison to BF results (see Table 5.1). The necessary details are missing inside cell and model is not able to perform better in case of DIC input. On the other hand, cell size and shape accuracies are comparable to that of BF results suggesting learnt features for this classification are domain independent. Table 5.5 shows the F1 scores for inclusion labels. Clearly, results with BF (refer Table 5.3) outperforms DIC F1 scores. Due to absence of colour features in DIC, the results are bad in comparison to BF crops. Among different input types, merged type results look better.

Input type (DIC)	Size	Shape	Hemo dist.	Inclusion
Full crop	**92.90**	93.12	80.53	92.81
Center cell	91.76	**93.67**	80.66	92.67
Merged	92.31	92.57	**83.93**	**93.64**

Table 5.4: Accuracy (in %) using different DIC input types

Input type (DIC)	BAST	HOJO	MALA	NONE	NRBC	PABO	RETI
Full crop	0.69	0.12	0.6	0.97	0.91	0	0.54
Center cell	**0.7**	0.09	0.6	**0.98**	**1**	0.13	0.56
Merged	0.63	**0.27**	**0.66**	0.97	**1**	**0.18**	**0.58**

Table 5.5: F1 scores for Inclusion labels using different DIC input types

Confusion matrices for size, shape and hemoglobin distribution using merged DIC input are displayed in Figures 5.21, 5.22 and 5.23 accordingly. The confusion among size classes is slightly less in comparison to stained results in Figure 5.11. Apart from Acanthocytes and Bite cells, the shape confusion matrix in Figure 5.22 looks good. In Figure 5.23, performance drop is observed for Stomatocyte and Target cell in contrast to that observed with BF (refer Figure 5.13). It becomes difficult for model to learn hemoglobin distribution, that can distinguish classes in absence of colour. Figure 5.24 shows prediction on some DIC samples shown along X-axis and true labels on Y-axis. These results are obtained with merged type input, but center cell image is shown for visualization. The order of labels displayed are size, shape, hemoglobin distribution and inclusion respectively.

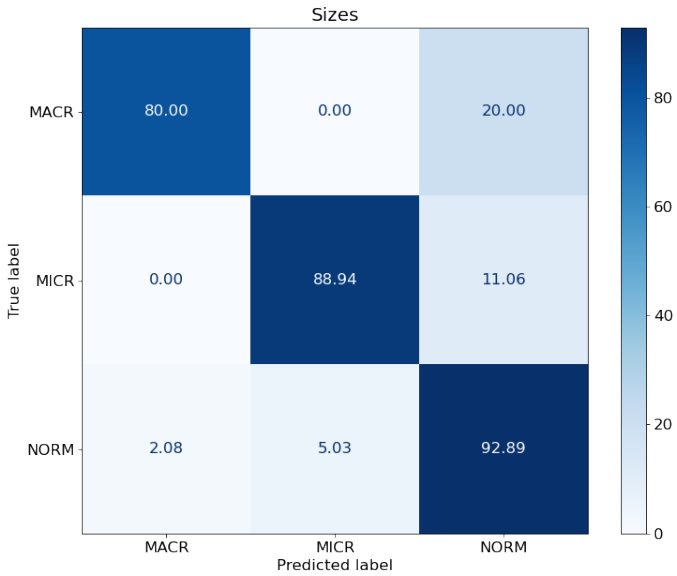


Figure 5.21: Size confusion matrix (DIC)

Figure 5.22: Shape confusion matrix (DIC)

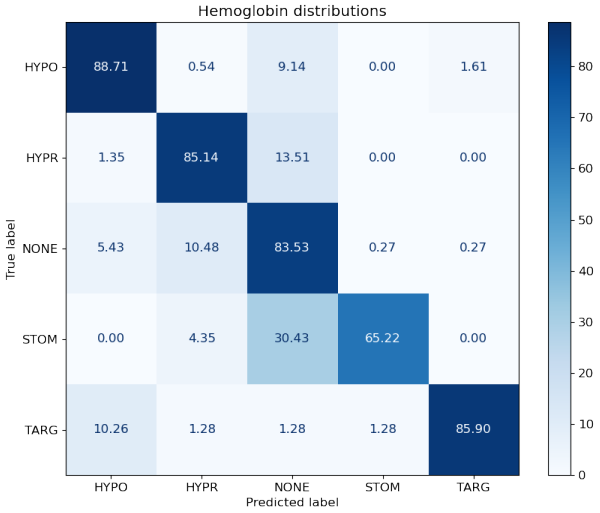


Figure 5.23: Hemoglobin distribution confusion matrix (DIC)

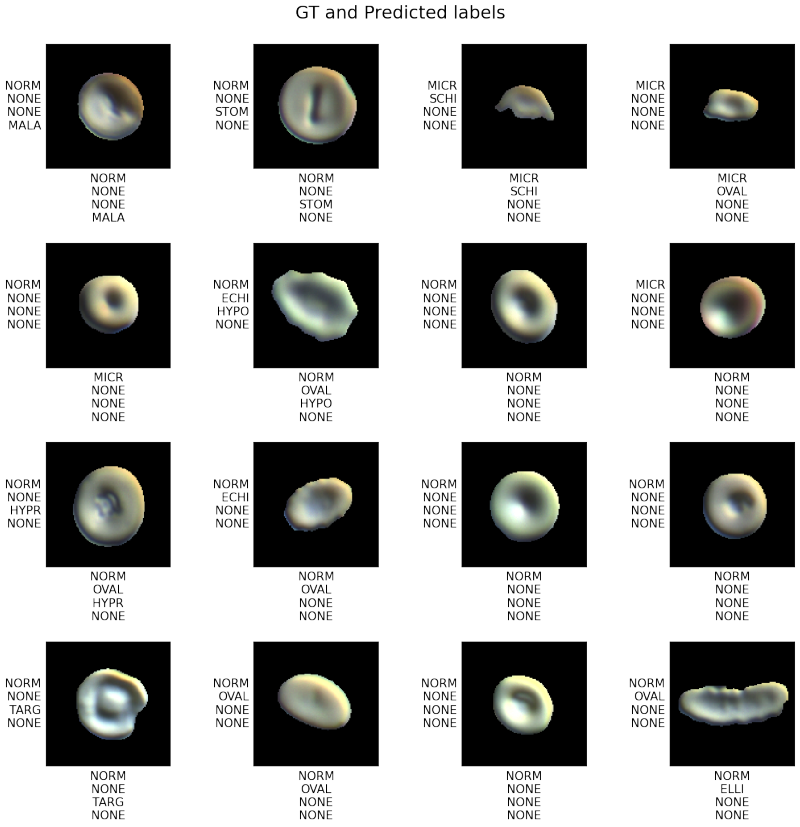


Figure 5.24: Classification results (DIC)

5.3 Domain Adaptation

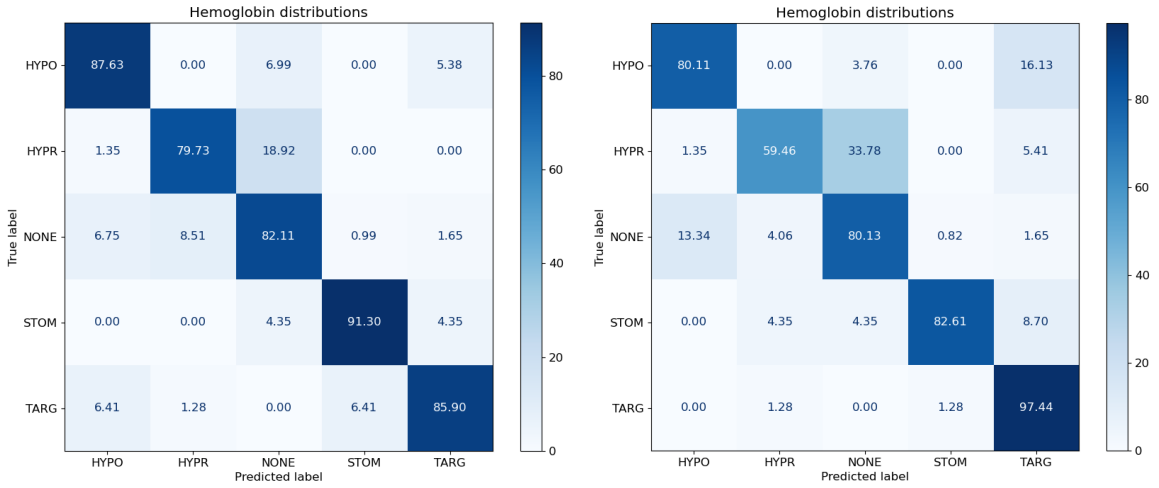
5.3.1 Supervised

Table 5.6 shows DIC accuracy of hemoglobin distribution and inclusion for three different models, multi-output, baseline and feature alignment model with α as 1. On comparison of the three models, we can see that accuracy of baseline model is best for hemoglobin distribution classification. The feature alignment didn't seem to improve DIC accuracy. With BF crops, the feature alignment model produced accuracy of 81.09%.

Model	Hemo dist.	Inclusion
Multi-Output	80.66	92.67
Baseline	**82.68**	91.15
Feature Alignment	80.04	**94.84**

Table 5.6: DIC Accuracy (in %) for different models

Figure 5.25 displays comparison of hemoglobin distribution confusion matrices between baseline and feature alignment model. Clearly, baseline model dominates feature alignment model for all hemoglobin distribution labels except *TARG* cells.



(a) Baseline model (DIC)

(b) Feature alignment model (DIC)

Figure 5.25: Comparison of hemoglobin distribution confusion matrices

On the other hand, feature alignment model seems to dominate for inclusion prediction in comparison to other models as seen from Table 5.6. The model produced BF inclusion accuracy of 96.53%. However, accuracy is not a good metric for comparison especially with imbalanced dataset.

Table 5.7 shows F1 scores for inclusion labels obtained from different models. The results from feature alignment model with BF crops is also tabulated. There is no big improvement in performance for feature alignment model with DIC crops. In general, all models with DIC input show significant performance drop in comparison to feature alignment model with BF input. As features are not clearly visible in DIC crops, the performance cannot be improved using feature alignment.

Input	Model	BAST	HOJO	MALA	NONE	NRBC	PABO	RETI
DIC	Multi-Output	0.7	**0.09**	**0.6**	**0.98**	**1**	0.13	0.56
DIC	Baseline	0.65	0.08	0.58	**0.98**	**1**	0.13	**0.59**
DIC	Feature alignment	**0.72**	0	0.42	**0.98**	**1**	**0.4**	0.38
BF	Feature alignment	0.86	0.58	0.86	0.99	1	0.62	0.65

Table 5.7: F1 scores for Inclusion labels using different models

The above mentioned results with feature alignment used weighting factor (α) as 1. Now, let us check whether variation in α can produce better results. Table 5.8 illustrates the results of feature alignment model with different α values using DIC input. The performance for hemoglobin distribution classification is good with α as 1 and inclusion prediction is slightly better using α as 0.2. Table 5.9 shows F1 scores of inclusion labels with various α values. All displayed results seem to be very low in comparison to feature alignment model with BF input (see Table 5.7).

Weighting factor (α)	Hemo dist.	Inclusion
0.2	76.36	**95.03**
1	**80.04**	94.84
5	77.93	94.57

Table 5.8: DIC Accuracy (in %) for feature alignment model with different α values

Weighting factor (α)	BAST	HOJO	MALA	NONE	NRBC	PABO	RETI
0.2	**0.73**	0	**0.75**	**0.98**	**1**	0.3	0.42
1	0.72	0	0.42	**0.98**	**1**	**0.4**	0.38
5	0.64	**0.09**	0.58	**0.98**	0.91	0.14	**0.51**

Table 5.9: DIC F1 scores for Inclusion labels using feature alignment model with different α values

Visualization of heatmap gives idea on where the network focusses to assign a prediction label. This helps us to verify whether relevant features for correct prediction are present in

DIC crops or not. Figure 5.26 shows the class activation maps obtained using Grad-CAM method for inclusion predictions. The activation maps are shown for both BF and DIC crops obtained from the feature alignment model. First image shows *BAST* cell, which is predicted correctly in case of BF input. For DIC input, we have two predictions, *BAST* and *RETI*. The second image has *HOJO* on cell, which is predicted only with BF input. Third and fourth images show *MALA* and *NRBC* cells. They are predicted correctly for both input types. The activation maps for these images are almost present in same correct regions producing good predictions. For some cases, the network doesn't get activated around proper cell patterns and sometimes, activations are present outside the cell. This means the network hasn't properly learnt the underlying patterns, may be due to lack of training data or absence of relevant features. Also, the activations might be observed exterior to cell as the feature alignment model is trained, which align the global source and target distributions losing fine-grained information for each category. This concludes that DIC crops have some missing details in comparison to BF crops.

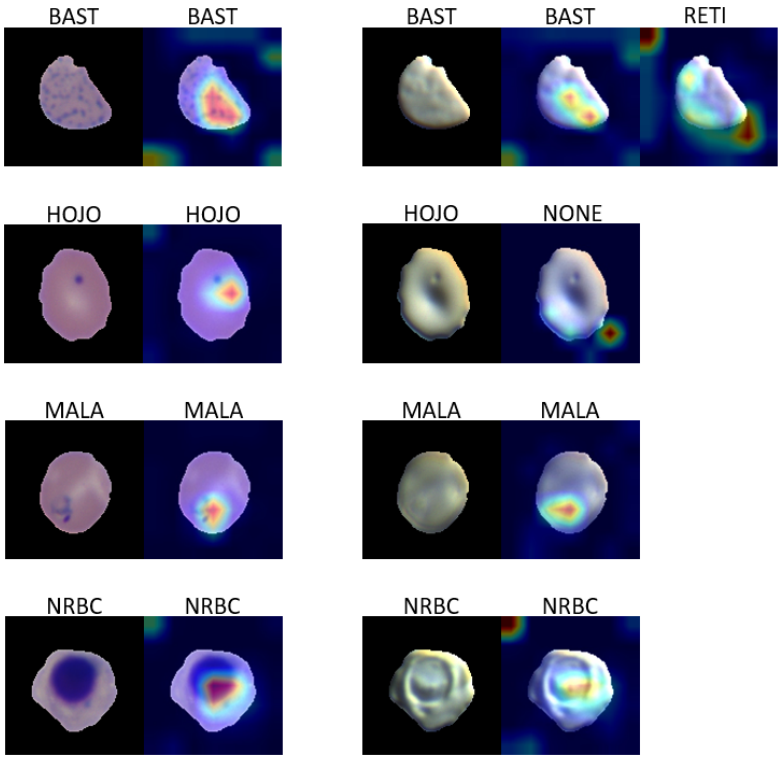


Figure 5.26: Visualization of heatmaps for Inclusion prediction. BF (on left) and DIC crops (on right) along with ground truth and predicted label are displayed.

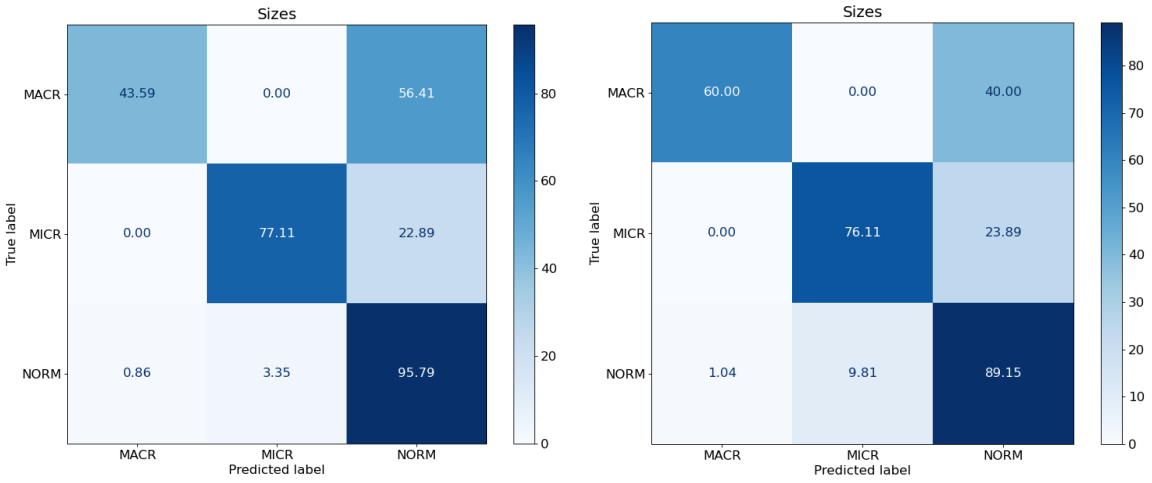
5.3.2 Unsupervised

Table 5.10 shows the obtained accuracy of two domains using different classifiers. Remember that the network is trained with labelled BF and unlabelled DIC data. Our aim is to get good accuracy using DIC crops. The network performs good when tested with DIC crops. On comparison of results from Table 5.10, we see that size and hemoglobin distribution accuracy are good for source data, while shape accuracy is good with target data. Clearly from validation data of Figures 4.10 and 4.11, there is class imbalance present among size, shape and hemoglobin distribution labels. As accuracy is not good metric for comparison with imbalanced data, we will analyze the confusion matrices for different cell properties.

Domain	Size	Shape	Hemo dist.
Source (BF)	92.98	88.55	87.48
Target (DIC)	87.4	91.14	83.56

Table 5.10: BF and DIC Accuracy (in %) for different classifiers

Figure 5.27 shows comparison of confusion matrices for size variation between BF and DIC crops. *NORM* sized crops dominate in number from the dataset. Higher accuracy for BF is observed because more *NORM* crops are classified correctly as seen from Figure 5.27a. For both domains, the network learnt well to distinguish between *MACR* and *MICR* labels. On the other hand, there is lot of confusion between *MACR* and *NORM* cells and *MICR* and *NORM* cells. Note that the ground truth size labels cannot be clearly defined. As a result, this confusion is inevitable. On the whole, the performance looks good for DIC crops.

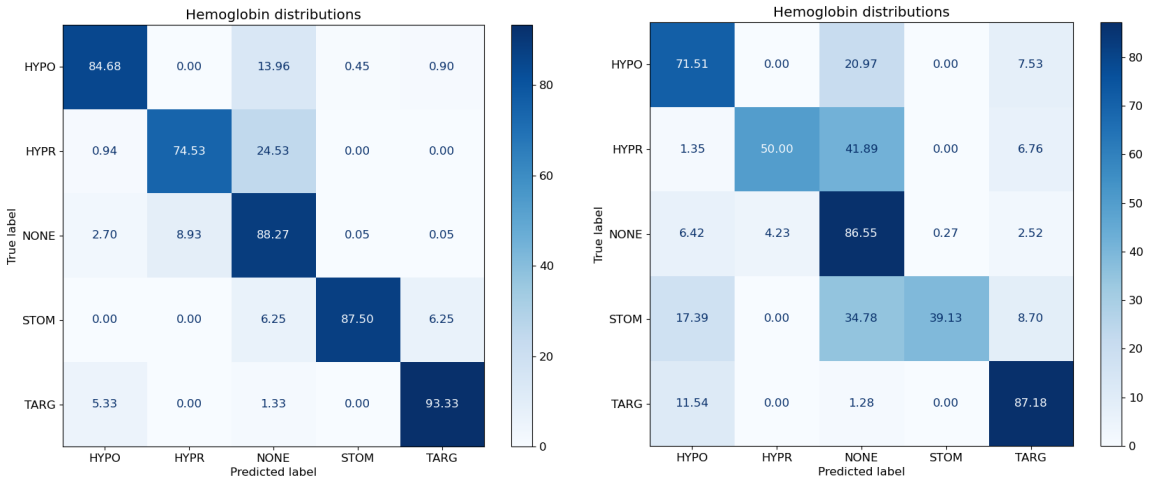


(a) BF

(b) DIC

Figure 5.27: Size confusion matrices - DANN

The hemoglobin distribution confusion matrices for BF and DIC crops are shown in Figure 5.28. BF results are very good in comparison except for confusion between *HYPO* and *NONE* label and *HYPR* and *NONE* label. On the other hand, huge performance drop is observed for *HYPR* and *STOM* DIC crops. Domain adaptation becomes challenging for hemoglobin distribution classification as internal cell structure is involved, unlike size and shape classification. The features are not clearly visible in DIC crops making it difficult for emergence of good domain-invariant features. Also, the performance is low due to lack of training data.

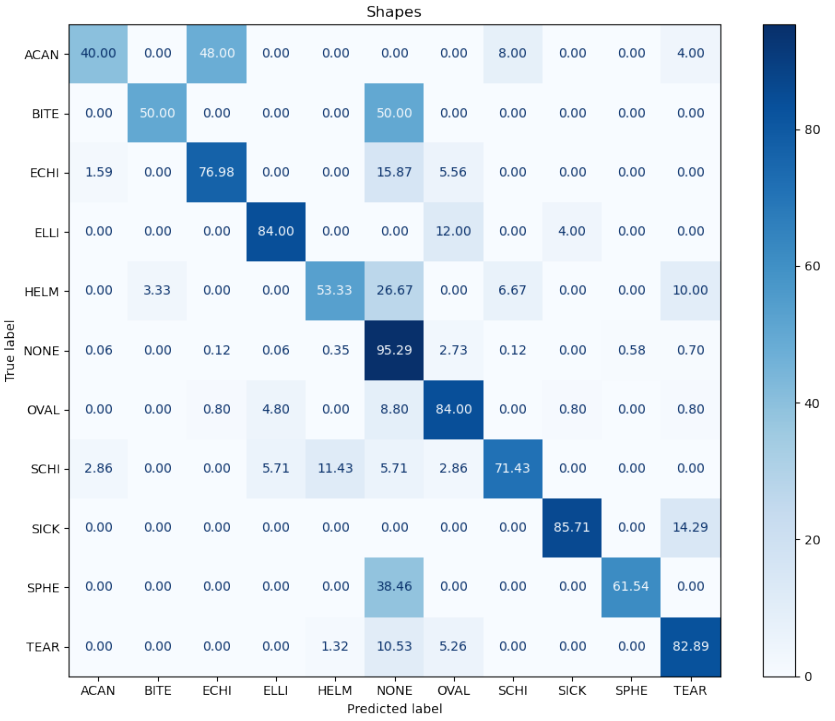


(a) BF (b) DIC

Figure 5.28: Hemoglobin distribution confusion matrices - DANN

Figure 5.29 displays shape confusion matrices for BF and DIC data. High accuracy is observed in Table 5.10 with DIC crops, because more crops of majority class (*NONE*) are classified correctly. The classification is very good using BF data, as the network is trained with these labels. The results with DIC data are good and comparable to BF results. For both domains, low performance is observed with *ACAN* and *BITE* cells. As *BITE* cells are low in number, the performance is bad. The learnt domain-invariant features can't properly classify *ACAN* cells.

(a) BF



(b) DIC

Figure 5.29: Shape confusion matrices - DANN

6 Conclusions and Future Work

6.1 Conclusions

In this thesis, different topics related to RBC classification are addressed. Firstly, the derived quantitative features using feature engineering look promising. These features convey biophysical meaning that are easily interpretable by hematologists unlike deep learning features. Secondly, multi-output classification is performed for BF and DIC crops. Good results are obtained with merged type input for both crops. DIC performance is lower for hemoglobin distribution and inclusion on comparison with BF results. This is due to absence of more details in DIC data. The performance improvement can be observed for good DIC microscopic images with more features. This can possibly remove time-consuming staining procedures. In general, medical image analysis is challenging due to lack of labelled data unlike natural image analysis with large-scale labelled datasets such as ImageNet [66]. Increase in annotated data ameliorates the performance of deep learning algorithms. It gets even better with more data from rare classes. Labelling medical images is normally expensive, difficult and time-consuming process. Also, the labelled data might not be accurate because of label inconsistency between hematologists.

Thirdly, supervised and unsupervised domain adaptation are performed to learn from multiple domains with different goals. Supervised domain adaptation to improve DIC performance of hemoglobin distribution and inclusion does not seem to work because of absence of good quality DIC features. Unsupervised domain adaptation provides way to get DIC labels without training using DIC labelled data. Hematologists are not trained to work with unstained images for label preparation. So, DIC labels can only be obtained from matching the BF and DIC crops. Without exploiting this matching information, DIC labels are obtained using unsupervised domain adaptation. However, predicted hemoglobin distribution labels from DIC crops are only satisfactory due to insufficient training data. On the whole, this work explores the potential of automated algorithms for cell classification, which can perform better with more and good quality data.

6.2 Future Work

More robust physical features can be extracted using feature engineering. Such features promote unsupervised learning for cell classification, which requires no labelled data. Fusion of hand-crafted and deep learning features might help in improvement of classification performance.

Patient-wise classifier can be trained that allows the model to predict based on information available from different cells of one patient during the classification process. For multi-label inclusion classification, Graph Neural Networks (GNNs) [76] can be used to provide the network with the information on which label combinations are more probable. This will greatly boost the performance of inclusion prediction.

In this thesis, unsupervised domain adaptation is performed only for cell size, shape and hemoglobin distribution prediction. Unsupervised adversarial domain adaptation approach can be applied for multi-label inclusion classification [68]. Deep domain adaptation methods focus on learning a global domain shift without exploiting the complex multi-mode structures present in the same category of different domains. Instead, subdomain adaptation methods can be employed that focuses on accurately aligning the distributions of the relevant subdomains within the same category in the source and target domains [77, 78]. These methods have the capability to capture the fine-grained information for each category. This will largely increase the classification performance on target data.

List of Figures

List of Tables

Bibliography

[1] B. Bain, *Blood Cells: A Practical Guide*. Wiley, 2015.

[2] A. Tomczak, S. Ilic, G. Marquardt, T. Engel, F. Forster, N. Navab, and S. Albarqouni, "Multi-task multi-domain learning for digital staining and classification of leukocytes," *IEEE Transactions on Medical Imaging*, pp. 1–1, 2020.

[3] D. B. Murphy and M. W. Davidson, *Fundamentals of Light Microscopy and Electronic Imaging*. Wiley, 2013.

[4] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[5] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[6] S. Lloyd, "Least squares quantization in pcm," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[7] R. J. Campello, D. Moulavi, A. Zimek, and J. Sander, "Hierarchical density estimates for data clustering, visualization, and outlier detection," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 10, no. 1, pp. 1–51, 2015.

[8] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[9] K. Fukushima, "Neocognitron," *Scholarpedia*, vol. 2, no. 1, p. 1717, 2007. revision #91558.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

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

[12] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *arXiv preprint arXiv:1506.01497*, 2015.

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

[14] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[15] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. J. Cardoso, "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations," in *Deep learning in medical image analysis and multimodal learning for clinical decision support*, pp. 240–248, Springer, 2017.

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

[17] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks. arxiv 2016," *arXiv preprint arXiv:1608.06993*, vol. 1608, 2016.

[18] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*, pp. 6105–6114, PMLR, 2019.

[19] D. Xu, Y. Shi, I. W. Tsang, Y.-S. Ong, C. Gong, and X. Shen, "Survey on multi-output learning," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 7, pp. 2409–2429, 2019.

[20] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[21] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proceedings of the IEEE international conference on computer vision*, pp. 2650–2658, 2015.

[22] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[23] H. Guan and M. Liu, "Domain adaptation for medical image analysis: A survey," *arXiv preprint arXiv:2102.09508*, 2021.

[24] G. Wilson and D. J. Cook, "A survey of unsupervised deep domain adaptation," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 11, no. 5, pp. 1–46, 2020.

[25] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, pp. e49–e57, 2006.

[26] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," *arXiv preprint arXiv:1412.3474*, 2014.

[27] G. D. Evangelidis and E. Z. Psarakis, "Parametric image alignment using enhanced correlation coefficient maximization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 10, pp. 1858–1865, 2008.

[28] Y. Dong, Z. Jiang, H. Shen, W. D. Pan, L. A. Williams, V. V. Reddy, W. H. Benjamin, and A. W. Bryan, "Evaluations of deep convolutional neural networks for automatic identification of malaria infected cells," in *2017 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, pp. 101–104, IEEE, 2017.

[29] M. Delgado-Ortet, A. Molina, S. Alférez, J. Rodellar, and A. Merino, "A deep learning approach for segmentation of red blood cell images and malaria detection," *Entropy*, vol. 22, no. 6, p. 657, 2020.

[30] M. Xu, D. P. Papageorgiou, S. Z. Abidi, M. Dao, H. Zhao, and G. E. Karniadakis, "A deep convolutional neural network for classification of red blood cells in sickle cell anemia," *PLoS computational biology*, vol. 13, no. 10, p. e1005746, 2017.

[31] J. Bacus and J. Weens, "An automated method of differential red blood cell classification with application to the diagnosis of anemia.," *Journal of Histochemistry & Cytochemistry*, vol. 25, no. 7, pp. 614–632, 1977.

[32] R. Tomari, W. N. W. Zakaria, M. M. A. Jamil, F. M. Nor, and N. F. N. Fuad, "Computer aided system for red blood cell classification in blood smear image," *Procedia Computer Science*, vol. 42, pp. 206–213, 2014. Medical and Rehabilitation Robotics and Instrumentation (MRRI2013).

[33] L. L. Wheeless, R. D. Robinson, O. P. Lapets, C. Cox, A. Rubio, M. Weintraub, and L. J. Benjamin, "Classification of red blood cells as normal, sickle, or other abnormal, using a single image analysis feature," *Cytometry: The Journal of the International Society for Analytical Cytology*, vol. 17, no. 2, pp. 159–166, 1994.

[34] C. L. Chen, A. Mahjoubfar, L.-C. Tai, I. K. Blaby, A. Huang, K. R. Niazi, and B. Jalali, "Deep learning in label-free cell classification," *Scientific reports*, vol. 6, no. 1, pp. 1–16, 2016.

[35] B. T. Grys, D. S. Lo, N. Sahin, O. Z. Kraus, Q. Morris, C. Boone, and B. J. Andrews, "Machine learning and computer vision approaches for phenotypic profiling," *Journal of Cell Biology*, vol. 216, no. 1, pp. 65–71, 2017.

[36] D. Bhaskar, D. Lee, H. Knútsdóttir, C. Tan, M. Zhang, P. Dean, C. Roskelley, and L. Edelstein-Keshet, "A methodology for morphological feature extraction and unsupervised cell classification," *bioRxiv*, 2019.

[37] D. Mundhra, B. Cheluvaraju, J. Rampure, and T. R. Dastidar, "Analyzing microscopic images of peripheral blood smear using deep learning," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 178–185, Springer, 2017.

[38] J. Angulo and G. Flandrin, "Automated detection of working area of peripheral blood smears using mathematical morphology," *Analytical cellular pathology*, vol. 25, no. 1, pp. 37–49, 2003.

[39] A. Loddo, C. Di Ruberto, and M. Kocher, "Recent advances of malaria parasites detection systems based on mathematical morphology," *Sensors*, vol. 18, no. 2, p. 513, 2018.

[40] H. A. Aliyu, M. A. A. Razak, and R. Sudirman, "Segmentation and detection of sickle cell red blood image," in *AIP Conference Proceedings*, vol. 2173, p. 020004, AIP Publishing LLC, 2019.

[41] S. Rajaraman, S. K. Antani, M. Poostchi, K. Silamut, M. A. Hossain, R. J. Maude, S. Jaeger, and G. R. Thoma, "Pre-trained convolutional neural networks as feature extractors toward improved malaria parasite detection in thin blood smear images," *PeerJ*, vol. 6, p. e4568, 2018.

[42] F. Al-Hafiz, S. Al-Megren, and H. Kurdi, "Red blood cell segmentation by thresholding and canny detector," *Procedia Computer Science*, vol. 141, pp. 327–334, 2018.

[43] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[44] H. A. Elsalamony, "Healthy and unhealthy red blood cell detection in human blood smears using neural networks," *Micron*, vol. 83, pp. 32–41, 2016.

[45] K. Naruenatthanaset, T. H. Chalidabhongse, D. Palasuwan, N. Anantrasirichai, and A. Palasuwan, "Red blood cell segmentation with overlapping cell separation and classification on imbalanced dataset," *arXiv preprint arXiv:2012.01321*, 2020.

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

[47] M. Zhang, X. Li, M. Xu, and Q. Li, "Automated semantic segmentation of red blood cells for sickle cell disease," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, pp. 3095–3102, 2020.

[48] C. Patgiri and A. Ganguly, "Adaptive thresholding technique based classification of red blood cell and sickle cell using naïve bayes classifier and k-nearest neighbor classifier," *Biomedical Signal Processing and Control*, vol. 68, p. 102745, 2021.

[49] J. A. Akrimi, A. Suliman, L. E. George, and A. R. Ahmad, "Classification red blood cells using support vector machine," in *Proceedings of the 6th International Conference on Information Technology and Multimedia*, pp. 265–269, 2014.

[50] P. D. C. Divina, J. P. T. Felices, C. C. Hortinela, J. C. Fausto, F. L. Valiente, and J. R. Balbin, "Classification of red blood cell morphology using image processing and support vector machine," in *Proceedings of the 2020 10th International Conference on Biomedical Engineering and Technology*, ICBET 2020, (New York, NY, USA), p. 22–27, Association for Computing Machinery, 2020.

[51] T. J. Durant, E. M. Olson, W. L. Schulz, and R. Torres, "Very deep convolutional neural networks for morphologic classification of erythrocytes," *Clinical chemistry*, vol. 63, no. 12, pp. 1847–1855, 2017.

[52] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *International conference on machine learning*, pp. 97–105, PMLR, 2015.

[53] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4893–4902, 2019.

[54] B. B. Damodaran, B. Kellenberger, R. Flamary, D. Tuia, and N. Courty, "Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 447–463, 2018.

[55] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li, "Deep reconstruction-classification networks for unsupervised domain adaptation," in *European Conference on Computer Vision*, pp. 597–613, Springer, 2016.

[56] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," *arXiv preprint arXiv:1608.06019*, 2016.

[57] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko, "Simultaneous deep transfer across domains and tasks," in *Proceedings of the IEEE international conference on computer vision*, pp. 4068–4076, 2015.

[58] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International conference on machine learning*, pp. 1180–1189, PMLR, 2015.

[59] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The journal of machine learning research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[60] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7167–7176, 2017.

[61] X. Vasques, L. Vanel, G. Villette, and L. Cif, "Morphological neuron classification using machine learning," *Frontiers in neuroanatomy*, vol. 10, p. 102, 2016.

[62] M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE transactions on information theory*, vol. 8, no. 2, pp. 179–187, 1962.

[63] L. McInnes and J. Healy, "Accelerated hierarchical density based clustering," in *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 33–42, IEEE, 2017.

[64] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise.," in *kdd*, vol. 96, pp. 226–231, 1996.

[65] R. Caruana, "Multi-task learning," *Machine Learning*, vol. 28, pp. 41–75, 1997.

[66] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[67] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

[68] D. Pham, S. Koesnadi, G. Dovletov, and J. Pauli, "Unsupervised adversarial domain adaptation for multi-label classification of chest x-ray," in *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pp. 1236–1240, IEEE, 2021.

[69] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *arXiv preprint arXiv:1912.01703*, 2019.

[70] N. Ibtehaz and M. S. Rahman, "Multiresunet: Rethinking the u-net architecture for multimodal biomedical image segmentation," *Neural Networks*, vol. 121, pp. 74–87, 2020.

[71] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

[72] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[73] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[74] I. E. Sutherland, R. F. Sproull, and R. A. Schumacker, "A characterization of ten hidden-surface algorithms," *ACM Computing Surveys (CSUR)*, vol. 6, no. 1, pp. 1–55, 1974.

[75] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[76] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, 2020.

[77] Z. Pei, Z. Cao, M. Long, and J. Wang, "Multi-adversarial domain adaptation," in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[78] Y. Zhu, F. Zhuang, J. Wang, G. Ke, J. Chen, J. Bian, H. Xiong, and Q. He, "Deep subdomain adaptation network for image classification," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 4, pp. 1713–1722, 2020.