

Diagnosing Cervical Cancer with Deep Learning

A Thesis

submitted to

Indian Institute of Science Education and Research Pune in partial
fulfilment of the requirements for the BS-MS Dual Degree Programme

by

Snehal Bhartiya



Indian Institute of Science Education and Research Pune

Dr. Homi Bhabha Road,

Pashan, Pune 411008, INDIA.

April, 2020

Supervisor: Dr. Pranay Goel

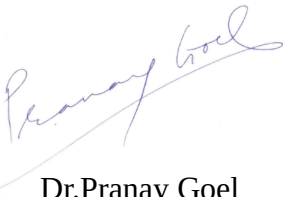
Snehal Bhartiya

All rights reserved

Certificate

This is to certify that this dissertation entitled **Diagnosing Cervical Cancer with Deep Learning** towards the partial fulfilment of the BS-MS dual degree programme at the **Indian Institute of Science Education and Research, Pune** represents study/work carried out by **Snehal Bhartiya** at Indian Institute of Science Education and Research under the supervision of **Dr.Pranay Goel, Associate Professor, Department of Biology**, during the academic year 2019-2020.

Committee:

A handwritten signature in blue ink that reads "Pranay Goel". The signature is written in a cursive style and is positioned above the printed name.

Dr.Pranay Goel

Dr.Richa Rikhy

This thesis is dedicated to the AI community who have made Deep Learning approachable and conducive for interdisciplinary researchers.

Declaration

I hereby declare that the matter embodied in the report entitled **Diagnosing Cervical Cancer with Deep Learning** are the results of the work carried out by me at the Department of Biology, Indian Institute of Science Education and Research, Pune, under the supervision of **Dr.Pranay Goel** and the same has not been submitted elsewhere for any other degree



Snehal Bhartiya

April, 2020

Acknowledgments

I would first like to thank my thesis adviser Dr. Pranay Goel for letting me venture into Deep Learning and guiding me through the process. He gave me the freedom to find my way through this research and giving me the time to learn while keeping me in the right direction.

I want to extend my acknowledgment to all the professors of IISER who made BSMS a great learning experience. Taking courses in Biology inspired me to understand myself better, and giving me an eye to appreciate the world as beautiful it is. I am deeply grateful to Dr. Amrita Hazra who was always just a knock away for help. Meetings with her have always been inspiring and I hope to find people like her in all walks of my life. I was also lucky to get an introductory course on Ecology and Evolution taught by Dr. Sutirth Dey which gave me a new interest for life, and led me to one of the wildest work experiences in the rain forests of Assam.

Additionally, I would also want to thank Dr. Richa Rikhy who was actively involved in validating this research project. Beyond a doubt, this project has been possible only because of the continuous help from my peers Alekh Mahankudo, Dhruv Khatri and Jaydeep Lohia. These guys helped me overcome the local minima, where I found myself getting stuck frequently. I cannot not mention about my senior Amar Alok, who gave me the long lost courage which I desperately needed to start writing my thesis.

Lastly, I would like to thank all the lab members, my friends and family for continuous support.

Table of Contents

1 INTRODUCTION.....	1
1.1 PROBLEM STATEMENT.....	1
1.2 PAP- TEST.....	1
1.3 COMPUTER AIDED DIAGNOSIS	2
1.4 STATE OF ART: DEEP LEARNING IN MEDICAL IMAGING.....	2
1.5 OBJECTIVE.....	3
1.6 PREVIOUS STUDES.....	3
2 DEEP LEARNING.....	6
2.1 ARTIFICIAL NEURAL NETWORKS.....	6
2.2 CONVOLUTIONAL NEURAL NETWORK.....	8
2.4 ACTIVATION FUNCTIONS.....	12
2.5 LOSS FUNCTION.....	14
2.6 OPTIMIZERS.....	14
2.7 TRANSFER LEARNING.....	17
3 TRAINING	22
3.1 DATASET.....	22
3.2 DATA PROCESSING.....	22
3.3 TRAIN TEST SPLITTING.....	23
3.4 REGULARISATION.....	24
3.5 PROGRAMMING FRAMEWORKS.....	24
3.6 LIBRARIES.....	24
4 METHODS	25
4.1 MULTI-LAYER PERCEPTRONS.....	25
4.2 CONVOLUTIONAL NEURAL NETWORKS.....	26
4.3 TRANSFER LEARNING.....	27
5 RESULTS AND DISCUSSION	34
5.1 MULTI-LAYER PERCEPTRONS.....	34
5.2 CONVOLUTIONAL NEURAL NETWORKS.....	37
5.3 TRANSFER LEARNING.....	39
5.3.1 Model Selection	39
5.3.2 Classifier Selection	40
5.3.3 Optimizer Selection	41
5.3.4 Two class Classification	42
3.5 Seven class classification.....	43
5.3.5 Columnar detection.....	44
5.3.6 Carcinoma detection.....	45
5.4 SUMMARY OF RESULTS.....	46
5.5 DISCUSSION.....	46
6 CONCLUSION.....	49
7 APPENDICES.....	50

List of Figures

- Fig. 2.1: Schematic representation of a Multi Layer Perceptron
- Fig. 2.2: Visualisation of a 2D convolution
- Fig. 2.3: VisualRepresentation of Max Pooling
- Fig. 2.4: Flatten layer
- Fig. 2.5: Schematic representation of connections in FC layer
- Fig. 2.6: Interconnection of neurons in FC layers
- Fig. 2.7: Sigmoid curve
- Fig. 2.8: ReLU activation curve
- Fig. 2.9: Schematic representation of Transfer Learning
- Fig. 2.10: Addition of a residual connection (below) to a plain layer [He K et al 2015]
- Fig. 2.11: Schematic of Resnet architecture. Adapted from ‘Very Deep Convolutional Neural Networks for Complex Land Cover Mapping Using Multispectral Remote Sensing Imagery’ ,by Mahdianpari et al. 2018. Remote Sensing .p 6 [Refer Appendices]
- Fig. 2.12: Original Inception module.
- Fig. 2.13: Factorized module of Inception-V3
- Fig. 2.14: Schematic of InceptionResnetV2. Adapted from ‘Very Deep Convolutional Neural Networks for Complex Land Cover Mapping Using Multispectral Remote Sensing Imagery’ ,by Mahdianpari et al. 2018.Remote Sensing.p 7[Refer Appendices]
- Fig. 4.1: Schematic representation of MLP constructed
- Fig. 4.2: Schematic representation of the Convolutional Neural Network
- Fig. 4.3: Importing models in tensorflow
- Fig. 4.4: Assigning the arguments for the model
- Fig. 4.5: Schematic representation of different classification architectures
- Fig. 5.1: Accuracy curve(y-axis) wrt epochs(x-axis) :Two class
- Fig. 5.2: Loss function (y-axis) wrt epochs(x-axis):Two class

Fig. 5.3: Loss curve(y-axis) wrt epochs(x-axis)

Fig. 5.4: Accuracy curve(y-axis) wrt epochs(x-axis)

Fig. 5.5: Loss(y-axis) wrt epochs(x-axis)

Fig. 5.6: Accuracy(y-axis) wrt epochs(x-axis)

Fig. 5.7: Accuracy (y-axis) wrt epochs(x-axis)

Fig. 5.8: Loss (y-axis) wrt epochs(x-axis)

Fig. 5.9: Loss varition(y-axis) wrt epochs(x-axis) with RMSprop

Fig. 5.10: Loss varition(y-axis) wrt epochs(x-axis) with Stochastic Gradient Descent

Fig. 5.11: Accuracy(y-axis) wrt epochs(x-axis) for two class classification

Fig. 5.12: Loss varition(y-axis) wrt epochs(x-axis) for two class classification

Fig. 5.13: Loss varition(y-axis) wrt epochs(x-axis) for seven category classification with InceptionResnetV2

Fig. 5.14: Accuracy(y-axis) wrt epochs(x-axis) for seven category classification with InceptionResnetV2

Fig. 5.15: Accuracy(y-axis) wrt epochs(x-axis) for detecting columnar class

Fig. 5.16: Loss(y-axis) wrt epochs(x-axis) for detecting columnar class

Fig. 5.17: Loss varition(y-axis) wrt epochs(x-axis)

Fig. 7.1: Confusion matrix for 7-class classification using GoogleNet-5.

Copyright© [2019] IEEE

List of Tables

Table 1: Comparitive analysis of existing studies on Herlev Database

Table 2: Herlev Database

Table 3: Comparitive Analysis of different models implemented for binary classification

Table 4: Comparitive analysis of different classifiers on InceptionResnetV2

Table 5: Performance comparision of various optimizers

Table 6: Comparision of models for seven category classification

Table 7: Summary of results

List of Abbreviations

ML	Machine Learning
DL	Deep Learning
TL	Transfer Learning
SGD	Stochastic Gradient Descent
ILSVRC	ImageNet Large Scale Visual Recognition Competition
Train_acc	Training Accuracy
Val_acc	Validation Accuracy
RMS	Root Mean Square
SCC	Sparse Cateogorical Crossentropy
IRV2	InceptionResnetV2

Abstract

Cervical cancer ranks as the fourth most prevalent cancer worldwide, affecting mostly the developing countries. However, early diagnosis can help facilitate the clinical management of the patient. The problem lies in the sparse presence of qualified and professional health cytotechnicians as compared to the number of people that need to be diagnosed. Computer-Aided Diagnostic system can be of a lot of help in making the diagnosis more accurate, reliable, faster and cheaper. Most of the existing algorithms require precise image segmentation to distinguish the cell. The traditional machine learning diagnostic system work similarly to the cytopathologists who rely on handcrafted morphological features such as nucleus area, nucleus-cytoplasm perimeter ratio, etc to determine the malignancy in a cell. However, our study uses Convolutional Neural Networks(CNN) which could potentially allow us to eliminate the computationally expensive tasks of segmentation and feature selection. Our results evidenced best accuracy scores of **96.25% for binary classification** and **66.87% for seven class classification**, which are comparable to the results achieved with established Machine Learning techniques. This study addresses the different aspects of training Deep networks on a publicly available cervical cancer database by Herlev Hospital. We also did a comparative investigation to establish the most suitable working hyperparameters, optimizers and classifiers for the dataset.

1 Introduction

1.1 Problem Statement

Cervical cancer is caused by Human Papillomavirus(HPV), out of the multiple types, HPV 16 genotype being the most carcinogenic. Being the fourth most common form of cancer, there were almost 570,000 cases in 2018, out of which 85% cases have been accounted in developing countries[Ferlay et al.2019; WHO press release]. Deaths brought about by cervical malignant growth can be hindered by having viable screening programs set up, which can fundamentally lead to a decrease in morbidity and mortality rate[Schwaiger et al.,2012]. Cervical cancer screening services are very sparse in developing countries due to the lack of qualified and professional health workers and limited health care resources to support screening programs[Mutyaba et. al. 2006]. This leads to a skewed ratio of the number of patients to be diagnosed and the number of cytotechnicians, however,lack of awareness can be regarded as one of the issues too. The most common methods used for screening comprise Human Papillomavirus DNA testing, Visual Inspection with Acetic Acid, Colposcopy and Liquid Cytology [Brown et al. 2012]. HPV DNA testing has shown to be highly effective in screening but is expensive and cumbersome [Brown et al. 2012].

1.2 Pap Test

Papanicolaou Test(Pap test), however is one of the most popular cervical cancer tests, which comprises taking a smear of cells from the cervix, placing them on a slide and observing the cells under the microscope. However, making pap-smears is time-consuming with each slide containing up to 300,000 cells, leading to the formation of clusters and making cell segmentation problematic [Chen et al 2014]. The cytotechnicians differentiate the cells based on color and shape properties(morphology) of the cell nuclei and cytoplasm. However performing a visual examination of population-wide screening is extremely tedious, time-consuming and requires at least two skilled cytotechnicians to make decisions.

1.3 Computer Aided Diagnosis

To tackle this problem, Computer-Aided diagnosing(CADx) systems have flourished in the past three decades. CAD's have shown promising performance in decreasing error rates and enable more efficient measurements. Notwithstanding critical research progress in these applications, the application of Computer-aided design frameworks in real scenarios has been quite difficult as these systems require labeled and annotated datasets. Creating these annotated datasets by a specialist is also tedious and costly [Le Lu et al. 2017]. There is a wide range of computer vision tasks that are highly relevant to this application field, such as automated smear variation handling; artifact identification; segmentation of individual cells and cell clusters; segmentation of nuclei and cytoplasm for each cell; and automated detection of irregular cell morphology changes[Teresa et al. 2019]

1.4 State of Art: Deep Learning in Medical Imaging

Historically most of CADx systems have been using machine learning classifiers comprising of Support Vector Machine, Random Forest and Decision trees, etc. However, these classifiers require handcrafted features and manually chosen morphological features. As Summers points out that hand-tuned parameters would not be very dependable when CAD systems would be applied to a new data [Le Lu et al. 2017]. Hence, there would still be a gap in training data and the real testing data in a clinical setting. These handpicked features from the training data would fail drastically when tested on real images in a clinical setting. However, Deep Neural Network (DNN) or Convolutional Neural Networks (CNN) showed a promising methodology of avoiding the idea of hand-picking the features. CNNs pick abstract features just by looking at the images and could potentially use them for making classifications. Deep Learning is quite new in medical image processing.

A major problem with most of the cervical cancer cell image dataset is that each slide of cells consists of hundreds of cells and therefore the priority is to have a good resolution of the images and have enough resolving power to be able to distinguish between the cells. Besides clustering in a plane, the cells on the slide are stacked in a multi-layer manner making it an even more challenging task to be able to distinguish between the cells. Ronneberger et al. showed the implementation of CNN for segmenting the cells based on pixel-level classification. However the

dataset that has been used in this study i.e Herlev dataset has images captured at the level of individual cells hence eliminating the need to segment the image at a cellular level.

Deep learning has a huge potential as it evolves to become better in coming years and hopefully replicate the similar growth in developing computer aided screening systems for cervical cancer.

1.5 Objective

In this study we are using Deep Learning algorithms for 2 class(cancerous and non-cancerous),3 class(normal, columnar and dysplastic) and 7 class classification. Unlike the previous methods which rely on feature extraction, we come up with a novel approach of extracting deep heirarchical features embedded in the cell images. This approach allows us to eliminate the essential need of segmentation, which is indeed a very challenging task. Mathematical computation of all the features and combat the error that comes with the calculation. Our techniques have shown considerably good performance as compared to the other machine techniques that have been developing for years.

The novelty of this approach lies in eliminating two of the most computationally complex, time-consuming and error-prone tasks that have been used by almost all CADx systems :

1. Segmenting the image to identify different regions explicitly.
2. Manual feature selection from visual morphological data.

The objective of this study is to construct Deep learning models from scratch and implement existing architectures to obtain a high classification score on Herlev database. Nevertheless,a proof of concept of the aforementioned technology would be required for these Deep Networks to replicate this performance equally well on clinical settings and other pap-smear datasets.

1.6 Previous Studies

Please refer to the Table 1.

Zhang et al. 2017 showed an accuracy and specificity of 98.3% for binary classification (cancerous and non-cancerous). The study uses ConvNet pre-trained network on ImageNet and fine-tunes it for cervical cancer images. The distinctive approach of this study is to locate the nucleus and use the nucleus abnormalities to provide substantially discriminative information. Lin et al. 2019 published a study where the nucleus and cytoplasm mask were concatenated to the RGB image channel. This allowed them to increase the channel number by two, hence providing specific morphological information about the nucleus and cytoplasm. They received an accuracy of 94.3% for binary classification and 64.5% for seven way classification.

Bora et al. proposed an intelligent system consisting three independent classifiers Multilayer Perceptron(MLP), Random Forest and Support Vector machines. They calculated multiple features and grouped them into four major categories. Multiple combinations of these features were fed to the classifiers to understand the principal features to make the classification. MLP got the highest classification accuracy of 96.51% as compared to other classifiers based on the selected features for two classes was attained and an accuracy of 91.71% was achieved by Support Vector Machine for 3 class classification.

Author	Paper	Technique used	Pre-processing	Classifiers	Results
Chakong et al. 2013	Automatic cervical cell segmentation and classification in Pap smears	Extracted morphohological feature for classification.	Cell Segmentation Median filters	Fuzzy-C means algorithm	99.27% for binary classification 93.78% for 7-class classification
Sreedevi et al. 2012	Pap smear image- based detection of cervical cancer	Nucleous area	Color conversions and contrast enhancements. Segmentation using Iterative Thresholding Method	New classification algoritm used	Binary classification: sensitivityof 100% and specificity of 90% achieved
Ampazis et al. 2004	Pap-smear classifcation using efficient second-order neural network	Classification based on combinations of different features(20) extracted	Contrast Enhancement and Segmentation using neural networks	LMAM and OLMAM optimization algorithms	Classifcation accuracy of 98.86% was obtained
Bora et al.2016	Pap Smear Image Classification Using Convolutional Neural Network	Morphological feature extraction using CNN	None	LSSVM and Softmax classifier	95.88 % accuracy for binary classification.
Zhang et al. 2018	DeepPap: Deep Convolutional Networks for Cervical Cell Classification	Resampled images with patches coarsely centered on nuclei	None	CNN	98.3% accuracy for binary classification.
Lin H et al. 2019	Fine-Grained Classification of Cervical Cells Using Morphological and Appearance Based Convolutional Neural Networks	Concatenated nucleous and cytoplasm segmented s mask with RGB images	None	CNN	94.3% for binary classification. 64.5% for seven way classification.

Table 1: Comparative analysis of existing studies on Herlev Database

2 Deep Learning

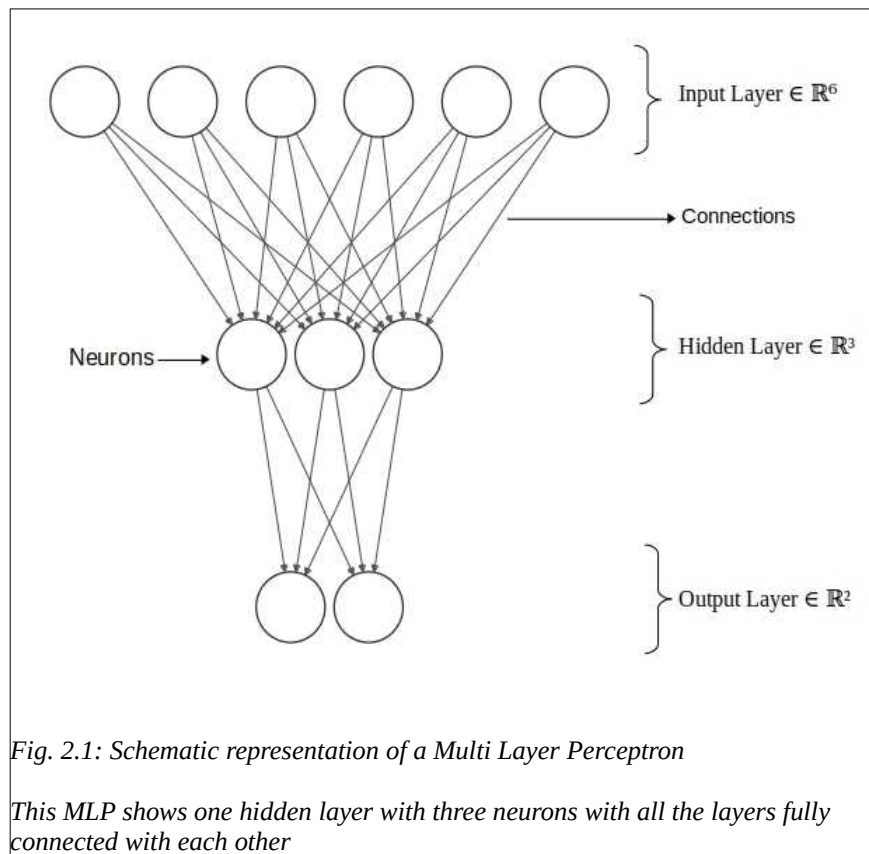
Machine learning (ML) is a practice of using algorithms to analyze data, learn from the data and then make predictions about new data. As compared to other algorithms that perform computation based on a given set of instructions to accomplish a particular task, a machine is *trained*. Using large amounts of data and algorithms that give it an ability to perform a task without being explicitly told how to do so. Let's take an example for comparing the methodology of doing a task with machine learning vs doing the same task with a traditional programming algorithm. For example, If we have to analyze the sentiment of popular posts in a specific area and classify the overall sentiment as positive or negative. With a traditional programming approach, we give our algorithm a list of words that can be classified as positive or negative. The words like sad, unfortunately, regret, won't, hard, sorry, etc can be stored in the negative category and words like happily, thankfully, great, etc can be stored in the positive category. This list can be provided to the traditional algorithm. In the end the algorithm can compare the number of positive and negative words that were used in the posts and classify that post as positive or negative based on the numbers. However, with the machine learning approach, there wouldn't be an algorithm with an explicit set of words to segregate. Instead an ML algorithm would be trained to iterate over a large amount of positive and negative posts. Training allows it to learn abstract features/words from the given training data and use that knowledge to classify a new post into negative or positive.

Deep learning (DL) is a part of a more extensive family of machine learning which uses algorithms inspired by the structure and function of a biological brain's neural network. These algorithms can be trained using supervised and unsupervised learning. Supervised learning occurs when our model learns and make inferences from data that has already been labeled. Unsupervised learning occurs when the model learns and makes inferences from unlabelled data.

2.1 Artificial Neural Networks

Artificial neural systems (ANNs) were motivated by the data processing and dispersed communication structure of real biological systems. However, ANNs are slightly different in contrast to real brains. In particular, neural systems are in general very static and representative, made of a collection of units called artificial neurons. Each connection between these neurons can transmit a signal from one neuron to another with the receiving neuron again

processing the signal and transmitting the signal to the downstream neurons connected to it. These neurons are organized in the form of layers, with different layers performing different kinds of transformations on their inputs. The signal is passed from the starting layer called an Input layer to the last layer called Output Layer. Any layer between the input and output layers are called hidden layers.



Multi-Layer perceptrons (MLP) are a kind of artificial neural network, which uses backpropagation for updating the parameters and minimizing the overall loss. MLPs have been widely used for pattern recognition tasks. Prominently known for their ability to model any function, they are also called universal approximators. A schematic representation of an MLP with 3 standard layers is shown in Fig. 2.1.

MLP can be viewed as a logistic regression classifier where the input is first transformed using a learnt non-linear transformation. This transformation projects the input data into a space where it becomes linearly separable allowing us to segregate data which are not. MLPs are a class of artificial neural networks and are capable of classifying non-linearly separable data. Experiments

were conducted in the initial phase to approximate the ability of a network to classify using non-linear combinations of data.

Khotanzad A et al. 1998: “The function mapping ability of the MLP allows it to act as a feature space divider and create complex decision boundaries. This property makes the MLP an ideal classifier and a powerful alternative to the conventional classification approaches”.

Neural networks have been used in computer vision problems since the 1980s but have been dormant until the development of graphic processing systems emerged in the mid-2000s. Mashor et al. 2000 showed that multilayered perceptrons can be widely used in non-linear modeling by using a composition of multiple neural layers.

2.2 Convolutional Neural Networks

Convolutional Neural Networks are a form of Artificial Neural Networks that have been vastly used for analyzing images. CNNs have an ability to detect patterns in the data, which makes them so useful for image analysis. Hidden layers in a CNN called ‘convolutional’ layer differentiates it from other ANN’s. The neurons in these layers have weights and biases which are modulated through training. The different layers in a CNN are explained in further subsections.

2.2.1 Convolution Layer

Convolutions are operation that compute the integral of the product of two functions. Convolution applied on images, is the sum of pixel wise multiplication of the image vector and convolution weight matrix [Fig. 2.2]. The hyperparameters for the layer include number of filters, filter size and stride length. The weights of the filters are randomly initialized and optimized through the training process.

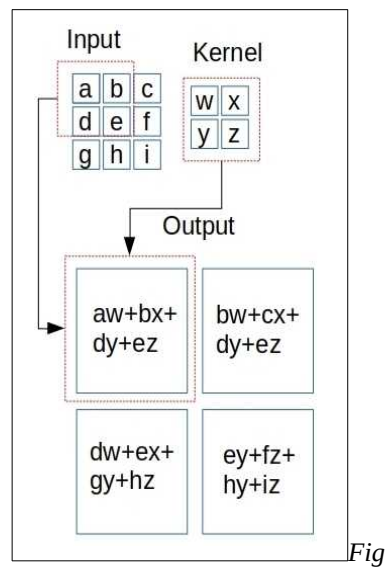


Fig. 2.2: Visualisation of a 2D convolution

2.2.2 Pooling Layer

The Pooling layer is used to extract the summary statistic for a given feature map while maintaining spatial invariance and is usually added post convolutional layer. When added to the model, they alter the dimensionality of the feature maps from the previous layer and output an upsampled/downsampled depending on the type of pooling used. One of commonly used pooling is Max Pooling [fig 2.2.2]. It outputs the maximum values of the image pixels within the kernel as compared to Average Pooling, which computes the average value for the input image pixels within the kernel.

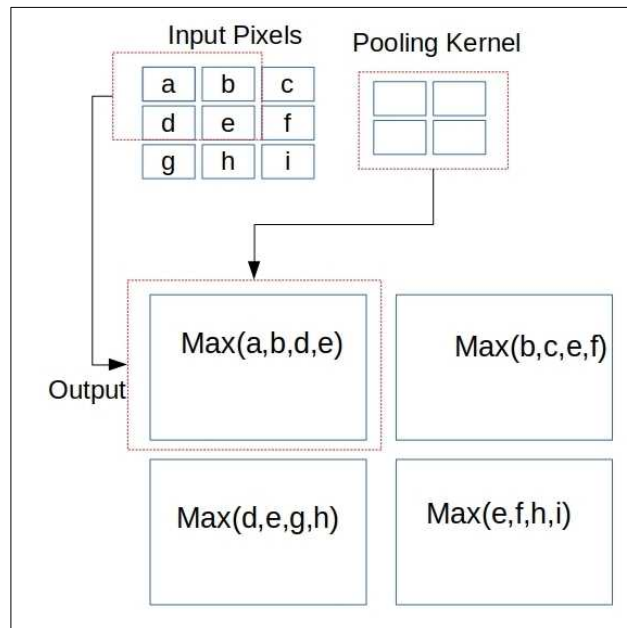


Fig. 2.3: VisualRepresentation of Max Pooling

2.2.3 Flatten Layer

This layer creates a one-dimensional array of the feature maps created by the previous layers[refer Fig. 2.4]. Considering the dimension of the input feature map is (m,n,o) where m, n are the height, width of the map and 'o' represents the number of channels, the output of the flatten layer would have a length of $m*n*o$.

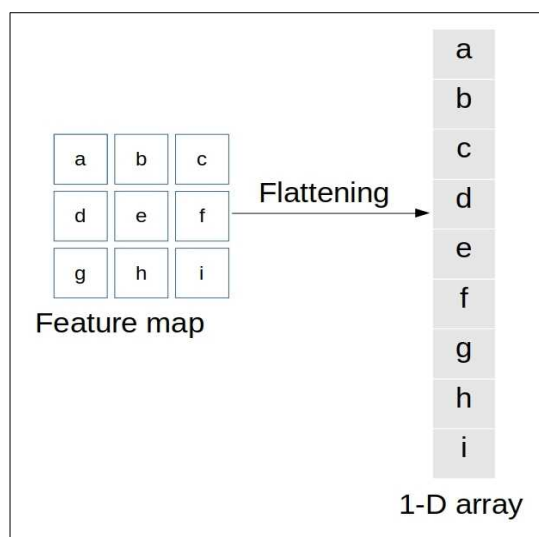
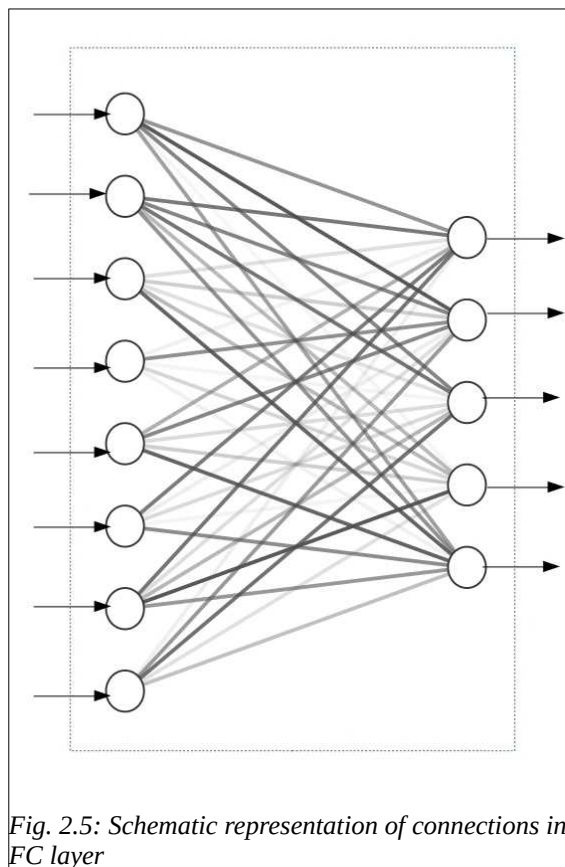


Fig. 2.4: Flatten layer

2.2.4 Fully Connected Layer

FC layers or Dense Layers extract information from the features extracted from the Conv layers. This extracted information is then used to train a softmax classifier for making predictions. Each FC layer is a mapping from R_m to R_n . Performing a simple computation of $W*x + B$, where W represents the weight from the previous layer and B is the bias term. The addition of a bias term acts as a threshold for activation of the various neurons in that particular layer. Shown in Fig. 2.5 is a schematic representation of connections between neurons in two fully connected layers.



2.3 Parameters : Weights and biases

The parameters that make up a neural network model include weights of each node and biases for every layer. A neuron is interchangeably used with words such as node, kernel, and filters. Each node is a simple matrix of real numbers for computing activations. The numbers in each matrix are optimized during training. The 'weight' parameter quantifies the connection strength between two neurons. In Fig. 2.6, red lines show positive correlation and blue lines show negative correlation. This blue connection shows that increasing the weight would decrease the output.

efficiency and increasing the weights of neuron with red connection would improve the efficiency. No connections between the neurons imply the bias has term has decreased the particular weight for the neuron to zero and changing that weight wouldn't impact the output.

Bias assigns a threshold value for a neuron to get activated, regulating the number of neurons that can transmit a signal to a downstream layer.

The hyperparameters in a layer include the total number of filters and dimensions of the filters. A dummy convolution layer shown below has 10 kernels. Each kernel has 5 rows and 5 columns, making up 25 elements, resulting in 251 ($10*5*5+1$) parameters for one layer.

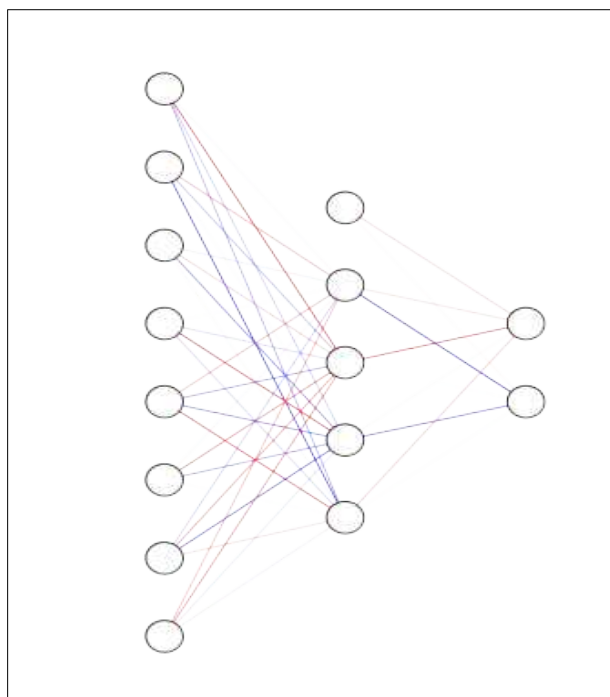


Fig. 2.6: Interconnection of neurons in FC layers

2.4 Activation Functions

Activation functions are used for inculcating non-linearity into the system makes neural networks universal approximators and giving them an ability to model complex tasks. They determine if a neuron should transmit the signal. Neural networks without activation functions would just be linear combinations of weights and biases and would be similar to linear regressions.

2.4.1 Sigmoid activation

The computation of weights(x) by a sigmoid function is expressed as:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

This function squishes the weights in a range between 0 to 1, with the outputs

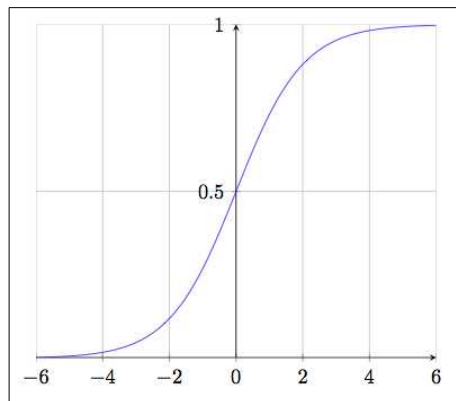


Fig. 2.7: Sigmoid curve

2.4.2 ReLU

ReLU has replaced sigmoid and tanh activation as it makes computations faster without affecting the learning significantly. They output the same entity for positive values and zero for negative numbers as represented in Fig.2.8.

Expressed as:

$$f(x) = x, \text{ if } x > 0$$

$$f(x) = 0, \text{ if } x \leq 0$$

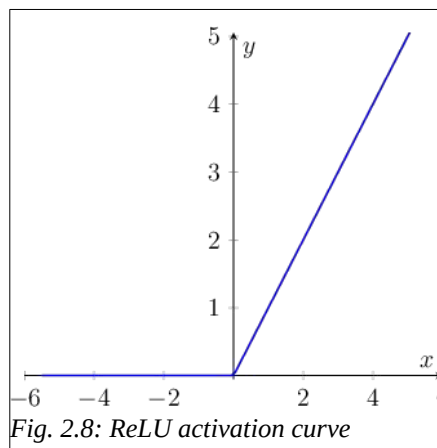


Fig. 2.8: ReLU activation curve

2.4.3 Softmax activation

We have used softmax activation in the last layer as it takes input as vector of n real numbers and normalizes into a probability distribution into n probabilities.

The formulae for computing the softmax probability is given below:

$$\text{Softmax}(x) = \frac{e^x}{\sum e^x}$$

2.5 Loss Function

Loss is calculated by finding the difference in the model's predicted output and the real output. The model calculates the error on each input by looking at what output is predicted for that input and the difference between the predicted value to the true value for that input. Overall loss for the whole data is then calculated by taking an average of all the errors commonly referred to as Mean square error(MSE). The optimizer then updates the parameters after every iteration with an aim to reduce the overall loss.

However, there are more sophisticated ways of calculating losses and we have majorly used **Sparse Categorical Crossentropy(SCC)** in our study. For discrete probability distributions, the crossentropy of q relative to p is expressed as :

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

SCC is used when the categories are mutually exclusive with each other and do not share probabilities. Since this loss function uses one hot embeddings for comparison, saving computational memory and hence increasing the overall speed.

2.6 Optimizers

The objective of the optimization algorithm is to find the model parameters for which the loss function is minimum. During the training process, the model calculates the loss of input by quantifying the difference in the outputs and the correct label for that input. The loss function is then minimized using optimization algorithms that continuously updates the weights of the model

during training. The value of loss function is expected to decrease as the model parameters get updated in every epoch.

The whole optimization is dependent upon forward and backpropagation. Forward propagation is simply the transmission of the signal from the input layer, performing activations in the and using them for making predictions in the final layer. These

predictions are then compared to the true labels, from which the loss is computed. With an objective to decrease the overall loss(J), the parameters are modified. The value for the weights (θ) is computed using the formulae :

$$\theta = \theta - \eta \frac{\partial J}{\partial \theta}$$

Here, J represents the loss function and η represents the learning rate.

The derivative can also be written as the gradient of the loss function, $\frac{\partial J}{\partial \theta} = \nabla J$.

The derivatives of the loss function gives an estimate of weight change. The proportion in which the weights are updated depends on how much does the update contributes to decreasing the loss. However, there are better strategies for updating the weights and three of them described below have been used in our study.

2.6.1 SGD with momentum

The traditional Gradient Descent updated the model parameters after completing the iteration over the whole training data. This makes the process extremely slow. Stochastic Gradient (SGD) updates the model parameters after iterating over *every* training example.

$$\theta = \theta - \eta \nabla J(\theta; x^{(i)}, y^{(i)})$$

Here, x^i represents the training example and y^i represents the corresponding label.

This does increase the computational speed, however, it may end up creating huge fluctuations in the loss curve. It has been seen that SGD hesitantly progresses around the local minima decreasing the overall training speed. This problem is dealt by **Adding Momentum**. Momentum [Nesterov 1983] accelerates the SGD process by dampening oscillations in irrelevant directions. This is done by adding a fraction (γ) of the update vector of the past time step (v_{t-1}) to the current update vector :

$$v_t = \gamma v_{t-1} + \eta \nabla J(\theta)$$

$$\theta = \theta - v_t$$

Here, γ denotes the momentum constant.

Gabriel Goh has given a beautiful working illustration of the optimizer and the impact of different hyperparameters on optimization process on his Distil blog “Why momentum really works” which can be found at <https://distill.pub/2017/momentum/>.

2.6.2 RMSprop

RMSprop[Hinton G. et al] solved one of the major problems in optimization which is to choose a suitable learning rate. . A high η can overshoot, and may miss out on local minima, whereas a high η can make the training process extremely slow. The idea was to optimize the learning rate(η) continuously.

This technique modifies learning rate throughout the training process by using adaptive learning rates and is computed by:

$$\theta_{t+1} = \theta - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

The learning parameter are reduced by a mean value of previous gradients.

A small smoothig term (ϵ) is added to keep denominator non-zero.

The mean value of the gradient is recursively calculated using the sum of decaying average of all past squared gradients:

$$E[g^2]_t = 0.9 E g^{2_{t-1}} + 0.1 g_t^2 \quad \text{where, } g_{t,i} = \nabla_{\theta} J(\theta_{t,i}) .$$

$E[g^2]_t$ refers to the decaying average of square of gradients which depends on the previous average and the current gradient. $E[g^2]_{t-1}$ refers to the decaying average of previous square of gradient and $g_{t,i}$ refers to the gradient of loss function for a given parameter i at a time t .

2.6.3 ADAM

Adaptive momentum is another techniqe which uses adaptive moments of gradient. However, as compared to RMSprop which optimizes learning based on the first moment(mean), second moment of the gradient (variance) is also used. The algorithm has been adapted from Kingma et al. 2015.

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v_t + \epsilon}} m_t$$

Here, the first moment of gradient is expressed as $v_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$

and the second moment gradient v_t expressed as: $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$.

2.7 Transfer Learning

Transfer Learning is one of the most powerful ideas in Deep Learning, which says that sometimes we can take the knowledge that a neural network has learned from one task and apply that knowledge to a different task. This technique eliminates the need for a large training dataset and the costly computational time that is required to train a model from scratch. For example, If a network learned to recognize an object like cars then we could potentially use that knowledge to help us to do a better job at recognizing trucks. Radiological Imaging is one such field where this technique has been heavily extrapolated i.e a network trained to recognize everyday objects could be used to differentiate medical images.

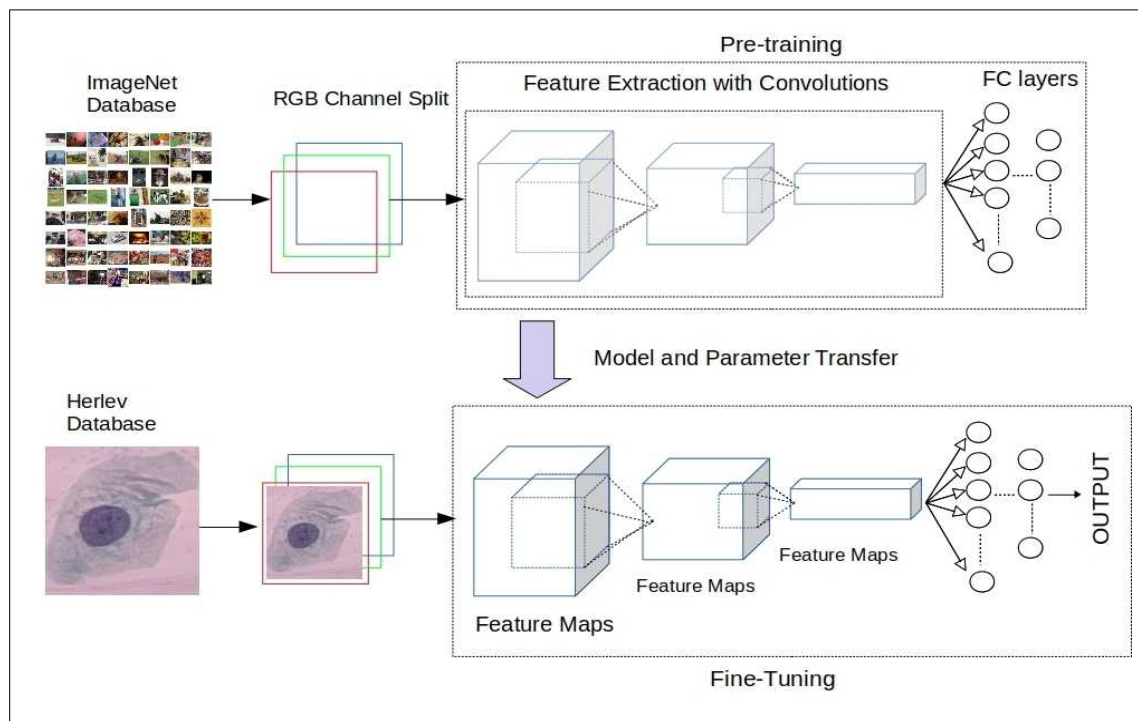


Fig. 2.9: Schematic representation of Transfer Learning

Multiple models pre-trained on ImageNet database were used fine tuned on Herlev Database. However, the classification layers were removed and a new softmax classifier was trained for the job.

Definition [Lin et al. 2017] : Given a task T_s in a source domain D_s and a learning task T_t in a target domain D_t . Transfer Learning aids to improve the target predictive function using the knowledge from the source domain and learning tasks where $D_s \neq D_t$ or $T_s \neq T_t$.

The process involves transferring the models from the original model trained on a particular dataset and fine-tuning the parameters to fit a new dataset. The models implemented this study won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) to form the base of our model for extracting features from the dataset and concatenate several fully connected layers with randomly initialized weights.

These models were pre-trained on the ImageNet dataset [<http://www.image-net.org/>] which contain over ten million images classified into 1000 classes.

Keras applications provide these pre-deep learning models that are made available as packages that can be used for defining, training and evaluating models. These models can be used for prediction, feature extraction, and fine-tuning.

2.7.1 Resnet152V2

Resnet was one of the revolutionary studies published in deep learning with around 42k citations which successfully dealt with the aforementioned problems and won **the ILSVRC-2015 competition** [He K et al 2015]. The major problems while training a deep neural network are :

- **Vanishing and Exploding gradient**

Vanishing Gradient: The gradient becomes vanishingly small due to the weights of earlier layers. Since the weights of the final layers are updated by using this gradient, an extremely small gradient would lead to a very small or negligible change in the weight. Therefore, the weights do not get updated and the network fails to converge to a local minimum.

Exploding Gradient: When the weights of the initial layers are more than 1, it would multiplicatively lead to a way bigger gradient and thus exploding in size. With this gradient, the weights are updated by a great number, and perhaps an optimal value of the would never be achieved.

- **Degradation Problem:** The problem of a deep network is that the overall error increases as we add more layers. This was explicitly found by K He et al. 2015, where two plain models (without residual connections) of 20 and 56 layers were trained. It was observed that the training and test accuracy were significantly better for the 20 layered network. This problem

was dealt with an addition of residual (shortcut connection) from input to the output of the stacked layer.

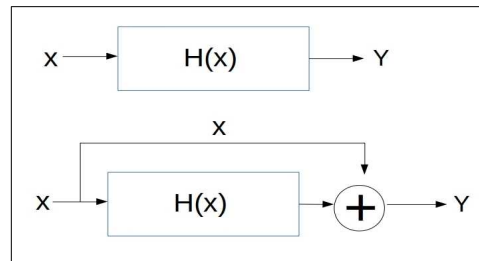


Fig. 2.10: Addition of a residual connection (below) to a plain layer [He K et al 2015]

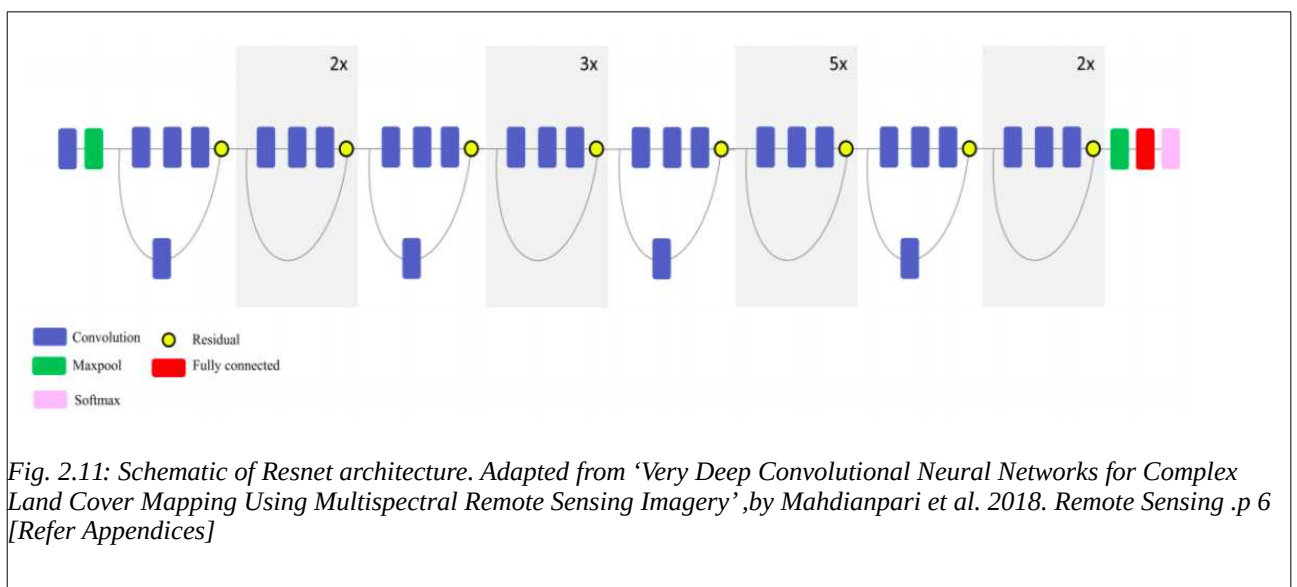


Fig. 2.11: Schematic of Resnet architecture. Adapted from ‘Very Deep Convolutional Neural Networks for Complex Land Cover Mapping Using Multispectral Remote Sensing Imagery’, by Mahdianpari et al. 2018. Remote Sensing .p 6 [Refer Appendices]

The Vanishing/Exploding gradient problem was dealt with by adding Batch Normalisations (Batch Norms) after every layer. Batch norms as the name suggests normalizes the weights from each layer, compressing them from a value of 0 to 1. Resnet152V2 is an upgraded version of ResnetV2, which improves the impact of pre-activation for tackling the feature degradation problem in deep networks by two folds. The idea is to add residual connections, which turns the network into its counterpart residual version, inspired from an earlier version of the model published by K. He. et al. 2015. The residual connections [shown in fig.2.10] are basically identity mappings between the input and output of stacked layers.

2.7.2 InceptionV3

Inception V1 also known as GoogLeNet was introduced by Szegedy et al. 2015. With only 42 layers, it placed as the first runner up for ILSVRC 2015. Szegedy et al. introduced the idea of Batch Normalization in version 2, published in 2016. As explained in 2.7.1, Batch Norms overcomes the well know Vanishing/Exploding gradient. This model was further improved by adding factorizing convolutions.

Inception V3 [Szegedy et al. 2015], found a way of dealing with the overcomplexity of the model by reducing the number of parameters in each Inception block, termed as factorizing convolutions. Fig 2.12 and 2.13 are one of the many modules. They substituted one $n \times n$ filter with two $m \times m$ and $m \times m$ filters, where $n > m$.

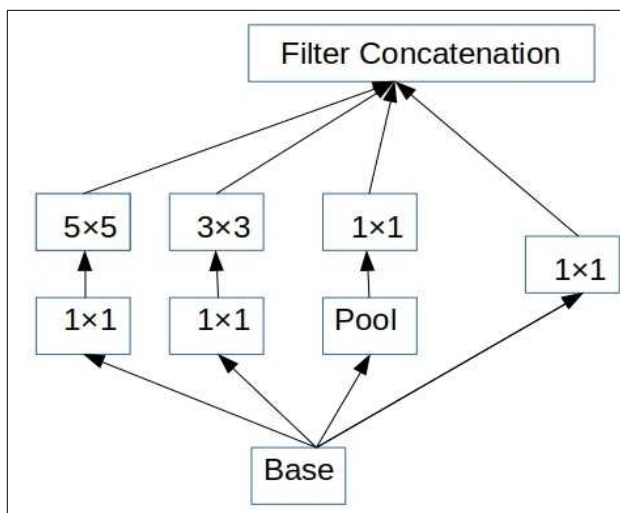


Fig. 2.12: Original Inception module.

(Adapted from Szegedy et al. 2015)

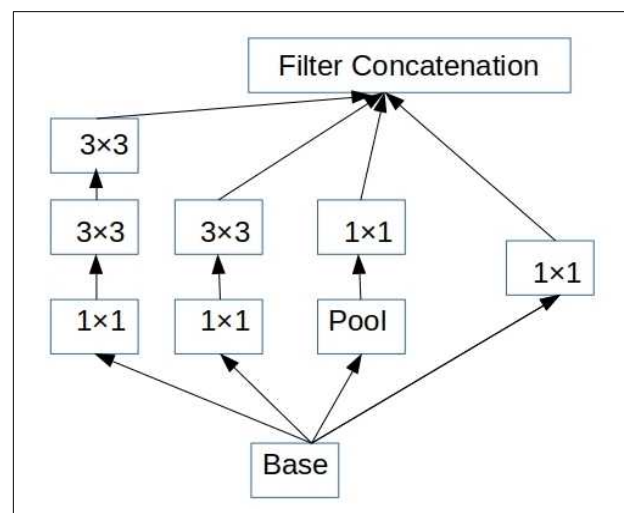


Fig. 2.13: Factorized module of Inception-V3

(Adapted from Szegedy et al. 2015)

For this particular Inception module, they reduced the number the complexity by 28%. No. of parameters with 5×5 filter = 25, whereas the total parameters with 3×3 and 3×3 filter = $3 \times 3 + 3 \times 3 = 18$.

Increasing factorisation[fig.2.13] technique led to a huge decrease in the computational cost and reducing the overall model size while maintaining the performance of the model.

They have also used auxiliary layers in between the inception blocks, to make predictions with a significantly lower level of feature maps.

2.7.3 InceptionResnet-V2

InceptionResnetV2 (IRV2) concatenated the working ideas of ResnetV2 and InceptionV3 to create a hybrid yet version out of the two. In the below diagram shows the compressed architecture of architecture shown above. IRV2 has a higher number of Inception models as compared to InceptionV3, however, the performance does not decrease, which is usually expected as we increase the layers. This could be credited to the residual blocks added which keep a check that the signal does not degrade out while training.

Factorization that comes from InceptionV3 has been applied in every module of IRV2 to reduce the number of parameters and hence the overcomplexity. Also, adding skip connections like ResnetV2 provides identity mapping from the previous layers to overcome the problem of feature degradation in deep networks. Residual connections have allowed researchers to make even deeper models. We observe that adding residual connections to the Inception module improved the testing performed on our dataset while keeping the computational cost almost similar.

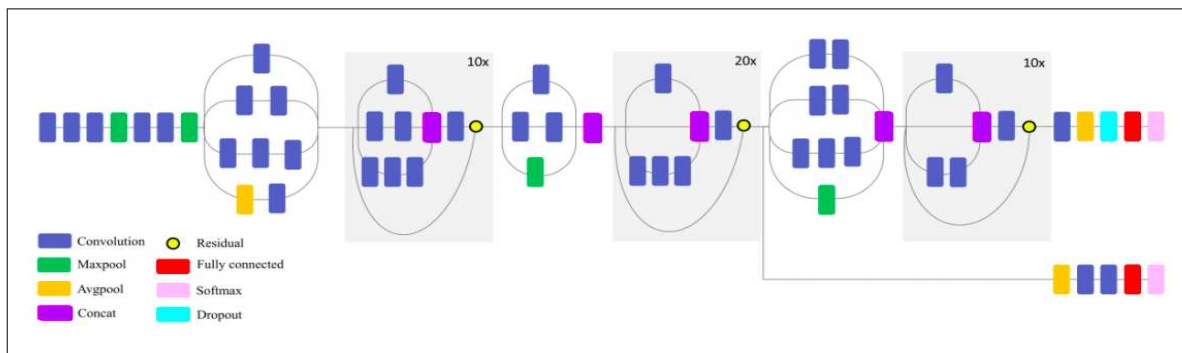


Fig. 2.14: Schematic of InceptionResnetV2. Adapted from 'Very Deep Convolutional Neural Networks for Complex Land Cover Mapping Using Multispectral Remote Sensing Imagery', by Mahdianpari et al. 2018. Remote Sensing, p 7 [Refer Appendices]

3 TRAINING

3.1 Dataset

Herlev database is a publicly available benchmark database for cervical cancer created by Herlev University Hospital in collaboration with Norup and Jantzen et al.2005. This database consists of 917 sample images distributed over 7 categories representing the cell type. It was created by a method called Papanicolaou Smear (Pap-smear). The slides were prepared by spreading a cytological sample on a glass slide. To clearly distinguish between the different regions, the cells were stained. The classification was based on “The Bethesda System” as described by Solomon et al. in 2004.

Category	Class	Cell type	Number of Images
Normal	1	Superficial squamous epithelial	74
Normal	2	Intermediate squamous epithelial	70
Normal	3	Columnar epithelial	98
Abnormal	4	Mild squamous non-keratinizing dysplasia	182
Abnormal	5	Moderate squamous non-keratinizing dysplasia	146
Abnormal	6	Severe squamous non-keratinizing dysplasia	197
Abnormal	7	Squamous cell carcinoma in situ intermediate	150

Table 2: Herlev Database

3.2 Data processing

Most of the images are single-celled. There are a few images with multiple cells and essential categorization is important, as two cells from different cell types might create a misinterpretation. Since the dataset was prepared in 2003, the quality isn’t comparable to the cervical cell images captured with the current state of the art microscopy techniques.

3.2.1 Resizing Images

The pixel values were normalized from 0-255. Since the datasets have images that vary in shape and size, all the images were resized according to the requirement of the different networks. The

library used for this was PIL which uses the nearest pixel value for assigning values to a local region. Eg: InceptionResnetV2 requires images in shape 150×150×3.

3.2.2 Image Augmentation

Since the data contains only 917 images, it is a very small dataset for training a Deep Neural network. Therefore we need to perform image augmentation. This technique does not manipulate the images, instead shows a different perspective of the image to the model. Cells being rotationally invariant, adding rotations to images would not alter the characteristics of the cells.

The following augmentations were done :

- Rescaled the pixel values from 0 to 255.
- Rotations with 40°
- Width and height shift with .2 value
- Sheared with a value of .2
- Zooming with value of .2

However, no augmentations were applied on the testing data apart from normalizing the pixel values.

3.3 Train-Test Splitting

The data was fractionated in the ratio of 4:1 for training and validation using the `test_train_split` function from the scikit library.

3.4 Regularisation

Regularisation technique helps reduce overfitting or reduce the variance in our network by penalizing the complexity. The idea is that certain complexity in our model may make our model unlikely to generalize well, even though its training data. So by adding regularisation we are trading the ability of the model to fit the training data well to the ability of the model to generalize better on the data it hasn't seen before (testing data). To implement regularization we add a term to the loss function that penalizes for large weights.

The most common technique is L2 regularisation, which has been used for the experiment, which is also commonly known as Ridge Regression or Tikhonov regularisation.

$$\text{Costfunction} = \text{Loss} + X$$

$$X = \left(\sum_{j=1}^n \|w^{[j]}\|^2 \right) \frac{\beta}{2m}$$

m = Number of inputs, n = Number of layer

β = Regularisation parameter, $w^{[j]}$ = Weight matrix of j th layer

If we set β to a relatively large number, then it would signify to the model to choose the weights close to zero because the objective of the optimizer is to minimize the overall loss.

Intuitively, this technique could set the weight so close to zero that it might completely diminish the impact of some of the layers, conceptually simplifying the model which may, in turn, reduce variance and overfitting

3.5 Programming Frameworks

3.5.1 GPU

Deploying deep neural networks like INCEPTION, ResNet etc requires high computational power. GPU cluster of IISER Pune was obtained for carrying out computational tasks. IISER Pune has various high-performance computing clusters ranging from 6-TeraFLOPs to 80-TeraFLOPs based on core Infiniband low latency network. The GPU assigned to our project was Tesla V100 card-500 core with 16GB CPU memory powered by VOLTA architecture.

3.5.2 Deep Learning Framework

Google's TensorFlow was the major framework used during the study. Tensorflow is an end to end open source platform for constructing machine learning models. It provides a Python based front-end interface. Keras API in TensorFlow was used to design neural network architectures.

3.6 Libraries

A few libraries and packages that were used in the

- Numpy: Numpy provides mathematical function to operate on multi-dimensional tensors.
- Matplotlib: Matplotlib is a plotting library to produce figures in a variety of hardcopy formats and interactive environments across platforms.
- OS: This module gives a versatile method for utilizing system dependent functionality.
- OpenCV: OpenCV, developed by Intel is a real time computer vision library for manipulating and visualising images/videos.
- Scikit : Scikit-learn is an open source Python library that comp a range of machine learning, pre-processing, cross-validation and visualization algorithms using a unified interface.

4 Methods

4.1 Multi-Layer Perceptrons

MLPs are one of the fundamental building blocks for Neural networks that were used as a first approach towards solving the classification problem.

Although there is no theoretical approach for determining the number of hidden layers, we constructed an architecture [schematic shown in Fig. 4.1] with seven fully connected layers. The initial layer contains 2250 nodes followed by 5 hidden layers containing 1225,750,356 and 175 nodes. The output from each node is known as its activation, which was computed using ReLU [refer 2.4]. We also added batch normalization to account for a change in the distribution of the weights in each layer. Adding batch norm affected the learning rate significantly. The number of nodes in the last layer is 2 for binary classification and 7 for seven-way classification. Softmax classifier was trained to compute the probabilities of predicted classes. The learning rate was chosen to be 2×10^{-6} after doing a grid search over multiple learning rates. The loss function was chosen to be as sparse categorical cross-entropy. The model weights were updated after training every batch (of 40 images) with Stochastic Gradient Descent optimizer[2.6.1]with a learning rate of 0.05, with a momentum of 0.9.

The images were segregated for training and validation by using the `train_test_split` function of the scikit library. The ratio of training to the validation dataset was 4:1. The training data was augmented using an `ImageDataGenerator` module and the operations performed were rotations, zooming, width shift and scaling. The validation data was not augmented as it may alter the network's ability of the model to predict on unseen data.

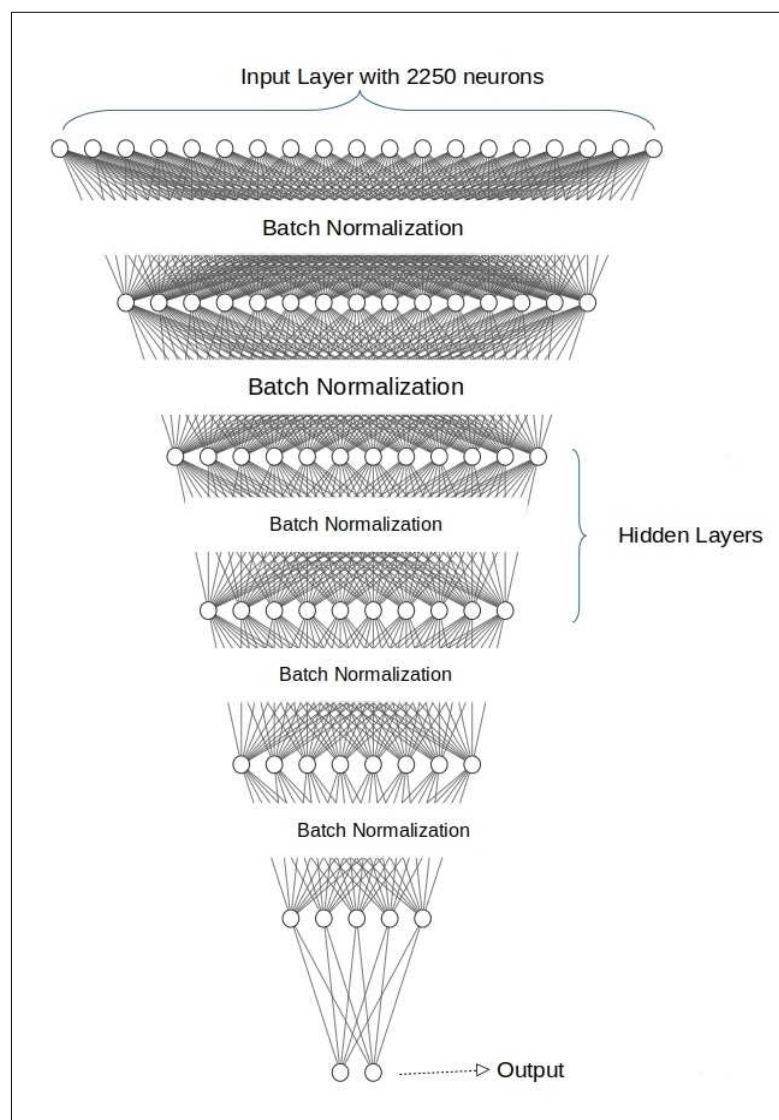


Fig. 4.1: Schematic representation of MLP constructed

4.2 Convolutional Neural Networks

We propose a convolution neural network[schematic shown in fig.4.2] with eight learning layers- five convolutional and three-fully connected. The first convolutional layer filters $70 \times 70 \times 3$ input image with 96 kernels of size $11 \times 11 \times 3$ with a stride 1. The second convolutional layer takes the output of the first convolutional layer and filters it with 256 kernels of size 5×5 .

The feature maps from each convolutional layer are downsampled by max-pooling of 2×2 , reducing the size of the feature map by half. We have also added batch normalization after every

convolution for standardizing the inputs in each layer. The third, fourth and fifth layer have 512,1024 and 1024 kernels of size 3×3 respectively. The outputs of which are fed into two fully connected layers of 3078 and 4096 neurons. The number of neurons in the last FC layer depended on the type of classification i.e 2 for binary and 7 for seven-way classification. The learning rate was chosen to be 2×10^{-6} after doing a grid search over multiple learning rates. The model weights were updated after training every batch of 40 images. The loss function was chosen to be sparse categorical cross-entropy which was optimized using SGD with a learning rate of 0.05 and a momentum of 0.9.

The images were segregated for training and validation by using the `train_test_split` function of the scikit library. The ratio of training to the validation dataset was 4:1. The training data was augmented using an `ImageDataGenerator` module and the operations performed were rotations, zooming, width shift and scaling. The validation data was not augmented as it may alter the network's ability of the model to classify on unseen data.

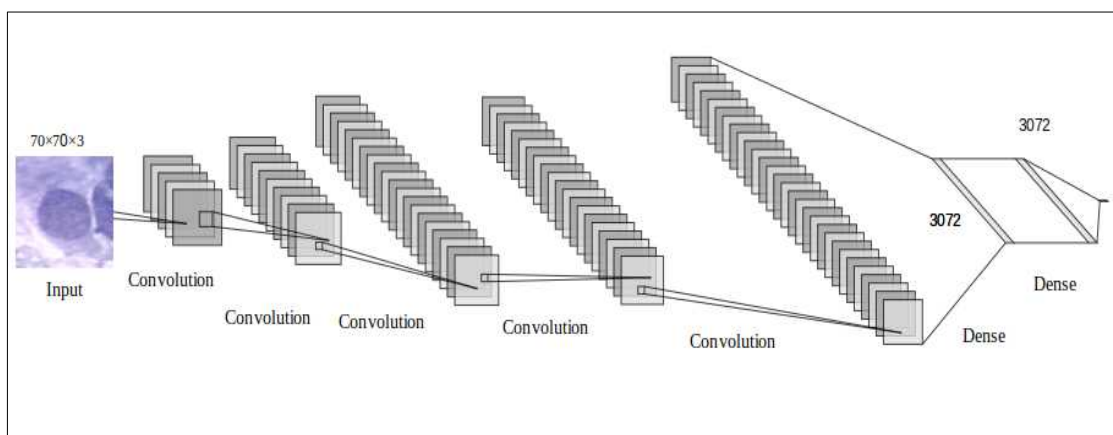


Fig. 4.2: Schematic representation of the Convolutional Neural Network

4.3 Transfer Learning

Transfer Learning (TL) is one of the most powerful ideas in Deep Learning, which says that sometimes we can take the knowledge that a neural network has learned from one task and apply that knowledge to a different task. This technique eliminates the need for a large training dataset and the costly computational time that is required to train a model from scratch. For example, If a network learned to recognize an object like cars then we could potentially use that knowledge to help us do a better job at recognizing trucks. Radiological Imaging is one such field where this

technique has been heavily extrapolated i.e a network trained to recognize everyday objects could be used to differentiate medical images. The process involves transferring the parameters from the original model trained on a particular dataset and fine-tuning them to create a new model for a different dataset. Training a deep classifier to identify objects and patterns with human-level accuracy could be an extremely challenging task. Training a deep network from scratch would require huge amounts of data. There are datasets available with millions of images to train, but it's extremely expensive and almost impossible to make datasets for specific problems, like pap-smears in this case. TL helps us tackle this issue in an extremely sophisticated manner by allowing us to fine-tune the pre-trained models for our specific problem.

We deploy multiple models on the binary dataset and subsequently choose the best working model for seven category classification, three category classification and in carcinoma and columnar cell detection. These models are available in Keras and can be imported using the functions. An example is shown below:

```
from tensorflow.keras.applications import VGG19
```

Fig. 4.3: Importing models in tensorflow

The input shape for different models vary, therefore the images had to be resized and pre-processed according to the different network. The top classification layers were removed, by keeping the *include_top* argument as "False" as shown below:

```
conv_base = VGG19(weights=None, include_top=False, input_shape=(150,150,3))
```

Fig. 4.4: Assigning the arguments for the model

The starting weights are always random, howbeit Keras provides an option to start training with the pre-trained weights. The pre-trained weights were obtained by training these models on ImageNet database, which contains 1.4 million images of 1000 different categories. These models can be used for prediction, feature extraction, and fine-tuning.

Note: For Transfer Learning, we first investigated the best working model, classifier and optimizer as discussed in sections 4.3.1,4.3.2 and 4.3.3 respectively. The best working model,classifier and optimizer were then used for two class classification (section 4.3.4), seven class classification (section 4.3.5), columnar detection (section 4.3.6) and carcinoma detection(section 4.3.7). The results of these experiments have been discussed in the same order within section 5.3.

4.3.1 Model Selection

The following models that were tested :

- Resnet152V2
- InceptionV3
- InceptionResnetV2
- DenseNet201
- VGG19

Please refer to section 2.7 brief description of the networks.

The hyperparameters were same for all the models. Sparse categorical crossentropy was used as the loss function. Softmax classifier with two output neurons was used along with three fully connected layers containing 1000, 500 and 50 neurons respectively. Pretrained weights on Imagenet database were used for initializing the weights. Since the models reached a maximum accuracy at different epochs, hence the number of training epochs vary for each one of them. The data had two categories normal and abnormal[refer 4.3.4], with train-test split of 4:1. Stochastic gradient descent with a momentum of 0.9 was used to optimize the model weights.

4.3.2 Classifier Selection

Grid search was done to find the suitable number of fully connected layers to make an appropriate model for classification. The classification layers were concatenated on top of our deep neural network as shown in Fig. 4.5. The training data consisted of seven categories as mentioned in section 4.3.5. The loss function was chosen to be Sparse categorical loss and was minimized using RMSprop with a learning rate of 2×10^{-5} . The training images were augmented using the ImageDataGenerator[section 3.2.2]. The training was done for 50 epochs.

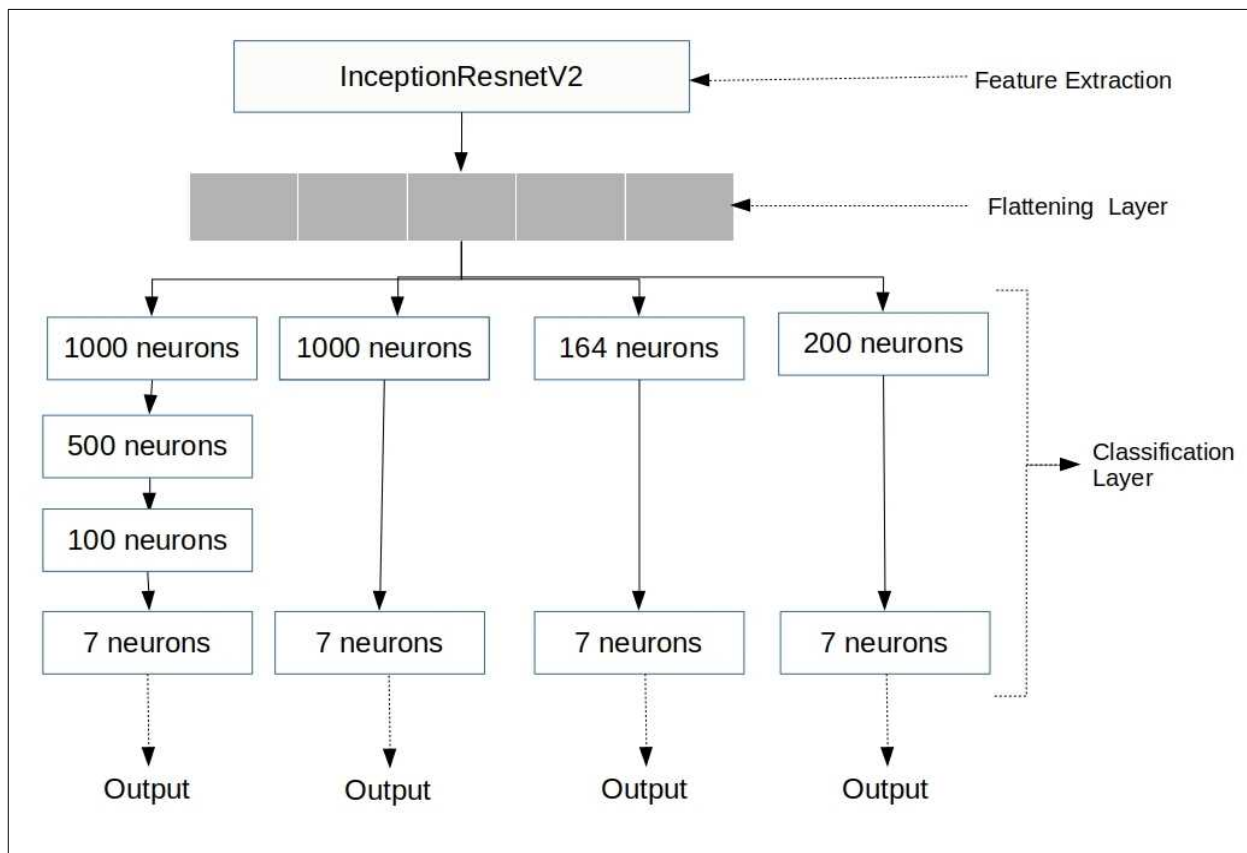


Fig. 4.5: Schematic representation of different classification architectures

4.3.3 Optimizer Selection

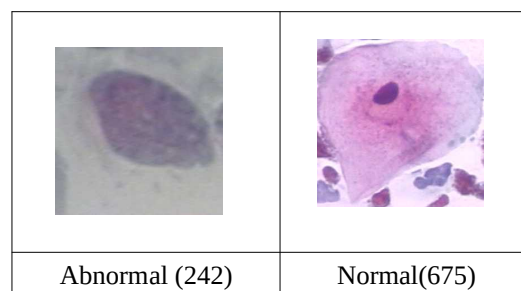
Since optimizers are one of the most important determinant of training a network, we did a comparative analysis of the optimizer in terms of achieved performance, memory usage and operational time [GPU details in section 3.5.1]. These optimizer were tested on InceptionResnetV2 with 4 fully connected layers. The FC layers consist of 1000, 500, 100 and 7 neurons, the first classifier shown in Fig. 4.5]. Softmax function was used to evaluate the output in terms of probabilities. The three optimizers tested were:

- RMSprop (Root mean square propagation)
- ADAM (Adaptive Momentum)
- Stochastic Gradient Descent (SGD)

Please refer to section 2.6 for a brief description of the optimizers.

4.3.4 Binary Classification


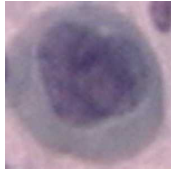


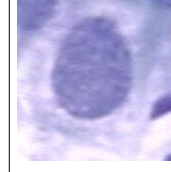
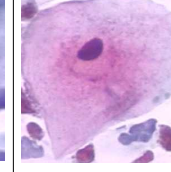
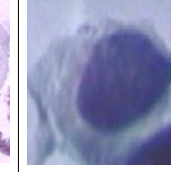
We merged all the cancerous and non-cancerous categories into normal and abnormal classes, making this a two-class problem. The total number of abnormal cells and normal cell images are 675 and 242 respectively. A visual representation of the images along with the numbers in each category are shown below.



The first aim of this classification is to obtain maximize the true positives and true negatives, but a bigger problem that needs to be worked on is to reduce the number of false negatives i.e. the abnormal cells being classified as normal. Since the highest number of false-positive i.e. non-cancerous cells being predicted as cancerous is 242, the worst overall error possible is 26.39% considering al the abnormal ones were classified correctly as True Negatives with no cases of False Negatives and False positives.


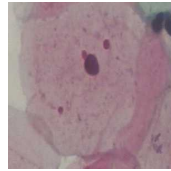
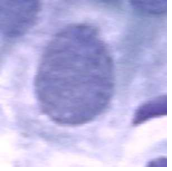
4.3.5 Seven class classification

Herlev dataset has seven classes. We split the data into training and validation as 80% and 20% respectively. The validation accuracy metric we are using gives us an estimate of the correctly detected True Positive and False Negative cases. However, it is crucial to look at the cases of false negative and false positives. Doing a seven-way classification could distinctly show the categories with the highest misclassification rate, which could be further dealt with by creating models that are better at classifying those particular categories. Building models around the bottleneck category(the highest misclassified category) could specifically help us improve the overall model. A visual representation of images along with the numbers in each category are shown below.

						
Carcinoma in Situ (150)	Light Dysplastic (182)	Moderate Dysplastic (146)	Normal Superficial (74)	Normal Columnar (98)	Normal Intermediate (70)	Severe Dysplastic (197)



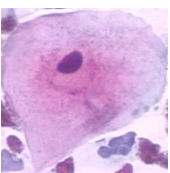
4.3.6 Columnar detection

The following trials are a comparative analysis for training with different hyperparameters to classify Normal, Abnormal and Columnar cells. The normal columnar was taken as a separate category for classification as the number of false positives is the highest for this particular category [Lin et al. 2019]. If we can achieve a high accuracy in classifying columnar against the other two classes, we could use this extrapolate this model in the secondary stage of diagnosis. A visual representation of images along with the numbers in each category are shown below.

		
Abnormal (675)	Normal (144)	Normal Columnar (98)

4.3.7 Carcinoma detection

The trial was to segregate carcinoma images from other categories. Therefore we merged light dysplastic and moderate dysplastic into abnormal category with 525 images. Similarly, the normal superficial, normal columnar and normal intermediate were merged into a normal category with 242 images. The carcinoma in situ was kept in the third category called carcinoma with 150 images. A visual representation of images along with the numbers in each category are shown below.

		
Abnormal (525)	Carcinoma (150)	Normal (242)

5 Results and Discussion

5.1 Multi-Layer Perceptrons

We tested the MLP for two category and seven category classification.

5.1.1 Binary Classification

A mean classification accuracy of 73.15% was achieved in the last 5 iterations. Taking a look at the training accuracy curve in Fig.5.1, it is noticeable that the validation accuracy is fluctuating between 60 to 70% and the system doesn't appear to gain significant information even after 25 epochs.

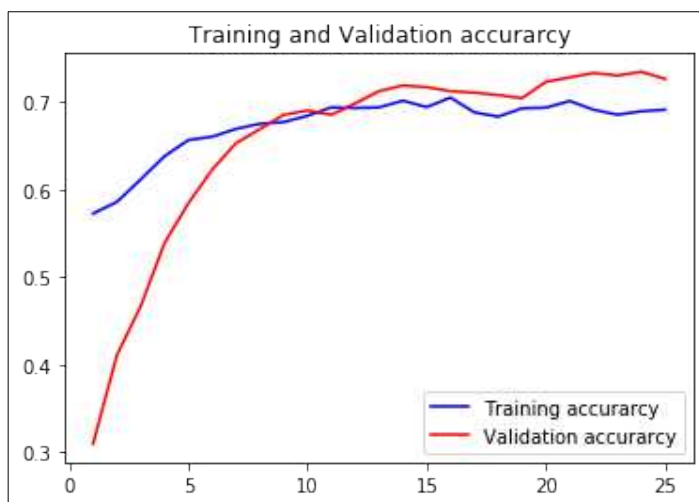


Fig. 5.1: Accuracy curve(y-axis) wrt epochs(x-axis) :Two class

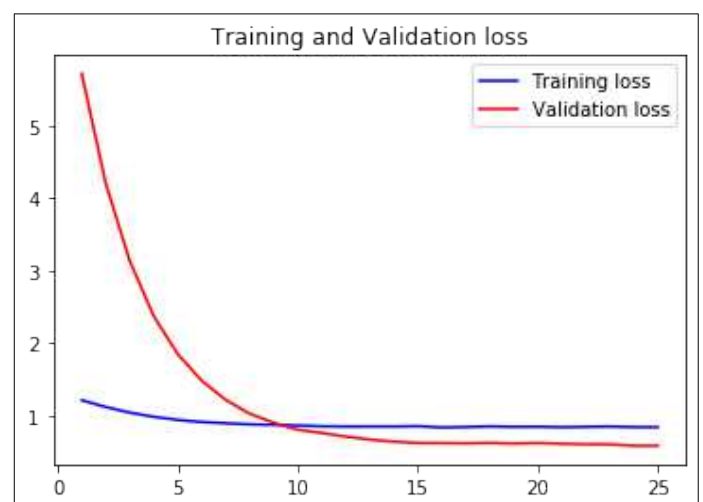


Fig. 5.2: Loss function (y-axis) wrt epochs(x-axis):Two class

The training begins from 55%, suggesting that both the classes have an equivalent probability(which appears legitimate) and increases up to 70%. The training accuracy of 70% in the 15th epoch can be accredited to the imbalanced distribution of the images, with 520 training images belonging to the abnormal class and 265 training images belonging to the normal class. We The class imbalance was normalized by giving the class weights as 1:2 for abnormal and normal categories respectively, however, that did not affect the model performance significantly.

The loss curve shows a decent declining trend which can be credited to smoothly working Stochastic Gradient Optimizer. The loss saturates achieves a minima implying that the model parameters were not updated after 10 epochs, which is analogous to the accuracy curve showing similar validation accuracy.

5.1.2 Seven class classification

The same multi-layer perceptron model was used for seven-way classification, achieving an accuracy of 26.30%. The model learns to classify the training dataset of 20% in the first iteration, which is slightly higher than the probability of selecting an image from any given class(14.28%). For a variation in images in every batch, the dataset is reshuffled for 25 iterations. But, there isn't much progress in learning as the training curve plateaus after training for 10 epochs.

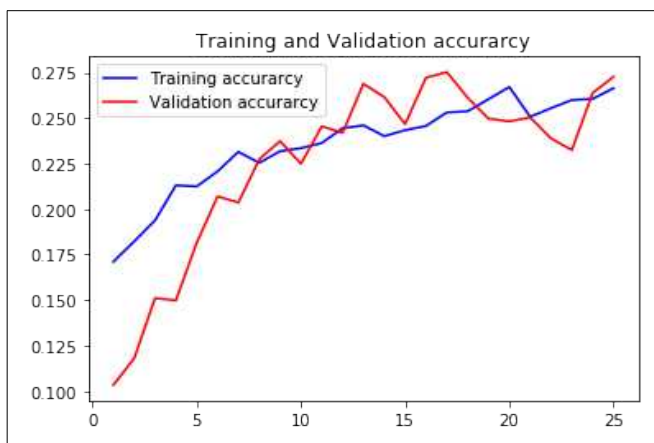


Fig. 5.4: Accuracy curve(y-axis) wrt epochs(x-axis)

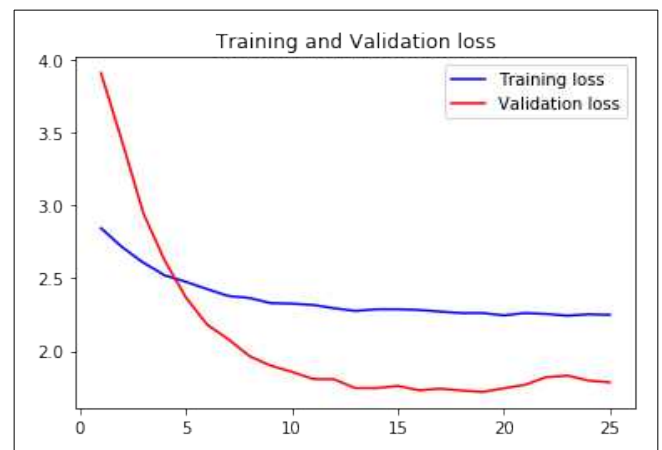


Fig. 5.3: Loss curve(y-axis) wrt epochs(x-axis)

However, it's quite intriguing to notice that the network seems to perform better on unseen images with only 5 iterations on the training data. Since the loss does not overshoot even after unnecessary training (after 15th epoch) implies that the model was not overtrained and this was the best fit this model is equipped of performing.

5.1.3 Discussion

After trying multiple multi-layer perceptrons, the model presented here showed the highest performance, with the highest validation accuracy of 70% for binary classification and 35% for seven-way classification. Although these networks look heavily undertrained due to multiple

reasons. One of the reasons for undertraining could be credited to the under-complexity of the model, however, more complex models with a higher number of layers were tested but no significant improvement was observed.

Four dysplastic cell types into the abnormal category and three normal cell types in the normal category, creating a huge variation within the categories making it extremely difficult for the network to make a generalization. Using proper segmentation of nucleus and cytoplasm could potentially allow to find better relation between cells and category type, however, that wouldn't ensure a higher testing accuracy on unsegmented testing dataset.

Since the number of categories was increased for seven class classification, the model is will have to make more explicit predictions and therefore an even more complex model would be required. It's quite obvious that a higher number of parameters would be required for creating a model that can make predictions for seven different cell types and this under-complexity of the model is the most parsimonious interpretation for its underperformance. Another rationale why the model underperformed for seven-way classification was due to the insufficient data in each category. Also, with only 70-100 images in each category, the network heavily suffers from data insufficiency which might responsible for undertraining different categories essential for a perceptron. Since we are limited by the amount of data, we introduced more sophisticated feature extraction techniques for classification.

5.2 Convolutional Neural Networks

5.2.1 Binary Classification

The loss is very close zero after the 20th epoch implying that the most suitable weights were obtained. However, this model could classify the validation images with an accuracy of 80.65%. Due to heavy inconstancies in a curve, we calculated a mean value of accuracy from the last 5 epochs. Comparing these results from the previous approach with 70% accuracy, introducing convolutions did show a modest improvement in learning. Merging the seven classes into two classes led to a huge variation within each class, therefore, features common in the subclasses had to be determined. Most of the common features were learned in the initial training phase which can be seen in Fig.5.6. The learning curve then saturates implying that more training did not help in finding more representable features.

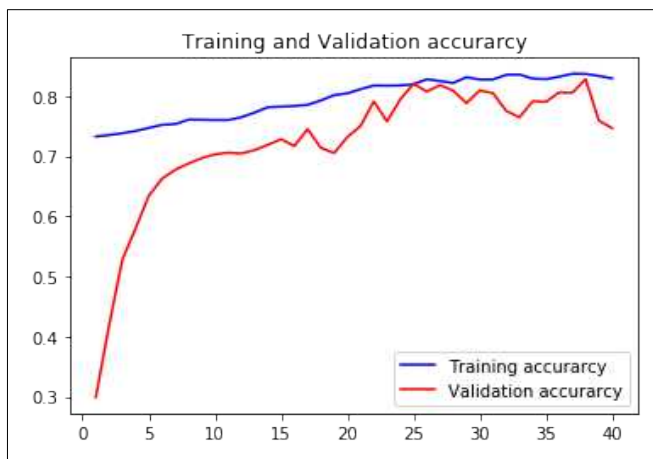


Fig. 5.6: Accuracy(y-axis) wrt epochs(x-axis)

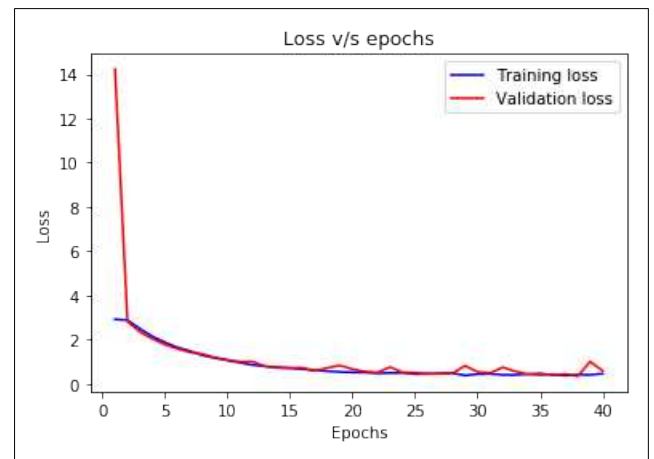


Fig. 5.5: Loss(y-axis) wrt epochs(x-axis)

(The curves shown in Fig.5.6 and Fig.5.5 are the smoothed version of the original graphs)

5.2.2 Seven class classification

The same convolutional neural network was implemented for solving the seven class problem, achieving a validation accuracy of 42%. There was a decent improvement in classification results, however, this is not sufficient. As seen in the training curve, the accuracy seems to increase, however, we stopped the training(early stopping) as soon the loss appeared to reach a minimum value.

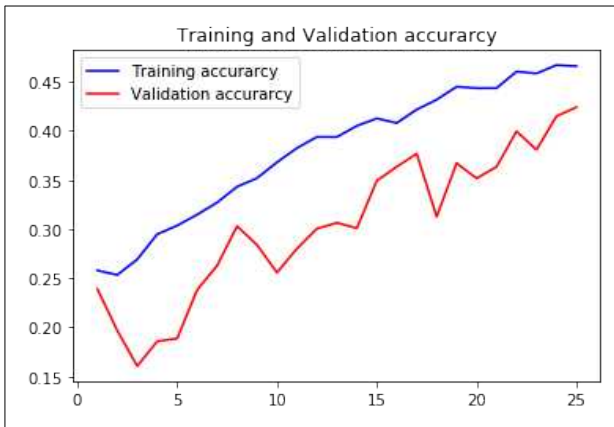


Fig. 5.7: Accuracy (y-axis) wrt epochs(x-axis)

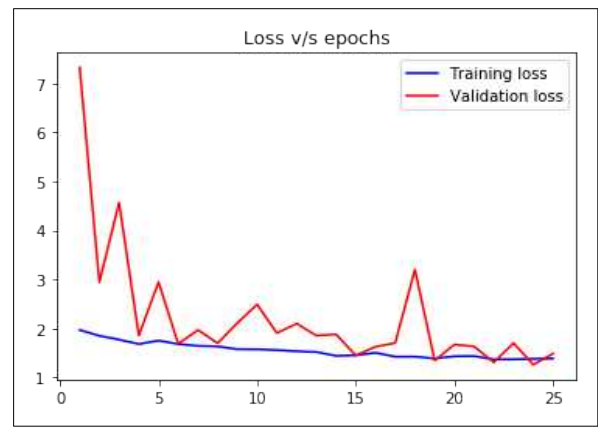


Fig. 5.8: Loss (y-axis) wrt epochs(x-axis)

5.2.3 Discussion

The fluctuations in the training curve are coarsely attributed to the variation within the categories. Different types of conv filters are used to extract different types of morphological features, hence the network aims to find those general features which are common among all the images present.

Since we merged the seven different categories (with different morphological features) in normal and abnormal, it appended one more level variation within each category making generalization even harder.

With Convolutional Neural Network the error rate was reduced to 16.44% (validation accuracy is 80.65%) for binary classification and there is still scope improving the results with improving the model architecture. Inculcating convolutions in the model helped to improve the seven category classification results only by a small amount, with the highest validation accuracy of 42%.

The loss in both cases converged to a minimum value, implying that the most suitable model weights were found and there is no or very little scope of improvement in them. And to make any better predictions we need to make deeper networks. However, with increasing depth we are doomed to encounter the well known **Feature Degradation problem**. Having large training data is another crucial need for training a deep network from scratch.

With these limitations, making more deeper networks wouldn't improve the performance, therefore, we then used a slightly different approach of training deep models called Transfer Learning.

5.3 Transfer Learning

This technique eliminates the need for a large dataset and can show comparable performance even with lesser training data. It is also capable of reducing the computational time that is required to train a model from scratch. TL has been briefly discussed in section 2.7.

We investigated the best working model, classifier and optimizer as discussed in sections 4.3.1, 4.3.2 and 4.3.3 respectively. The best working model, classifier and optimizer were then used for two-class classification (section 4.3.4), seven-class classification (section 4.3.5), columnar detection (section 4.3.6) and carcinoma detection (section 4.3.7). The results of these experiments have been discussed in the same order within section 5.3.

5.3.1 Model Selection

Since fine-tuning models with transfer learning is a computationally expensive process, a comparative study of different models was done to find out the most suitable option.

Model name	Network Depth	Model Accuracy	Number of training iterations	Training time per epoch (approx.)	Model Size	Total Parameters
Resnet 152V2	152 layers	87.88%	45 epochs	360 seconds	496 MB	60,380,648
InceptionV3	48 layers	93.3%	45 epochs	37 seconds	327 MB	40,761,436
InceptionResnetV2	164 layers	96.25%	40 epochs	150 seconds	552 MB	68,687,388
DenseNet201	201 layers	90.13%	30 epochs	500 seconds	175 MB	21,769,636
VGG19	16 layers	71.88%	45 epochs	200 seconds	230 MB	28,743,036

Table 3: Comparative Analysis of different models implemented for binary classification

It is usually expected that the performance of any machine learning model would increase as the complexity increases, however, our trials do not show the same. DenseNet and InceptionV3 have less number of parameters than Resnet152V2 and still perform marginally better.

InceptionResnetV2 is the most sophisticated model with 68 million parameters. However, the improved performance is not mainly because of increased depth. The architecture contains stacked Inception blocks along with ‘residual connections’ making it the most efficient model. InceptionV3 worked on the improved idea of factorizing convolutions and batch normalization showing the second-best performance among the tested architectures.

It is quite intriguing to notice that InceptionV3 might be the most suitable model when we have less computational power. The model contains 48 layers, taking only 37 seconds for each iteration and is exceptionally faster than other deep models. With 40 million parameters, the model size is just only 300 MB. The network didn't show the highest performance for Herlev Dataset, but it can be assured that using InceptionV3 could save time, computational power without compromising much on the final results. VGG19 did not show any learning as the validation accuracy remained constant throughout the training process.

InceptionResnetV2 outperformed all the other networks achieving an accuracy on 96.25% and was therefore used as a standard model for classification tasks.

5.3.2 Classifier Selection

A comparative study to make a suitable classifier was done by modifying the number of fully connected layers(FC layers) and the number of neurons as shown in Fig. 4.5. On comparing the various classifiers, we found that changing the fully connected layers did impact the overall performance.

Model Name	Classification Layers	Total Parameters	Validation Accuracy
5A	4 FC layers with 1000,500,50,7 neurons	68,717,743	66.87%
5B	2 FC layers with 1000 and 7 neurons	57,103,143	61.12%
5C	2 FC layers with 164 and 7 neurons	56,605,191	60.16%
5D	2 FC layers with 200 and 7 neurons	57,103,143	65.62%

Table 4: Comparative analysis of different classifiers on InceptionResnetV2

Comparing the model 5C and 5D, it's hard to reason out a change of 5% accuracy with only increasing 34 neurons. However, on comparing 5B and 5C we don't find any significant change in performance. Therefore, we can say that increasing the number of neurons may improve the network but we cannot conclude that there is a derivable relation between the two. On comparing 5A with other models, we can say increasing the number of FC layer does impacts the complexity of the model significantly and hence fitting the training data in a better fashion. Model 5A does show the highest accuracy but we do not find a relation between the number of FC layers and

model performance. Therefore, with these sets of experiments we cannot conclude the impact of increasing FC layers on model performance. However, 4 FC layers with a softmax classifier worked adequately for capturing the information. This set of FC layers was then standardized and were used for all the subsequent experiments.

5.3.3 Optimizer Selection

These trials were done to find an efficient optimizer for seven category classification and the training methods have been described in section 4.3.3. The comparative analysis was done in terms of model performance, memory usage and operational time.

Optimizer	Learning Rate	Validation Accuracy	Total training time for 60 epochs (approx.)	Training time per epoch (approx.)	Model Size
ADAM	2×10^{-5}	63.64 %	3 hours	180 seconds	827 MB
RMSprop	2×10^{-5}	66.87 %	2 hours	120 seconds	552 MB
SGD with momentum	.01	67 %	2 hours	119 seconds	550 MB

Table 5: Performance comparison of various optimizers

ADAM optimizer took the highest amount of time given the more amount of computation that it performs. It takes two moments of the gradient to update the learning rate, making the process slower. RMSprop on the other hand only depends on the decaying average of the square of gradients and hence requires lesser computational time than ADAM. We find that RMSprop and SGD achieve a similar validation accuracy on training for 60 epochs. The validation loss asymptotically decreases with SGD, signifying that the best fitting model weights were obtained within 10 epochs of training. RMSprop also uses a variable learning rate. It minimizes the learning rate with a factor of the square of the previous gradients. Surprisingly, it took the same amount of time to train while performing equally well as SGD. This may be because it uses faster learning rates initially and then decreases the rate of descent as it reaches the local optima balancing the overall time taken.

The loss function of the two optimizers has been shown in Fig. 5.9 and 5.10 below. SGD works seemingly well on the dataset as the loss decreases rapidly in the first few epochs and saturates

after 10 epochs. The loss does not increase as the further training increases implying that it does not modify the model weight much and hence there is no overfitting.

On the contrary, the loss with RMSprop decreases initially but explodes after the 30th epoch, showing that the model has started overfitting which is commonly referred as bias-variance tradeoff. The reason why SGD performs better may lie in a steady learning rate, due to which it is not able to overcome the local minima.

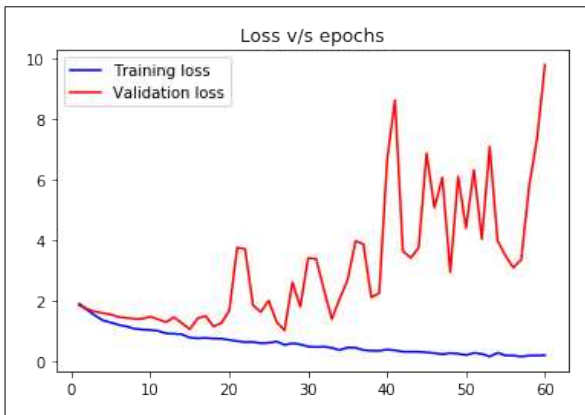


Fig. 5.9: Loss variation(y-axis) wrt epochs(x-axis) with RMSprop

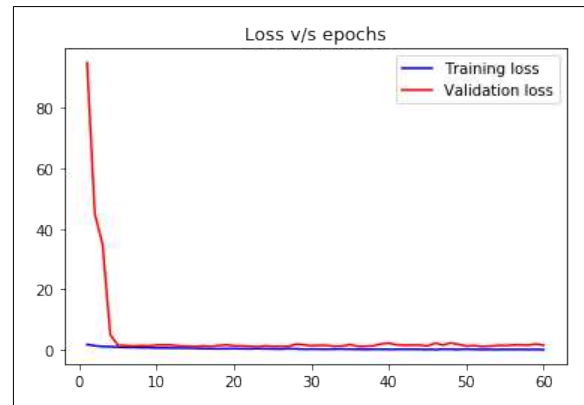


Fig. 5.10: Loss variation(y-axis) wrt epochs(x-axis) with Stochastic Gradient Descent

5.3.4 Binary Classification

We successfully classified the images into normal and abnormal with the highest testing accuracy of **96.25%**. This was achieved by fine-tuning InceptionResnetV2 (IRV2) with 4 fully connected layers [model 5A described in section 4.3.4]. The slope for the validation curve is highest in the initial epochs, showing the highest amount of learning by the network and flattens after a few epochs of training.

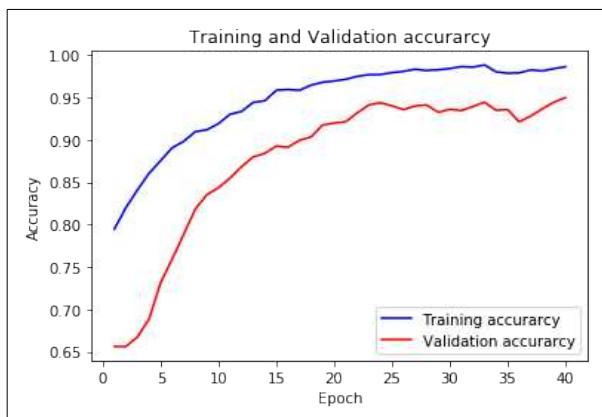


Fig. 5.11: Accuracy(y-axis) wrt epochs(x-axis) for two class classification

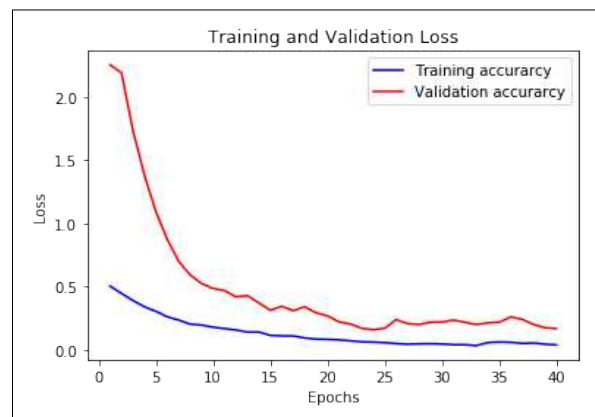


Fig. 5.12: Loss variation(y-axis) wrt epochs(x-axis) for two class classification

The loss as shown in fig.5.12 decreases exponentially in the initial epochs and saturates to a minimum after the 20th epoch. The accuracy and loss curve comply with each other. Since the loss of validation data does not increase after the 20th epoch, it can be inferred that this is a good working model.

5.3.5 Seven class classification

We found that InceptionV3 and Resnet showed comparable performance as InceptionResnetV2 for binary classifications[section 5.3.1]. Therefore, an experiment was conducted to compare the performance of the three networks on seven class classification. It was found that apart from InceptionResnetV2, other networks performed extremely poorly as shown in the table given below. RMSprop optimizer was used for all the three models. The training curve of InceptionResnetV2 for seven classification is shown in fig 5.11.

Model	Validation Accuracy
InceptionV3	20.31%
ResNet50	14.84%
InceptionResnetV2	66.87%

Table 6: Comparison of models for seven category classification

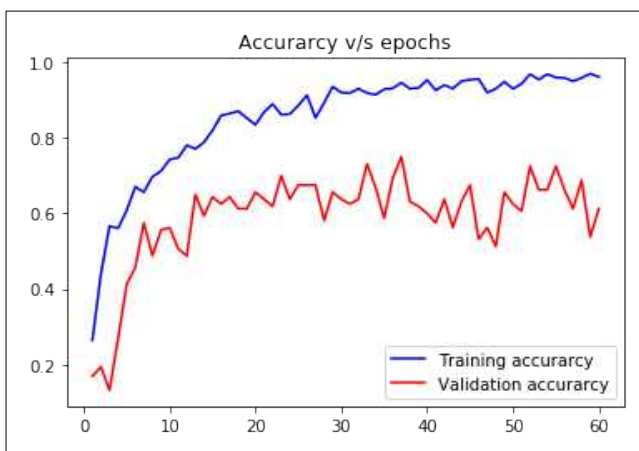


Fig. 5.14: Accuracy(y-axis) wrt epochs(x-axis) for seven category classification with InceptionResnetV2

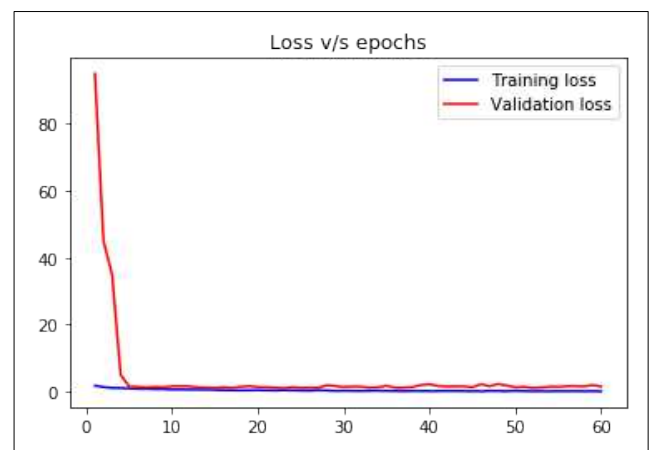


Fig. 5.13: Loss variation(y-axis) wrt epochs(x-axis) for seven category classification with InceptionResnetV2

Since the validation accuracy was fluctuating continuously, a mean value of accuracy from the last 10 epochs was taken. The training curve shows a decent trend as the accuracy increases in the first few iterations and saturates eventually. The loss instantaneously decreased within the first few epoch and remained closed, implying that the model parameters were not updated. Since the loss remains close to 0, the model did not overfit the training data.

5.3.6 Columnar detection

Classifying normal columnar cells was crucial as it occupied the highest number of False-positive cases [Lin et al. 2019] among all the other categories. To finally segregate the normal cells, we designed a model that is equipped to classify the columnar cells efficiently. This was a three-class problem, classifying normal, abnormal and columnar categories. We were successful in achieving the highest validation accuracy of 94.38% obtained by fine-tuning InceptionResnetV2.

The nuclear size of columnar cells is comparatively larger than the other normal cells and appears significantly similar to the dysplastic type, causing a disparity in classification, therefore we need to achieve a good accuracy in classifying this bottleneck category. Another major factor impacting the high number of FP could be the low number of images in making the feature selection harder for the category. Overall the accuracy was observed to increase with training.

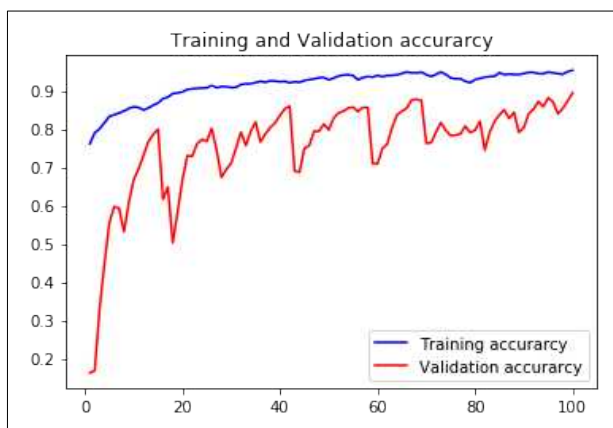


Fig. 5.15: Accuracy(y-axis) wrt epochs(x-axis) for detecting columnar class

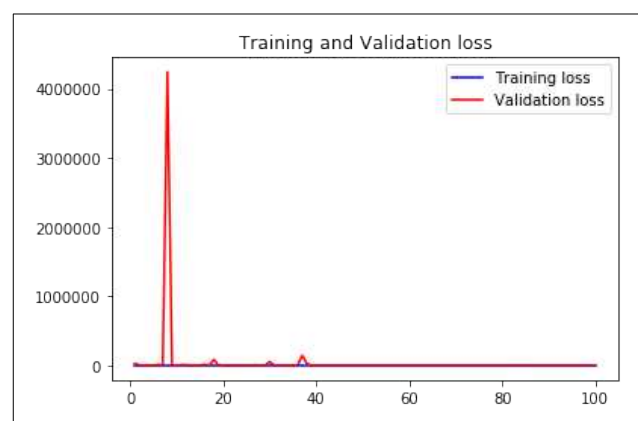


Fig. 5.16: Loss(y-axis) wrt epochs(x-axis) for detecting columnar class

As show, the training started from 75% and reached 100% after a few epochs. The validation accuracy curve shows a few declines, nevertheless, it increases as the training progresses. The loss curves [fig.5.16]show an anomalous behavior at 10th iteration. This kind of trend may be due to the exploding gradient, where the change in weights(which depends on the gradient) is inflated and the completely deviate from the ideal model weights, leading to a sharp increase in loss. However,

the network does not get stuck with that loss value and optimizes the parameters to fall back to the local minima. The loss looks nil for the rest of the training, solely due to the scaling error. The parameters from this model can be used as weight initialization for other models for columnar classification.

5.3.7 Carcinoma detection

Since carcinomas are contrastingly very different from normal cells and given that we were able to classify binary classes with a classification accuracy of 96%, we expected the network to have comparable performance. However, carcinomas were detected with the highest accuracy of **88%** against normal and abnormal classes. The validation curve shows a small drop at the 15th epoch, however, it continues to increase along with training and then eventually saturates achieving the highest value of 88%. We expect to improve this model by altering the classification layers.

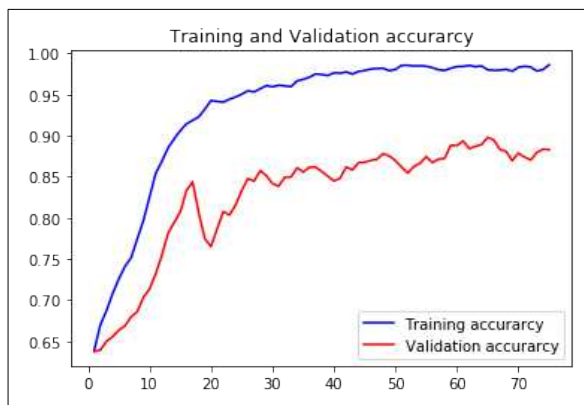


Fig. 5.18: Accuracy variation(y-axis) wrt epochs(x-axis) for carcinoma detection

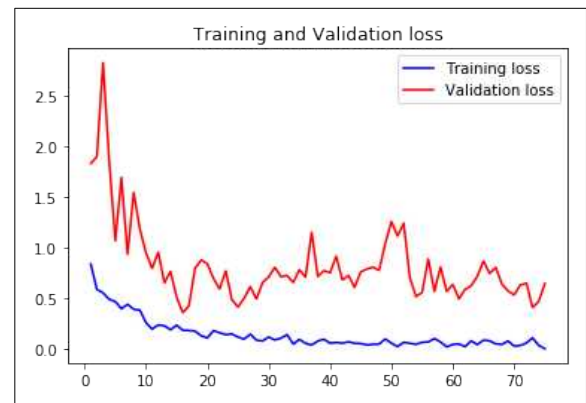


Fig. 5.17: Loss variation(y-axis) wrt epochs(x-axis) carcinoma detection

5.4 Summary of results

Classification type	MLP	CNN	Transfer Learning
Two class	73.5%	80.65%	96.25%
Seven class	26.3%	42%	66.87%
Columnar detection	-	-	94.38%
Carcinoma detection	-	-	88%

Table 7: Summary of results

We started with the approach to build a classifier using Multi-Layer perceptrons which is of the most traditional form of neural network and attained an accuracy of 73.5% for binary classification. With 675 images in abnormal category out of 917 images in total, the probability of getting an abnormal cell becomes 73% by default, implicating that the network did not learn any significant feature. We then improved the model by adding the Convolutional layer for feature extraction. Deep Convolutional Networks are very data-intensive and require high-quality data for extracting quantifiable features. Since our data is extremely sparse, validating the model turned out to be a very challenging task. However, after applying proper image augmentation techniques, our model learned to classify with 80.65% accuracy demonstrating a 10% improvement. With multiple trials of hyper-parameter tuning we did not find any noteworthy improvement. We then tried to replicate making deeper architectures that have won international competitions(ILSVRC) and retrained them with a new approach called Transfer Learning. This technique does not require training models from scratch, hence performed seemingly well with 917 images. We did multiple trials for finding the best working model, optimizers and classifiers. InceptionResnetV2 model coupled with RMSprop optimizers achieved 96.25% accuracy for two classes and 66.87% accuracy for seven classes. This model was then used for detecting carcinomas and columnar classes. We also observed that the standard Stochastic Gradient with momentum worked equally well and surprisingly converged faster than other optimizers taking less computational time.

5.5 Discussion

100% accuracy in medical imaging is highly uncommon, however, it does occur in extremely standardized classification tasks with extremely well-labeled data and low overlap in categories.

Achieving a 100% accuracy on real samples is practically impossible, and is mostly seen in cases in which training and test data are much alike, contrary to what is observed in real scenarios.

A perfect classification accuracy may look neat, even when assuming that the testing samples are a representative sample of the population, however, the model is deemed to encounter many new samples on which it will probably not be able to classify with 100% accuracy.

For real applications, these computer-aided diagnostic tools often include intervention from human employs in many domains. Especially in the medical image diagnosis where the rates of False-positive cases are very high, the models are only expected to generalize well to a certain degree, alleviating the monotonous task. The final verification of the False-positive is done by the medical practitioner. However, the case of False Negative(FN) is critical as the abnormal cells are detected as normal, and may get overlooked even with different stages of analysis. The models are aimed to minimize FN cases.

To achieve a high precision, a bigger dataset is required. And there is no particular formulae to ascertain the data size, but the rule of thumb says we might need 10 instances per predictor, where the predictors are the principal parameters of the model. For computer vision models, using the 1000 images per category may suffice. Although, there is no empirical evidence, this number originates from international challenges like ILSVRC which had less than 1000 images in each category, and the winning models performed reasonably well.

Sample complexity, a term commonly used for referring to the number of training samples that are needed by the algorithm, so that the function created by the algorithm is withing an arbitrarily small error and a probability close to 1. Practically, the sample size for training depends on the nature of the problem and the kind of machine learning architecture implemented. Deep Learning as compared to other machine learning is known to consume more data for learning. The Free lunch theorem proposed by Wolpert et al. 1997, in general says that the sample complexity is infinite i.e. no algorithm can learn globally-optimal target function(a predictor with 100% test-accuracy) using a finite number of training samples.

However, using pre-trained models for classification helps to avoid training from scratch and the need for a huge dataset, the approach used in this thesis. Limitation of data can also be coarsely dealt with image augmentations (rotations, zoom etc.).

Variation within the data is also an essential component as it helps the model to generalize well. Also, training on a wide spectrum of features increases robustness over unseen data. It was observed that the algorithm classified the columnar cells as severe dysplastic in many instances. This can partially be credited to the similar morphological features to the abnormal cells. Jantzen et al. 2005 calculated the ratio of Nucleus to Cytoplasm(N/C) area and found that columnar cells had comparable N/C ratio as severe and moderate dysplastic cells. N/C ratio can be considered as it has been used an essential parameter in studies which use handcrafted features for classification. The confusion matrix presented in Lin et al. 2019 [Appendix III] clearly shows that 23% of the columnar cells were classified as severe dysplasia. Moreover, the columnar cells have a dark nucleus and less area of cytoplasm which visually appears to be an abnormality, and expert cytotechnicians play a key role in detecting these cases.

All the classes were not classified with similar error rates. The ease of classification depends on the number of training images for the specific category. However, a clear distinction of nucleus and cytoplasm showed to be of more importance. Superficial and intermediate epithelial cells which had clearly distinguishable nucleus and cytoplasm were classified with highest accuracy. Moderate dysplastic cells on the other hand were classified with the least accuracy. They were categorized as light dysplasia(or mild dysplasia) at multiple instances and there were also occurrence where they were categorized as servere dysplasia. This trend could be attributed to the similar nuclear sizes of the three abnormal classes and visually similar cells.

Since detecting the various stages of lesions is extremely essential, we explicitly conducted trials to classify carcinoma-in-situ against other normal and abnormal cells. These trials were named as 'carcinoma detection' and successfully classified with an accuracy of 88%. Similarly, we did trials to determine our model's capability of classifying columnar cells, and achieved a 94.38% accuracy. The aim of doing the last two trials was to prepare models that are specific to detecting these bottleneck categories. Multiple models can be merged together to design an ensemble meta-algorithm that would improve the stability and accuracy of the system. This is a well-known approach in machine learning known as Bootstrap aggregating (Bagging) and is a future for making this system even more robust.

6 Conclusion

With the achieved classification accuracies, the results are not magnificent. But the idea of eliminating all forms of computationally expensive feature selection and pre-processing is of central importance. Multilayer perceptrons were the first approach towards solving the problem, and an accuracy of 73.15% for binary and 26.30% for seven-way classification was achieved. After failing to improve the performance even after adding more complexity and hyperparameter tuning, we further introduced convolutions to our fully connected model. Using Convolutional Neural Network, we gradually reduced the error rates, escalating up to an accuracy of 80.65% for binary and 42% for seven-way classification. After multiple trials it was observed that due to the poor quality of the data there is no feature generalization and hence it is extremely hard to train a deep network from scratch. Another major problem was that Herlev Dataset contains only 917 images with seven different categories, hence making generalization an extremely challenging task. The above two approaches (MLP and CNN) turned out inadequate, motivating us to employ other supervised learning techniques. We then used deeper pre-trained models and re-trained them on our dataset using Transfer Learning. Since Transfer Learning does not require training models from scratch, the necessity of having a large dataset is diminished. This methodology worked impeccably with the given constraints of the dataset. The best performing model, InceptionResnetV2 achieved an accuracy of 96.25% on two-class classification and 66.87% for seven-class classification. Our results are not the best but are comparable to the published studies which make use of Deep Learning. Zhang et al. 2018 achieved a 98.6% accuracy on the binary problem using a similar transfer learning based approach.

Unlike previous strategies, we have successfully eliminated the employment of handcrafted features as our methodology automatically extracts deep features embedded in the cell image. We may have not achieved the best performance, however, our model parameters can be used for initializing training of other sophisticated models on cervical cancer datasets. We hope to improve the prediction performance by concatenating other existing classifiers (eg. SVM's and Random Forests) with transfer learning models. Nonetheless, even with this performance, we cannot assure the model's reliability in real-world applications. Additionally, to create a more robust diagnostic system, these models must be tested on image datasets which are a better representation of pap-smears that are inspected in the cytology labs currently.

7 Appendices

Appendix I: Repository

The codes for different models are available on Github repository.

Link: <https://github.com/SnehalBhartiya>

Appendix II: Mail regarding permissions for reproducing Fig.2.11 and 2.14

My email request :

Dear Sir,

I am Snehal Bhartiya, a master's student from the [Indian Institute of Science Education and Research, Pune](#) and am currently conducting my master's thesis on "Detecting Cervical Cancer using Deep Learning" with Dr.Pranay Goel.

I am unaware if it's permissible to reproduce the figures published under the Creative Commons License. Hence,I seek your humble permission for reproducing the figures 2, figure 3 and figure 5 in my master's thesis from your publication "Very Deep Convolutional Neural Networks for Complex Land Cover Mapping Using Multispectral Remote Sensing Imagery". I will ensure I cite the name of authors, paper and the publication properly.

Hoping to hear from you soon.

Regards,

Snehal Bhartiya

Confirmation mail:

Hi Snehal,

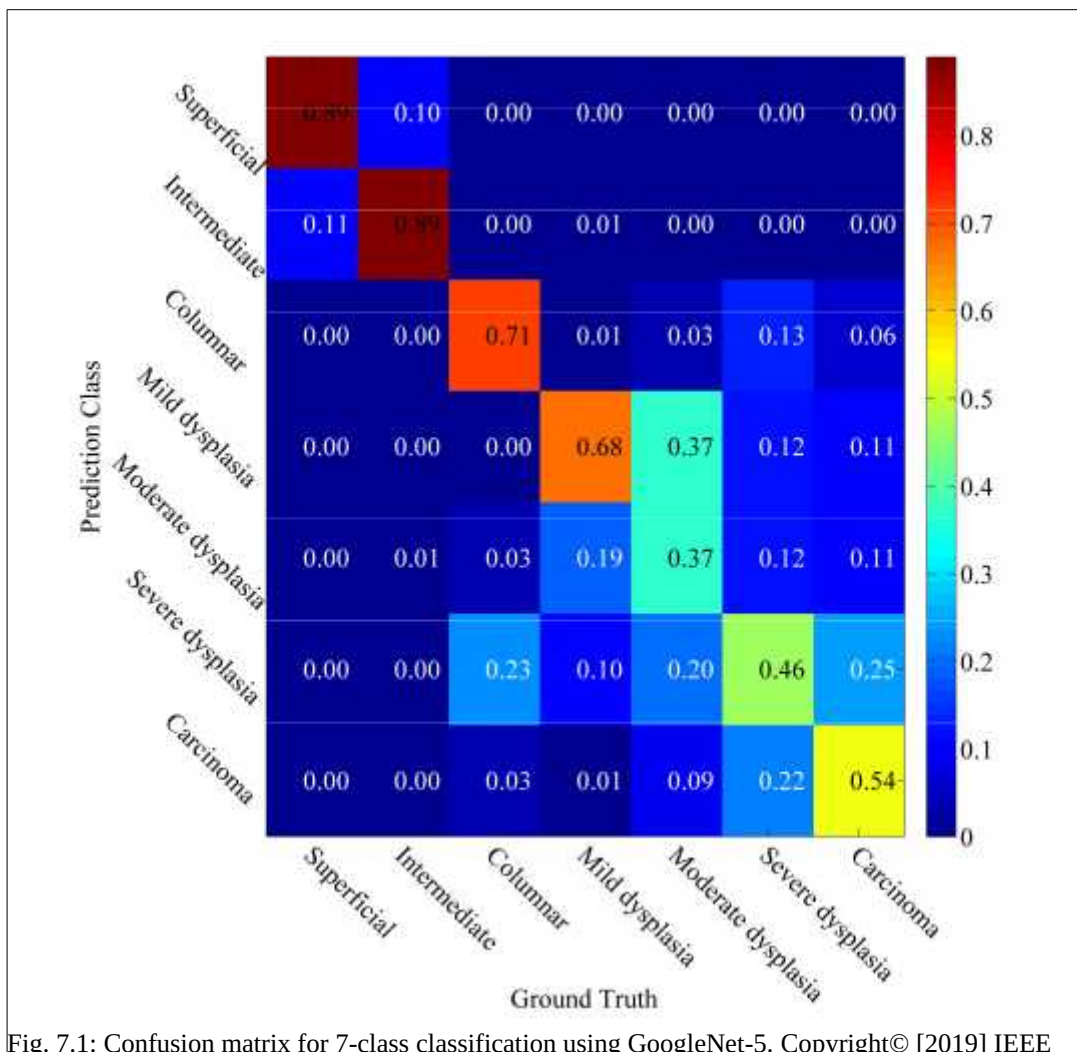
It is definitely OK. I'm very happy to see this information is useful in your research.

All the best,
Mohammad Rezaee, ph.D., PDF
Geodesy and Geomatics Engineering
University of New Brunswick
Fredericton, Canada

email: mrezaee@unb.ca

Appendix III: Confusion matrix for 7-class classification with GoogleNet

Adapted from: Lin, H.; Hu, Y.; Chen, S.; Yao, J.; Zhang, L. “Fine-Grained Classification of Cervical Cells Using Morphological and Appearance Based Convolutional Neural Networks”, IEEE 2019



Permission for reproducing figure: In the case of illustrations or tabular material in thesis, IEEE requires that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.

8 References

J. Ferlay, M. Ervik, F. Lam, M. Colombet, L. Mery, M. Pineros, et al., “Global cancer observatory: Cancer today,” Lyon, France: International Agency for Research on Cancer, 2018.

P. Release, “Latest world cancer statistics Global cancer burden rises to 14 . 1 million new cases in 2012 : Marked increase in breast cancers must be addressed.,” Int. Agency Res. Cancer, World Heal. Organ., no. December, pp. 2012–2014, 2013

C. Schwaiger, M. Aruda, S. Lacoursiere, and R. Rubin, “Current guidelines for cervical cancer screening,” *J. Am. Acad. Nurse Pract.*, vol. 24, no. 7, pp. 417–424, 2012

T. Mutyaba, F. A. Mmiro, and E. Weiderpass, “Knowledge, attitudes and practices on cervical cancer screening among the medical workers of Mulago Hospital, Uganda,” *BMC Med. Educ.*, vol. 6, 2006

Alaina J. Brown, M.D.¹ and Cornelia L. Trimble, *Best Pract Res Clin Obstet Gynaecol.* 2012 April ; 26(2): 233–242. doi:10.1016/j.bpobgyn.2011.11.001.

Teresa Conceição, Cristiana Braga, Luís Rosado and Maria João M. Vasconcelos “ A Review of Computational Methods for Cervical Cells Segmentation and Abnormality Classification”,*Int. J. Mol. Sci.* 2019

Le Lu et al. (eds.), *Deep Learning and Convolutional Neural Networks for Medical Image Computing, Advances in Computer Vision and Pattern Recognition.* Springer International Publishing, Switzerland 2017

Olaf Ronneberger, Philipp Fischer, and Thomas Brox 2015 ; U-Net: Convolutional Networks for Biomedical Image Segmentation; arXiv:1505.04597v1

Bora, K.; Chowdhury, M.; Mahanta, L.B.; Kundu, M.K.; Das, A.K. Automated classification of Pap smear images to detect cervical dysplasia. *Comput. Methods Progr. Biomed.* 2017

Zhang, L.; Lu, L.; Nogues, I.; Summers, R.M.; Liu, S.; Yao, J. DeepPap: Deep Convolutional Networks for Cervical Cell Classification. *IEEE J. Biomed. Health Inf.* 2017

Lin, H.; Hu, Y.; Chen, S.; Yao, J.; Zhang, L. Fine-Grained Classification of Cervical Cells Using Morphological and Appearance Based Convolutional Neural Networks

J. Jantzen, J. Norup, G. Dounias, and B. Bjerregaard, "Pap-smear benchmark data for pattern classification," *Nature inspired Smart Information Systems (NiSIS 2005)*, pp. 1–9, 2005

D. Solomon and R. Nayar, *The Bethesda System for reporting cervical cytology: definitions, criteria, and explanatory notes*. Springer Science & Business Media, 2004.

Ronneberger, O., Fischer, P., and Brox, T. (2015). "U-Net: convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer Assisted Intervention (Munich)*, 234–241.

Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation (2014)

Mashor MY. Hybrid multilayered perceptron networks. *Int J System Sci* 2000;31(6):771—85

Burnham, K. P.; Anderson, D. R. (2002), *Model Selection and Multimodel Inference* (2nd ed.), Springer-Verlag.

He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *CVPR*. (2016)

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *arXiv preprint arXiv:1512.00567*, 2015

C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *International Conference Learning Representations (ICLR) Workshop*, 2016

Nesterov, Y. (1983). A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$. *Doklady ANSSSR*

Geoffrey Hinton. 2012. Neural Networks for Machine Learning - Lecture 6a - Overview of mini-batch gradient descent

Kingma, D. P., & Ba, J. L. (2015). Adam: a Method for Stochastic Optimization. International Conference on Learning Representations,

Chankong T, Theera-Umpon N, Auephanwiriyakul S. Automatic cervical cell segmentation and classification in Pap smears. *Comput Methods Programs Biomed.* 2014

Journal I, Applications C, Bangalore- T. Papsmear image based detection of cervical cancer. *Int J Comput Appl.*2012

Ampazis N, Dounias G, Jantzen J. Pap-smear classification using efficient second order neural network training algorithms. Springer. 2004

Kangkana Bora, Manish Chowdhury, Lipi B. Mahanta, Malay K. Kundu, and Anup K. Das. 2016. Pap smear image classification using convolutional neural network. In Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP '16). Association for Computing Machinery, New York, NY, USA, Article 55, 1–8. DOI:<https://doi.org/10.1145/3009977.3010068>

Mahdianpari, Masoud & Salehi, Bahram & Rezaee, Mohammad & Mohammadimanesh, Fariba & Zhang, Yun. “Very Deep Convolutional Neural Networks for Complex Land Cover Mapping Using Multispectral Remote Sensing Imagery.”*Remote Sensing.* Vol 10. 2018

Wolpert, D.H., Macready, W.G. (1997), "No Free Lunch Theorems for Optimization", *IEEE Transactions on Evolutionary Computation* 1, 67.