Automated Detection of Malignant Lesions in the Ovary Using Deep Learning Models and XAI

by

Md. Hasin Sarwar Ifty
20101017
Nisharga Nirjan
20101020
M.A. Diganta
20101034
Labib Islam
20101039
Reeyad Ahmed Ornate
23141041

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
School of Data and Sciences
Brac University
January 2024

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Md. Hasin Sarwar Ifty
20101017

Nisharga Nirjan
20101020

M.A. Diganta
20101034

Labib Islam
20101039

Reeyad Ahmed Ornate
23141041

Approval

The thesis/project titled "Automated Detection of Malignant Lesions in the Ovary Using Deep Learning Models and XAI" submitted by

1. Md. Hasin Sarwar Ifty(20101017)
2. Nisharga Nirjan(20101020)
3. M.A. Diganta(20101034)
4. Labib Islam(20101039)
5. Reeyad Ahmed Ornate(23141041)

Of Fall, 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 18, 2024.

Examining Committee:

Supervisor:
(Member)

Md. Saiful Islam
Senior Lecturer
Department of Computer Science and Engineering
Brac University

Co-Supervisor:
(Member)

Ms. Anika Tasnim
Lecturer (Contractual)
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

Cancer is a complex and highly invasive disease that forms due to the abnormal growth of cells in any part of the body. A majority of cancers are unraveled and treated by incorporating advanced technology. However, ovarian cancer remains a dilemma as it has inaccurate non-invasive detection and a time consuming and invasive procedure for accurate detection. Medical professionals are constantly acquiring enhanced diagnostic and treatment abilities by implementing deep learning models to analyze medical data for better clinical decision, disease diagnosis and drug discovery. Thus, in this research, several Convolutional Neural Networks such as LeNet-5, ResNet, VGGNet and GoogLeNet/Inception have been utilized to develop a model that accurately detects and identifies ovarian cancer. For effective model training, the dataset OvarianCancer&SubtypesDatasetHistopathology from Mendeley has been used. After selecting a base model, we utilized XAI models such as LIME, Integrated Gradients and SHAP to explain the black box outcome of the selected model. For evaluating the performance of the base model, Accuracy, Precision, Recall, F1-Score and ROC Curve/AUC have been used. From the evaluation, it was seen that the slightly compact InceptionV3 model with ReLu had the overall best result achieving an average score of 94% across the performance metrics in the augmented dataset. Lastly for XAI, the three aforementioned XAI have been used for an overall comparative analysis. It is the aim of this research that the contributions of the study will help in achieving a better detection method for ovarian cancer.

Keywords: Convolutional Neural Network, Ovarian Cancer, Tumor, Deep Learning, XAI

Acknowledgement

Firstly, all praise to the Almighty Creator for whom our thesis have been completed without any major interruption.
Secondly, to our advisor and co-advisor for their kind support and advice in our work. They helped us whenever we needed help.
And finally, to our parents as without their throughout support, it may not have been possible. With their kind support and prayer, we are now on the verge of our graduation.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Cancer refers to a condition where some cells within the body grow uncontrollably, that is, the cells proliferate without any form of instruction from the body [31]. Malignant tumors or neoplasms are oftentimes correlated with cancer. One of the defining features of cancer is its ability to expand outside of its normal boundaries [43]. That is, a cancer affected part can create abnormal cells that can spread to other parts of the body and create cancerous cells there. This process is known as metastasis [43]. The primary cause of death due to cancer is metastasis. According to World Cancer Research Fund International [42] or WCRF International in short, the number of new cancer cases in the year 2020 was 18.1 million globally. A majority of cancer can be detected in early or middle stages and be treated effectively. However, there are cancers that cannot be detected until their advanced stage and thus, makes treatment of said cancers much harder. Ovarian cancer is one such cancer that is detected at its advanced stage only [49]. This cancer refers to abnormal growth of tumors in the ovaries. This makes it most lethal to women as it has no screening tests [51]. Many of the other cancers common among women such as Breast cancer, Cervical cancer can be detected via specialized tests. Mammograms and CBEs (Clinical Breast Exam) are commonly performed to detect Breast Cancer, whereas a Pap test is generally done for Cervical Cancer detection [20]. However, Ovarian Cancer has no proper prognosis method. Moreover, its status as the 7th most common cancer globally for women makes it a dangerous disease for half the population of the world. In recent years, Computer Aided Detection (CAD) for diseases has become prevalent in the medical sector. From running simple blood tests to complex disease detection, machine learning has surely aided medical professionals by providing concise data as well as shortening diagnosis time for a disease or medical condition. Cancer is the most recent field where the application of machine learning has been seen [41]. A majority of cancer has early detection or testing methods with relevant involvement using machine language. However, Ovarian cancer is more headache inducing when compared to the other forms of cancer disease as it has no early method of prognosis. Currently, the detection of Ovarian cancer is done via a transvaginal ultrasound, a pelvic exam as well as CA-125 blood test [50]. However, a definitive result is only found via a lab-run biopsy for ovarian cancer [50]. Hence, researchers worldwide are attempting to find out new detection methods or improve the accuracy of already implemented machine learning models. In fact, in a research done in 2022 [44], it is said that the current testing methods utilizing only transvaginal ultrasound alone or in combination with the serum tumor marker

CA-125 are not entirely accurate in all cases. This is because of the diversification in types of non-threatening and threatening cancerous lesions. Hence, the research agrees [44] that lab-based biopsy has more weight in cancer detection. However, one problem with lab based detection is that the lab analysis done by a professional will vary in outcome as it is based on clinical experience. Thus, introduction of an Artificial Intelligence that can accurately provide the resultant tumor type with minimal false report is an essential advancement. If we can implement an impeccable Artificial Intelligence that can accurately detect and determine cancer lesions, then it will be a blessing as the tedious process of biopsy can be shortened. In our study, we have utilized deep learning models such as LeNet, ResNet, VGG and GoogLeNet/Inception. We have selected an appropriate dataset for our base model testing and aim to utilize XAI to explain the future.

1.1 Problem Statement

According to Ovarian Cancer Research Alliance (OCRA)[39], there were 19,880 newly diagnosed instances and 12,810 fatalities due to ovarian cancer in the year 2022. The American Cancer Society projected that these numbers will increase to 19,710 cases and 13,270 deaths in the year 2023. In fact, they remarked that although ovarian cancer is only 11th most common in the United States of America, it ranks as the 5th leading cause of deaths associated with Cancer in women. Even in the United Kingdoms [15], about 4,100 deaths related to ovarian cancer occur every year. Thus, the lethality of ovarian cancer is undeniable. Even if Ovarian Cancer is detected, it is generally done so in the advanced stages [42][51][20]. Therefore, the treatment cost often reaches 6 digit figures [9]. However, the general detection methods are done manually via recto-vaginal pelvic exam, trans-vaginal sonogram, CA-125 blood test and lab biopsy [50]. The former three provide non-uniform results while the later one is extremely dependent on the experience of the analyst. Hence, an early prognosis or faster detection methods for ovarian cancer would be groundbreaking. In recent times, there have been multiple attempts at early prognosis as well as non-invasive detection methods. By incorporating machine learning models to create an assistant using artificial intelligence that can aid medical professionals in detecting ovarian cancer, we will be able to achieve faster detection time while having a similar, if not better, level of accuracy. Thus, we will be exploring the problem of a compact system of Machine Learning and XAI that can accurately detect and provide an early prognosis into the detection and classification of ovarian cancer.

1.2 Research Objective

Several different sources of information [49][51][20][44][38][22][40][36] acknowledge that ovarian cancer is more deadly when compared to other cancers in women. This is primarily because an early prognosis of ovarian cancer is not available yet. Even for detection methods in the advanced stage, a lab biopsy is the only method for accurate detection of ovarian cancer [51][20]. Hence, we decided to take this field for our research and build an accurate machine learning model that can be of great help for acting as an assistant for the medical professionals to analyze and detect ovarian cancer.

Chapter 2

Literature Review

2.1 Background Study

2.1.1 Convolutional Neural Network (CNN)

A Convolutional Neural Network or CNN is an artificial neural network that is used to analyze grid-like data such as images or time series data. It is widely used in tasks such as image classification, image recognition or segmentation and object detection. CNN can be of One, Two or Three dimensions [19]. The concept of Convolutional Neural Network was introduced by imitating the visual cortex in animals that respond to stimuli via special regions in their visual field [1][2]. The idea behind CNN is to memorize the features from input after applying several kinds of convolution filters or kernels to small regions of the output. These filters perform element-wise multiplications and addition over the entire output, producing feature maps that can accurately capture different aspects of the data. A Convolutional Neural Network consists of three primary layers: The Input Layer, the Output Layer and the Hidden Layer. There are four main components in the Hidden Layer. They are: Convolution Layer, Pooling Layer, Activation Function and the Fully-Connected Layer [37][18]. The Convolution Layer operates by applying a set of filters and convolving them with smaller regions of the input to produce feature maps [37]. It is the first and most important layer of CNN. Contrary to conventional approaches such as SIFT, the features in a Convolutional Neural Network are not predefined [18]. They are, in fact, identified and learnt during the training phase. After each Convolution Layer, comes the Pooling Layer. This layer mainly decreases the input size, thereby deducting the number of parameters or weights within the network. This aids in making the model train faster. There exist two forms of Pooling, namely Max-Pooling and Average-Pooling [37]. The former takes the highest value from a feature map, while the latter calculates the average of all the values in a pooling window. Non-linear activation functions, such as ReLU, are applied element-wise in feature maps. This replaces all negative values received as input with zero [18]. Lastly, the Fully Connected Layer, as its name suggests, connects the neurons or nodes of a layer with all of its previous ones. It is generally used towards the end of a neural network to map the learned features towards the desired output.

2.1.2 Explainable Artificial Intelligence

Models built using generic deep learning techniques will follow the norm of standardized artificial intelligence behavior. That is, a normal AI model will provide us with a black box answer. Often times it comes to light that the output of the AI model is not easily understandable and the process of generating the output is needed to fully understand the corresponding produced output, which is prevented by the black box nature of AI models. Thus, to solve this issue, Explainable Artificial Intelligence or XAI came into being. XAI follows the principles of explainability, transparency, and interpretability [23][17][13]. Thus, it is an extremely useful method of deriving the reasoning behind any result of an AI model. The architecture of XAI can be distinguished in two manners [55]. One is Direct XAI, which is built as a white box structure. That is, the model is designed to be easily interpretable from the start. On the other hand, post-hoc models are not interpretable from the outset. However, they can be explained using external techniques to derive the explanation from the post-hoc model. Generally, the method of derivation is categorized into two types, Global XAI and Local XAI [47]. Global XAI models offer a comprehensive insight into the decision-making process of an AI model [55]. These models generally encapsulate the associations between input features and predictions at a broad and abstract level. On the other hand, Local models deliver precise, instance-level explanations for singular predictions [55]. They evaluate the precise impact of each feature in relation to a specific prediction. Examples of Local XAI include LIME, Anchors and SHAP (local) while examples of global XAI are SHAP(global), PDP and Global Surrogate Models.

2.2 Related Works

A research article by Zhou et al. [44] contained a review of the recent trends in the application of Artificial Intelligence in the field of diagnostic and prognostic prediction of ovarian cancer. The researchers had systematically searched through PubMed and IEEE/IET Electronic Library for studies in between the timeframe of January 2000 to March 2020 that utilizes Artificial Intelligence in Ovarian Cancer. The keywords that they looked for include multiple machine learning and computer aided terminologies combined with ovarian cancer. They ended up with 39 studies that discussed the utilization of Artificial Intelligence in Ovarian Cancer. Of them, 7 studies used radiomics and pathological images, 19 studies utilized high-throughput omics data and 13 studies utilized high-serum markers and clinical data. They provided reasoning behind the larger number of high-throughput omic data that is the research trend on genomics and transcriptomes. They gave sound reasoning and reached the conclusion that the utilization of high-throughput data will increase not only in the field of cancer research but also in other medical sectors.

Another article by Hema et al. [38] presented a novel image classification model for ovarian cancer utilizing FaRe-ConvNN, which is a rapid region-based Convolutional neural network. In this model, the input image was segmented and then pre-processed. Afterwards, they applied FaRe-ConvNN to perform the annotation procedure. The classification is done using a combination of SVC and Gaussian Naive Bayes classifiers after the region based training is completed. For testing

the model, they utilized data from the Cancer Imaging Archive database where the suggested classifier was used on single-cell blood smear samples. The researchers used epithelial cells, germ cells, and stromal cells samples separately, which were readily available in the utilized database. For the results, a confusion matrix was created for both the SVC and Gaussian Naive Bayes. After comparing with existing models, the Gaussian Naive Bayes showed an accuracy score of 97%, with sensitivity and specificity of 97.7% and 98.69% respectively. Based on the results, it can be concluded that the proposed model for ovarian cancer is an important contribution in the medical sector.

In a research article by Wang et al. [22], the researchers developed a deep learning algorithm that can differentiate benign lesions from malignant lesions using magnetic resonance imaging in terms of ovarian cancer. They had tested a total 545 lesions from 451 patients, of which, 379 were benign and 166 were malignant. The model performance was then compared with 4 junior radiologists and 3 senior radiologists on the same test set. The results showed that the model had higher accuracy and specificity against both the juniors (0.87 vs 0.64, 0.92 vs 0.64) and the seniors (0.87 vs 0.74, 0.92 vs 0.70). With the assistance of the model, the juniors showed a huge improvement in their accuracy (0.77 vs 0.64) and specificity (0.81 vs 0.64). In fact, the juniors showed higher specificity (0.81 vs 0.70) but similar accuracy (0.77 vs 0.74) while utilizing the proposed model when compared with the senior radiologists. In conclusion, the researchers said that the utilization of Artificial Intelligence can assist radiologists in assessing the nature of ovarian lesions while also improving their performance.

Another research article by Schwartz et al. [40] proposed an automated framework that detects ovarian cancer from transgenic mice using optical coherence tomography (OCTT) recording. The basis of this proposal is the clear lack of non-invasive and viable source of early ovarian cancer prognosis. Hence, optical coherence tomography or OCT has been used as the sample input. The researchers utilized three neural networks namely, a VGG-supported feed-forward network, a 3D CNN, and a convolutional Long Short-Term Memory (LSTM). Their experiments showed favorable results while LSTM showed the best AUC of 0.81 with a standard deviation of 0.037. The authors of this research acknowledged that despite the absolute potential of this research, the experimental results can be made better by using a much larger dataset. However, they believe that the significance of this research lies in the fact that the usage of OCT can be a viable early prognosis for ovarian cancer.

A research paper by Hsu et al. [36] utilized ten convolutional neural network models that are popular in recent times for the detection and classification of ovarian cancer. To ensure robustness of the model, the researchers used random sampling of the training and validation data multiple times. This also ensured that they have ten readily available test results as the final assessment data. After completing the training, they selected three models with the highest ratio of accuracy to time and utilized them for ensemble learning. Finally, they used the interpretation of the ensemble classifiers as the result and visualized the decision making process using gradient-weighted class activation mapping (Grad-CAM) technology. For the database, they collected data from 587 patients from Taiwan following legal procedures. In their

testing, they selected ResNet-18, ResNet-50, and Xception for ensemble learning. They also used three different types of ensemble learning methods and suggested the one that involves decision-making with multiple models based on their confidence score for clinical application. In their final discussion, they suggested that the confidence threshold be set at 80%-100% for the best possible outcome when using their model.

A research article by Wang et al. [34] developed a predictive model that utilizes deep learning algorithms. This weakly supervised approach predicts the effectiveness of drug-based treatment from histopathology images. One of the defining features of this model is that the inputted histopathological slide images do not contain any form of regional annotations by pathologists. Their reasoning behind the creation of this model is that any form of histopathological analysis is normally dependent on the judgement ability of the examiner as well as the ability to integrate medical data and perceived data. Thus, they believe that this time consuming and difficult task can be made easier with the inclusion of deep learning algorithms. For their model, they utilized tissue sample slides from TSGH bank. After inputting the slides, they generate a tile based pyramid data structure, which generates a cascading network of data. Then, they ran the data through three models. The Cascade DL framework identifies the points of interests in the sample data. Afterwards, the points of interest are separated using the Classifier DL model and finally, the decision model for treatment effectiveness outputs whether the treatment was effective or not. As very little research has been done in this field, they benchmarked their results with two other top papers as the goal values. Finally, they remarked that their developed model was better than the benchmarked models as it had higher statistics in the five-fold verification method.

Another research paper by Ghoniem et al. [28] discussed an evolutionary deep learning model that is hybrid in nature. This model aims to diagnose Ovarian Cancer stages. To do this, it utilizes multi-modal data and combines gene and histopathological image modality. The researchers put up a deep feature extraction network based on the many states and forms of each modality. To analyze gene longitudinal data, it also incorporated a predictive antlion-optimized Long-Short-Memory Model. To process photos of histopathology, another predictive antlion-optimized CNN model is added into the hybrid model. The antlion optimization technique automatically sets the topology of each customized feature network to improve performance. Next, utilizing weighted linear aggregation, the output from the two enhanced networks is combined. Finally, the Ovarian Cancer stage is predicted using the deep fused characteristics. They conducted tests by comparing the model with 9 other evolutionary models that are distinct and utilizes multi-modal data. After using benchmarks for ovarian, breast and lung cancers, they concluded that their proposed model has greater precision and accuracy over the 9 other models.

In another research work by Binas et al. [45], an automated categorization system has been proposed that enables medical professionals to rapidly recognize intratumoral regions with various cellular compositions that are suggestive of tumor heterogeneity. They deemed their approach as novel due to its ability to rapidly construct medical image segmentation, visualize tumor fused in the T2W sequence, pipelines

including data I/O, pre-processing, metrics, a library with a state of the art feature extraction model, and model utilization such as training, tumor area classification, and fully automatic evaluation. They believe that this model will be beneficial as it can not only act as a form of guidance software for biomedical researchers in training their prediction abilities, but also aid medical professionals by creating individual medicine groups for varying risk groups via classification.

Another research paper by Kasture et al. [30] is the first to identify, predict, and categorize ovarian cancer subtypes from histopathological images using VGG16. Initially, they trained the model with 500 images, 100 for each class, and obtained an accuracy of 50%. They then multiplied the dataset of 500 images by doing several types of image augmentations to produce 24742 images. Then, they utilized this augmented image dataset to produce an accuracy of 84.64%. Their core contributions are actually a wonderfully segmented image dataset that accurately classifies the various categories of ovarian cancer. Moreover, they displayed a series of accurate statistics that solidified their contribution of combining the prediction of ovarian cancer & sub-type classification.

Chapter 3

Methodology

3.1 General Outline

In our research, we decided to use multiple Convolutional Neural Network (CNN) models such as LeNet, ResNet, VGGNet and GoogLeNet or Inception. Some of these models were chosen due to their simplicity and others due to their effectiveness. For our database, we will be using a publicly available dataset from Mendeley Data that contains 4 subtypes of Ovarian cancer and also non cancerous histopathological images. Our general workflow diagram of the proposed system is given in Figure-3.1:

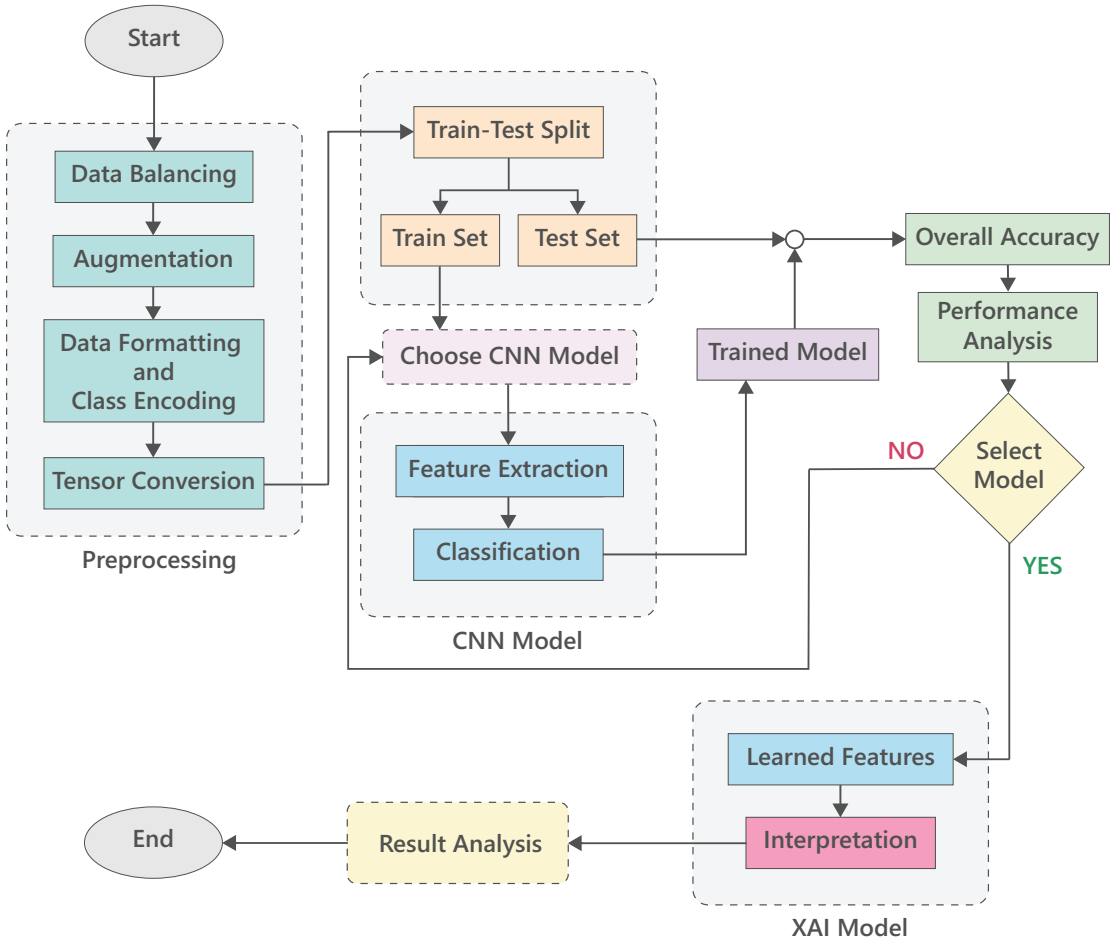


Figure 3.1: Workflow Diagram

3.2 Gathering the Dataset

Before the construction of a model for our work, we need to find an appropriate dataset that can be utilized to bring our future model to its full potential. As such, we selected the dataset "OvarianCancer&SubtypesDatasetHistopathology" from Mendeley[29] to be the basis of our research. We picked this dataset since it not only has mulitiple different types off malignant tumor classes, it also has samples of benign and non-tumor classes as well.

3.3 The Base Models

To select an appropriate model for accurate detection of different tumors from images, we aim to select an appropriate CNN model among the several variations of LeNet, ResNet, VGGNet and Inception. The base models are provided below and the variant models will be explained in the Implementation portion. One thing we have utilized throughout most of our models and variants is the usage of 'Softmax' activation function in the output layer.[54]. The equation for 'Softmax' function is:

$$softmax(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}}$$

Our reasoning for this is that 'Sigmoid' and 'Argmax' were utilized in multiclass classification problem and both of these turned out inappropriate for the classification under most circumstances. In a multiclass classification problem, where the classes are mutually exclusive, the entries of the 'Softmax' output sums up to 1. As such, we opted for 'Softmax' function for the activation function in our output layer.

3.3.1 LeNet

LeNet-5 or LeNet, which is a specific CNN structure, was primarily introduced in 1998 by Yann LeCun, Yoshua Bengio , Leon Bottou, and Patrick Haffner [3]. As a revolutionary architecture of that time, LeNet possesses the basic qualities of a normal CNN such as a convolutional layer, pooling layer as well as a fully connected layer. LeNet-5 employs seven layers in total as seen in Figure-3.2. Although not part of the seven layers, LeNet takes a 32x32 image in the Input Layer [24]. The first convolution layer (C1) employs 6 filters of size 5x5 on the input image and to obtain 6 feature maps of size 28. The size calculation here is:

$$32 - 5\,(Size) + 1\,(Stride) = 28$$

The general model for LeNet is quite simple in comparison to the other models we will be using [24]. There are three convolution layers, each utilizing 5x5 kernels with the filters being 6, 16, 120 for respectively. The output feature map from these convolution layers are 28x28x6, 10x10x16 and 120. After the first two convolution layers, a 2x2 max-pool is performed. After the final convolution layer, the nodes are flattened to ensure an easier time in constructing the output layer. The overall structure is given in Table-3.1. The variable aspects will be explained in the corresponding implementation section.

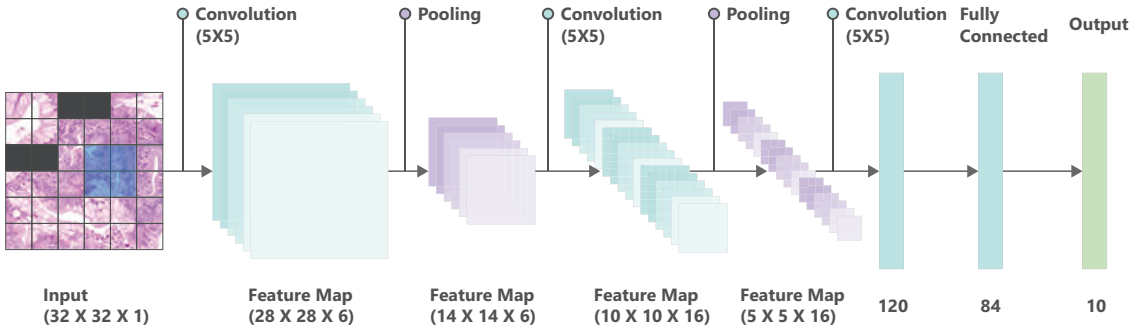


Figure 3.2: LeNet

Layer Name	Kernel Size	Filters	Activation Function	Output Size
Rescale	-	1	-	32x32x3
Convolution	5x5	6	ReLu	28x28x6
MaxPool	2x2	-	-	14x14x6
Convolution	5x5	16	ReLu	10x10x16
MaxPool	2x2	-	-	5x5x6
Convolution	5x5	120	ReLu	1x1x120
Dense	-	84	ReLu	84
Output	-	5	Softmax	5

Table 3.1: LeNet Base Model

3.3.2 ResNet

Residual Neural Network (ResNet) is a deep learning architecture that was initially developed by Kaiming He, Shaoqing Ren, Xiangyu Zhang, and Jian Sun in 2015 [5]. ResNet was the solution to the problem of vanishing gradient and degradation of network performance with increasing depth of neural network. The primary innovation of ResNet is the introduction of residual connections as seen in Figure-3.3, which enables the network to learn residual mapping [52]. This effectively allows the connections to bypass one or more layers and propagate information directly to subsequent layers [52][33]. The building blocks of ResNet are called Residual Blocks. These Residual Blocks are used to perform the connection skipping operations. The output of the Residual Neural Network is determined by the following equation.

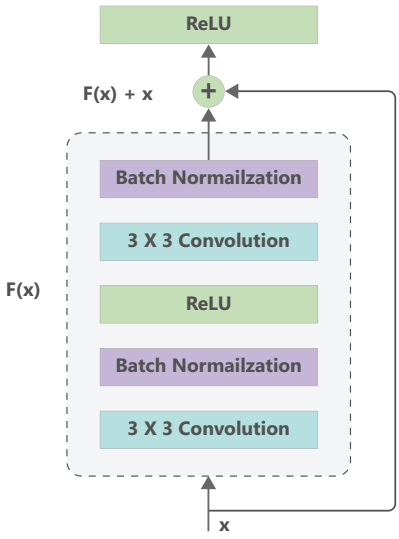


Figure 3.3: ResNet

$$y = F(x) + x$$

The skipping operations are done via two methods of signal propagation, namely, Forward Propagation and Backward Propagation.

$$x_{n+1} = F(x_n) + x_n$$

Applying this recursively, we have for Forward Propagation:

$$x_L = x_l + \sum_{i=l}^{L=1} F(x_l)$$

where, L is index of the last residual block and l is that of any earlier block. This suggests that a signal is passed from a block l to a deeper block L.

For Backward propagation, let us take the derivative of Forward Propagation with respect to xl:

$$\begin{aligned}
\frac{\delta\epsilon}{\delta x_l} &= \frac{\delta\epsilon}{\delta x_L} \times \frac{\delta x_L}{\delta x_l} \\
&= \frac{\delta\epsilon}{\delta x_L}\left(1 + \frac{\delta}{\delta x_l} \times \sum_{i=l}^{L=1} F(x_l)\right) \\
&= \frac{\delta\epsilon}{\delta x_L} + \frac{\delta\epsilon}{\delta x_L} \times \frac{\delta}{\delta x_l} \times \sum_{i=l}^{L=1} F(x_l)
\end{aligned}$$

Here, is the function where degradation has to be minimized. This suggests that a shallow signal $\frac{\delta\epsilon}{\delta x_l}$ has a term $\frac{\delta\epsilon}{\delta x_L}$ always added to it. Hence, the signal $\frac{\delta\epsilon}{\delta x_l}$ never disappears no matter how small the gradient of $F(x_l)$ becomes [7].

The basic building blocks of ResNet across its variations are quite similar [5]. We simply add residual connections every few 'blocks' or combinations of convolutional and maxpool layers. We will be primarily focusing on building ResNet-34 with two variable inputs of 32x32 and 224x224, ResNet-50 and ResNet-101 with 224x224 size image inputs. We optimized theses variants with some hyper-parameters that suits our needs such as utilizing learning rate and node dropouts for optimization control and avoiding overfitting. The various models each require a different ratio of learning rate and dropout rate. We cannot manually test each and every possible outcome as that would take an extremely long time. Thus, we took a randomized approach for every ResNet model. That is, we did the following for each variation of ResNet models: We initially set the range for Learning Rate to be from 0.0001 to 0.1 and that of Dropout Rate from 0.0 to 0.9. We then took random sets of learning rate and dropout rate over 10 iterations and inserted the random hyperparameters in the model and ran over 3 epochs. At last, we selected the best learning rate and dropout rate based on the best testing accuracy. Now, the only variable thing in between the different ResNet models is the input image size and the number of convolution layers. The specific details are explained in the implementation portion.

3.3.3 VGGNet

VGGNet is a deep convolutional neural network introduced by Karen Simonyan and Andrew Zisserman from the University of Oxford in 2014 [6]. The reason it is called a deep CNN is because it has multiple layers with VGG-19 consisting of 19 convolutional layers and VGG-16 having 16 convolutional layers as seen in Figure-3.4. The difference between the several variants is in the number of layers mostly. VGGNet is mostly used for object recognition and large scale image recognition. It does so by classifying images in predefined categories with high accuracy. As it is a very effective learning model, VGGNet is oftentimes used as a pre-trained model [37].

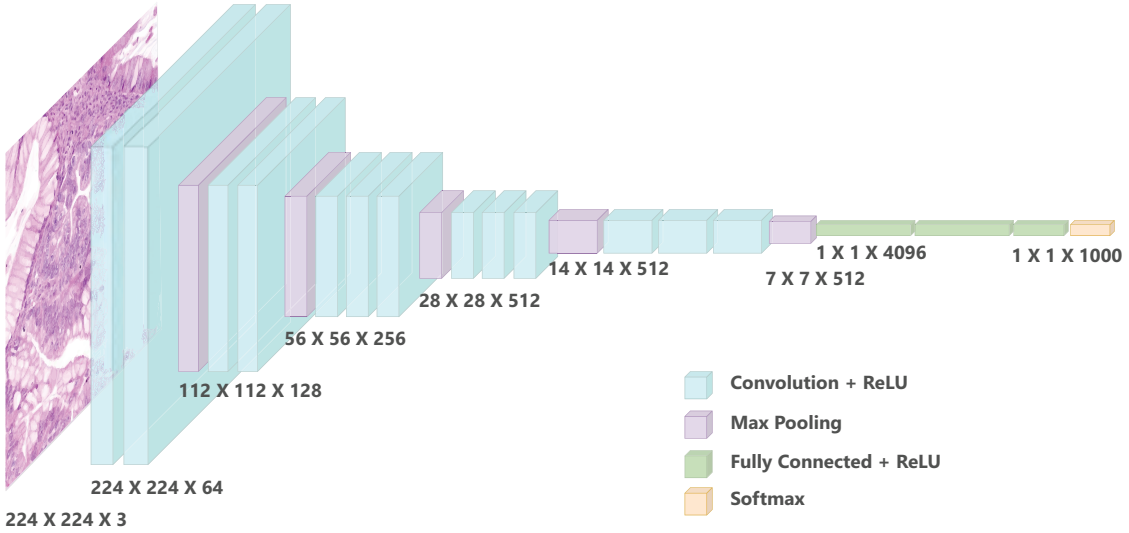


Figure 3.4: VGGNet

All variants of VGGNet work in similar manners. For the Input, VGGNet takes an input of 224x224 size image. In the convolution layer, VGG utilizes minimal receptive fields i.e. 3x3 kernels or convolution filters that can still capture directional features [7]. Furthermore, 1x1 convolution filters are utilized for linear transformation of the input. Like AlexNet, VGGNet also utilizes ReLU to have positive output for positive inputs and zero for non-positive inputs. In fact, LRN is not used by VGG due to the increase in memory consumption. After each convolution layer, a max-pooling is done to downsample the size of feature maps while retaining the core features. After the final convolution and pooling layer, comes the fully connected layer. The predictions are done via the interconnection created by the fully connected layer. The last fully connected layer is linked to the output layer, with neurons or nodes corresponding to the target classes. This layer utilizes softmax activation function [26][25][14].

We will be testing both VGG16 and VGG19 for the base model requirement. The variations in VGGNet will not matter much as we will be utilizing transfer learning. Compared to the other models, the heavy combination of consecutive convolution layers will result in extremely long training time that is also intensive in the aspect of resource usage [53]. Thus, we will opt to use transfer learning in this case and use a pretrained base model with only the fully connected layer customized to our

needs as seen in Figure-3.5.

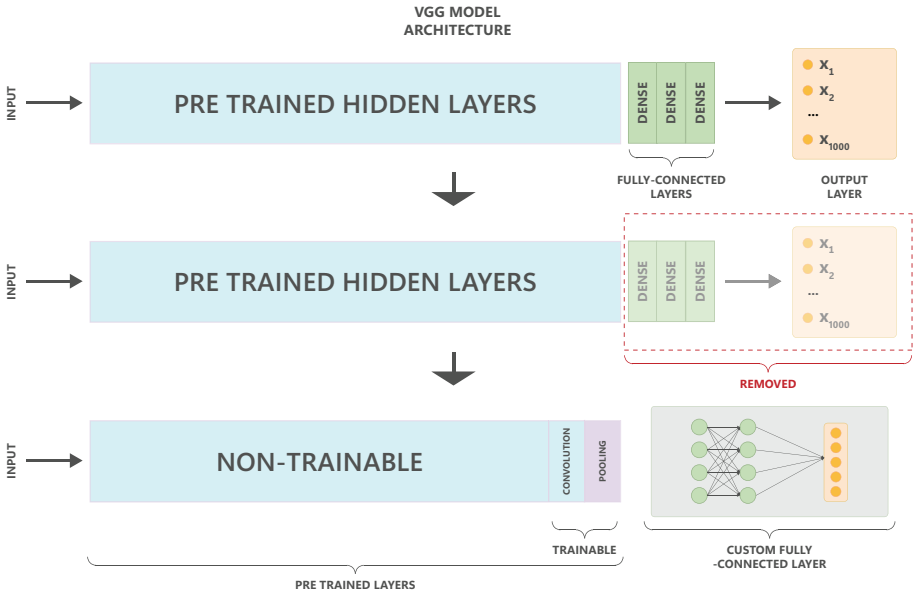


Figure 3.5: VGG Transfer learning process

3.3.4 GoogLeNet/Inception

GoogLeNet, previously known as InceptionNet, is a deep convolutional neural network architecture that was developed by researchers at Google, specifically Christian Szegedy et al., in 2014 [4]. GoogLeNet has a total of 22 parameterized layers and 27 in total if including the non-parameterized layers such as the Max-Pooling layer [46][35]. GoogLeNet has the ability to learn complex features and patterns due to its high number of layers. It has high computational efficiency due to the use of multiple parallel convolutional operations of differing kernels as seen in Figure-3.6. Moreover, it can absorb auxiliary classifiers at intermediate layers, aiding in dealing with vanishing gradient problems while training the model.

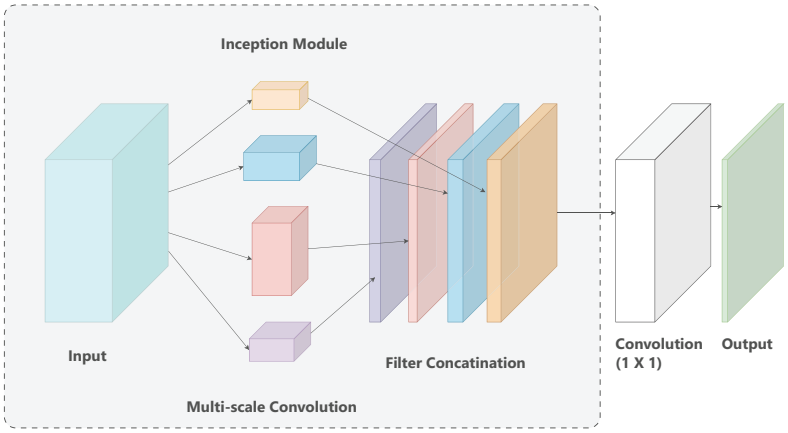


Figure 3.6: GoogLeNet

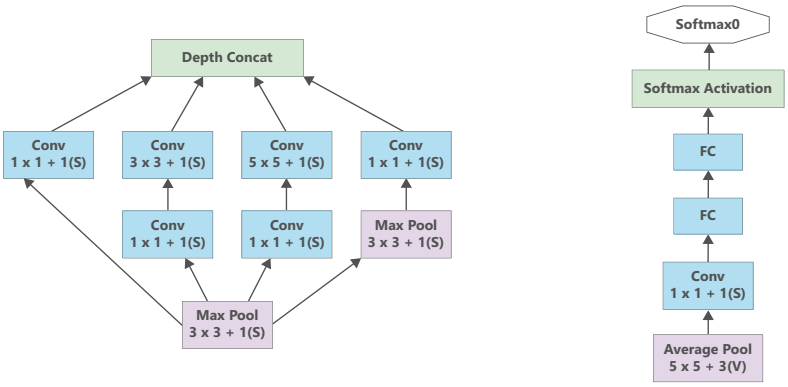


Figure 3.7: Inception Module (left) & Auxiliary Classifier (Right)

GoogLeNet introduced Inception modules. These modules have kernels of sizes 1x1, 3x3 and 5x5 as seen in Figure-3.7. All the outputs from the kernels are stacked towards the end of the inception model. The larger kernels will cover greater area while the smaller ones will cover the smaller but finer details in an image [35]. A total of 9 inception modules are used in GoogLeNet.

The Auxiliary Classifier is something that is added twice in the middle of GoogLeNet. In each auxiliary classifier of the network, there is a 5x5 average pooling layer followed by a 1x1 convolutional layer. Lastly, 2 fully connected layers with 1024 neurons and a softmax output layer with 1000 neurons is present.

For GoogLeNet or Inception, we will be looking into Inception V1 and Inception V3 as these two versions are readily accessible. Like most of our other approaches, this model will be built from scratch. The core mechanism of Inception is the usage of Inception modules. These modules utilize a series of 1x1, 2x2, 3x3, 5x5 convolution and maxpool layers to create a branching method such that the larger kernels cover the major details and the smaller ones will cover the smaller details [32]. In both Inception V1 and V3, we will be using only two inception modules as that will be more than enough to tackle our chosen dataset. If we were to work with larger, complex datasets then we can opt to add in more inception modules according to our needs.

3.4 Selecting the base AI model

To choose the better AI model, we need to check and analyze the overall generated outputs from the models. To facilitate an easier approach to this, we will check a few parameters such as [21][16][16]:

- Accuracy: Accuracy is utilized to determine the ratio of correct predictions versus the total number of items in a dataset.

$$Accuracy = \frac{TruePositives + TrueNegatives}{AllResults}$$

- Precision: Precision will inform us the number of true positives i.e. correct positive predictions made.It can be summarized as the ratio of true positives to sum off all positive outcomes.

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

- Recall: Recall, also known as Sensitivity, is a ratio of the correct true predictions made over all the true predictions.

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives}$$

- F1-Score: The Harmonic mean of Precision and Recall. A higher F1-Score is dependent on both the Precision and Recall Score. As such, a higher F1-Score will mean a better result.

$$F1Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

- ROC Curve: The Receiver Operating Characteristic (ROC) curve shows the performance of all a model for all the designated classes. It utilizes True Positive Rate and False Positive Rate to build a curve that can evaluate the model easily.

- AUC: Area Under the Curve or AUC is a method to easily find out the two dimensional area under a ROC curve using a generalized classification. We will be performing OvR (One Versus Rest) for each classifier to better understand the generic score and figure out if we need to further improve our dataset or model in any way.

3.5 The XAI Models

In contrast to the CNN models, the measure of performance for XAI requires a variety of on site information such as human-AI trust, measure of practicality, clarity etc. Unfortunately, this information can only be authenticated via expert opinion of lab analysts or medical professionals and thus, we are unable to properly select an XAI model. However, we will be preparing 2 local post-hoc XAI and 1 local variation of a global post-hoc XAI for an overview of our model.

3.5.1 LIME (Local)

Local Interpretable Model Agnostic Explanations or simply LIME was featured first on a paper by Marco Ribeiro et al [8] in February of 2016. As the full name shows, the model is a local model that explains only a short form of a black box or interprets only desired inputs by the user. The model agnostic portion of the name comes from LIME's ability to be used in most machine learning models.

For images, LIME utilizes the lime_image module for classification problem [27]. We will be utilizing the LimeImageExplainer() class, which will generate the coefficients referring to the features of an instance contributions, both positive and negative, to the prediction. Initially, an explainer object needs to be created from LimeImageExplainer(), then we will generate an explanation for using the explain_instance method, whose parameters are.

- image: The input image
- segments: The predictions of the sample image and the perturbed images. Defined with the same model as the black-box model.
- top_labels: Total number of unique classes.
- hide_color: Whether the explanation will have the color same as the original image(0) or just grayscale(1). Set to 0.
- num_samples: Number of perturbed samples(random default transformations) to be generated for the sample image. Set to 1000.

Next, the perturbed models are trained into a linear model via linear regression when utilizing explain_instance. while training, weights are assigned to the perturbed samples too. After that, coefficients are calculated weights: the more the perturbed sample is closer to the original sample, the higher the weight. When the code execution is done, the coefficients are stored in the explanation which refers to the image's every feature's contribution to the prediction. After this, we derive the image with feature contribution as well as the mask from get_image_and_mask.

- label: The input label to explain.
- positive_only: To select only the superpixels that contribute positively towards the prediction. Set to True.

- num_features: Since the image has every feature's contribution, this parameter selects the number of features with the highest contribution. Default value = 10. So, only 10 features will be shown in mask/area of interest.
- hide_rest: Hides non-explanation parts of the image. Set to True.

Next, we will use mark_boundaries from SK image segmentation for displaying the LIME explanation for enhanced visualization with boundaries being highlighted for the area of interest.

3.5.2 Integrated Gradients (Local)

Integrated Gradients was introduced by Mukund Sundarajan in March of 2017 [11]. Integrated Gradients is another Local XAI that we will be testing. Unlike LIME, Integrated Gradients is starting to become popular due to its ease of implementations as well as computational efficiency. Integrated Gradients creates a baseline image that utilizes the unique Attribution Mask to provide insights into which parts of the image are influential in the prediction. The main idea behind Integrated Gradients is to compute the integral of the gradients of the model's output with respect to the input image along a straight path from a baseline to the actual input image. This integral represents the accumulated effect of each pixel's contribution along the path [48].

We will be utilizing TensorFlow's GradientTape for automatic differentiation followed by calculating the integrated gradients by interpolating between a baseline and input image. Afterwards, numerical integration is done to approximate the integral of the gradients. Adjustable parameters in this include the interpolation steps and the batch size. We will be visualizing Integrated Gradients through three subplots. They are:

- Original Image.
- Attribution Mask (absolute sum of integrated gradients).
- Overlay of Attribution Mask on the Original Image.

Overlay plot combines the original image with the attribution mask for better interpretation. We will be utilizing the 'viridis' color map to represent the magnitude of the Integrated Gradients. These subplots will provide us with insights into pixel contributions for the target class prediction.

3.5.3 SHAP (Local)

SHAP (SHapley Additive exPlanations) is a versatile and theoretically grounded framework for explaining the output of machine learning models that was introduced by Scott Lundberg in 2017 [10]. It is based on the cooperative game theory and assigns a value to each feature, indicating its contribution to the model's prediction. For images, SHAP values can be used to explain the prediction of a model by

attributing contributions to individual pixels or groups of pixels. One of the advantages of SHAP is that it satisfies several desirable properties from cooperative game theory, such as efficiency, symmetry, and linearity. This makes it a theoretically sound approach for attributing contributions to individual features in a cooperative manner. As our work is a multi-class classification issue at its core, we will not be generalizing the features in a singular format. Instead, we will highlight an input image's features that will be of positive and negative weights with respect to all of the unique classes. This way, we will be able to identify both the positive and negative correlation of features and unique classes.

To create our explainer, we will use DeepExplainer from the SHAP package. DeepExplainer is an improved version of the DeepLIFT algorithm that functions similarly to Kernel SHAP[12]. DeepExplainer is used to estimate the conditional expectations of SHAP values using a set of background samples. The explainer estimates approximate SHAP values by integrating across numerous background samples so that they amount to the difference between the predicted model output on the passed background samples and the present model output.

For plotting the result, we will be utilizing image_plot to visualize the generated SHAP values. The plot displays five sets of SHAP values corresponding to each class for each input image, offering a comprehensive understanding of the model's decision-making process. The visual plot will show the generated output and feature correlation with the unique classes. The plot will be done in a manner such that the first column of images will refer to the classes Clear Cell, Endometri, Mucinous, Non Cancerous and Serous respectively. Additionally, rest of the columns will refer to the SHAP interpretation of each image across the unique classes. Here, the red pixels will indicate positive features that actively contribute to the likelihood of an image native to a specific class, whereas the blue pixels will signify negative features that weaken its probability for the target class.

Chapter 4

Implementation

4.1 Platform and Language

We opted to use Python as it is an easier approach to machine learning in general. For our platform, we decided to use Google Colaboratory as it is very easily accessible from anywhere and also has relatively decent capabilities for research purposes.

4.2 Data Preprocessing

We uploaded the dataset to github to facilitate an easier approach to data preprocessing and for parallel research across various potential model combinations. To determine the features of the dataset, we ran a few tests and determined that the dataset we selected was balanced in nature as in the dataset, there were 99-100 images per class to a total of 498 images in 5 classes. The data balancing is seen in Figure-4.1 for pre-augmented images.

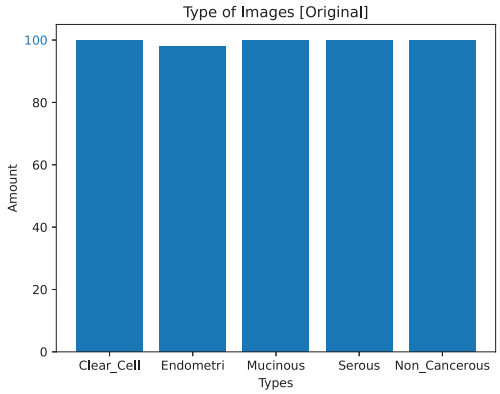


Figure 4.1: Data balancing bar chart featuring pre-augmented images

For pre-processing, the only significant change that we brought was image augmentation. We applied composite augmentation on the images using several types of transformations. These include: image rotation by up to 180 degrees, complete horizontal and/or vertical flipping, changes in brightness, contrast, saturation and hue to get 4 augmented images from each of the original images. To complete image augmentation, we used the Albumentations library and utilized the modules such

as Compose(), Rotate(), Oneof(), HorizontalFlip(), VerticalFlip() and ColorJitter(). The main reason for using the Albumentations library is because it has a variable probability in its transformations, indicating a higher level of randomness when augmenting any form of images. Furthermore, we ensured that the images follow the JPG image file format and that the color encoding of the images follow RGB. While performing the augmentation, we added the augmented and original images to a new sub-directory under our augmented dataset dictionary. Ultimately, our augmented dataset contained 5 subclasses with 2490 images in total which satisfied our requirements. The post-augmented data balance is seen in Figure-4.2.

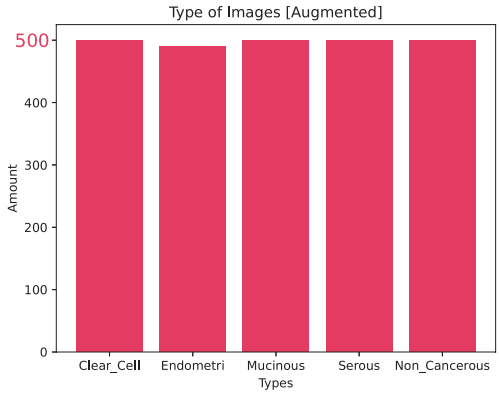


Figure 4.2: Data balancing bar chart featuring augmented images

4.3 Tensor Conversion

Next, we converted the augmented images to tensor data using the image_dataset _from_directory() method of the TensorFlow library. As we are performing a supervised learning approach for our model where the dataset directory we are passing in the method has subdirectories referring to the class names, we set the parameter "labels" to be "inferred". Furthermore, we decided that one-hot encoding will become complicated if we decide to introduce more future subclasses. Hence, our "label_mode" parameter was set to "int". Table-4.1 explains the label and integer correlation.

Sub-Class	Output Label
Clear Cell	0
Endometri	1
Mucinous	2
Non Cancerous	3
Serous	4

Table 4.1: Output Classification

The color mode and batch size of the tensor data are set to RGB and 32 respectively. Our image size is variable in our initial testing, switching in between 32x32 and 224x224 per model requirements. Also, we initially split the dataset into a 80-20 ratio for training and testing. The 80-20 split was done randomly thanks to "seed", "subset" and "shuffle" parameters. As a result, our tensor training dataset includes 1992 images while the tensor testing dataset contains 498 images.

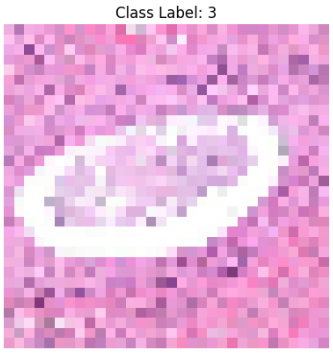


Figure 4.3: Data Sample (32x32 image)

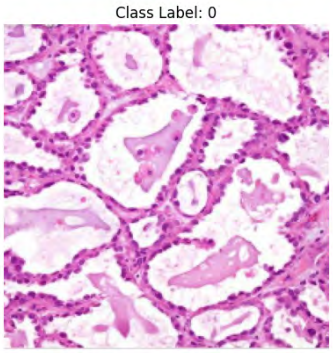


Figure 4.4: Data Sample (224x224 image)

After completing the tensor conversion, we decided to normalize the image dataset for a much smoother running experience with the various Convolutional Neural Network Models. Before doing that, the image portion of the tensor dataset has been converted from uint8 to float32 format using the convert_image_dtype() method of the tensorflow library. This is done so that the resultant scaling can be done much more easily. Only after doing so, we normalized the RGB values from a 0-255 range to a 0-1 range. We had tested this using some basic CNN structure and found that the later range provides a smoother convolution setup for the model. Lastly, we split the datasets, both training and testing into X and Y representing inputs and outputs. As our tensor conversion was done in batches, we used the concat() from tensorflow library method to create an input feature list and output label list for both training and testing datasets. Sample image data with data is shown in Figure-4.3 and Figure-4.4.

4.4 Preliminary model building

In this section, we will explain what significant changes we brought to the base model and highlight a few things we found significant during the model training.

4.4.1 LeNet

For LeNet, we tested 8 variations and concluded that the following 3 were the better variants in terms of overall performance. All of the following models are evaluated over 100 epochs.

1. LeNet-A: The first LeNet is actually just the base model with a customized learning rate. Here, the learning rate was set to 0.001. From the below graph in Figure-4.5, we can see that our training and testing results are far from what we desire which may occur due to overfitting:

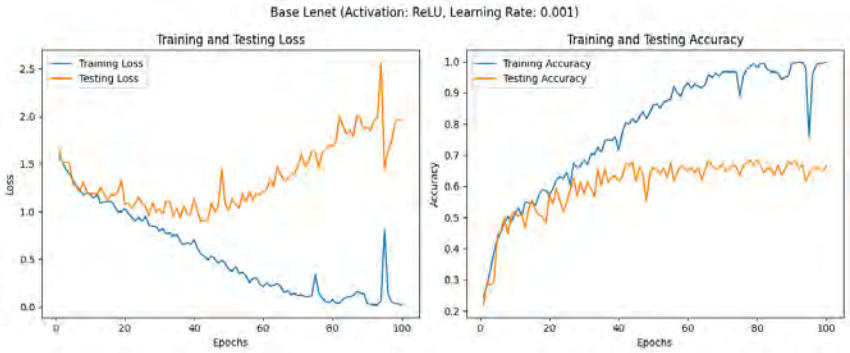


Figure 4.5: LeNet-A Training Vs Testing Loss and Accuracy

2. LeNet-B: This variant of LeNet takes the LeNet-A variant and adds in a dropout function to combat overfitting. From the below graph in Figure-4.6, we can see that our training and testing results are still extremely far apart and thus is not ideal nature:

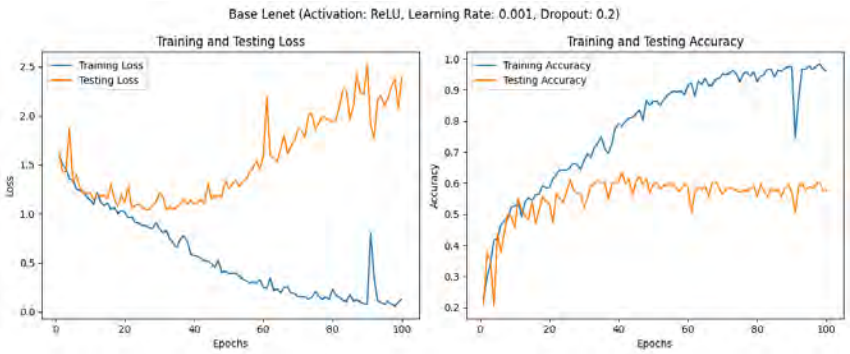


Figure 4.6: LeNet-B Training Vs Testing Loss and Accuracy

3. LeNet-C: This variant of LeNet takes the LeNet-B variant and introduces step decay. With Step-decay, we c. From the below graph in Figure-4.7, we can see that our training and testing results are finally similar to each other. However, our training/Testing accuracy rarely crossed the 55% mark:

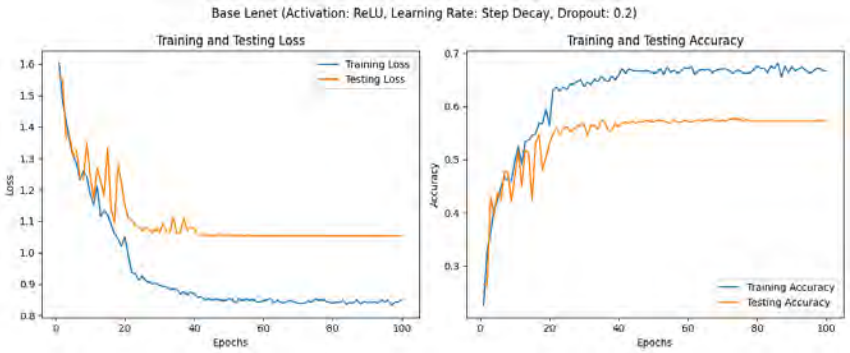


Figure 4.7: LeNet-C Training Vs Testing Loss and Accuracy

4.4.2 ResNet

As said in the methodology portion, the only major changes among the variants will be: Input Size, Number of Convolution Layer, learning rate and dropout. We are only changing these as any other significant changes are either revealing a significant drop in efficiency or is taking too much resource to compile. Our core variants are ResNet34 with 32x32x3 input, ResNet34 with 224x224x3 input, ResNet50 with 224x224x3 input and ResNet101 with 224x224x3 input. Our learning rate and dropout rate for all 4 of the variants are given in Figure-4.8.

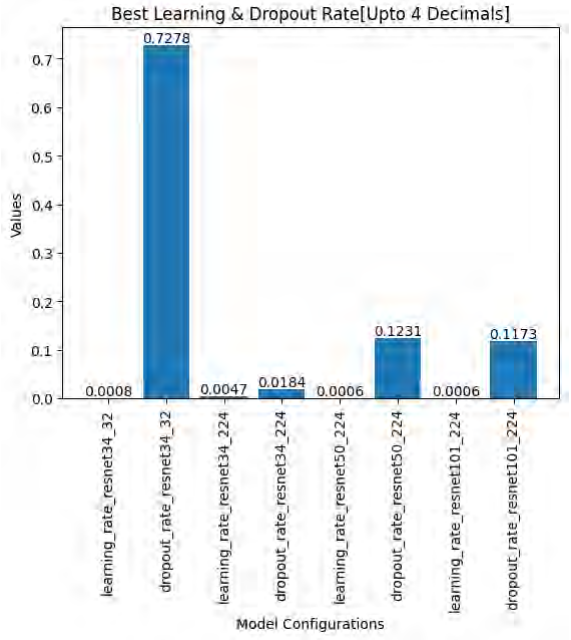


Figure 4.8: ResNet Best Learning Rate and Dropout rate tested over 30 iterations each

1. ResNet34_32: Training & Testing Accuracy/Loss Curve is given below in Figure-4.9:

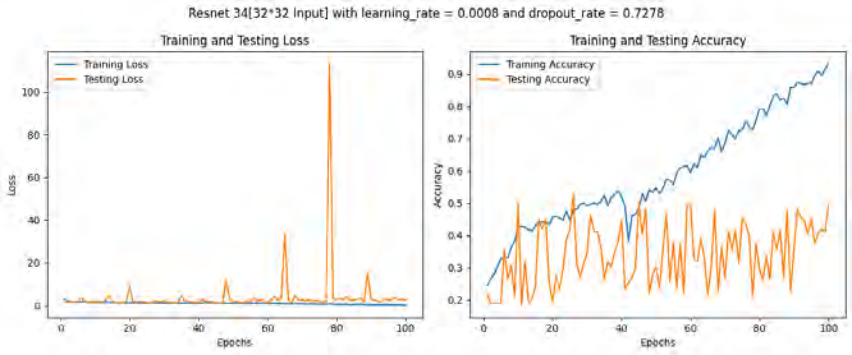


Figure 4.9: ResNet34_32 Training Vs Testing Loss and Accuracy

2. ResNet34_224: Training & Testing Accuracy/Loss Curve is given below in Figure-4.10:

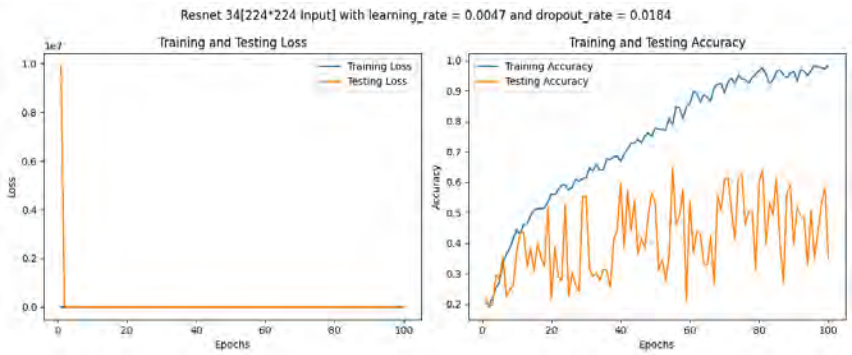


Figure 4.10: ResNet34_224 Training Vs Testing Loss and Accuracy

3. ResNet50: Training & Testing Accuracy/Loss Curve is given below in Figure-4.11:

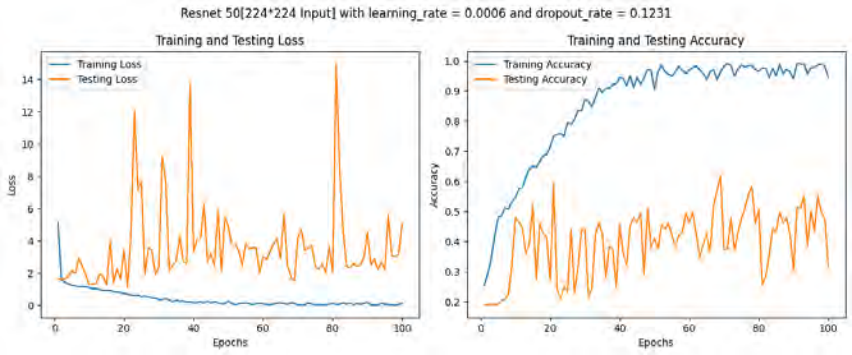


Figure 4.11: ResNet50 Training Vs Testing Loss and Accuracy

4. ResNet101: Training & Testing Accuracy/Loss Curve is given below in Figure-4.12:

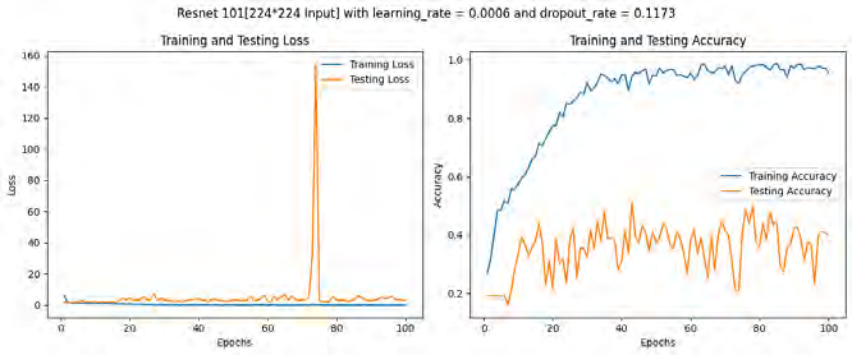


Figure 4.12: ResNet101 Training Vs Testing Loss and Accuracy

4.4.3 VGGNet

As mentioned before, we are utilizing transfer learning for VGG16 and VGG19 with custom top layers to avoid the extensive computational usage that it has. We do so by utilizing the pre-trained convolution layers and only customizing the fully connected layer. For VGG16, we utilize the 'vgg16' from applications module under Keras, TensorFlow. For VGG19, we utilize the 'vgg19' from applications module under Keras, TensorFlow. Our configuration for the Fully Connected Layer across the variants are:

1. VGG16-A: Here, we introduce a two dimensional global average pooling at the start of the fully connected layer. Next, we add three consecutive dense layers with ReLu activation functions with 1024, 1024 and 512 nodes respectively. Our output layer consists of 5 nodes and utilizes the softmax function.

2. VGG16-B: The variant B is similar to variant A but it uses tanh activation function in the dense layers instead.

3. VGG16-C: The variant C is similar to variant A but it adds learning rate of 0.03% and dropout rate of 20%.

4. VGG19: Like the VGG16-A base model, we introduce a two dimensional global average pooling at the start of the fully connected layer. Next, we add three consecutive dense layers with ReLu activation functions with 1024, 1024 and 512 nodes respectively. Our output layer consists of 5 nodes and utilizes the softmax function.

4.4.4 GoogLeNet/Inception

Our iteration of each Inception model was different compared to the generic Inception V1 and Inception V3. Each variants are:

1. InceptionV1-A: We utilized a 2x2 average pooling before the final convolution and layer flattening. We also removed the auxiliary classifier. Furthermore, a generic single output layer is utilized instead of three. Lastly, the activation functions used were purely ReLu excluding the output layer. The Training vs Testing Graph is given below in Figure-4.13:

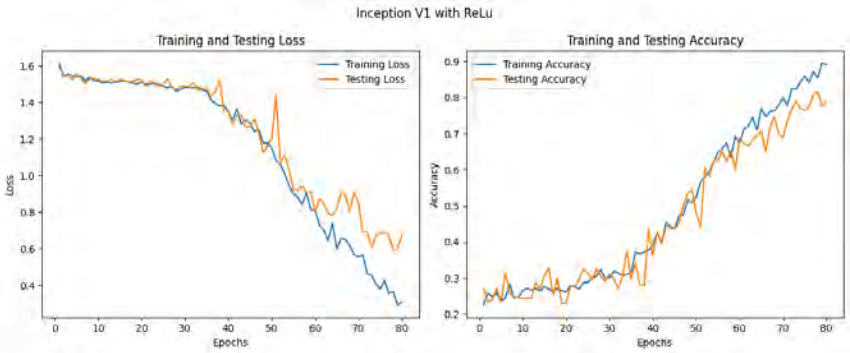


Figure 4.13: InceptionV1-A Training Vs Testing Loss and Accuracy

2. InceptionV1-B: InceptionV1-A's base is used here as well. The only difference here is that we utilized tanh activation function instead of ReLu. The Training vs Testing Graph is given below in Figure-4.14:

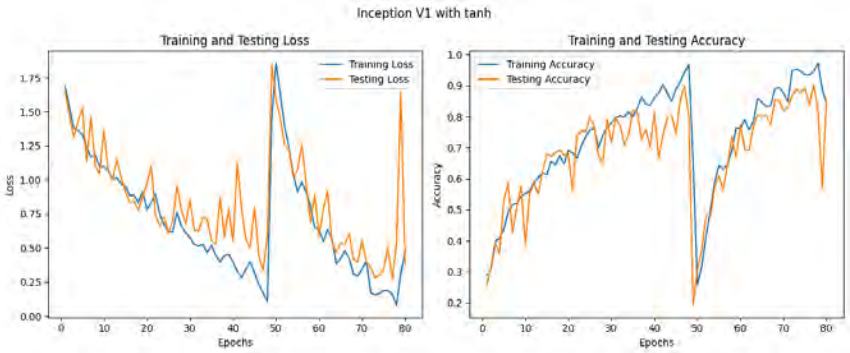


Figure 4.14: InceptionV1-B Training Vs Testing Loss and Accuracy

3. InceptionV3-A: We introduced batch normalization to the InceptionV1-A model. Here, 3 convolution layers have been utilized where the 1st layer is 3X3 instead of 7X7. Additionally, the filters of the inception modules have been modified where the 1st inception module has filters: 64, 128, 128, 32, 32, 32 and the 2nd inception module has filters: 128, 192, 96, 64, 64, 64. The Training vs Testing Graph is given below in Figure-4.15:

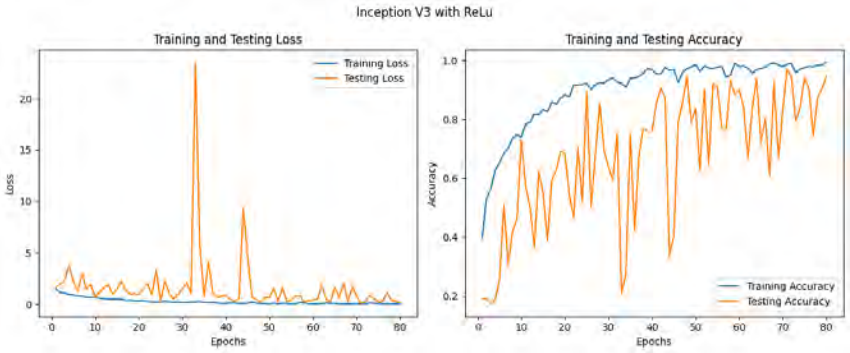


Figure 4.15: InceptionV3-A Training Vs Testing Loss and Accuracy

4. InceptionV3-B: InceptionV3-A's base is used here as well. The only difference here is that we utilized tanh activation function instead of ReLu. The Training vs Testing Graph is given below in Figure-4.16:

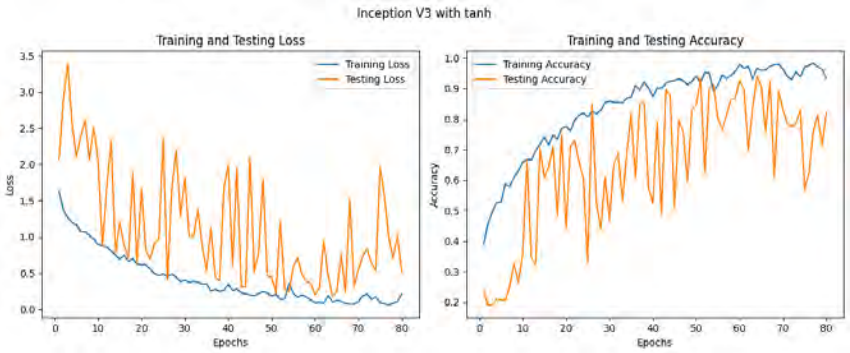


Figure 4.16: InceptionV3-B Training Vs Testing Loss and Accuracy

4.5 Implementation of XAI

Unlike the base models, we have not made any significant changes to the XAI models other than the visual representations. Most of the implementations or procedure of the XAI models that we wanted to use have already been explained in Section 3.5. One additional thing that we did for all three of the models is that we deliberately tested the same one image from each class to compare the feature correlation among the various XAI models.

Chapter 5

Result and Analysis

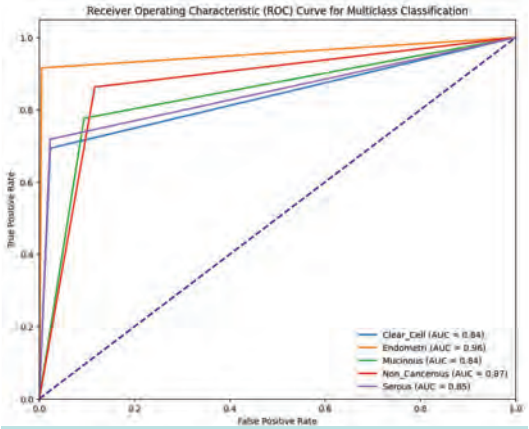
5.1 Result

5.1.1 Base Model

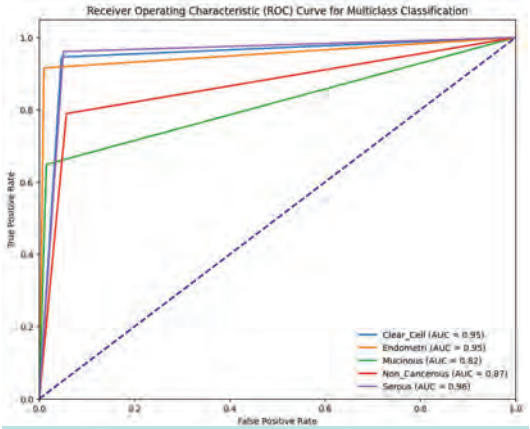
We tested several variants of the core 4 models on our selected dataset and believe that 15 models among them are worth mentioning. The Accuracy, Precision, Recall, F1-Score, ROC Curve and AUC of said models are given in Table-5.1. The ROC curves of the models are shown in images 5.1, 5.2 and 5.3.

Model Name	Accuracy	Precision	Recall	F1-Score
LeNet-A	61.85%	62.20%	61.85%	61.96%
LeNet-B	55.02%	54.51%	55.02%	53.94%
LeNet-C	53.21%	55.28%	53.21%	49.53%
ResNet34_32	43.78%	36.67%	43.78%	38.30%
ResNet34_224	57.03%	59.39%	57.03%	57.70%
ResNet50	34.14%	47.75%	34.14%	33.47%
ResNet101	43.17%	47.17%	43.17%	40.64%
VGG16-A	96.99%	96.98%	96.99%	96.97%
VGG16-B	96.18%	96.27%	96.18%	96.20%
VGG16-C	96.18%	96.32%	96.18%	96.18%
VGG19	97.19%	97.31%	97.19%	97.20%
InceptionV1-A	78.92%	81.58%	78.92%	79.33%
InceptionV1-B	85.74%	86.26%	85.74%	85.42%
InceptionV3-A	94.58%	94.75%	94.58%	94.62%
InceptionV3-B	82.13%	85.11%	82.13%	82.70%

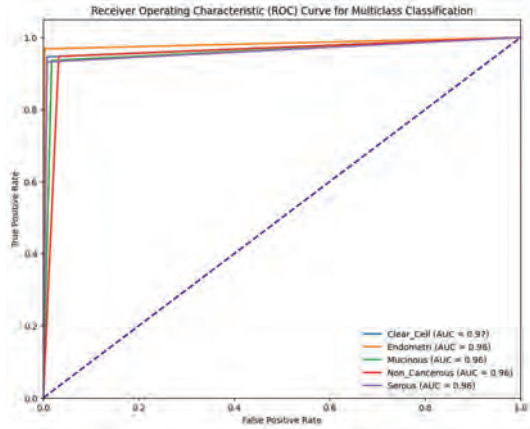
Table 5.1: Overall result



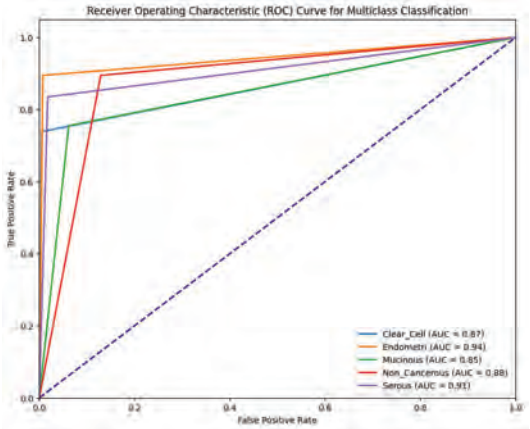
ROC curve: InceptionV1-A



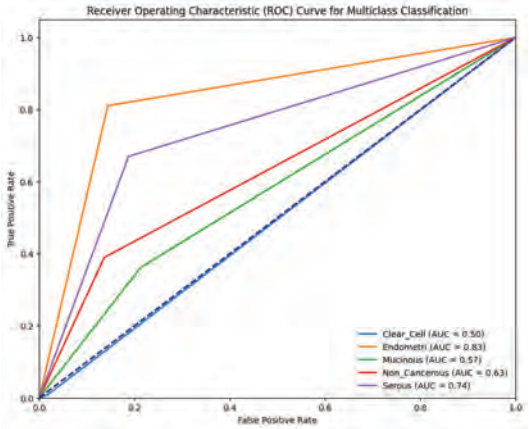
ROC curve: InceptionV1-B



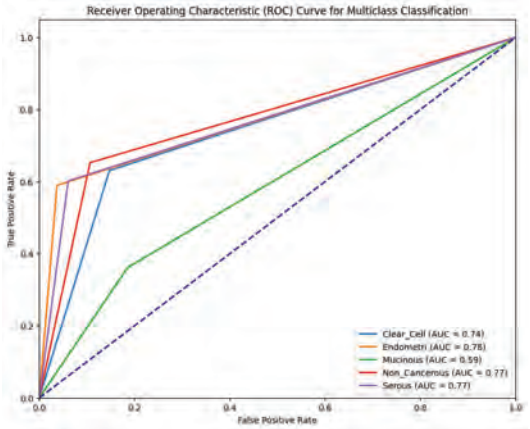
ROC curve: InceptionV3-A



ROC curve: InceptionV3-B

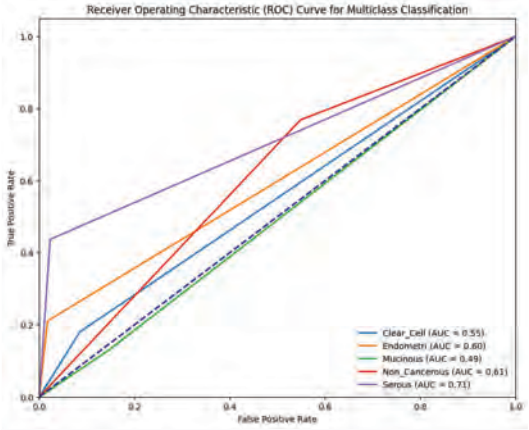


ROC curve: ResNet-34 (32x32)

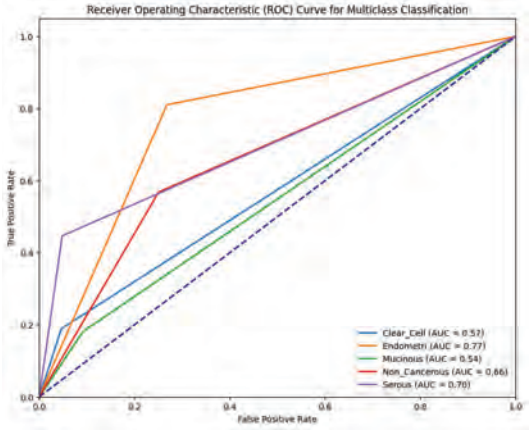


ROC curve: ResNet-34 (224x224)

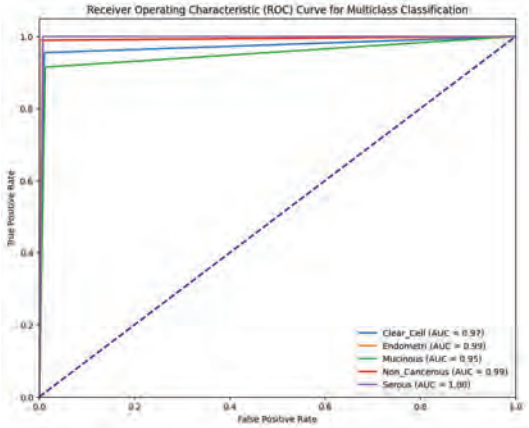
Figure 5.1: ROC Curve (Inception & ResNet)



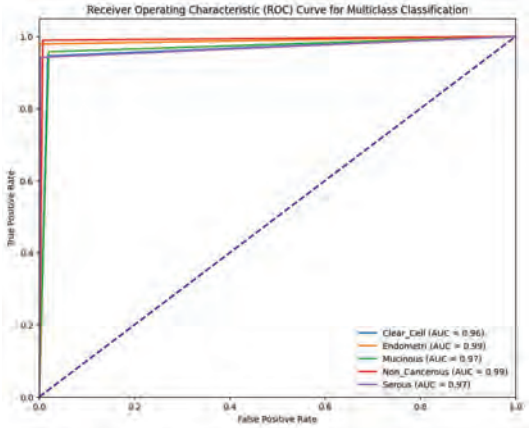
ROC curve: ResNet-50



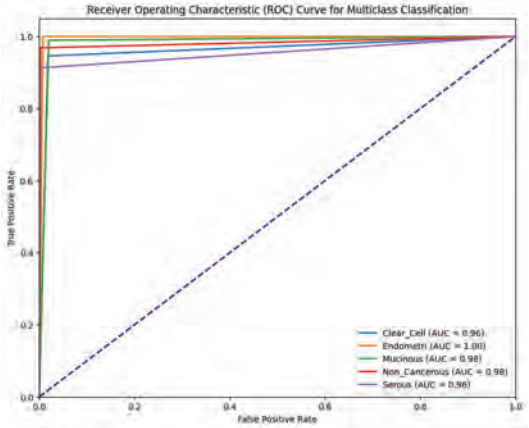
ROC curve: ResNet-101



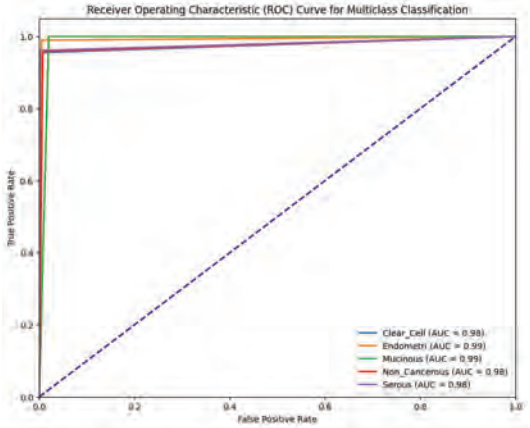
ROC curve: VGG16-A



ROC curve: VGG16-B

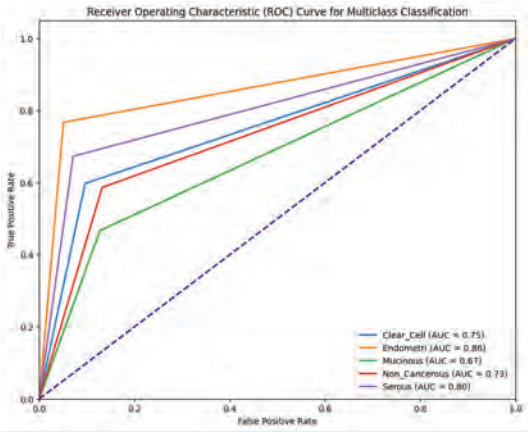


ROC curve: VGG16-C

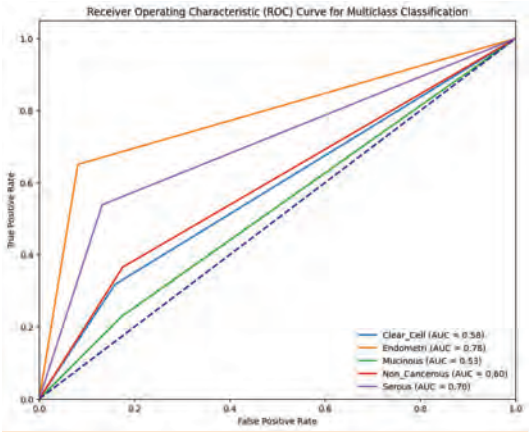


ROC curve: VGG19

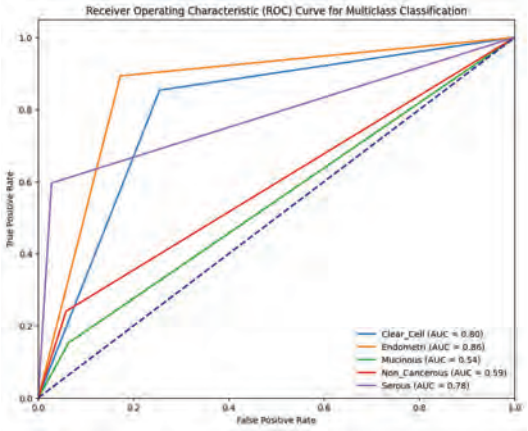
Figure 5.2: ROC Curve (ResNet & VGG)



ROC curve: LeNet-A



ROC curve: LeNet-B



ROC curve: LeNet-C

Figure 5.3: ROC Curve (LeNet)

5.1.2 XAI

The following image subplots from Figure-5.4 through Figure-5.14 are the generated outputs from XAI models.

LIME:

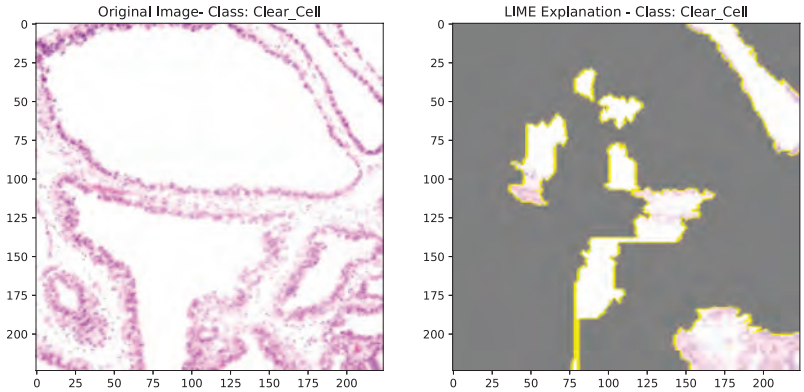


Figure 5.4: LIME (Class: Clear Cell)

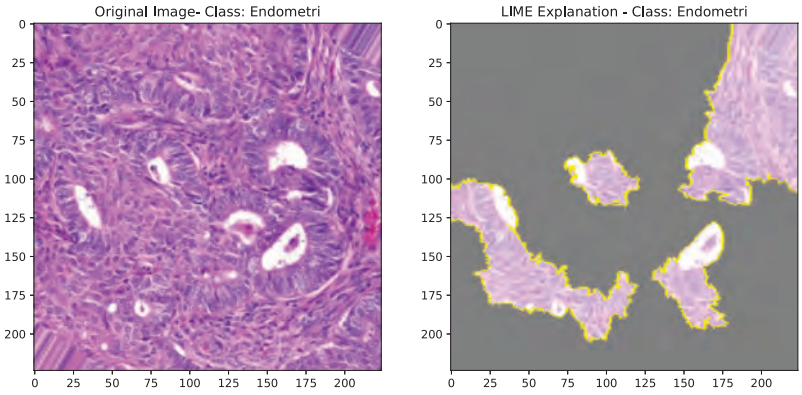


Figure 5.5: LIME (Class: Endometri)

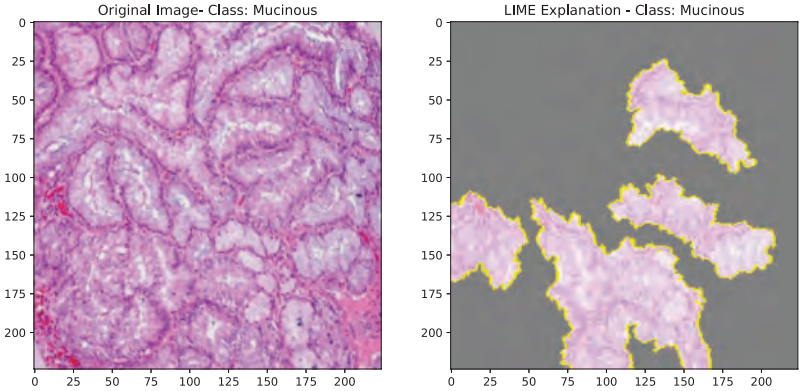


Figure 5.6: LIME (Class: Mucinous)

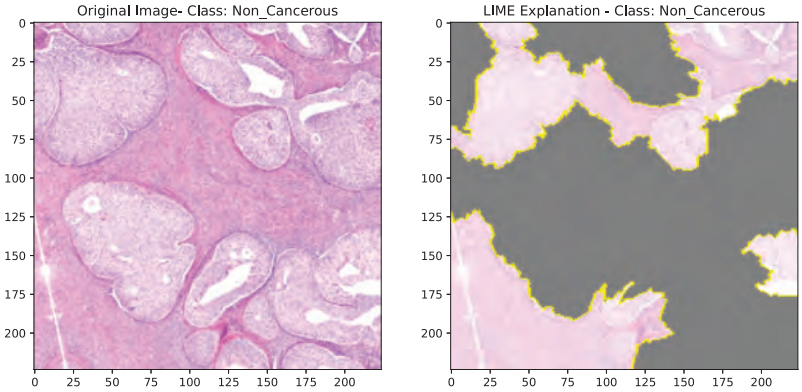


Figure 5.7: LIME (Class: Non Cancerous)

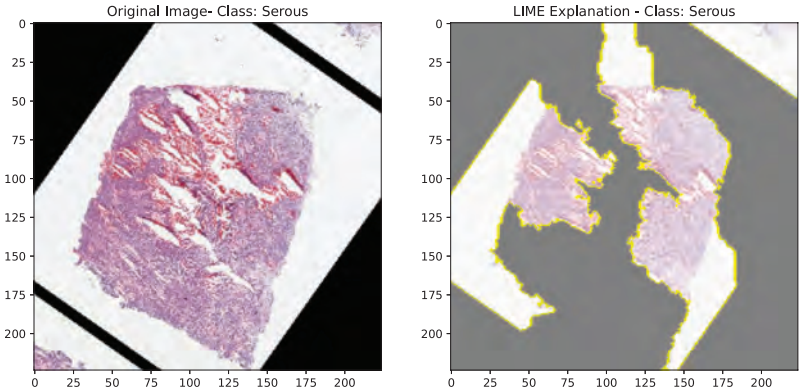


Figure 5.8: LIME (Class: Serous)

Integrated Gradients:

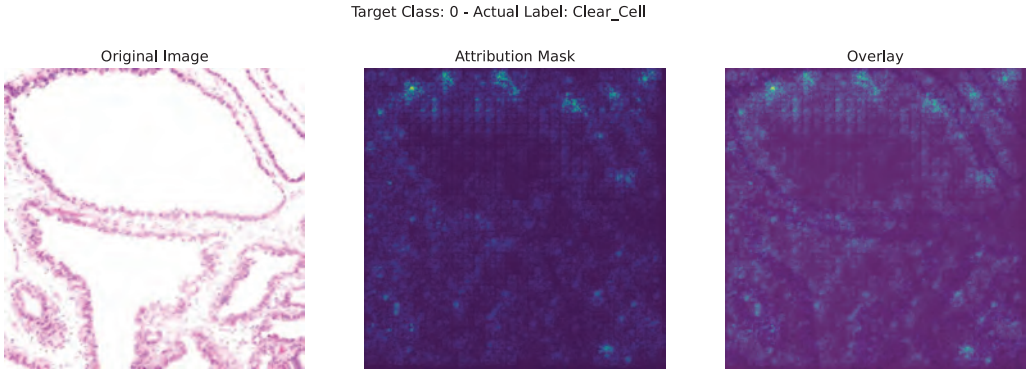


Figure 5.9: Integrated Gradients (Class: Clear Cell)

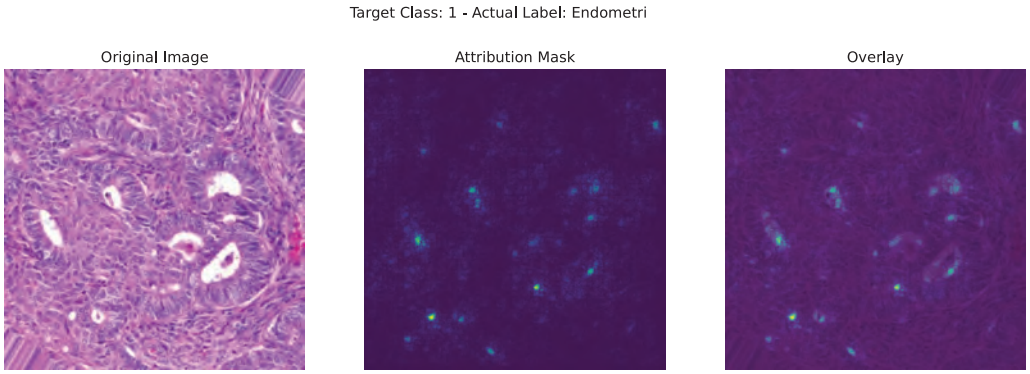


Figure 5.10: Integrated Gradients (Class: Endometri)

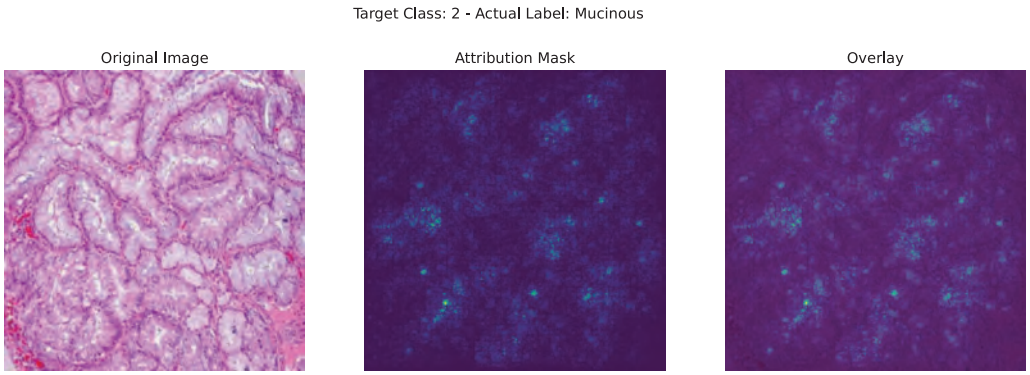


Figure 5.11: Integrated Gradients (Class: Mucinous)

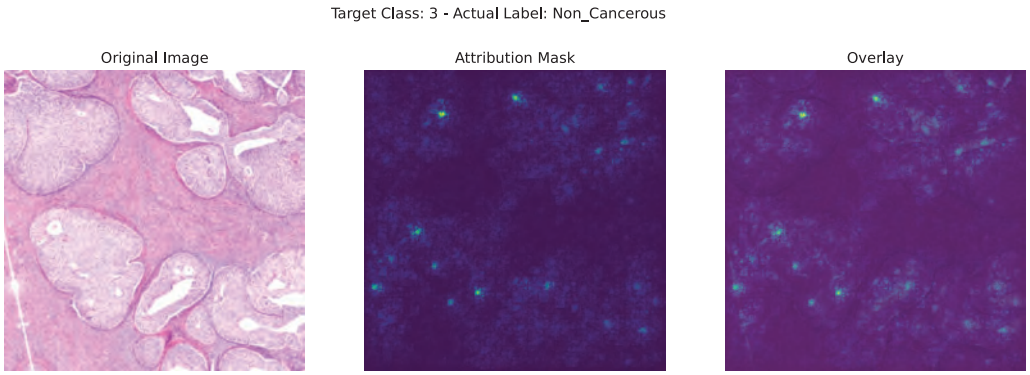


Figure 5.12: Integrated Gradients (Class: Non Cancerous)

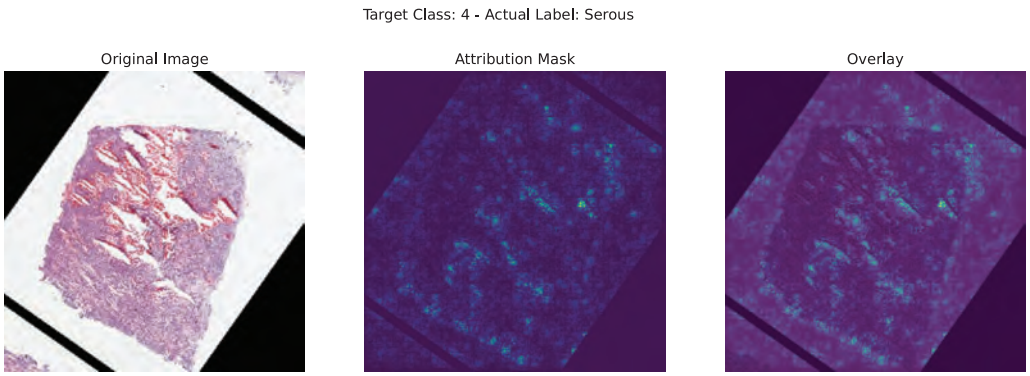


Figure 5.13: Integrated Gradients (Class: Serous)

SHAP:

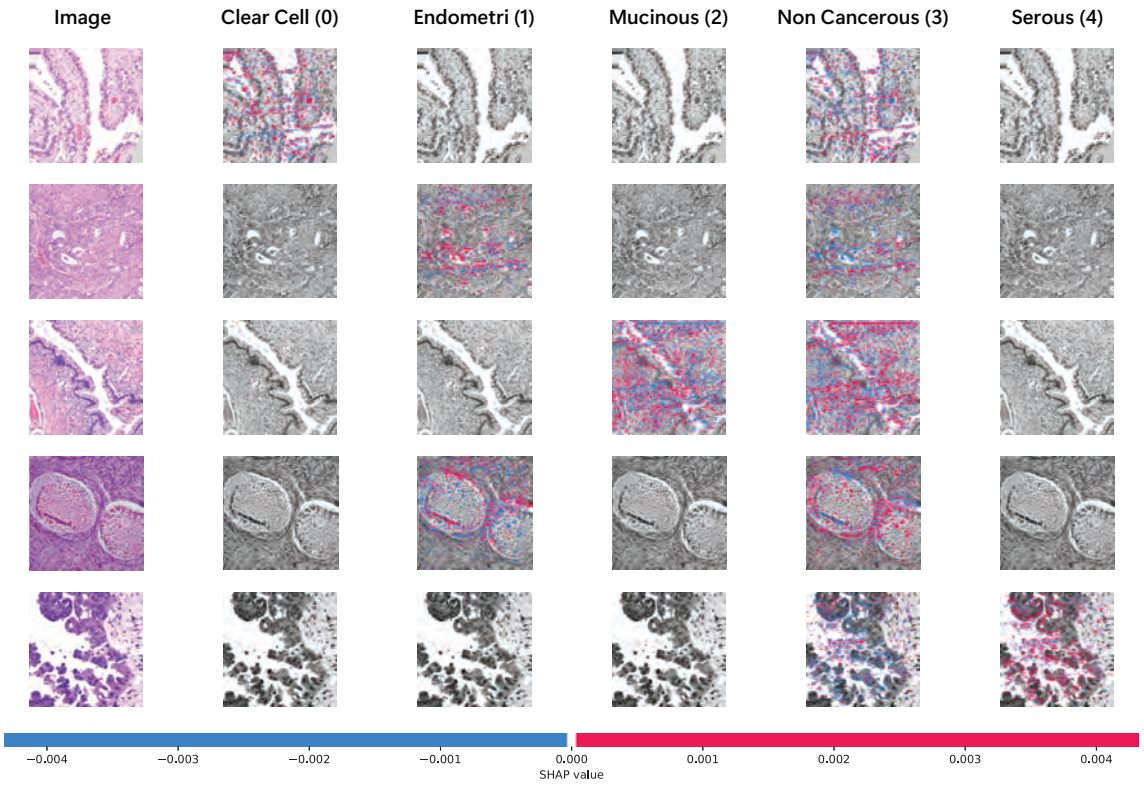


Figure 5.14: SHAP

5.2 Analysis

5.2.1 Base Model Selection

From the aforementioned result section, it can be observed that the highest results are in fact seen in VGG16's three variants, VGG19 and Inception V3's two variants. VGG16's three variants sported on average 96%+ on all the scores of accuracy, precision, recall and F1-Score while VGG19 sported 97%+ across the same fields. For VGG16-A, the lowest AUC score is for the Mucinous classifier at 0.95 and the highest for Serous classifier at 1.0. For VGG16-B, the lowest AUC score is for the Clear Cell classifier at 0.96 and the highest for Non-Cancerous and Endometri classifier at 0.99. For VGG16-C, the lowest AUC score is for the Clear Cell and Serous classifier at 0.96 and the highest for Endometri classifier at 1.0. Next, for VGG19, the AUC score is 0.99 for Endometri and Mucinous while the AUC score for the rest of the classifiers is 0.98. Next, the variant InceptionV3-A had 94.58%, 94.75%, 94.58% and 94.62% for scores of accuracy, precision, recall and F1-Score respectively. For AUC score, InceptionV3-A's lowest and highest are 0.91 for Mucinous classifier and 0.99 for Endometri classifier. Thus, it can be seen that utilizing transfer learning allowed us to achieve high scores across all the relevant fields for VGG while among the models that are built from scratch, Inception V3 performed the best. We ultimately decided to use InceptionV3-A variant of Inception V3 model.

There is a reason why InceptionV3-A has been selected and not the other models that were better performing. Let us first come to terms with the problems that will occur if VGG models were to be used. After the selection of a base model, we needed to work with explainable artificial intelligence or XAI to comprehend the black box answer that is produced via our selected model. The core of transfer learning makes it so that the utilization of XAI on models made via transfer learning is tremendously difficult when compared to that of a model that is built from scratch. Hence, the VGG models were rejected despite their high scores. Now, the model with the next highest score across all fields is InceptionV3-A. Thus, ultimately, our choice of model is the custom Inception V3 with ReLu activation function.

Model	VGG16-O[30]	**VGG16-A**	**InceptionV3-A**
Original Dataset	50%	77.78%	20.20%
Augmented Dataset	84.64% (20 epoch, 24742 images)	96.99% (80 epoch, 2490 images)	94.58% (80 epoch, 2490 images)

Table 5.2: Comparison of Average Model Accuracy between two of our models and one predecessor model.

Another thing that has been tested was our model score with the model of another paper by Kasture et al. that utilized the same dataset[30] (Henceforth, referred to as VGG16-O). According to Table-5.2, VGG16-O achieved a score of 50% with the non-augmented dataset. We also ran a minor test with our models VGG16-A and InceptionV3-A by running them for 20 epoch under our original conditions. The average accuracy achieved was 27.78% higher than that of VGG16-O. This can

be attributed to the fact that Tensor Conversion had been performed after image augmentation. By converting the images to Tensor Data and further normalizing the values in a range of 0 to 1, enabled for further computational efficiency and easier training for the base models. As for InceptionV3-A, the reason that it is performing much worse than VGG16-O and VGG16-A for the original dataset is because it is not pre-trained like the other models.

5.2.2 XAI

From Figure-5.14, it has been observed that the first image belonging to "clear cell" class and find that it has more positive correlation associated with the "clear cell" category in comparison to other classes. Similarly, upon examining the remaining images, a consistent dominance of positive features aligned with their respective actual target classes can be noticed, providing a solid and precise rationale for its specific classification.

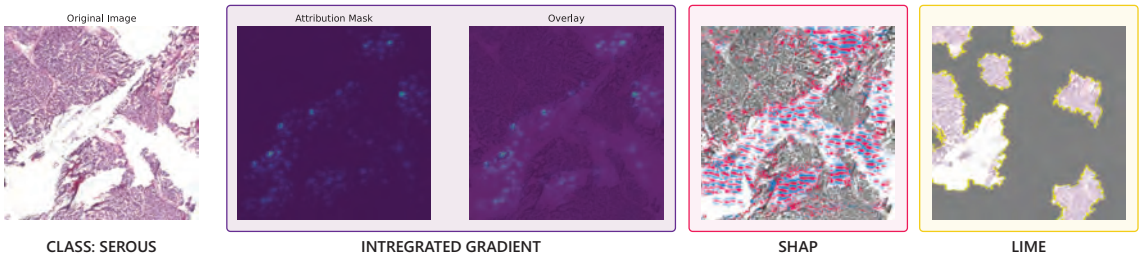


Figure 5.15: Comparative Analysis of Generated XAI outputs

In Figure-5.15, local visualized interpretation from LIME, SHAP and Integrated Gradients indicate that all three interpretations have similar highlighted features that contribute to the prediction of Serous Class. However, the reason why some highlighted features from SHAP and Integrated Gradients do not exist in LIME is that, LIME interpretation has been capped to showing only 10 important features to reduce complexity in analysis.

Chapter 6

Conclusion

Cancer is a highly invasive disease that forms due to the abnormal growth of cells in any part of the body. Ovarian cancers are considered to be more deadly than other common cancers among women because of its late-stage prognosis. A late stage prognosis often means a high risk of the cancer cells spreading to other organs and thus increasing the chance of mortality. In the United States of America, ovarian cancer is deemed as the deadliest gynecologic cancer. Due to its high lethality, researchers all over the world are attempting to find either a faster and accurate detection method or a non-invasive detection method. In this paper, an automated detection system has been created that utilizes Convolutional Neural Networks (CNN) to detect ovarian cancer fast and accurately. For building such a system, different CNN models such as LeNet-5/LeNet, Residual Neural Network (ResNet), VGGNet and GoogLeNet/Inception have been utilized. After testing various iterations of the CNN models, Inception V3 has been used as the base AI for this endeavour. Explainable Artificial Intelligence (XAI) models such as Local Interpretable Model-agnostic Explanations (LIME), SHapley Additive exPlanations (SHAP) and Integrated Gradients has also been implemented for this system so that the outcome of the system can be interpreted and judged accordingly. Ultimately, a great initial success has been achieved by building a sandbox InceptionV3 model with the selected model that achieved an average score of 94.5% to 94.75% in the performance metrics such as Accuracy, Precision, Recall and F1-Score. Moreover, the model also had one of the better ROC Curves and AUC scores when compared to the other 14 variations of the different CNN models that were experimented with. Next, a Comparative Analysis has been performed on the generated output of three different XAI models namingly LIME (Local Interpretable Model Agnostic Explanations), SHAP (SHapley Additive exPlanations) and Integrated Gradients with the results indicating that the generated outputs had some highlighted features that were common across the 3 models. This signifies that the black-box interpretation occurred successfully. Thus, it can be noted that an initial step was taken towards completing a system that can provide either an accurate, faster detection model or an early prognosis model. In the future, we aim to streamline the system and pivot towards early prognosis and faster detection by using non-invasive data as our image dataset.

Chapter 7

Future Works

We have taken an initial step towards our goal of achieving an early prognosis or faster detection method. However, there remains several areas where we can bring improvements. Such as:

- Utilize a larger, more comprehensive dataset with much variations in image classification. We will also utilize tissue samples and convert them to images to create a more distinct dataset.
- Streamline the current DCNN and XAI models into a singular structure. We will also work towards a mobile system that displays the XAI reports.
- Pivot towards an efficient, optimized DCNN model structure to achieve even faster detection process.
- Incorporate non-invasive image data from lab test mechanism to ensure a stable method of achieving early prognosis.

There are several more works that can be done in the future in this field. However, we will prioritize on the aforementioned objectives unless circumstances say otherwise.

Bibliography

[1] D. H. Hubel and T. N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex," *The Journal of Physiology*, vol. 148, no. 3, pp. 574–591, 1959. DOI: 10.1113/jphysiol.1959.sp006308.

[2] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, 1980. DOI: 10.1007/bf00344251.

[3] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, 1998. DOI: 10.1109/5.726791.

[4] C. Szegedy et al., "Going deeper with convolutions," 2014. arXiv: 1409.4842 [cs.CV].

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. arXiv: 1512.03385 [cs.CV].

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015. arXiv: 1409.1556 [cs.CV].

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," 2016. arXiv: 1603.05027 [cs.CV].

[8] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," 2016. arXiv: 1602.04938 [cs.LG].

[9] A. S. Bercow et al., "Cost of care for the initial management of ovarian cancer," *Obstetrics & Gynecology*, vol. 130, pp. 1269–1275, Dec. 2017. DOI: 10.1097/aog.0000000000002317.

[10] S. Lundberg and S.-I. Lee, *A unified approach to interpreting model predictions*, 2017. arXiv: 1705.07874 [cs.AI].

[11] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," 2017. arXiv: 1703.01365 [cs.LG].

[12] Scott Lundberg Revision, *Shap documentation - shap.deepexplainer*, shap-lrjball.readthedocs.io, 2018. [Online]. Available: https://shap-lrjball.readthedocs.io/en/latest/.

[13] A. B. Arrieta et al., "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," 2019. arXiv: 1910.10045 [cs.AI].

[14] A. Anwar, *Difference between alexnet, vggnet, resnet and inception*, Medium, Jun. 2019. [Online]. Available: https://towardsdatascience.com/the-w3h-of-alexnet-vggnet-resnet-and-inception-7baaaecccc96.

[15] Cancer Research UK, *Ovarian cancer statistics*, Cancer Research UK, Jul. 2019. [Online]. Available: https://www.cancerresearchuk.org/health-professional/cancer-statistics/statistics-by-cancer-type/ovarian-cancer.

[16] Google, *Classification: Roc curve and auc | machine learning crash course*, Google Developers, 2019. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc.

[17] W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl, and B. Yu, "Definitions, methods, and applications in interpretable machine learning," *Proceedings of the National Academy of Sciences*, vol. 116, pp. 22 071–22 080, Oct. 2019. DOI: 10.1073/pnas.1900654116.

[18] K. Smeda, *Understand the architecture of cnn*, Medium, 2019. [Online]. Available: https://towardsdatascience.com/understand-the-architecture-of-cnn-90a25e244c7.

[19] S. Verma, *Understanding 1d and 3d convolution neural network | keras*, Medium, Sep. 2019. [Online]. Available: https://towardsdatascience.com/understanding-1d-and-3d-convolution-neural-network-keras-9d8f76e29610.

[20] C. DeMarco, *How is ovarian cancer diagnosed?* MD Anderson Cancer Center, Feb. 2020. [Online]. Available: https://www.mdanderson.org/cancerwise/how-is-ovarian-cancer-diagnosed.h00-159616278.html.

[21] T. Kanstrén, *A look at precision, recall, and f1-score*, Medium, Oct. 2020. [Online]. Available: https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec.

[22] R. Wang et al., "Evaluation of a convolutional neural network for ovarian tumor differentiation based on magnetic resonance imaging," *European Radiology*, vol. 31, pp. 4960–4971, Oct. 2020. DOI: 10.1007/s00330-020-07266-x.

[23] R. Roscher, B. Bohn, M. F. Duarte, and J. Garcke, "Explainable machine learning for scientific insights and discoveries," *IEEE Access*, vol. 8, pp. 42 200–42 216, 2020. DOI: 10.1109/access.2020.2976199.

[24] P. Varshney, *Lenet architecture: A complete guide*, kaggle.com, 2020. [Online]. Available: https://www.kaggle.com/code/blurredmachine/lenet-architecture-a-complete-guide.

[25] P. Varshney, *Vggnet-16 architecture: A complete guide*, kaggle.com, 2020. [Online]. Available: https://www.kaggle.com/code/blurredmachine/vggnet-16-architecture-a-complete-guide.

[26] G. Boesch, *Vgg very deep convolutional networks (vggnet) - what you need to know*, viso.ai, Oct. 2021. [Online]. Available: https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/.

[27] D. Garreau and D. Mardaoui, "What does lime really see in images?," 2021. arXiv: 2102.06307 [cs.LG].

[28] R. M. Ghoniem, A. D. Algarni, B. Refky, and A. A. Ewees, "Multi-modal evolutionary deep learning model for ovarian cancer diagnosis," *Symmetry*, vol. 13, no. 4, 2021, ISSN: 2073-8994. DOI: 10.3390/sym13040643.

[29] K. R. Kasture, "Ovariancancer&subtypesdatasethistopathology," *data.mendeley.com*, vol. 1, Mar. 2021. DOI: 10.17632/kztymsrjx9.1.

[30] K. R. Kasture, B. B. Sayankar, and P. N. Matte, "Multi-class classification of ovarian cancer from histopathological images using deep learning - vgg-16," *2021 2nd Global Conference for Advancement in Technology (GCAT)*, Oct. 2021. DOI: 10.1109/gcat52182.2021.9587760.

[31] National Cancer Institute, *What is cancer?* National Cancer Institute, Oct. 2021. [Online]. Available: https://www.cancer.gov/about-cancer/understanding/what-is-cancer.

[32] N. K. Pilla, *Understand googlenet (inception v1) and implement it easily from scratch using tensorflow and keras*, Medium, Mar. 2021. [Online]. Available: https://nitishkumarpilla.medium.com/understand-googlenet-inception-v1-and-implement-it-easily-from-scratch-using-tensorflow-and-keras-5404239f361.

[33] S. A. G. Shakhadri, *What is resnet | build resnet from scratch with python*, Analytics Vidhya, 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/06/build-resnet-from-scratch-with-python/.

[34] C.-W. Wang et al., "Weakly supervised deep learning for prediction of treatment effectiveness on ovarian cancer from histopathology images," *Computerized Medical Imaging and Graphics*, vol. 99, p. 102 093, Jul. 2022. DOI: 10.1016/j.compmedimag.2022.102093.

[35] A. Chaturvedi, *Googlenet model*, www.codingninjas.com, May 2022. [Online]. Available: https://www.codingninjas.com/codestudio/library/googlenet-model.

[36] S. T. Hsu, Y. J. Su, C. H. Hung, M. J. Chen, C. H. Lu, and C. E. Kuo, "Automatic ovarian tumors recognition system based on ensemble convolutional neural network with ultrasound imaging," *BMC Medical Informatics and Decision Making*, vol. 22, Nov. 2022. DOI: 10.1186/s12911-022-02047-6.

[37] A. Kumar, *Different types of cnn architectures explained: Examples*, Data Analytics, Apr. 2022. [Online]. Available: https://vitalflux.com/different-types-of-cnn-architectures-explained-examples/.

[38] L. K. Hema et al., "Region-based segmentation and classification for ovarian cancer detection using convolution neural network," *Contrast Media & Molecular Imaging*, vol. 2022, pp. 1–12, Nov. 2022. DOI: 10.1155/2022/5968939.

[39] Ovarian Cancer Research Alliance, *Ovarian cancer statistics - usa*, OCRA, 2022. [Online]. Available: https://ocrahope.org/get-the-facts/statistics/.

[40] D. Schwartz, T. W. Sawyer, N. Thurston, J. Barton, and G. Ditzler, "Ovarian cancer detection using optical coherence tomography and convolutional neural networks," *Neural Computing & Applications*, vol. 34, pp. 8977–8987, 2022. DOI: 10.1007/s00521-022-06920-3.

[41] U.S. Government Accountability Office, *Machine learning's potential to improve medical diagnosis*, www.gao.gov, Nov. 2022. [Online]. Available: https://www.gao.gov/blog/machine-learnings-potential-improve-medical-diagnosis.

[42] World Cancer Research Fund International, *Worldwide cancer data | world cancer research fund international*, WCRF International, 2022. [Online]. Available: https://www.wcrf.org/cancer-trends/worldwide-cancer-data/.

[43] World Health Organization, *Cancer*, World Health Organization, Feb. 2022. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/cancer.

[44] J. Zhou, W. Cao, L. Wang, Z. Pan, and Y. Fu, "Application of artificial intelligence in the diagnosis and prognostic prediction of ovarian cancer," *Computers in Biology and Medicine*, vol. 146, p. 105 608, 2022, ISSN: 0010-4825. DOI: 10.1016/j.compbiomed.2022.105608.

[45] D. A. Binas et al., "A novel approach for estimating ovarian cancer tissue heterogeneity through the application of image processing techniques and artificial intelligence," *Cancers*, vol. 15, p. 1058, Feb. 2023. DOI: 10.3390/cancers15041058.

[46] MathWorks, *Googlenet convolutional neural network - matlab googlenet*, www.mathworks.com, 2023. [Online]. Available: https://www.mathworks.com/help/deeplearning/ref/googlenet.html.

[47] C. Munoz, K. da Costa, B. Modenesi, and A. Koshiyama, "Local and global explainability metrics for machine learning predictions," 2023. arXiv: 2302.12094 [cs.LG].

[48] TensorFlow, *Integrated gradients | tensorflow core*, TensorFlow, Oct. 2023. [Online]. Available: https://www.tensorflow.org/tutorials/interpretability/integrated_gradients.

[49] World Ovarian Cancer Coalition, *Ovarian cancer key stats**, World Ovarian Cancer Coalition, 2023. [Online]. Available: https://worldovariancancercoalition.org/about-ovarian-cancer/key-stats/.

[50] World Ovarian Cancer Coalition, *Ovarian cancer testing and detection*, World Ovarian Cancer Coalition, 2023. [Online]. Available: https://worldovariancancercoalition.org/about-ovarian-cancer/detection-testing/.

[51] World Ovarian Cancer Coalition, *What is ovarian cancer?* World Ovarian Cancer Coalition, 2023. [Online]. Available: https://worldovariancancercoalition.org/about-ovarian-cancer/what-is-ovarian-cancer/.

[52] Deepchecks, *What is resnet*, Deepchecks. [Online]. Available: https://deepchecks.com/glossary/resnet/.

[53] J. McDermot, *Hands-on transfer learning with keras and the vgg16 model*, www.learndatasci.com. [Online]. Available: https://www.learndatasci.com/tutorials/hands-on-transfer-learning-keras/.

[54] B. Priya, *Softmax activation function: Everything you need to know*, Pinecone. [Online]. Available: https://www.pinecone.io/learn/softmax-activation/.

[55] Qlik, *What is explainable ai? benefits and best practices*, Qlik. [Online]. Available: https://www.qlik.com/us/augmented-analytics/explainable-ai.