DESIGN AND DEVELOPMENT OF DEEP NEURAL NETWORKS ARCHITECTURES FOR IMBALANCED DATASETS IN COMPUTER VISION

A thesis submitted to

Delhi Technological University

for the Award of Degree of

Doctor of Philosophy

in

Computer Science and Engineering

by

MANISHA SAINI

(2K17/Ph.D./CSE/09)

Under the Supervision of

Prof. Seba Susan

Professor, Department of Information Technology

Department of Computer Science and Engineering

Delhi Technological University

(Formerly Delhi College of Engineering)

Delhi-10042, India

AUGUST-2023

DECLARATION

I declare that the research work reported in the thesis entitled "**DESIGN AND DEVELOPMENT OF DEEP NEURAL NETWORKS ARCHITECTURES FOR IMBALANCED DATASETS IN COMPUTER VISION**" for the award of the degree of *Doctor of Philosophy* in the *Department of Computer Science and Engineering (CSE)* has been carried out by me under the supervision of *Prof. Seba Susan*, Professor in *Information Technology*, Delhi Technological University (DTU), Delhi, India.

The research work embodied in this thesis, except where otherwise indicated, is my original research. This thesis has not been submitted earlier in part or full to any other University or Institute for the award of any degree or diploma. This thesis does not contain other person's data, graphs, or other information unless specifically acknowledged.

Date: 26th August 2023

(Manisha Saini)

2K17/Ph.D./CSE/09
Department of CSE
Delhi Technological University,
Delhi-110042, India

CERTIFICATE

This is to certify that the thesis entitled **"DESIGN AND DEVELOPMENT OF DEEP NEURAL NETWORKS ARCHITECTURES FOR IMBALANCED DATASETS IN COMPUTER VISION"**, submitted by **Ms. Manisha Saini** (Roll. No: 2K17/Ph.D./CSE/09) for the award of the degree of Doctor of Philosophy, to the Delhi Technological University, is based on the original research work carried out by her. She has worked under my supervision and has fulfilled the requirements, which to my knowledge have reached the requisite standard for the submission of this thesis. It is further certified that the work embodied in this thesis is neither partially nor fully submitted to any other university or institution for the award of any other degree or diploma.

Prof. Seba Susan

Professor, Information Technology

Delhi Technological University, Delhi- 110042, India

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to ***Prof. Seba Susan,*** Professor, Information Technology, Delhi Technological University (DTU), Delhi for her motivation, patience, guidance, continuous support and her invaluable feedback on my analysis and framing of this research work in entirety. She has widened the horizons of my thinking and promoted research skills. She is an extraordinary mentor to share her wisdom and knowledge and put her tireless efforts to support and motivate me throughout the journey of my doctoral degree. Her brilliant, skilful supervision and clarity of thought stemming from her immense experience and deep expertise in the subject enriched this study and have been a pillar of support that enabled me to bounce back from my failures on multiple occasions. I thank her for accepting me as her student and also for her confidence in me which was invaluable. This research work would not have been possible without her stimulation, inspiration, and cooperation as she is not only my guide but a mentor in life.

I am deeply grateful to the DRC chairman, *Prof. Rajni Jindal,* and my SRC members; for evaluating the performance and giving their valuable feedback frequently to improve the work. I would also like to thank *Prof. Vinod Kumar,* HOD, CSE, DTU, Delhi for providing all the essential facilities in the department.

Lastly, words cannot completely express my love and gratitude towards my family *Mr. Shiv Kumar Saini*, *Mrs. Darshana Saini,* and *Mr. Gaurav Saini* who have supported and encouraged me through this journey and this could not have been

possible without my family's strong support, love and motivation throughout my Ph.D. work. I owe a lot to my parents and brother, who encouraged and helped me at every stage of my personal and academic life, and longed to see this achievement come true. I would like to thank my husband *Mr. Harsh Saini* for his support and motivation towards the successful completion of this study. Finally, I would like to express my deep gratitude to my friends: *Dr. Shalini Gakhar, Dr. Sameer Arora*, and *Ms. Isha Khatterpal*. Without their tremendous understanding, encouragement, and support over the past few years, it would be impossible for me to complete my study.

Sincere thanks to my M.Tech supervisor *Prof. (Dr.) Rita Chhikara,* Professor and Head, Department of CSE, The NorthCap University, Gurugram, for her blessings, encouragement, and inculcating interest in research since my M.Tech. research work. Lastly, I thank the almighty God for the passion, strength, perseverance, and resources to complete this study.

Manisha Saini

ABSTRACT

This thesis involves the design and development of deep learning techniques to address the class imbalance problem in the computer vision domain. The core idea behind this study is to examine various approaches which would help to combat the impact of biases towards the majority class which could leave the minority classes undetected, that overall might lead to misleading performance results. Very limited empirical study is found in this research area of deep learning while dealing with class imbalance. Several challenges are involved while dealing with imbalanced datasets due to the unequal distribution of samples corresponding to each class present in the dataset, including biased and lower performances in minority classes. Hence it is very challenging to deal with such imbalanced problems, especially in the case of the multi-class imbalanced datasets. Different evaluation parameters also need to be considered for evaluating the overall performance of the model. Throughout the study, we have tried to bring the changes at the data level and the algorithm level by designing and developing novel deep learning techniques to deal with the imbalanced data to solve computer vision problems. For validating our approach, we have used various challenging binary and multi-class imbalanced datasets including Graz-02 dataset, TF-Flowers dataset, BreakHis dataset, Breast-Histopathological-Images dataset, Kaggle Diabetic Retinopathy dataset, DDR Dataset, Indian Diabetic Retinopathy Image (IDRiD) dataset and Intel MobileODT Cervical Cancer Screening dataset. We present insights into the design and implementation of deep learning models with imbalanced datasets of various scales. In support of the same, we have conducted a detailed curated set of experiments on the available benchmark datasets. A detailed

comprehensive experimental analysis is conducted on the datasets, comparing our results with the state-of-the-art methods in the field. Our contributions are summarized below, highlighting our key findings and innovations. We have performed a thorough analysis of multiple state-of-the-art pre-trained networks across various tasks, including classification, object detection, and segmentation on varied size datasets (small, medium, and large). These tasks were evaluated on diverse applications, such as diabetic retinopathy, breast cancer, cervical cancer, and more. Additionally, we proposed efficient machine learning classifiers, such as Chi² SVM, Quasi SVM and weighted SVM, to address the challenges posed by imbalanced datasets. These classifiers aim to mitigate the impact of class imbalance and improve overall performance. A novel model using visual codebook generation obtained from ResNet-50 deep features along with the Chi² SVM classifier is proposed to effectively tackle the class imbalance problem that arises while dealing with multi-class image datasets. Another contribution consists of exploring the effect of data augmentation on the overall performance of the deep learning models. The effect of data augmentation approaches was seen after applying (i) Traditional affine transformation (shifted, zoomed in/out, rotated, flipped, distorted, cropping, rescaling or shaded with hue, etc.) and (ii) Generative Adversarial Nets (GANs) to generate synthetic samples from the original images which makes the models more robust and also helps in resolving the class imbalance issue. We have proposed a novel learning framework in collaboration with the Deep Convolutional Generative Adversarial network (DCGAN). The DCGAN is used in the initial phase for data augmentation of the minority class only with the modified less computationally challenging VGG16 deep network architecture. The significance of adding batch normalization layers is discussed as it helps to mitigate the effect of covariance shift. Additionally, emphasis

is given to hyperparameters, and fine-tuning also plays a crucial role in the overall model performance. Major contribution is the development of novel deep learning architecture VGGIN-NET which adapts to class imbalance in both binary and multi-class datasets.

TABLE OF CONTENTS

LIST OF FIGURES

Figure 5.2. Validation accuracy and loss plot corresponding to the proposed architecture VGGIN-Net for different magnification factors (40X, 100X, 200X and 400X). Purple line indicates the start of fine tuning.

Figure 5.3. ROC curve comparison of proposed approach with state-of-the-art networks in case of (i) 40X, (ii) 100X, (iii) 200X and (iv) 400X.

Figure 5.4. Training, validation loss and accuracy while training VGGIN-Net on BreakHis dataset for different magnification factors. Pre-trained weights from the same are used to fine-tune on Breast Histopathological Images dataset.

Figure 5.5. Performance evaluation of our proposed VGGIN-Net across epochs. (a) without data augmentation and (b) with data augmentation. (Top to Bottom) Learning curves are shown for accuracy and loss for training and test sets for different magnification factors of the BREAKHIS dataset (Left to Right respectively).

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Definition
DL	Deep Learning
ML	Machine Learning
BoVW	Bag of Visual Words
DCGAN	Deep Convolutional Generative Adversarial network
GAN	Generative Adversarial network
CNN	Convolutional Neural Networks
FID	Fréchet Inception Distance
IDRiD	Indian Diabetic Retinopathy Image Dataset
ROC	Receiver Operating Characteristics
AUC	Area Under the Curve
IoU	Intersection over Union
mAP	Mean Average Precision
Chi^2 SVM	Chi-squared Kernelized Support Vector Machine
SIFT	Scale-Invariant Feature Transform
MaxAbsScalar	Maximum Absolute Scaling
IBA	Index Balanced Accuracy
Avg	Average

LIST OF SYMBOLS

Abbreviation	Definition
G	Generator Model of GAN
D	Discriminator Model of GAN
T_P	True Positive
T_N	True Negative
F_P	False Positive
F_N	False Negative
Σ_r	Covariance of the feature vector from real images
Σ_g	Covariance of the feature vector from fake images
α	Learning rate
C	Regularization parameter
ω	Weight vector
b	Bias term
k	Kernel
γ	Scaling factor in kernel function of S

Chapter 1
Introduction

In this chapter we have provided an overview of our research along with the formulated research problems and the motivation which encouraged us to carry forward the research in the same direction. The scope of the study is also added to provide the background and enhance the aspects for conducting the research, along with the research gaps, contributions and research objectives. A further detailed set of experiments along with the results' discussion to support the evidence of the research and the detailed section providing the methodology used for the creation of unique combinations of deep learning architecture is presented to tackle the class imbalance problem provided in the corresponding chapters mentioned in the thesis under each section.

1.1 Overview

Imbalanced datasets are a characteristic of most real-world applications (Provost 2000). The problem associated with an imbalanced dataset occurs due to the disproportionate number of samples present in each class. In an imbalanced dataset, the class distribution generates a bias towards the majority class due to the presence of a limited number of training samples in the minority classes. This problem of non-uniform class distribution is generally ignored by many researchers while proposing the appropriate solution to their problem domain which generally leads to misleading model performance. There is a need for incorporating appropriate measures or procedures within the model and selecting the right performance evaluation criteria

while measuring the performance of the imbalanced dataset instead of just classification accuracy as the evaluation criteria. Various traditional machine learning techniques have been proposed for imbalanced datasets in order to counter the adverse effects of class imbalanced scenarios. We have focused on the challenging aspects which could occur while dealing with the imbalanced binary and multi-class datasets.

Generally, methods involved while dealing with the class imbalance distribution are categorized into the following categories. (1) Data level techniques (2) Algorithmic level and (3) hybrid techniques. At the data level, we alter the distribution of samples by applying sort of augmentation techniques, and sampling techniques (oversampling and undersampling methods) (Oskouei and Bigham 2017). Oversampling will result in increasing the samples in the training set which might result in overfitting problems. While applying under-sampling will result in the loss of useful information which is problematic while training the model. At the algorithm level, cost or weight schema are adapted, including modification of the underlying learner or its output. In the case of hybrid systems, we can strategically combine both the sampling as well as algorithmic techniques to create hybrid versions. So throughout the study, we have tried to create several approaches by doing modifications at all three levels.

Further, we have done analysis of the effects of imbalanced training data on different convolutional neural networks (CNNs) and their respective performance on image classification, object detection, and segmentation tasks from the domain of computer vision. Over the past decade, deep learning has been rapidly achieving remarkable performance in various applications such as document analysis, computer

vision, text processing, speech recognition, as well as many other domains due to the large availability of data and computational resources both in the form of hardware and software but limited studies were found related to deep learning dealing with imbalanced datasets in comparison to the traditional machine learning approaches. Despite these advances in the technologies and wide usage of deep learning in various domains due to the availability of huge data and computational resources, only a subset of the works was found to use proper evaluation criteria and techniques for addressing class imbalanced scenarios using different deep learning architectures. Various researchers also agree with the fact that deep learning for handling class imbalanced datasets is still a subject that is less studied (Minaam and Amer 2019, Yang, et al. 2011, Fernández, et al. 2018). Many studies conducted by various researchers have also inspired us to conduct the study regarding deep learning and imbalanced datasets (Voulodimos, et al. 2018). Krawczyk, et al. 2014 also focused on the fact that there will be a good impact on the model's overall performance, if both the groups are well represented provided, they belong to non-overlapping distributions. After that, we found another study conducted including the effect of class imbalance in various settings (Japkowicz, et al. 2002) which discusses the effect of class imbalance by modifying various parameters such as training size, complexity, and degrees of imbalance. They have conducted experiments by creating artificial data sets from the original datasets. Experimental analysis proves that if there will be an increase in the complexity of the problem then it will automatically lead to an increase in the sensitivity to imbalance.

Deep learning is artificially formulated with the help of the base foundation of artificial neural networks which are inspired by biological neurons. Deep learning is

also categorized as a distinct sub-field of machine learning. In the case of deep learning, automatic feature extraction and classification steps, together take place as a single unit whereas handcrafted features are extracted separately in the case of machine learning and passed to the different machine learning classifiers. One of the basic building blocks for deep learning relies upon different layers used to construct convolutional neural networks (CNNs). CNN's are comprised of different functional layers such as convolution, pooling, batch normalization, flatten, and fully connected as well as activation layers like ReLU and softmax, etc. The initial layers consisting of convolution and pooling contribute to the feature learning stage and the dense (fully connected), flatten layers are used in the classification unit. Each convolutional layer will combine the input image with the filter which would result in convoluted features, which will be forwarded to the next layers of the neural network. Pooling layers are applied to reduce the spatial dimensions by combining similar features into smaller dimensions. Flatten layer will help to result in a single-dimensional vector and fully connected layers will connect all the neurons present in one layer with adjacent layers and softmax will ensure the probability of occurrence of each class. Also, CNN's can be categorized into two distinguished categories. The CNNs that are trained from scratch on other datasets and the second category of CNNs are pre-trained networks i.e. not trained from scratch unlike other variants of CNN which require training from scratch, they can transfer-learn their knowledge from one dataset to another. The pre-trained networks are the ones that are popularly used in a lot of applications nowadays.

There are various pre-trained networks available based upon works inspired by deep learning literature. Notably, LeNet was the first network introduced initially which was proposed by Yann LeCun (LeCun, et al. 1989) early in the year 1989

consisting of basic layers such as convolutional, pooling, and fully connected layers which laid the foundation of all modern CNNs. However, due to limited resources and GPU (Graphics Processing Unit) at that time it was not widely used in comparison to SVM and other machine learning algorithms. The AlexNet network was introduced in 2012 (Krizhevsky, et al. 2012) which later led to the popularity of CNN in various applications, especially in the computer vision domain. AlexNet as well as other advanced pre-trained networks are based upon the foundation of LeNet. AlexNet consists of an eight-layer architecture having the first five layers of the convolution along with a few max pooling layers in continuation. The remaining terminal layers correspond to three fully connected layers. In the computer vision research arena, AlexNet was able to trigger back the usage of CNNs widely, but it was only with the advent of GPUs and VGG-Nets proposed in 2014 (Simonyan and Zisserman 2014) that CNNs became popular to be used for different image classification tasks especially being recognized for its popular use in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012. Simonyan and Zisserman modified the large kernel filter size (11x11) in AlexNet with multiple 3x3 kernel filter sizes and obtained a more efficient network consisting of 138 million parameters that were trained on multiple GPUs by horizontally sharding the weights of the network across different GPU devices. GoogLeNet (the original incarnation of the Inception architecture) and its subsequent variants: Inception-V2 (i.e. Inception with Batch Normalization), Inception-V3, etc. were proposed in 2014-16 by Christian Szegedy and others (Szegedy, et al. 2016). The architecture is formulated after combining various Inception blocks together such that each Inception block is formed by stacking 1x1, 3x3, and 5x5 convolutions. In 2015, ResNet was invented by Kaiming He, et al., which is more efficient, faster, and less computationally complex (He, et al. 2016).

ResNet is formulated by combining various residual building blocks using skip or shortcut connections. Further after ResNet, various other CNN architectures were proposed by researchers such as DenseNet (Landola, et al. 2014), SqueezeNet (Landola, et al. 2016), EfficientNet (Tan, et al. 2019), etc. which address the different problems for effective CNN training persistent in previous versions of architectures and enhances image classification performance. However, artificial neural networks is not a new concept and it is known since ages, but with the huge accessibility of GPU computing, and large-scale datasets, the deep learning frameworks have accelerated traction for both productions as well as research purposes. Plus, with the advent of cloud services, training of deeper neural networks became easier and has resulted in increased research productivity while evolving, sparking interest in the research community and leading tech companies by application of deep learning in various domains especially computer vision with higher efficiency and deployability.

1.2 Motivation

We have been motivated by the fact that mostly skewed data exists in most real-world applications as samples are not equally distributed throughout all the classes. Training the machine learning or deep learning models could be challenging with the skewed and imbalanced dataset as this might have an adverse impact on the model performance. With the large availability of high-end GPUs and datasets with an increasingly high number of samples, it has become easier to train deep learning models which have brought revolutionary game-changing impact in most of the applications over other traditional approaches. So, we have strived to design and develop novel deep architectures keeping in mind the computational budget which will be able to deal with an imbalanced dataset with efficacy for both multi-class and

binary imbalanced datasets, along with analyzing the best state-of-the-art approaches used to deal with class imbalanced scenarios. We have focused more on the extraction of deep and automated features instead of hand-crafted features and exploring the role of data augmentation and fine-tuning throughout the study.

1.3 Scope of Work

Most real-world datasets have some extent of imbalanced sample distributions and finding a perfectly balanced dataset can be difficult. This study aims to help analyze and apply techniques for creating a less biased model which can be applied in various areas or sectors ranging from healthcare to object detection etc.

1.4. Research Gaps

After conducting a literature survey, the following research gaps have been observed are illustrated below:

i. Limited work has been found in dealing with Machine Learning and Deep Learning techniques to solve imbalanced data problems.

ii. Dealing with multi-class/multi-label datasets for imbalanced problems is quite challenging for most classification models

iii. Selecting the appropriate evaluation metrics while handling imbalanced datasets is generally ignored.

iv. Limited deep learning architectures have been given in the existing literature to tackle class imbalance.

v. Current advancements and focus of the machine learning and deep learning research community are generally towards model and training techniques

whereas hyperparameter tuning and adept data pre-processing methods like synthetic augmentation techniques also play an important role in imbalanced situations.

1.5. Research Objectives

The accumulative objectives obtained from the above-mentioned research gaps are as follows:

- **Objective-1**: To study and analyze new machine learning techniques for Bag of Visual Words representation for imbalanced datasets.
- **Objective-2**: Implementation of pre-trained deep neural networks for image classification using imbalanced datasets.
- **Objective-3**: Exploring data augmentation in deep learning for imbalanced data.
- **Objective-4**: Development of novel deep learning architectures for multi-class imbalance problems.
- **Objective-5**: Application to the class imbalance problem in object detection using deep learning.

1.6 Study Area and Experimental Data

We have used various datasets for analysis and evaluation for all of our proposed approaches as discussed which are enlisted and described beneath.

1.6.1 Graz-02 Dataset

Graz-02 dataset (Opelt, et al. 2004) is one of the two benchmark datasets used for one of our experimental tasks which relates to a bag of visual words and multi-class imbalanced classification. It consists of four classes: class 0 (Bike), class 1 (car), class 2 (None), and class 3 (Person) with a total of 1,476 images. Figure 1.1 shows various samples from the original Graz-02 database to depict the intra-class and inter-class similarity. It depicts the class imbalance in the Graz-02 dataset. The number of samples in class 1 (car) is higher in comparison to other classes. The distribution of the number of samples present in each class of the Graz-02 dataset is presented in Figure 1.2.

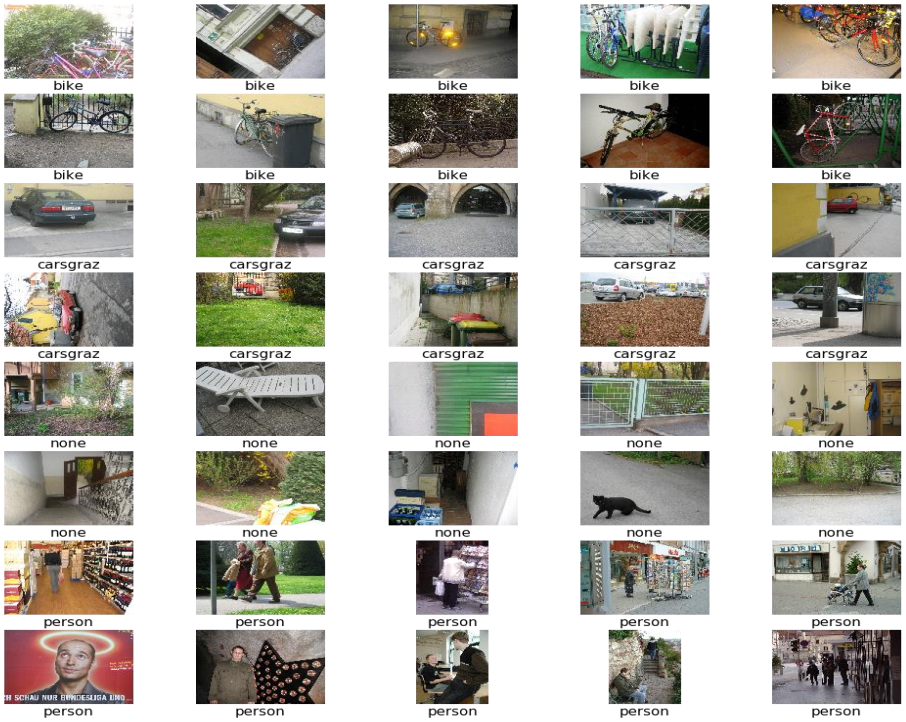


Figure 1.1. Samples from the Graz-02 dataset depicting inter and intra-class similarity among the images of the training dataset.

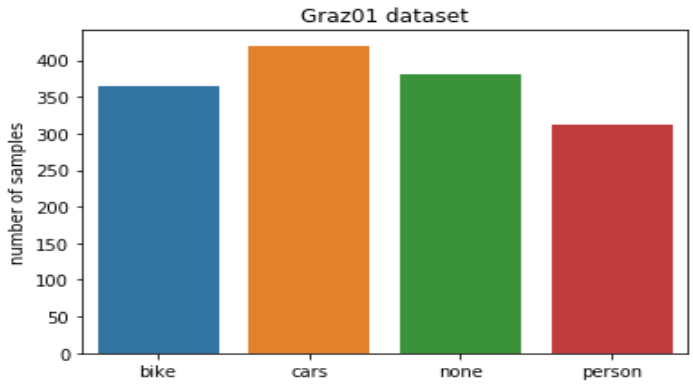


Figure 1.2. Distribution of the number of samples present in each class of the Graz-02 dataset.

1.6.2 TF-Flowers Dataset

The TF-Flowers dataset (Tensorflow, et al. 2019), samples of images shown in Figure 1.3, is another challenging imbalanced dataset that consists of images from five different categories of flowers. It contains high-quality images of flowers of different sizes and aspect ratios, which makes this dataset more challenging to tackle for the image classification task. All the images were collected specifically for deep learning workloads. The imbalanced multi-class dataset consists of different classes: Class 0 (daisy), Class 1 (dandelion), Class 2 (tulips), Class 3 (roses), and Class 4 (sunflowers), with the class distributions shown in Figure 1.4.

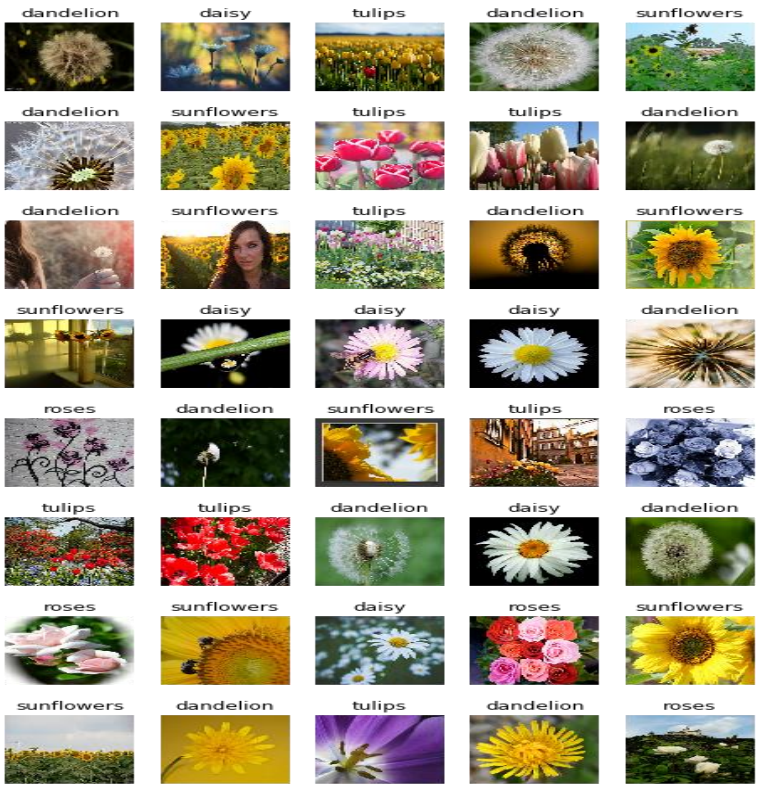


Figure 1.3. Samples of the TF-Flowers dataset depicting inter and intra-class similarity present among images of the training dataset.

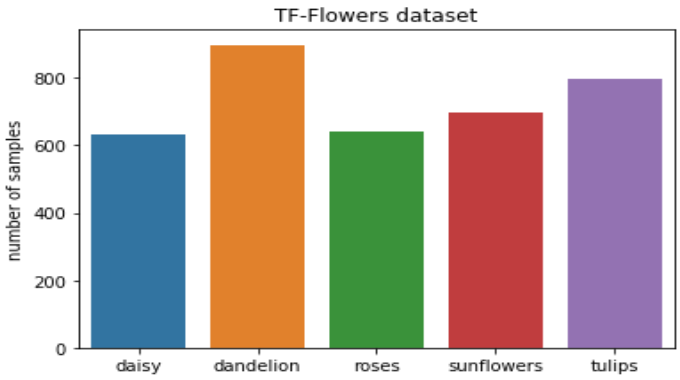


Figure 1.4. Distribution of the number of samples present in each class of TF-Flowers dataset.

1.6. 3 BreakHis Dataset

For imbalanced breast cancer classification tasks, we have primarily used the BreakHis dataset which comprises 7,909 histopathological images (Spanhol, et al. 2015). The images are jointly placed into two categories: Benign and Malignant. The

Benign and Malignant images comprise different magnification factors: 40X, 100X, 200X, and 400X; all the images are resized to 224 x 224 pixels using method of the bi-linear interpolation to make the images suitable for our image classification experiments. There is a disproportionate class distribution of samples in Benign and Malignant classes respectively. The benign class consists of fewer samples so it is considered the Minority class and the Malignant consists of more samples in comparison to the Benign so it is termed the Majority class. In Figure 1.5, the distribution of the number of samples given in each class of BeakHis Dataset is presented. The lower number of samples of Benign (Minority class) in comparison to Malignant (Majority class) indicates the class imbalance problem as illustrated in Figure 1.6.

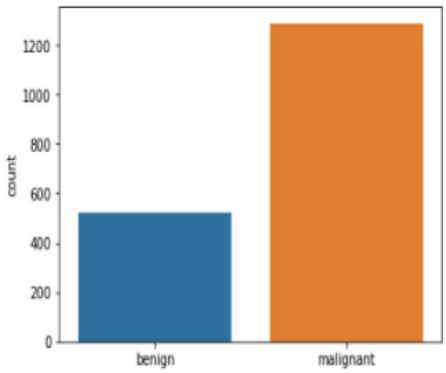


Figure 1.5. Distribution of the number of samples present in each class of BreakHis Dataset.

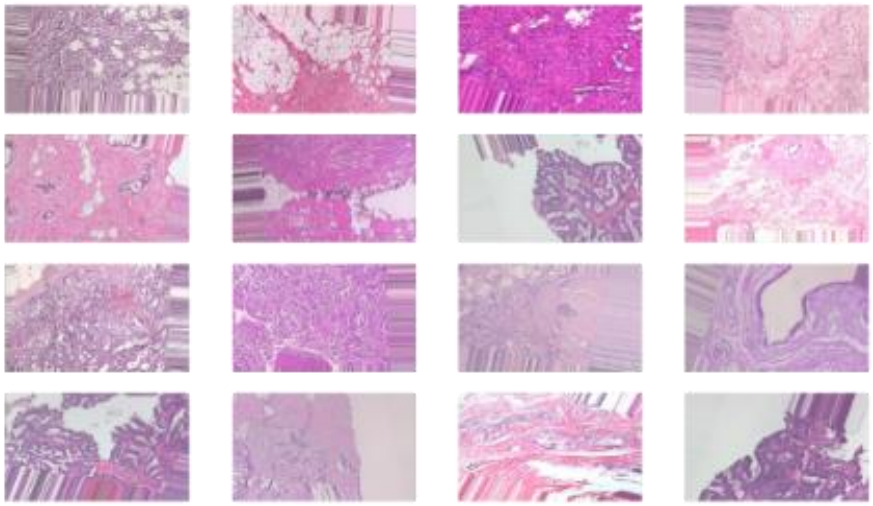


Figure 1.6. Illustration of random samples from BreakHis Dataset.

1.6.4 Breast-Histopathological-Images Dataset

We have also considered another breast cancer classification dataset, the Breast Histopathological Image dataset (Cruz-Roa, et al. 2014). The Breast Histopathological Images dataset comprises 277524 image samples having microscopic views of breast cell specimens at a 40X magnification factor. Each image patch extracted from whole slides is of size 50 x 50. The dataset tries to address growing challenges in detecting invasive ductal carcinoma (IDC), which is considered as a common type of breast cancer. A highly imbalanced class distribution is observed in this dataset with 198738 IDC –ve images and 78786 IDC +ve images. It is interesting to note in this particular dataset, images from the IDC -ve class are in majority because their count is much higher than images belonging to the IDC +ve class.

1.6.5 Kaggle Diabetic Retinopathy Dataset

It is a very large publicly available dataset that was originally published in a Kaggle competition. It consists of high-resolution images captured in varied imagining conditions, belonging to four classes: class 0 (NO DR), class 1(Mild), class 2 (Moderate), class 3 (Severe), and class 4 (proliferative DR). An illustration of a few random samples taken from different diabetic retinopathy grades from the Kaggle DDR dataset is presented. In total 53,576 and 35,126 images are present in the test and train datasets, respectively (Eyepacs 2015). The Kaggle DDR dataset is imbalanced in nature, and the test dataset has a reasonably high number of image samples existing in each class. In figure 1.7, the distribution of the number of samples present in each class of the Kaggle Diabetic Retinopathy Dataset is displayed. Our experiments have a composition of samples from the original competition: Class 0 has 31,403 samples, class 1 consists of 3042 samples, class 2 has 6282 and further classes 3 and 4 have 977 and 966 samples, respectively. It was observed that the minority class comprises 2.016% of the entire dataset.

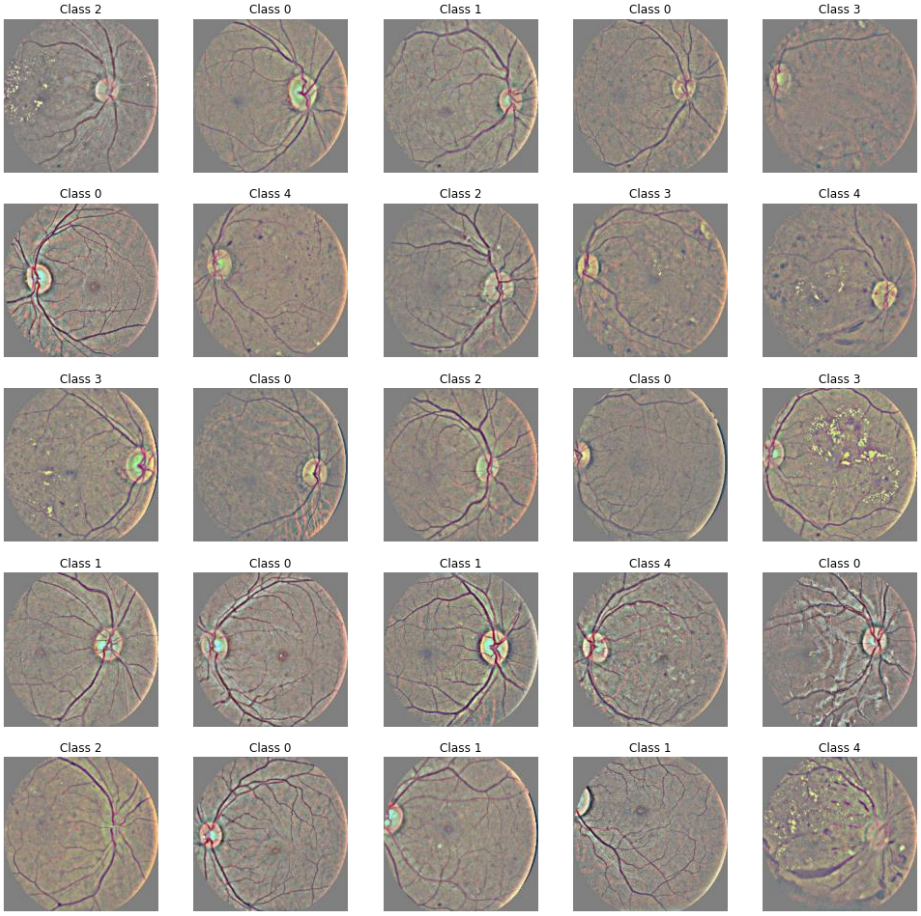


Figure 1.7. Illustration of a few random samples present in each class of Kaggle Diabetic Retinopathy Dataset.

1.6.6. DDR Dataset

We have also used the DDR dataset proposed by Tao Li et al (Li, et al. 2019) for diabetic retinopathy experimentation. The dataset consists of three categories of annotations such as bounding box, pixel, and DR grading level annotations. There are a huge number of sample images (13,673 fundus images) presented in the dataset, which was collected from 9,598 patients consisting of 6 classes. Out of 13,673 fundus images, 6835 and 2733 images, respectively, are used for training and validation, and the leftover 4105 images are taken out for testing respectively. Figure 1.8, depicts a few random samples taken from different diabetic retinopathy grades from the DDR

dataset. We found that in this dataset, the most under-represented class is around 1.726% of the total samples. In the case of segmentation, there were a few samples (very minimal, around 5) with corrupted masks in the dataset which had to be removed before training.

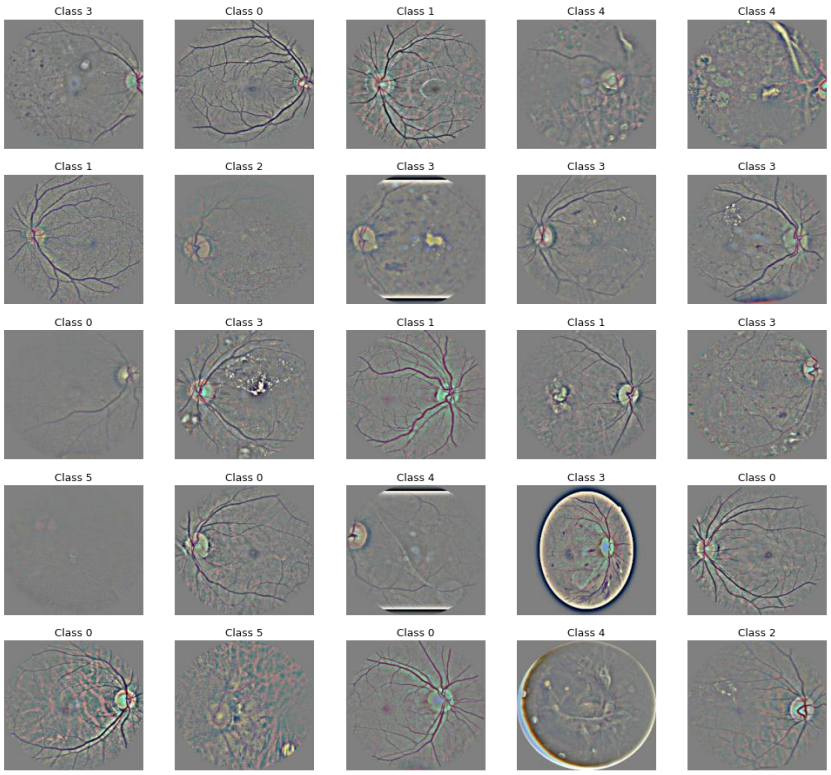


Figure 1.8. Illustration of a few random samples present in each class of DDR Dataset.

1.6.7 Indian Diabetic Retinopathy Image (IDRiD) Dataset

IDRiD is the first available diabetic retinopathy images database based on the Indian population and it consists of three parts: segmentation, disease grading (classification), and localization (object detection). The dataset consists of 4288×2848 size jpg file images (Porwal, et al. 2018). In both the cases of disease grading and localization, a total of 516 images are present in the dataset, out of which 413 images are considered for training and the remaining 103 images are kept for testing. Figure

1.9, illustrates a few random samples taken from different diabetic retinopathy grades from the IDRiD dataset. The segmentation masks consisting of the true labels are used for the coordinates of the center of the optic disc and that of the fovea center in the retinopathy images. The number of samples in the minority class constitutes approximately 4.986% of the complete dataset, rendering it to be a typical case of severe class imbalance. In the case of the localization task for this dataset, the centroids of each of the objects to be localized i.e either lesion or fovea or optic disc, and we used the centroid information to generate bounding boxes of fixed sizes using a preset radius.

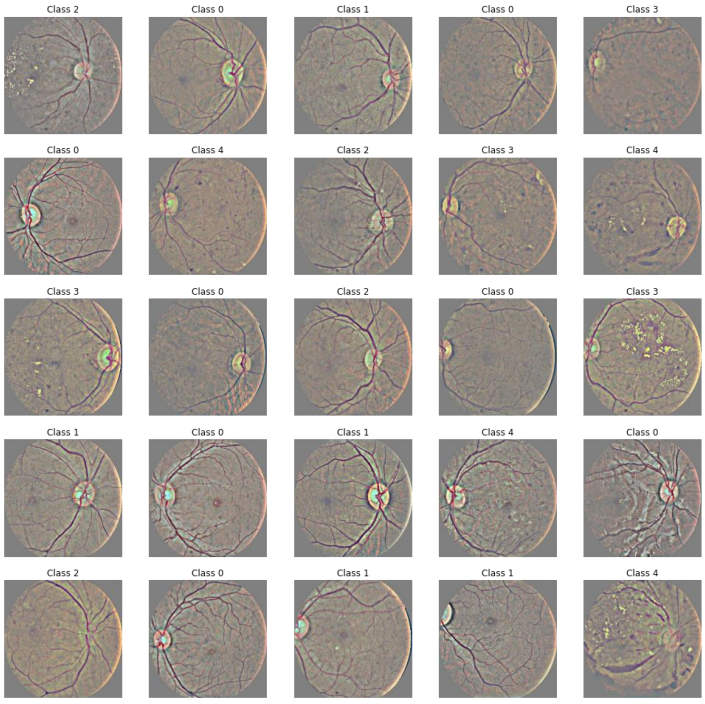


Figure 1.9. Illustration of a few random samples present in each class of Indian Diabetic Retinopathy Image (IDRiD) Dataset.

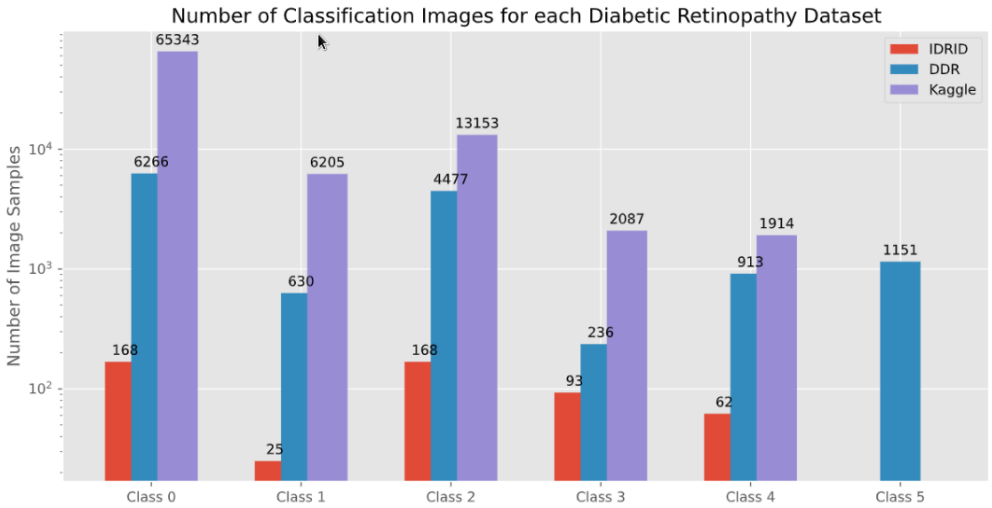


Figure 1.10. Histogram depicting number of image samples present for classification task for respective diabetic datasets.

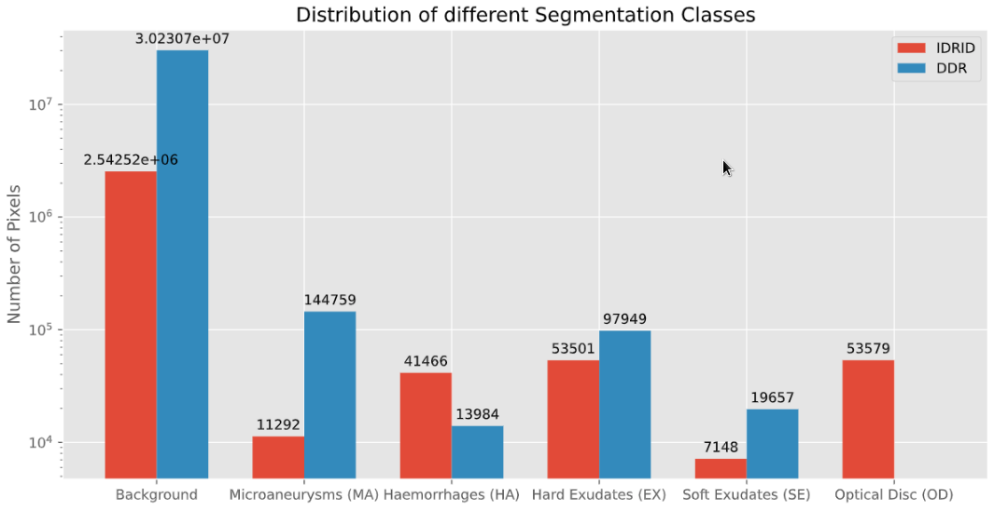


Figure 1.11. Histogram depicting number of segmentation pixels present in each class for respective diabetic datasets.

Figure 1.10 depicts the histogram consisting of the number of image samples present for above mentioned diabetic datasets. (Kaggle Diabetic Retinopathy Dataset, Indian Diabetic Retinopathy Image (IDRiD) Dataset, and Indian Diabetic Retinopathy Image (IDRiD) Dataset. Figure 1.11. Illustrates Histogram depicting number of segmentation pixels present in each class of respective diabetic datasets. Further

Figure 1.12 displays the random samples generated with the help of modified RandAugment and after applying normalization and pre-processing operations.

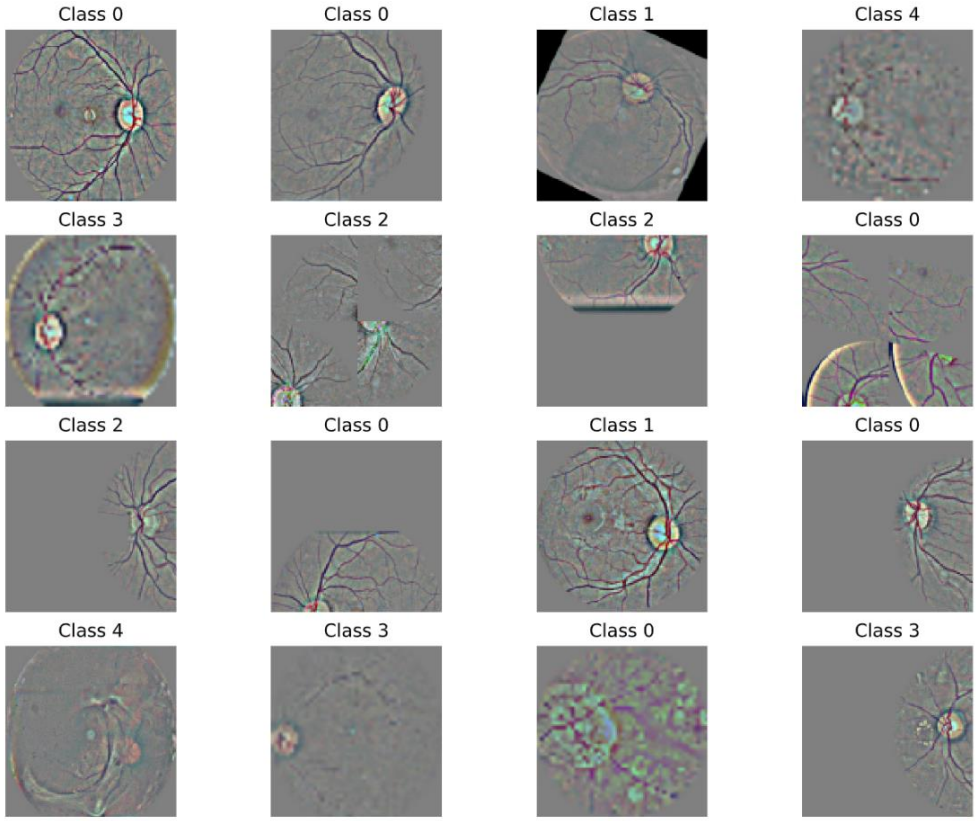


Figure 1.12. Random samples generated with the help of modified RandAugment and after applying normalization and pre-processing operations

1.6.8 Intel MobileODT Cervical Cancer Screening Dataset

MobileODT in collaboration with Intel Technologies proposed the cervical cancer screening dataset which was made publically available on Kaggle (Mobileodt, et al. 2017). The dataset consists of image samples of types of cervical cancer as shown in Figure 1.12. As observed in Figure 1.13, there are visual similarities between different types, rendering the problem to be a challenging task. This cervigram dataset consists of 3 classes (Type 1, Type 2, and Type 3). Where Type 1 refers to cervixes that are ectocervical entirely, and are fully visible, such that they could be either small

or large. Type 2 cervices have an endocervical zone but are still completely visible and may or may not have an ectocervical component either large or small. For Type 3, the endocervical zone present is not entirely visible and it might have either a small or large ectocervical component. Figure 1.14, illustrates the samples of the cervical cancer dataset depicting class imbalance distribution.

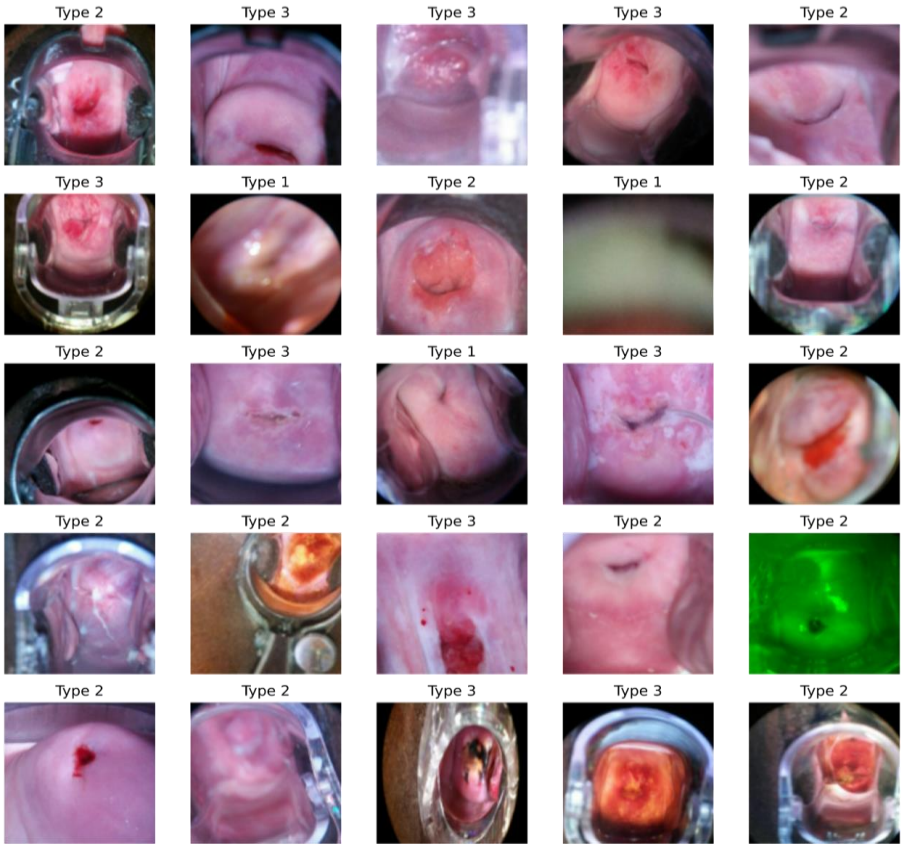


Figure 1.13. Samples of the Cervical cancer screening dataset depicting random images of each cervix type.

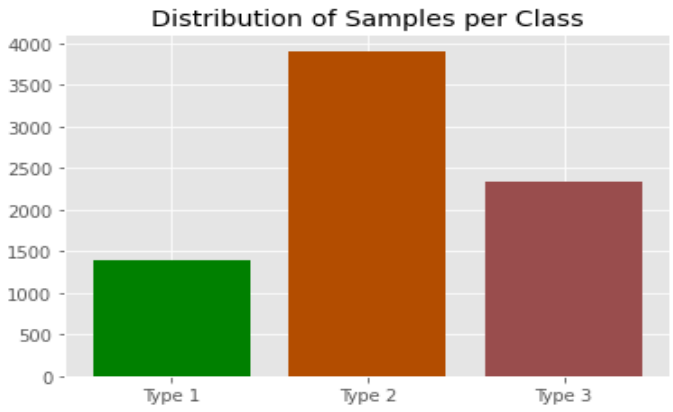


Figure 1.14. The distribution of the number of samples present in each class shows a class imbalance.

Table 1.1 illustrates the list of all datasets used for our experiments and their respective Imbalance Ratios (IR) which are used to grade the type of imbalance setting on the basis of majority and minority class for each dataset (Susan and Kumar 2019). A higher imbalance ratio is representative of more imbalance as compared to lower values of IR.

Table 1.1. Details of imbalanced datasets used in the experiments.

Dataset	**Classes**	**Minority Class**	**Majority Class**	**Minority Number of Samples**	**Majority Number of Samples**	**Imbalance Ratio**	**Imbalance Type / Grade**
Graz-02 Dataset	Bike, Cars, Person, None	Person	Cars	311	420	1.35	Low Imbalanced
TF-Flowers Dataset	Daisy, Dandelion, Roses, Sunflowers, Tulips	Daisy	Dandelion	633	898	1.42	Low Imbalanced
BreakHis Dataset	Benign, Malignant	Benign	Malignant	2480	5429	2.19	Moderately Imbalanced
Breast Histopathological Images Dataset	IDC +ve, -ve	IDC +ve	IDC -ve	78786	198738	2.52	Moderately Imbalanced
Kaggle Diabetic Retinopathy Dataset	Class 0, 1, 2, 3, 4	Class 4	Class 0	1914	65343	34.14	Extremely Imbalanced
Dataset for Diabetic Retinopathy (DDR)	Class 0, 1, 2, 3, 4, 5	Class 3	Class 0	236	6266	26.55	Extremely Imbalanced

Indian Diabetic Retinopathy Image (IDRiD) Dataset	Class 0, 1, 2, 3, 4	Class 1	Class 0	25	168	6.72	Moderately Imbalanced
Intel MobileODT Cervical Cancer Screening Dataset	Type 1, 2, 3	Type 1	Type 2	1525	4610	3.02	Moderately Imbalanced

1.7 Contribution

Our work encompasses several contributions in the field of deep learning and computer vision, specifically focusing on imbalanced datasets.

(i) We analyzed and evaluated state-of-the-art pre-trained networks for various applications such as diabetic retinopathy, breast cancer, and cervical cancer for various tasks, including classification, object detection, and segmentation on varied size datasets (small, medium, and large).

(ii) We have also presented efficient machine learning classifiers such as Quasi SVM and weighted SVM, with a specific emphasis on the effectiveness of the Chi² SVM classifier for imbalanced datasets based upon the experimental results.

(iii) We have further investigated deep feature extraction and codeword generation using histograms to improve classification performance. A novel model using visual codebook generation obtained from ResNet-50 deep features along with the Chi² SVM classifier is proposed to effectively tackle the class imbalance problem that

arises while dealing with multi-class image datasets.

(iv) To address the challenges of imbalanced datasets, we explored two forms of minority data augmentation: Deep Convolutional Generative Adversarial Networks (DCGAN) and image transformations like rotation and flipping, hue, and translation. These augmentation techniques helped to improve model robustness and resolve class imbalance problems. In collaboration with Deep Convolutional Generative Adversarial Networks (DCGAN), we proposed a novel combination using data augmentation. DCGAN was employed in the initial phase to augment the minority class, using a modified VGG16 deep network architecture. The proposed approach proves to be an effective approach to mitigate the effect of the class imbalance problem.

(v) Additionally, we highlighted the importance of batch normalization layers to mitigate the effect of covariance shift and also emphasized on the significance of hyperparameters and fine-tuning in achieving optimal model performance.

(vi) A novel deep learning architecture is proposed that can handle class imbalance in both binary and multi-class problems. Notably, we developed a novel deep learning architecture called VGGIN-NET, specifically designed to handle class imbalance in both binary and multi-class problems. VGGIN-Net which combines the benefits of the naïve Inception module with appropriate layers of VGG16 along with the combination of flatten, batch normalization, and dense layers has proved to be a significant contribution to be applied on various applications of binary and multi-class imbalanced datasets.

Overall, our contributions involved evaluating pre-trained networks, exploring data augmentation techniques applied on minority classes, utilizing efficient classifiers, extracting deep features, and developing novel architectures to address class imbalance problems.

1.8 Reading roadmap

This thesis consists of six chapters corresponding to the five objectives. The chapters are structured as follows: chapter 2 illustrates the extensive literature review. The complete description with respect to each objective, with the experimental work done along with the major findings and contributions, are presented in Chapters 3, 4, and 5. The conclusion, limitations, and future scope of the work are discussed in Chapter 6 of the thesis.

Chapter 1 mentions the introduction of the research consisting of an overview and motivation to conduct the study along with the problem formulation, contributions and study area, and a brief description of the experimental data. This chapter consists of the scope of the study along with research gaps and objectives.

Chapter 2 has extensive background literature work related to five objectives. The literature review helped us to understand the research gaps and formulate the research objectives in a compiled way. In this chapter, we have presented a tabular comparative analysis of the latest work corresponding to the study area.

Chapter 3, 4, and 5 consists of discussions about each objective which includes problem statements along with some background literature, contributions, methodology, workflow, pictorial representation along with the implementation details, results, and discussion section followed by a summary of the findings and limitations corresponding to each chapter.

Chapter 6 provides the final summary and conclusions derived after conducting the study. The same chapter also discusses the research contributions and future scope of the work.

Chapter 2
Theoretical Background

In this chapter we have done relevant literature review corresponding to the research direction emphasized in the thesis. The findings of the work done by various researchers have been outlined and summarized in the fashion by highlighting the various research gaps and limitations corresponding to the class imbalanced datasets applicable in various applications.

2.1 Literature Review

Recent advancements in deep learning observed in various fields of computer vision ranging from object detection (Hou, et al. 2017), image classification (Bosch, et al. 2007), image segmentation (Haralick, et al. 1985), human pose detection (Voulodimos et al 2017) to visual tracking (Yang, et al. 2011), etc. have been tremendously gaining traction in the computer vision research community. It has been possible nowadays because of the availability of high-performance computational machines in comparison to the previous decades. The use of graphical processing units (GPU's), advanced computing, and other hardware accelerators specifically designed for machine learning applications have made real-time processing possible. To tackle challenges associated with imbalanced datasets, various deep learning and machine learning approaches are proposed in the literature for imbalanced datasets in computer vision (Saini and Susan 2023b). The deep learning and machine learning based techniques have outgrown manifold times in the last few decades to analyze the difficult "real-world" problems characterized by imbalanced data in computer vision.

Machine learning allows computer programs to get the ability to automatically learn and improve from experience without the necessity of it being needed to be programmed manually (Minaam and Amer 2019). However, deep learning implies the automatic learning of features at many levels of abstraction that allow systems to learn increasingly complex functions (Bengio, et al. 2009). Despite having several advantages and applications, machine learning and deep learning techniques generally give a poor performance on minority class data because of differences in the distribution of data (Saini and Susan 2018). Hence, to overcome this issue, it is required to explore several methods to efficiently handle the imbalanced nature of data for binary-class and multi-class classification. Predictive accuracy itself alone might not be an appropriate measure when the data is imbalanced. There are various other performance measures available techniques (Garcia, et al. 2009) that are more appropriate while dealing with the classification task of imbalanced datasets along with the accuracy such as Receiver Operating curve (ROC) curve, Area under the curve (AUC) curve, Precision, Recall, Mathews' correlation Coefficient, Cohen's Kappa etc., that needs to be considered.

According to literature work, various researchers have extracted traditional handcrafted features for a bag of visual word approach using image classification. Suh, et al. 2018 describe an approach for distinguishing between sugar beets and volunteer potatoes with the help of Bag-of-Visual-Words model. The low-level features used for constructing the visual dictionary are Scale Invariant Feature Transform (SIFT) and Speeded-Up Robust Feature (SURF) features with Out-of-Row Regional Index (ORRI). From the experiments, it was found that the highest accuracy was obtained for the combination of SIFT with SVM, in the case of the traditional

Bag-of-Visual-Words approach. Cheng, et al. 2017 proposed a novel approach for scene classification, defined as a bag of convolutional features (BoCF). In which the visual words are extracted from deep CNN features derived from convolutional neural networks. Feng, et al. 2017 used CNN with the BOVW approach for the classification of geographical scene datasets by constructing a convolutional dictionary using pre-trained ImageNet models. SIFT features are handcrafted features that have been proved to be successful for various image classification tasks ranging from object and place recognition to face recognition (Geng and Jiang 2009). The conventional BOVW utilizes SIFT as the low-level feature based on which the visual codebook is constructed. Georgescu, et al. (2020) presented an improvisation recently, where they combined the handcrafted SIFT features with CNN extracted features in the BOVW model. They performed facial expression recognition on various datasets such as the Facial Expression Recognition (FER) challenge dataset, the FER+ dataset, and the AffectNet dataset (Georgescu, et al. 2019). However, the addition of handcrafted features with deep learning features increases the computational complexity of the classification pipeline. The traditional Bag-of-Visual-Words (BOVW) approach, as described in the literature, primarily relies on handcrafted features. However, when applied to imbalanced datasets, this approach alone often fails to yield unbiased results. The reason is that the outcomes tend to be biased towards the majority classes. The problem becomes more challenging in case of multi class imbalanced dataset. The work, therefore, emphasizes upon the classification of imbalanced multi-class image datasets in computer vision and associated challenges.

Spanhol, et al. 2016 had conducted an experiment using CNN based on deep learning for the BREAKHIS cancer dataset, and they further proposed a new approach

using image patches with the sliding window mechanism. Litjens, et al. 2016 presented a deep learning based tool and CNN that can effectively perform cancer detection using a network trained on histopathological images of prostate and breast tissue by applying patch extraction. Another variant of CNN, pre-trained networks are already trained on a larger standard dataset, and knowledge is transformed from one domain to another domain which makes it quite useful in various sectors. Many researchers have worked on pre-trained networks in the biomedical field, such as Xie, et al. 2019 who conducted supervised and unsupervised experiments using a transfer learning approach. From the supervised classification results, it was found that the Inception ResNet-v2 pre-trained network performs well. Deniz, et al. 2018 proposed concatenating the features extracted by combining the layers from AlexNet and the VGG16 pre-trained model. The new features were learned by support vector machines (SVM) for the breast cancer classification task. Garud, et al. 2017 proposed a pre-trained GoogleNet based approach for separating Benign samples from Malignant using microscopic high magnification multi-view samples. From the literature survey, it is noted that many researchers have indeed applied CNN and pre-trained models for overall improvement in the performance of the model used in the diagnosis and detection of breast cancer. But there are still various issues that need to be carefully dealt with, such as the imbalanced nature of the biomedical datasets, which occurs because of the uneven distribution of samples related to cancerous and non-cancerous cells. To address the class-imbalance problem in data mining (Susan and Kumar 2019, Susan and Kumar 2018), various methods have been proposed such as oversampling, undersampling, and hybrid sampling approaches. Whereas in computer vision, the general method of data augmentation has been applied in several works for improving class distribution and the overall model performance. Various data augmentation

operations were applied to achieve better performance by including affine transformation operations on images such as rotation, scaling, translating, etc. and further comparative analysis is conducted by comparing various configuration combinations of CNN with other traditional approaches (Howard 2013). From the study conducted, they have also emphasized the role of data augmentation operations on the overall model performance. From the performance analysis, it was validated that the deep features extracted using CNN outperform the handcrafted features. Antoniou, et al. (2017), proposed the use of a generative adversarial network for synthetically increasing data samples. It was observed that the data augmentation technique was able to adapt to the training distribution and improved the performance of classifiers on various datasets vastly. Their approach was able to improvise the classification results on VGG-Face and EMNIST datasets (Antoniou, et al. 2017). Also, Lyu, et al. 2020 proposed a novel de-noising approach based on a generative adversarial network variant that made use of VGG architecture. Frid-Adar, et al. 2018 presented a novel model in Generative Adversarial Networks (GAN) using synthetic data augmentation for liver lesion image dataset. The literature extensively covers traditional approaches such as data augmentation, training CNN networks from scratch, and utilizing pre-trained networks. However, some methods have predominantly focused on evaluating the model's performance based solely on accuracy metrics, disregarding the crucial aspect of ensuring sufficient representation of minority class sample images during model training. This oversight hinders fair evaluation. Furthermore, the importance of applying different data augmentation techniques specifically to the minority classes has been overlooked. By incorporating separate data augmentation approaches for minority classes, the model can effectively learn from both majority and minority class samples, improving its ability to handle

imbalanced datasets. It is important to address these issues and consider a more comprehensive evaluation approach that takes into account not only accuracy, which provides a more holistic assessment of the model's performance on imbalanced datasets. Additionally, giving proper attention to data augmentation strategies for minority classes can lead to more robust and accurate models.

Hagos and Kant 2019 highlighted the application of the Inception-V3 pre-trained model on a smaller subset of diabetic retinopathy detection dataset by considering accuracy as an evaluation measure and achieved 90.9% accuracy. For an imbalanced dataset, there are many other reliable evaluation metrics other than accuracy which should be considered as well while measuring the performance of the model which are missing in many works. Thota and Reddy 2020 proposed a variant of the VGG16 pre-trained model for performing the classification task on the Eye-PACS dataset whose performance evaluation was done using different metrics such as sensitivity, specificity, and AUC along with accuracy. Lam, et al. 2018 used GoogleNet and AlexNet pre-trained models on the Messidor-1 diabetic retinopathy dataset. Various studies have been conducted where many researchers have taken the concatenated features from different pre-trained networks and used them as input in other machine learning classifiers. Kassani, et al. 2019 had shown improved performance by considering Xception as the pre-trained network for the purpose of feature extraction. The extracted features were passed to a multi-layer perceptron neural network, on the APTOS dataset (Kassani, et al. 2019). Wan, et al. 2018 obtained high performance on the publicly available Kaggle DRD dataset by using VGGNet outperformed other pre-trained networks such as AlexNet, GoogleNet, and ResNet, and they also emphasized the role of fine-tuning and transfer learning.

Numerous works in literature have been conducted related to object detection and segmentation using various diabetes datasets. Zhang, et al. 2020 proposed a ConvNet formulated by combining a few convolutional and deconvolutional layers along with the Fully Connected (FC) and softmax layers to distinguish between the presence and absence of microaneurysms in retinal fundus images. Oliveira, et al. 2021 proposed a variant of Faster R-CNN, with data augmentation, for Diabetic foot Ulcer Detection. Porwal, et al. 2018 reported results compiled by various researchers who took part in the grand diabetic retinopathy segmentation and grading challenges on the Indian retinopathy Image dataset (IDRiD). The existing literature lacks a comprehensive analysis of deep learning models on imbalanced datasets of varied sizes. This includes datasets with varying sample sizes, ranging from very small to larger datasets, and covering all classification, segmentation, and object detection tasks within a single module on a single application domain. This limitation hampers the ability to capture useful insights and gain an in-depth understanding of the challenges associated with imbalanced datasets. To address these limitations, it is crucial to conduct comprehensive research that encompasses datasets of different sizes, covering various classification, segmentation, and object detection tasks along with the evaluation metrics for an imbalanced dataset. This would contribute to the development of more robust and effective customized approaches for handling imbalanced datasets in deep learning applications which can be deployed in the real-world scenario.

From the overall analysis, it was interpreted that the top performance approaches involved some form of data augmentation or ensemble models. Another major finding was that resolving the imbalance problem would lead to a tremendous improvement in the overall performance of the model. So dealing with class

imbalance is an important aspect, along with the overfitting problem, to generally enhance the performance of the network. Data Augmentation is a regularization method (Howard 2013) that helps to resolve the overfitting problem which generally occurs in deep learning models. It is equally important to select the correct set of data augmentation operations to overall model improvement. However, selecting the correct set of data augmentation operations is a tedious and manual process and also it is quite difficult to design the correct pipeline of operations. So, in literature, certain automated approaches are discussed to choose the correct set of data augmentation operations such as AutoAugment (Cubuk, et al. 2019) and RandAugment (Cubuk, et al. 2020). AutoAugment is a computationally expensive process and it is formulated by the Proximal Policy Optimization (PPO) algorithm with a large search space of different augmentation operations and their magnitudes. However, the RandAugment approach proposed by the same authors removed the PPO algorithm resulting in a much smaller search space. A simple grid search-based tuning approach reduces the computational complexity as well as the search space, deeming it to be one of the most effective automated approaches for data augmentation of CNN. The work presented in the literature served as inspiration for us to apply RandAugment, which has proven to be beneficial in improving the overall performance, particularly when dealing with imbalanced datasets. Table 2.1, illustrates some of the recent deep learning methodologies.

Table 2.1. Survey of some recent deep learning methodologies for imbalanced problems.

Method	Data Pre-processing	Distinctiveness factor / Achievements	Performance Evaluation	Limitations	Datasets	Experimental details
Imbalanced Classific	Feature Normalization	A novel model based on Generative	Average Precision, ROC AUC	The proposed method is	KEEL and CelebA datasets	• Cycle GAN

ation via a Tabular Translation GAN (Gradstein, et al. 2022)		Adversarial Networks is proposed which uses regularization losses to map majority samples to corresponding synthetic minority samples.		evaluated on a small number of datasets, so it is not clear how well it would perform on other datasets.		• Learning rate of 10−4 • SELU • activation • Batch size: 64, 128, 256
HardVis: Visual Analytics to Handle Instance Hardness Using Undersampling and Oversampling Techniques (Chatzimparmpas, et al. 2022)	Undersampling and Oversampling	HardVis, the system is designed for visual analytics to handle instance hardness in imbalanced classification settings.	Precision, Recal, F1-Score, AUC	The comparison of the proposed system to other state-of-the-art systems for handling instance hardness is missing.	UCI ML iris flower datasets	• kNN algorithm • Undersampling + Oversampling approach
Imbalanced Classification via Explicit Gradient Learning From Augmented Data (Yasinnik, et al. 2022)	Data Augmentation, Feature Normalization	Proposed method for classification using explicit gradients which demonstrates performance on synthetic and real-world datasets with various imbalance ratios.	ROC AUC, Confusion Matrix, F1-Score	The proposed method may not be effective for datasets with a very high imbalance ratio and with a large number of features.	Kohavi Adult dataset	• PCA from 101 to 64 • Learning rate 10−3 • β1 = 0.5, β2 = 0.999 • 128 batch size is taken • 200 epochs
Multi-loss ensemble deep learning	Feature Normalization, Data Augmentation	Ensemble model trained using various loss functions that exhibit superior	Matthews correlation coefficient (MCC), F1 score,	The proposed method may be sensitive	Montgomery TB CXRs and	• CCE with entropy-based

for chest X-ray classification (Rajaraman, et al. 2021)		performance on pediatric chest X-ray (CXR) dataset.	Confusion matrix	to the choice of hyperparameters.	pediatric pneumonia dataset	regularization • data is augmented using random affine transformations • Tensorflow Keras 2.4
Deep Metric Learning Model for Imbalanced Fault Diagnosis (Gui, et al. 2021)	Oversampling	Deep metric learning model is designed for imbalanced data pairs and a quadruplet loss function which takes into account the inter-class distance and the intra-class data distribution together.	AUC, Precision, Recall, F1 Score	Comparison of the proposed method to other state-of-the-art methods for imbalanced fault diagnosis is missing.	Tennessee East-man process dataset and CWRU bearing fault dataset	• Quadruplet and softmax loss combination was used. • LSTM feature extraction • Feature embedding • Quadruplet Pair Mining
Improving Model Accuracy for Imbalanced Image Classification Tasks by Adding a Final Batch	Feature Normalization	VGG19 model modified with incorporation of Batch Normalization before the output layer to enhance performance on minority class.	Sensitivity, Specificity, Mean class accuracy	Comparison of the proposed method to other state-of-the-art methods for imbalanced image classification is not present.	Wall crack and skin cancer datasets	• Batch normalization layers • Lower learning rates

Normalization Layer: An Empirical Study (Kocaman, et al. 2021)						
Deep Synthetic Minority Over-Sampling Technique (Mansourifar and Shi. 2020)	Feature Normalization	To train the inputs and outputs of the SMOTE, a deep neural network regression model was used.	Precision, Recall, F1 Score	The computational complexity of the proposed method is ignored.	WBC, Pima, Parkinson, and other disease datasets	• DA-SMOTE and GAN applied
Generative Adversarial Networks for Failure Prediction (Zhen, et al. 2019)	Feature Normalization, Data Resampling	Algorithm proposed for failure prediction using two GAN networks is used together for generating fake samples.	AUC, Precision, Recall, Confusion Matrix	This approach is difficult to be applied to smaller datasets.	APS, CMAPSS dataset	• Re-sampling (oversampling/ under-sampling) and cost-sensitive learning • Mini-batch SGD • Weighted Loss, SMOTE, ADASYN

2.2 Methodology

2.2.1. Classification

Classification is one of the most studied areas under the spectrum of supervised learning and in this case, the class label for the given task is well known in advance. It is usually regarded as a global labeling task as one single label is present for the complete image. The model is trained using the training dataset having multiple images so that if any unknown sample images from the test dataset are passed to the trained model, it can predict the category or class to which the test sample belongs. The predicted output is matched with the target output to check how well the model has learned. With the recent advancements in high-performance computing devices, deep learning is being widely used for various image classification workloads nowadays.

2.2.2. Object Detection

Object detection also comes under the periphery of supervised machine learning but it is regarded as a sparsely labeled task since the information available in the form of class labels is around selected regions containing different foreground and background classes. Object detection is used to determine the presence of an object in an image by creating the bounding box around the object. The objective of object detection is to not only determine the presence of objects but also to localize them within the image by providing the coordinates of their locations The object detection task can fall into either the single or multi-class category depending on presence of multiple objects in the image. Pathak, et al. (2018), highlighted the usage of CNN-based deep learning approaches and discussed their utility in object detection tasks

across various fields, including robotics, surveillance, transportation, autonomous driving, and the medical domain, etc.

2.2.3. Segmentation

In image segmentation, the input image is divided into distinct regions or segments where each pixel is labeled with class labels. Each region has pixels having similar characteristics, allowing for the identification and delineation of objects or boundaries within the image. In other words, segmentation is the process used to find boundaries to locate objects. Labels are provided to each pixel in the images such that the multiple pixels having similar characteristics are assigned the same label. To perform segmentation we can use multiple pre-trained networks. In the basic CNN architectures for segmentation, the model usually has an encoder and a decoder, which work together to facilitate the segmentation process. However, the encoder is used for extracting high-level features from the input image, while the decoder further takes these features extracted from the encoder and provides a segmentation map that assigns class labels to each pixel. The combined encoder-decoder in the architecture helps to capture minute details from the image and hence allows for more accurate segmentation results.

2.3 Evaluation Measures

To evaluate the performance of different classification models, we considered multiple evaluation metrics including Accuracy, Precision, Recall, F1-score, Geometric Mean, Index Balanced Accuracy, Cohen’s Kappa, Matthew's Correlation Coefficient as well as Receiver Operating Characteristics (ROC) and it’s an area under the curve (ROC AUC). Accuracy determines how well the model can

distinguish the different samples belonging to different classes based on true label and predicted label (Japkowicz 2013). Precision is the degree of how close the predicted and real labels are to each other whereas recall is the probability of correct samples that can be classified from the data. F1 score combines precision and recall in a way such that it is able to approximately measure how close the predicted and true samples distribution are to each other. Mathematically, F1-score can be defined using harmonic mean of recall and precision. It is suggested that accuracy alone cannot be the sole parameter to gauge the model's performance in tackling class imbalance scenarios. Hence, the F1-score must also be taken into consideration to analyze the model's performance while dealing with imbalanced datasets. Matthew's correlation is used to find the correlation that occurs between actual and target variables for binary classification. Moreover, Cohen's Kappa is used to calculate the agreement that occurs between the real and the predicted model. The geometric mean is used to calculate the confidence interval between the distribution of true labels and predicted labels. Index Balanced Accuracy is a measure used to evaluate both multi-class and binary classification tasks. It combines the measure of overall accuracy with the accuracy corresponding to the class with the highest accuracy. This effectively describes the performance of a classifier trained on an imbalanced dataset. Receiver Operating Characteristics curves (ROC curve) along with area under the curve helps to depict the performance of a classification model at different thresholds based on false positive and true positive rate.

In the case of segmentation, we consider Intersection over Union (IoU) and Dice Score as the primary evaluation metrics. Intersection over Union (IoU) is known as Jaccard Index which is used to obtain the measure of correctness of a segmentation

or even object detection model. It is expressed as the ratio of the area of overlap divided by area of union between the regions as provided by ground truth labels for segmentation compared to the class labels of the region predicted by the segmentation model. Dice Score (Sørensen Index) is the measure of similarity between segmented regions derived from known true labels and that predicted by the classifier. It is derived from the harmonic mean between the area of overlap along with the area of union (total number of pixels considering both regions).

For object detection, we evaluate the different models based on mean average precision and mean average recall at fixed thresholds of intersection over the union. AP is formulated for each detection class and we averaged it to obtain the mAP. The mean average precision or mAP score is obtained by using the mean AP over all classes and overall IoU thresholds, based on various detection regions of foreground and background. Average recall describes the area doubled under the Recall multiplied by the IoU curve. Similar to mAP, mAR is the mean of average recalls over the different number of classes within the dataset. Table 2.2 illustrates the mathematical equation of different evaluation criteria for image classification, object detection, and segmentation tasks.

Table 2.2. Various performance metrics used for the evaluation.

Evaluation Parameters	
Accuracy	$\frac{T\_P + T\_N}{T\_P + T\_N + F\_P + F\_N}$
Precision	$\frac{T\_P}{T\_P + T\_N}$

Recall	$\frac{T\_P}{T\_P + F\_N}$
F1-Score	$\frac{2 * (Precision * Recall)}{(Precision + Recall)}$
Matthew's Correlation Coefficient	$\frac{T\_P * T\_N - F\_P * F\_N}{\sqrt{(T\_P + F\_P)(T\_P + F\_N)\ (T\_N + F\_P)(T}}$
Cohen Kappa	$\frac{p_0 - p_c}{1 - p_0}$
Geometric Mean	$T\_P\_rate = T\_P/(T\_P + F\_N)$ $T\_N\_rate = T\_N/(T\_N + F\_P)$ $Gmean = \sqrt{T\_P\_rate * T\_N\_rate}$
Index Balanced Accuracy	$Dominance = (T\_P\_rate - T\_N\_rate)$ $IBA = (1 + Dominance) * Gmean^2$
Intersection over Union (IoU)	$\frac{Area\_of\_Intersection}{Area\_of\_Union}$
Dice Score	$\frac{2 * Area\_of\_Intersection}{Area\_of\_Union}$

Chapter 3
Machine Learning and Deep Learning Techniques for Multi-class Imbalanced Dataset in Computer Vision

From the literature review conducted we have formulated research objectives. The research objectives of this study aim to address the limitations identified in the existing literature and make significant contributions in the field of imbalanced dataset analysis using machine learning and deep learning models to mitigate the effect of class imbalance problem. In Chapter 3, a novel approach is proposed by combining deep features from pre-trained deep neural networks with an appropriate classifier using the traditional Bag of Visual Words based machine learning approach. Additionally, the study involves implementing various pre-trained deep neural networks and selecting the most suitable model to address class imbalance issues in classification, object detection, and segmentation. The research explores the significance of working with deep learning models on imbalanced datasets of varying sizes, ranging from small to large, within a single application domain. Various evaluation metrics are employed to assess the effectiveness of the models for classification, segmentation, and object detection tasks. All approaches are designed specifically to tackle the challenges associated with multi-class imbalanced datasets.

: [1] The contents of this chapter are published in "Bag-of-Visual-Words codebook generation using deep features for effective classification of imbalanced multi-class image datasets." Multimedia Tools and Applications 80, no. 14 (2021): 20821-20847 and "Diabetic retinopathy screening using deep learning for multi-class imbalanced datasets." Computers in Biology and Medicine 149 (2022): 105989.

This chapter is divided further into two subsections where section 3.1 consists of the discussion of the methodology adopted for the creation of the modified Bag-of-visual-words representation using the extracted deep features and machine learning approaches, illustrated along with the experimental setup and result analysis covering objective 1. Section 3.2. consists of objective 2 and 5, where implementation of pre-trained deep neural networks for image classification using imbalanced datasets and application to the class imbalance problem in object detection using deep learning is illustrated in depth.

3.1. Objective 1: To study and analyze new Machine Learning technique for Bag of Visual Words Representation for Imbalanced datasets

In the computer vision domain one of the most challenging tasks is to classify images into different categories. There are numerous challenges associated with the image classification task such as viewpoint, partial occlusion, clutter, illumination, and inter-class and intra-class visual diversity which causes difficulty for models to classify effectively. The Bag-of-visual words (BoVW) model gained traction in the research community for performing image classification tasks. Bag of Visual Words representation (Suh, et al. 2018) formulated from the Bag-of–words (BOW) concept, similar concept is used in BOVW to be applied to image classification problems by extracting the visual words from the trained vocabulary as features to classify images. Various features can be extracted such as SIFT and SURF, etc. instead of color, shape, and texture for classification for the BOVW approach (Saini and Susan 2021). We have modified the original BOVW approach by choosing the right set of features and

classifiers and further a combination of visual codebook generation using deep features along with the Chi² SVM classifier (Bellet, et al. 2013, Bellet, et al. 2015) is proposed in order to tackle the imbalance problem that occurs while dealing with various multi-class datasets. In the first stage, deep features are extracted using the ResNet-50 pre-trained network, and further, those extracted features are clustered together using the k-means algorithm. Each image is then converted into a features set called the Bag-of-Visual-Words (BOVW) which is derived from the histogram counts of visual words in the trained vocabulary. For performing these experiments on our proposed BoVW approach we have considered two imbalanced datasets. The problem becomes more challenging and complex while dealing with multi-class imbalanced datasets as there is difficulty in addressing the multi-class imbalance problem because of the random and haphazard manner in which samples are distributed into multiple classes. We seek to incorporate the deep features extracted from the latest state-of-the-art pre-trained networks into the traditional BOVW approach for constructing the visual codebook or dictionary (Saini and Susan 2021). Specifically, we focus on the features extracted from the last residual block of the fifth convolutional layer just preceding the global average pooling and dense layer of the pre-trained model ResNet-50, defined as the Res5c features (Mahmood, et al. 2017). The 3D features are of the form HxWxC for an input image, where H and W are the height and width of the 2D feature maps and C denotes the number of channels in the last residual block of the fifth convolutional layer that is used for feature extraction. In this work, we have utilized the deep features extracted from pre-trained deep convolutional neural networks (CNN) by transfer learning. The extracted features are used for codebook generation in the Bag-of-Visual-Words (BOVW) model that traditionally creates a visual dictionary from handcrafted features. Features extracted

using the transfer learning approach with pre-trained models have certain benefits over the handcrafted features. The pre-trained networks are already trained on the large-scale ImageNet dataset (Krizhevsky, et al. 2012) which overall reduces the training time required to train the model and also leads to the extraction of the most relevant low-level features for codebook generation. Furthermore, the non-linear Chi^2 SVM using the one-versus-all scheme is found to be an optimal choice of classifier, from our experiments, while dealing with the multi-class imbalanced datasets. There are the following contributions of this work: (i) Successfully applied transfer learning approach to extract deep features using pre-trained models, for generating the visual codebook in our improved Bag-of-Visual-Words model (ii) Conducted an extensive empirical analysis to find the optimal set of deep features for codebook generation and the appropriate classifier to resolve the class imbalance problem prevalent in two of the benchmark datasets (iii) Analyzing the role of chi-squared Kernelized SVM as an effective classifier for the histogram features. Additionally, we experiment with a set of neural networks used to approximate linear as well as kernel SVMs to investigate an alternative learning methodology for dealing with large-scale settings (Wang, et al. 2019, Rahimi and Recht 2008).

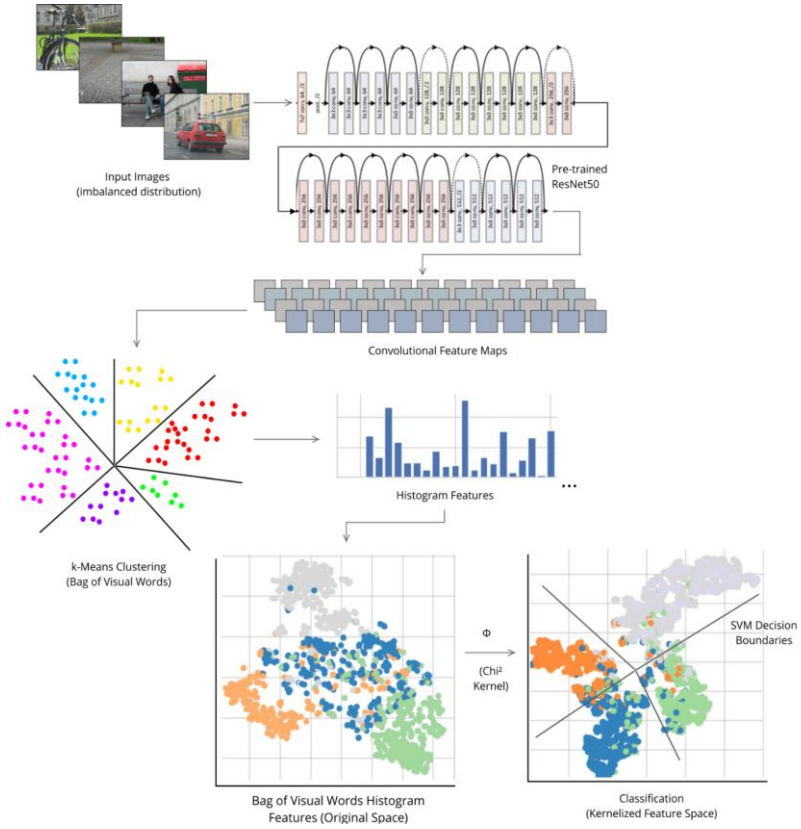


Figure 3.1. Proposed Bag-of-Visual Word codebook generation from deep features and subsequent classification using Chi² SVM.

The complete process of BOVW can be categorized into the following steps (Suh, et al. 2018) (1) Low-level feature extraction (2) Feature clustering and quantization for the generation of the visual vocabulary (3) Classification using a suitable classifier. Initially, from each image, 128-dimensional keypoint descriptors are extracted. This procedure pertains to the most commonly used feature descriptor for BOVW namely, the Scale-Invariant Feature Transform (SIFT) which are classified as handcrafted features, a genre that preceded deep learning. The keypoint descriptors are then clustered. The cluster centers are interpreted as visual words, and in this manner, a visual vocabulary is formulated. The feature vectors for classification are derived from the normalized frequency of occurrences of the visual words in an input image. It is noted that the Support Vector Machine (SVM) classifier

is popularly used for classifying the BOVW features. The proven success of the BOVW model motivated us to explore the incorporation of deep features derived from pre-trained networks into the existing scheme for efficient classification, with a special focus on improving the class-wise performance of imbalanced multi-class datasets. We have performed extensive experiments to find out the optimal choice of low-level deep features to be integrated into the BOVW approach, along with the optimum choice of classifier, to overcome the biasness problem that arises due to unequal class distribution. Our end-to-end approach is further detailed in Algorithm 3.1 mentioned below and illustrated in Figure 3.1.

Algorithm 3.1. ResNet-50 Deep Feature Extraction for Bag-of-Visual-Words codebook generation with Chi² SVM Classification

Input :
Dataset, D (IMAGES, LABELS)
Output:
Performance metrics for Test set after classification by Chi² SVM classifier (CLF)

Construct a model V_f using all layers from pre-trained ResNet-50 model V except final GAP and Dense layer
Construct a new list for all deep features of all images, X_{deep}
Construct a new list for all class labels, Y
M ← Number of classes in D
for each (IMAGE, LABEL) in D **do**
    IMG ← Preprocess IMAGE (using mean normalization)
    FV ← Use model V_f to extract 3D feature of dimension 7 x 7 x 2048 from IMG
    Append list X_{deep} with FV
    Append list Y with LABEL
end for
X ← Concatenate the feature vectors X_{deep} into 2D matrix of size 49 x 2048
Create an empty list for storing visual word features, X_{bovw}
KM ← Apply k-means clustering algorithm on X using k number of clusters
for each FEATURE in X **do**
    NC ← Determine which cluster each vector in FEATURE belong to using KM
    Create empty array F of size k
    for I in [1….k] **do**
        F[I] ← 0
        Increment F[I] with number of occurrences of I in NC
    end for
    Append list X_{bovw} with the histogram features F, and normalize the value in each feature column to [0,1] using MaxAbsScaler
end for
Instantiate a Chi² SVM classifier, CLF, and divide X_{bovw} into X_{train} and X_{test}, and create corresponding target vectors Y_{train} and Y_{test} from Y
X_{chi^2} ← $K_{chi^2}(X_{train}, X_{train})$, according to Eq. (3)

Fit support vector machine CLF using (X_{chi^2}, Y_{train}), according to Eq. (2)
Create an empty list Y_{pred}
for each F in X_{test} **do**
    C_{pred} ← Use support vectors from trained CLF to predict class label of F
    Append list Y_{pred} with C_{pred}
end for
ACC ← Calculate accuracy using (Y_{test}, Y_{pred})
Create new lists for storing precision, recall, and F1-score: P, R, F1
for each class J in [1..M] **do**
    P[J], R[J], F1[J] ← Calculate and store F1-score, precision, recall using (Y_{test}, Y_{pred})
end for

3.1.1 Deep Feature extraction using Pre-trained Models

Traditional BOVW uses handcrafted features such as scale-invariant feature transform (SIFT). In the last decade, deep learning became one of the most sought-after techniques for image classification while traditional approaches like BOVW were almost pushed into obsolescence. Computational challenges have not deterred the deep convolutional networks from advancing to the state-of-the-art today in the form of pre-trained deep networks trained on millions of images. Then we have used the deep features extracted from the latest state-of-the-art pre-trained networks in the traditional BOVW approach for constructing the visual codebook or dictionary. Specifically, we focus on the features extracted from the last residual block of the fifth convolutional layer just preceding the global average pooling and dense layer of the pre-trained model ResNet-50 (He, et al. 2016), defined as the Res5c features in (Mahmood, et al. 2017). The 3D features are of the form H x W x C for an input image, where H and W are the height and width of the 2D feature maps and C denotes the number of channels in the last residual block of the fifth convolutional layer that is used for feature extraction. The set of features is converted into two-dimensional (H x W) feature vectors, each with the dimension of C. From the ResNet-50 architecture, we see that H=7, W=7, and C=2048. Thus 49 deep feature vectors each of dimension 2048 are extracted from each input image. The total number of feature

vectors extracted from the training set is no. of training images x 49, with each feature vector having the dimension of 2048. The deep features are used for constructing the visual vocabulary associated with the Bag-of-Visual-Words model.

ResNet is a fast network (faster at train time and same or improved speed at inference time) having residual connections or skip connections, which provides significant contributions to the computer vision domain. These skip connections help to learn the identity function well and also mitigate the effect of the vanishing gradient problem effectively. The introduction of the skip connections in the ResNet architecture leads to efficient propagation of the gradients in the network architecture from the output to the input. So the advantage of using residual networks would be the effective propagation of the gradients (moving backward). One of the most prevalent problems faced in the previous ConvNet architectures such as VGG16, AlexNet, etc. was the inability to effectively propagate gradients due to the lack of identity mappings and the increasing depth of the architectures. Another advantage of the ResNet is that it is an ensemble of different networks of varying depths which definitely provides an extra edge over other networks. The presence of a batch normalization layer inside every convolutional block in the ResNet architecture also improves gradient backpropagation to a greater extent. ResNet architectures are usually very deep, which overall achieves good performance without degrading the performance of the network due to their increased depth. So we can state that a very deep network with residual connections can achieve significantly lower training error than its non-residual counterparts.

3.1.2 Visual Codebook generation and feature engineering

The first step in codebook generation is clustering. The 2048-dimensional deep features ensuing from all the images in the training set are clustered into k groups based on the similarity between the feature vectors. k-means is an unsupervised clustering algorithm (Syarif, et al. 2012) that is popularly used in the Bag-of-Visual-Words model for grouping the low-level features into distinctive clusters. Choosing an appropriate value of the number of clusters k plays a crucial role in k-means clustering. Silhouette analysis is an approach that determines the optimal number of clusters (Géron 2019). We chose a local maximum value of silhouette score in the silhouette plot as the value of k for the two benchmark datasets used in our experiments. As noted from the local maxima of silhouette score plots in Figure 3.2, the optimum value of k is determined to be 60 for the Graz-02 dataset and 100 for the TF-Flowers dataset.

For all the experiments we have used the widely adapted k-means++ algorithm to initialize the cluster centroids. The k-means++ is a cluster center initialization strategy. In comparison to the traditional approach, it leads to the smarter initialization of the centroids which automatically helps to enhance the clustering quality. In the k-means++ clustering initialization strategy, firstly the random centroids are selected from the available data points. Then the distance of all the available data points is calculated from the nearest centroid. After the first iteration phase, the next set of centroids is selected on the basis of the calculation of the distance from the farthest available data point. This procedure is repeated till the k centroids have been sampled. The choice of this popular initialization strategy is solely in favor of improving the

performance and it is known to act as a robust method for a variety of scenarios to improve k-means convergence. After clustering, the cluster centers form the bins of a histogram and are defined as the visual words in the codebook generated. The next task is to find which of these visual words are present in each image. The histogram over the visual codebook is generated next for each image in the training and the test sets as shown in Figure. 3.3. This is achieved by considering each of the 49 x 2048 deep feature vectors of the image, and considering which of the histogram bins it is closest to in terms of smallest Euclidean distance. The frequency of this bin is then increased by one. The histogram features are then normalized by scaling, and the resulting feature vectors are learned by a suitable classifier. While performing clustering using the k-means model, in our experiments, the number of instances is much larger than the dimensionality of features.

3.1.3 Scaling of histogram features

The resulting histogram features, which we now call BOVW features, are normalized using maximum absolute scaling (MaxAbsScalar) (Tax and Duin 2000), wherein we scale each feature column by dividing it by the maximum absolute value present in the respective feature column. Equation 3.1 illustrates the scaling technique, where X[i] denotes the ith feature and Xmax [i] denotes the maximum value in the ith feature column of the BOVW feature matrix. Thus, the normalized feature x[i] is confined to a range of [0, 1]. Chi-Square (Chi^2) SVM kernel, used in the next phase of experiments, is a positive semi-definite kernel that expects the input to be in a non-negative range.

$$x[i] = \frac{X[i]}{X_{max}[i]} \tag{3.1}$$

3.1.4. Chi² SVM Classifier for classification of multi-class imbalanced datasets

We have conducted extensive experiments using various classifiers in the evaluation phase to find the perfect choice of the classifier for our BOVW features. From the empirical analysis, the results of which are presented it is found that the Chi² SVM classifier results in better generalization performance and works effectively for classifying the under-represented classes in imbalanced multi-class datasets.

We discussed, in this section, some of the aspects of the SVM kernels that we investigated in our study to understand their suitability for resolving the class-imbalance problem in multi-class datasets.SVM classification for a multi-class problem is solved using the one-vs-rest approach (Cortes and Vapnik 1995). Equation 3.2 discusses the generalized function h, which shows the optimization problem associated with a kernelized SVM by applying a kernel function ϕ on the original input data x.

$$h_{\hat{w},\hat{b}}\left(\phi(x^{(n)})\right) = \hat{w}^T\phi(x^{(n)}) + \hat{b} = \left(\sum_{i=1}^{m}\hat{\alpha}^{(i)}t^{(i)}\phi(x^{(i)})\right)^T\phi(x^{(n)}) + \hat{b}$$

$$= \sum_{i=1}^{m}\hat{\alpha}^{(i)}t^{(i)}\left(\phi(x^{(i)})^T\phi(x^{(n)})\right) + \hat{b} \quad = \sum_{\substack{i=1 \\ \hat{\alpha}^{(i)}>0}}^{m}\hat{\alpha}^{(i)}t^{(i)}K(x^{(i)},x^{(n)}) + \hat{b} \tag{3.2}$$

For our proposed methodology, we have used a Chi² kernel (Bellet, et al. 2013, Bellet, et al. 2015) which is known to measure similarities of data points using a

weighted exponential method. Equation 3.3 defines the Chi² kernel function for the data points x and y with the help of gamma term γ.

$$k(x, y) = e^{-\gamma \sum_{i=0}^{n} \frac{(x[i]-y[i])^2}{x[i]+y[i]}} \tag{3.3}$$

For our experimental setup, the value of $\gamma = 1$ and the kernel is of the form $e^{-\gamma d(x,y)}$ as given in equation 3.3. The value of $d(x, y)$ which is calculated based on affinity is 0 for a pair of similar feature vectors and tends to ∞ for highly dissimilar feature pairs such that $e^{-1.0} = 1$ and $e^{-1.\infty} = 0$, respectively. Thus, the Chi² kernel when applied on histogram feature vectors produces a pairwise distance matrix in the range $[0, 1]$. The exponent of the exponential term in equation 3.3 is a non-linear squaring function that brings similar points closer and dissimilar points farther apart to the theoretical concept of distance metric learning (DML) which is a transformation of the input space (Bellet, et al. 2013, Bellet, et al. 2015). The feature transformation as a result of the kernel dot product, in the case of Chi² SVM, is able to capture the correlation between pairs of features. The weighted pairwise feature distance used in chi-square allows it to internally calculate the similarities between combinations of samples from the majority as well as minority classes. Due to which the class imbalance problem has been tackled to an extent such that the classifier is able to give almost equal results in the case of minority as well as majority classes.

Additionally, we conducted another set of experiments to support larger-scale use cases. A simple fully connected neural network is used that is trained with the

help of hinge loss instead of logistic loss (softmax) which is generally used in most artificial neural networks with terminal full-connected layers for classification problems. These neural networks are termed as Quasi-SVMs, and the work done is inspired by (Rahimi, et al. 2008) (the neural network remains completely the same only the loss function is replaced). The choice of hinge loss for training our artificial neural networks used in the proposed approach is due to hinge loss being better suited to deal with imbalanced datasets. It was motivated by the fact that SVMs, as well as Quasi-SVMs, are able to deal with different datasets better than softmax classifiers due to their property of robustness, these classifiers are able to learn features distinctively and it does help the gradients to propagate well enough making it easy to generalize on a wider range of datasets. Apart from the Quasi-SVMs being trained in linear mode, we also train them in kernelized mode. In the kernelized model, we add another RFF (Random Fourier Features) layer, as per work done by Rahimi, et al. 2008, which is used to provide a simple yet effective kernel transformation directly inside the neural network allowing us to construct a true Quasi SVM that deals with the kernel approximation internally and making our approach highly suitable for larger datasets also. This technique is basically used to kernelize the linear model using a non-linear transformation such that it approximates a kernel SVM with hinge loss providing us with the added advantage of online learning for use with our proposed approach which is generally not supported by generic SVMs.

For the experiments, the proposed mechanism was implemented as described in Algorithm 3.1. For the purpose of our experiments, we have used various network architectures pre-trained on the large-scale ImageNet dataset which is constructed using tf.keras (TensorFlow Keras) framework (Abadi, et al. 2016). The pre-trained

weights are available from the keras_applications repository hosted by Francois Chollet. The deep feature extraction phase has been accelerated by the use of the NVIDIA Tesla T4 GPU. For each of the experiments used in the Bag-of-Visual-Words based codebook generation approach the features of the different deep learning models were extracted and stored prior to running the k-Means clustering step. The initial feature extraction step involved running the deep networks like VGG, ResNet, Inception on GPU hardware and the stored features were reused across experiments. Although the computation time for the deep feature extraction step took around 20 minutes for each dataset, the actual time would vary depending upon complexity of the dataset, the size of the feature vectors, the number of visual words, and the specific acceleration hardware being used. The remaining steps of the experiment including k-Means clustering and training of the different classifiers were carried out without usage of any GPU and directly on regular CPU hardware, and the k-Means step would dominate the majority time consumed which was up to 5 hours per clustering when the maximum value of k=150 was used. The k-means clustering and SVM classification that is a part of the BOVW approach have been performed using the Scikit-Learn package (Kramer 2016) with Python 3.6 version on, a Google Compute Engine VM with 16 GB RAM and quad-core Intel Xeon CPU. The imbalanced multi-class datasets: Graz-02 and TF-Flowers are used in the experimental tasks. Each class is split into a 70:30 train test ratio using the stratified shuffle-split technique. The same is repeated over 10 folds to evaluate all the classifiers using a cross-validation-based approach. The observed metrics are aggregated over all folds and the reported results include their mean and standard deviation. For our analysis, we compare our proposed approach with VGG16, InceptionV3, and ResNet50 pre-trained networks. The VGG16 network architecture consists of 3 fully connected layers whereas the

InceptionV3 and the ResNet50 architectures use a single fully connected layer. For our experimental task, we use these network architectures primarily using pre-trained weights, except that all of the dense layers use weights randomly initialized from a Glorot uniform distribution while the biases are initially set to zeros. All the dense layers in the respective architectures have been trained in order to adapt the network to the new classification task (instead of ImageNet classification) which is the procedure followed in several deep learning experiments (Tajbakhsh, et al. 2016). In the case of the VGG16 network, we train the FC1, FC2, and a final classification output layer which provides the softmax predictions of the various classes. For the InceptionV3 and ResNet-50, we train the last dense layer in order to adapt the network to the new classification task. For all of these experiments, we train the network for a total of 20 epochs to minimize the categorical cross-entropy loss using the Adam optimizer, with a learning rate of 0.01. Similar to our other experiments, we train these models over 10 stratified folds. Additionally, with the help of a few experiments, we demonstrate another variant of our proposed Bag-of-Visual-Words approach. Our proposed approach with minor modifications and the learned BOVW features can be made suitable for large-scale classification tasks as well. We make use of neural networks that work like Quasi SVMs with and without kernel approximation. Kernel approximations have been done with the help of Random Fourier Features. The following experimental settings have been used while conducting the experiments for Quasi SVM (which approximates a Linear SVM) and Kernelized Quasi SVM (which approximates a kernel SVM). We make use of the Adam optimizer combined with hinge loss. We noticed that the choice of hinge loss function significantly improved the convergence of our classifiers to a noticeable extent. We apply the L2 regularization penalty of 1e-4 on all the trainable network weights and biases.

Initially, we set the learning rate as 5e-4 which is inverse time decayed by a factor of 1e-5 and all the experiments are carried out till 1000 epochs. We have also analyzed the role of adding various hidden layers to these neural networks (which are used to approximate large-scale support vector machines). For the experiments, input layers consist of 60 or 100 nodes (based upon the number of clusters and the dataset) and the output layers are adjusted based on the number of classes in the target dataset. Further, the addition of hidden (fully connected) layers leads to the formation of complex neural networks but it's important to analyze the effect in each scenario to observe how the addition of hidden layers affects working in large-scale settings. We added three hidden layers to the Quasi SVM i.e. with 64, 128, and 256 hidden units respectively. In the case of Kernelized Quasi-SVM, Random Fourier Features are used (as the kernel approximation step) with 1024 feature dimensions. Also, we consider adding two dense layers, one with 256 and another with 512 hidden nodes. In the case of a larger scale setting it is recommended to use mini-batch k-means (Sculley 2010) instead of the standard k-means implementation used earlier in our experiments. This would ensure that the clustering for codebook generation can be conducted easily on more samples at lesser computational complexity.

In Table 3.1, the comparison has been done using different deep features extracted using pre-trained models (VGG16, Inception-V3, ResNet-50) along with the traditional SIFT descriptor, for the BOVW codebook generation, in combination with the different variants of SVM classifier (Linear, RBF, Chi^2 SVM using scalar and without using scalar approach). From the analysis, it was found that the deep features extracted using the ResNet-50 pre-trained model in combination with the Chi^2 SVM classifier work best for our BOVW model, since it shows high accuracies

for all the four classes present in the Graz-02 dataset. Another observation that was inferred from the result analysis is that scaling operation plays a significant role because avoiding scaling would have an adverse impact on the overall performance of the classifier as some minority classes were not detected. The same pattern is observed in a few approaches such as in SIFT + BOVW and ResNet-50 + BOVW as shown in Table 3.1. A third observation that was inferred from the analysis is that Chi^2 SVM exhibits an overall better performance since it is able to identify all the classes separately and more efficiently in comparison to other variants of SVM (linear and RBF SVM).

Further, experiments were conducted for the Graz-02 dataset in Table 3.3 and Table 3.5 to validate the proposed combination of the ResNet-50 + BOVW approach with the Chi^2 SVM classifier. The analysis drawn from Table 3.3 proves that the proposed combination for BOVW works well in comparison to direct deep learning using state-of-the-art pre-trained approaches (VGG-16, Inception-V3, and ResNet-50). Five different supervised classifiers are evaluated and analyzed in Table 3.5 for our BOVW features. A comparative experimental analysis of various popularly used classifiers such as Logistic Regression, Linear Discriminative Analysis, K-Nearest Neighbors, Decision Tree, and Gaussian Naïve Bayes classifiers was conducted with the Chi^2 SVM classifier, for the proposed approach. From the classification scores in Table 3.5, it is noted that the other classifiers have not performed sufficiently well in comparison to the Chi^2 SVM classifier. Therefore, Chi^2 SVM proves to be a better choice of the classifier in comparison to other classifiers, for the imbalanced multi-class dataset.

The experiments are done, on similar lines, in the case of the TF-Flowers dataset. It was also validated from the result analysis of the TF-Flowers dataset that the proposed Bag-of-visual-words model constructed with deep features and integrated with the Chi² SVM classifier works well in comparison to all other combinations, as shown in Table 3.2. It is noted that the deep features work well in comparison to the handcrafted SIFT features, for the BOVW model. Further observations from Table 3.2 led us to the conclusion that ResNet-50 features integrated with the BOVW approach prove to be the best amongst the other features extracted using different pre-trained networks, and the Chi² SVM classifier emerges as the optimal choice of the classifier in this combination. Comparisons to other variants of SVM such as RBF and Linear SVMs prove the efficacy of our learning model, as observed in Table 3.2. The comparison of the proposed combination of the BOVW approach with the Chi² SVM classifier and the state-of-the-art pre-trained networks (VGG16, Inception-V3, and ResNet-50) is shown in Table 3.4, and from the analysis, it was found that the proposed BOVW approach works well in comparison to the other state-of-the-art pre-trained networks. To validate our approach, we have shown a comparative analysis of various supervised machine learning classifiers combined with deep features for the BOVW model. Hence, we have also presented a comparative analysis of the performance of various machine learning classifiers for classifying the ResNet-50-based BOVW features. It is noted that the Chi² SVM classifier works well in comparison to the other five supervised classifiers as per the classification scores in Table 3.6.

Macro Average ROC AUC readings for different features for the Graz-02 and TF-Flowers datasets are presented in Tables 3.7 and 3.8. Along with that in Tables

3.9 and 3.10, Macro Average ROC AUC readings for different classifiers and ResNet-50 deep features for the Graz-02 dataset and TF-Flowers dataset are displayed respectively. Figure 3.4. depicts the ROC curves for different features in comparison with the Chi^2 SVM classifier in the case of the Graz-02 and TF-Flowers datasets respectively and the ROC curve depicting different classifiers in combination with ResNet−50 deep features in the case of Graz-02 and TF-Flowers datasets respectively. We have also applied various resampling techniques which are known to be baseline methods that are applied to imbalanced distributions to balance out the number of samples in each class (except the majority class). These strategies involve either oversampling or under-sampling approaches, such as ADASYN, SMOTE, random oversampling, and random under-sampling (Susan and Kumar 2019). The aim was to investigate the effect of these balancing tools, popular in data mining, in our classification scheme. These resampling strategies have been applied directly at the feature level before the first phase of our deep feature-based BOVW scheme. We did not observe any significant improvement in the classification results even by applying the well-known sampling techniques in the proposed combination, as evident from the scores in Tables 3.11. and 3.12. for the two datasets respectively. In the case of deep learning-based approaches, we popularly use affine transformations to randomly augment the data samples. It significantly increases the total number of samples used to train the deep learning network and is known to improve classification performance on the smaller dataset through regularization. For our proposed approach, we have used the ResNet-50 network to extract deep features, which are in turn pre-trained on the million-scale ImageNet dataset. Thus, applying random data augmentation at the image level before the feature extraction step would not effectively improve the clustering step in the Bag-of-Visual-Words model.

From the different results of the various experiments in Tables 3.13. and 3.14., it was found that both Quasi SVM and Kernelized Quasi SVM were able to achieve acceptable classification performance on our studied datasets. The addition of hidden (fully connected) layers in these combinations affected the performance of the classifier to some extent but their role is quite stochastic in nature. So, it is recommended that the use of this alternate approach (large-scale kernel machines) on larger datasets would require us to perform some hyperparameter tuning with respect to the target dataset.

From our extensive experiments and analysis, it is summarized that the deep features extracted using ResNet-50 pre-trained CNN when integrated with the BOVW approach, with the combination of the Chi^2 SVM classifier, proved to be optimal for the image classification task when dealing with multi-class imbalanced datasets. The same pattern is observed for both the benchmark datasets: Graz-02 and TF-Flowers, as validated by the experimental results on the basis of various evaluation measures: ROC curve, F1-score, and accuracy. The proposed learning model is able to respond to and detect all the classes (Majority and Minority classes) equally and effectively. The application of resampling strategies had little or no effect on the learning model which makes it a distinctive approach. In the case of the TF-Flowers dataset, there are 3670 images, out of which 70% images are taken for training i.e. 2569 image samples. From each image present in the dataset, ResNet-50 features are extracted. The dimension of features extracted from the training set is (2569, 7, 7, 2048) which is converted to the 2-dimensional matrix of size (2569 * 7 * 7, 2048) as in (125881, 2048). The 2-dimensional matrix X was passed to the k-means algorithm. So, in this particular case of the TF-Flowers dataset, N is 12588 (number of instances) and D is

equal to 2048 (number of features) as $X \in \mathbb{R}^{N \times D}$ (matrix X has N x D dimension). Similarly for the Graz-02 dataset, out of 1,476 images, 70% of images are taken i.e. the 1033 image samples for the training set. So (1, 60) features are obtained and subsequently passed to the SVM classifier. Therefore, it can be inferred that even after applying k-means for BOVW codebook generation, each image yields k number of features only (k=100 for TF-Flowers dataset, k=60 for Graz02 dataset). Therefore, we did not experience any issues related to the curse of dimensionality problem as N is sufficiently greater than D (N >> D) in the proposed BOVW approach.

The experiments conducted for deep feature based BOVW classification were performed using 10-fold cross validation. Each fold chosen for the purpose of the experiment was randomly chosen yet prepared such that each fold makes use of stratified sampling mechanism which ensures that sampling error can be reduced to a good extent and the samples are representative of the distribution in the respective original datasets. When stratified folds and cross validation are used together, it helps to produce more reliable and accurate statistical inferences (Szeghalmy and Fazekas 2023). It helps to ensure that the model is representative of the overall sample space which is sufficiently large in case of TF-Flowers and Graz-02 datasets, that allows the performance measures represented in our work to be statistically significant even for slight deviations. The average of the results from stratified folds and cross validation can be used to get a more accurate estimate of the model's performance, which is what we have used in our study. This is because the average of the results will be less affected by any one fold that may be biased, which hence indicates that slight differences in model performance between different approaches are representative of marginal change. As all results depicted are calculated using 10-fold cross validation,

it does reduce the variance of the results making them more reliable. Therefore, any minor changes or updates to the model are likely to have a significant impact on the overall performance as the model is being trained on a different subset of the data each time providing overall statistically significant inferences when averaged.

In this work, we have done experimental analysis on imbalanced multi-class datasets such as Graz-02 and TF-Flowers, to validate the proposed BOVW approach with Chi² SVM using ResNet-50 deep features for visual codebook generation. The choice of ResNet-50 (ImageNet pre-trained model) deep features over traditional handcrafted features helps improve the overall classification performance of BOVW. The role of feature scaling is also highlighted which significantly improves classification of the histogram representation. The role of Chi² kernel transformation has been analyzed to be an effective similarity metric for classifying histogram-based features using one-vs-rest Support Vector Machines. The proposed model has been evaluated in comparison to various state-of-the-art techniques using various evaluation measures like F1-score, accuracy, ROC curve, and its AUC. All the experiments have been performed on a total of 10 folds (cross-validation strategy) for both datasets. The analysis of experimental results depicts that the presented BOVW approach using ResNet-50 deep features and Chi² SVM is able to surpass the baseline methods while dealing with imbalanced datasets. Additionally, we have shown results with an alternate approach suitable for large-scale settings with the help of Quasi SVMs constructed with the help of neural networks. This is especially useful to vouch for the scalability of our BOVW-based approach on data expensive scenarios as well. An added advantage of using this alternate approach for kernel SVM approximation

would be the support for online learning which could be highly desirable for many applications.

Table 3.1. Performance evaluation of different low-level features for BOVW codebook generation and choice of classifier for Graz -02 dataset (Best performance highlighted by gray cells).

	Classifier	None (F1)	Person (F1)	Car (F1)	Bike (F1)	Accuracy
SIFT + BOVW	Chi² SVM	0.530 ± 0.0218	0.4414 ± 0.0270	0.5331 ±0.0246	0.7115 ± 0.0174	0.5616 ±0.0123
	Chi² SVM (without scalar)	0.0 ± 0.0	0.0 ± 0.0	0.4447± 0.0012	0.0423 ± 0.0270	0.2898 ±0.0035
	Linear SVM	0.4403 ± 0.0328	0.4150 ± 0.0328	0.4774 ± 0.0317	0.6906 ± 0.0141	0.5076 ± 0.0184
	RBF SVM	0.0234 ± 0.0185	0.0 ± 0.0	0.4848 ± 0.0076	0.5432 ± 0.0344	0.3844 ± 0.0110
VGG16 + BOVW	Chi² SVM	0.8303 ± 0.0212	0.9187 ± 0.0131	0.8926 ± 0.0146	0.9407 ± 0.0096	0.8918 ± 0.0119
	Chi² SVM (without scalar)	0.069 ± 0.0085	0.0021 ± 0.0131	0.4451 ± 0.0012	0.0423 ± 0.0270	0.2909 ± 0.0037
	Linear SVM	0.7959 ± 0.0188	0.9061 ± 0.0187	0.8774 ± 0.0136	0.9416 ± 0.093	0.8742 ± 0.0118
	RBF SVM	0.6851 ± 0.0179	0.8050 ± 0.0212	0.8234 ± 0.0105	0.8203 ± 0.0249	0.7717 ± 0.0130
Inception-V3 + BOVW	Chi² SVM	0.7992 ± 0.0208	0.8848 ± 0.0399	0.8781 ± 0.0225	0.9360 ± 0.0568	0.8702 ± 0.0138
	Chi² SVM (without scalar)	0.0156 ± 0.0480	0.03291 ± 0.0321	0.4693 ± 0.0074	0.3372 ± 0.0568	0.3584 ± 0.0145
	Linear SVM	0.7839 ± 0.0181	0.875 ± 0.0245	0.8692 ± 0.0189	0.3372 ± 0.0568	0.8586 ± 0.0116
	RBF SVM	0.6988 ± 0.0259	0.7324 ± 0.0535	0.8563 ± 0.0171	0.8730 ± 0.0254	0.7857 ± 0.0219
ResNet-50 + BOVW	Chi² SVM	**0.8509 ± 0.142**	**0.9405 ± 0.0141**	**0.9013 ± 0.0184**	**0.9609 ± 0.117**	**0.9097 ± 0.0106**
	Chi² SVM (without scalar)	0.0 ± 0.0	0.0 ± 0.0	0.4461 ± 0.0012	0.0731 ± 0.0268	0.2939 ± 0.0036
	Linear SVM	0.8399 ± 0.0184	0.9318 ± 0.0158	0.8969 ± 0.0139	0.9592 ± 0.108	0.9029 ± 0.0116
	RBF SVM	0.7263 ± 0.0185	0.8347 ±0.0258	0.8381 ± 0.0268	0.8935 ± 0.2069	0.8124 ± 0.0155

Table 3.2. Performance evaluation of different low-level features for BOVW codebook generation and choice of classifier for TF-Flowers dataset (Best performance highlighted by gray cells).

		Dandelion (F1)	Daisy (F1)	Sunflowers (F1)	Roses (F1)	Tulips (F1)	Accuracy

SIFT + BOVW	**Chi² SVM**	0.6571± 0.0244	04871 ±0.0375	0.5891± 0.0262	0.3737± 0.0314	0.4841 ± 0.0239	0.5364± 0.0192
	Chi² SVM (without scalar)	0.3930± 0.0002	0.0 ± 0.0	0.0075± 0.056	0.0 ± 0.0	0.0 ± 0.0	0.2450± 0.0005
	Linear SVM	0.6962± 0.0111	0.5092± 0.0195	0.6169± 0.0195	0.4188± 0.0222	0.4567 ± 0.0123	0.5405± 0.0082
	RBF SVM	0.4625± 0.085	0.1615± 0.0243	0.4765± 0.0271	0.0378± 0.0140	0.3218 ± 0.0230	0.3722± 0.0108
VGG 16 + BOVW	**Chi² SVM**	0.8961± 0.0115	0.8610± 0.02	0.8511± 0.00092	0.8114± 0.0211	0.8211 ± 0.0117	0.8503± 0.0087
	Chi² SVM (without scalar)	0.3931± 0.0001	0.0041± 0.051	0.0075± 0.0056	0.0010± 0.031	0.0008 ± 0.002	0.2455± 0.0004
	Linear SVM	0.8757± 0.0185	0.8380± 0.0248	0.8144± 0.0202	0.7786± 0.0215	0.7760 ± 0.0164	0.8190± 0.0150
	RBF SVM	0.7853± 0.0166	0.6622± 0.0278	0.6202± 0.0435	0.2573± 0.0823	0.6012 ± 0.0188	0.6292± 0.0196
Inception-V3 + BOVW	**Chi² SVM**	0.8651± 0.0122	0.8414± 0.0174	0.7477± 0.0220	0.7162 ± 0.0283	0.7120 ± 0.0205	0.7836± 0.0124
	Chi² SVM (without scalar)	0.4185± 0.0044	0.3135± 0.0606	0.1014± 0.0247	0.0636± 0.0165	0.1476 ± 0.0447	0.3103± 0.0121
	Linear SVM	0.8585± 0.0115	0.8392± 0.0146	0.7327± 0.0140	0.6996± 0.0341	0.7246 ± 0.0212	0.7729± 0.0151
	RBF SVM	0.7932± 0.0209	0.8064± 0.0204	0.6102± 0.0435	0.2141± 0.1519	0.6269 ± 0.0229	0.6552± 0.0226
ResNet50 + BOVW	**Chi² SVM**	**0.9162± 0.0141**	**0.9159± 0.0136**	**0.8883± 0.0133**	**0.8335± 0.0141**	**0.8492 ± 0.0147**	**0.8816± 0.0084**
	Chi² SVM (without scalar)	0.3931± 0.0003	0.004± 0.0051	0.0075 ± 0.0056	0.0041± 0.0050	0.8008 ± 0.0241	0.2457± 0.0004
	Linear SVM	0.9070± 0.0115	0.9059± 0.171	0.8648± 0.0177	0.8054± 0.0135	0.8211 ± 0.0101	0.8619± 0.0053
	RBF SVM	0.8206± 0.0271	0.7425± 0.0249	0.7934± 0.0355	0.4089± 0.0901	0.6700 ± 0.0275	0.7127± 0.0190

Table 3.3. Performance evaluation of various state-of-the-art pre-trained networks for the Graz-02 dataset.

Pre-trained Networks	**None (F1)**	**Person (F1)**	**Car (F1)**	**Bike(F1)**	**Accuracy**
VGG16	0.788 ± 0.0313	0.9046 ± 0.0244	0.8585 ± 0.0215	0.9367 ± 0.0262	0.8693 ± 0.0167
Inception-V3	0.4301 ± 0.0574	0.8338 ± 0.165	0.7711 ± 0.0301	0.8946 ± 0.0453	0.7568 ± 0.0109
ResNet-50	0.670 ± 0.1136	0.8981 ± 0.0228	0.8350 ± 0.0302	0.9471 ± 0.0186	0.8460 ± 0.0302

Table 3.4. Performance evaluation of various state-of-the-art pre-trained networks for TF-Flowers dataset.

Pre-trained networks	**Dandelion (F1)**	**Daisy (F1)**	**Sunflowers (F1)**	**Roses (F1)**	**Tulips (F1)**	**Accuracy**
VGG16	0.9108± 0.0149	0.8761± 0.0236	0.8752± 0.0299	0.8343± 0.0245	0.8395± 0.0254	0.8689± 0.0196
Inception -V3	0.7413± 0.0378	0.6331± 0.250	0.5199± 0.0179	0.6201± 0.0629	0.7651± 0.0156	0.6640± 0.0190
ResNet-50	0.8901± 0.210	0.8394± 0.0369	0.8509± 0.0190	0.6504± 0.0797	0.8031± 0.0203	0.8203± 0.0176

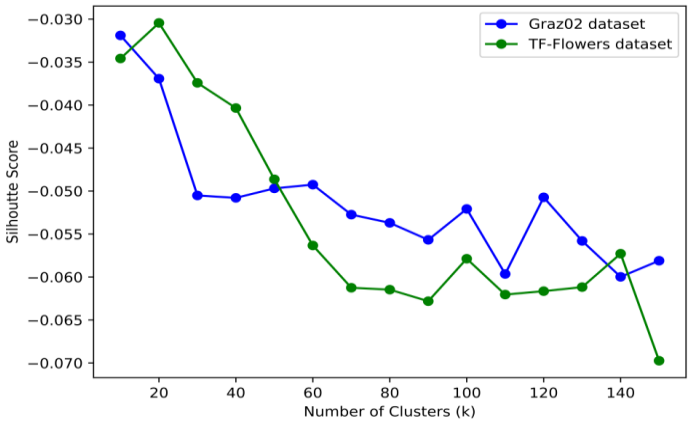


Figure 3.2. Silhouette score to find the optimum value of k (number of clusters in BoVW k-Means) for Graz-02 and TF-Flowers dataset.

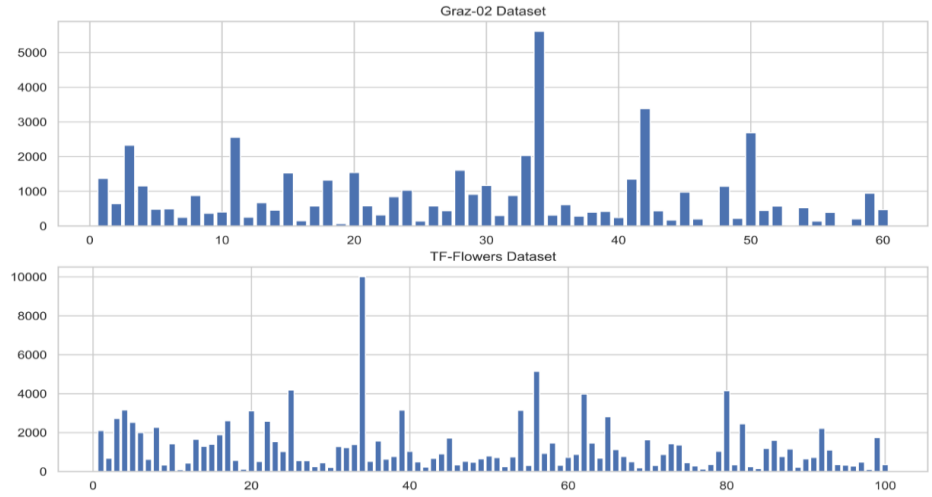


Figure 3.3. Histogram representation of proposed ResNet-50 based BoVW features for Graz-02 and TF-Flowers dataset.

Table 3.5. Performance evaluation of various supervised classifiers on our BOVW features (ResNet-50 + BOVW approach) for the Graz-02 dataset.

Classifiers	None (F1)	Person (F1)	Car (F1)	Bike(F1)	Accuracy
Logistic Regression	0.8294 ± 0.0177	0.9330 ± 0.0164	0.8938 ± 0.0165	0.9532 ± 0.0127	0.8979 ± 0.0133
Linear Discriminative Analysis	0.8245 ± 0.0169	0.9224 ± 0.0196	0.8933 ± 0.0174	0.9503 ± 0.0116	0.8925 ± 0.0124
K-Nearest Neighbors Classifier	0.7803 ± 0.0154	0.8953 ± 0.0160	0.8452 ± 0.0143	0.9324 ± 0.0137	0.8580 ± 0.0079
Decision Tree	0.7852 ± 0.0213	0.8911 ± 0.0153	0.8609 ± 0.0152	0.9167 ± 0.0200	0.8623 ± 0.0079
Gaussian Naïve Bayes	0.3763 ± 0.2451	0.9264 ± 0.0138	07720 ± 0.0630	0.9112 ± 0.0365	0.7785 ± 0.0594

Table 3.6. Performance evaluation of various supervised classifiers on our BOVW features (ResNet-50 + BOVW approach) for the TF-Flowers dataset.

Classifiers	Dandelion (F1)	Daisy (F1)	Sunflowers (F1)	Roses (F1)	Tulips (F1)	Accuracy
Logistic Regression	0.9052 ± 0.0096	0.9027 ± 0.0160	0.8618 ± 0.0100	0.8055 ± 0.0197	0.8288 ± 0.0120	0.8621 ± 0.0044
Linear Discriminative Analysis	0.8603 ± 0.082	0.9054 ± 0.0131	0.8628 ± 0.0156	0.8171 ± 0.0176	0.8171 ± 0.0129	0.8603 ± 0.0082

K-Nearest Neighbors Classifier	0.8707 ± 0.0190	0.8537 ± 0.0241	0.8179 ± 0.0139	0.7714 ± 0.0291	0.7714 ± 0.0217	0.8153 ± 0.0135
Decision Tree	0.8274 ± 0.0115	0.8208 ± 0.0252	0.7564 ± 0.0162	0.7259 ± 0.224	0.7522 ± 0.0230	0.7791 ± 0.0082
Gaussian Naïve Bayes	0.8619 ± 0.0277	0.884 ± 0.0141	0.8105 ± 0.0328	0.7447 ± 0.505	0.7960 ± 0.0207	0.8214 ± 0.0187

Table 3.7. Macro Average ROC AUC readings for different features for the Graz-02 dataset and TF-Flowers dataset.

Features	**Classifier**	**Graz-02 Dataset**	**TF-Flowers Dataset**
ResNet-50 BOVW Features	**Chi² SVM**	**0.9837**	**0.9824**
	Logistic Regression	0.9828	0.9806
	Linear Discriminative Analysis	0.9822	0.9788
	K Nearest Neighbors Classifier	0.9549	0.9482
	Decision Tree	0.9166	0.8796
	Gaussian Naïve Bayes	0.9552	0.9517

Table 3.8. Macro Average ROC AUC readings for different features for Chi² SVM classifier for Graz-02 dataset and TF-Flowers dataset.

Features	**Classifier**	**Macro Average ROC AUC**	
		Graz-02 Dataset	**TF-Flowers Dataset**
ResNet-50 BOVW Features	**Chi² SVM**	**0.9837**	**0.9824**
Inception-V3 BOVW features		0.9651	0.9457
VGG16 BOVW Features		0.9758	0.9705
SIFT BOVW Features		0.8017	0.8275

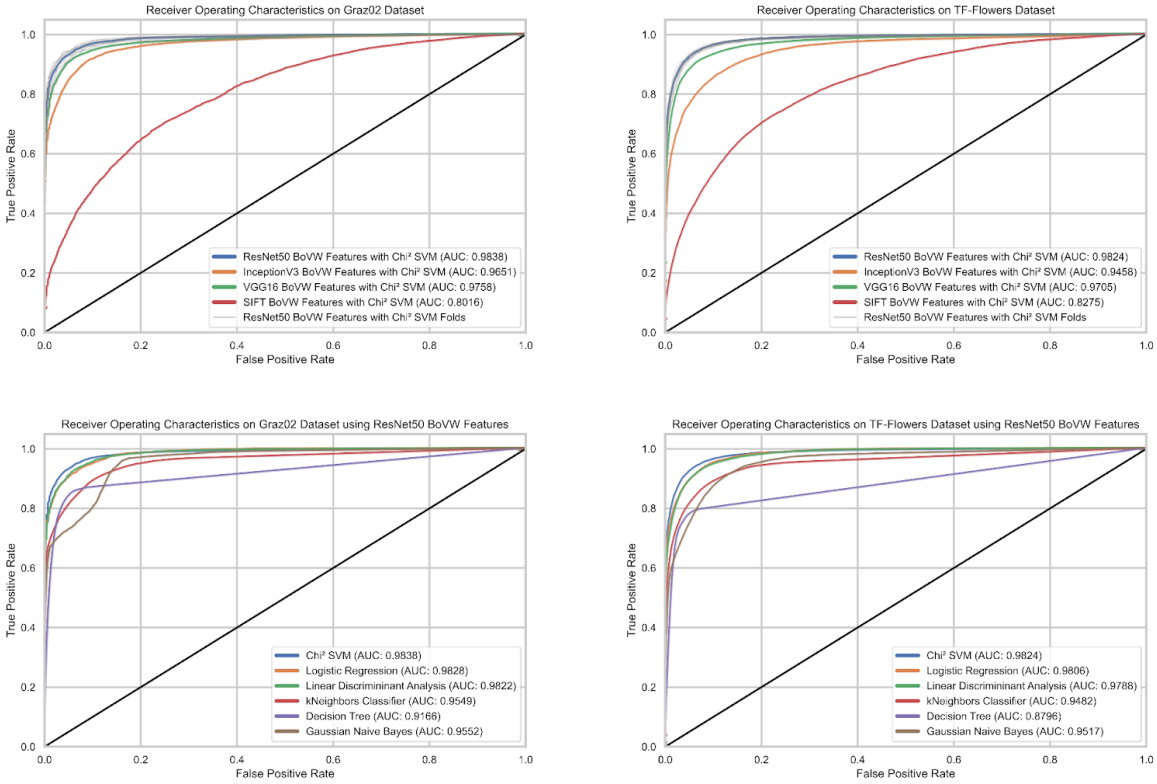


Figure 3.4. (Top row left to right) ROC curves depicting different features in comparison with Chi² SVM classifier in the case of Graz-02 and TF-Flowers dataset respectively. (Bottom row left to right) ROC curve depicting different classifiers in combination with ResNet−50 deep features in case of Graz-02 and TF-Flowers dataset respectively.

Table 3.9 Macro Average ROC AUC readings for different classifiers and ResNet-50 deep features for the Graz-02 dataset and TF-Flowers dataset.

Features	**Classifier**	**Macro Average ROC AUC**	
		Graz-02 Dataset	**TF-Flowers Dataset**
ResNet-50 BOVW Features	**Chi² SVM**	**0.9837**	**0.9824**
	Logistic Regression	0.9828	0.9806
	Linear Discriminative Analysis	0.9822	0.9788
	K Nearest Neighbors Classifier	0.9549	0.9482
	Decision Tree	0.9166	0.8796
	Gaussian Naïve Bayes	0.9552	0.9517

Table 3.10. Macro Average ROC AUC readings for different features for Chi² SVM classifier for Graz-02 dataset and TF-Flowers dataset.

Features		**Macro Average ROC AUC**

	Classifier	**Graz-02 Dataset**	**TF-Flowers Dataset**
ResNet-50 BOVW Features		**0.9837**	**0.9824**
Inception-V3 BOVW features		0.9651	0.9457
VGG16 BOVW Features		0.9758	0.9705
SIFT BOVW Features	**Chi² SVM**	0.8017	0.8275

Table 3.11. Performance evaluation of our BOVW model with different sampling techniques applied at the feature level for the Graz-02 dataset.

Sampling Technique	**None (F1)**	**Person (F1)**	**Car (F1)**	**Bike (F1)**	**Accuracy**
ADASYN	0.8478 ± 0.0128	0.9315 ± 0.0157	0.9003 ± 0.0192	0.9609 ± 0.0117	0.9069 ± 0.0094
SMOTE	0.8528 ± 0.0128	0.9389 ± 0.0142	0.9020 ± 0.0183	0.9623 ± 0.0106	0.9103 ± 0.0100
Random Over sampling	0.8508 ± 0.0130	0.9390 ± 0.0129	0.9002 ± 0.0177	0.9624 ± 0.0116	0.9094 ± 0.0100
Random Under sampling	0.8495 ± 0.0137	0.9405 ± 0.0141	0.8985 ± 0.0212	0.9599 ± 0.0137	0.9081 ± 0.0126
None (proposed Approach)	0.8509 ± 0.0142	0.9405 ± 0.0141	0.9013 ± 0.0184	0.9609 ± 0.0117	0.9097 ± 0.0126

Table 3.12. Performance evaluation of our BOVW model with different sampling techniques applied at the feature level for TF-Flowers datasets.

Sampling Technique	**Dandelion (F1)**	**Daisy (F1)**	**Sunflowers (F1)**	**Roses (F1)**	**Tulips (F1)**	**Accuracy**
ADASYN	0.9164 ± 0.0117	0.9190 ± 0.0117	0.8900 ± 0.0130	0.8339 ± 0.0151	0.8502 ± 0.0141	0.8829 ± 0.0073
SMOTE	0.9190 ± 0.0127	0.9158 ± 0.0118	0.8892 ± 0.0120	0.8363 ± 0.0105	0.8502 ± 0.0125	0.8831 ± 0.0070
Random Over sampling	0.9165 ± 0.0145	0.149 ± 0.0136	0.8859 ± 0.0127	0.8363± 0.0129	0.8509 ± 0.0141	0.8800 ± 0.0064
Random Under sampling	0.9174 ±0.0128	0.9170 ± 0.0137	0.8886 ± 0.0134	0.8324 ± 0.0136	0.8405 ± 0.0150	0.8800 ± 0.0064
None (proposed Approach)	0.9162 ± 0.0141	0.9159 ± 0.0136	0.8883 ± 0.0133	0.8335 ± 0.0141	0.82 ± 0.0147	0.8816 ± 0.0084

Table 3.13. Performance evaluation of ResNet-50 BOVW features combined with Quasi SVM and its variants for the Graz-02 dataset.

Classifier	None (F1)	Person (F1)	Car (F1)	Bike (F1)	Accuracy
Quasi Linear SVM	0.8186 ± 0.0166	0.9234 ± 0.0167	0.8833 ± 0.0144	0.9497 ± 0.0152	0.8826 ± 0.0130
Quasi Linear SVM (w/ 1 hidden layer)	0.8490 ± 0.0116	0.9322 ± 0.0157	0.8958 ± 0.0110	0.9011 ± 0.0080	0.9063 ± 0.085
Quasi Linear SVM (w/ 2 hidden layers)	0.8474 ± 0.0164	0.9331 ± 0.0139	0.8985 ± 0.0178	0.9617 ± 0.0077	0.9072 ± 0.0105
Quasi Linear SVM (w/ 3 hidden layers)	0.8506 ± 0.0188	0.9298 ± 0.0178	0.8966 ± 0.0149	0.96630 ± 0.012	0.9081 ± 0.0125
Quasi Kernelized SVM	0.8383 ± 0.0148	0.9338 ± 0.0164	0.8911 ± 0.0147	0.95966 ± 0.085	0.9018 ± 0.0112
Quasi Kernelized SVM (w/ 1 hidden layer)	0.8382 ± 0.0174	0.9318 ± 0.0147	0.8903 ± 0.0167	0.9588 ± 0.0080	0.9090 ± 0.0109
Quasi Kernelized SVM (w/ 2 hidden layers)	0.8393 ± 0.0175	0.9241 ± 0.0219	0.8905 ± 0.0203	0.9606 ± 0.0096	0.9004 ± 0.0123

Table 3.14. Performance evaluation of ResNet-50 BOVW features combined with Quasi SVM and its variants for TF-Flowers dataset.

Classifier	Dandelion (F1)	Daisy (F1)	Sunflowers (F1)	Roses (F1)	Tulips (F1)	Accuracy
Quasi Linear SVM	0.8942 ± 0.0160	0.8880 ± 0.0204	0.8491 ± 0.0205	0.7882 ± 0.0177	0.8047 ± 0.0163	0.8459 ± 0.0106
Quasi Linear SVM (w/ 1 hidden layer)	0.9075 ± 0.0124	0.9239 ± 0.0122	0.8661 ± 0.0159	0.8095 ± 0.0161	0.8251 ± 0.0153	0.8643 ± 0.0051
Quasi Linear SVM (w/ 2 hidden layers)	0.7957 ± 0.2022	0.6910 ± 0.3504	0.6746 ± 0.3394	0.4555 ± 0.3735	0.6103 ± 0.3089	0.6958 ± 0.2321
Quasi Linear SVM (w/ 3 hidden layers)	0.9006 ± 0.0162	0.9028 ± 0.0132	0.8678 ± 0.0143	0.8065 ± 0.0223	0.8290 ± 0.0095	0.8630 ± 0.0061
Quasi Kernelized SVM	0.9022 ± 0.0119	0.9044 ± 0.0135	0.8646 ± 0.0173	0.8034 ± 0.0210	0.8206 ±0.0108	0.8603 ± 0.0062
Quasi Kernelized SVM (w/ 1 hidden layer)	0.9075 ± 0.0124	0.9074 ± 0.0120	0.8661 ± 0.0159	0.8095 ± 0.0161	0.8095 ± 0.0161	0.8643 ± 0.0051
Quasi Kernelized SVM (w/ 2 hidden layers)	0.7957 ± 0.2022	0.6910 ± 0.3504	0.6746 ± 0.3394	0.4555 ± 0.3735	0.6103 ± 0.3089	0.6958 ± 0.2321

3.2. Objective 2 and Objective 5: Implementation of pre-trained deep neural networks for image classification using imbalanced datasets and application to the class imbalance problem in object detection using deep learning

Most of the state-of-the-art techniques in the field of computer vision that are widely impacted by the advent of deep learning include vision tasks such as classification, localization, and segmentation. With the high availability of computational power (in the form of Graphical Processing Units (GPUs), large compute clusters), deep Convolutional Neural Networks (CNN) are now conveniently used for computer vision applications. Broadly categorizing, there are two variations of CNN: (a) trained from scratch and (b) pre-trained models. The CNNs are able to combine automatic feature extraction along with a discriminative classifier in one stage, which makes them different from traditional machine learning techniques. Due to time restraint or computational constraints, it's not always possible to create a model from scratch, which is why pre-trained models can be used as a benchmark to either improve the existing model or to test your own model against it. The neural network models are trained using data and further gained knowledge and then stored weights of the network can be used further. These extracted weights which are saved can be further used to transfer knowledge to other neural networks. These pre-trained networks are another variant of CNN, that is trained on a larger standard dataset and further knowledge is transformed from one domain to another domain which makes it quite useful in various other sectors (Tan, et al. 2018). The extracted features using the pre-trained models have certain benefits over the handcrafted features as seen in

literature work. The pre-trained models are the ones which are trained on large-scale datasets such as ImageNet, PASCAL VOC, MS COCO, etc. Using a pre-trained model will overall reduce the training time required to train the model and also leads to the extraction of the relevant features efficiently. There are several popular well-known pre-trained models such as Alex-Net, VGG16, VGG19, ResNet50, InceptionV3, DenseNet169, GoogLeNet, EfficientNet, InceptionResNetV2, and Xception, etc., which are widely available. The main limitation of convolutional neural network models is that it might take a few days while training on larger datasets. This limitation can be overcome by re-using the trained model weights from the existing pre-trained models that are obtained after training from the standard large-scale benchmark datasets. The weights of the models can be downloaded and used further for training the new network architecture for another set of computer vision problems.

In this chapter we have applied numerous pre-trained deep neural networks on varied size imbalanced datasets for classification, segmentation and object detection tasks by considering a single application in the biomedical domain which is diabetic retinopathy screening, and finding the best deep learning model for these datasets. As mostly observed in literature, experiments related to a single task are provided, instead of combining the mutiple tasks of classification, segmentation and object detection in a single framework.

The experiments were conducted to do a comparative analysis with various standard state-of-the-art pre-trained networks by using various imbalanced datasets (Saini and Susan 2022b). Transfer learning is a way of transferring knowledge from a specific domain or either task to another task (Tan, et al. 2018). There are multiple

advantages of using the transfer learning approach (1) Firstly, the model is already trained on a large-scale dataset using the existing model along with predefined weights for our own classification task, which saves a lot of computational processing time. (2) Secondly, we can transfer the knowledge from the large-scale dataset and perform classification well even with a small dataset. Pre-trained networks are trained on a large-scale dataset such as ImageNet, which is composed of millions of high-resolution images belonging to multiple categories or classes (approximately 1.4 million images and 1000 classes).

Three biomedical domain datasets are used to conduct experiments: (i) Kaggle DRD Dataset, (ii) DDR Dataset, and (iii) Indian Diabetic Retinopathy Image (IDRiD) Dataset. We have performed classification, object detection, and segmentation tasks on the above-mentioned three available diabetic retinopathy datasets as shown in Figure 3.5 and Table 3.15, and further emphasized the role of using a transfer learning approach for a biomedical dataset using pre-trained networks. A dataset of varying sizes was used while conducting the experiments ranging from small, medium, and larger image sample sizes. In this work, we have done a comparative analysis between different state-of-the-art pre-trained networks by using different performance evaluation metrics. For the classification task, various performance evaluation parameters are considered such as Cohen's Kappa (unweighted, linear, quadratic weighted), Accuracy, ROC-AUC (weighted and macro average), F1-score, Index Balanced Accuracy (IBA), and Geometric Mean (GMean). In the case of object detection, we have considered the following evaluation parameters for measuring the performance of the object detection models: Mean Average Precision (mAP), mAP @ 0.5IoU, mAP @ 0.75IoU, mAP (small, medium, large) and Average Recall (AR):

AR @ 1,10,100. Further, Intersection over Union (IoU) and Dice Score evaluation metrics are used for segmentation. We have performed image classification on three varied size diabetic retinopathy image datasets to detect the degree of severity of diabetic retinopathy for the retina image of a given patient. Various pre-trained deep learning CNN architectures have been applied on three diabetic retinopathy datasets:- Kaggle DRD, IDRiD and DDR, in order to perform classification: VGG16, VGG19, ResNet50, ResNet101, ResNet152, Inception-V3, ResNet50 v2, ResNet101 v2, ResNet152 v2, Xception, InceptionResNet v2, MobileNet v2, DenseNet169, DenseNet201 and EfficientNetB0 (Simonyan and Zisserman 2014, He, et al. 2016, Szegedy, et al. 2016, Chollet 2017, Szegedy, et al. 2017, Sandler, et al. 2018, Landola, et al. 2014, Tan, et al. 2019). We have applied EfficientDet-D0, ResNet50 based Faster RCNN, SSD using MobileNet v1 and MobileNet v2, as well as ResNet50 based RetinaNet pre-trained networks on two popular datasets: IDRiD Dataset for Fovea and optical Disc Detection along with DDR Dataset for lesion detection (Tan, et al. 2020, Ren, et al. 2015). DeepLab v2, DeepLab v3, and PSPNet with cross-entropy loss and focal loss, respectively were applied to two popular datasets: DDR dataset for Lesion Detection and IDRiD dataset for both lesion as well as organ (fovea and optic disc) segmentation.

Table 3.15. Distribution of classes across different tasks (Segmentation, Object Detection, and Classification) for respective diabetic retinopathy datasets.

Dataset	Segmentation	Object Detection	Classification
Kaggle DRD (Eyepacs 2015)	-	-	Diabetic Retinopathy Grading - No DR - Class 0 - Mild DR - Class 1 - Moderate DR - Class 2 - Severe DR - Class 3 - Proliferative DR - Class 4
IDRiD (Porwal, et al. 2018)	Lesion Segmentation - Hemorrhages (HA) - Microaneurysms (MA) - Soft Exudates (SE)	Detection of Organ Centroids - Optical Disc - Fovea Centralis	

	- Hard Exudates (EX) Organ Segmentation - Optical Disc (OD)		
DDR (Li, et al. 2019)	Lesion Segmentation - Hemorrhages (HA) - Microaneurysms (MA) - Soft Exudates (SE) - Hard Exudates (EX)	Detection of Lesion - Hemorrhages (HA) - Microaneurysms (MA) - Soft Exudates (SE) - Hard Exudates (EX)	Diabetic Retinopathy Grading - No DR - Class 0 - Mild DR - Class 1 - Moderate DR - Class 2 - Severe DR - Class 3 - Proliferative DR - Class 4 - Ungradable - Class 5

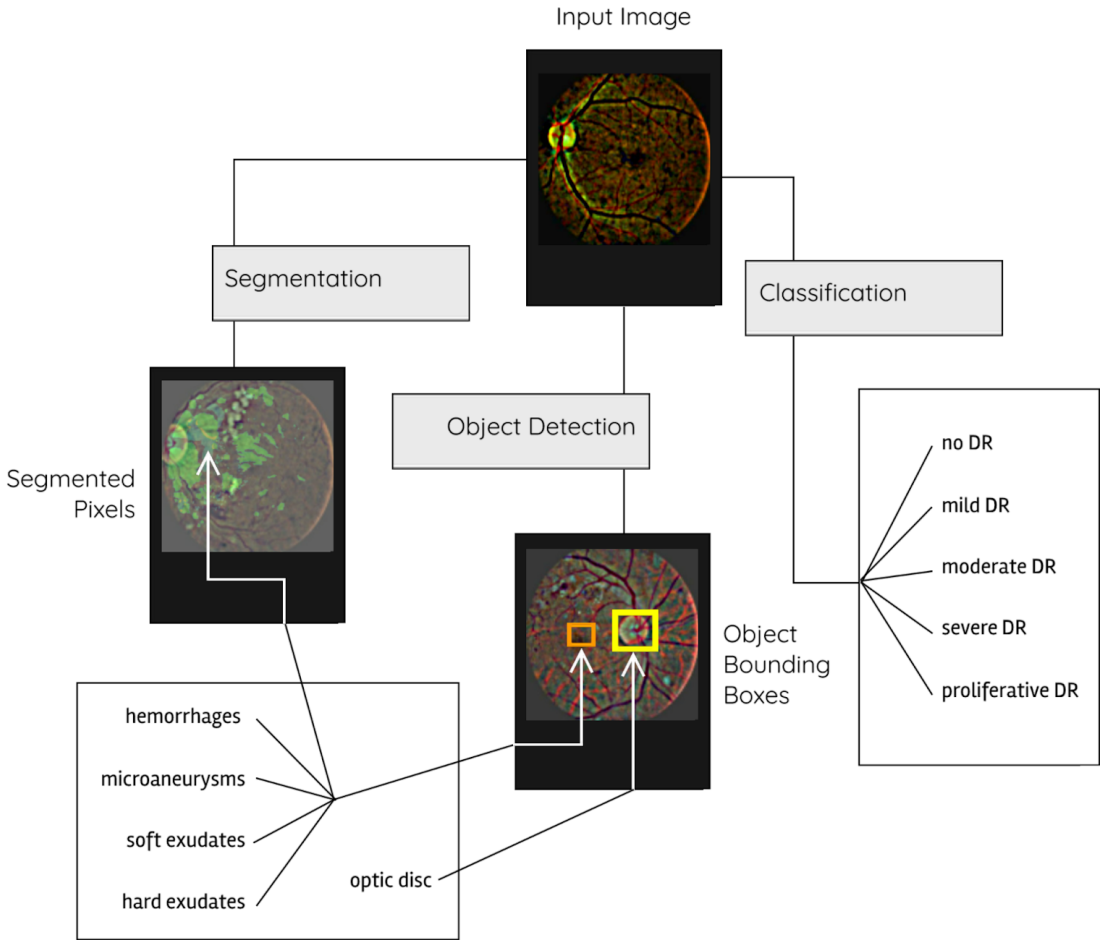


Figure 3.5. Imbalanced diabetic retinopathy detection problem for classification, segmentation, and object detection tasks.

For conducting all experiments related to this work, the TensorFlow v2.3 framework was used with Python 3.8. For the classification and segmentation tasks,

the CNNs were trained with the help of Keras models, while for object detection with the TensorFlow Object Detection API (Huang, et al. 2017) was used. The experiments were accelerated using Google Cloud TPU hardware, access to which was available through the TensorFlow Research Cloud (TRC) program. In all the experiments, TPU v3-8 cores were used for training the deep learning models. The runtime for deep learning experiments related to diabetic retinopathy detection can vary depending on multiple factors, including the specific pre-trained network architecture, the size of the dataset, the complexity of the model, the TPU hardware being used, and the efficiency of the implementation. Approximately 2 hours is a reasonable duration for running such experiments on a TPU, which varies slightly based upon the pre-trained networks for specific tasks involving classification, segmentation and object detection. TPUs (Tensor Processing Units) are specifically designed to accelerate deep learning workloads and can provide significant speed improvements over traditional CPUs or GPUs for certain tasks as they excel at performing matrix operations and handling large-scale neural network computations. For our experimental setup, the batch size of 512 is considered for all the datasets. The total training steps taken was 12,000 (400 epochs x 30 steps per epoch) for DDR and for Kaggle DRD Dataset, while for IDRiD dataset it was set to 1000 (200 epochs x 5 steps per epoch). Various data augmentation operations were applied to all the images present in the train datasets while conducting the classification task, such as horizontal shifts, vertical shifts, rotation, flip, etc., to reduce the over-fitting problem. In all the classification datasets, the original class distributions of samples were imbalanced in nature. The target distribution was achieved such that each class has an equal number of samples represented during the training of the pipeline in all the batches. This was achieved using the rejection resampling technique which is also popularly known as

random undersampling. Hence undersampling technique was applied on-the-fly during the training process at the time of selection of each mini-batch, and it was also ensured that the samples are shuffled and randomly picked from the original distribution. The training process is made to last for sufficiently large numbers of epochs such that all training samples are covered irrespective of the fact that sample rejection has been applied at the batch level. It was observed that without the application of this rejection resampling strategy, each classifier would face the adverse effects of class imbalance. However, data augmentation operations were applied in all the experiments corresponding to the classification and object detection tasks. Data augmentation is a regularization technique that helps to enhance the overall performance of deep learning models. Some data augmentation learning policies had also automated the data augmentation process. We have used the RandAugment technique while training the network. There are several advantages of using RandAugment over other augmentation techniques as it will lead to the reduction of the search space or removal of separate search space because more computational expensive resources were required while training which will help to achieve optimal performance. The RandAugment approach works well for numerous datasets as well as tasks such as object detection and classification. Due to all these unique properties, it results in better performance in comparison to other augmentation methods. The various data augmentation operations that we have applied are random grid shuffle, rescale, horizontal and vertical flips in horizontal and vertical flips, shear in x and y direction, translate in x and y direction, rotate, posterize, contrast, sharpness, and cutout. A cutout is like a dropout at the data level, which creates random black square patches in the images; it was found to be an essential augmentation operation. After many iterations through our experimental setup, we

were able to use the following hyperparameters for all of our classification experiments. The training process was intended at fine-tuning each of the classification networks using pre-trained weights in order to adapt the models for diabetic retinopathy tasks. A standard categorical cross-entropy loss and adaptive momentum optimization (Adam) variant of SGD was used while training. The choice of learning rate was set to lower values in order to ensure that our models continually learn from retinal images without completely exterminating the knowledge on the large-scale ImageNet. Similar to fine-tuning of CNNs in other works, we used an exponentially decaying learning rate with a warmup in the range 1e-5 and 2e-4. Under this setting, the learning rate was initially ramped up linearly to 2e-4 for the first 160 epochs, sustained at that rate for another 80 epochs, and finally decayed exponentially for the remaining epochs using a decay of 0.8. Evaluation on validation was performed each 10 epochs to ensure models can train well for longer. Each of the models was trained for a net budget of 400 sweeps (epochs) and the largely set value of a number of warmup epochs is solely due to the random undersampling technique which significantly reduces samples from each epoch in order to obtain a balanced class distribution at train time. The warmup and sustain phase of the learning rate schedule is slightly longer to ensure that the complete dataset can be iterated across during the training phase by the models before the value of the learning rate drops significantly. The batch size for the experiments was set to 512 and images were rescaled to 224 x 224 which were found to be ideal for training on TPU v3-8 accelerators. TPU v3-8 hardware are ASIC chips designed by Google and operated on the Google Cloud Platform and are specifically developed for fast machine learning and deep learning workloads. It can consume up to 128GB of on-chip high bandwidth memory (HBM) for up to 8 TPU cores which was helpful for accelerating

our experiments, and the higher batch sizes significantly helped consume the large amount of available on-chip memory for which the learning rate was scaled accordingly. (Ying, et al. 2018) For data augmentation along with the rejection resampling technique (random undersampling approach), we slightly modified the RandAugment technique (Cubuk, et al. 2020) to incorporate random flips, and random grid shuffle and rescale transforms. A few sets of randomly augmented samples were generated using this technique. After a series of manual tuning trials, RandAugment was applied with m=8 and n=2 and it was found that adding the extra image operations would greatly improve the classification performance for the diabetes datasets. In the case of lesion segmentation on the DDR and IDRiD datasets, we fine-tuned the segmentation models that are pre-trained on the PASCAL VOC 2012 dataset. The training process was carried out on TPUs using focal loss with gamma set to 2.0 and batch size of 64. The exponentially decayed learning rate schedule was similar to the classification experiments and in the range 1e-5 and 5e-4 with 80 warmup epochs. The focal loss models were able to significantly perform better than categorical cross-entropy loss due to the very high number of background pixels which causes imbalance. Random flip-based data augmentation was used during training on images of size 384 x 384, validation was performed every 25 epochs and each model was trained for a total of 250 epochs. Similar to segmentation and classification, we then used the Adam optimizer for training the object detection models as well. Images of size 512 x 512 were used to train the object detectors for around 10000 steps for each dataset and with each step having a batch size of 32. The training process also involved data augmentation using random flips, random square crop, and random padding with different combinations of random hue, saturation, contrast, and brightness. Due to the stochastic nature of neural networks in all of the

deep learning models, we trained all classification, segmentation, as well as object detection models for a total of five runs with different random seeds, set each time, and the results are represented with the help of mean and standard deviation for each evaluation metric.

Tables 3.16, 3.17, 3.18, 3.19(a), 3.19(b), 3.19(c), 3.20(a), 3.20(b), 3.20(c), 3.21(a), 3.21(b), 3.21(c) illustrate the comparative analysis between various pre-trained models, for the classification task for the three diabetic retinopathy datasets: Kaggle DRD, DDR and IDRiD. From the analysis, it was found that DenseNet121 proves to be an effective model in the case of all three datasets, in comparison to InceptionV3, MobileNetV2, Xception, ResNet50, EfficientNet-B0, VGG16, VGG19, ResNet152, ResNet101, ResNet152V2, ResNet101V2, DenseNet169, DenseNet201, ResNet50V2 and InceptionResNetV2 pre-trained networks. After DenseNet121, the second-most effective pre-trained network is Xception for all the three datasets with respect to various classification evaluation metrics: Cohen's Kappa (unweighted, linear, quadratic weighted), Accuracy, ROC-AUC (weighted and macro average), and F1-score, Index Balanced Accuracy (IBA) and Geometric Mean (GMean) (Japkowicz 2013). Another observation found was that Class 1, early stage diabetes is difficult to detect irrespective of whether that class falls under the minority category or not, in the case of all the three datasets. The third observation inculcated after conducting the experiments was that data augmentation used for training is suitable for longer training (around 400 epochs). The fourth observation found was that samples are equally distributed during training using rejection resampling which had a magnificent impact on the overall performance of the classification task. DenseNet and Xception architectures are both simple architectures amongst other prevalent

architectures present in the study. DenseNet works well in evading the vanishing gradient problem and also enables the reuse of features. DenseNet consists of various dense blocks, and further the layers of the dense block concatenates data from all the previous layers. DenseNet121 pre-trained model is the best suited for diabetic retinopathy image classification task is based upon empirical analysis. The model was trained using five repeated trials, which helps to ensure that the results are statistically significant. The model was compared to other pre-trained models, such as ResNet50, InceptionV3 etc. and it consistently outperformed them across trials. In addition to the empirical results, there are also theoretical reasons to believe that DenseNet121 is a good choice for diabetic retinopathy image classification. DenseNet121 is a deep convolutional neural network that has been shown to be effective for a variety of image classification tasks. It is also known to be relatively robust to overfitting, which is an important consideration for medical image classification tasks. The DenseNet121 model is well-suited for diabetic retinopathy image classification because it has a dense connectivity architecture. This means that each layer of the model is connected to all of the layers that come before it. This allows the model to learn more complex features from the images. Overall, the evidence suggests that DenseNet121 is the best pre-trained model for diabetic retinopathy image classification.We have done extensive comparative analysis between various state-of-the-art methods on three benchmark datasets of diabetic retinopathy: Kaggle DR detection, IDRiD and DDR, for classification by training deep models on varied size datasets. Throughout our study, we have focused on the transfer learning approaches for improving the performance of models. After conducting the analysis, it was found that the DenseNet121 pre-trained model is the best suited for diabetic retinopathy image classification. Tables 3.22 and 3.23 depict the Detection Boxes precision (mean

average precision (mAP)) and Detection Boxes recall (average recall (AR)) metrics for object detection tasks for both DDR and IDRiD datasets. In this mAP metric, we also have an overlap criterion that lays down the minimum value of the intersection over union (IoU), which is used for correct detection. The value was taken as 0.5, 0.75I for the IoU criterion. The results illustrate that in the case of the DDR dataset, SSD (MobileNetV1) is more efficient in lesion detection in comparison to EfficientDet-D0, Faster RCNN (ResNet-50), RetinaNet (ResNet50), and SSD (MobileNetV2) pre-trained networks. However, in the case of the IDRiD dataset for Fovea and Optic Disc Detection, EfficientDet-D0, Faster RCNN (ResNet-50), SSD (MobileNetV1) works well in comparison to other pre-trained networks (RetinaNet (ResNet50) and SSD (MobileNetV2)) while considering mAP without IoU evaluation metrics. But after taking into consideration mAP with IoU value we can see that EfficientDet-D0 is showing better results in comparison to other pre-trained networks in all aspects (when considering small and large objects). Tables 3.24 (a), 3.24 (b), 3.25 (a), and 3.25 (b) show a comparative analysis between various state-of-the-art pre-trained networks for segmentation in the case of DDR and IDRiD datasets, respectively, using Dice score and IoU (intersection over union) evaluation metrics between various pre-trained networks: DeepLabV2 and DeepLabV3 and PSPNet with cross-entropy loss and focal loss. After experimentation it was observed that PSPNet (with Focal Loss) is working best in comparison to other pre-trained networks taken into consideration, results from which are shown in Figure 3.6. Final conclusion which can be inferred from the experimental task was that the DenseNet121 pre-trained model is the best suited for the diabetic retinopathy image classification task. Whereas, EfficientDet-D0 and SSD (MobileNetV1) are best suited based on the diabetic retinopathy dataset for object detection tasks. In case of segmentation PSPNet

(with focal loss) performs best in comparison to other pre-trained networks. It was also observed experimentally that in the case of Class 1, early-stage diabetics are difficult to detect irrespective of whether that class falls under the minority category in the case of all the three available diabetic retinopathy datasets.

Table 3.16. Illustration of classification results of various pre-trained network on Kaggle DR Dataset using Cohen's Kappa and ROC AUC evaluation metrics.

Models	Cohen's Kappa			Accuracy	ROC AUC	
	Unweighted	Linearly Weighted	Quadratic Weighted		Weighted Average	Macro Average
VGG16	0.4305 ± 0.0360	0.6516 ± 0.0175	0.5547 ± 0.0281	0.7706 ± 0.0394	0.8087 ± 0.0030	0.8251 ± 0.0038
VGG19	0.4514 ± 0.0053	0.6638 ± 0.0026	0.5718 ± 0.0034	0.7874 ± 0.0029	0.8078 ± 0.0038	0.8284 ± 0.0019
InceptionV3	0.4134 ± 0.0042	0.6357 ± 0.0050	0.5374 ± 0.0048	0.7814 ± 0.0024	0.8046 ± 0.0020	0.8261 ± 0.0010
ResNet50	0.4336 ± 0.0046	0.6514 ± 0.0048	0.5561 ± 0.0045	0.7906 ± 0.0015	0.8101 ± 0.0021	0.8302 ± 0.0016
ResNet50V2	0.4127 ± 0.0028	0.6341 ± 0.0034	0.5361 ± 0.0034	0.7782 ± 0.0029	0.8004 ± 0.0020	0.8224 ± 0.0021
ResNet152	0.4316 ± 0.0035	0.6465 ± 0.0014	0.5526 ± 0.0020	0.7914 ± 0.0012	0.8103 ± 0.0011	0.8305 ± 0.0005
ResNet101	0.4317 ± 0.0050	0.6488 ± 0.0061	0.5539 ± 0.0057	0.7913 ± 0.0022	0.8097 ± 0.0018	0.8306 ± 0.0017
ResNet152V2	0.4266 ± 0.0051	0.6456 ± 0.0042	0.5493 ± 0.0043	0.7846 ± 0.0017	0.8069 ± 0.0018	0.8252 ± 0.0012
ResNet101V2	0.4229 ± 0.0028	0.6435 ± 0.0011	0.5464 ± 0.0017	0.7827 ± 0.0031	0.8026 ± 0.0015	0.8236 ± 0.0007
Xception	0.4371 ± 0.0019	0.6587 ± 0.0033	0.5618 ± 0.0027	0.7906 ± 0.0014	0.8139 ± 0.0010	0.8310 ± 0.0005
InceptionResNet V2	0.4324 ± 0.0048	0.6513 ± 0.0049	0.5558 ± 0.0044	0.7962 ± 0.0010	0.8140 ± 0.0015	0.8320 ± 0.0009
MobileNetV2	0.4013 ± 0.0050	0.6345 ± 0.0049	0.5309 ± 0.0048	0.7608 ± 0.0058	0.8010 ± 0.0011	0.8227 ± 0.0013
DenseNet121	0.4465 ± 0.0023	0.6678 ± 0.0030	0.5716 ± 0.0027	0.7911 ± 0.0029	0.8189 ± 0.0018	0.8340 ± 0.0021
DenseNet169	0.4462 ± 0.0045	0.6655 ± 0.0044	0.5705 ± 0.0044	0.7963 ± 0.0020	0.8198 ± 0.0020	0.8334 ± 0.0017
DenseNet201	0.4408 ± 0.0071	0.6567 ± 0.0062	0.5628 ± 0.0065	0.7986 ± 0.0012	0.8199 ± 0.0016	0.8344 ± 0.0013
EfficientNetB0	0.4288 ± 0.0035	0.6579 ± 0.0046	0.5565 ± 0.0042	0.7713 ± 0.0028	0.8134 ± 0.0018	0.8317 ± 0.0013

Table 3.17. Illustration of classification results of various pre-trained network on DDR Dataset using Cohen's Kappa and ROC AUC evaluation metrics.

Models	Cohen's Kappa			Accuracy	ROC AUC	
	Unweighted	Linearly Weighted	Quadratic Weighted		Weighted Average	Macro Average
VGG16	0.6228 ± 0.0099	0.8451 ± 0.0059	0.7499 ± 0.0078	0.7589 ± 0.0062	0.9165 ± 0.0029	0.9013 ± 0.0025
VGG19	0.6330 ± 0.0067	0.8490 ± 0.0031	0.7574 ± 0.0042	0.7646 ± 0.0041	0.9135 ± 0.0036	0.9048 ± 0.0029
InceptionV3	0.5662 ± 0.0050	0.8205 ± 0.0039	0.7092 ± 0.0045	0.7249 ± 0.0026	0.8936 ± 0.0037	0.8840 ± 0.0034
ResNet50	0.5950 ± 0.0048	0.8314 ± 0.0056	0.7287 ± 0.0052	0.7427 ± 0.0030	0.9089 ± 0.0020	0.8956 ± 0.0049
ResNet50V2	0.5826 ± 0.0133	0.8261 ± 0.0049	0.7198 ± 0.0091	0.7350 ± 0.0077	0.8989 ± 0.0014	0.8894 ± 0.0036
ResNet152	0.5934 ± 0.0140	0.8320 ± 0.0088	0.7282 ± 0.0118	0.7422 ± 0.0081	0.9093 ± 0.0041	0.8939 ± 0.0019
ResNet101	0.5909 ± 0.0081	0.8285 ± 0.0055	0.7251 ± 0.0066	0.7408 ± 0.0049	0.9119 ± 0.0026	0.8961 ± 0.0037
ResNet152V2	0.5943 ± 0.0056	0.8325 ± 0.0023	0.7292 ± 0.0037	0.7424 ± 0.0037	0.9112 ± 0.0032	0.8977 ± 0.0017
ResNet101V2	0.5909 ± 0.0091	0.8307 ± 0.0056	0.7261 ± 0.0072	0.7398 ± 0.0055	0.9086 ± 0.0017	0.8965 ± 0.0013
Xception	0.5856 ± 0.0199	0.8273 ± 0.0082	0.7221 ± 0.0137	0.7365 ± 0.0120	0.8981 ± 0.0033	0.8826 ± 0.0013
InceptionResNet V2	0.5903 ± 0.0137	0.8296 ± 0.0067	0.7256 ± 0.0099	0.7403 ± 0.0082	0.9012 ± 0.0021	0.8817 ± 0.0049
MobileNetV2	0.5064 ± 0.0290	0.7965 ± 0.0151	0.6689 ± 0.0215	0.6898 ± 0.0170	0.8880 ± 0.0041	0.8694 ± 0.0072
DenseNet121	0.6102 ± 0.0061	0.8393 ± 0.0023	0.7402 ± 0.0033	0.7514 ± 0.0038	0.9139 ± 0.0027	0.8985 ± 0.0065
DenseNet169	0.6043 ± 0.0068	0.8376 ± 0.0043	0.7367 ± 0.0051	0.7485 ± 0.0039	0.9119 ± 0.0029	0.8969 ± 0.0029
DenseNet201	0.6172 ± 0.0045	0.8433 ± 0.0032	0.7458 ± 0.0031	0.7568 ± 0.0026	0.9129 ± 0.0021	0.8907 ± 0.0019
EfficientNetB0	0.5910 ± 0.0059	0.8364 ± 0.0031	0.7306 ± 0.0042	0.7374 ± 0.0040	0.9091 ± 0.0042	0.8962 ± 0.0058

Table 3.18. Illustration of classification results of various pre-trained network on IDRiD Dataset using Cohen's Kappa and ROC AUC evaluation metrics.

Models	Cohen's Kappa			Accuracy	ROC AUC	
	Unweighted	Linearly Weighted	Quadratic Weighted		Weighted Average	Macro Average
VGG16	0.3938 ± 0.0500	0.5662 ± 0.0600	0.4914 ± 0.0538	0.5728 ± 0.0357	0.8114 ± 0.0048	0.7939 ± 0.0074

VGG19	0.3793 ± 0.0303	0.5920 ± 0.0378	0.4979 ± 0.0243	0.5592 ± 0.0244	0.8136 ± 0.0032	0.7987 ± 0.0129
InceptionV3	0.4327 ± 0.0371	0.6007 ± 0.0488	0.5319 ± 0.0341	0.5961 ± 0.0280	0.7841 ± 0.0110	0.7245 ± 0.0187
ResNet50	0.3965 ± 0.0421	0.6079 ± 0.0485	0.5227 ± 0.0471	0.5709 ± 0.0278	0.8013 ± 0.0135	0.7561 ± 0.0226
ResNet50V2	0.3921 ± 0.0481	0.5616 ± 0.0251	0.4888 ± 0.0298	0.5689 ± 0.0340	0.7838 ± 0.0132	0.7335 ± 0.0332
ResNet152	0.4208 ± 0.0563	0.5986 ± 0.0306	0.5212 ± 0.0432	0.5883 ± 0.0404	0.7968 ± 0.0140	0.7539 ± 0.0141
ResNet101	0.4447 ± 0.0472	0.6071 ± 0.0274	0.5389 ± 0.0366	0.6058 ± 0.0334	0.8037 ± 0.0123	0.7577 ± 0.0217
ResNet152V2	0.3421 ± 0.0520	0.5373 ± 0.0264	0.4530 ± 0.0383	0.5340 ± 0.0370	0.7720 ± 0.0193	0.7134 ± 0.0288
ResNet101V2	0.4066 ± 0.0341	0.5916 ± 0.0214	0.5120 ± 0.0131	0.5786 ± 0.0244	0.7735 ± 0.0153	0.7125 ± 0.0271
Xception	0.4182 ± 0.0276	0.5939 ± 0.0189	0.5116 ± 0.0222	0.5903 ± 0.0199	0.8006 ± 0.0091	0.7474 ± 0.0138
InceptionResNet V2	0.3834 ± 0.0429	0.5632 ± 0.0341	0.4867 ± 0.0390	0.5650 ± 0.0310	0.7881 ± 0.0029	0.7560 ± 0.0084
MobileNetV2	0.3768 ± 0.0347	0.5608 ± 0.0268	0.4838 ± 0.0275	0.5592 ± 0.0354	0.7759 ± 0.0096	0.7396 ± 0.0082
DenseNet121	0.3882 ± 0.0243	0.6049 ± 0.0410	0.5092 ± 0.0318	0.5612 ± 0.0160	0.7878 ± 0.0099	0.7352 ± 0.0189
DenseNet169	0.4244 ± 0.0241	0.6206 ± 0.0115	0.5352 ± 0.0127	0.5922 ± 0.0182	0.8021 ± 0.0060	0.7489 ± 0.0144
DenseNet201	0.4089 ± 0.0285	0.5844 ± 0.0210	0.5102 ± 0.0244	0.5806 ± 0.0210	0.8034 ± 0.0065	0.7555 ± 0.0081
EfficientNetB0	0.4089 ± 0.0057	0.6376 ± 0.0299	0.5375 ± 0.0192	0.5767 ± 0.0053	0.8059 ± 0.0096	0.7633 ± 0.0178

Table 3.19 (a). Illustration of classification results of various pre-trained network on Kaggle DR Dataset using F1 Score evaluation metrics.

Models	**F1 Score**					
	Class 0	**Class 1**	**Class 2**	**Class 3**	**Class 4**	**Weighted Average**
VGG16	0.8781 ± 0.0275	0.0935 ± 0.0384	0.5527 ± 0.0286	0.3429 ± 0.0371	0.5364 ± 0.0086	0.7543 ± 0.0208
VGG19	0.8905 ± 0.0019	0.0804 ± 0.0129	0.5701 ± 0.0033	0.3171 ± 0.0196	0.5337 ± 0.0194	0.7644 ± 0.0014
InceptionV3	0.8871 ± 0.0016	0.0659 ± 0.0054	0.5171 ± 0.0067	0.3257 ± 0.0133	0.5066 ± 0.0088	0.7526 ± 0.0009
ResNet50	0.8928 ± 0.0010	0.0606 ± 0.0046	0.5366 ± 0.0067	0.3446 ± 0.0164	0.5232 ± 0.0148	0.7601 ± 0.0010
ResNet50V2	0.8850 ± 0.0019	0.0701 ± 0.0031	0.5150 ± 0.0057	0.3335 ± 0.0254	0.5252 ± 0.0056	0.7517 ± 0.0014

ResNet152	0.8927	±	0.0544	±	0.5331	±	0.3494	±	0.5357	±	0.7595	±
	0.0007		0.0028		0.0045		0.0053		0.0050		0.0005	
ResNet101	0.8931	±	0.0470	±	0.5348	±	0.3444	±	0.5248	±	0.7591	±
	0.0013		0.0090		0.0046		0.0219		0.0106		0.0019	
ResNet152V2	0.8891	±	0.0710	±	0.5280	±	0.3535	±	0.5280	±	0.7572	±
	0.0010		0.0108		0.0060		0.0202		0.0070		0.0009	
ResNet101V2	0.8879	±	0.0701	±	0.5260	±	0.3408	±	0.5290	±	0.7557	±
	0.0020		0.0039		0.0044		0.0101		0.0045		0.0011	
Xception	0.8928	±	0.0657	±	0.5421	±	0.3436	±	0.5396	±	0.7616	±
	0.0011		0.0044		0.0035		0.0127		0.0095		0.0009	
InceptionResNet V2	0.8960	±	0.0506	±	0.5310	±	0.3350	±	0.5315	±	0.7609	±
	0.0008		0.0076		0.0052		0.0184		0.0148		0.0012	
MobileNetV2	0.8748	±	0.1043	±	0.5180	±	0.3421	±	0.5077	±	0.7468	±
	0.0032		0.0062		0.0031		0.0189		0.0251		0.0031	
DenseNet121	0.8935	±	0.0745	±	0.5527	±	0.3530	±	0.5474	±	0.7647	±
	0.0018		0.0068		0.0025		0.0158		0.0094		0.0013	
DenseNet169	0.8964	±	0.0525	±	0.5494	±	0.3488	±	0.5511	±	0.7648	±
	0.0010		0.0040		0.0063		0.0050		0.0064		0.0017	
DenseNet201	0.8970	±	0.0541	±	0.5431	±	0.3430	±	0.5343	±	0.7639	±
	0.0008		0.0068		0.0077		0.0081		0.0158		0.0020	
EfficientNetB0	0.8806	±	0.1043	±	0.5411	±	0.3623	±	0.5590	±	0.7562	±
	0.0018		0.0028		0.0042		0.0071		0.0044		0.0018	

Table 3.19 (b). Illustration of classification results of various pre-trained network on Kaggle DR Dataset using Index Balanced Accuracy (IBA) evaluation metrics.

Models	**Index Balanced Accuracy (IBA)**											
	Class 0		**Class 1**		**Class 2**		**Class 3**		**Class 4**		**Weighted Average**	
VGG16	0.5312	±	0.0695	±	0.4990	±	0.2582	±	0.4026	±	0.5201	±
	0.0058		0.0540		0.0118		0.0776		0.0563		0.0045	
VGG19	0.5426	±	0.0487	±	0.5182	±	0.2152	±	0.3806	±	0.5304	±
	0.0040		0.0101		0.0101		0.0241		0.0210		0.0032	
InceptionV3	0.4914	±	0.0375	±	0.4350	±	0.2372	±	0.3632	±	0.4930	±
	0.0110		0.0035		0.0161		0.0158		0.0079		0.0068	
ResNet50	0.5036	±	0.0332	±	0.4531	±	0.2503	±	0.3758	±	0.5041	±
	0.0116		0.0033		0.0161		0.0178		0.0180		0.0074	
ResNet50V2	0.4975	±	0.0407	±	0.4400	±	0.2467	±	0.3812	±	0.4961	±
	0.0080		0.0028		0.0078		0.0262		0.0125		0.0047	
ResNet152	0.4962	±	0.0291	±	0.4470	±	0.2525	±	0.3850	±	0.4997	±
	0.0092		0.0017		0.0113		0.0078		0.0039		0.0056	
ResNet101	0.4980	±	0.0252	±	0.4494	±	0.2507	±	0.3729	±	0.5006	±
	0.0100		0.0054		0.0123		0.0256		0.0158		0.0064	
ResNet152V2	0.5059	±	0.0405	±	0.4504	±	0.2637	±	0.3812	±	0.5038	±
	0.0081		0.0075		0.0154		0.0178		0.0093		0.0054	
ResNet101V2	0.5045	±	0.0403	±	0.4499	±	0.2528	±	0.3789	±	0.5022	±
	0.0075		0.0027		0.0126		0.0107		0.0054		0.0044	

Xception	0.5092	±	0.0371	±	0.4582	±	0.2487	±	0.3964	±	0.5080	±
	0.0028		0.0029		0.0055		0.0145		0.0092		0.0019	
InceptionResNet V2	0.4859	±	0.0264	±	0.4319	±	0.2401	±	0.3762	±	0.4942	±
	0.0084		0.0045		0.0130		0.0195		0.0186		0.0055	
MobileNetV2	0.5167	±	0.0758	±	0.4517	±	0.2614	±	0.3666	±	0.5045	±
	0.0070		0.0072		0.0095		0.0241		0.0261		0.0033	
DenseNet121	0.5251	±	0.0434	±	0.4752	±	0.2613	±	0.4020	±	0.5190	±
	0.0044		0.0048		0.0043		0.0224		0.0138		0.0023	
DenseNet169	0.5105	±	0.0283	±	0.4623	±	0.2541	±	0.4016	±	0.5106	±
	0.0028		0.0025		0.0068		0.0101		0.0091		0.0024	
DenseNet201	0.4924	±	0.0282	±	0.4450	±	0.2421	±	0.3860	±	0.4997	±
	0.0118		0.0039		0.0142		0.0091		0.0185		0.0078	
EfficientNetB0	0.5384	±	0.0708	±	0.4852	±	0.2804	±	0.4309	±	0.5234	±
	0.0018		0.0025		0.0039		0.0062		0.0053		0.0012	

Table 3.19 (c). Illustration of classification results of various pre-trained network on Kaggle DR Dataset using Geometric Mean (Gmean) evaluation metrics.

Models	**Geometric Mean (Gmean)**											
	Class 0		**Class 1**		**Class 2**		**Class 3**		**Class 4**		**Weighted Average**	
VGG16	0.7170	±	0.2632	±	0.7192	±	0.5228	±	0.6521	±	0.7176	±
	0.0078		0.0901		0.0090		0.0708		0.0416		0.0018	
VGG19	0.7235	±	0.2304	±	0.7323	±	0.4820	±	0.6358	±	0.7239	±
	0.0028		0.0246		0.0064		0.0258		0.0171		0.0022	
InceptionV3	0.6861	±	0.2034	±	0.6745	±	0.5057	±	0.6218	±	0.6966	±
	0.0084		0.0095		0.0115		0.0163		0.0065		0.0052	
ResNet50	0.6945	±	0.1914	±	0.6879	±	0.5191	±	0.6320	±	0.7043	±
	0.0088		0.0096		0.0113		0.0180		0.0145		0.0056	
ResNet50V2	0.6910	±	0.2117	±	0.6780	±	0.5150	±	0.6364	±	0.6991	±
	0.0061		0.0072		0.0056		0.0267		0.0099		0.0037	
ResNet152	0.6890	±	0.1792	±	0.6836	±	0.5215	±	0.6395	±	0.7009	±
	0.0069		0.0052		0.0080		0.0078		0.0031		0.0042	
ResNet101	0.6903	±	0.1661	±	0.6853	±	0.5192	±	0.6297	±	0.7016	±
	0.0075		0.0180		0.0087		0.0259		0.0128		0.0048	
ResNet152V2	0.6967	±	0.2108	±	0.6857	±	0.5325	±	0.6365	±	0.7045	±
	0.0060		0.0188		0.0109		0.0175		0.0074		0.0040	
ResNet101V2	0.6958	±	0.2107	±	0.6853	±	0.5218	±	0.6346	±	0.7034	±
	0.0058		0.0071		0.0088		0.0107		0.0044		0.0034	
Xception	0.6987	±	0.2023	±	0.6917	±	0.5176	±	0.6485	±	0.7072	±
	0.0022		0.0079		0.0039		0.0147		0.0072		0.0015	
InceptionResNet V2	0.6809	±	0.1704	±	0.6729	±	0.5086	±	0.6323	±	0.6965	±
	0.0063		0.0144		0.0094		0.0207		0.0149		0.0041	
MobileNetV2	0.7066	±	0.2878	±	0.6862	±	0.5299	±	0.6243	±	0.7066	±
	0.0056		0.0133		0.0065		0.0242		0.0215		0.0028	
DenseNet121	0.7103	±	0.2184	±	0.7036	±	0.5299	±	0.6528	±	0.7153	±
	0.0034		0.0121		0.0028		0.0217		0.0107		0.0018	

DenseNet169	0.6992	±	0.1767	±	0.6948	±	0.5231	±	0.6526	±	0.7087	±
	0.0020		0.0077		0.0048		0.0100		0.0070		0.0017	
DenseNet201	0.6856	±	0.1764	±	0.6826	±	0.5110	±	0.6402	±	0.7005	±
	0.0088		0.0119		0.0103		0.0094		0.0146		0.0058	
EfficientNetB0	0.7219	±	0.2786	±	0.7097	±	0.5487	±	0.6749	±	0.7199	±
	0.0014		0.0048		0.0027		0.0059		0.0040		0.0008	

Table 3.20 (a). Illustration of classification results of various pre-trained networks on DDR Dataset using F1 Score evaluation metrics.

Models	**F1 Score**						
	Class 0	**Class 1**	**Class 2**	**Class 3**	**Class 4**	**Class 5**	**Weighted Average**
VGG16	0.8487 ± 0.0044	0.0610 ± 0.0294	0.6592 ± 0.0097	0.2698 ± 0.0191	0.7188 ± 0.0220	0.8606 ± 0.0086	0.7327 ± 0.0069
VGG19	0.8588 ± 0.0027	0.0609 ± 0.0153	0.6741 ± 0.0071	0.2879 ± 0.0322	0.6971 ± 0.0207	0.8561 ± 0.0080	0.7407 ± 0.0044
InceptionV3	0.8292 ± 0.0016	0.0360 ± 0.0093	0.5780 ± 0.0098	0.2967 ± 0.0348	0.6470 ± 0.0133	0.8671 ± 0.0066	0.6922 ± 0.0044
ResNet50	0.8392 ± 0.0041	0.0316 ± 0.0066	0.6208 ± 0.0104	0.3520 ± 0.0383	0.6774 ± 0.0193	0.8636 ± 0.0062	0.7133 ± 0.0037
ResNet50V2	0.8338 ± 0.0062	0.0507 ± 0.0060	0.5976 ± 0.0202	0.2816 ± 0.0672	0.6895 ± 0.0122	0.8611 ± 0.0038	0.7035 ± 0.0102
ResNet152	0.8393 ± 0.0064	0.0387 ± 0.0117	0.6182 ± 0.0207	0.2760 ± 0.0316	0.6683 ± 0.0106	0.8622 ± 0.0070	0.7108 ± 0.0100
ResNet101	0.8381 ± 0.0028	0.0362 ± 0.0139	0.6145 ± 0.0118	0.2750 ± 0.0288	0.6700 ± 0.0078	0.8666 ± 0.0105	0.7093 ± 0.0060
ResNet152 V2	0.8386 ± 0.0037	0.0423 ± 0.0171	0.6131 ± 0.0101	0.2682 ± 0.0389	0.7081 ± 0.0155	0.8651 ± 0.0067	0.7118 ± 0.0039
ResNet101 V2	0.8393 ± 0.0067	0.0599 ± 0.0119	0.6087 ± 0.0133	0.3112 ± 0.0598	0.6877 ± 0.0241	0.8550 ± 0.0045	0.7100 ± 0.0065
Xception	0.8342 ± 0.0091	0.0263 ± 0.0064	0.6031 ± 0.0295	0.3615 ± 0.0366	0.6794 ± 0.0059	0.8643 ± 0.0037	0.7054 ± 0.0146
Inception ResNetV2	0.8371 ± 0.0062	0.0260 ± 0.0148	0.6083 ± 0.0227	0.2460 ± 0.0265	0.6867 ± 0.0115	0.8728 ± 0.0038	0.7076 ± 0.0106
MobileNet V2	0.8136 ± 0.0131	0.0220 ± 0.0185	0.5256 ± 0.0488	0.3024 ± 0.0669	0.3582 ± 0.0788	0.8069 ± 0.0184	0.6429 ± 0.0226
DenseNet 121	0.8473 ± 0.0044	0.0482 ± 0.0054	0.6322 ± 0.0105	0.3278 ± 0.0314	0.7048 ± 0.0183	0.8663 ± 0.0037	0.7231 ± 0.0043
DenseNet 169	0.8425 ± 0.0031	0.0431 ± 0.0148	0.6253 ± 0.0069	0.3165 ± 0.0084	0.7038 ± 0.0162	0.8679 ± 0.0027	0.7183 ± 0.0049
DenseNet 201	0.8489 ± 0.0039	0.0223 ± 0.0123	0.6461 ± 0.0058	0.3029 ± 0.0281	0.7010 ± 0.0178	0.8686 ± 0.0055	0.7268 ± 0.0027
EfficientNet B0	0.8399 ± 0.0035	0.0830 ± 0.0125	0.5928 ± 0.0068	0.3961 ± 0.0174	0.7205 ± 0.0112	0.8668 ± 0.0041	0.7107 ± 0.0035

Table 3.20 (b). Illustration of classification results of various pre-trained networks on DDR Dataset using Indexed Balanced Accuracy (IBA) evaluation metrics.

Models	Index Balanced Accuracy (IBA)						
	Class 0	**Class 1**	**Class 2**	**Class 3**	**Class 4**	**Class 5**	**Weighted Average**
VGG16	0.7278 ± 0.0084	0.0342 ± 0.0176	0.5072 ± 0.0118	0.1780 ± 0.0111	0.6238 ± 0.0312	0.9103 ± 0.0087	0.6384 ± 0.0081
VGG19	0.7494 ± 0.0059	0.0341 ± 0.0085	0.5303 ± 0.0110	0.1912 ± 0.0146	0.6148 ± 0.0397	0.8866 ± 0.0092	0.6499 ± 0.0057
Inception-V3	0.6842 ± 0.0046	0.0199 ± 0.0052	0.4181 ± 0.0112	0.1966 ± 0.0302	0.5329 ± 0.0218	0.9194 ± 0.0066	0.5908 ± 0.0045
ResNet50	0.7065 ± 0.0091	0.0170 ± 0.0042	0.4654 ± 0.0136	0.2393 ± 0.0360	0.5574 ± 0.0255	0.9098 ± 0.0062	0.6142 ± 0.0047
ResNet50 V2	0.6936 ± 0.0150	0.0275 ± 0.0039	0.4373 ± 0.0232	0.1914 ± 0.0514	0.5735 ± 0.0094	0.9193 ± 0.0064	0.6035 ± 0.0118
ResNet152	0.7051 ± 0.0153	0.0199 ± 0.0062	0.4612 ± 0.0238	0.1808 ± 0.0321	0.5552 ± 0.0137	0.9106 ± 0.0158	0.6127 ± 0.0124
ResNet101	0.7017 ± 0.0070	0.0190 ± 0.0075	0.4554 ± 0.0144	0.1859 ± 0.0239	0.5587 ± 0.0068	0.9081 ± 0.0131	0.6104 ± 0.0068
ResNet152 V2	0.7038 ± 0.0072	0.0228 ± 0.0091	0.4544 ± 0.0103	0.1729 ± 0.0302	0.5979 ± 0.0193	0.9139 ± 0.0079	0.6129 ± 0.0048
ResNet101 V2	0.7057 ± 0.0151	0.0332 ± 0.0067	0.4501 ± 0.0164	0.2074 ± 0.0502	0.5761 ± 0.0201	0.9081 ± 0.0050	0.6112 ± 0.0086
Xception	0.6953 ± 0.0209	0.0142 ± 0.0034	0.4437 ± 0.0342	0.2473 ± 0.0315	0.5651 ± 0.0128	0.9294 ± 0.0074	0.6062 ± 0.0172
Inception ResNetV2	0.7000 ± 0.0146	0.0133 ± 0.0078	0.4498 ± 0.0260	0.1622 ± 0.0237	0.5794 ± 0.0134	0.9238 ± 0.0136	0.6094 ± 0.0120
MobileNet V2	0.6464 ± 0.0353	0.0114 ± 0.0104	0.3680 ± 0.0556	0.2759 ± 0.1484	0.2112 ± 0.0589	0.9336 ± 0.0044	0.5470 ± 0.0237
DenseNet 121	0.7230 ± 0.0094	0.0265 ± 0.0026	0.4766 ± 0.0142	0.2179 ± 0.0286	0.5991 ± 0.0301	0.9220 ± 0.0122	0.6275 ± 0.0059
DenseNet 169	0.7115 ± 0.0073	0.0228 ± 0.0085	0.4662 ± 0.0090	0.2177 ± 0.0109	0.6017 ± 0.0259	0.9190 ± 0.0096	0.6212 ± 0.0059
DenseNet 201	0.7250 ± 0.0091	0.0114 ± 0.0064	0.4912 ± 0.0072	0.2045 ± 0.0197	0.5939 ± 0.0291	0.9168 ± 0.0050	0.6324 ± 0.0042
EfficientNet B0	0.7079 ± 0.0066	0.0537 ± 0.0086	0.4285 ± 0.0076	0.3139 ± 0.0315	0.6233 ± 0.0108	0.9316 ± 0.0072	0.6124 ± 0.0047

Table 3.20 (c). Illustration of classification results of various pre-trained networks on DDR Dataset using Geometric Mean (Gmean) evaluation metrics.

Models	Geometric Mean (Gmean)						
	Class 0	**Class 1**	**Class 2**	**Class 3**	**Class 4**	**Class 5**	**Weighted Average**
VGG16	0.8431 ± 0.0051	0.1876 ± 0.0561	0.7250 ± 0.0080	0.4397 ± 0.0133	0.8032 ± 0.0191	0.9562 ± 0.0042	0.8026 ± 0.0050
VGG19	0.8569 ± 0.0038	0.1931 ± 0.0233	0.7398 ± 0.0069	0.4553 ± 0.0169	0.7974 ± 0.0236	0.9449 ± 0.0045	0.8099 ± 0.0035
Inception-V3	0.8153 ± 0.0034	0.1472 ± 0.0200	0.6611 ± 0.0083	0.4608 ± 0.0336	0.7458 ± 0.0144	0.9605 ± 0.0032	0.7725 ± 0.0030

ResNet50	0.8295 ± 0.0059	0.1366 ± 0.0163	0.6956 ± 0.0094	0.5070 ± 0.0378	0.7619 ± 0.0166	0.9560 ± 0.0031	0.7874 ± 0.0030
ResNet50 V2	0.8211 ± 0.0098	0.1741 ± 0.0124	0.6756 ± 0.0170	0.4520 ± 0.0630	0.7724 ± 0.0059	0.9604 ± 0.0030	0.7805 ± 0.0077
ResNet152	0.8282 ± 0.0099	0.1468 ± 0.0258	0.6927 ± 0.0170	0.4418 ± 0.0389	0.7605 ± 0.0089	0.9563 ± 0.0075	0.7863 ± 0.0080
ResNet101	0.8259 ± 0.0046	0.1425 ± 0.0296	0.6889 ± 0.0101	0.4486 ± 0.0283	0.7628 ± 0.0043	0.9553 ± 0.0064	0.7849 ± 0.0043
ResNet152 V2	0.8274 ± 0.0043	0.1557 ± 0.0345	0.6881 ± 0.0075	0.4323 ± 0.0381	0.7876 ± 0.0118	0.9580 ± 0.0038	0.7865 ± 0.0030
ResNet101 V2	0.8287 ± 0.0097	0.1908 ± 0.0189	0.6849 ± 0.0118	0.4708 ± 0.0594	0.7739 ± 0.0128	0.9550 ± 0.0024	0.7855 ± 0.0056
Xception	0.8224 ± 0.0135	0.1247 ± 0.0150	0.6800 ± 0.0248	0.5156 ± 0.0310	0.7669 ± 0.0081	0.9652 ± 0.0035	0.7822 ± 0.0110
Inception ResNetV2	0.8249 ± 0.0094	0.1170 ± 0.0361	0.6846 ± 0.0188	0.4194 ± 0.0305	0.7760 ± 0.0084	0.9627 ± 0.0064	0.7842 ± 0.0077
MobileNet V2	0.7907 ± 0.0237	0.1042 ± 0.0469	0.6203 ± 0.0434	0.5276 ± 0.1402	0.4742 ± 0.0664	0.9656 ± 0.0019	0.7437 ± 0.0161
DenseNet 121	0.8396 ± 0.0059	0.1711 ± 0.0085	0.7038 ± 0.0096	0.4847 ± 0.0326	0.7881 ± 0.0186	0.9617 ± 0.0057	0.7958 ± 0.0037
DenseNet 169	0.8321 ± 0.0047	0.1565 ± 0.0300	0.6968 ± 0.0063	0.4852 ± 0.0118	0.7897 ± 0.0159	0.9604 ± 0.0045	0.7917 ± 0.0038
DenseNet 201	0.8404 ± 0.0058	0.1083 ± 0.0335	0.7142 ± 0.0048	0.4703 ± 0.0220	0.7849 ± 0.0179	0.9594 ± 0.0023	0.7987 ± 0.0027
EfficientNet B0	0.8303 ± 0.0039	0.2426 ± 0.0196	0.6697 ± 0.0056	0.5790 ± 0.0273	0.8031 ± 0.0065	0.9662 ± 0.0033	0.7865 ± 0.0029

Table 3.21 (a). Illustration of classification results of various pre-trained networks on IDRiD Dataset using F1 Score evaluation metrics.

Models	**F1 Score**					
	Class 0	**Class 1**	**Class 2**	**Class 3**	**Class 4**	**Weighted Average**
VGG16	0.7112 ± 0.0537	0.0000 ± 0.0000	0.5901 ± 0.0266	0.4909 ± 0.0176	0.2259 ± 0.1669	0.5371 ± 0.0385
VGG19	0.6759 ± 0.0274	0.0000 ± 0.0000	0.5764 ± 0.0388	0.4809 ± 0.0758	0.3474 ± 0.0419	0.5347 ± 0.0202
InceptionV3	0.7322 ± 0.0336	0.0000 ± 0.0000	0.6058 ± 0.0288	0.5716 ± 0.0249	0.2500 ± 0.0859	0.5669 ± 0.0283
ResNet50	0.7461 ± 0.0275	0.0000 ± 0.0000	0.5911 ± 0.0306	0.4537 ± 0.0808	0.1732 ± 0.1196	0.5355 ± 0.0323
ResNet50V2	0.6768 ± 0.0250	0.0000 ± 0.0000	0.5963 ± 0.0207	0.5574 ± 0.1054	0.2089 ± 0.1419	0.5378 ± 0.0379
ResNet152	0.7228 ± 0.0201	0.0000 ± 0.0000	0.6128 ± 0.0587	0.5330 ± 0.0670	0.2599 ± 0.0483	0.5601 ± 0.0380
ResNet101	0.7287 ± 0.0331	0.0500 ± 0.1118	0.6263 ± 0.0397	0.5679 ± 0.0602	0.2704 ± 0.0375	0.5764 ± 0.0326

ResNet152V2	0.6520	±	0.0000	±	0.5840	±	0.4406	±	0.1948	±	0.5025	±
	0.0385		0.0000		0.0517		0.0693		0.0450		0.0346	
ResNet101V2	0.6897	±	0.0000	±	0.5941	±	0.5423	±	0.3076	±	0.5511	±
	0.0186		0.0000		0.0362		0.0698		0.0483		0.0253	
Xception	0.7218	±	0.0000	±	0.5945	±	0.5507	±	0.2732	±	0.5590	±
	0.0355		0.0000		0.0218		0.0266		0.0644		0.0172	
InceptionResNet V2	0.6947	±	0.0000	±	0.6088	±	0.4599	±	0.1665	±	0.5243	±
	0.0343		0.0000		0.0294		0.0629		0.0457		0.0263	
MobileNetV2	0.7046	±	0.0421	±	0.5600	±	0.4197	±	0.2943	±	0.5232	±
	0.0396		0.0942		0.0335		0.0462		0.0487		0.0164	
DenseNet121	0.7039	±	0.0500	±	0.5759	±	0.4787	±	0.3117	±	0.5413	±
	0.0300		0.1118		0.0229		0.0609		0.0923		0.0198	
DenseNet169	0.7349	±	0.0000	±	0.6012	±	0.5069	±	0.3032	±	0.5611	±
	0.0175		0.0000		0.0200		0.0692		0.0616		0.0171	
DenseNet201	0.7147	±	0.0000	±	0.6057	±	0.5272	±	0.2191	±	0.5490	±
	0.0309		0.0000		0.0205		0.0368		0.0531		0.0209	
EfficientNetB0	0.7148	±	0.0000	±	0.5450	±	0.5358	±	0.4210	±	0.5573	±
	0.0179		0.0000		0.0100		0.0748		0.0347		0.0058	

Table 3.21 (b). Illustration of classification results of various pre-trained networks on IDRiD Dataset using Indexed Balanced Accuracy (IBA) evaluation metrics.

Models	**Index Balanced Accuracy (IBA)**											
	Class 0		**Class 1**		**Class 2**		**Class 3**		**Class 4**		**Weighted Average**	
VGG16	0.6088	±	0.0000	±	0.4857	±	0.3579	±	0.1395	±	0.4579	±
	0.0748		0.0000		0.0353		0.0206		0.1110		0.0368	
VGG19	0.5667	±	0.0000	±	0.4711	±	0.3559	±	0.2227	±	0.4453	±
	0.0354		0.0000		0.0471		0.0739		0.0324		0.0216	
InceptionV3	0.6451	±	0.0000	±	0.5078	±	0.4544	±	0.1520	±	0.4869	±
	0.0450		0.0000		0.0375		0.0257		0.0583		0.0276	
ResNet50	0.6644	±	0.0000	±	0.4873	±	0.3497	±	0.1093	±	0.4623	±
	0.0423		0.0000		0.0374		0.0853		0.0784		0.0313	
ResNet50V2	0.5779	±	0.0000	±	0.4935	±	0.4229	±	0.1370	±	0.4559	±
	0.0344		0.0000		0.0261		0.1064		0.0983		0.0339	
ResNet152	0.6175	±	0.0000	±	0.5185	±	0.4143	±	0.1525	±	0.4784	±
	0.0299		0.0000		0.0765		0.0646		0.0316		0.0420	
ResNet101	0.6328	±	0.0361	±	0.5360	±	0.4466	±	0.1536	±	0.4958	±
	0.0448		0.0808		0.0533		0.0733		0.0309		0.0367	
ResNet152V2	0.5381	±	0.0000	±	0.4811	±	0.3029	±	0.1220	±	0.4206	±
	0.0456		0.0000		0.0640		0.0575		0.0304		0.0380	
ResNet101V2	0.5895	±	0.0000	±	0.4929	±	0.4244	±	0.1939	±	0.4658	±
	0.0284		0.0000		0.0460		0.0656		0.0320		0.0252	
Xception	0.6282	±	0.0000	±	0.4934	±	0.3957	±	0.1662	±	0.4749	±
	0.0510		0.0000		0.0279		0.0221		0.0399		0.0215	
InceptionResNet V2	0.5958	±	0.0000	±	0.5098	±	0.3366	±	0.0958	±	0.4512	±
	0.0520		0.0000		0.0362		0.0750		0.0366		0.0320	
MobileNetV2	0.6228	±	0.0669	±	0.4490	±	0.2604	±	0.1803	±	0.4444	±
	0.0595		0.1495		0.0407		0.0426		0.0380		0.0271	

DenseNet121	0.5930	±	0.0361	±	0.4710	±	0.3701	±	0.2188	±	0.4529	±
	0.0354		0.0808		0.0288		0.0730		0.0897		0.0181	
DenseNet169	0.6450	±	0.0000	±	0.5014	±	0.3781	±	0.1936	±	0.4804	±
	0.0225		0.0000		0.0264		0.0914		0.0576		0.0176	
DenseNet201	0.6089	±	0.0000	±	0.5082	±	0.4118	±	0.1247	±	0.4692	±
	0.0429		0.0000		0.0268		0.0389		0.0314		0.0214	
EfficientNetB0	0.6247	±	0.0000	±	0.4319	±	0.4296	±	0.2929	±	0.4664	±
	0.0266		0.0000		0.0103		0.0887		0.0559		0.0042	

Table 3.21 (c). Illustration of classification results of various pre-trained networks on IDRiD Dataset using Geometric Mean (Gmean) evaluation metrics.

Models	**Geometric Mean (GMean)**											
	Class 0		**Class 1**		**Class 2**		**Class 3**		**Class 4**		**Weighted Average**	
VGG16	0.7834	±	0.0000	±	0.6915	±	0.6151	±	0.3376	±	0.6847	±
	0.0453		0.0000		0.0254		0.0164		0.2145		0.0270	
VGG19	0.7565	±	0.0000	±	0.6838	±	0.6107	±	0.4892	±	0.6760	±
	0.0223		0.0000		0.0326		0.0586		0.0332		0.0157	
InceptionV3	0.8040	±	0.0000	±	0.7122	±	0.6890	±	0.3994	±	0.7061	±
	0.0273		0.0000		0.0243		0.0185		0.0822		0.0194	
ResNet50	0.8155	±	0.0000	±	0.6980	±	0.6033	±	0.3028	±	0.6886	±
	0.0240		0.0000		0.0275		0.0728		0.1835		0.0228	
ResNet50V2	0.7600	±	0.0000	±	0.7035	±	0.6625	±	0.3396	±	0.6835	±
	0.0210		0.0000		0.0180		0.0797		0.2023		0.0246	
ResNet152	0.7907	±	0.0000	±	0.7142	±	0.6577	±	0.4057	±	0.6997	±
	0.0176		0.0000		0.0533		0.0506		0.0389		0.0304	
ResNet101	0.7983	±	0.0885	±	0.7281	±	0.6819	±	0.4075	±	0.7120	±
	0.0269		0.1979		0.0345		0.0527		0.0378		0.0253	
ResNet152V2	0.7373	±	0.0000	±	0.6891	±	0.5661	±	0.3617	±	0.6571	±
	0.0307		0.0000		0.0455		0.0522		0.0498		0.0294	
ResNet101V2	0.7694	±	0.0000	±	0.7016	±	0.6655	±	0.4566	±	0.6909	±
	0.0164		0.0000		0.0312		0.0477		0.0397		0.0178	
Xception	0.7945	±	0.0000	±	0.6983	±	0.6465	±	0.4224	±	0.6971	±
	0.0298		0.0000		0.0189		0.0170		0.0493		0.0153	
InceptionResNet V2	0.7734	±	0.0000	±	0.7107	±	0.5938	±	0.3193	±	0.6800	±
	0.0302		0.0000		0.0263		0.0648		0.0596		0.0232	
MobileNetV2	0.7823	±	0.1185	±	0.6734	±	0.5276	±	0.4400	±	0.6751	±
	0.0335		0.2650		0.0281		0.0434		0.0467		0.0194	
DenseNet121	0.7755	±	0.0885	±	0.6850	±	0.6217	±	0.4769	±	0.6821	±
	0.0227		0.1979		0.0202		0.0587		0.0951		0.0133	
DenseNet169	0.8049	±	0.0000	±	0.7058	±	0.6277	±	0.4535	±	0.7014	±
	0.0136		0.0000		0.0149		0.0729		0.0654		0.0125	
DenseNet201	0.7850	±	0.0000	±	0.7086	±	0.6567	±	0.3660	±	0.6935	±
	0.0259		0.0000		0.0176		0.0289		0.0513		0.0152	
EfficientNetB0	0.7909	±	0.0000	±	0.6610	±	0.6673	±	0.5573	±	0.6918	±
	0.0155		0.0000		0.0080		0.0658		0.0523		0.0030	

Table 3.22. Illustration of lesion detection results of various pre-trained networks on DDR Dataset using mAP, AR evaluation metrics.

Models	Detection Boxes Precision						Detection Boxes Recall					
	mAP	**mAP@ 0.5IoU**	**mAP@ 0.75IoU**	**mAP (small)**	**mAP (medium)**	**mAP (large)**	**AR @1**	**AR @10**	**AR @100**	**AR @100 (small)**	**AR @100 (medium)**	**AR @100 (large)**
Efficient Det-D0	0.0065 ± 0.012	0.0189 ± 0.0210	0.0047 ± 0.0011	0.0015 ±0.0001	0.0129 ±0.0121	0.0296 ±0.0001	0.0070 ±0.0018	0.0123 ±0.0007	0.0175 ±0.0017	0.0037 ±0.0000	0.0401 ± 0.0014	0.0770 ± 0.0010
Faster RCNN (ResNet-50)	0.0042 ± 0.0023	0.0139 ± 0.0021	0.0001 ± 0.0001	0.0007 ±0.0000	0.0098 ±0.0000	0.0441 ±0.0021	0.0083 ± 0.0001	0.0151 ±0.0012	0.0712 ±0.0102	0.0016 ±0.0007	0.0318 ±0.0024	0.1199 ±0.0171
SSD (MobileNetV1)	0.0206 ± 0.0101	0.0425 ± 0.0201	0.0312 ±0.0001	0.0035 ± 0.0009	0.0482 ±0.0013	0.1038 ±0.0012	0.0180 ± 0.0023	0.00387 ± 0.0102	0.0501 ±0.0012	0.0192 ±0.0103	0.1201 ±0.0027	0.01753 ±0.0102
RetinaNet (ResNet50)	0.0163 ± 0.0064	0.0381 ±0.0016	0.0152 ± 0.0230	0.0012 ± 0.0102	0.0327 ± 0.0121	0.1319 ± 0.0014	0.0199 ± 0.0101	0.0402 ± 0.0078	0.0513 ± 0.0018	0.0191 ± 0.0012	0.1298 ± 0.0201	0.1645 ± 0.0019
SSD (MobileNetV2)	0.0192 ±0.0209	0.0283 ± 0.0023	0.0089 ± 0.1023	0.0012 ± 0.0029	0.0284 ± 0.0029	0.00934 ± 0.0012	0.0132 ± 0.012	0.0277 ±0.100	0.0359 ±0.0080	0.0132 ± 0.0012	0.0792 ± 0.0010	0.1537 ± 0.0208

Table 3.23. Illustration of fovea and optic disc detection results of various pre-trained networks on IDRiD Dataset using mAP, AR evaluation metrics.

Models	Detection Boxes Precision				Detection Boxes Recall			
	mAP	**mAP@0.5 IoU**	**mAP@0.75 IoU**	**mAP (large)**	**AR @1**	**AR @10**	**AR @100**	**AR@100 (large)**
EfficientDet-D0	0.7221 ± 0.0076	0.9810 ± 0.0010	0.9353 ± 0.0019	0.7419 ± 0.121	0.8161 ± 0.0094	0.8121 ± 0.0191	0.8243 ± 0.0201	0.8254 ± 0.0092
Faster RCNN (ResNet-50)	0.7828 ± 0.02319	0.9532 ± 0.0167	0.9021 ± 0.0143	0.7292 ± 0.0132	0.8181 ± 0.0129	0.8914 ± 0.0132	0.8500 ± 0.0103	0.8510 ± 0.0190
SSD (MobileNetV1)	0.7822 ± 0.0101	0.9790 ± 0.0230	0.9289 ± 0.0012	0.7821 ± 0.0023	0.8241 ± 0.0102	0.8332 ± 0.0121	0.8212 ± 0.0121	0.8190 ± 0.0120

RetinaNet (ResNet50)	0.6912 ± 0.0012	0.9690 ± 0.0102	0.8791 ± 0.0561	0.7133 ± 0.0131	0.7123 ± 0.0012	0.7277 ± 0.0121	0.7278 ± 0.0121	0.7245 ± 0.0234
SSD (MobileNetV2)	0.7612 ± 0.0076	0.9756 ± 0.0129	0.9144 ± 0.0075	0.7624 ± 0.0034	0.8012 ± 0.0121	0.8004 ± 0.0103	0.8190 ± 0.0013	0.8037 ± 0.0612

Table 3.24 (a). Illustration of lesion segmentation results of various pre-trained networks on DDR Dataset using Dice Score evaluation metric.

Models	**Segmentation - Dice score**					
	Dicescore (Background)	**Dicescore (EX)**	**Dicescore (HA)**	**Dicescore (MA)**	**Dicescore (SE)**	**mDicescore**
PSPNet (w/ Focal Loss)	0.9910 ± 0.0001	0.0978 ± 0.0098	0.1319 ± 0.0138	0.0022 ± 0.0007	0.0324 ± 0.0028	0.0661 ± 0.0058
DeepLab v2 (w/ Focal Loss)	0.9901 ± 0.0002	0.0295 ± 0.0185	0.0575 ± 0.0403	0.0000 ± 0.0000	0.0078 ± 0.0123	0.0237 ± 0.0165
DeepLab v3 (w/ Focal Loss)	0.9912 ± 0.0002	0.1909 ± 0.0155	0.1569 ± 0.0255	0.0180 ± 0.0053	0.0323 ± 0.0053	0.0995 ± 0.0101
PSPNet (w/ Crossentropy Loss)	0.9908 ± 0.0003	0.0993 ± 0.0052	0.1260 ± 0.0063	0.0015 ± 0.0004	0.0265 ± 0.0032	0.0634 ± 0.0029
DeepLab v2 (w/ Crossentropy Loss)	0.9898 ± 0.0002	0.0307 ± 0.0194	0.0236 ± 0.0398	0.0000 ± 0.0000	0.0043 ± 0.0047	0.0146 ± 0.0134
DeepLab v3 (w/ Crossentropy Loss)	0.9912 ± 0.0001	0.1907 ± 0.0116	0.1699 ± 0.0256	0.0142 ± 0.0087	0.0314 ± 0.0070	0.1016 ± 0.0063

Table 3.24 (b). Illustration of lesion segmentation results of various pre-trained networks on DDR Dataset using Intersection over Union (IoU) evaluation metric.

Models	**Segmentation - Intersection over Union (IoU)**					
	IoU (Background)	**IoU (EX)**	**IoU (HA)**	**IoU (MA)**	**IoU (SE)**	**mIoU**
PSPNet (w/ Focal Loss)	0.9954 ± 0.0001	0.1582 ± 0.0147	0.2096 ± 0.0197	0.0040 ± 0.0014	0.0458 ± 0.0032	0.1044 ± 0.0084
DeepLab v2 (w/ Focal Loss)	0.9950 ± 0.0001	0.0502 ± 0.0313	0.0947 ± 0.0654	0.0000 ± 0.0000	0.0117 ± 0.0176	0.0391 ± 0.0265
DeepLab v3 (w/ Focal Loss)	0.9955 ± 0.0001	0.2886 ± 0.0203	0.2439 ± 0.0352	0.0312 ± 0.0095	0.0457 ± 0.0075	0.1524 ± 0.0136
PSPNet (w/ Crossentropy Loss)	0.9953 ± 0.0002	0.1599 ± 0.0077	0.2008 ± 0.0085	0.0029 ± 0.0008	0.0390 ± 0.0038	0.1007 ± 0.0040
DeepLab v2 (w/ Crossentropy Loss)	0.9948 ± 0.0001	0.0530 ± 0.0333	0.0393 ± 0.0650	0.0000 ± 0.0000	0.0071 ± 0.0078	0.0249 ± 0.0223

DeepLab v3 (w/ Crossentropy Loss)	0.9955 ± 0.0000	0.2887 ± 0.0149	0.2630 ± 0.0374	0.0250 ± 0.0147	0.0445 ± 0.0092	0.1553 ± 0.0082

Table 3.25 (a). Illustration of lesion, fovea and optic disc segmentation results of various pre-trained networks on IDRiD Dataset using Dice Score evaluation metric.

Models	**Segmentation - Dice score**						
	Dicescore (Background)	**Dicescore (Microaneurysms)**	**Dicescore (Haemorrhages)**	**Dicescore (Hard Exudates)**	**Dicescore (Soft Exudates)**	**Dicescore (Optic Disc)**	**mDicescore**
PSPNet (w/ Focal Loss)	0.9611 ± 0.0002	0.1921 ± 0.0160	0.3011 ± 0.0053	0.0038 ± 0.0018	0.1346 ± 0.0137	0.8857 ± 0.0073	0.3035 ± 0.0065
DeepLab v2 (w/ Focal Loss)	0.9502 ± 0.0018	0.0031 ± 0.0043	0.0287 ± 0.0406	0.0000 ± 0.0000	0.0000 ± 0.0000	0.8158 ± 0.0119	0.1695 ± 0.0049
DeepLab v3 (w/ Focal Loss)	0.9633 ± 0.0005	0.1213 ± 0.0351	0.3634 ± 0.0183	0.0298 ± 0.0270	0.0570 ± 0.0233	0.8986 ± 0.0044	0.2940 ± 0.0029
PSPNet (w/ Crossentropy Loss)	0.9616 ± 0.0013	0.1876 ± 0.0029	0.3026 ± 0.0019	0.0012 ± 0.0020	0.1154 ± 0.0001	0.8914 ± 0.0041	0.2997 ± 0.0101
DeepLab v2 (w/ Crossentropy Loss)	0.9509 ± 0.0011	0.0000 ± 0.0000	0.0898 ± 0.0418	0.0000 ± 0.0000	0.0000 ± 0.0000	0.8138 ± 0.0358	0.1807 ± 0.0140
DeepLab v3 (w/ Crossentropy Loss)	0.9631 ± 0.0012	0.1859 ± 0.0657	0.3460 ± 0.0276	0.0033 ± 0.0022	0.0595 ± 0.0056	0.8990 ± 0.0055	0.2988 ± 0.0184

Table 3.25 (b). Illustration of lesion, fovea, and optic disc segmentation results of various pre-trained networks on IDRiD Dataset using Intersection over Union (IoU) evaluation metric.

Models	**Segmentation - Intersection over Union (IoU)**						
	IoU (Background)	**IoU (Microaneurysms)**	**IoU (Haemorrhages)**	**IoU (Hard Exudates)**	**IoU (Soft Exudates)**	**IoU (Optic Disc)**	**mIoU**
PSPNet (w/ Focal Loss)	0.9800 ± 0.0001	0.3009 ± 0.0232	0.4457 ± 0.0069	0.0075 ± 0.0035	0.1919 ± 0.0152	0.9386 ± 0.0040	0.3769 ± 0.0074
DeepLab v2 (w/ Focal Loss)	0.9742 ± 0.0010	0.0057 ± 0.0080	0.0462 ± 0.0654	0.0000 ± 0.0000	0.0000 ± 0.0000	0.8957 ± 0.0058	0.1895 ± 0.0103
DeepLab v3 (w/ Focal Loss)	0.9811 ± 0.0002	0.1997 ± 0.0545	0.5140 ± 0.0200	0.0551 ± 0.0496	0.0830 ± 0.0381	0.9460 ± 0.0024	0.3596 ± 0.0056
PSPNet (w/ Crossentropy Loss)	0.9803 ± 0.0027	0.2940 ± 0.0017	0.4461 ± 0.0120	0.0024 ± 0.0012	0.1696 ± 0.0105	0.9423 ± 0.0011	0.3709 ± 0.0206

DeepLab v2 (w/ Crossentropy Loss)	0.9745 ± 0.0006	0.0000 ± 0.0000	0.1505 ± 0.0644	0.0000 ± 0.0000	0.0000 ± 0.0000	0.8938 ± 0.0238	0.2089 ± 0.0163
DeepLab v3 (w/ Crossentropy Loss)	0.9810 ± 0.0007	0.2908 ± 0.0929	0.4970 ± 0.0289	0.0064 ± 0.0041	0.0839 ± 0.0124	0.9459 ± 0.0030	0.3648 ± 0.0240

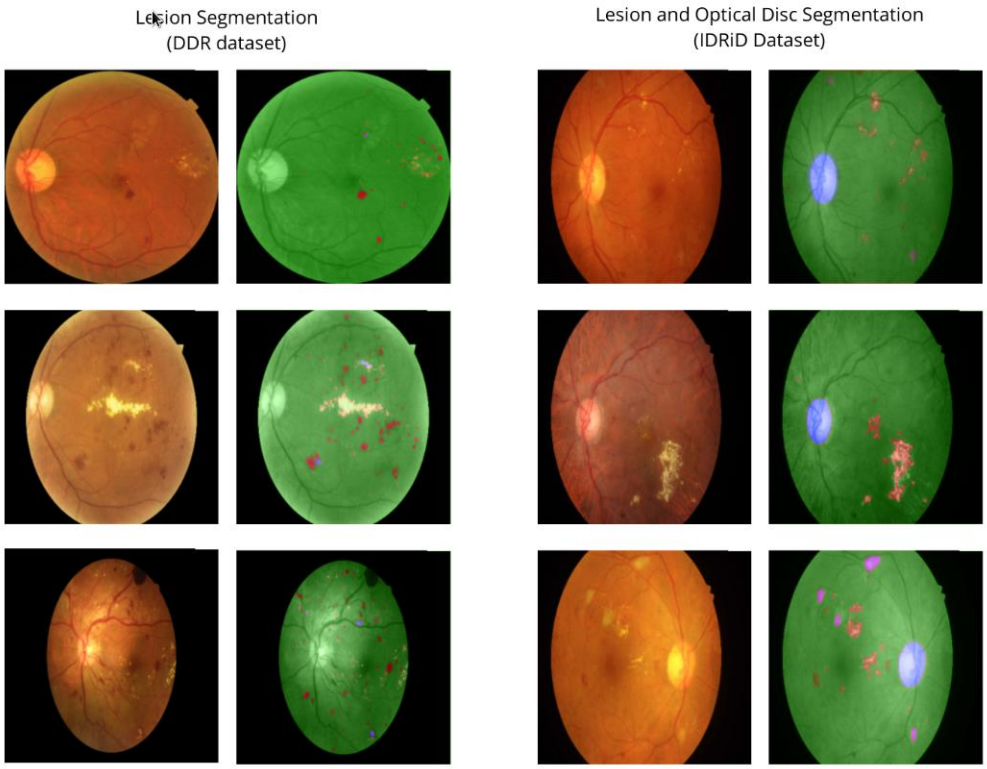


Figure 3.6. Results of segmentation model from PSPNet trained using focal loss for (i) DDR and (ii) IDRiD datasets.

3.3 Limitations

One significant limitation of BOVW is that it discards the spatial relationships between visual words within an image. By using BOVW, the original spatial arrangement of features within an image is lost during the quantization process. This means that important information about the relative positions and relationships

between visual elements is not captured. For example, the order, size, and orientation of objects in an image are not taken into account by BOVW. Further proposed work involves conducting a comprehensive evaluation to determine the appropriate deep learning model for classification, object detection, and segmentation. The aim is to ensure fair evaluation without biases when deploying deep learning models in real-world scenarios. The objective is to avoid failure cases and address the challenges encountered in practical applications considering the imbalanced dataset into account. However, a limitation of the proposed work is the need to extend the approach by considering the deployment of deep learning models in cloud-based environments for real-world applications. While developing a model for evaluation is valuable, it is equally important to address the practical implementation and usage of these models by doctors in real-world scenarios. Deploying deep learning models in cloud-based environments offers advantages such as scalability, accessibility, and ease of integration into existing healthcare systems. It is further necessary to create a customized model by incorporating the appropriate deep learning models that have been identified. By extending the proposed approach to include cloud-based deployment, doctors can take advantage of scalable infrastructure, remote accessibility, and seamless integration of deep learning models into their clinical workflow. This would enable effective utilization of the models in real-world applications, facilitating early detection and diagnosis of diseases like diabetic retinopathy, and ultimately improving patient outcomes.

Chapter 4
Implementation of Data Augmentation for Imbalanced Datasets in Computer Vision

This section[1] highlights the significance of data augmentation techniques in deep learning for imbalanced datasets. The chapter delves into two forms of minority data augmentation: Deep Convolutional Generative Adversarial Networks (DCGAN) and traditional image transformations such as rotation, flipping, shear and zoom. By employing these augmentation techniques, the aim is to improve model robustness and tackle class imbalance problems. To augment the minority class, a novel combination of techniques is proposed. DCGAN is used in the initial phase to generate synthetic samples for the minority class. Additionally, a modified VGG16 deep network architecture is employed to mitigate the effects of the imbalanced class problem. Another approach presented in this chapter also involves applying data augmentation exclusively to the minority class. This is achieved through transfer learning using pre-trained networks and subsequently classifying the data using a Weighted Support Vector Machine (SVM). Throughout both approaches, the primary focus is on the application of traditional data augmentation operations and synthetic generated samples specifically for the minority class.

: [1] The contents of this chapter are published in "Deep transfer with minority data augmentation for imbalanced breast cancer dataset." Applied Soft Computing 97 (2020): 106759. and "Data augmentation of minority class with transfer learning for classification of imbalanced breast cancer dataset using Inception-V3." In Iberian Conference on Pattern Recognition and Image Analysis, pp. 409-420. Springer, Cham, 2019.

4.1 Objective 3: Exploring Data Augmentation in deep learning for Imbalanced data

In real-world, problems might occur if images are taken under some specific limited set of conditions but it might happen that targeted applications may exist in varied different conditions such as various orientations, scales, brightness, location, etc. In order to resolve such challenges, we can train our neural network with synthetically modified data. The data augmentation in other words is a technique used to synthetically increase the number of sample images from already existing data. There exist two ways for performing augmentation. The first method is offline augmentation which is used to perform necessary transformations in order to increase the samples of the dataset. This approach is well-suited for small-scale datasets and another alternative approach is known as augmentation on the fly which is used to perform a random set of transformations on each mini-batch prior to providing the samples to any deep learning or machine learning models. This is performed on larger datasets, as with this method we can't expect there is an explosive increase in size but transformations are performed on mini-batches before feeding to the model. Various data augmentation techniques can be applied such as (1) Traditional affine transformations that are shifted, zoomed in /out, rotated, flipped, distorted, cropping, rescaling, or shaded with hue, etc. which artificially create or extend the dataset (Howard 2014). (2) Generative Adversarial Nets (GANs) can be applied to generate images of different styles in the dataset. Data Augmentation is generally used to reduce overfitting problems, where we can increase the amount of training data using the information in the training dataset (Howard 2014). Data Augmentation using CNN (Convolution Neural Networks) architecture is used to obtain transformed images

from the original images. Data augmentation techniques will help to train any machine learning or deep learning models to become more robust by seeing more synthetic created samples and also helps in resolving class imbalance issues. We emphasized on (a) applying the data augmentation technique to minority classes to overcome problems raised due to imbalanced data. (b) Data augmentation technique when applied to all the classes despite considering majority or minority proves to be a good regularization technique. (c) We have applied both offline and online, or augmentation on the fly augmentation technique over various deep learning models.

We have proposed a novel approach that involves a deep transfer network in collaboration with a Deep Convolutional Generative Adversarial network (DCGAN) (Radford, et al. 2015) as shown in Figure. 4.1. In the initial phase at the data level DCGAN technique as data augmentation is sought into the minority class to balance the minority class equivalent to the majority class by generating fake image samples. Data balanced at data level step is processed further to modified VGGIN-Net architecture consisting of block 4 pool layer of the VGG16 deep neural network combined with batch normalization, 2D convolutional (CONV2D) layer, Global Average Pooling 2D, dropout, and dense layers (Saini and Susan 2020). The block diagram for different layers is illustrated in Figure 4.2. Further in Figure. 4.3. Illustration of feature maps obtained by applying filters at the convolutional (CONV2D) layer after the VGG16 network of the proposed model is displayed. DCGAN network (Radford, et al. 2015) with the help of this architecture, trains quite well and generates a better quality of fake samples than the Generative Adversarial Network (GAN) which is only based upon fully connected neurons as displayed in

Figure 4.4. The addition of the Batch Normalization layer in the DCGAN network has significantly helped in training the network by normalizing the intermediate input values. DCGAN is applied only to the minority class to balance out the distribution samples of both the classes, to make the sample distribution of the minority class (Benign) equivalent to that of the majority class (Malignant), which will ultimately lead to an overall improvement in the performance of the proposed deep transfer network. Whereas in Figure. 4.5. activation map corresponding to the Benign and Malignant Images to illustrate the prominent features to detect cancer cells is shown.

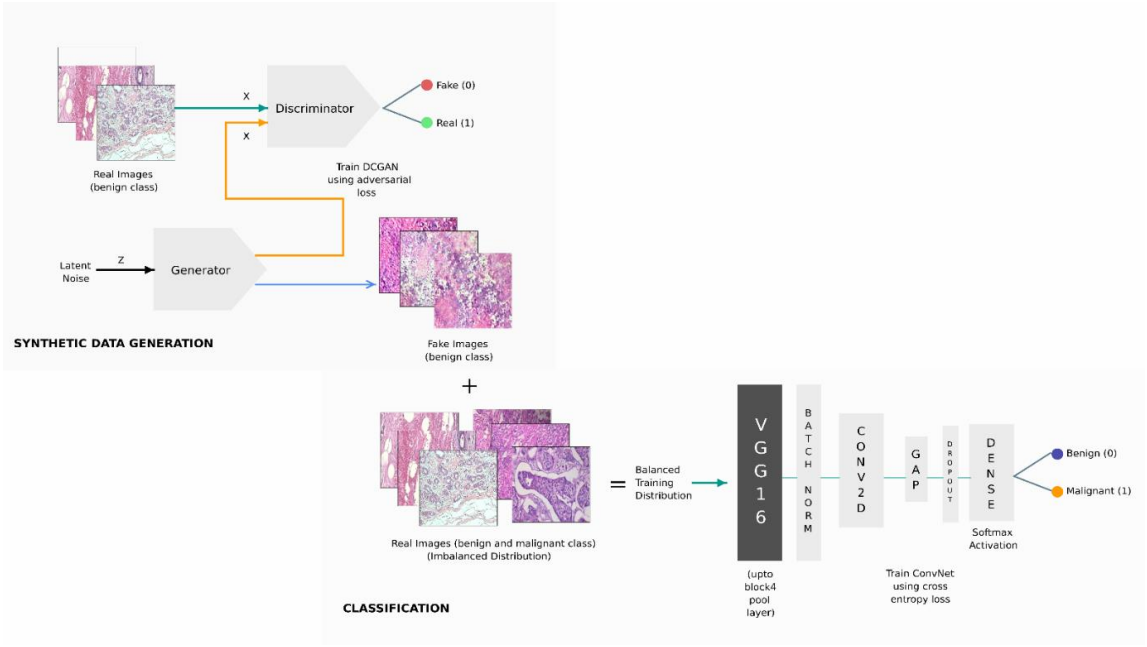


Figure 4.1. Proposed novel deep transfer network in collaboration with Deep Convolutional Generative Adversarial network (DCGAN).

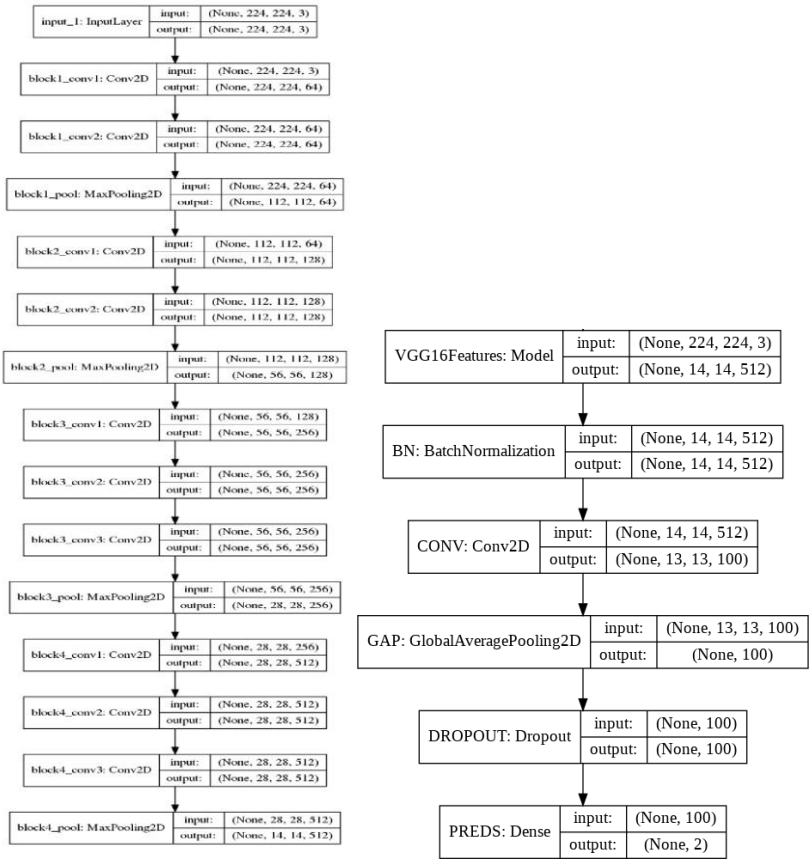


Figure 4.2. VGG16 architecture upto block4_pool layer and layers added after block4_pool layer for proposed network architecture.

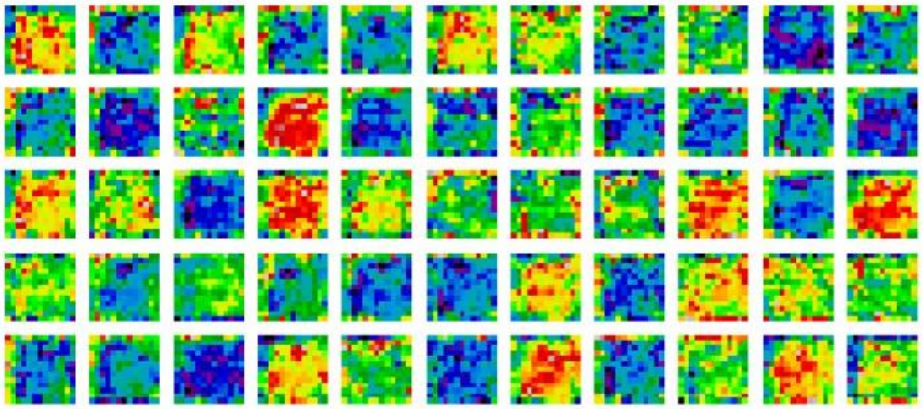


Figure 4.3. Illustration of feature maps obtained by applying filters at the convolutional (CONV2D) layer after VGG16 of the proposed model.

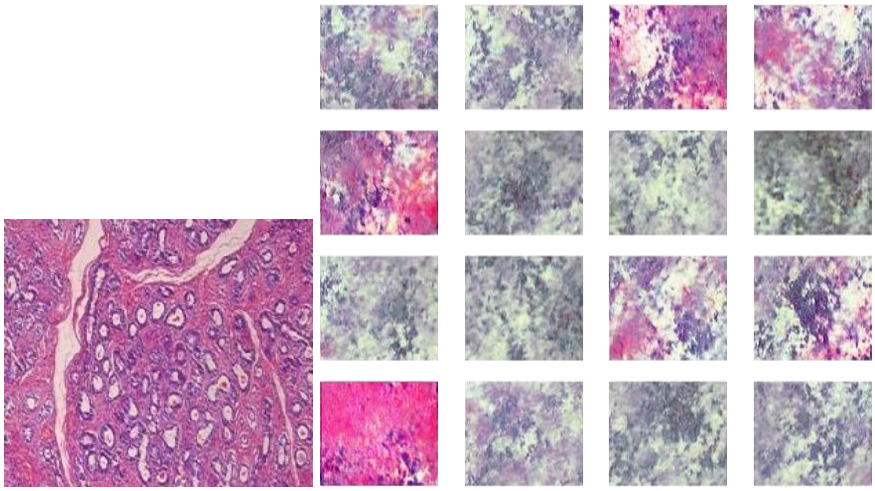


Figure 4.4. (a) Original Image sample for Benign class from BreakHis dataset (b) Fake images samples generated for Benign class using DCGAN.

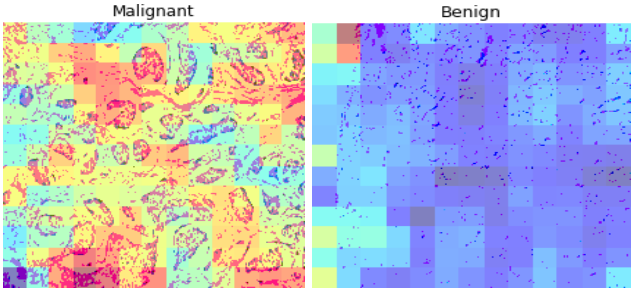


Figure 4.5. Activation map corresponding to the Benign and Malignant Images to illustrate the prominent features to detect cancer cells.

Table 4.1. Performance analysis of GAN and DCGAN based upon FID evaluation criteria.

Magnification factor	**No of Real Images**	**No of fake Images**	**FID of GAN**	**FID of DCGAN**
40X	522	718	386482.139	**401.592**
100X	544	793	165530.598	**480.946**
200X	523	767	122297.189	**465.874**
400X	488	644	143042.212	**517.281**

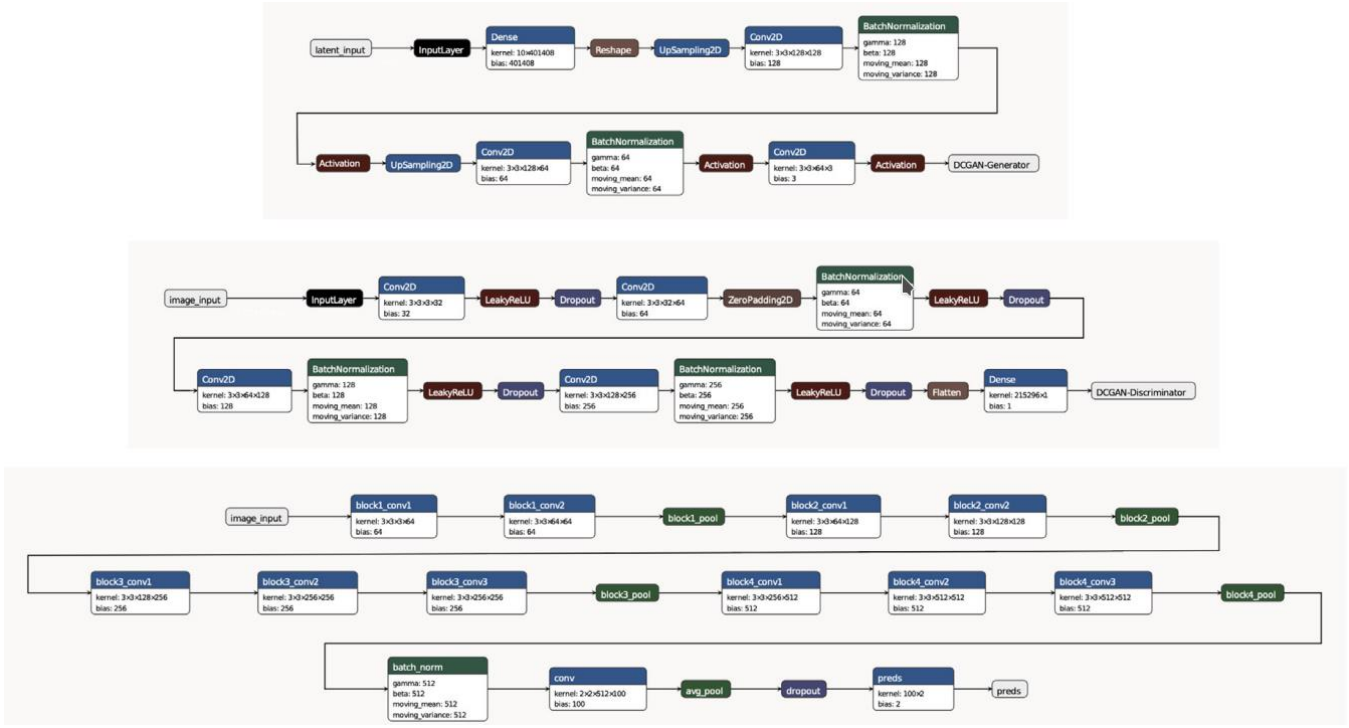


Figure 4.6. Proposed network architecture with VGG16 upto block4_pool layer along with Batch Normalization, Convolution 2D, Global Average Pooling, Dropout, and Dense layer.

The notable significance of this work is summarized as follows: (a) we have successfully constructed a novel deep transfer model using DCGAN as a data augmentation technique for cancer-related biomedical imbalanced datasets for various magnification factors: 40X, 100X, 200X, and 400X. (b) Mitigate the effect of covariant shift by using the Batch Normalization layer. (c) Proposed a novel deep network model using a transfer learning approach by transferring the knowledge from the source ImageNet object dataset to the target Breast cancer dataset effectively. (d) Analyzed the proposed transfer learning approach in comparison to other state-of-the-art networks for distinguishing Benign samples from Malignant in the case of an Imbalanced dataset.

Here a novel deep transfer learning approach is presented for the classification of the BreakHis dataset, specifically for the biomedical datasets. Further, DCGAN has been used for data augmentation of the minority class to resolve the class imbalance problem and overall boost the performance of the classification model. In the proposed approach, the Visual Geometry Group (VGG16) network is modified. We have used the VGG16 network in the proposed approach, as VGG16 can work well on the images of different scales as inspired by the work given in the original paper on VGG16. As in our work, images of different magnification factors were taken from the BreakHis dataset (40X, 100X, 200X and 400X), making it a suitable choice for dealing with microscopic images of varying magnification factors. An additional important factor that has been taken into consideration with respect to the VGG16 network would be that it uses smaller receptive fields in comparison to AlexNet and previous Convnets. In the proposed work, we have used configuration D as given in the original paper of VGG16, where 3x3 Convolution and 16 layers are taken into consideration. Moreover, configuration D which has been used for the proposed method is computationally less expensive than the VGG19 network and yet is able to achieve significant accuracy on large-scale image classification tasks. As in the case of VGG16, different convolutional layers make use of different receptive field sizes allowing it to learn multi-scale (even, object level, and as well as low level) information easily which will ultimately be useful for edge detection (Liu, et al. 2017). This makes the VGG16 network more suitable to work with images of different scales and varying aspect ratios. Hence, when dealing with histopathological images of different zoom factors VGG16 network is robust and more adept in comparison to other networks. The previous work (Perdana, et al. 2019, Kumar, et al. 2020, Baheti, et al. 2018, Aravind, et al. 2019) has inspired us to modify the VGG16 network for

our proposed methodology. Original VGG16 network architecture consists of a large number of parameters which results in more memory constraints and more floating-point operations. In this work, we have aimed to reduce certain shortcomings in the original VGG16 network architecture. As inspired by work done by Liu, et al. 2018, we have intended to minimize the number of parameters by replacing the flatten and three dense layers present in the original network architecture with the global average pooling (GAP) and one dense layer only. This replacement of layers has ultimately reduced the memory constraints and improved the time taken to train the network. The addition of one convolutional layer as used in our proposed approach is inspired by the concept given by Yosinshi, et al. 2014, that the top layer learns the feature specific to the targeted biomedical dataset. Whereas, on the other hand, the lower layers are intended for learning the more generic features. Further, the DCGAN approach has been used at the data level for generating fake images to balance the dataset so as to bring the minority class (having less number of samples) equivalent to the majority class thus, overcoming the imbalance class problem. To elaborate the proposed network, we have embedded the pre-trained layers (from the initial layer up to block 4 pool layer), from the lower level of the hierarchy and fused them with the newly learnt layers targeted for the biomedical classification task. Also, the original VGG16 pre-trained end-to-end architecture have convolutional layers along with pooling layers and fully connected layers from which, only the layers till block 4 have been taken; instead of considering all the layers of the pre-trained network to extract the useful bottleneck features. After that, all the layers of the original pre-trained VGG16 network are replaced by new layers which were selected in order to improve the performance of the model. Batch normalization (Ioffe, et al. 2015) layer has been added along with 2D convolution (CONV2D), Global Average Pooling 2D, dropout,

and dense layers as illustrated in Figure 4.6. Proposed network architecture with VGG16 upto block4 pool layer along with Batch Normalization, Convolution 2D, Global Average Pooling, Dropout, and Dense layer. Features extracted from the VGG16 network till block4 pool result in the features of dimensions 14 x 14, which will be more informative for the edge localization than the use of 7 x 7 features resulting from block 5 pool layer. Since we use an additional convolutional layer to learn the features specific to the targeted biomedical classification problem, we are required to choose a robust bottleneck end point from the pre-trained model. The use of block 4 instead of block 3 or even block 2 ensures that the number of trainable parameters in the added convolutional layer remains minimum, helping us reduce overall training time. Hence, bottleneck features extracted till block 4 pool layer works well as also proved by conducting experiments.

The batch normalization layer normalizes the data values to mitigate the effective covariant shift, whereas not applying the batch normalization will lead to biased results as observed in the results section by performing suitable experiments. In addition to that, a convolutional (CONV2D) layer is applied, based on the concept proposed by Yosinski, et al. 2014 that if both the source as well as target datasets are dissimilar, then features need to be extracted from the target-specific domain at the higher layer to improve the overall performance of the model. Here we have taken the source domain as the ImageNet dataset and considered the target domain as the breast cancer dataset. So based on the same concept, features are extracted from lower layers specific to the ImageNet dataset, and the features specific to the cancer domain are learned onto the higher layers by applying the convolutional (CONV2D) layer. After that Global Average Pooling 2D layers are used, instead of max pooling, because this

makes the model more robust to spatial translations in the data (Ioffe, et al. 2015) and receptive to higher intensities. A dropout layer is also added to reduce the overfitting problem and further, the dense layer is added towards the end of the proposed network to make the network along with the softmax activation function. As we have used a deep convolutional network (DCGAN) as a data augmentation technique to generate the fake images of the Minority class (Benign), DCGAN is used to generate the synthetic high-quality fake images from the available sample distribution of the training data. As DCGAN uses a stable architecture, it is able to learn good representation of hierarchical features more precisely, capturing higher details from images quickly. Also, the technique is in general more automatically adaptive in comparison to other traditional approaches. The DCGAN network with the help of this architecture trains quite well and generates a better quality of fake samples than the Generative Adversarial Network (GAN) which is only based upon fully connected neurons. Fake images generated using GAN are generally of low quality and possess a higher noise ratio. However, there are some GAN models that produce images that are of lower quality and have higher noise ratios. One of the studies presented in "Detecting and Simulating Artifacts in GAN Fake Images" by Zhang et al. (2019) shows that GAN-generated fake images can have higher noise ratios than real images. This means that GAN models may not be able to accurately model all of the features of real images, which can lead to the introduction of noise.

The addition of the Batch Normalization layer in the DCGAN network has significantly helped in training the network by normalizing the intermediate input values. DCGAN approach is applied only to the minority class in order to balance out the distribution samples of both the classes, to make the sample distribution of the

minority class (Benign) equivalent to that of the majority class (Malignant), which will ultimately lead to an overall improvement in the performance of the proposed deep transfer network.

Goodfellow, et al. (2014) proposed the Generative adversarial networks (GAN), using two important components: generator (G) and discriminator (D) modules (Goodfellow, et al. 2014). While the generator generates fake images, the role of the discriminator is to detect whether the fake images generated are actually fake images or real. In GAN, the generator and discriminator are rapidly competing with each other by following the min-max strategy as represented below in equation 4.1.

$$min_G max_D\, V(D,G) = \quad E_{x\sim p_{data}(x)}[logD(x)] + E_{z\sim\, p_{z\,(z)}}\big[log\,(1-D\big(G(z)\big))\big] \qquad (4.1)$$

In equation (1), D is denoted as discriminator and G is represented as a generator; whereas p denotes the sample and z represents the noise components. Radford, et al. (2015), further proposed a variant of GAN known as DCGAN based upon the same concept of generator and discriminator by using deep convolutional neural networks for generator and discriminator. Using the same concept of DCGAN, the fake images are generated for Benign (minority class), in our work, to overcome the class imbalance problem. The generator used in the experiment uses the input vector of 100 as random noise distribution (a normal distribution with zero mean and

unit variance) and the output of the generator is of size 224 x 224 x 3. The discriminator will take input of size 224 x 224 x 3 and the output of the discriminator will be a single value of either 0 or 1 (based upon whether the image is detected as fake or real).

The network architecture of the generator and the discriminator used in DCGAN includes batch normalization layers that have been able to normalize the values during gradient propagation as well as a forward pass. Further, this has been observed to be effective in tackling the class imbalance problem and to boost the performance of the classification model. However, the same is lacking in the GAN (Radford, et al. 2015) approach. So, we have used the DCGAN technique instead of GAN as the data augmentation to synthetically generate the fake images with the proposed approach. Experiments have been conducted as illustrated in Table 5.1 using Fréchet Inception Distance (FID) evaluation criteria for comparing GAN and DCGAN. The evaluation criteria FID is defined as follows in equation 4.2:

$$FID = (\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}) \tag{4.2}$$

Where μ_r is the mean of feature vector calculated from the real images, μ_g is the mean of the feature vector calculated from the fake images, Σ_r is the covariance of the feature vector calculated from the real images, Σ_g is the covariance of the feature vector calculated from the fake images. Whereas the feature vectors of 2048 Dimensional are calculated using a pre-trained Inception-V3 network. FID is used to calculate the quality of images generated and lower scores indicate that the generated

image correlates well with the high-quality real images. Mathematically Fréchet distance is calculation of the measure similarity between curves on the basis of location and ordering of points. While calculating FID, we make use of the feature vector of both real and fake images.

The original imbalanced distribution from where the images have been sampled to train the various classifiers to have an imbalance factor of 2.30, 2.45, 2.46, and 2.32 respectively in 40X, 100X, 200X, and 400X different Magnification factors. After augmenting image synthesis by DCGAN to the train datasets, the imbalanced ratio obtained is 1:1 in the case of both Malignant and Benign classes of all Magnification factors. As the pre-trained VGG16 network takes the use of the Keras v2.2.3 deep learning framework with TensorFlow v1.15.3 backend.

Additionally, we have used a support vector classifier (SVM) from Scikit-learn (Kramer 2016), which has also been used to gather evaluation metrics. For training all of the deep learning-based classifiers we make use of the standard cross entropy loss (log-likelihood) without any form of L2 regularization. In the case of DCGAN, the ReLU activation function is applied along with the Tanh activation function for the generator and the LeakyReLU activation function is used for all the layers of the discriminator, along with that, the sigmoid activation function is there in the last layer of the discriminator. In training all the deep learning networks, Stochastic Gradient Descent (SGD) with momentum optimizer has been used to bring stability to the training. In the experiments related to the proposed work, we used a learning rate schedule with cosine decay restarts as per SGDR. Experiments with Stochastic gradient descent with momentum (using SGDR learning rate schedule)

optimizer have been performed which also seems to have overall improved the consistency of the learning curve for the last few sets of epochs. According to the input image of size 224 x 224, all the images including original and fake are presented to the network in the specific size of 224 x 224.

We have conducted all our experiments on a GCP (Google Cloud Platform) virtual machine with 8 GB RAM, a dual-core Intel Xeon processor, and a single NVIDIA Tesla K80 GPU that has 12 GB DDR5 virtual memory. For all deep learning-related model training, Loshchilov, et al. 2016, in the case of an SGDR optimizer, it is recommended to start with a smaller cycle weight and increase the cycle steps linearly when scaling up. As SGDR performs warm restarts after cyclic intervals, ensuring that a higher learning rate would always be able to push the gradient in the direction away from the local optima. SGD approach uses cosine decay annealing to decay the learning rate after interval epochs and performs a warm restart after every iteration cycle. Hence, we performed all our deep learning experiments starting with a default value of T_0= 1, T_{mul}=2, for the cosine decay learning rate schedule. As per the experiments performed in the SGDR paper, a learning rate of 0.05 adapts well to the CIFAR-10 dataset. Initially, for all of the deep learning experiments, we used the initial learning experiments by setting the value to 0.05 which is tuned later to further improve the performance of our approach using the grid search strategy. The training times for DCGAN and modified VGG16 are within a range, but it varies depending on the specific setup and conditions. Training a DCGAN (Deep Convolutional Generative Adversarial Network) to generate synthetic samples would require an approximate duration of 14 hours. The DCGANs involve training both the generator and discriminator networks iteratively, which can require

numerous epochs to converge adequately. On the other hand, training a modified VGG16 deep neural network after applying synthetic samples at data level on minority class takes less time compared to generative models like DCGANs. With the reduced parameters it contributes to faster training times. It took approximately 1 hour 30 minutes to train the modified VGG16. The training times can vary significantly based on the hardware used. Powerful GPUs or TPUs can significantly accelerate the training process, while using CPUs might result in longer training times. Additionally, factors such as the batch size, learning rate schedule, data augmentation techniques, and convergence criteria can also have an effect on the overall training time. All our models are trained with the net budget of 100 epochs with the exception of the ResNet-50, Inception-V3, and VGG16 models which were trained till 50 epochs as they couldn't converge significantly even after 50 epochs. The choice of SGDR optimizer for our experiments helps us to determine a recommendation even for various hyperparameters without the need for a separate validation dataset as emphasized in the original SGDR work (Loschilov, et al. 2016). Whereas in the case of the state-of-the-art networks used for comparisons, the following implementation of the network was used while conducting the experiment. The conventional Bag of visual words (BOVW) (Suh, et al. 2018) is used for the comparison, SIFT (scale-invariant feature transform) features are used along with K-Means clustering (number of clusters set to 10) and SVM classifier (kernel used in the classifier is 'linear', C is set as 1.0 and gamma is set as '1/number of features). For the conventional CNN used as the comparison method, a five layer architecture is used which consists of repeated blocks of Convolution with Batch normalization and ReLU activation and pooling layers along with two fully connected layers. In addition, dropout with the drop rate of 0.2 has been applied after the first fully connected layers. For all of our

experiments, we use the Image Data Generator from the Keras framework with a batch size set to 128. In the case of pre-trained convolutional neural networks, Inception-V3, VGG16, and ResNet, we use fixed pre-trained features and the terminal dense layer (classifier) was trained. The experiments conducted related to deep features extracted from pre-trained networks were done in conjugation with SVM classifiers. Whereas in the case of the SVM classifier (Cortes and Vapnik 1995), linear and RBF kernels both were used while conducting the experiments.

From experimental results, it was found that the proposed deep transfer learning-based network using DCGAN as a data augmentation approach, works significantly well in comparison to other conventional approaches in the case of 40X, 100X, 200X, and 400X Magnification factors as illustrated in the results compiled in Table 4.2(i), 4.2(ii). Further compared the proposed approach with the other state-of-the-art models: (i) Bag of visual words (BOVW) (ii) CNN network trained from scratch (iii) pre-trained models (ResNet-50, Inception-V3, and VGG16) (iv) deep feature extracted from pre-trained models and trained via machine learning models. Various deep features and classifier combinations tested are: Inception-V3 + SVM (with RBF and linear kernel); VGG16 + SVM (with RBF and linear kernel) and ResNet-50 +SVM (with RBF and linear kernel).The evaluation was done on the basis of various evaluation criteria: Accuracy, F1 score, Mathews correlation, Cohen kappa, and ROC with AUC. The pre-trained networks (Inception-V3, ResNet, and VGG16) are not working effectively in comparison to other conventional approaches, since the results are biased towards either the majority or minority class. Features extracted from the pre-trained network: Inception-V3, VGG16, and ResNet-50, when given to the SVM classifier (having kernel value set to ‘rbf’ and ‘linear’) have shown

improvement in the performance of the result in comparison to the pre-trained networks. The BOVW technique is also able to detect the minority class and majority class separately but still they are not able to show significant performance. CNN trained from scratch are also working well in comparison to the pre-trained networks but they have not shown remarkable performance in comparison to the proposed network in conjunction with the SVM classifier in case of 40X and 200X magnification factors. But in the case of 100X and 400X magnification factors, CNN has shown improvement over a few deep features and classifier combinations as shown in Table 4.2 (i) and (ii).

For our experiments, we make use of the mean normalization technique which helps to rescale the intensity of each pixel and standardize them to a zero mean and unit variance normal distribution before it is input to the network. This preprocessing step was particularly useful when the CNN is trained from scratch and for it, we have used RGB normalization based on the BREAKHIS dataset. We have also found that VGG16 and ResNet50 networks make use of the same preprocessing technique with the exception that has mean and standard deviation from the ImageNet dataset as we use these networks with pre-trained weights. The Inception-style preprocessing aims to rescale image pixels to a range [-1, 1] without performing any standardization of the RGB values. As an additional step, we also tried to use the Inception-style preprocessing technique for CNN trained from scratch but with it, results were seen to reduce significantly from which we decided to move forward with the mean normalization step for all forms of pre-processing. Experiments reflect that for this imbalanced breast cancer dataset the mean normalization step is better suited than rescaling (Inception-style [-1, +1] feature scaling) due to improved generalization

with RGB mean subtraction. Our proposed approach is based on a pre-trained VGG16 network which also uses RGB normalization based preprocessing.

Our proposed deep transfer network developed by considering DCGAN and Batch normalization with VGG16 layers till block 4 pool layer along with the Convolutional layer, Batch normalization, Global Average Pooling 2D, dropout, and dense layers is working well in comparison to other approaches. Before selecting the appropriate combination, we performed experiments with the different combinations to analyze the effect of DCGAN synthetic sample generation and Batch normalization on the proposed transfer network. Initially, the combination of layers was tried (i) without batch normalization (iii) secondly without batch normalization and with the DCGAN samples, it was found that the results were biased towards the minority or majority class. But after applying batch normalization with the DCGAN samples leads to an overall improvement in the performance of the network as shown in Tables 4.2 (i) and (ii). The effect of the DCGAN and the effect of covariance shift by use of the batch normalization layer to the proposed deep transfer network has been analyzed experimentally. Another set of experiments is performed to find the appropriate deep features for the proposed approach. An experimental study is performed to extract the appropriate features from VGG16 by analyzing and comparing the deep features from different blocks of the VGG16 network as shown in Table 4.3 (i) and (ii) for different magnification factors. From the analysis, it was found that extraction of layers till block 4 layer of VGG16 architecture is the appropriate choice to be used in the proposed network in comparison to other layers. The proposed deep transfer network incorporates initial layers till block 4 pool layer of VGG16 is deriving the relevant features. Further, the tuning of hyperparameters experiments is conducted on the

proposed approach using a grid search-based strategy in order to find the best hyperparameters combination by changing the initial learning rate, $T_{mul,}$ and T_0 value of the SGDR optimizer as shown in the case of 40X, 100X, 200x and 400x magnification factors. Whereas in Table 4.4. (i), (ii), (iii), and (iv), performance evaluation of the proposed approach with different tested combinations is performed experimentally as shown for 40X, 100X, 200X, and 400X magnification factors to find the best combination of hyperparameters. From the analysis, it was found that the proposed approach is more stable in comparison to other approaches using SGDR optimizer, which further leads to improved convergence. Performance evaluation of the proposed approach with the help of ROC (Receiver operating curve) with AUC depicts the comparison of the proposed approach with the other approaches. This proves that the proposed approach is the best choice in comparison to the other conventional approaches. As shown in Figure 4.7. Learning curve depicting test accuracy across epochs and demonstrating better anytime performance with proposed approach in comparison to other approaches for various magnification factors of BreakHis dataset and Figure 4.8. Depicts the Comparison of the proposed approach with other approaches using ROC (Receiver Operating Curve) for various magnification factors of the BreakHis dataset.

The performance improvement of our proposed approach is primarily based on the use of DCGAN for synthetic Malignant sample generation. We conducted experiments with GAN and DCGAN initially. Based upon Fréchet Inception Distance (FID) evaluation criteria as illustrated in Table 4.1, the observation drawn from the analysis is that DCGAN produces higher quality samples in comparison to GAN. Fréchet Inception Distance (FID) score is the metric that has been used for the

performance analysis of generative modeling. FID is used to calculate the difference between Inception features extracted from real images and fake images generated using Generative Adversarial Networks and their variants. It is a useful metric indicator for quality in noise analysis of synthetic samples where a lower FID usually indicates a better generation of fake samples. It is evident from our analysis, that the addition of convolutional layers to generative networks improves the quality of synthetic sample generation as seen in the case of DCGAN over basic GAN.

Results obtained from experiments infer that the use of transfer learned weights on the same architecture of previously trained available pre-trained networks (which are already trained on large scale ImageNet dataset) helps improve the classification task. The use of transfer learning is also appropriate in this use case as we have a comparatively much less number of samples present in the available biomedical image dataset. Training of CNN from scratch would require a large set of samples and it has to ensure that the large number of parameters of the CNN be trained effectively with many generalizations in order to yield good results. So we combined the popular pre-trained network as per architecture cited in the literature and one convolutional layer of basic CNN architecture. The lower layers will learn the more generic features as in our case the source and target dataset belong to a dissimilar domain. The DCGAN fake image generation approach along with modified VGG16 network by replacing the flatten and three dense layers present in the original network architecture with the global average pooling (GAP) and one dense layer has been proposed in this work. This replacement of layers has resulted in a minimization of the number of parameters. It was observed that the GAN approach separately on the minority class to generate fake images works quite well to deal with the imbalance

situation. It is also experimentally observed that when the batch normalization layer is not included in the proposed network, the results get completely biased towards the majority class. The batch normalization layer scales intermediate input values of the VGG16 block 4 pool features resulting in effective training of the classifier. Without it, the classifier suffers from the exploding gradient problem and we incur NaN loss during training due to quite a high log-likelihood value with the imbalanced class setting.

Even using the SGDR optimizer has made the proposed network and other comparison approaches more stable as shown experimentally. Also, it helps to generalize well allowing faster convergence. An experimental task has been performed by using the BreakHis because it consists of images of different magnification factors and also consists of adequate image samples and is one of the few available datasets related to breast cancer. Further extensive hyperparameter tuning has been conducted to find the optimum hyperparameters as shown in Table 4.5. From the analysis, it was found that the choice of the exact hyperparameters plays a crucial role in the overall learning and performance of the model. Using our proposed network, the best determining hyperparameters lead to better generalization. The hyperparameter tuning of the proposed network has been performed using a simple grid search strategy and we choose test accuracy as the maximization objective of this search step. Due to the use of the SGDR learning rate schedule, as per recommendation of Loschilov, et al. 2017 an extra validation would not be required in this step. In this study, we have also compared and analyzed the features extracted from the different VGG16 blocks for our proposed network in order to find relevant features. Results show that the features extracted by the block4 pool prove to be quite

suitable. So we have used features till block 4 pool layers only in the proposed work instead of till block 5 as there in the original architecture of VGG16. Features extracted till block 4 pool layers are considered bottleneck endpoint and hence, ideal. So, 14 x 14 x 512 features extracted from the VGG16 network as justified experimentally are used in the proposed work.

To conclude we have proposed a deep transfer learning based novel approach for the classification of an Imbalanced breast cancer dataset. In this work, we have also explored the effect of DCGAN and the effect of Batch normalization on the proposed transfer network architecture. The proposed network contains the VGG16 pre-trained model layers till block 4 pool layers along with the Batch Normalization, convolutional (CONV2D), Global Average Pooling 2D layer, dropout, and dense layers. It would help in the accurate detection of cancer cells at an early stage. The limitation of the proposed work is that applying DCGAN at the data level is directly dependent on the number of samples of minority classes as we employ deep generative modeling for enhancing the performance of the classifier. For scenarios with a much less number of minority samples, the DCGAN training distribution would not be able to generalize well and would fail to generate high-quality samples resulting in suboptimal performance. Even considering the limitations also, the proposed methodology is able to tackle the class imbalance problem which is faced by many state-of-the-art deep learning networks as well as other traditional machine learning techniques. The proposed architecture is able to learn fine-grained features on top of ImageNet pre-trained deep features specific to the biomedical datasets. A total number of network parameters is reduced by an order of magnitude by introducing the global average pooling layer instead of flatten in the original VGG16

architecture. This also helps to reduce the number of FLOPS. The proposed approach is able to work well even when the microscopic images are very different at the data level as the BreakHis dataset consists of four different magnification factors. Our focus was on the combination of a pre-trained network with CNN. We have included the VGG16 pre-trained network because of its simplicity and as it can be considered a popular pre-trained network with its ability to work well with images of different scales and aspect ratios.

Table 4.2. (i) Performance evaluation on 40X and 100X Magnification factors.

Magnification Factor	**40X**					**100X**				
		F1 score		**Matthews Correlation Coefficient**	**Cohen's Kappa**		**F1 score**		**Matthews Correlation Coefficient**	**Cohen's Kappa**
Approach	**Accuracy**	**Benign**	**Malignant**			**Accuracy**	**Benign**	**Malignant**		
BOVW	0.65	0.49	0.73	0.3904	0.3	0.615	0.4	0.72	0.3266	0.2299
CNN	0.67	0.51	0.75	0.4525	0.3399	0.88	0.87	0.89	0.7655	0.76
VGG16	0.5	0.0	0.67	0.0	0.0	0.5	0.67	0.00	0.0	0.0
InceptionV3	0.56	0.21	0.69	0.2526	0.12	0.50	0.06	0.66	0.0	0.0
ResNet50	0.53	0.11	0.68	0.1758	0.06	0.51	0.06	0.67	0.02	0.0714
VGG16 + Linear SVM	0.855	0.84	0.87	0.7295	0.71	0.87	0.86	0.88	0.7473	0.74
InceptionV3 + Linear SVM	0.86	0.85	0.87	0.7319	0.72	0.815	0.80	0.83	0.6353	0.63
ResNet50 + Linear SVM	0.91	0.91	0.91	0.8226	0.82	0.91	0.91	0.91	0.8241	0.82
VGG16 + RBF SVM	0.85	0.83	0.87	0.7249	0.7	0.855	0.84	0.87	0.7231	0.71
InceptionV3 + RBF SVM	0.82	0.79	0.84	0.6627	0.64	0.795	0.75	0.83	0.6298	0.59

ResNet50 + RBF SVM	0.89	0.88	0.90	0.7901	0.78	0.86	0.85	0.87	0.7319	0.72
Proposed Network (w/o Batch Normalization)	0.5	0.67	0.0	0.0	0.0	0.5	0.67	0.0	0.0	0.0
Proposed Network (w/o Batch Normalization and w/ DCGAN samples)	0.5	0.67	0.0	0.0	0.0	0.5	0.67	0.0	0.0	0.0
Proposed Network (w/ BatchNormalization and w/ DCGAN samples)	0.935	0.93	0.94	0.8774	0.87	0.925	0.92	0.93	0.8551	0.85
Proposed Network (w/ BatchNormalization, w/ DCGAN samples and w/ hyper parameter tuning)	0.965	0.96	0.97	0.9304	0.9299	0.94	0.94	0.94	0.8815	0.88

Table 4.2. (ii) Performance evaluation on 200X and 400X Magnification factors.

Magnification Factor	**200X**					**400X**				
		F1 score		**Matthews Correlation Coefficient**	**Cohen's Kappa**		**F1 score**		**Matthews Correlation Coefficient**	**Cohen's Kappa**
Approach	**Accuracy**	**Benign**	**Malignant**			**Accuracy**	Benign	Malignant		
BOVW	0.56	0.24	0.69	0.2211	0.12	0.575	0.35	0.68	0.2072	0.15
CNN	0.775	0.72	0.81	0.5972	0.55	0.885	0.88	0.89	0.7731	0.77
VGG16	0.5	0.67	0.0	0.0	0.0	0.5	0.0	0.67	0.0	0.0

InceptionV3	0.51	0.08	0.67	0.0586	0.02	0.5	0.0	0.67	0.0	0.0
ResNet 50	0.5	0.0	0.67	0.0	0.0	0.5	0.0	0.67	0.0	0.0
VGG16 + Linear SVM	0.835	0.82	0.85	0.6852	0.6699	0.835	0.81	0.85	0.6958	0.6699
InceptionV3 + Linear SVM	0.795	0.77	0.81	0.6034	0.59	0.82	0.80	0.84	0.656	0.64
ResNet 50 + Linear SVM	0.93	0.93	0.93	0.86	0.8627	0.915	0.91	0.92	0.835	0.83
VGG16 + RBF SVM	0.84	0.81	0.86	0.7128	0.6799	0.82	0.78	0.85	0.6805	0.64
InceptionV3 + RBF SVM	0.76	0.7	0.8	0.5729	0.52	0.795	0.75	0.83	0.6298	0.59
ResNet 50 + RBF SVM	0.86	0.84	0.88	0.75	0.72	0.885	0.87	0.89	0.7813	0.77
Proposed Network (w/o Batch Normalization)	0.5	0.67	0.0	0.0	0.0	0.5	0.67	0.0	0.0	0.0
Proposed Network (w/o Batch Normalization and w/ DCGAN samples)	0.5	0.67	0.0	0.0	0.0	0.5	0.67	0.00	0.0	0.0
Proposed Network (w/ BatchNormalization and w/ DCGAN	0.955	0.95	0.96	0.91	0.91	0.915	0.91	0.92	0.832	0.83

samples)										
Proposed Network (w/ BatchNormalization, w/ DCGAN samples and w/ hyper parameter tuning)	0.955	0.95	0.96	0.9111	0.91	0.93	0.93	0.93	0.8627	0.86

Table 4.3. (i) Performance evaluation of proposed VGGIN-Net architecture with different VGG16 blocks as the backbone for 40x and 100x magnification factors.

Magnification Factor	**40X**					**100X**				
Approach	**Accuracy**	**F1 score**		**Matthews Correlation Coefficient**	**Cohen's Kappa**	**Accuracy**	**F1 score**		**Matthews Correlation Coefficient**	**Cohen's Kappa**
		Benign	**Malignant**				**Benign**	**Malignant**		
Proposed Network (using block3_pool layer)	0.895	0.89	0.90	0.799	0.79	0.91	0.90	0.92	0.8307	0.82
Proposed Network (using block4_pool layer)	**0.935**	**0.93**	**0.94**	**0.8774**	**0.87**	**0.925**	**0.92**	**0.93**	**0.8551**	**0.85**
Proposed Network (using block5_pool layer)	0.855	0.84	0.87	0.7231	0.71	0.87	0.86	0.88	0.7552	0.74

Table 4.3. (ii) Performance evaluation of proposed VGGIN-Net architecture with different VGG16 block as the backbone for 200x and 400x magnification factors.

Magnification Factor	**200X**				**400X**			
		F1 score				**F1 score**		

Approach	Accuracy	Benign	Malignant	Matthews Correlation Coefficient	Cohen's Kappa	Accuracy	Benign	Malignant	Matthews Correlation Coefficient	Cohen's Kappa
Proposed Network (using block3_pool layer)	0.93	0.93	0.93	0.8643	0.86	0.935	0.93	0.94	0.8753	0.87
Proposed Network (using block4_pool layer)	**0.955**	**0.95**	**0.96**	**0.91**	**0.91**	**0.915**	**0.91**	**0.92**	**0.832**	**0.83**
Proposed Network (using block5_pool layer)	0.865	0.85	0.88	0.7435	0.73	0.855	0.84	0.86	0.716	0.71

Table 4.4. (i) Results for hyper-parameter tuning of SGDR optimizer with proposed CNN for 40x magnification factor of BreakHis dataset.

40X				
Hyperparameters	**Evaluation Metric**	**SGDR (T_0=1, T_{mul}=2)**	**SGDR (T_0=10, T_{mul}=2)**	**SGDR (T_0=50, T_{mul}=1)**
LR=0.001	**Accuracy**	0.91	0.905	0.91
	Matthews Correlation Coefficient	0.8281	0.8192	0.8281
	Cohen's Kappa	0.82	0.81	0.82
LR=0.005	**Accuracy**	0.945	0.935	0.94
	Matthews Correlation Coefficient	0.8921	0.8735	0.8828
	Cohen's Kappa	0.89	0.87	0.88
LR=0.01	**Accuracy**	**0.965**	0.95	0.955
	Matthews Correlation Coefficient	**0.9304**	0.9007	0.9111
	Cohen's Kappa	**0.9299**	0.9	0.91
LR=0.05	**Accuracy**	0.935	0.955	0.96
	Matthews Correlation Coefficient	0.8774	0.91	0.9201
	Cohen's Kappa	0.87	0.91	0.92
LR=0.1	**Accuracy**	0.96	0.95	0.96

	Matthews Correlation Coefficient	0.92	0.9028	0.9201
	Cohen's Kappa	0.92	0.9	0.92

Table 4.4. (ii) Results for hyper-parameter tuning of SGDR optimizer with proposed CNN for 100x magnification factor of BreakHis dataset.

100X				
Hyperparameters	**Evaluation Metric**	**SGDR (T_0=1, T_{mul}=2)**	**SGDR (T_0=10, T_{mul}=2)**	**SGDR (T_0=50, T_{mul}=1)**
LR=0.001	**Accuracy**	0.935	0.935	0.93
	Matthews Correlation Coefficient	0.8753	0.8735	0.8643
	Cohen's Kappa	0.87	0.87	0.86
LR=0.005	**Accuracy**	0.935	0.935	0.93
	Matthews Correlation Coefficient	0.8721	0.8735	0.8627
	Cohen's Kappa	0.87	0.87	0.86
LR=0.01	**Accuracy**	0.925	0.93	0.93
	Matthews Correlation Coefficient	0.8551	0.8643	0.8627
	Cohen's Kappa	0.85	0.86	0.86
LR=0.05	**Accuracy**	0.93	0.94	0.925
	Matthews Correlation Coefficient	0.8627	0.8815	0.8534
	Cohen's Kappa	0.86	0.88	0.85
LR=0.1	**Accuracy**	0.93	**0.94**	0.925
	Matthews Correlation Coefficient	0.8627	**0.8815**	0.8534
	Cohen's Kappa	0.86	**0.88**	0.85

Table 4.4. (iii) Results for hyper-parameter tuning of SGDR optimizer with proposed CNN for 200x magnification factor of BreakHis dataset.

200X				
Hyperparameters	**Evaluation Metric**	**SGDR (T_0=1, T_{mul}=2)**	**SGDR (T_0=10, T_{mul}=2)**	**SGDR (T_0=50, T_{mul}=1)**
LR=0.001	**Accuracy**	0.92	0.93	0.92

	Matthews Correlation Coefficient	0.8442	0.8662	0.8461
	Cohen's Kappa	0.84	0.86	0.84
LR=0.005	**Accuracy**	0.955	**0.955**	0.945
	Matthews Correlation Coefficient	0.9104	**0.9111**	0.8921
	Cohen's Kappa	0.91	**0.91**	0.89
LR=0.01	**Accuracy**	0.945	0.94	0.955
	Matthews Correlation Coefficient	0.8936	0.8815	0.9111
	Cohen's Kappa	0.89	0.88	0.91
LR=0.05	**Accuracy**	0.955	0.935	0.955
	Matthews Correlation Coefficient	0.91	0.8735	0.9111
	Cohen's Kappa	0.91	0.87	0.91
LR=0.1	**Accuracy**	0.895	0.95	0.95
	Matthews Correlation Coefficient	0.8046	0.9016	0.9007
	Cohen's Kappa	0.79	0.9	0.9

Table 4.4. (iv) Results for hyper-parameter tuning of SGDR optimizer with proposed CNN for 400x magnification factor of BreakHis dataset.

400X				
Hyperparameters	**Evaluation Metric**	**SGDR (T_0=1, T_{mul}=2)**	**SGDR (T_0=10, T_{mul}=2)**	**SGDR (T_0=50, T_{mul}=1)**
LR=0.001	**Accuracy**	0.895	0.895	0.905
	Matthews Correlation Coefficient	0.7967	0.7967	0.8169
	Cohen's Kappa	0.79	0.79	0.81
LR=0.005	**Accuracy**	0.925	0.925	0.92
	Matthews Correlation Coefficient	0.8520	0.8534	0.8442

	Cohen's Kappa	0.85	0.85	0.84
LR=0.01	**Accuracy**	0.925	**0.93**	0.925
	Matthews Correlation Coefficient	0.8534	**0.8627**	0.8520
	Cohen's Kappa	0.85	**0.86**	0.85
LR=0.05	**Accuracy**	0.915	0.925	0.92
	Matthews Correlation Coefficient	0.832	0.8534	0.8427
	Cohen's Kappa	0.83	0.85	0.84
LR=0.1	**Accuracy**	0.895	0.92	0.925
	Matthews Correlation Coefficient	0.8046	0.8415	0.852
	Cohen's Kappa	0.79	0.84	0.85

Table 4.5. Best SGDR hyper-parameters for 40x, 100x, 200x, 400x magnification factors.

Selected Hyper-parameter	**40x**	**100x**	**200x**	**400x**
Initial Learning Rate	0.01	0.1	0.004	0.01
SGDR T_0	1	10	10	10
SGDR T_{mul}	2	2	2	2
Accuracy	0.965	0.94	0.955	0.93

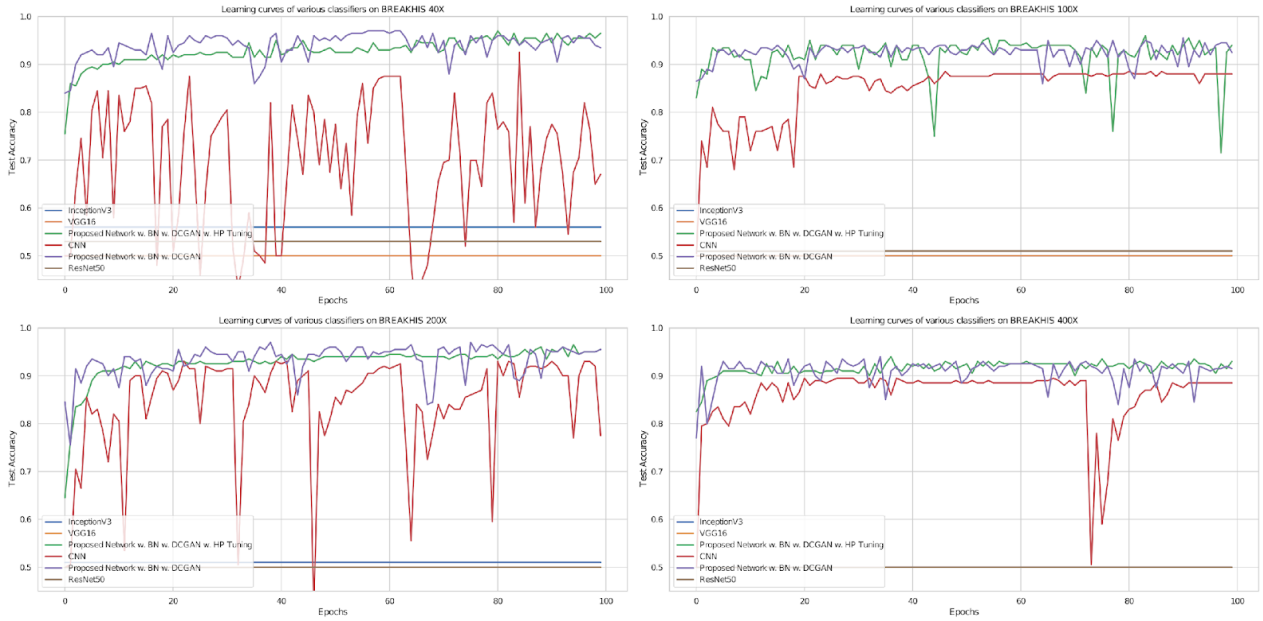


Figure 4.7. Learning curve depicting test accuracy across epochs and demonstrating better anytime performance with proposed approach in comparison to other approaches for various magnification factors of BreakHis dataset.

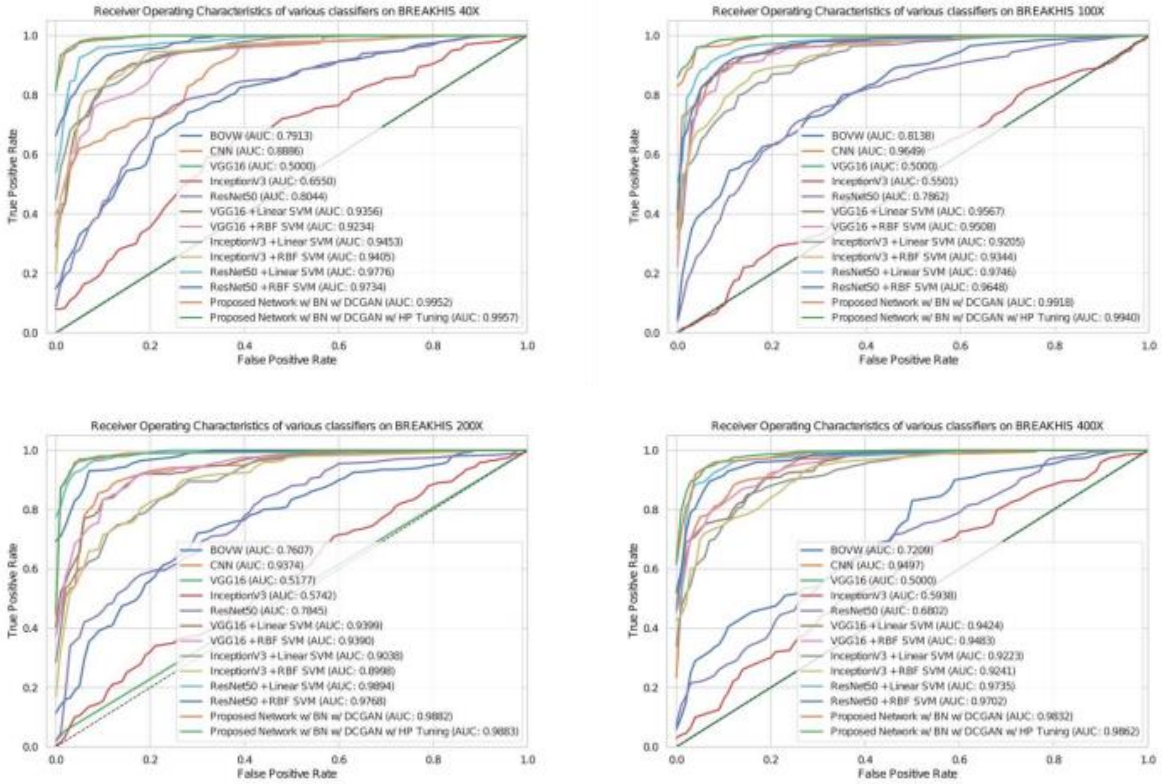


Figure 4.8. Comparison of the proposed approach with other approaches using ROC (Receiver Operating Curve) for various magnification factors of the BreakHis dataset.

Further, we have also done work to gauge the effect of data augmentation where deep learning-based experiments are conducted to observe the effect of data augmentation on the minority class for the imbalanced breast cancer histopathology dataset (BREAKHIS). Two different pre-trained networks are fine-tuned with the minority-augmented dataset. The pre-trained networks were already trained on the well-known ImageNet dataset consisting of millions of high-resolution images belonging to multiple object categories. The model so trained is further subjected to transfer learning, to correctly classify cancerous patterns from non-cancerous conditions, in a supervised manner. Experiments were carried out in two phases. Phase-I investigates the effect of data augmentation applied on minority classes for the Inception-V3 and ResNet-50 pre-trained networks (Saini and Susan 2019). Results of phase-I are further enhanced in phase-II by the transfer learning approach in which features extracted from all layers of Inception-V3 are learned by the SVM and weighted SVM classifiers. From experimental results, it was found that the pre-trained Inception-V3 model with data augmentation on minority class outperforms other network types. Results also indicate that Inception-V3 with data augmentation of minority class and transfer learning with weighted SVM gives the highest classification accuracies.

An investigation is conducted in our work using the breast cancer dataset, to effectively discriminate minority class samples from the majority class through deep learning-based experiments. A data augmentation-based approach using pre-trained networks and transfer learning is used to correctly classify cancerous patterns from non-cancerous conditions. The transfer learning approach is used in the experiment to

transfer the knowledge from one domain to another domain for performing a two-class classification task to segregate Benign from Malignant classes.

Following the cue from the existing work, our work induces data augmentation, however on the minority class only, similar to the oversampling step (Susan and Kumar 2019). The minority augmented dataset, which is, in essence, balanced, is then applied for transfer learning through pre-trained networks and eventually classified using Weighted Support Vector Machine (SVM). The following data augmentation operations were applied to the minority class for the complete dataset: shear with a range of 10, upper and lower zoom with the range of 20 percent, and horizontal flip, along with resizing and pre-processing operations.

In Phase I: Classification Using Pre-trained Networks. The pre-trained model was used for performing the classification experiment. Phase II: Deep Feature Extraction and Transfer Learning Phase II, comprises deep feature extraction from the pre-trained model fine-tuned with the minority augmented dataset. The deep features are extracted from all the layers of the Inception-V3 pre-trained model taken in the right order. The classifiers used for our experiment involving transfer learning based on the deep feature extracted are SVM and weighted SVM. For the purpose of our deep learning experiments, the popular python framework Keras has been used. It uses a Tensorflow backend to perform all its internal computations. The model will be trained across 3 epochs with 29 steps in each epoch. While training the pre-trained network, sparse categorical cross-entropy is used as the loss function for each fold along with Adam optimizer (Adaptive Moment Estimation), batch size set as 32 in each fold, with RELU activation function in a dense layer, is used for the experiment.

Classification report has been generated using the scikit-learn package in python for analyzing the performance of the model with precision, recall, F1-score, and ROC curve, (Receiver Operating curve).

In Phase I, from the experimental analysis, it was found that there is a huge difference between test and train accuracy. As average train accuracy in the case of Inception-V3 is approximately between 70 to 80% range with and without data augmentation. Whereas, the average test accuracy has improved from 47 to 50% as shown in Table 4.6. Experiments performed using Inception-V3 pre-trained networks with and without data augmentation are shown in Tables 4.7. From the experimental evaluation, it was found that Inception-V3 is unable to detect minority classes. According to the literature, it was found that while dealing with the imbalanced dataset, results are biased towards the majority class as shown in Table 4.8. Figure 4.9 depicts the test samples tested on the Inception-V3 pre-trained network after applying data augmentation on minority class using one of the best folds out of the average (Avg) five-fold. Whereas, data augmentation applied separately on the minority is able to detect minority class efficiently in comparison to the Inception-V3 pre-trained network after data augmentation is applied on both classes. Figure 4.10 represents sample test images without applying data augmentation. A similar experiment was conducted with ResNet-50 architecture to measure the performance of data augmentation techniques as illustrated in Tables 4.9 and 4.10. From experiment results, it was found that Inception-V3 outperforms ResNet-50 in all the cases. Phase II experiments were there to evaluate the performance of Inception-V3 under different conditions on the basis of different evaluation parameters such as precision, recall, F1-score, and accuracy to evaluate performance of the model. The

comparison was done between Inception-V3 with data augmentation applied on the minority class, obtained as the best result in the Phase I experiment, with the extracted features from the Inception-V3 pre-trained model and trained on the SVM and weighted SVM classifiers. From the experimental analysis, it was found that Inception-V3 with weighted SVM and data augmentation applied to the minority class outperforms other Inception-V3-based experiments. A solution for countering class imbalance in deep learning is proposed in this work specifically for the BREAKHIS breast cancer dataset. It is proved that we cannot only rely on accuracy for the identification of an imbalanced dataset. Various other parameters need to be considered such as F1-score, Precision, Recall, and ROC. We have conducted experiments in two phases. Phase-I investigates the effect of the data augmentation technique when applied to minority classes only for the pre-trained networks Inception-V3 and ResNet-50. Results obtained are found better with the application of data augmentation on the minority class in the Inception-V3 pre-trained model. In Phase II, features were extracted from all layers of Inception-V3 and learned by the SVM and weighted SVM classifiers. Results show that Inception-V3 with data augmentation on minority classes works best with transfer learning using weighted SVM as compared to other networks.

However, it was also observed that despite having high test accuracy, if the model is unable to give correct predictions and the results are biased towards the majority class then the minority class is left undetected and the model will evaluate

Table 4.6. Comparison of different Performance evaluation scores using cancer dataset in case of Inception-V3 pre-trained network.

Classes	**Inception-V3**		**Inception-V3 (with data augmentation on both class)**		**Proposed Inception-V3 (with data augmentation on minority class)**	
	Benign	**Malignant**	**Benign**	**Malignant**	**Benign**	**Malignant**
Avg precision (five-fold)	0.076	0.49	0.05	0.49	0.46	0.44
Avg recall (five-fold)	0.006	0.48	0.002	0.99	0.56	0.35
Avg F1-score (five-fold)	0.012	0.66	0.004	0.66	0.5	0.39
Support	100	100	100	100	100	100
Fold 1 Accuracy	0.5		0.51		0.46	
Fold 2 Accuracy	0.5		0.5		0.48	
Fold 3 Accuracy	0.49		0.5		0.47	
Fold 4 Accuracy	0.49		0.5		0.48	
Fold 5 Accuracy	0.48		0.5		0.46	
Avg test accuracy (five fold)	0.49		0.49		0.47	

Table 4.7. Comparison of different Performance evaluation scores using cancer dataset in case of ResNet-50 pre-trained network.

Classes	**ResNet-50**		**ResNet-50 (with data augmentation on both class)**		**ResNet-50 (with data augmentation on minority class)**	
	Benign	**Malignant**	**Benign**	**Malignant**	**Benign**	**Malignant**
Avg precision (five-fold)	0.2	1.00	0.2	0.5	0.0	0.5

Avg recall (five-fold)	0.002	1.0	0.002	1.0	0.0	1.0
Avg Fl-score (five-fold)	0.004	0.67	0.008	0.67	0.0	0.67
Support	100	100	100	100	100	100
Fold 1 Accuracy	0.5		0.5		0.51	
Fold 2 Accuracy	0.51		0.5		0.52	
Fold 3 Accuracy	0.5		0.5		0.52	
Fold 4 Accuracy	0.5		0.5		0.51	
Fold 5 Accuracy	0.51		0.5		0.5	
Avg test accuracy (five fold)	0.5		0.5		0.51	

Table 4.8. Comparison of different Performance evaluation scores using cancer dataset in case of proposed Inception-V3 with SVM.

Classes	**Inception-V3**		**Inception-V3 + SVM**		**Proposed Inception-V3 + weighted SVM**			
	Data augmentation on minority class				**Without data augmentation**		**Proposed method with data augmentation on minority class**	
	Benign	**Malig nant**	**Benign**	**Malign ant**	**Benign**	**Malign ant**	**Benign**	**Malign ant**
Avg precision (five fold)	0.49	0.48	0	0.5	0.66	0.58	0.68	0.59
Avg recall (five fold)	0.56	0.41	0	1	0.44	0.77	0.46	0.78
Avg F1-score (five fold)	0.52	0.44	0	0.67	0.53	0.66	0.55	0.67

Support	100	100	100	100	100	100	100	100
Fold 1 Accuracy	0.46		0.51		0.6		0.62	
Fold 2 Accuracy	0.48		0.51		0.61		0.63	
Fold 3 Accuracy	0.48		0.5		0.61		0.62	
Fold 4 Accuracy	0.47		0.5		0.6		0.62	
Fold 5 Accuracy	0.46		0.5		0.6		0.63	
Avg test accuracy (five fold)	0.47		0.5		0.6		0.62	

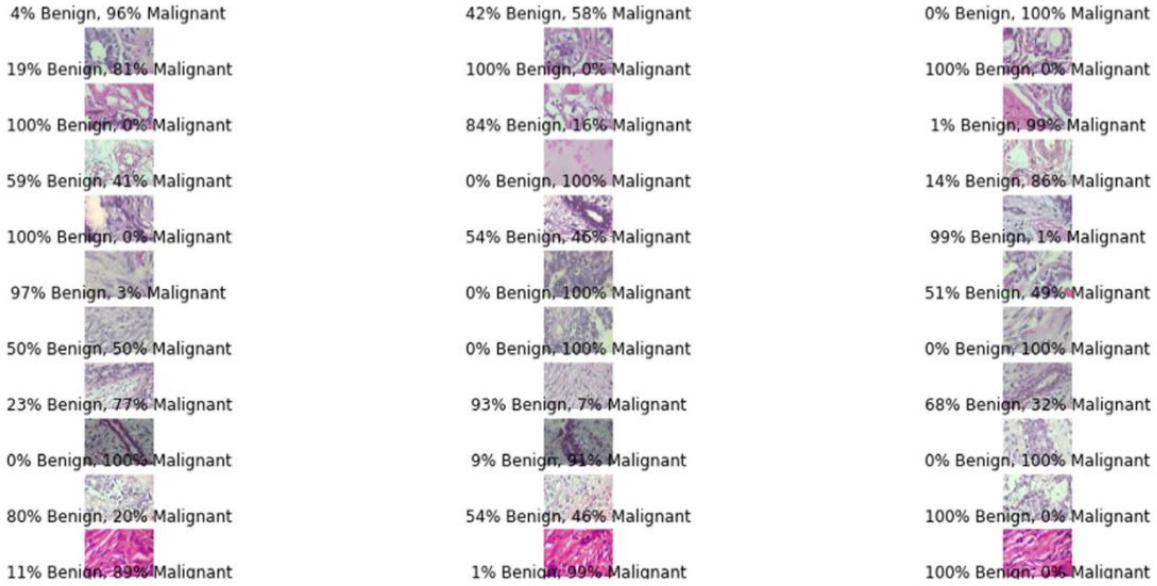


Figure 4.9. Samples tested on Inception-V3 pre-trained network after applying data augmentation on minority class.

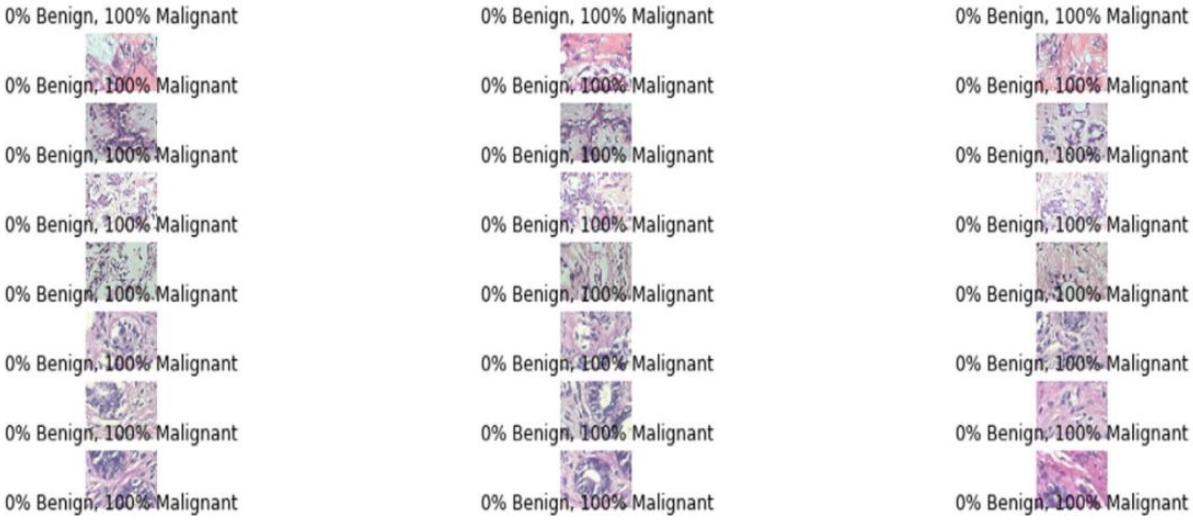


Figure 4.10. Samples tested on Inception-V3 pre-trained network without applying data augmentation.

In case of individual augmentation techniques applied one at a time one can isolate its effect on the model's performance for each individual operation; however, this technique influences the model's ability to generalize and detect important features on a robust variety of conditions. In addition to shift and rotation, these techniques can simulate variations in object position and orientation, allowing the model to learn robustness to these changes. Addition of horizontal flip can introduce horizontal symmetry and help the model learn invariant features across orientations. Addition of noise can help the model become more resilient to variations in the input data and improve its generalization. However applying multiple augmentation operations together allows the model to learn the robustness of these changes similar to real-world settings. By combining multiple augmentation techniques, the target training dataset which is used in the model training pipeline does more closely resemble the real-world scenarios the model would encounter.

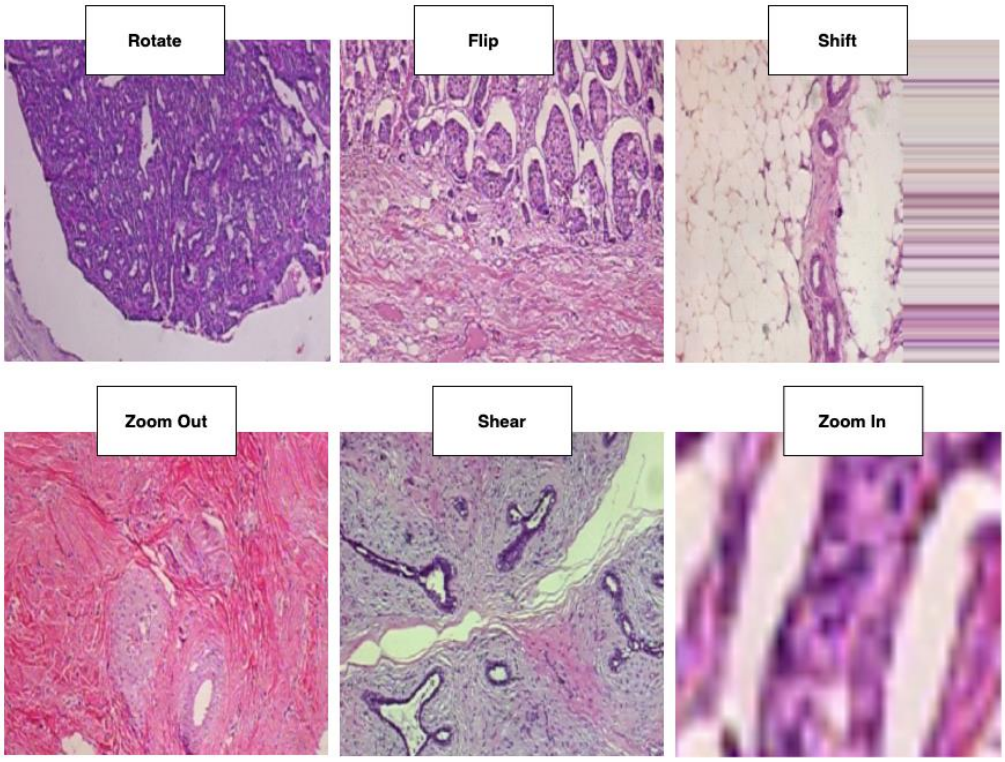


Figure 4.11. Illustration of images after applying individual data augmentations on samples of BreakHis dataset.

Real-world images in the wild often have a combination of various transformations, and training on augmented data helps the model adapt to such variations which helps improve model robustness to a vast extent also leading to better performance in handling variations during inference, and also improves the model's ability to generalize to unseen data. It is similar in case of DCGAN used for artificial synthetic generation of samples to counter minority imbalance classes over majority class samples. As image samples generated by generative models apply complex sets of transformations through the use of convolutional and deconvolutional layers, to achieve the same they have much more operations involved than randomly affine transformations. Both of the techniques apply transformations on minority classes to counter the effect of class imbalance and are seen as a measure to reduce disparity in

the number of samples of the skewed class distribution observed in the imbalanced datasets used for our experiments.

4.2 Limitations

The limitation of the proposed work is that applying DCGAN at data level is directly dependent on the number of samples of minority class as we employ deep generative modeling for enhancing the performance of the classifier. For scenarios with much less number of minority samples, the DCGAN training distribution would not be able to generalize well and would fail to generate high quality samples resulting in sub-optimal performance. Despite the limitations, the proposed methodology is able to tackle the class-imbalance problem that adversely affects many state-of-the-art deep learning networks as well as other traditional machine learning techniques. The proposed architecture is able to learn fine grained features on top of ImageNet pre-trained deep features, specific to the biomedical datasets. Total number of network parameters is reduced by introducing the Global Average Pooling layer instead of flatten in the original VGG16 architecture. This also helps to reduce the number of FLOPS. The proposed approach is able to work well even when the microscopic images are very different at data level as in the BreakHis dataset that contains images of four different magnification factors. However, the limitation is that application of proposed DCGAN at data level is directly dependent on the number of samples of minority class.

When applying Weighted Support Vector Machine (SVM) to multi-class imbalanced datasets, there are specific limitations. Firstly, there is a difficulty in

assigning weights to the different classes. This task can be challenging, as determining the optimal weight values for each class becomes more complex compared to binary or balanced datasets. Improper weight assignment may lead to biased or suboptimal results, ultimately impacting the overall performance of the model. Additionally, the performance of Weighted SVM heavily relies on the appropriate selection of weights. It is crucial to determine the optimal weights for each sample or class, which can be challenging. If the weight assignment is not properly done, it can result in biased results or degraded performance of the model. It is important to consider these limitations and carefully select and assign weights in multi-class imbalanced datasets to ensure accurate and unbiased classification results.

Chapter 5
VGGIN-Net: Novel Deep Learning Architecture for Binary and Multi-Class Imbalance Problem

This section[1] will contain the terminology, and methodology adopted for the creation of the novel deep learning architectures along with the detailed discussion of implementation steps and hyperparameters required while executing the approach followed by the outcome of the research. The proposed novel deep learning architecture, VGGIN-Net, can be employed for various binary-class and multi-class problems. VGGIN-Net is specifically designed to enable the transfer of domain knowledge from the extensive ImageNet object dataset to smaller imbalanced breast cancer datasets. Its design ensures effective transferability, allowing it to be applied to other imbalanced biomedical datasets to address class imbalance, irrespective of binary or multi-class scenarios.

: [1] The contents of this chapter are published in "VGGIN-Net: Deep Transfer Network for Imbalanced Breast Cancer Dataset," in IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 20, no. 1, pp. 752-762, 1 Jan.-Feb. 2023 and "Cervical Cancer Screening on Multi-class Imbalanced Cervigram Dataset using Transfer Learning." In 2022 15th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), pp. 1-6. IEEE, 2022.

5.1 Objective 4: Development of Novel Deep learning architectures for Multi-class Imbalance Problem

There exist various issues in binary class and multi-class imbalanced datasets which exist in most real-world applications. A Novel deep learning architecture is designed by combining various layers and architectural blocks based on different variants of convolutional neural networks and pre-trained networks to deal with binary and multi-class imbalance problems as illustrated below:

5.1.1 Binary-class Classification

A deep neural network architecture is proposed based on the transfer learning concept which is formulated by fusion of concatenating and freezing and all the layers till the block4 pool layer of the VGG16 pre-trained model along with the randomly initialized naïve Inception block module (Saini and Susan 2023a). Various other appropriate layers as displayed in Figure 5.1 were also part of the classification network. Dropout and data augmentation is also added as regularization techniques. The proposed architecture is formulated and structured in a manner that it can be effectively transferred and learned on other binary and multi-class imbalanced dataset.

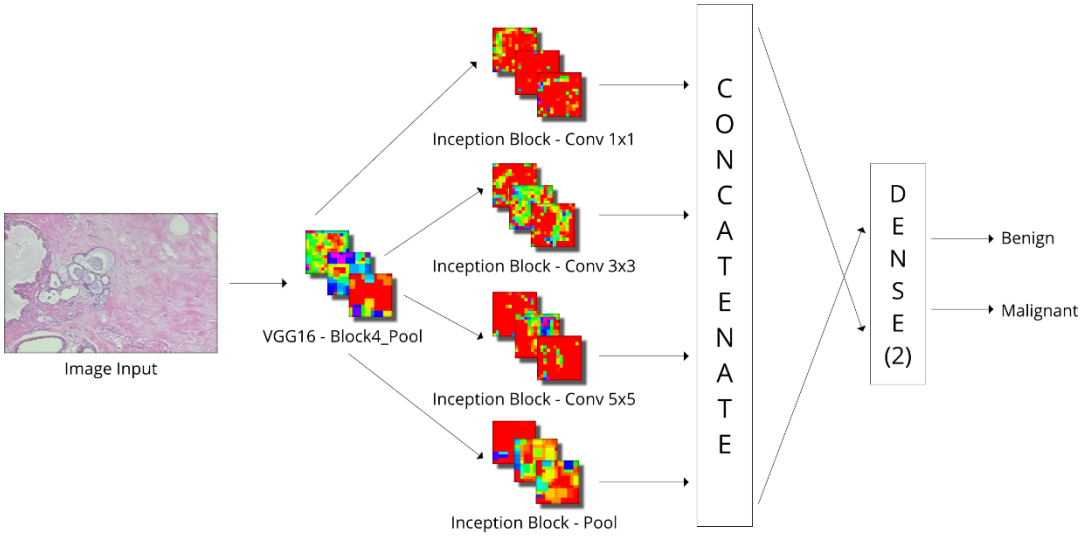


Figure 5.1. Proposed deep network architecture VGGIN-Net showing the lower layers till block4 pool layer of VGG16 pre-trained network and the higher layers comprising of naïve Inception module and the dense layers.

However, the imbalanced nature of this dataset brings in several challenges as the class imbalance problem causes several incorrect predictions. To tackle this problem, we have proposed a novel deep learning based architecture incorporating transfer learning in this work. The transfer learning approach using the pre-trained networks has the advantage of transferring the learned weights from an architecture that is trained in another domain to the biomedical domain, which will ultimately reduce the training time along with the computational cost to train the model from scratch. The contribution of the work is elaborated below: (a) Successful design of novel deep network architecture using transfer learning approach to solve class imbalance problem in breast cancer datasets. The proposed architecture is created by combining the relevant layers from the VGG16 pre-trained network (layers till block4 pool layer) along with the naïve Inception module in combination with flatten, batch normalization, and dense layers. Also, certain regularization techniques have been infused in the proposed model such as data augmentation and dropout which overall helps to reduce the overfitting to a great extent. (b) The proposed network has been

successfully tried and tested on images of different magnification factors, establishing the network's invariance to size and scale of the image. The results indicate an overall improvement in the classification performance. (c) We have analyzed the effect and significance of block-wise fine-tuning on the proposed deep transfer neural network architecture which has a substantial impact on the classification performance. (d) The proposed 24-layer network architecture has been articulated by integrating the right combination of layers in a well-ordered way to reduce the computational complexity involved. The formulated 24-layer architecture is constructed as shown in Figure 5.2 by first stacking the VGG16 layers, starting from the VGG16 pre-process layer till the block4 pool layers. The 224 x 224 x 3 image is given to the VGG16 pre-trained network as input. The reason for considering the features till block4 pool layers from the VGG16 pre-trained model is to extract the significant bottleneck features. The consideration of features beyond the block4 pool layer would only increase the computational difficulties with improvement in the performance of the model, as validated by our experimental results. Further, the obtained relevant bottleneck features till block4 pool layer of VGG16 pre-trained network are concatenated with the layers of naïve Inception block. The naïve Inception block consists of convolutional layers having filters of sizes of 1x1, 5x5, and 3x3, with each layer having a stride of size 3x3 and ReLU (Rectified linear unit) activation function (Szegedy, et al. 2015), with the addition of max pooling 2D layer. In this work, we have considered the goodness of both the pre-trained models (VGG16 and Inception) to create a more robust architecture that effectively resolves the class imbalance problem. The use of VGG16 pre-trained layers in the initial stage of our network was motivated by the fact that VGG16 architecture achieves good accuracy for most of the image classification problems and this network is efficient in dealing with images

of different scales and magnification factors. This is particularly useful for our study as one of the target datasets that we experiment with, is the BreakHis dataset which consists of images of varying magnification factors.

There are certain crucial deployment challenges faced in the VGG-based architecture which motivated us to modify the VGG16 architecture. As inspired by the previous works (Perdana, et al. 2019, Kumar, et al. 2020, Baheti, et al. 2018) we have modified the VGG16 architecture so as to overcome the deployment challenges that come with VGG-Nets. Such computational challenges are prevalent even on powerful single-GPU systems (Graphical Processing Units) due to their large memory footprint. The sequential ConvNet, VGG16 bears a large number of parameters (140 million) due to the presence of multiple convolutional layers of varying receptive fields, hence, it can become inefficient for inference at test time. Due to the presence of the large number of parameters, the VGG16 network is also prone to vanishing gradient problems. The presence of three fully connected layers present in the original VGG16 architecture is primarily responsible for the major bulkiness of the model. So, we have extracted the relevant deep features uptill block4 of the pre-trained model and removed all the layers after that which added extra complexity and computation to the proposed architecture. Further, to address the shortcomings seen in the VGG16 architecture, we have added the naïve Inception block as an additional block in the proposed architecture. The GoogLeNet incarnation of the Inception architecture uses multiple auxiliary classifiers to tackle the vanishing gradient problem. In our case, any auxiliary loss has not been used to train the inception block because of the presence of a lesser number of images available in the currently used dataset in comparison to the large-scale ImageNet. The reason for the addition of a single

Inception block in the higher layers of the proposed architecture is that it would not require auxiliary classifiers and the model can converge by itself using a single loss only. Also, it would be less computationally expensive to add a single Inception module instead of the addition of multiple similar modules. The main idea behind adding the naïve Inception module to the higher end of our network is to cover a larger region of a convoluted image while preserving the finer details. The naïve inception block is specifically engineered to convolve in parallel such that accurate detailing is possible through 1x1, 3x3, and 5x5 convolutions (64, 128, and 32 filters were used respectively). The goal of adding the naïve module is to increase the CNN's learning ability and abstraction of complex filters which was also found to be a drawback in VGG-based architectures. Moreover, the advantage of the Inception architecture is that it is able to perform well even with a single fully connected layer (Szegedy, et al. 2015). The consideration of the naïve Inception block as a suitable choice along with a single dense layer makes the architecture less computationally expensive. A single dense (fully connected) layer with a softmax activation function is present at the output to learn the proposed network architecture to deal with the higher-end linear features and to find the probability of occurrence of the image belonging to each class for the two-class classification problem (i.e. Benign and Malignant). Further, batch normalization, flatten, and dropout layers are added to enhance the network performance. We refrain from using multiple batch normalization layers (one batch normalization per convolutional layer) as per inspiration from (He, et al. 2016) as a single batch normalization should suffice when layers are being concatenated as in the case of the Inception module. Flatten layer is added after batch normalization to reduce the features into one dimension of size 144256. Dropout regularization (Srivastava, et al. 2014) with a rate of 0.4 is added after the flatten layer which in turn

helps to avoid overfitting problems. Our transfer network learns the higher layer features specific to cancer images. In doing so, we follow the guidelines as per work done by Yosinski, et al. who propounded the theory that when the base and target datasets are dissimilar, the use of pre-trained weights alone may degrade the performance (Yosinski, et al. 2014). It is essential that the higher-level features should be specific to the target dataset instead of the base dataset. Another regularization technique involved in the proposed architecture, specifically at the data level, is data augmentation which is applied to the training dataset in order to synthetically increase the number of samples and also to improve the overall performance of the network by reducing the fitting problem (Shorten, et al. 2019). The typical CNN training process employing data augmentation would include on the fly generation of random image samples across training mini-batches by use of affine transformations. In the proposed network, we have applied certain data augmentation operations as inspired by Howard 2013 on both the classes (Benign and Malignant). The data augmentation operations applied on images include: (a) random rotation within a range of 20 degrees, (b) random width and height shift operation with a range of 0.2 i.e. translation of the images both horizontally as well as vertically by the number of pixels less than or equal to 20% of the actual image dimensions, (c) random horizontal and vertical flip, combined with (d) random shear and random zoom operations with the same range as that of translation. These values were determined after lots of experimental trials in order to maximize performance. To improve the regularization of our network further we make use of random crop which helps the network to learn even better due to the translation invariance property of convolutional networks (Howard 2013). The image patches are resized to 224 x 340 using bilinear resizing and then randomly cropped into patches of 224 x 224. At inference (test) time, a central crop was used. The fine-

tuning approach has further been adapted on the proposed deep transfer learning architecture, inspired by several works in literature. In the case of the DeTrac approach, Abbas et al focused on the relevance of applying fine-tuning to different architectural blocks of pre-trained CNN (Abbas, et al. 2020). Similarly, Sharma and Mehra (2018) emphasized the role of transfer learning, full training, and fine-tuning of several pre-trained networks for the medical image dataset (Sharma and Mehra 2020). From the analysis, it was found that VGG16 pre-trained network features with logistic regression as classifier give the best performance amongst other combinations of VGG19 and ResNet-50 pre-trained network with regression. The same authors further extended their work and elaborated on the role of layer-wise fine-tuning and presented the in-depth study of layer-wise fine-tuning on AlexNet for the BreakHis dataset (Sharma and Mehra 2020). Authors have emphasized the role of appropriately fine tuning the network for different magnification factors Kandel and Castelli (2020) did the comparative analysis by emphasizing the role of fine-tuning of complete networks on various pre-trained networks. For their study, they used VGG16, VGG19, and Inception pre-trained networks on the histopathological image dataset (Kandel and Castelli 2020). From the analysis, it was found that fine-tuning the complete pre-trained network might not be the ideal choice in all situations while considering different magnification factor images. All these previous works motivated us to apply a block-wise fine-tuning approach to the proposed network.

Our research work proposes a novel approach by combining modified VGG16 architecture with naïve Inception block to tackle imbalanced problems in breast cancer classification. The same has been empirically validated by conducting extensive experimentation along presented with the ablation study to prove the

veracity of our claim that our proposed architectural combination is able to solve the breast cancer classification task effectively by proposing an explainable and less computationally costly architecture. By adjusting the sequence and the right combination of appropriate layers in the proposed architecture we are able to obtain the competent architecture to enhance the performance of the classifier that can be utilized for transfer learning on any other breast cancer dataset. The modified VGG16 architecture is chosen in such a manner as to resolve the deployment issues associated with original VGG-Nets by reducing the number of dense layers to one and extracting the appropriate features from suitable layers. In addition, we have introduced the naive Inception block with the batch normalization layer to address the vanishing gradient problem, and data augmentation, regularization, and fine-tuning techniques have been used for improving the prediction performance.

All the experiments related to this work were conducted on the Google Cloud Platform using a single Compute Engine VM instance with dual-core Intel Xeon CPU (2.00 GHz) and 8GB RAM, and an NVIDIA Tesla T4 GPU accelerator with 16 GB memory. For performing all our experiments, we used the TensorFlow v2.3.0 framework with the help of Keras API (Chollet 2021) using Python v3.8.9. The proposed deep network architecture, VGGIN-Net, takes approximately 1 hour to train on a GPU when training on BREAKHIS dataset. The different hyperparameters were selected so as to maximize the performance. (i) Adam optimizer, used with learning rate initially assigned as 0.001; the Adaptive Momentum optimization automatically adjusts the learning rate for further training. (ii) The loss function used was categorical cross-entropy. (iii) The training batch size was set to 128 with a net budget of 100 epochs. The proposed architecture is fine-tuned for four different magnification

factors. While fine-tuning the network, the learning rate is significantly reduced so as to make sure that any large gradient updates would not cause the network to abruptly change any of the pre-trained weights. The training process is conducted with the help of a simple learning rate schedule where we exponentially decay the learning rate after 15 starting warmup epochs. While performing the fine-tuning process, the network is trained for a total of 50 sweeps (epochs). The warmup steps linearly increase the learning rate from 1e-5 to 5e-5 which is further exponentially decayed by a factor of 0.8. To verify our claim that the proposed network architecture can further support transfer learning based tasks on other target histopathological images dataset, we use the weights of our VGGIN-Net trained on BreakHis 40X dataset with some amount of fine-tuning to classify IDC +ve and –ve images from Breast-Histopathology-Images dataset. The weights chosen were from the 40X magnification factor as the target dataset also consists of images scanned at a 40X zoom factor. The hyperparameters for training the terminal dense layer for the new classification are similar to our other experiments except that have used Adam optimizer with a learning rate of 0.001. As any learning rate higher than that would have an effect on the overall performance of the classifier.

In our study, we have compared the proposed architecture with a few state-of-the-art deep learning approaches as well as popularly used CNN architectures. For this curated set of architectures, we primarily applied a transfer learning approach. This is due to the fact that our target imbalanced classification dataset contains much fewer samples in comparison to large-scale datasets (a few million images) which is almost always required to effectively train a large ConvNet from scratch. Initially, we experimented with VGG architecture using the well-known VGG16 network

proposed by Zisserman, et al. (Simonyan and Zisserman 2014). We also used the GoogLeNet incarnation of the Inception architecture as well as deep residual network ResNet-50 which was invented by He, et al. 2018 to classify histopathological images from the imbalanced BreakHis dataset. These methods when applied using transfer learning serve as effective baseline methods. So, we have evaluated the performance of popular deep learning approaches i.e., VGG16, GoogLeNet, and ResNet-50. We have also done the comparative analysis between these methods and our proposed deep transfer network, named VGGIN-Net, obtained after considering the features extracted till block4 pool layer of VGG16 with a single dense layer, and also with the addition of an Inception block and a single dense layer which is the proposed architecture. The VGG16 architecture had been modified with a single dense layer but after the addition of a naïve Inception block, the same network architecture had shown tremendous improvement in the results. From the results tabulated in Tables 5.1, 5.2 (i) and (ii) and 5.3 (i) and 5.3 (ii), it is observed that VGGIN-Net shows remarkable improvement in terms of accuracy, F1 score, IBA, and GMean. In Figure 5.2. Validation accuracy and loss plot corresponding to the proposed architecture VGGIN-Net for different magnification factors are displayed. ROC curve analysis with its AUC is also shown in Figure 5.3 to validate the proposed approach for different magnification factors 40X, 100X, 200X, and 100X. Figure 5.4. Training, validation loss, and accuracy while training VGGIN-Net on BreakHis dataset for different magnification factors are displayed. Further Pre-trained weights from the same are used to fine-tune the Breast Histopathological Images dataset. In Table 5.1, comparative analysis with state-of-the-art methods is shown for the BreakHis dataset using accuracy as the evaluation parameter with scores reported across several runs. Hence, it is evident that our proposed network with and even without fine-tuning

shows remarkable improvement in results. Table 5.2. (i) and (ii) Performance Evaluation of VGG16, GoogLeNet, And ResNet-50 with the Modified VGG16 Architecture and the proposed approach on Breakhis Dataset for 40X, 100X, 200X and 400X magnification factors is shown respectively.

To emphasize the veracity of our claim that the proposed network architecture helps to tackle the class imbalance problem, certain experiments were conducted. A comparative analysis illustrates the use of various well-known approaches that deems to solve the class imbalance problem i.e., with undersampling and oversampling techniques. We observe the comparisons to sampling experiments in Table 5.3. (i) and Table 5.3 (ii), that the proposed architecture itself is able to tackle the class imbalance problem by itself without the requirement of any sampling technique. Extensive experiments were conducted related to the block-wise fine-tuning technique applied to the proposed network. It is observed from the analysis that the block-wise fine-tuning operations have shown significant improvement in performance as depicted in Table 5.4. (i) and Table 5.4 (ii). It is evident that different fine-tuning combinations are found suitable for different magnification factors. For 40X, fine-tuning of block3, block4, and Inception block seems to be an ideal choice, whereas, in the case of 400X, fine-tuning of block4 and Inception block was only found to be the perfect fit. Fine-tuning of the complete network was found to be ideal in the case of 100X and 200X magnification factor images. It can be inferred from the results that complete fine-tuning of the network is not always the perfect choice for different magnification factor images as different block-wise fine-tuning combinations can also be deemed to be suitable in certain scenarios. It is clearly visible that with the help of suitable block-wise fine-tuning, the network training

improves its anytime performance, and learning curves get more and more stable. The proposed approach along with fine tuning has shown significant improvements in the classification performance. For 40X, 100X, 200X and 400X magnification factors of the BreakHis dataset, the best obtained accuracies are 98.51%, 97.53%, 96.688% and 95.528% respectively. So, in a nutshell, it is notable to mention that our work demonstrates that single branch models can converge quite well and our work is not intended to deal with the training of increasingly complex residual models. Rather, we are aimed at building a simple model with reasonable depth and favorable accuracy that can be simply implemented using basic architecture blocks (like convolution, ReLU, max pooling, etc.) on a single branch while tackling imbalanced biomedical datasets. From the analysis, it was validated that the VGGIN-Net architecture also supports the transfer learning concept when tested on other breast cancer datasets. An ablation study has been conducted for the proposed VGGIN-Net architecture. In Table 5.5, we show experimental results on the 40X magnification factor to compare and contrast the use of blocks 3, 4, and 5 as the backbone features for our network. We inferred that feature extraction till the block4 pool layer is an ideal combination. Another set of experiments was conducted to demonstrate the selection of naïve Inception block in the proposed architecture. Comparison with another variant of the Inception block as tabulated in Table 5.6. proves that the naïve inception block is apt for our model. Although the dimensionality reduction block variant of the Inception module is less computationally expensive in comparison to the naïve inception block, the attained model performance obtained after combining the dimensionality reduction block to the proposed architecture is less in comparison to the naïve Inception block as validated by the experimental results illustrated in Table 5.6. The naïve Inception block was initially used with 64, 128, and 32 filters

for 1x1, 3x3, and 5x5 conv layers respectively. In Table 5.7., we show experiments on using diverse widening factors (K) where each value of K indicates the multiple factors by which the filters are increased. It was found that a K value as 1 is the ideal choice instead of K values of 2, 3, 5, and 10 in the naïve Inception block. Also, it helps to validate our choice of the number of filters besides keeping the computational complexity optimum since K=1 is the lowest possible considered value. More results in the supplementary file highlight the significance of data augmentation for enhancing the performance of the various deep networks including VGGIN-Net. Table 5.8 illustrates the experiments related to the proposed VGGIN-Net with and without data augmentation for different magnification factors for the BreakHis dataset. Experiments show that VGGIN-Net with data augmentation works significantly better in comparison to VGGIN-Net without Data Augmentation. Figure 5.5. Performance evaluation of our proposed VGGIN-Net across epochs with and without data augmentation learning curves are shown for accuracy and loss for training and test sets for different magnification factors of the BREAKHIS dataset shown. The incorporation of data augmentation in the training pipeline helps reduce overfitting by imparting the necessary regularization, allowing the models to learn continually across several epochs. For our case, we have applied all random transformations including random cropping of samples to fixed crop size. To validate that our design of the proposed deep transfer architecture supports further transfer learning on any other breast cancer biomedical dataset we performed experiments as illustrated in Table 5.9 using the Breast-Histopathological-Images dataset. Table 5.10. analysis of the effect of data augmentation (applied at a mini-batch level using transformations as aforementioned) on baseline VGG16, GoogLeNet networks and

comparing it with proposed VGGIN-Net network using the ratio of inter-class F1 scores for models trained on BREAKHIS dataset.

Final conclusion drawn from this work is a novel deep learning based network VGGIN-Net has been proposed using layers from the pre-trained deep network VGG-16 at the lower level, and a trainable Inception module and dense layers at the higher level. The proposed transfer network has been compared with various state-of-the-art approaches on the basis of various performance evaluation metrics. It is validated from the experiments that VGGIN-Net was designed to deal with the imbalanced breast cancer dataset and overall helps to improvise the robustness and generalizability of the approach. The proposed deep transfer network with fine-tuning has achieved accuracies of 97.10%, 96.67%, 97.16%, and 93.68% for the different magnification factors, for the BreakHis dataset. The proposed network was able to classify both the minority and majority classes effectively. We also validated through experiments that the trained VGGIN-Net model supports transfer learning on other breast cancer datasets.

Table 5.1. Comparison Of The Proposed Approach With The State-Of-The-Art Approaches On BreakHis Dataset.

Technique	**40X**	**100X**	**200X**	**400X**
Spanhol, et al. 2016	0.8960 ± 0.0650	0.8500 ± 0.0480	0.8400 ± 0.0320	0.8080 ± 0.0310
Spanhol, et al. 2017	0.8460 ± 0.0290	0.8480 ± 0.0420	0.8420 ± 0.0170	0.8160 ± 0.0370
Bayramoglu, et al. 2016	0.8300 ± 0.0300	0.8310 ± 0.0350	0.8460 ± 0.0270	0.8210 ± 0.0440
Zhu, et al. 2019	0.8570 ± 0.0190	0.8420 ± 0.0320	0.8490 ± 0.0220	0.8010 ± 0.0440
Gupta and Bhavsar 2017	0.8674 ± 0.0237	0.8856 ± 0.0273	0.9031 ± 0.0376	0.8831 ± 0.0301

Deniz, et al. 2018	0.9096 ± 0.0159	0.9058 ± 0.0196	0.9137± 0.0172	0.9130 ± 0.0740
Song, et al. 2017	0.9002 ± 0.0302	0.9120 ± 0.0440	0.8780 ± 0.0530	0.8740 ± 0.0720
Gupta and Bhavsar 2018	0.9471 ± 0.0088	0.9590 ± 0.0420	0.9676 ± 0.0109	0.8911 ± 0.0012
Ours	**0.9588 ± 0.0033**	**0.9657 ± 0.0087**	**0.9500 ± 0.0122**	**0.9315 ± 0.0034**
Ours (with fine-tuning)	**0.9710 ± 0.0046**	**0.9667 ± 0.0022**	**0.9716 ± 0.0033**	**0.9368 ± 0.0053**

Table 5.2. (i) Performance Evaluation of VGG16, GoogLeNet, And ResNet-50 with the Modified VGG16 Architecture and the Proposed Approach on Breakhis Dataset for 40x and 100x magnification factors.

Technique	**40X**				**100X**			
	Accuracy	**F1**	**IBA**	**GMean**	**Accuracy**	**F1**	**IBA**	**GMean**
VGG16	0.9294	0.87 / 0.95	0.79	0.89	0.9240	0.86 / 0.95	0.80	0.89
GoogLeNet	0.8682	0.78 / 0.91	0.71	0.84	0.8674	0.78 / 0.91	0.72	0.85
ResNet50	0.9350	0.89 / 0.95	0.85	0.92	0.9381	0.89 / 0.96	0.86	0.93
Modified VGG16 w/ Single Dense Layer	0.9387	0.89 / 0.96	0.84	0.91	0.9522	0.92 / 0.97	0.90	0.95
Modified VGG16 w/ Inception Block w/ Single Dense Layer	**0.9628**	**0.93 / 0.97**	**0.93**	**0.96**	**0.9681**	**0.95 / 0.98**	**0.93**	**0.96**

Table 5.2. (ii) Performance Evaluation of VGG16, GoogLeNet, And ResNet-50 with the Modified VGG16 Architecture and the Proposed Approach on BreakHis Dataset for 200x and 400x magnification factors.

Technique	**200X**				**400X**			
	Accuracy	**F1**	**IBA**	**GMean**	**Accuracy**	**F1**	**IBA**	**GMean**
VGG16	0.9119	0.86 / 0.94	0.83	0.91	0.8913	0.81 / 0.92	0.72	0.85
GoogLeNet	0.8880	0.82 / 0.92	0.78	0.88	0.8668	0.77 / 0.91	0.69	0.83
ResNet50	0.9431	0.90 / 0.96	0.87	0.93	0.9221	0.87 / 0.94	0.81	0.90
Modified VGG16 w/ Single Dense Layer	0.9357	0.88 / 0.96	0.82	0.91	0.8893	0.82 / 0.92	0.77	0.88
Modified VGG16 w/ Inception Block w/ Single Dense Layer	**0.9651**	**0.88 / 0.96**	**0.80**	**0.89**	**0.9364**	**0.89 / 0.95**	**0.86**	**0.93**

Table 5.3. (i) Performance Evaluation of the Proposed Approach with Undersampling and Oversampling Techniques on BreakHis dataset for 40X and 100X magnification factors.

Sampling Technique	**40X**				**100X**			
	Accuracy	**F1**	**IBA**	**GMean**	**Accuracy**	**F1**	**IBA**	**GMean**
Undersampling	0.9591	0.93 / 0.97	0.89	0.94	0.9381	0.90 / 0.96	0.89	0.94
Oversampling	0.9406	0.90 / 0.96	0.89	0.94	0.9593	0.93 / 0.97	0.90	0.95
None	**0.9628**	**0.93 / 0.97**	**0.93**	**0.96**	**0.9681**	**0.95 / 0.98**	**0.93**	**0.96**

Table 5.3. (ii) Performance Evaluation of the Proposed Approach with Undersampling and Oversampling Techniques on BreakHis dataset for 200X and 400X magnification factors.

Sampling Technique	**200X**				**400X**			
	Accuracy	**F1**	**IBA**	**GMean**	**Accuracy**	**F1**	**IBA**	**GMean**
Undersampling	0.9540	0.91 / 0.97	0.86	0.92	0.9262	0.88 / 0.95	0.84	0.91
Oversampling	0.9669	0.94 / 0.98	0.92	0.96	0.9303	0.88 / 0.95	0.83	0.91
None	**0.9651**	**0.88 / 0.96**	**0.80**	**0.89**	**0.9364**	**0.89 / 0.95**	**0.86**	**0.93**

Table 5.4. (i) Performance Evaluation of Block Wise Fine-Tuning on Proposed VGGIN-Net on Breakhis Dataset for 40x, 100x Magnification Factors.

Fine Tuning	**40X**				**100X**			
	Accuracy	**F1**	**IBA**	**GMean**	**Accuracy**	**F1**	**IBA**	**GMean**
Complete Network	0.9666	0.99 / 0.97	0.89	0.94	**0.9753**	**0.96 / 0.98**	**0.96**	**0.98**
Block 2, 3, 4, Inception block	0.9777	0.96 / 0.98	0.95	0.97	0.8674	0.71 / 0.91	0.56	0.74
Block 3, 4, Inception block	**0.9851**	**0.97 / 0.99**	**0.96**	**0.98**	0.9646	0.94 / 0.98	0.89	0.94
Block 4, Inception block	0.9610	0.93 / 0.97	0.88	0.94	0.9700	0.95 / 0.98	0.92	0.96
No Fine Tuning	0.9628	0.93 / 0.97	0.93	0.96	0.9752	0.95 / 0.98	0.93	0.97

Table 5.4. (ii) Performance Evaluation of Block Wise Fine-Tuning on Proposed VGGIN-Net on Breakhis Dataset for 200x, 400x Magnification Factors.

Fine Tuning	200X				400X			
	Accuracy	F1	IBA	GMean	Accuracy	F1	IBA	GMean
Complete Network	**0.9688**	**0.95 / 0.98**	**0.92**	**0.96**	0.9077	0.86 / 0.93	0.85	0.93
Block 2, 3, 4, Inception block	0.9467	0.91 / 0.96	0.91	0.95	0.9323	0.89 / 0.95	0.83	0.91
Block 3, 4, Inception block	0.9651	0.94 / 0.98	0.89	0.94	0.9426	0.90 / 0.96	0.85	0.92
Block 4, Inception block	0.9614	0.93 / 0.97	0.92	0.96	**0.9528**	**0.92 / 0.97**	**0.91**	**0.95**
No Fine Tuning	0.9651	0.88 / 0.96	0.80	0.89	0.9364	0.89 / 0.95	0.86	0.93

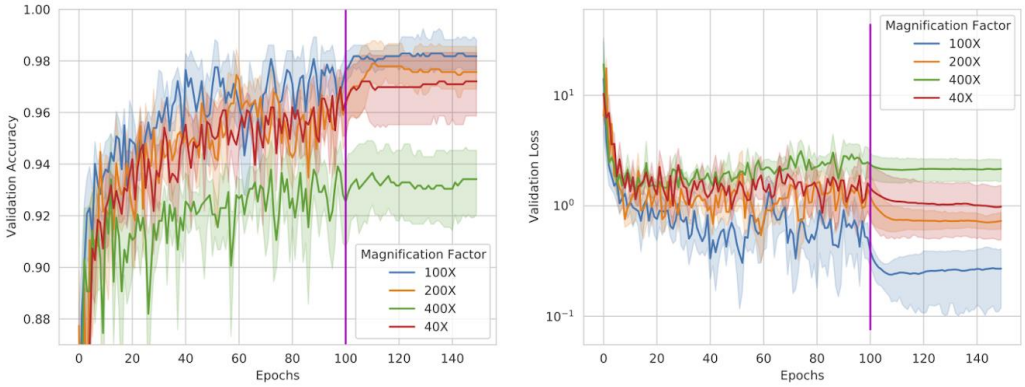
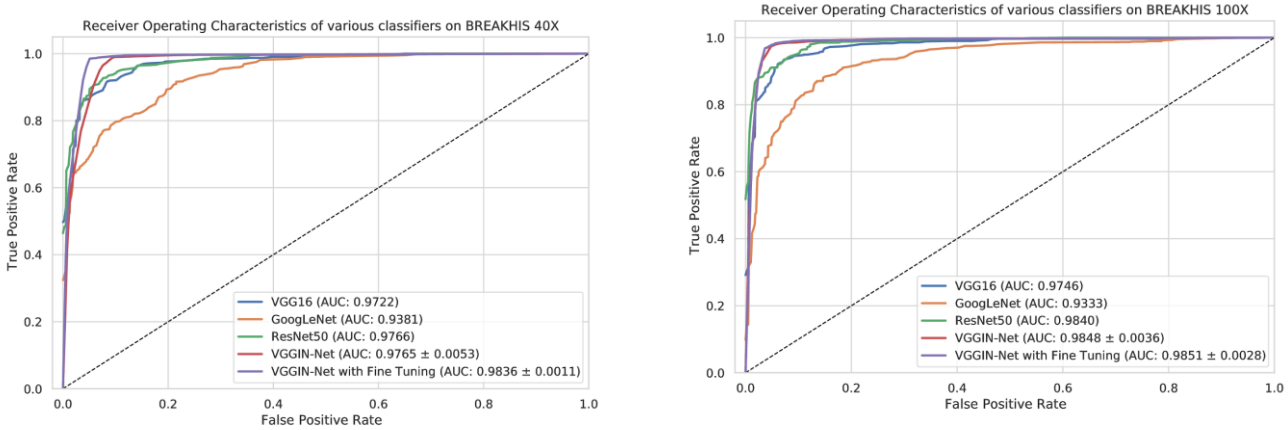


Figure 5.2. Validation accuracy and loss plot corresponding to the proposed architecture VGGIN-Net for different magnification factors (40X, 100X, 200X and 400X). Purple line indicates the start of fine tuning.



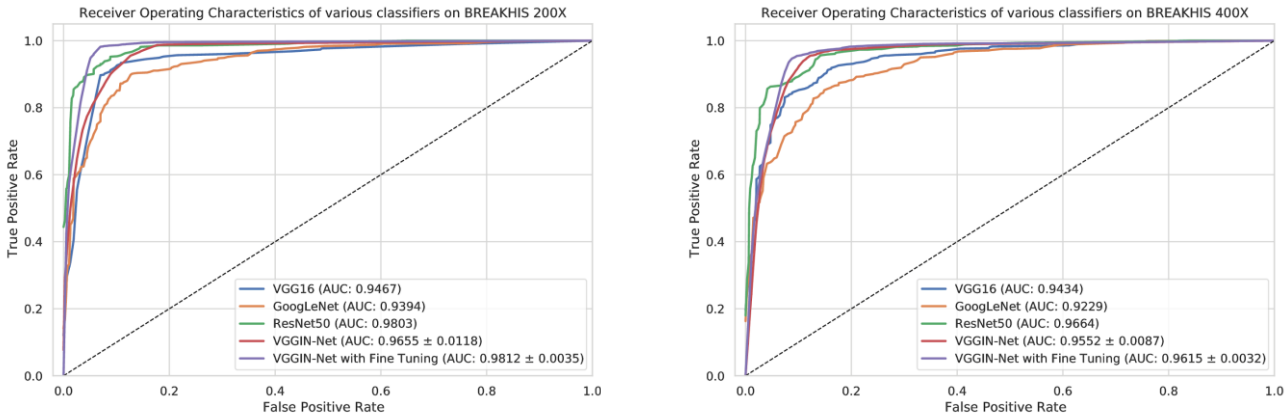


Figure 5.3. ROC curve comparison of proposed approach with state-of-the-art networks in case of (i) 40X, (ii) 100X, (iii) 200X and (iv) 400X.

Table 5.5. Analysis of features extracted from different blocks of VGG16 architecture to find the appropriate features in the Proposed architecture for 40X magnification factor on BreakHis Dataset.

Technique	**40X**			
	Accuracy	**F1**	**IBA**	**GMean**
Proposed Network using block3_pool	0.9536	0.92 / 0.97	0.89	0.94
Proposed Network using block4_pool	**0.9628**	**0.93 / 0.97**	**0.93**	**0.96**
Proposed Network using block5_pool	0.9536	0.92 / 0.97	0.89	0.94

Table 5.6. Analysis of Proposed architecture with the Inception block and dimensionality reduction Inception block for 40X magnification factor on BreakHis Dataset.

Technique	**40X**			
	Accuracy	**F1**	**IBA**	**GMean**
Proposed Network w/ Naïve Inception Block	**0.9628**	**0.93 / 0.97**	**0.93**	**0.96**

Proposed Network w/ Dimensionality Reduction Inception Block	0.9443	0.90 / 0.96	0.82	0.90

Table 5.7. Analysis Of Appropriate Number Of Filters In Inception Block To Be Used In The Proposed Architecture For 40x Magnification Factor On Breakhis Dataset.

Widening Factor	**40X**			
	Accuracy	**F1**	**IBA**	**GMean**
k=1	**0.9628**	**0.93 / 0.97**	**0.93**	**0.96**
k=2	0.9443	0.90 / 0.96	0.83	0.91
k=5	0.9684	0.95 / 0.98	0.93	0.96
k=10	0.9684	0.94 / 0.98	0.90	0.95

Table 5.8. Proposed VGGIN-Net with and without data augmentation for 40x, 100x, 200x And 400x Magnification Factors On Breakhis Dataset.

Technique	**40X**				**100X**			
	Accuracy	**F1**	**IBA**	**GMean**	**Accuracy**	**F1**	**IBA**	**GMean**
VGGIN-Net w/ Data Augmentation	**0.9628**	**0.93 / 0.97**	**0.93**	**0.96**	**0.9681**	**0.95 / 0.98**	**0.93**	**0.96**
VGGIN-Net w/o Data Augmentation	0.9239	0.86 / 0.95	0.80	0.89	0.9134	0.85/ 0.94	0.78	0.88

Technique	**200X**				**400X**			
	Accuracy	**F1**	**IBA**	**GMean**	**Accuracy**	**F1**	**IBA**	**GMean**

VGGIN-Net w/ Data Augmentation	**0.9651**	**0.88 / 0.96**	**0.80**	**0.89**	**0.9364**	**0.89 / 0.95**	**0.86**	**0.93**
VGGIN-Net w/o Data Augmentation	0.9155	0.85 / 0.94	0.78	0.88	0.8852	0.79/ 0.92	0.68	0.82

Table 5.9. Transfer Learning of Proposed VGGIN-Net On Breast Histopathological Images Dataset with and without Fine-Tuning after pre-training on BreakHis dataset.

Transfer Learning	**Accuracy**	**F1**	**IBA**	**GMean**
VGGIN-Net as Fixed Feature Extractor	0.8470	0.89 / 0.73	0.66	0.81
Fine Tuning the VGGIN-Net Inception Block	**0.8678**	**0.91 / 0.75**	**0.67**	**0.82**

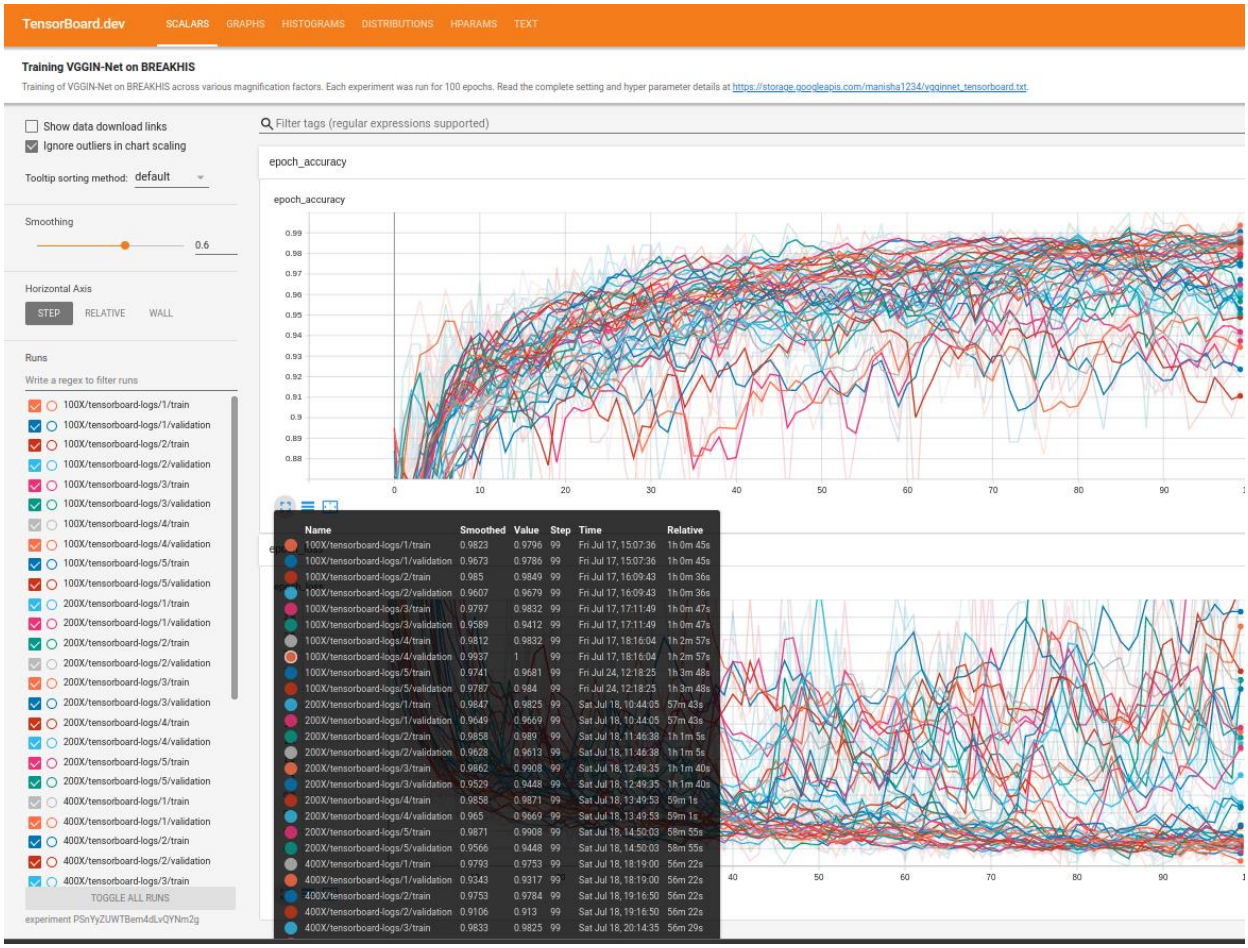


Figure 5.4. Training, validation loss, and accuracy while training VGGIN-Net on BreakHis dataset for different magnification factors. Pre-trained weights from the same are used to fine-tune on Breast Histopathological Images dataset.[1]

Table 5.10. Analysis of effect of data augmentation (applied at mini-batch level using transformations as aforementioned) on baseline VGG16, GoogLeNet networks and comparing it with proposed VGGIN-Net network using ratio of inter-class F1 scores for models trained on BREAKHIS dataset.

Model	**Magnification Factor**	**Without Data Augmentation**		**With Data Augmentation**	
		F1 Score		**F1 Score**	
		Benign	**Malignant**	**Benign**	**Malignant**
Baseline Consideration					
VGG16	**40X**	0.80	0.92	0.87	0.95
		Ratio: 0.8695		Ratio: 0.9157	

[1] https://tensorboard.dev/experiment/PSnYyZUWTBem4dLvQYNm2g/

	100X	0.80	0.92	0.86	0.95
		Ratio: 0.8695		Ratio: 0.9052	
	200X	0.83	0.93	0.86	0.94
		Ratio: 0.8924		Ratio: 0.9148	
	400X	0.74	0.90	0.81	0.92
		Ratio: 0.8222		Ratio: 0.8804	
GoogLeNet	**40X**	0.81	0.93	0.78	0.91
		Ratio: 0.8709		Ratio: 0.8571	
	100X	0.72	0.90	0.78	0.91
		Ratio: 0.8		Ratio: 0.8571	
	200X	0.83	0.93	0.82	0.92
		Ratio: 0.8924		Ratio: 0.8913	
	400X	0.79	0.92	0.77	0.91
		Ratio: 0.8586		Ratio: 0.8461	
Proposed Approach					
VGGIN-Net	**40X**	0.86	0.95	0.93	0.97
		Ratio: 0.9052		Ratio: 0.9587	
	100X	0.85	0.94	0.95	0.98
		Ratio: 0.9042		Ratio: 0.9693	
	200X	0.85	0.94	0.88	0.96
		Ratio: 0.9042		Ratio: 0.9166	
	400X	0.79	0.92	0.89	0.95
		Ratio: 0.8586		Ratio: 0.9368	

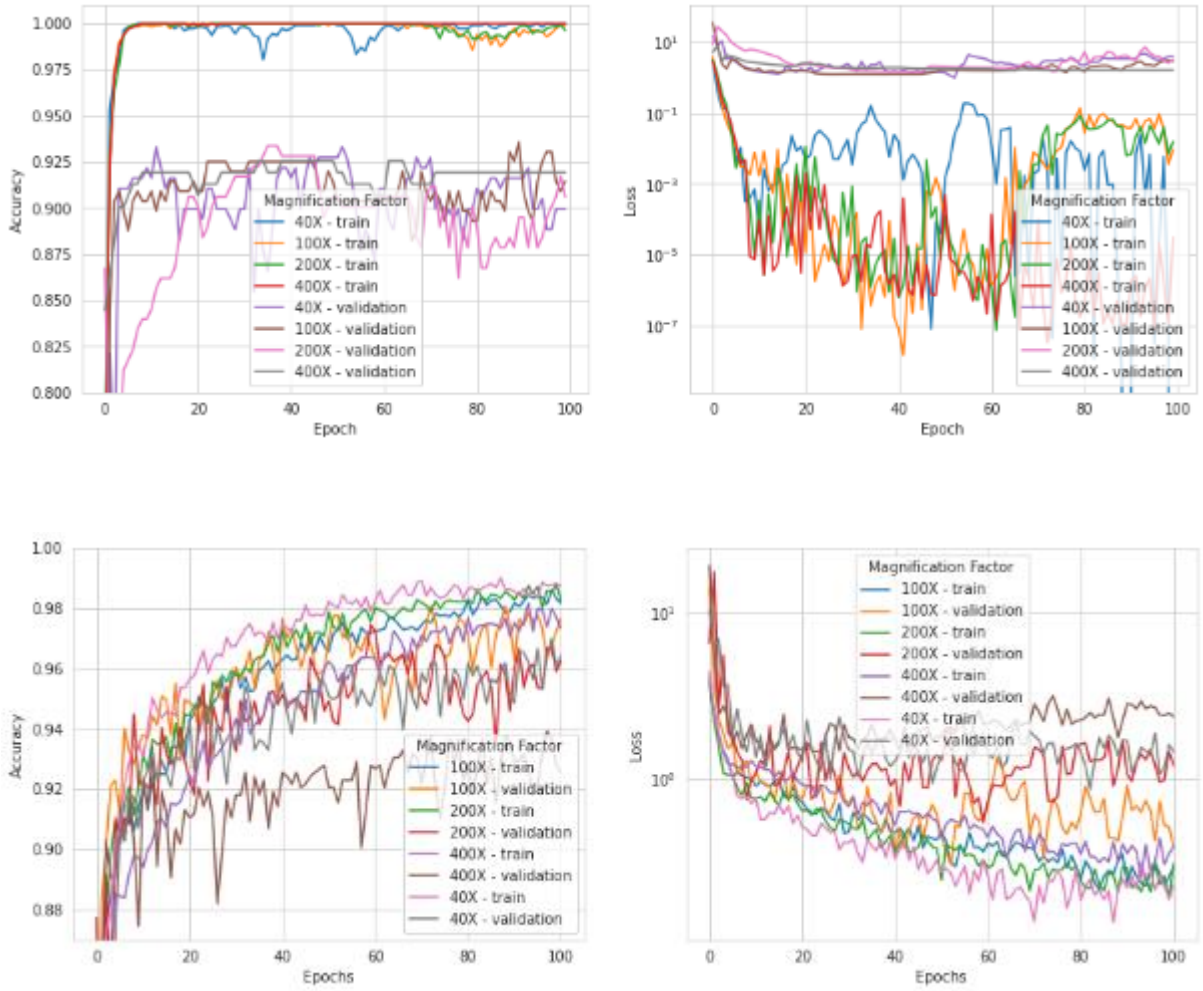


Figure 5.5. Performance evaluation of our proposed VGGIN-Net across epochs. (a) without data augmentation and (b) with data augmentation. (Top to Bottom) Learning curves are shown for accuracy and loss for training and test sets for different magnification factors of the BREAKHIS dataset (Left to Right respectively).

Further, we have designed a novel approach to correctly classify the cervix type from cervigram images for a multi-class imbalanced dataset. We have extended our work VGGIN-Net for multi-class imbalanced datasets. In the proposed model, we have used regularization in the form of dropout and data augmentation. Also, we have used the RandAugment approach for data augmentation for our multi-class imbalanced task which was found better as opposed to random combinations of flip, rotate, shift, and zoom which was used with this model on the binary classification

task of BreakHis. The experimental results prove the proposed approach along with the data augmentation and random undersampling strategy is effective for multi-class imbalanced datasets. As analyzed through our comparative analysis, the proposed approach in comparison to the other state-of-the-art approaches is vividly proven to be a more efficient method.

5.1.2 Multi-class Classification using VGGIN-Net

Cervical cancer is a deadly type of cancer that occurs in females. Screening for cancer is a very crucial aspect in order to cure it at its early stages. For cervical cancer screening tasks, the first primary task is the detection of the female cancer type which can be any of three known types. Type 1 cervixes don't require screening but women having Type 2 and Type 3 cervix require screening for further cancer detection (Matsuo, et al. 2019). Despite the availability of advanced medical and science facilities, there are still challenges to detect the cervix transformation zones correctly for further treatment later. So correct screening is very important to strongly fight against cancer through further diagnosis if required. However, screening and detection of the cervix type manually are problematic, time-consuming, and tedious due to the high probability and occurrence of manual errors. So, an automated screening approach can increase the efficiency of cancer detection tasks as well. In this work we have proposed the deep learning based architecture for multi-class imbalanced datasets. The crucial contributions of the work are (1) transfer learning of the features from a large dataset to a small cervical cancer multi-class imbalanced dataset and (2) data augmentation was applied successfully which has an overall impact on the deep learning architectures along with random undersampling strategy (Pereira and Nunes 2018) on the performance of the deep learning networks and also

avoids overfitting or underfitting problems to great extend (3) Extensive set of experiments were performed to demonstrate that the proposed approach achieves better performance using various evaluation measures such accuracy (weighted average and micro average), precision, recall, F1 score, geometric mean, index balanced accuracy, (Japkowicz 2013) in comparison to various state-of-the-art deep learning architectures. We have designed a novel approach to correctly classify the cervix type from cervigram images for a multi-class imbalanced dataset. We have extended our work on VGGIN-Net for multi-class imbalanced datasets and applied the proposed VGGIN-Net model to multi-class cervical cancer screening tasks instead of the binary classification of Breast cancer datasets (Saini and Susan 2022a). The proposed novel deep neural network architecture is based on the transfer learning approach by using the features till the block4 pool layer of the VGG16 pre-trained model (Simonyan and Zisserman 2014) along with the naïve Inception block module (Szegedy, et al. 2015). Further, we have added the batch normalization (Ioffe, et al. 2015), flatten, dense, flatten layers in the proposed architecture, and constructed the 24-layer architecture by stacking the appropriate layers of VGG16 layers with the naïve Inception block and a few more layers. In the proposed model, we use regularization in the form of dropout and data augmentation. We have used the RandAugment (Cubuk, et al. 2020) approach for data augmentation for our multi-class imbalanced task which was found better as opposed to random combinations of flip, rotate, shift, and zoom which was used with this model on the binary classification task of BreakHis (Spanhlol, et al. 2015). All our experiments related to this work were trained on Google Cloud TPU hardware (Ying, et al. 2018) access to which was granted through the TensorFlow Research Cloud (TRC) program. Each of our deep learning models was trained on the TPU v3-8 accelerator with the help of

128 GB high bandwidth memory and the TensorFlow Keras framework and the training of the networks for multi-class classification took around 30 minutes on Intel MobileODT Cervical Cancer Dataset. While training the models, we have considered 50 steps per epoch, the number of epochs as 300 and 512 as the batch size. Similar to the training setting used previously. We set the learning rate to 0.0001 and allow the models to train using the SGD algorithm with 0.9 as momentum. As a part of the data pipeline for model training, the training images were resized to 299 x 224, applied RandAugment with m=8 and n=2, and further randomly cropped into images of size 224 x 224. During testing and evaluation, images are centrally cropped to 224 x 224 size.

We have done the comparative analysis between various pre-trained networks with our proposed approach on a multi-class imbalanced dataset containing cervigram images in order to detect different cervix types helpful for cancer screening tasks. The models used for comparison are trained on a large-scale ImageNet dataset and further pre-trained on the cervix dataset similar to the training setting for our proposed VGGIN-Net model. Table 5.11. illustrates an analysis between our proposed VGGIN-Net model and other state-of-the-art CNN architectures such as VGG16, InceptionV3, ResNet50, ResNet50V2, Xception, InceptionResNetV2, DenseNet121, EfficientNet-B0. Imbalanced measures such as Precision, Recall, F1 Score, Index Balanced Accuracy, and Geometric Mean for each class are used to determine which approach is able to tackle imbalance to a greater extent. Our analysis shows that among all the models VGGIN-Net is able to perform quite better and gives significant performance improvement over other pre-trained models. In Table 5.12. We have done a comparison of our proposed VGGIN-Net approach with other state-of-the-art pre-

trained models even without applying the rejection resampling technique by using imbalanced evaluation measures. It was observed that there is a significant drop in the results which proves that application of rejection resampling is important for the given classification task. Further, we have conducted the ablation study to show the efficacy of different hyperparameters chosen in our proposed approach as shown in Tables 5.13 (i) and (ii). Additionally, we have shown the efficacy of transfer learning in our proposed approach by tabulating the results of different CNN models trained from scratch. Due to the smaller dataset having fewer samples of images, visual similarities present in various classes, and a multi-class imbalanced dataset, this classification task was challenging to deal with. To conclude, here in this work we have proposed an approach along with the data augmentation and random crop and rejection resampling techniques to combat the challenges faced by multi-class imbalanced classification tasks. We have done a comparative analysis of various pre-trained networks with a cervical cancer dataset on the basis of various evaluation metrics such as accuracy, precision, recall, F1 score, geometric mean and index balanced accuracy. The experimental results show that the proposed approach is demonstrating better results than compared to pre-trained networks.

Table 5.11. Comparison of proposed VGGIN-Net Approach with other state-of-the-art pre-trained models using imbalanced evaluation measures.

Model	Accuracy	Precision				Recall				F1 Score				Index Balanced Accuracy				Geometric Mean			
		Type 1	Type 2	Type 3	Avg	Type 1	Type 2	Type 3	Avg	Type 1	Type 2	Type 3	Avg	Type 1	Type 2	Type 3	Avg	Type 1	Type 2	Type 3	Avg
VGG16	0.71	0.59	0.73	0.76	0.72	0.63	0.77	0.66	0.71	0.61	0.75	0.71	0.71	0.56	0.54	0.59	0.56	0.76	0.73	0.77	0.75
InceptionV3	0.71	0.64	0.75	0.68	0.71	0.67	0.72	0.71	0.71	0.65	0.74	0.7	0.71	0.6	0.54	0.59	0.56	0.78	0.73	0.78	0.75
ResNet50V2	0.73	0.65	0.75	0.74	0.73	0.59	0.8	0.69	0.73	0.62	0.77	0.72	0.73	0.53	0.57	0.6	0.58	0.74	0.75	0.79	0.76
Xception	0.59	0.39	0.74	0.61	0.64	0.62	0.48	0.77	0.59	0.48	0.58	0.68	0.6	0.49	0.38	0.6	0.47	0.7	0.63	0.77	0.69
InceptionResnetV2	0.63	0.48	0.72	0.63	0.65	0.66	0.58	0.71	0.63	0.55	0.65	0.67	0.64	0.55	0.44	0.57	0.5	0.75	0.67	0.76	0.71

DenseNet121	0.72	0.61	0.74	0.75	0.72	0.62	0.78	0.67	0.72	0.62	0.76	0.71	0.72	0.55	0.55	0.59	0.56	0.76	0.74	0.78	0.75
EfficientNet-B0	0.56	0.37	0.66	0.62	0.6	0.63	0.49	0.64	0.56	0.47	0.56	0.63	0.57	0.49	0.35	0.51	0.42	0.7	0.6	0.72	0.65
VGGIN-Net	**0.75**	**0.74**	**0.76**	**0.73**	**0.75**	**0.66**	**0.8**	**0.7**	**0.75**	**0.7**	**0.78**	**0.72**	**0.74**	**0.61**	**0.58**	**0.61**	**0.59**	**0.79**	**0.76**	**0.79**	**0.77**

Table 5.12. Comparison of proposed VGGIN-Net Approach with other state-of-the-art pre-trained models without applying rejection resampling using imbalanced evaluation measures.

Model Accuracy		**Precision**				**Recall**				**F1 Score**				**Index Balanced Accuracy**				**Geometric Mean**			
		Type 1	**Type 2**	**Type 3**	**Avg**	**Type 1**	**Type 2**	**Type 3**	**Avg**	**Type 1**	**Type 2**	**Type 3**	**Avg**	**Type 1**	**Type 2**	**Type 3**	**Avg**	**Type 1**	**Type 2**	**Type 3**	**Avg**
VGG16	0.71	0.65	0.74	0.72	0.71	0.68	0.77	0.64	0.71	0.66	0.75	0.68	0.71	0.61	0.55	0.56	0.56	0.79	0.74	0.75	0.75
InceptionV3	0.72	0.7	0.71	0.76	0.73	0.6	0.83	0.61	0.72	0.65	0.77	0.68	0.72	0.55	0.54	0.54	0.54	0.75	0.73	0.75	0.74
ResNet50V2	0.72	0.69	0.73	0.74	0.72	0.55	0.83	0.65	0.72	0.61	0.77	0.69	0.72	0.5	0.56	0.57	0.55	0.72	0.74	0.76	0.75
Xception	0.6	0.53	0.6	0.61	0.59	0.28	0.77	0.47	0.6	0.36	0.68	0.53	0.58	0.24	0.36	0.39	0.35	0.51	0.59	0.64	0.59
InceptionResnetV2	0.67	0.69	0.66	0.7	0.68	0.4	0.84	0.54	0.67	0.51	0.74	0.61	0.66	0.37	0.46	0.47	0.45	0.62	0.67	0.7	0.67
DenseNet121	0.74	0.76	0.72	0.8	0.75	0.59	0.87	0.62	0.74	0.66	0.79	0.7	0.74	0.54	0.57	0.56	0.56	0.75	0.74	0.76	0.75
EfficientNet-B0	0.66	0.57	0.65	0.69	0.65	0.33	0.82	0.57	0.66	0.42	0.73	0.63	0.64	0.3	0.45	0.49	0.44	0.56	0.66	0.71	0.66
VGGIN-Net	**0.71**	**0.83**	**0.68**	**0.73**	**0.72**	**0.6**	**0.84**	**0.56**	**0.71**	**0.69**	**0.75**	**0.63**	**0.7**	**0.56**	**0.49**	**0.49**	**0.5**	**0.76**	**0.69**	**0.71**	**0.71**

Table 5.13. (a) Ablation experiments to determine the veracity of the proposed VGGIN-Net approach.

Model Accuracy		**Precision**				**Recall**				**F1 Score**				**Index Balanced Accuracy**				**Geometric Mean**			
		Type 1	**Type 2**	**Type 3**	**Avg**	**Type 1**	**Type 2**	**Type 3**	**Avg**	**Type 1**	**Type 2**	**Type 3**	**Avg**	**Type 1**	**Type 2**	**Type 3**	**Avg**	**Type 1**	**Type 2**	**Type 3**	**Avg**
Proposed Approach	**0.75**	**0.74**	**0.76**	**0.73**	**0.75**	**0.66**	**0.8**	**0.7**	**0.75**	**0.7**	**0.78**	**0.72**	**0.74**	**0.61**	**0.58**	**0.61**	**0.59**	**0.79**	**0.76**	**0.79**	**0.77**
Proposed Approach with multiple Inception blocks	0.75	0.76	0.76	0.72	0.75	0.67	0.8	0.72	0.75	0.71	0.78	0.72	0.75	0.62	0.59	0.62	0.6	0.8	0.76	0.79	0.78
Proposed Approach with Adam	0.72	0.67	0.73	0.73	0.72	0.71	0.78	0.62	0.72	0.69	0.76	0.67	0.72	0.65	0.65	0.54	0.56	0.81	0.74	0.75	0.75
Proposed Approach with SGDR	0.74	0.7	0.74	0.78	0.74	0.63	0.83	0.65	0.74	0.66	0.78	0.71	0.74	0.58	0.58	0.58	0.58	0.77	0.76	0.77	0.76

Table 5.13 (b) Ablation experiments to compare with and without transfer learning of various pre-trained networks.

Model Accuracy	**Precision**	**Recall**	**F1 Score**	**Index Balanced Accuracy**	**Geometric Mean**

		Type 1	Type 2	Type 3	Avg	Type 1	Type 2	Type 3	Avg	Type 1	Type 2	Type 3	Avg	Type 1	Type 2	Type 3	Avg	Type 1	Type 2	Type 3	Avg
VGG16 (w/o Transfer Learning)	0.40	0.30	0.0	0.47	0.2	0.71	0.0	0.89	0.4	0.42	0.0	0.61	0.26	0.47	0.0	0.5	0.24	0.68	0.0	0.7	0.33
InceptionV3 (w/o Transfer Learning)	0.39	0.28	0.0	0.5	0.2	0.8	0.0	0.82	0.39	0.42	0.0	0.62	0.26	0.48	0.0	0.52	0.24	0.68	0.0	0.72	0.33
ResNet50V2 (w/o Transfer Learning)	0.38	0.29	0.0	0.46	0.19	0.74	0.0	0.83	0.38	0.41	0.0	0.59	0.26	0.47	0.0	0.48	0.23	0.68	0.0	0.68	0.33
EfficientNet-B0 (w/o Transfer Learning)	0.36	0.26	0.0	0.46	0.19	0.77	0.0	0.72	0.36	0.38	0.0	0.57	0.24	0.43	0.0	0.45	0.21	0.65	0.0	0.67	0.32
VGGIN-Net (w/o Transfer Learning)	0.39	0.35	0.0	0.41	0.19	0.69	0.0	0.88	0.39	0.46	0.0	0.56	0.25	0.51	0.0	0.4	0.21	0.71	0.0	0.56	0.25
VGG16 (w/ Transfer Learning)	0.71	0.59	0.73	0.76	0.72	0.63	0.77	0.66	0.71	0.61	0.75	0.711	0.71	0.56	0.54	0.59	0.56	0.76	0.73	0.77	0.75
InceptionV3 (w/ Transfer Learning)	0.71	0.64	0.75	0.68	0.71	0.67	0.72	0.71	0.71	0.65	0.74	0.7	0.71	0.6	0.54	0.59	0.56	0.78	0.73	0.78	0.75
ResNet50V2 (w/ Transfer Learning)	0.73	0.65	0.75	0.74	0.73	0.59	0.8	0.69	0.73	0.62	0.77	0.72	0.73	0.53	0.57	0.6	0.58	0.74	0.75	0.79	0.76
EfficientNet-B0 (w/ Transfer Learning)	0.56	0.37	0.66	0.62	0.6	0.63	0.49	0.64	0.56	0.47	0.56	0.63	0.57	0.49	0.35	0.51	0.42	0.7	0.6	0.72	0.65
VGGIN-Net (w/ Transfer Learning)	**0.75**	**0.74**	**0.76**	**0.73**	**0.75**	**0.66**	**0.8**	**0.7**	**0.75**	**0.7**	**0.78**	**0.72**	**0.74**	**0.61**	**0.58**	**0.61**	**0.59**	**0.79**	**0.76**	**0.79**	**0.77**

5.2 Limitations

The proposed VGGIN-Net has demonstrated excellent performance in image classification tasks. However, it was primarily designed for binary and multi-class classification problems, specifically in distinguishing between different types of cancer in biomedical datasets. One limitation associated with VGGIN-Net is to explore the performance and generalizability of VGGIN-Net on different types of biomedical datasets, particularly in various cancer classification tasks. This limitation needs to be addressed in order to explore the potential of VGGIN-Net for multi-class challenges, ultimately improving its overall generalizability and performance across diverse biomedical datasets.

Chapter 6
Summary and Conclusions

In this chapter we have presented the overall conclusion of the thesis derived from different experimental results. This section is presented in two parts: a list of conclusions and the curated set of research contributions which are based upon our work on the design and development of deep learning architectures to address the class imbalance problem described in various chapters above. Further, we have added the future scope of our research as a separate subsection in this chapter to briefly describe the future direction and scope of the conducted research work.

6.1. Conclusion

In the real-world, many machine learning problems are prone to the curse of class imbalance. The most popular applications in this field are generally around fraud detection or are usually native to the biomedical domain. In the case of fraud detection and other imbalanced applications, we deal with supervised learning problems for structured data, text, etc. However, for image classification problems, we naturally found many datasets that belong to the biomedical domain and have a sufficiently high number of samples for training deep models which motivated us to work with such datasets for most of our work apart from the natural scene features dataset consisting of images of objects. Within the biomedical domain itself, we could find different datasets having different imbalanced ratios for both multi-class and binary problems for classification. Interestingly, both object detection and segmentation problems were also inherently found to be highly imbalanced in nature

due to the large number of background pixels outnumbering the other foreground pixels.

We have used a variety of different data augmentation strategies for tackling the class imbalance problem. We found that data augmentation is a regularization technique that can be used to resolve the class imbalance scenario to an extent. Applying data augmentation to minority classes is well suited for the minority class where samples are low as it is easy to balance the distribution by nearly creating the artificial samples using synthetic generation. For larger datasets, applying data augmentation on both classes is more suitable as augmentation can be used directly in the pipeline at a mini-batch level which helps the model to learn different variations within the dataset for imbalance settings and is much more suitable for large datasets. In case the dataset is large and highly imbalanced, we can use data augmentation on both the classes as well as use random undersampling at a mini-batch level to help solve the imbalance problem.

However there are certain limitations that come in while using deep learning-based systems as deep learning comes at the expense of computational resources, and it can be difficult to use deep learning models on the edge, like IoT devices, embedded chips, etc. Specialized hardware specific to ML applications would be required in such scenarios. Deep learning and neural networks in general try to capture the different variations in the trained dataset and given task, for applications where the data input to the model doesn't have many variations, deep learning would be overkill. Extremely vast amount of training data is required to train models that are able to

yield good performance. Transfer learning serves as a solution but it can be challenging to find pre-trained models for specific applications.

Some of the major conclusions that can be drawn from the study are listed as follows:

i. Deep features extracted appear to be more relevant in comparison to the other traditional features as it shows tremendous improvement in the performance of the Bag-of-visual-words approach.

ii.Accuracy cannot always be considered the right evaluation metric in case of an imbalanced dataset. There is a need to consider other evaluation metrics in order to gauge the actual performance of any model as considering accuracy as the only evaluation measure can many times lead to false results.

iii. Transfer learning is appropriate in order to transfer the knowledge from the larger dataset to a smaller targeted dataset. It will save the training time to train any network from scratch and improve the performance and generalizability of the model.

iv. Explored the effect of applying the data augmentation (i) Traditional affine transformation (shifted, zoomed in /out, rotated, flipped, distorted, cropping, rescaling or shaded with hue, etc.) (ii) Generative Adversarial Nets (GANs) to generate synthetic samples from the original images which will help to train the models to

become more robust by seeing more synthetic created samples and also helps in resolving class imbalance issues.

v. Applying different regularization techniques like dropout and data augmentation helps reduce overfitting to a great extent even for highly imbalanced datasets.

vi. Created the novel deep learning architecture VGGIN-Net by combining the appropriate layers from VGG16 architecture with naive Inception block and a single dense (fully connected) layer to proposed network architecture to deal with the higher-end linear features and to find the probability of occurrence of the image belonging to each class for the classification problem. Further, batch normalization, flatten, and dropout layers are included to improvise the network performance.

vii. The DenseNet121 pre-trained model is well suited for diabetic retinopathy image classification. The EfficientDet-D0 and SSD (MobileNetV1) are best suited for object detection on diabetic retinopathy datasets. In the case of segmentation, PSPNet (with focal loss) is working best in comparison to other pre-trained networks.

viii. VGGIN-Net network is also able to work well on multi-class imbalanced datasets apart from binary classification datasets which is empirically evaluated with the help of applying the same model to the Cervical cancer screening dataset.

x. VGGIN-Net can be deemed to be a novel deep transfer network obtained by the fusion of VGG16 and Inception architectures which works quite well on multiple classification tasks from the biomedical domain.

xi. Application of RandAugment and other automated data augmentation techniques can be used to rapidly simplify the data pre-processing pipeline used for deep network training which could lead to quite better model performance with a certain amount of parameter tuning.

xii. For highly imbalanced datasets, especially the ones which are present in the biomedical domain, rejection resampling or mini-batch level random oversampling serves as a simple yet effective method to tackle class imbalance as the training distribution gets balanced.

6.2. Research Contributions

The research contributions are listed briefly as follows:

i. Analyzed and explored the relevance of using the data augmentation operations to the minority class only to enhance the efficiency of the network architecture instead of applying to both the classes (majority and minority).

ii. Assessed the significance of applying pre-trained deep neural networks for Image classification, object detection, and classification tasks using imbalanced datasets.

iii. Development of unique combinations of deep learning architecture to handle the class imbalance problem for multi-class and binary class and imbalanced datasets.

iv. Successfully applied transfer learning approach to extract deep features using pre-trained models, for generating the visual codebook in our improved Bag-of-Visual-Words model.

v. Conducted an extensive empirical analysis to find the optimal set of deep features for codebook generation and the appropriate classifier for class imbalance problem.

vi. Analyzing the role of Chi-Squared kernelized SVM as an effective classifier for the histogram features.

vii. Successfully created a novel deep transfer model using DCGAN as a data augmentation technique for cancer-related biomedical imbalanced datasets for images of varying magnification factors

viii. Mitigated the effect of covariance shift by using the Batch Normalization layer which effectively normalizes data inputs tackling imbalanced situations.

ix. Proposed a novel approach using transfer learning by transferring the knowledge from the source ImageNet object dataset to the target Breast cancer dataset effectively.

x. Analyzed the proposed transfer learning approaches with respect to other state-of-the-art networks for distinguishing the Benign samples from Malignant in case of the imbalanced dataset.

xi. Analyzed the effect of weighted SVM in comparison to different SVM variants when applied to extracted features from the images.

xii. Emphasis on the aspect that accuracy is not the only evaluation metric to check the efficacy of deep learning architecture on imbalanced datasets. There are other evaluation metrics that need to be considered while measuring the imbalanced dataset such as Precision, Recall, Receiver Operating Characteristics (ROC) analysis, Area Under the curve (AUC), Mathew's correlation, and Cohen's kappa coefficient.

xiii. Analyzed the significance of SGDR optimizer in different networks. From the analysis, it was found that the proposed approach is more stable in comparison to other approaches using SGDR optimizer, which further leads to improved convergence.

xiv. Analyzed the effect of applying fine-tuning by retraining pre-trained networks on the deep learning architectures.

xv. Emphasized research on the medical diabetic retinopathy imbalanced datasets for classification, object detection, and segmentation tasks together as a single unit by conducting extensive comparative analysis between various state-of-the-art pre-trained models on varied size datasets.

xvi. Effective classification of cervigram images used for cancer screening to detect different cervix types with the proposed VGGIN-Net model, which proves the efficacy of the model even on multi-class imbalanced datasets.

xvii. Society could be benefitted through research in biomedical systems and models for detecting and screening different diseases because most of our work is focused in the biomedical domain and inherently, we find that problems in biomedical are also highly imbalanced in nature.

6.3. Future Scope

This section discusses the different limitations in our current research work followed by the future scope of the work ahead. Object detection and segmentation tasks are more challenging for multi-class imbalanced datasets than classification tasks. Thus, further research will need to be carried out in the creation of novel deep learning architectures for object detection and segmentation. Also, new variants of deep learning architectures can be designed to deal with imbalanced scenarios for various other challenging real-world problems apart from natural scene features, objects, and or biomedical domain. We shall also create different deep learning architectures in the future for a multi-class imbalanced dataset for object detection and segmentation by exploring the effect and usage of focal loss in the creation of

deep learning to handle class imbalance problems. We would also look into constructing other deep network architectures as well as explore more advanced pre-trained network architectures combining them with custom layers which would have significance in the overall healthcare sector. Additionally, we shall be developing a cloud-based framework where the deep neural network architectures with the best performance could be easily accessible to diagnose diseases accurately so that disease can be tackled at an early stage by automatic deep learning and computer vision-based approaches.

Table 6.1: Summarization of results in thesis

Chapter	**Proposed method and its performance**	**Database Used**	**Classifier used**	**Comparison Methods**	**Highlights/ Novelties**
Chapter 3	ResNet-50 Deep Feature Extraction for Bag-of-Visual-Words codebook generation with Chi² SVM Classification	-Graz-02 -TF-Flowers	- Chi² SVM - Quasi SVM	➢VGG16 ➢Inception-V3 ➢ResNet-50 ➢SIFT + BOVW ➢VGG16 + BOVW ➢Inception-V3 + BOVW ➢ResNet-50 BOVW Features +Logistic Regression ➢ResNet-50 BOVW Features + Linear Discriminative Analysis ➢ResNet-50 BOVW Features	➢Designed to tackle the challenges associated with multi-class imbalanced datasets. ➢Features are extracted from residual block of the fifth layer just preceding the global average pooling and dense layer of pre-trained model ResNet-50, defined as the Res5c features are most suitable deep features

				+KNearest Neighbors Classifier ➢ResNet-50 BOVW Features + Decision Tree ➢ResNet-50 BOVW Features +Gaussian Naïve Bayes	incorporated in the proposed approach. ➢ non-linear layer Chi^2 SVM using the one-versus-all scheme is found to be an optimal choice of classifier ➢Alternative approach suitable for large-scale settings is proposed with the help of Quasi SVMs constructed with the help of neural networks. This approach vouches for the scalability of the proposed BOVW-based approach on data expensive scenarios as well.
Chapter 3	Imbalanced diabetic retinopathy detection problem for classification, segmentation and object detection tasks.	- IDRiD - DDR - Kaggle Diabetic Retinopathy		➢VGG16 ➢VGG19 ➢InceptionV3 ➢ResNet50 ➢ResNet50V2 ➢ResNet152 ➢ResNet101 ➢ResNet152V2 ➢ResNet101V2 ➢Xception ➢InceptionResNetV2 ➢MobileNetV2 ➢DenseNet121 ➢DenseNet169 ➢DenseNet201 ➢EfficientNetB0 ➢EfficientDet-D0 ➢Faster RCNN (ResNet-50) ➢SSD (MobileNetV1) ➢RetinaNet (ResNet50)	➢DenseNet121 is the best suited for the diabetic retinopathy image classification task. ➢EfficientDet-D0 and SSD (MobileNetV1) are best suited based on the diabetic retinopathy dataset for object detection tasks. ➢In case of segmentation, PSPNet (with focal loss) performs best in comparison to other pre-trained networks. ➢It was also observed experimentally that in the case of Class 1, early-stage diabetics is difficult to detect

				➢SSD (MobileNetV2) ➢PSPNet (w/ Focal Loss) ➢DeepLab v2 (w/ Focal Loss) ➢DeepLab v3 (w/ Focal Loss) ➢PSPNet (w/ Cross Entropy Loss) ➢DeepLab v2 (w/ Cross Entropy Loss) ➢DeepLab v3 (w/ Cross Entropy Loss)	irrespective of whether that class falls under the minority category in the case of all the three available diabetic retinopathy datasets.
Chapter 4	➢Proposed novel deep transfer network in collaboration with Deep Convolutional Generative Adversarial network (DCGAN). ➢Proposed Network (w/ BatchNormalization and w/ DCGAN samples.	- BreakHis		➢BOVW ➢CNN ➢VGG16 ➢InceptionV3 ➢ResNet50 ➢VGG16 + Linear SVM ➢InceptionV3 + Linear SVM ➢ResNet50 + Linear SVM ➢VGG16 + RBF SVM ➢InceptionV3 + RBF SVM ➢ResNet50 + RBF SVM ➢Proposed Network (w/o Batch Normalization) ➢Proposed Network (w/o Batch Normalization and w/ DCGAN samples)	➢The proposed approach works well for cancer-related biomedical imbalanced datasets for various magnification factors: 40X, 100X, 200X, and 400X. ➢Mitigate the effect of covariant shift by using the Batch Normalization ➢Proposed transfer learning approach effectively transfer the knowledge from the source ImageNet object dataset to the target Breast cancer dataset effectively ➢Proposed approach is more stable approaches using SGDR optimizer with hyper parameter tuning, which further leads to improved

					convergence and have improved the consistency of the learning curve for the last few sets of epochs.
Chapter 4	Proposed Inception-V3 + weighted SVM with data augmentation on minority class	- BreakHis - Breast Histopathological	- Weighted SVM	➢Inception-V3 ➢Inception-V3 (with data augmentation on both class) ➢Inception-V3 (with data augmentation on minority class) ➢ResNet-50 ➢ResNet-50 (with data augmentation on both class) ➢ResNet-50 (with data augmentation on minority class) ➢Inception-V3 + SVM	➢Pre-trained Inception-V3 model with data augmentation on minority class outperforms other network types. ➢ Inception-V3 with data augmentation of minority class and transfer learning with weighted SVM gives overall better performance.
Chapter 5	VGGIN-Net	-BreakHis - Breast Histopathological -Intel MobileODT Cervical Cancer Screening	VGG16 pre-trained (layers till block 4 pool layer) along with the naive Inception module in combination with flatten, batch normalization and dense layer.	➢ VGG16 ➢ GoogLeNet ➢ ResNet50 ➢Modified VGG16 w/ Single Dense Layer ➢Modified VGG16 w/ Inception Block w/ Single Dense Layer ➢ Spanhol, et al. 2016 ➢ Spanhol, et al. 2017 ➢ Bayramoglu, et al. 2016 ➢ Zhu, et al. 2019 ➢ Gupta and Bhavsar 2017 ➢ Deniz, et al. 2018 ➢ Song, et al. 2017	➢ Proposed novel deep learning architecture, VGGIN-Net, can be employed for various binary-class and multi-class problems. ➢ In this work, the goodness of both the pre-trained models (VGG16 and Inception) to are considered create a more robust architecture that effectively resolves the class imbalance problem. ➢ Further, batch normalization, flatten, and dropout layers are

				➢ Gupta and Bhavsar 2018 ➢VGG16 ➢InceptionV3 ➢ResNet50V2 ➢Xception ➢InceptionResnet V2 ➢DenseNet121 ➢EfficientNet-B0 ➢VGG16 (w/o Transfer Learning) ➢InceptionV3 (w/o Transfer Learning) ➢ResNet50V2 (w/o Transfer Learning) ➢EfficientNet-B0 (w/o Transfer Learning) ➢VGG16 (w/ Transfer Learning) ➢InceptionV3 (w/ Transfer Learning) ➢ResNet50V2 (w/ Transfer Learning) ➢EfficientNet-B0 (w/ Transfer Learning) ➢VGGIN-Net (w/o Transfer Learning)	added to enhance the network performance. ➢ To improve the regularization of proposed network the random crop was applied which helps the network to learn even better due to the translation invariance property of convolutional networks ➢ Block-wise fine-tuning operations have shown significant improvement in performance. It is evident that different fine-tuning combinations are found suitable for different magnification factors. For 40X, fine-tuning of block3, block4, and Inception block seems to be an ideal choice, whereas, in the case of 400X, fine-tuning of block4 and Inception block was only found to be the perfect fit. Fine-tuning of the complete network was found to be ideal in the case of 100X and 200X magnification factor images. ➢Transfer learning of the features from a large dataset to a

					small cervical cancer multi-class imbalanced dataset and data augmentation was applied successfully which has an overall impact on the deep learning architectures ➢RandAugment approach for data augmentation was applied for multi-class imbalanced tas have significant impact in the overall performance. ➢ Proposed approach along with the data augmentation and random crop and rejection resampling techniques to combat the challenges faced by multi-class imbalanced classification tasks.

REFERENCES

Abadi, Martín, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 'TensorFlow: A system for large-scale machine learning'. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, 265–83, 2016.

Abbas, Asmaa, Mohammed M. Abdelsamea, and Mohamed Medhat Gaber. 'Detrac: Transfer learning of class decomposed medical images in convolutional neural networks'. *IEEE Access* 8 (2020): 74901–13.

Antoniou, Antreas, Amos Storkey, and Harrison Edwards. 'Data augmentation generative adversarial networks'. *arXiv preprint arXiv:1711. 04340*, 2017.

Aravind, Krishnaswamy R., Purushothaman Raja, Rajendran Ashiwin, and Konnaiyar V. Mukesh. 'Disease classification in Solanum melongena using deep learning'. *Spanish Journal of Agricultural Research* 17, no. 3 (2019): e0204–e0204.

Baheti, Bhakti, Suhas Gajre, and Sanjay Talbar. 'Detection of distracted driver using convolutional neural network'. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 1032–38, 2018.

Bayramoglu, Neslihan, Juho Kannala, and Janne Heikkilä. 'Deep learning for magnification independent breast cancer histopathology image classification'. In *2016 23rd International conference on pattern recognition (ICPR)*, 2440–45. IEEE, 2016.

Bellet, Aurélien, Amaury Habrard, and Marc Sebban. 'A survey on metric learning for feature vectors and structured data'. *arXiv preprint arXiv:1306. 6709*, 2013.

Bellet, Aurélien, Amaury Habrard, and Marc Sebban. 'Metric learning'. *Synthesis lectures on artificial intelligence and machine learning* 9, no. 1 (2015): 1–151.

Bengio, Yoshua, and Others. 'Learning deep architectures for AI'. *Foundations and trends® in Machine Learning* 2, no. 1 (2009): 1–127.

Bosch, Anna, Andrew Zisserman, and Xavier Munoz. 'Image classification using random forests and ferns'. In *2007 IEEE 11th international conference on computer vision*, 1–8. Ieee, 2007.

Chatzimparmpas, Angelos, Fernando V. Paulovich, and Andreas Kerren. 'HardVis: Visual Analytics to Handle Instance Hardness Using Undersampling and Oversampling Techniques'. *arXiv preprint arXiv:2203. 15753*, 2022.

Chen, Liang-Chieh, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. 'Deeplab: Semantic image segmentation with deep convolutional nets,

atrous convolution, and fully connected crfs'. *IEEE transactions on pattern analysis and machine intelligence* 40, no. 4 (2017): 834–48.

Cheng, Gong, Zhenpeng Li, Xiwen Yao, Lei Guo, and Zhongliang Wei. 'Remote sensing image scene classification using bag of convolutional features'. *IEEE Geoscience and Remote Sensing Letters* 14, no. 10 (2017): 1735–39.

Chollet, Francois. *Deep learning with Python*. Simon and Schuster, 2021.

Chollet, François. 'Xception: Deep learning with depthwise separable convolutions'. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1251–58, 2017.

Cortes, Corinna, and Vladimir Vapnik. 'Support-vector networks'. *Machine learning* 20, no. 3 (1995): 273–97.

Costa Oliveira, Artur Leandro da, Andre Britto de Carvalho, and Daniel Oliveira Dantas. 'Faster R-CNN Approach for Diabetic Foot Ulcer Detection'. In *VISIGRAPP (4: VISAPP)*, 677–84, 2021.

Cruz-Roa, Angel, Ajay Basavanhally, Fabio González, Hannah Gilmore, Michael Feldman, Shridar Ganesan, Natalie Shih, John Tomaszewski, and Anant Madabhushi. 'Automatic detection of invasive ductal carcinoma in whole slide images with convolutional neural networks'. In *Medical Imaging 2014: Digital Pathology*, 9041:904103. SPIE, 2014.

Cubuk, Ekin D., Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. 'Autoaugment: Learning augmentation strategies from data'. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 113–23, 2019.

Deniz, Erkan, Abdulkadir Şengür, Zehra Kadiroğlu, Yanhui Guo, Varun Bajaj, and Ümit Budak. 'Transfer learning based histopathologic image classification for breast cancer detection'. *Health information science and systems* 6, no. 1 (2018): 1–7.

Eyepacs. 'Diabetic retinopathy detection'. *Kaggle*. EyePACS, 2015. https://www.kaggle.com/c/diabetic-retinopathy-detection.

Feng, Jiangfan, Yuanyuan Liu, and Lin Wu. 'Bag of visual words model with deep spatial features for geographical scene classification'. *Computational intelligence and neuroscience* 2017 (2017).

Fernández, Alberto, Salvador García, Mikel Galar, Ronaldo C. Prati, Bartosz Krawczyk, and Francisco Herrera. *Learning from imbalanced data sets*. vol. 10. Springer, 2018.

Frid-Adar, Maayan, Idit Diamant, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. 'GAN-based synthetic medical image augmentation for

increased CNN performance in liver lesion classification'. *Neurocomputing* 321 (2018): 321–31.

García, Vicente, Ramón Alberto Mollineda, and José Salvador Sánchez. 'Index of balanced accuracy: A performance measure for skewed class distributions'. In *Iberian conference on pattern recognition and image analysis*, 441–48. Springer, 2009.

Garud, Hrushikesh, Sri Phani Krishna Karri, Debdoot Sheet, Jyotirmoy Chatterjee, Manjunatha Mahadevappa, Ajoy K. Ray, Arindam Ghosh, and Ashok K. Maity. 'High-magnification multi-views based classification of breast fine needle aspiration cytology cell samples using fusion of decisions from deep convolutional networks'. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 76–81, 2017.

Geng, Cong, and Xudong Jiang. 'SIFT features for face recognition'. In *2009 2nd IEEE international conference on computer science and information technology*, 598–602. IEEE, 2009.

Georgescu, Mariana-Iuliana, Radu Tudor Ionescu, and Marius Popescu. 'Local learning with deep and handcrafted features for facial expression recognition'. *IEEE Access* 7 (2019): 64827–36.

Géron, Aurélien. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. ' O'Reilly Media, Inc.', 2019.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 'Generative adversarial nets'. *Advances in neural information processing systems* 27 (2014).

Gradstein, Jonathan, Moshe Salhov, Yoav Tulpan, Ofir Lindenbaum, and Amir Averbuch. 'Imbalanced Classification via a Tabular Translation GAN'. *arXiv preprint arXiv:2204. 08683*, 2022.

Gui, Xingtai, and Jiyang Zhang. 'Deep metric learning model for imbalanced fault diagnosis'. *arXiv preprint arXiv:2107. 03786*, 2021.

Gupta, Vibha, and Arnav Bhavsar. 'Breast cancer histopathological image classification: is magnification important?' In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 17–24, 2017.

Gupta, Vibha, Arnav Bhavsar. 'Sequential modeling of deep features for breast cancer histopathological image classification'. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2254–61, 2018.

Hagos, Misgina Tsighe, and Shri Kant. 'Transfer learning based detection of diabetic retinopathy from small dataset'. *arXiv preprint arXiv:1905. 07203*, 2019.

Haralick, Robert M., and Linda G. Shapiro. 'Image segmentation techniques'. *Computer vision, graphics, and image processing* 29, no. 1 (1985): 100–132.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 'Deep residual learning for image recognition'. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–78, 2016.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 'Identity mappings in deep residual networks'. In *European conference on computer vision*, 630–45. Springer, 2016.

Hou, Qibin, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip H. S. Torr. 'Deeply supervised salient object detection with short connections'. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3203–12, 2017.

Howard, Andrew G. 'Some improvements on deep convolutional neural network based image classification'. *arXiv preprint arXiv:1312. 5402*, 2013.

Huang, Chaolin, Yeming Wang, Xingwang Li, Lili Ren, Jianping Zhao, Yi Hu, Li Zhang, et al. 'Clinical features of patients infected with 2019 novel coronavirus in Wuhan, China'. *The lancet* 395, no. 10223 (2020): 497–506.

Huang, Jonathan, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, et al. 'Speed/accuracy trade-offs for modern convolutional object detectors'. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7310–11, 2017.

Iandola, Forrest, Matt Moskewicz, Sergey Karayev, Ross Girshick, Trevor Darrell, and Kurt Keutzer. 'Densenet: Implementing efficient convnet descriptor pyramids'. *arXiv preprint arXiv:1404. 1869*, 2014.

Iandola, Forrest N., Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, and Kurt Keutzer. 'SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size'. *arXiv preprint arXiv:1602. 07360*, 2016.

Intel, Mobileodt. 'Intel & MobileODT Cervical Cancer Screening', 2017. https://www.kaggle.com/competitions/intel-mobileodt-cervical-cancer-screening/data.

Ioffe, Sergey, and Christian Szegedy. 'Batch normalization: Accelerating deep network training by reducing internal covariate shift'. In *International conference on machine learning*, 448–56. PMLR, 2015.

Japkowicz, Nathalie, and Shaju Stephen. 'The class imbalance problem: A systematic study'. *Intelligent data analysis* 6, no. 5 (2002): 429–49.

Japkowicz, Nathalie. 'Assessment metrics for imbalanced learning'. *Imbalanced learning: Foundations, algorithms, and applications*, 2013, 187–206.

Kassani, Sara Hosseinzadeh, Peyman Hosseinzadeh Kassani, Reza Khazaeinezhad, Michal J. Wesolowski, Kevin A. Schneider, and Ralph Deters. 'Diabetic retinopathy classification using a modified xception architecture'. In *2019 IEEE international symposium on signal processing and information technology (ISSPIT)*, 1–6. IEEE, 2019.

Kocaman, Veysel, Ofer M. Shir, and Thomas Bäck. 'Improving model accuracy for imbalanced image classification tasks by adding a final batch normalization layer: An empirical study'. In *2020 25th International Conference on Pattern Recognition (ICPR)*, 10404–11. IEEE, 2021.

Kramer, Oliver. 'Scikit-learn'. In *Machine learning for evolution strategies*, 45–53. Springer, 2016.

Krawczyk, Bartosz, Michał Woźniak, and Gerald Schaefer. 'Cost-sensitive decision tree ensembles for effective imbalanced classification'. *Applied Soft Computing* 14 (2014): 554–62.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. 'Imagenet classification with deep convolutional neural networks'. *Advances in neural information processing systems* 25 (2012).

Kumar, Abhinav, Sanjay Kumar Singh, Sonal Saxena, K. Lakshmanan, Arun Kumar Sangaiah, Himanshu Chauhan, Sameer Shrivastava, and Raj Kumar Singh. 'Deep feature learning for histopathological image classification of canine mammary tumors and human breast cancer'. *Information Sciences* 508 (2020): 405–21.

Lam, Carson, Darvin Yi, Margaret Guo, and Tony Lindsey. 'Automated detection of diabetic retinopathy using deep learning'. *AMIA summits on translational science proceedings* 2018 (2018): 147.

LeCun, Yann, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, and Lawrence D. Jackel. 'Backpropagation applied to handwritten zip code recognition'. *Neural computation* 1, no. 4 (1989): 541–51.

Li, Tao, Yingqi Gao, Kai Wang, Song Guo, Hanruo Liu, and Hong Kang. 'Diagnostic assessment of deep learning algorithms for diabetic retinopathy screening'. *Information Sciences* 501 (2019): 511–22.

Lin, Tsung-Yi, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 'Focal loss for dense object detection'. In *Proceedings of the IEEE international conference on computer vision*, 2980–88, 2017.

Litjens, Geert, Clara I. Sánchez, Nadya Timofeeva, Meyke Hermsen, Iris Nagtegaal, Iringo Kovacs, Christina Hulsbergen-Van De Kaa, Peter Bult, Bram Van Ginneken, and Jeroen Van Der Laak. 'Deep learning as a tool for increased accuracy and efficiency of histopathological diagnosis'. *Scientific reports* 6, no. 1 (2016): 1–11.

Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. 'Ssd: Single shot multibox detector'. In *European conference on computer vision*, 21–37. Springer, 2016.

Liu, Yun, Ming-Ming Cheng, Xiaowei Hu, Kai Wang, and Xiang Bai. 'Richer convolutional features for edge detection'. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3000–3009, 2017.

Loshchilov, Ilya, and Frank Hutter. 'Sgdr: Stochastic gradient descent with warm restarts'. *arXiv preprint arXiv:1608. 03983*, 2016.

Lyu, Qiongshuai, Min Guo, and Zhao Pei. 'DeGAN: Mixed noise removal via generative adversarial networks'. *Applied Soft Computing* 95 (2020): 106478.

Mahmood, Ammar, Mohammed Bennamoun, Senjian An, and Ferdous Sohel. 'Resfeats: Residual network based features for image classification'. In *2017 IEEE international conference on image processing (ICIP)*, 1597–1601. IEEE, 2017.

Mansourifar, Hadi, and Weidong Shi. 'Deep synthetic minority over-sampling technique'. *arXiv preprint arXiv:2003. 09788*, 2020.

Matsuo, Koji, Sanjay Purushotham, Bo Jiang, Rachel S. Mandelbaum, Tsuyoshi Takiuchi, Yan Liu, and Lynda D. Roman. 'Survival outcome prediction in cervical cancer: Cox models vs deep-learning model'. *American journal of obstetrics and gynecology* 220, no. 4 (2019): 381-e1.

Minaam, D. S. Abdul, and Eslam Amer. 'Survey on machine learning techniques: Concepts and algorithms'. *International Journal of Electronics and Information Engineering* 10, no. 1 (2019): 34–44.

Opelt, Andreas, Michael Fussenegger, Axel Pinz, and Peter Auer. 'Weak hypotheses and boosting for generic object detection and recognition'. In *European conference on computer vision*, 71–84. Springer, 2004.

Oskouei, Rozita Jamili, and Bahram Sadeghi Bigham. 'Over-sampling via under-sampling in strongly imbalanced data'. *Int. J. Adv. Intell. Paradigms* 9, no. 1 (2017): 58–66.

Perdana, Anugrah Bintang, and Adhi Prahara. 'Face recognition using light-convolutional neural networks based on modified Vgg16 model'. In *2019 International Conference of Computer Science and Information Technology (ICoSNIKOM)*, 1–4. IEEE, 2019.

Pereira, Lucas, and Nuno Nunes. 'Performance evaluation in non-intrusive load monitoring: Datasets, metrics, and tools—A review'. *Wiley Interdisciplinary Reviews: data mining and knowledge discovery* 8, no. 6 (2018): e1265.

Porwal, Prasanna, Samiksha Pachade, Ravi Kamble, Manesh Kokare, Girish Deshmukh, Vivek Sahasrabuddhe, and Fabrice Meriaudeau. 'Indian diabetic

retinopathy image dataset (IDRiD): a database for diabetic retinopathy screening research'. *Data* 3, no. 3 (2018): 25.

Provost, Foster. 'Machine learning from imbalanced data sets 101'. In *Proceedings of the AAAI'2000 workshop on imbalanced data sets*, 68:1–3. AAAI Press, 2000.

Radford, Alec, Luke Metz, and Soumith Chintala. 'Unsupervised representation learning with deep convolutional generative adversarial networks'. *arXiv preprint arXiv:1511. 06434*, 2015.

Rajaraman, Sivaramakrishnan, Ghada Zamzmi, and Sameer Antani. 'Multi-loss ensemble deep learning for chest X-ray classification'. *arXiv preprint arXiv:2109. 14433*, 2021.

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. 'Faster r-cnn: Towards real-time object detection with region proposal networks'. *Advances in neural information processing systems* 28 (2015).

Saini, Manisha, and Seba Susan. 'Comparison of deep learning, data augmentation and bag-of-visual-words for classification of imbalanced image datasets'. In *International conference on recent trends in image processing and pattern recognition*, 561–71. Springer, 2018.

Saini, Manisha, and Seba Susan. 'Data augmentation of minority class with transfer learning for classification of imbalanced breast cancer dataset using inception-v3'. In *Iberian Conference on Pattern Recognition and Image Analysis*, 409–20. Springer, 2019.

Saini, Manisha, and Seba Susan. 'Deep transfer with minority data augmentation for imbalanced breast cancer dataset'. *Applied Soft Computing vol. 97, 106759, 1568-4946, 2020.*

Saini, Manisha, and Seba Susan. 'Bag-of-Visual-Words codebook generation using deep features for effective classification of imbalanced multi-class image datasets'. *Multimedia Tools and Applications* 80, vol. 14 (2021): 20821–47.

Saini, Manisha, and Seba Susan. 'Cervical Cancer Screening on Multi-class Imbalanced Cervigram Dataset using Transfer Learning'. In *15th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI),* 1-6. IEEE, 2022.

Saini, Manisha, and Seba Susan. 'Diabetic retinopathy screening using deep learning for multi-class imbalanced datasets'. *Computers in Biology and Medicine,* vol. 149, p. 105989, 2022.

Saini, Manisha, and Seba Susan. 'VGGIN-Net: Deep Transfer Network for Imbalanced Breast Cancer Dataset'. *IEEE/ACM Transactions on Computational Biology and Bioinformatics,* vol. 20, no. 01, 752-762, 2023.

Saini, Manisha, and Seba Susan. "Tackling class imbalance in computer vision: a contemporary review." Artificial Intelligence Review, 1-57, 2023.

Sandler, Mark, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 'Mobilenetv2: Inverted residuals and linear bottlenecks'. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4510–20, 2018.

Sculley, David. 'Web-scale k-means clustering'. In *Proceedings of the 19th international conference on World wide web*, 1177–78, 2010.

Sharma, Shallu, and Rajesh Mehra. 'Conventional machine learning and deep learning approach for multi-classification of breast cancer histopathology images—a comparative insight'. *Journal of digital imaging* 33, no. 3 (2020): 632–54.

Sharma, Shallu, and Rajesh Mehra. 'Effect of layer-wise fine-tuning in magnification-dependent classification of breast cancer histopathological image'. *The Visual Computer* 36, no. 9 (2020): 1755–69.

Shorten, Connor, and Taghi M. Khoshgoftaar. 'A survey on image data augmentation for deep learning'. *Journal of big data* 6, no. 1 (2019): 1–48.

Simonyan, Karen, and Andrew Zisserman. 'Very deep convolutional networks for large-scale image recognition'. *arXiv preprint arXiv:1409. 1556*, 2014.

Song, Yang, Hang Chang, Heng Huang, and Weidong Cai. 'Supervised intra-embedding of fisher vectors for histopathology image classification'. In *international conference on medical image computing and computer-assisted intervention*, 99–106. Springer, 2017.

Spanhol, Fabio A., Luiz S. Oliveira, Caroline Petitjean, and Laurent Heutte. 'A dataset for breast cancer histopathological image classification'. *Ieee transactions on biomedical engineering* 63, no. 7 (2015): 1455–62.

Spanhol, Fabio A., Luiz S. Oliveira, Paulo R. Cavalin, Caroline Petitjean, and Laurent Heutte. 'Deep features for breast cancer histopathological image classification'. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 1868–73. IEEE, 2017.

Spanhol, Fabio Alexandre, Luiz S. Oliveira, Caroline Petitjean, and Laurent Heutte. 'Breast cancer histopathological image classification using convolutional neural networks'. In *2016 international joint conference on neural networks (IJCNN)*, 2560–67. IEEE, 2016.

Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 'Dropout: a simple way to prevent neural networks from overfitting'. *The journal of machine learning research* 15, no. 1 (2014): 1929–58.

Suh, Hyun K., Jan Willem Hofstee, Joris IJsselmuiden, and Eldert J. van Henten. 'Sugar beet and volunteer potato classification using Bag-of-Visual-Words model, Scale-

Invariant Feature Transform, or Speeded Up Robust Feature descriptors and crop row information'. *Biosystems engineering* 166 (2018): 210–26.

Susan, Seba, and Amitesh Kumar. 'Hybrid of intelligent minority oversampling and PSO-based intelligent majority undersampling for learning from imbalanced datasets'. In *International Conference on Intelligent Systems Design and Applications*, 760–69. Springer, 2018.

Susan, Seba, and Amitesh Kumar. 'SSOMaj-SMOTE-SSOMin: Three-step intelligent pruning of majority and minority samples for learning from imbalanced datasets'. *Applied Soft Computing* 78 (2019): 141–49.

Syarif, Iwan, Adam Prugel-Bennett, and Gary Wills. 'Unsupervised clustering approach for network anomaly detection'. In *International conference on networked digital technologies*, 135–45. Springer, 2012.

Szegedy, Christian, Sergey Ioffe, Vincent Vanhoucke, and Alexander A. Alemi. 'Inception-v4, inception-resnet and the impact of residual connections on learning'. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 'Rethinking the inception architecture for computer vision'. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818–26, 2016.

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 'Going deeper with convolutions'. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1–9, 2015.

Szeghalmy, Szilvia, and Attila Fazekas. 2023. 'A Comparative Study of the Use of Stratified Cross-Validation and Distribution-Balanced Stratified Cross-Validation in Imbalanced Learning' In Sensors 23, no. 4: 2333, 2023.

Tajbakhsh, Nima, Jae Y. Shin, Suryakanth R. Gurudu, R. Todd Hurst, Christopher B. Kendall, Michael B. Gotway, and Jianming Liang. 'Convolutional neural networks for medical image analysis: Full training or fine tuning?' *IEEE transactions on medical imaging* 35, no. 5 (2016): 1299–1312.

Tan, Chuanqi, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. 'A survey on deep transfer learning'. In *International conference on artificial neural networks*, 270–79. Springer, 2018.

Tan, Mingxing, and Quoc Le. 'Efficientnet: Rethinking model scaling for convolutional neural networks'. In *International conference on machine learning*, 6105–14. PMLR, 2019.

Tan, Mingxing, Ruoming Pang, and Quoc V. Le. 'Efficientdet: Scalable and efficient object detection'. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10781–90, 2020.

Tax, D. M., and R. P. Duin. 'Feature scaling in support vector data descriptions'. *Learning from Imbalanced Datasets*, 2000, 25–30.

Team, The Tensorflow. 'Flowers', 2019. http://download.tensorflow.org/example_images/flower_photos.tgz.

Voulodimos, Athanasios, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. 'Deep learning for computer vision: A brief review'. *Computational intelligence and neuroscience* 2018 (2018).

Wan, Shaohua, Yan Liang, and Yin Zhang. 'Deep convolutional neural networks for diabetic retinopathy detection by image classification'. *Computers & Electrical Engineering* 72 (2018): 274–82.

Wang, Xiao-Dong, Rung-Ching Chen, Fei Yan, Zhi-Qiang Zeng, and Chao-Qun Hong. 'Fast adaptive k-means subspace clustering for high-dimensional data'. *IEEE Access* 7 (2019): 42639–51.

Xie, Juanying, Ran Liu, Joseph Luttrell IV, and Chaoyang Zhang. 'Deep learning based analysis of histopathological images of breast cancer'. *Frontiers in genetics* 10 (2019): 80.

Yang, Hanxuan, Ling Shao, Feng Zheng, Liang Wang, and Zhan Song. 'Recent advances and trends in visual tracking: A review'. *Neurocomputing* 74, no. 18 (2011): 3823–31.

Yasinnik, Bronislav. 'Imbalanced Classification via Explicit Gradient Learning From Augmented Data'. *arXiv preprint arXiv:2202. 10550*, 2022.

Ying, Chris, Sameer Kumar, Dehao Chen, Tao Wang, and Youlong Cheng. 'Image classification at supercomputer scale'. *arXiv preprint arXiv:1811. 06992*, 2018.

Yosinski, Jason, Jeff Clune, Yoshua Bengio, and Hod Lipson. 'How transferable are features in deep neural networks?' *Advances in neural information processing systems* 27 (2014).

Zhang, Xinpeng, Jigang Wu, Zhihao Peng, and Min Meng. 'SODNet: small object detection using deconvolutional neural network'. *IET Image Processing* 14, no. 8 (2020): 1662–69.

Zhao, Hengshuang, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. 'Pyramid scene parsing network'. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2881–90, 2017.

Zhu, Chuang, Fangzhou Song, Ying Wang, Huihui Dong, Yao Guo, and Jun Liu. 'Breast cancer histopathology image classification through assembling multiple compact CNNs'. *BMC medical informatics and decision making* 19, no. 1 (2019): 1–17.

Zhang, X., Karaman, S. and Chang, S.F., 2019, December. 'Detecting and simulating artifacts in gan fake images.' *In Proceedings of the 2019 IEEE international workshop on information forensics and security (WIFS)* pp. 1-6, 2019.

LIST OF PUBLICATIONS

International Journals

i. Saini, Manisha, and Seba Susan. "Deep transfer with minority data augmentation for imbalanced breast cancer dataset." **Applied Soft Computing** 97 (2020): 106759.

ii. Saini, Manisha, and Seba Susan. "Bag-of-Visual-Words codebook generation using deep features for effective classification of imbalanced multi-class image datasets." **Multimedia Tools and Applications** 80, no. 14 (2021): 20821-20847.

iii. Saini, Manisha, and Seba Susan. "VGGIN-Net: Deep Transfer Network for Imbalanced Breast Cancer Dataset," in **IEEE/ACM Transactions on Computational Biology and Bioinformatics** 20, no. 1 (2023): 752-762.

iv. Saini, Manisha, and Seba Susan. "Diabetic retinopathy screening using deep learning for multi-class imbalanced datasets." **Computers in Biology and Medicine** 149 (2022): 105989.

v. Saini, Manisha, and Seba Susan. "Tackling class imbalance in computer vision: a contemporary review." Artificial Intelligence Review (2023): 1-57.

Journals Papers Communicated

vi. Saini, Manisha, and Seba Susan. "A review of Deep learning Solutions for Class Imbalance Problems." (2023).

Papers in International Conference

i. Saini, Manisha, and Seba Susan. "Data augmentation of minority class with transfer learning for classification of imbalanced breast cancer dataset using Inception-V3." In Iberian Conference on Pattern Recognition and Image Analysis, pp. 409-420. Springer, Cham, 2019.

ii. Saini, Manisha, and Seba Susan. "Comparison of deep learning, data augmentation and bag-of-visual-words for classification of imbalanced image datasets." In International conference on recent trends in image processing and pattern recognition, pp. 561-571. Springer, Singapore, 2018.

iii. Saini, Manisha, and Seba Susan. "Cervical Cancer Screening on Multi-class Imbalanced Cervigram Dataset using Transfer Learning." In 2022 15th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), pp. 1-6. IEEE, 2022.

Author's Biography

Manisha Saini
Registration Number: **2K17/PhD/CSE /09**
Department of Computer Science and Engineering
Delhi Technological University, Delhi-110042
Email: manisha.saini44@gmail.com
ResearchGate Profile: https://www.researchgate.net/profile/Manisha-Saini-7
Google Scholar : https://scholar.google.com/citations?user=EqomJrIAAAAJ&hl=en

Ms. Manisha Saini is currently pursuing a Ph.D. in Computer Science and Engineering Department from Delhi Technological University, Delhi, India. Her research interests include Computer Vision, Neural Networks, Machine Learning, and Deep Learning. She has eight years of combined academic and industrial experience. She is currently working as Artificial Intelligence Researcher prior to that she was working as Artificial Intelligence Research Engineer and Senior Computer Vision Engineer for early-stage tech startups. Previously, she had worked as an Assistant Professor in the Department of Computer Science and Engineering at Manav Rachna International Institute of Research and Studies, Faridabad, Haryana, India; after serving as an Assistant Professor at the Department of Computer Science and Engineering, G D Goenka University, Gurgaon, Haryana, India.