Detection and Segmentation of Nuclei from Pathological Images

Thesis
Submitted in partial fulfillment of the requirements for the degree
of

MASTER OF TECHNOLOGY

in

COMPUTATIONAL MATHEMATICS

by

DEBARATI BHATTACHARJEE
(16CMA06F)

DEPARTMENT OF MATHEMATICAL AND COMPUTATIONAL SCIENCES
NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA
SURATHKAL, MANGALORE - 575025

June, 2018

DETECTION AND SEGMENTATION OF NUCLEI FROM PATHOLOGICAL IMAGES

by

DEBARATI BHATTACHARJEE
(16CMA06F)

Under the Guidance of

Dr. B. R. Shankar
Professor
Department of Mathematical and Computational Sciences
NITK Surathkal

Thesis
Submitted in partial fulfillment of the requirement for the degree of

MASTER OF TECHNOLOGY

in

COMPUTATIONAL MATHEMATICS

DEPARTMENT OF MATHEMATICAL AND COMPUTATIONAL SCIENCES
NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA
SURATHKAL, MANGALORE - 575025

June, 2018

DECLARATION

I hereby declare that the thesis entitled **"DETECTION AND SEGMENTATION OF NUCLEI FROM PATHOLOGICAL IMAGES"** which is being submitted to National Institute of Technology Karnataka, Surathkal, in partial fulfilment for the requirements of the award of degree of Master of Technology in Computational Mathematics in the department of Mathematical and Computational Sciences, is a *bonafide report of the work carried out by me*. The material contained in this report has not been submitted at any other University or Institution for the award of any degree.

.....

(16CMA06F, Debarati Bhattacharjee)
Department of Mathematical and Computational Sciences

Place: NITK, Surathkal
Date:

CERTIFICATE

This is to certify that the Thesis entitled **"DETECTION AND SEGMENTATION OF NUCLEI FROM PATHOLOGICAL IMAGES"** submitted by **Debarati Bhattacharjee** (Register number 16CMA06F) as the record of the work carried out by her, is *accepted as the P.G Major Project Thesis submission* in partial fulfilment for the requirements of the award of degree of Master of Technology in Computational Mathematics in the department of Mathematical and Computational Sciences at National Institute of Technology Karnataka, Surathkal during the academic year 2016-2018.

.....	.....	.....
Dr. Jeny Rajan,	**Dr. B. R. Shankar,**	
External Guide	Internal Guide	Chairman - DPGC
Department of Computer Science and Engineering, NITK, Surathkal	Department of Mathematical and Computational Sciences, NITK, Surathkal	Department of Mathematical and Computational Sciences, NITK, Surathkal

ACKNOWLEDGEMENT

This project would not have come into fruition without the help and guidance of many individuals. I would like to formally thank them and express my gratitude in this acknowledgement.

I express my gratitude to my project guide Dr. Jeny Rajan for giving me an exciting project like this. His help, advice, valuable insights have, I believe, enhanced my scholastic abilities considerably.

I would also like to thank Girish GN for always being present as a guide and friend during the entire duration of the project. This project would not have been completed without his expertise and knowledge in image processing. He helped me a lot in all stages of this project.

Lastly, I have to thank Dr. B. R. Shankar, the Head of the Department, Mathematical and Computational Sciences, and internal guide for this project for giving me general advices about how to carry out a challenging project.

In the end I express my heartfelt gratitude to the department of Mathematical and Computational Sciences for providing me thourough assistance throughout the duration of this project.

Place: Surathkal, Mangalore, India Debarati Bhattacharjee
Date:

ABSTRACT

Image analysis of cells and tissues has been an emerging field of research due to its immense importance in the field of medical informatics for comprehensive studies of cell morphology and tissue structure. Automated cell segmentation is extremely laborious due to the variability in size, shape, appearance, and texture of the individual nuclei and the scarcity of pathological images. Manual assessment of pathological images is impossible due to its proneness to failure in observing subtle microscopic details and is needed extreme unrealistic human labour. In this project, a deep learning based automated nuclei segmentation method is applied to four types of cancerous cells: glioblastoma, log grade glioma, non small cell lung cancer, head and neck squamous cell carcinoma cancer. Two different ways to divide the dataset into training set and test set are followed. In one way, both of the training set and the test set consist of images of cells of all of the four types (heterogeneous test set). In the other way, the test set consists only of non small cell lung cancer cells, on which the model is not trained at all, and, the training set consist of cells of other three types (homogeneous test set). A convolutional neural network based model is trained with both the training sets and is tested with both types of the test sets. For the heterogeneous test set the model achieves a mean Dice coefficient value of 0.79, and for homogeneous test set the model achieves a mean Dice coefficient value of 0.68.

Keywords **—** Nuclear segmentation, nuclei, convolutional neural networks, deep learning, digital pathology, medical imaging.

Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

The pathologist plays a central role in therapeutic decision making. Accordingly, diagnosis from pathology images contributes in diagnosing a number of diseases including most cancers. Diagnosing a disease after manually analyzing numerous biopsy slides represents a laborious work for pathologists. Due to advancements in digital pathology, the automated recognition of pathology patterns in a slide image has the potential to provide valuable assistance to the pathologist in his daily practice.

Researchers in pathology have been familiar with the importance of quantitative analysis of pathology images. Quantitative analysis can be used to support pathologists' decisions about the presence of a disease, and also to help in disease progression evaluation. In addition, quantitative characterization is important, not only for clinical usage (e.g., to increase the diagnostic reliability), but also for research applications (e.g., drug discovery and biological mechanisms of disease). As a consequence, the use of computer-aided diagnosis (CAD) in pathology can substantially enhance the efficiency and accuracy of pathologists' decisions, and overall benefits the patient.

1.1 Nuclei Segmentation

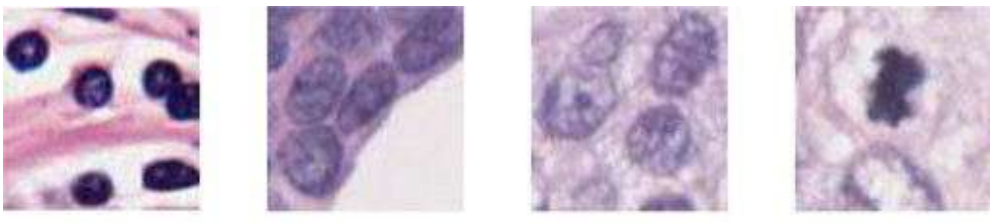
Image Segmentation is a classical computer vision problem wherein parts of the image have to be grouped into a set number of classes. Unlike image classification, where the entire image is analyzed as a whole and assigned to a class, in segmentation, per-pixel classes are assigned. Our proposed technique is motivated by the need to identify pixels in background (outside all nuclei)

and foreground (inside any nucleus). Thus, the problem is reduced to a binary classification problem for every pixel.

Supervised methods or learning is a branch of machine learning where the computer is taught by showing examples. In this approach, there is a training data set where the input is shown to the machine along with its corresponding correct label. The machine learns to predict the label of the test set based on the knowledge it has gained from the training set. In unsupervised learning, the machine makes decisions by looking at the data itself; there is no prior knowledge that the machine utilizes. In the proposed approach, the supervised learning method of convolutional neural networks (CNN), which are a special type of ANNs has been applied.

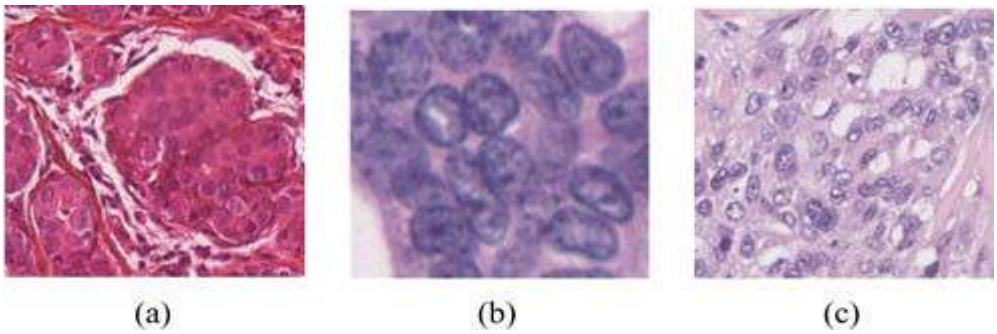
1.2 Challenges in Nuclei Segmentation and Classification

Nuclei may look very different according to a number of factors such as nuclei type, malignancy of the disease, and nuclei life cycle. Lymphocyte is a type of white blood cell that has a major role in the immune system. Lymphocyte nuclei (LN) are inflammatory nuclei having regular shape and smaller size than epithelial nuclei (EN) [see Fig. 1(a)]. Nonpathological EN have nearly uniform chromatin distribution with smooth boundary [see Fig. 1(b)]. In high-grade cancer tissue, EN are larger in size, may have heterogeneous chromatin distribution, irregular boundaries, referred to as nuclear pleomorphism, and clearly visible nucleoli as compared to normal EN [see Fig. 1(c)]. The variation in nuclei shape, size, and texture during nuclei life cycle, mitotic nuclei (MN), is another factor of complexity [see Fig. 1(d)].



1.2.1 Different types of nuclei. (a) LN. (b) EN. (c) EN (Cancer). (d) EN (Mitosis)

Detection and segmentation of nuclei in pathology images pose a difficult computer vision problem due to high variability in images caused by a number of factors including differences in slide preparation and image acquisition (artefacts introduced by the compression of the image, presence of digital noise, specific features of the slide scanner, etc.). All these problems (highlighted in Fig. 2) make the nuclei detection and segmentation a challenging problem. A successful image processing approach will have to overcome these issues in a robust way in order to maintain a high level in the quality and accuracy in all situations.



1.2.2 Examples of challenging nuclei to detect and segment (a) Blur (b) Overlaps (c) Hetero – geneity

1.3 Motivation

Nucleus appearance features such as density, nucleus-to-cytoplasm ratio, average size etc. can be helpful for assessing cancer grades and predicting treatment effectiveness ([1], [2]). Identifying

different types of nuclei based on their segmentation can also yield information about gland shapes, which is important for cancer grading [3]. Thus, techniques that accurately segment nuclei in diverse images, spanning a range of patients, organs, and disease states, can significantly contribute to the development of clinical and medical research software.

1.4 Summary

Chapter 2 contains a brief overview of the current research already done in this and closely-related fields. It also contains an overview of the state-of-the-art machine and deep learning models currently being used to solve complex problems in the field of Image Processing and Computer Vision. Chapter 3 describes the complete methodology of the proposed method in detail. It starts with a detailed background to the working of Convolutional Neural Networks and goes on to explain the network architecture used in the proposed method. All experiments performed in finding the correct architecture and then training the actual model are described in Chapter 4. It also specifies the hardware and software requirements of the proposed method. The chapter also includes the results obtained by the proposed method . Chapter 5 concludes the report and provides an overview of any future work that can be performed as an extension of this work.

Chapter 2

LITERATURE REVIEW

This chapter contains a brief overview of the current research, mostly on deep learning, already done in this and closely-related fields.

Mina Khoshdeli et al. [4] proposed a convolutional neural network (CNN) model to detect nuclei in 2017. Detection of nuclei is accomplished by using a CNN which is composed of two basic parts of feature extraction and classification. Feature extraction includes several convolution layers followed by max-pooling and an activation function. The classifier usually consists of fully connected layers. This results in a probability map which indicates the probability of each pixel to be the centroid of a nucleus. The batch gradient descent with the batch size of 256 is used for back propagation optimization. The learning rate is set at 10^{-5} and the learning rate decay is set at 10^{-7} . L1 and L2 regularizations were performed with weights of 0.001 and 0.01, respectively. Nuclei detection accuracy of the various approaches is calculated based on precision, recall, and F-Score. Since some of the nuclei may be detected more than one time, the percentage of over-detected nuclei are also reported. The validation dataset consists of 29 histology sections of size 1k × 1k, which includes 21 brain and 8 breast images. These images have been hand segmented totaling 13,766 nuclei. Images were equally divided between training and testing samples.

Laith Alzubaidi et al. [5] proposed a robust deep learning approach to detect nuclei centres in histo-pathological images in 2017. The proposed a continuous formulation of nuclei detection problem where the input image is mapped into a continuous valued map and all background pixels are assigned zero but nuclei are assigned a positive value corresponding to their distance to the

nuclei boundary. The model convolutional regression network architecture consists of four types of layers: (1) convolutional, (2) pooling, (3) convolution prediction (FC), and (4) up-Sampling. The proposed model was trained with the colorectal dataset and, then, was tested on the human bone marrow dataset without retrain. The method handled most nuclei detection's problems (merged cells, size, and color). The result is assessed in terms of precision recall and F-measure where TP is true positives, FP is false positives, FN is false negatives. This model achieved 85.5% precision, 94.7% recall, and 89.9%F-measure.

Neeraj Kumar et al. [6] used a convolutional neural network (CNN) based model for prostate cancer (PCa) recurrence prediction in 2017. In the first stage, a CNN is used to detect the nuclear centres with high accuracy in a given tissue image. The algorithm was trained to detect both epithelial and stromal nuclei in tumour as well as non-tumour regions of tissue images. The second stage uses another CNN which takes patches centred around the detected nuclear centres as input to estimate patch-wise cancer recurrence probability as its output. This stage can easily be modified and retrained to predict the probability of different sub-types of cancers in different types of tissue images. The final sub-type probability for a given patient (probability of recurrence) is determined by aggregating its patch-wise probabilities determined by the CNN. The CNN was trained gave an accuracy of 89.6% for nucleus detection on an independent set of validation images. The nucleus detection scheme was able to detect both the annotated and unannotated nuclei. However, it failed to detect a few crowded nuclei. The PCa recurrence prediction scheme gave 0.81 area under curve (AUC) as compared to 0.59 AUC using the usual clinical features.

Eliezer Farrant Braz et al. [7] proposed a nuclei detection model for cervical cells using deep learning. The CNN used here was trained to classify each pixel as belonging to one of the following

classes: nucleus, cytoplasm, background. The method achieved a precision of .929 which is equivalent to that of the other state-of-the-art methods and a superior recall of .917. If two or more nucleus are close to each other, then, sometimes, this model failed to detect all the nucleus.

Fuyong Xing et al. [8] proposed a novel nucleus segmentation framework using deep convolutional neural network and selection-based sparse shape model. The approach starts with a deep learning-based iterative region merging algorithm to initialize the contours, and thereafter alternately performs efficient bottom-up shape deformation and robust top-down shape inference to achieve correct nucleus segmentation.

Prabhjot Kaur [9] discussed a mammogram image nucleus segmentation and classification method using CNN for detection of breast cancer. Nucleus segmentation is used to find out the region of interest (ROI). The result of ROI is further used for extracting the valuable shape and textural features by using geometrical features, Gray Level Co-occurrence Matrix (GLCM) and Gray Level Difference Method (GLDM) for classifying the cancer through the machine learning approach i.e. CNN. CNN removes the overlapping of features obtained after segmentation. The performance of proposed CNN based classifier approach is measured in terms of accuracy, precision and recall and compared to other approaches based on neural network, logistic regression. It is found thet the proposed approach achieves an accuracy of 96.66, a precision of 95.23 and, a recall of 99.86 and outperforms other approaches.

Mohammad Havaei et al. [10] proposed a CNN based approach to brain tumour segmentation in 2015. The high performance is achieved with the help of a novel two-pathway architecture of CNN (which can model both the local details and global context). Apart from that, a two phase patch wise

training procedure is incorporated to the model to train in only in a few hours. Also, this model is as fast as segmenting a complete brain image in only 3 minutes. The two-pathway method allows the model to have more contextual information of the tumour and yields better segmentations. By allowing for a second phase training and learning from the true class distribution, the model corrects most of the misclassifications produced in the first phase.

Angel Cruz-Roa et al. [11] proposed an automatic detection of invasive ductal carcinoma (IDC) tissue regions in whole slide images (WSI) of breast cancer with CNNs. The methodology consists of image patch sampling, CNN based processing (convolution layer, pooling layer, fully connected layer, classification layer), building IDC probability map. This model yields precision, recall or sensitivity and specificity of 65.40%, 79.60% and 88.86% respectively. Future work will explore the effects of CNN models with deeper architectures of more layers, neurons and validation on larger cohorts.

Neeraj Kumar et al. [12] introduced a large dataset of human tissue images with annotated nuclear boundaries from a diverse set of patients and organs, and, proposed a technique of nuclear segmentation using a CNN. They showed that it gives reasonable results even on organs on which it was not trained, thus demonstrating generalization. They proposed a new evaluation metric, called, Aggregated Jaccard Index (AJI) to evaluate the performance of a nuclear segmentation method. Their architecture for binary pixel classification gave 92% accuracy.

Olaf Ronneberger et al. [13] presented a network that relies on the use of data augmentation to use the avilable annotated samples more efficiently. The proposed architecture, named as u-net architecture, consists of a contracting path and a symmetric expanding path. The proposed

architecture is fast. It was claimed that the segmentation of a 512x512 image took less than a second on a recent GPU. They achieved an average IOU (“intersection over union”) of 92%, which was significantly better than the second best algorithm with 83% on “PhC-U373” dataset, and, an average IOU of 77.5% which is significantly better than the second best algorithm with 46% on “DIC-HeLa” dataset.

2.1 Problem Statement

To design a general algorithm, to segment nuclei from cells in pathological images, which performs reasonably well on datasets spans several organs. To demonstrate generalization, the algorithm is tested on organs on which it was trained, and, also organs on which it was not trained.

Chapter 3

BACKGROUND & METHODOLOGY

3.1 Preliminary Background

Before diving into the methodology and framework of the proposed solution, there is requirement of some preliminary background into the working of neural networks in general and convolutional neural networks in particular. Artificial Neural Networks are a type of machine learning algorithm which essentially are a stack of linear models connected in succession. A linear model is basically a function *f (x)* which is defined as:

$$f(x) = W.x + b \tag{3.1.1}$$

Here, x is a vector of inputs to the function, W is a matrix of weights mapping the input vector to the output and b is a vector of bias parameters added to the output of the vector-matrix multiplication. The output of this linear model will also be a vector of the size defined by the weight matrix and bias vector. The simplest classification strategy can be implemented with just one linear model. In that case, x will be the input to be classified, lets say a vector of size D × 1. The task is to classify the input into one of C classes and hence, the output of the linear model will be a vector of size C × 1. So logically, the weight matrix W has to be of a size C × D and the bias vector will be of size C × 1. The vector output from this model is called a score vector, as it typically contains the confidence value of each output class associated with the current input. Generally, the vector index with the greatest value associated with it is considered the class predicted by the model. In machine learning, such type of models are used to make a computer learn to accurately solve complex problems. This is done with the process of optimization. Optimization in machine learning is an

iterative process where a model learns to correctly classify the input by looking at a large number of similar examples given to the model, gradually improving its predictions. This is done with the aid of an error or a loss function.

3.1.1 Optimization and Loss Functions

In standard machine learning workflow, the first task performed is to divide the data into a separate training set and test set. The training set is used to optimize the model to make correct predictions and the test set is used to model the real world and judge how well the model performs in the specified task. No optimization is performed on the test set. After the division, the model is initialized with some random weights and the data in the training set is fed into the model in some order. The input passes through the random weights and outputs a score vector. This score vector is compared with the actual class output of the provided input using a loss or error function. Based on the magnitude of this loss function, the weights are updated such that the model gives a lower loss the next time this input is seen.

Cross-Entropy Loss

There are different loss functions proposed in literature to optimize such models. One of the popular ones is the cross-entropy loss that has been used in the proposed method. The cross-entropy loss is defined as [14]:

$$L_i = -\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right) \quad (3.1.1.1)$$

where, L_i is the loss and y_i is the actual correct label for input i . f_j denotes the

j th element of the vector of class scores f . The cross-entropy loss is composed of the softmax function :

$$\sigma(\ f_{y_i}\ ) = \frac{e^{f_{y_i}}}{\sum_j e^{f_j}} \qquad (3.1.1.2)$$

The softmax function can be intuitively understood from a probabilistic perspective. It can be can be interpreted as the (normalized) probability assigned to the correct label y_i given the input x and parameterized by W and b. The Softmax function interprets the scores inside the output vector f as unnormalized log probabilities. Exponentiating these quantities therefore gives the unnormalized probabilities, and the division performs the normalization so
that the probabilities sum to one [15]. Therefore, the cross entropy loss minimizes the negative log likelihood of the correct class.

L_i is the loss associated with a single input value. The total loss value L over all N inputs is defined as the mean of the individual loss function values for each input [14]:

$$L = \frac{1}{N} \sum_i L_i \qquad (3.1.1.3)$$

Optimization: Stochastic Gradient Descent

The standard gradient descent algorithm updates the parameters θ of the objective $J(\theta)$ as,

$$\theta = \theta - \alpha \nabla_\theta E[J(\ \theta\ )] \qquad (3.1.1.4)$$

where the expectation in the above equation is approximated by evaluating the cost and gradient over the full training set. Stochastic Gradient Descent (SGD) simply does away with the expectation in the update and computes the gradient of the parameters using only a single or a few training examples. The new update is given by,

$$\theta = \theta - \alpha \nabla_{\theta} J(\theta; x(i),y(i)) \quad (3.1.1.5)$$

with a pair *(x(i),y(i))* from the training set. Generally each parameter update in SGD is computed w.r.t a few training examples or a minibatch as opposed to a single example. The parameter α refers to the step size or learning rate of the gradient descent. It defines the magnitude of change on the original parameters in each iteration. The learning rate is an important hyperparameter in machine learning algorithms, which defines how fast the model will train. A hyperparameter is a parameter that must be explicitly specified by the user and cannot be learned by a machine. A higher learning rate, which makes the loss converge faster, may not always be ideal as the updated parameters may overshoot the minima. The gradients are subtracted from the original values because a positive slope value of a function is always in the direction of the minima.

In stochastic or mini-batch gradient descent, the gradient is calculated over a set batch of inputs and the weights are updated. These batches are fed into the model one after the other until the entire training set is exhausted. This is called an epoch. The size of the batches are generally constrained by the amount of physical memory of the machine, as the entire batch has to be fed into the memory at once.

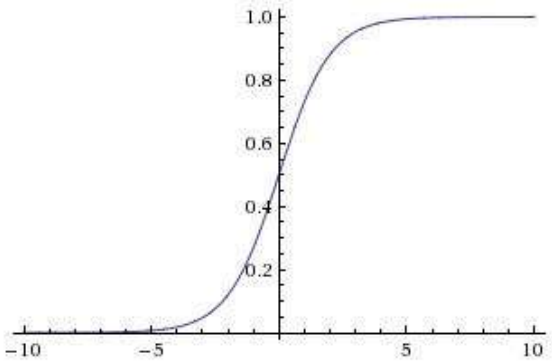
3.1.2 Artificial Neural Networks

A single linear model is generally not powerful enough to emulate complex mappings from the input to the output classes. There is a need to stack these linear models one after another to better describe complex functions that occur in the real world. A stack of linear models arranged in a manner such that the output of one linear model becomes the input of another linear model forms the basis for artificial neural networks. An ANN is just a stack of linear models with activation functions between them that map some complex relation between the input to some output.

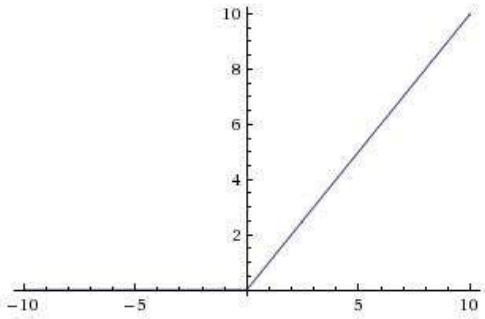
Activation Functions

Activation functions or non-linear functions are functions applied on the output of a linear model to make it non-linear. This non-linear output may be passed as the input to another linear model. Activation functions are required because of the property of composition of linear models. This property states that, if f and g are two linear models and we have a composite function $y = f(g(x))$, then there always exists another linear model h that accurately describes *y*, i.e.

$$y = f(g(x)) = h(x) \tag{3.1.2.1}$$



(a) Sigmoid Activation



(b) ReLU Activation

3.1.2.1 Non-Linear Activation Functions [15]

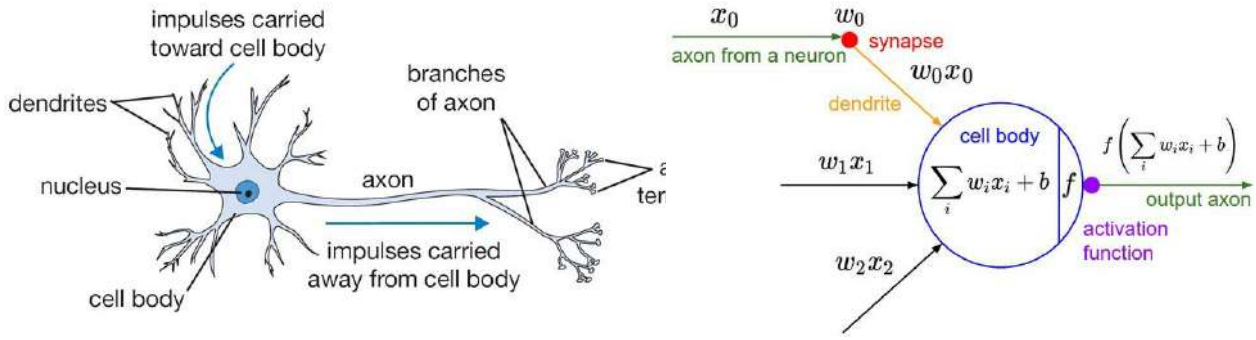
Practically, this means that multiple linear models without a non-linearity in between can be broken down into a single linear model. To better map complex relationships, a non-linear function is applied on the output of a linear model. The two most commonly used non-linearities are the sigmoid function σ and the ReLU function. The sigmoid function takes a value and squashes it into the range 0 to 1. The sigmoid function is of the form [14]:

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{3.1.2.2}$$

where, x is the input to the sigmoid function σ. The sigmoid function can be interpreted as a binary softmax function, where its value can be thought of as the confidence level between 2 distinct

classes. If the value is leaning more toward 0 or 1, the model is more confident about the input belonging to that particular class. Median values near 0.5 can be treated as unsure observations. The sigmoid function is generally not used in any intermediate layers of an ANN because, if a sigmoid model output is 0 or 1, its gradient becomes 0, after which the model cannot update the weights. Sigmoid can be used in the last layer if it is a binary classification problem. The ReLU or Rectified Linear Unit non-linearity is the most commonly used activation function for intermediate layers. Its simple definition of thresholding the output at zero (3.9) greatly accelerates the training time of neural networks applying gradient descent [14]:

$$f(x) = max(0, x) \tag{3.1.2.3}$$

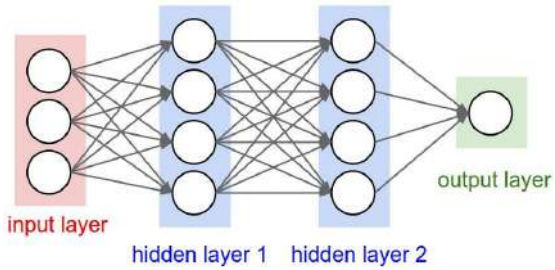


(a) Model of nerve cell **(b) Neuron as a Linear Model**

3.1.2.2 Models of Biological and Computational Neurons [15]

It has been proved that a stack of one or more linear models followed by a non-linearity, work as universal approximators, i.e, they can approximate the output of any mathematical function with a tolerance value of ϵ [16]. This concept is analogous to how the human brain works. As shown in figure 3.1.2.2. (a), the working component of the human brain is a nerve cell, called a neuron. A neuron accepts inputs from other neurons with variable synaptic strengths through its dendrites. Its

output is propagated through the axon towards its terminals and is accepted by the dendrites of some other neurons with their respective strengths. In the computational model of the neuron, figure 3.1.2b, multiple inputs coming in may be multiplied with some pre-defined weights and then added. An activation function is applied to the sum of the weighted inputs and the bias parameter of the neuron, thus functioning as a linear model passed through a non-linearity. The output is then passed to other neurons. A combination of these neurons arranged in an acyclic graph architecture forms a Neural Network or ANN. Neurons are arranged in distinct layers. The neurons in a single layer accept inputs from the previous layer and passes outputs to the next layer, but individual neurons in a single layer are not connected. A simple ANN with fully connected layers is described in figure 3.1.3. A fully connected layer is a layer in which every neuron is connected to every neuron in the previous layer. The input and output layer out of a neural network do not have any weights assigned. The layers which perform the linear computation with weights are present in the hidden layers.



3.1.2.3 An Artificial Neural Network with 2 hidden layers [15]

Backpropagation

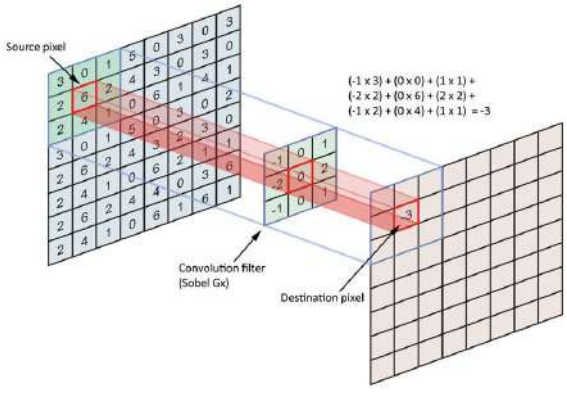
All the weights of an ANN are randomly assigned when it is first created. The weights are automatically updated based on the training data and its associated labels by computing the loss function at the output layer. The entire network then acts as a single model which accepts an input and outputs a class as the output. Based on the loss obtained, the weights of the hidden layers are

updated such that the loss is minimized. Propagating the gradient of the loss function through the layers of the network is performed by a method called backpropagation. Backpropagation or reverse-mode differentiation is a method by which we can check the influence of every input and intermediate node of a computational graph in the output. In the context of neural networks, it can be used to calculate the effect of every neuron in the loss function. Before computing the reverse gradient, for all the layers, the gradient of that layer with respect to the previous layer must be known. Then, starting from the output layer, the reverse is calculated using the chain rule of differentiation. Let the gradient of the loss function L with respect to the weights of the output layer p be δ_{L_p} . Then the gradient of the loss function with respect to the weights of the previous layer will be :

$$\delta_{L_{p-1}} = \frac{\partial L}{\partial f_{p-1}} = \frac{\partial L}{\partial f_p} \quad \frac{\partial f_p}{\partial f_{p-1}}$$

$$\text{i.e.,} \quad \delta_{L_{p-1}} = \delta_{L_p} \quad \frac{\partial f_p}{\partial f_{p-1}} \tag{3.1.2.4}$$

where f_p and $\overset{\cdots}{f}_{p-1}$ signify the activated outputs of the p th and (p − 1) th layer respectively. Equation (3.1.2.4) can be iteratively applied on every layer in sequence till the input layer is reached.



3.1.2.4 Convolution with a Sobel Filter

3.1.3 Convolutional Neural Networks

In Convolutional Neural Networks, the standard linear model of an ANN is replaced by a convolution operation. All other concepts of ANNs like Loss Functions, Optimizers, Activation Functions and Backpropagation carry forward to CNNs. The successful training and implementation of CNN architectures have been a landmark in the field of Computer Vision and Machine Intelligence. On the ImageNet classification challenge, widely regarded as the Olympics of Computer Vision, different CNN architectures have brought down the error rate from 28% in 2011 to 2.9% in 2016. In comparison, humans have an error rate of 5.1%.

The Convolution Operation

Mathematically, a convolution is an integral that expresses the amount of overlap of one function g as it is shifted over another function f. Discrete time convolution is an operation on two discrete time signals defined by the integral

$$f * g = \sum_{k=-\infty}^{\infty} f(k)g(n-k) = \sum_{k=-\infty}^{\infty} f(n-k)g(k) \qquad (3.1.3.1)$$

for all signals f, g defined on $\mathbb{Z}$. It is important to note that the operation of convolution is commutative. In the context of image In the context of image processing, a convolution is the dot product of a convolution matrix or kernel G being shifted over an input image matrix F . The kernel or filter is aligned with a portion of the image having the same size as the kernel itself, all the values in the corresponding positions are pairwise multiplied and finally added to get a single value as the output. It can be intuitively understood that the convolution output will have a higher value when the overlapped region of the image is very similar to the convolution kernel. As a result, filters can be said to identify particular visual “elements” of an image. The advantage of using convolutions for detecting these features of the image is positional invariance. Since one filter is slided over the

entire image, it will detect the feature wherever it is located in the image. A collection of filters which work on the same input is grouped together and named a convolutional layer. The reason why Convolutional Networks work so well is because they function as automated hierarchical feature extractors. The first convolution layer accepts the image itself as the input. Later layers, on the other hand, take the output activation of the previous layer as its input. This forms a hierarchical structure.

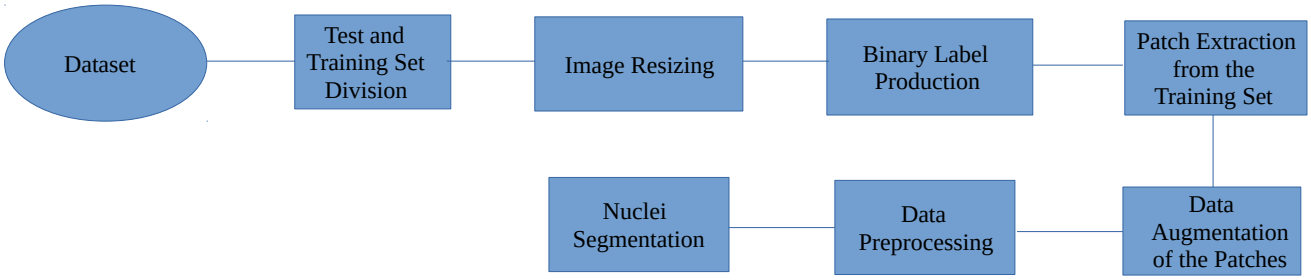
CNN Components

Apart from a convolutional layer, CNNs stack pooling layers and fully connected layers to construct a complete architecture. A pooling layer is a downsampling operation performed on the input. This is done for two reasons: Reducing the number of parameters in the network and also to counter overfitting. It accepts two hyperparameters as input: the filter size and the stride. These parameters decide the extent of downsampling to be performed. For example, a 2x2 pooling size with stride 2 will reduce the spatial resolution of the input by exactly half. The most common pooling operation performed is max-pooling, where the maximum value in a filter-size space in the input is given as the output. Fully connected layers in CNNs function in the same way as in Artificial Neural Networks. Thay are generally used as the final layers of a CNN architecture to map the features extracted from convolutions to the corresponding class score outputs. Stacking convolutional layers with pooling, fully-connected layers etc. in different arrangements form a CNN architecture. Coming up with the best architecture and hyperparameters for the specific problem at hand is the core research area when working with Neural Networks.

3.2 Methodology

The methodology followed in the proposed approach can be mainly broken down into five phases:

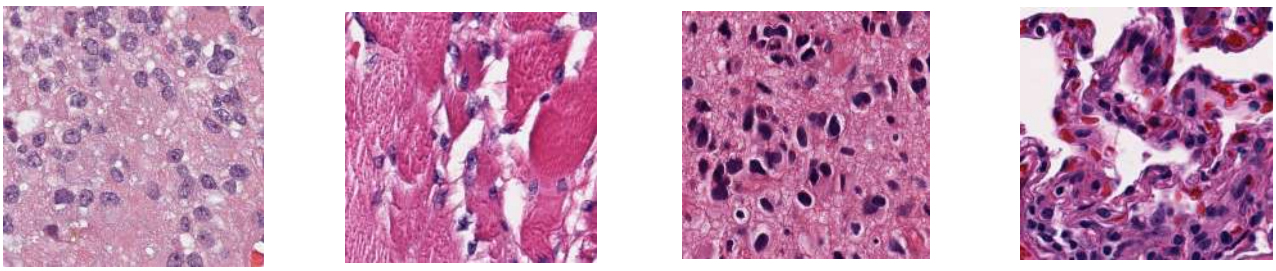
image resizing, binary label production, patch extraction from the training set, data augmentation of the patches, and, lastly, nuclei segmentation.



3.2.1 Flow Chart of the Proposed Methodology

3.2.1 Test and Training Set Division of Images

The entire image dataset consists of 32 images and their corresponding labels. These images are spanned from various organs of human bodies, and, are collected from patients of different diseases. Among those 32 images 8 are of glioblastoma (GBM) cells, 8 are of head and neck squamous cell carcinoma (HNSC), 8 are of log grade glioma (LGG) cells, and, the rest are of non small cell lung cancer (lung) cells. All cells are cancerous.



GBM HNSC LGG Lung

3.2.1.1 Samples of Imges of Four types of Cells in the Dataset

The division of images into test set and training set is done in two different ways. In first way, the test set is heterogeneous collection of images. Among 8 test images, 2 are of GBM, 2 are of HNSC,

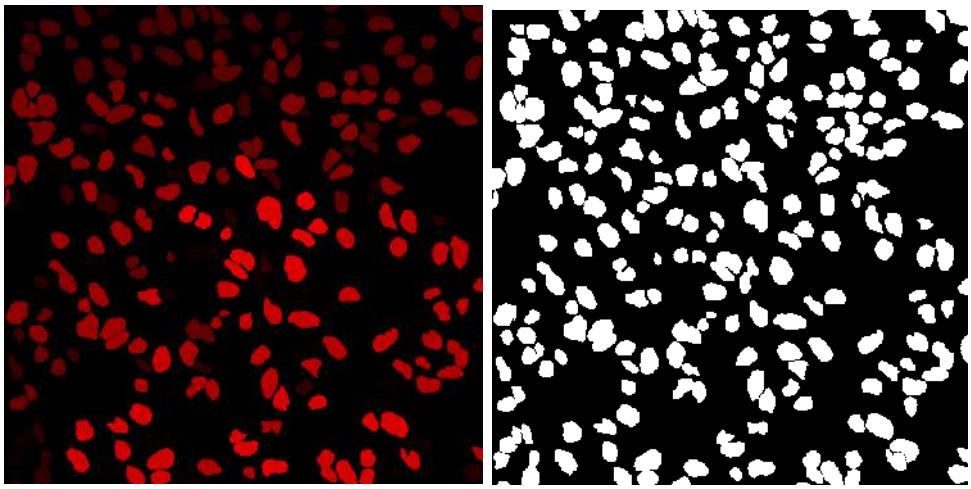
2 are of LGG and, the remaining 2 are of lung. The remaining 24 are chosen to form the training set. In second way too the training set contains 24 images and the test set contains 8 images, but, the test set is homogeneous. It consists of only lung images, with which the model is not trained at all. Hence, it makes the segmentation task much more challenging for the model than the first way of division.

3.2.2 Image Resizing

The sizes of the images in the dataset are unevenly spanned from 603×603 to 495×495. So, all the images are resized to 500×500.

3.2.3 Binary Label Production

The labels of all training images and test images are RGB images. All the label images are first converted to gray images. For color conversion the OpenCV function *cv2.cvtColor*(*input_img, flag*) is used. Here *flag* determines the type of conversion. For BGR to gray conversion, the flag is assigned as *cv2.COLOR_BGR2GRAY* in OpenCV. Now the gray image is converted to binary image by simple thresholding. The function used is *cv2.threshold*(*source*, *threshold, max*, THRESH_BINARY). First argument is the source image, which should be a grayscale image. Second argument is the threshold value which is used to classify the pixel values. Third argument is the maximum value which represents the value to be given if pixel value is more than (sometimes less than) the threshold value. OpenCV provides different styles of thresholding and it is decided by the fourth parameter of the function. The chosen *threshold* value for this project is 0 and *max* value is 255. The THREASH_BINARY function simply changes intensity of the pixels to max value if the intensity more than the threshold, otherwise intensity is assigned to 0.



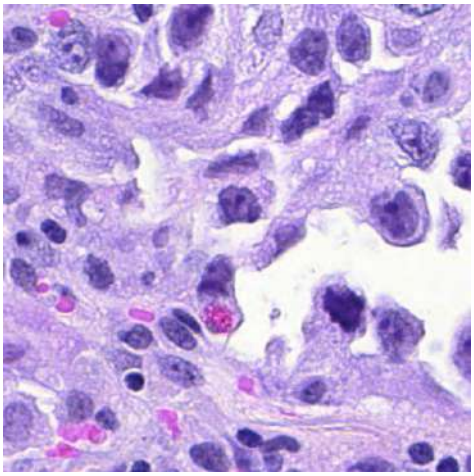
(a) Original Label Image **(b) Binary Image**

3.2.3.1 Binary Label Production

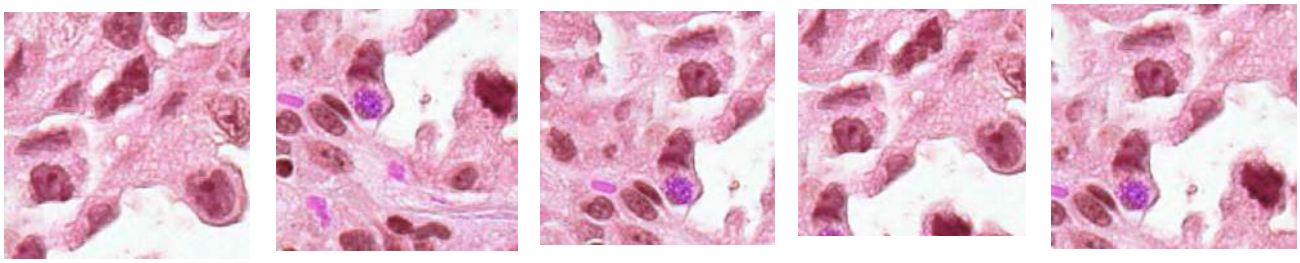
3.2.4 Patch Extraction from the Training Set

As the number of images for training the model is very scanty, only 24, patch extraction and data augmentation (discussed in the next section) are used to increase the number of images for training. From each training image, five patches (total 120 patches) of size 256×256 are extracted with the function with *extract_patches_2d*(*img*, *patch_size*, *max_patches=5*, r*andom_state=1*) funion which stores the patches in a dedicated array. The first parameter *img* is a 3-dimensional array which contains the original image data in this order: image height, image width, number of channels. The second parameter *patch_size* contains the dimension of a patch (height and width). The third parameter *max_patches* decides the maximum number of patches to extract. The third parameter *random_state* is a pseudo number generator state, used for random sampling to use if *max_patches* is not None. If int, random_state is the seed used by the random number generator; If RandomState instance, random_state is the random number generator; If None, the random number generator is the RandomState instance used by *np.random*. The function returns the collection of patches extracted from the image, where number of patches is either the *max_patches* or the number of

patches that can be extracted. Patches are extracted from both training images and corresponding label images.



(a) Original image



(b) Extracted patches

3.2.4.1 Patch Extraction

3.2.5 Data Augmentation

data augmentation is a technique to generalize the model and reduce over-fitting. So it is applied to data before training to increase data. Augmented data is generated by applying rotation, width shift, height shift, zoom, horizontal flip, shearing. After data augmentation the size of training set increases to 3112.

3.2.6 Data Preprocessing

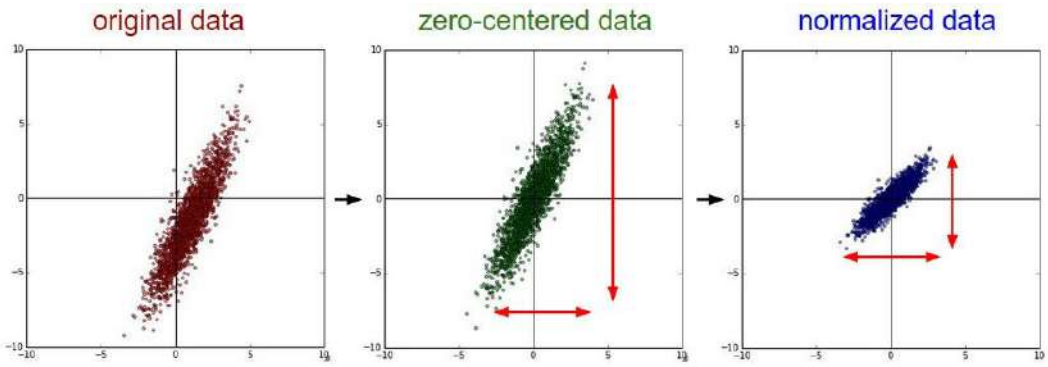
Two common forms of data preprocessing are used. They are discussed in the next two paragraphs.

Mean Subtraction

It involves subtracting the mean across every individual *feature* in the data, and has the geometric interpretation of centering the cloud of data around the origin along every dimension. With images specifically, it is common to subtract the mean from all pixels, or to do so separately across the three color channels.

Normalization

It refers to normalizing the data dimensions so that they are of approximately the same scale. To achieve this normalization, each dimension is divided by its standard deviation, once it has been zero-centered after mean subtraction.



3.2.6.1 Data Preprocessing [15]

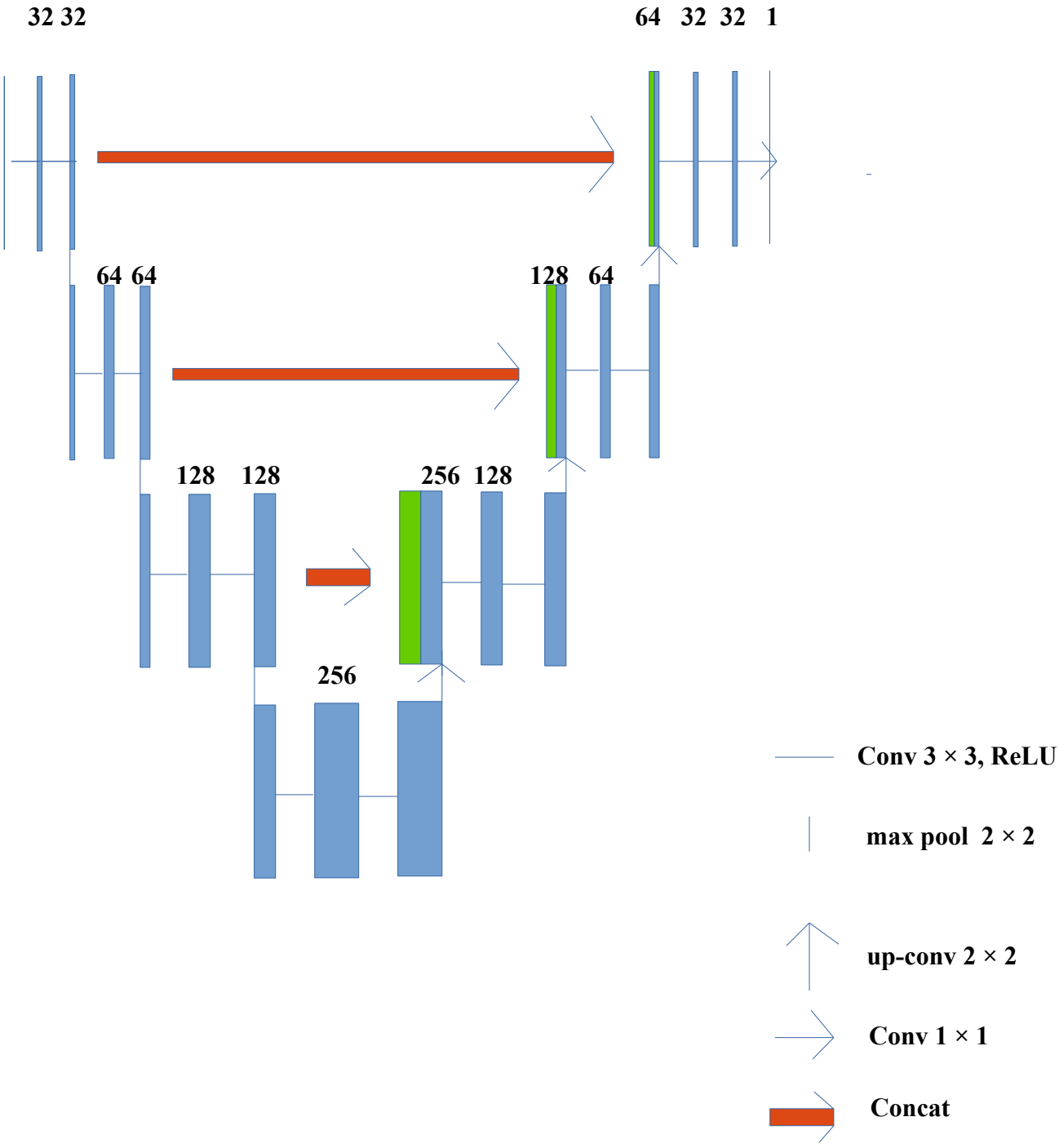
3.2.7 Nuclei Segmentation

The nuclei segmentation is performed with a Fully Convolutional Neural Network (FCN) first described by Long et al. [17]. Such a network can take an image as input and produces a prediction score matrix of the same size. This prediction score matrix is used to contruct the binary segmentation output. The network architecture used in the proposed approach is inspired by Ronneberger's U-Net architecture [13]. This FCN captures both local and global features from the input image to construct an accurate segmentation map. Global features indicate the exact location

and relative size of the nucleus region, whereas the local features determine the exact boundaries. The U-Net architecture follows a two phase stage wise approach. The first phase performs the convolution operation in 4 stages. Every stage has two convolutional layers followed by a downsampling layer with max-pooling operations to garner a larger receptive field. The second phase aims to revert back the activations of the first phase to the original resolution. This phase contains deconvolution layers in 3 stages, finally enabling end-to-end predictions. The weights in the deconvolution layers are also trained, thus they are also termed as learnable upsampling. The advantage of this architecture is that every pixel is considered as an individual training example with per-pixel back-propagation error thus greatly enhancing the size of the training set. For accurate segmentation, there is a need to take into account both local and global features of the input. To combine both of these features, the features extracted from the earlier stages in phase one are forwarded and concatenated with the features in the corresponding upsampled output in phase two.

N**etwork Architecture**

The model is set to work on images of size 256 × 256. In the analysis/downsampling phase, there are two 3 × 3 convolutions before a 2 × 2 max pooling layer which reduces the resolution of the image exactly by half. All the convolutions are followed by a Rectified Linear Unit (ReLU) activation function. The number of filters in each convolutional layer are also doubled after every stage. The second phase of the network upsamples the activations using upconvolution which is basically an upsampling operation of size 2 × 2, followed by a 2 × 2 convolution. The last layer in the network is a 1 × 1 convolution layer with sigmoid activation function which maps the signal to a probability map having the same dimensions as the input image.



3.2.7.1 Architecture of The Proposed Model

Chapter 4

EXPERIMENTS & RESULTS

4.1 Experimental Setup

To evaluate the performance of the proposed method, experiments are conducted with Python3 with the help of certain libraries like Keras, OpenCV, NumPy, Matplotlib. Program is written on Jupyter Notebook with Tensorflow back-end on an HP laptop with 64-bit Ubuntu 16.04 LTS operating system, Intel Core i3-5005U CPU @ 2.00GHz × 4, 8 GB of RAM, 611 GB of disk space.

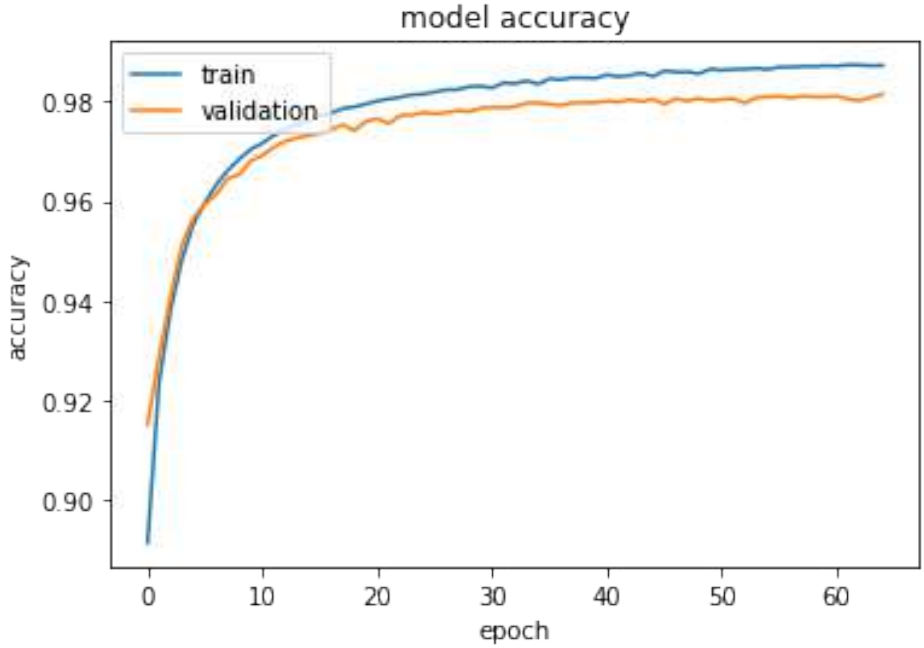
4.2 Hyperparameter Optimization

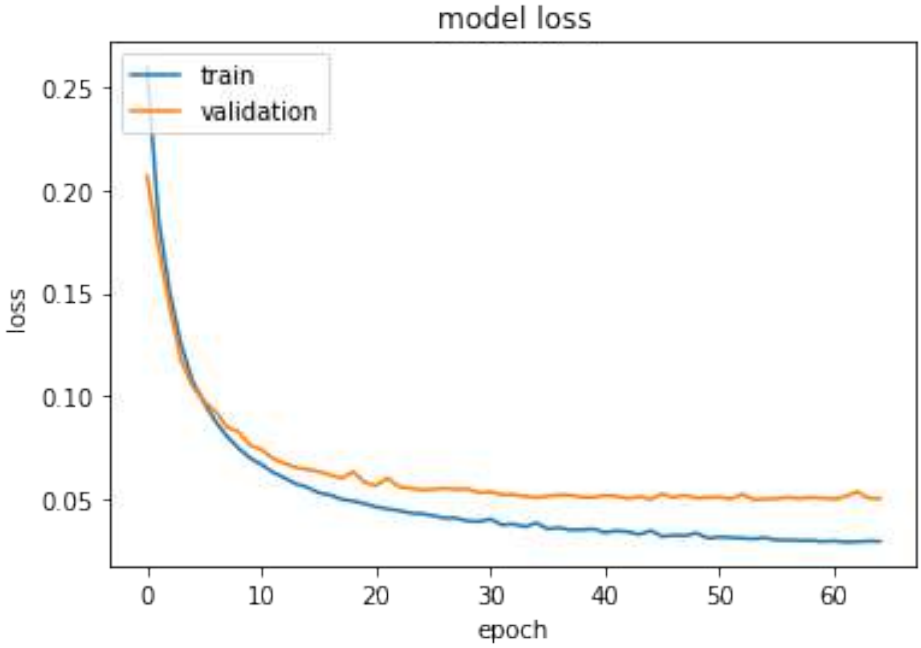
Hyperparameters of a machine learning model are parameters which cannot be learned automatically. They have to be explicitly specified. The process of finding the correct hyperparameters such that the model gives the best result for the current data and problem is known as hyperparameter optimization. In the context of CNNs, the most important hyperparameters are the number of parameters of the network (number of weights and biases), the number of layers, the number of filters/kernels in each layer and the learning rate of the model. Hyperparameters are optimized over a subset of the training set called the validation set. The test set is not used for this purpose as the test set should model the real-world unseen data. Instead of the test set, the validation set is considered a model of the test set for hyperparameter optimization. Several experiments have been performed to validate the architecture described in Section 3.2.7. As described in the architecture proposed by Ronneberger et al. [13], two convolutional layers are always followed by a max-pooling layer or a upconvolution layer. Keeping this feature constant and the number of filters as described, the number of layers can be experimented with and different architectures can be designed. An overview of these architectures are described in Table 1. Here, Depth signifies the

number of max-pooling layers, parameters signify all the weights and biases in the network and layers explicitly specify convolutional layers.

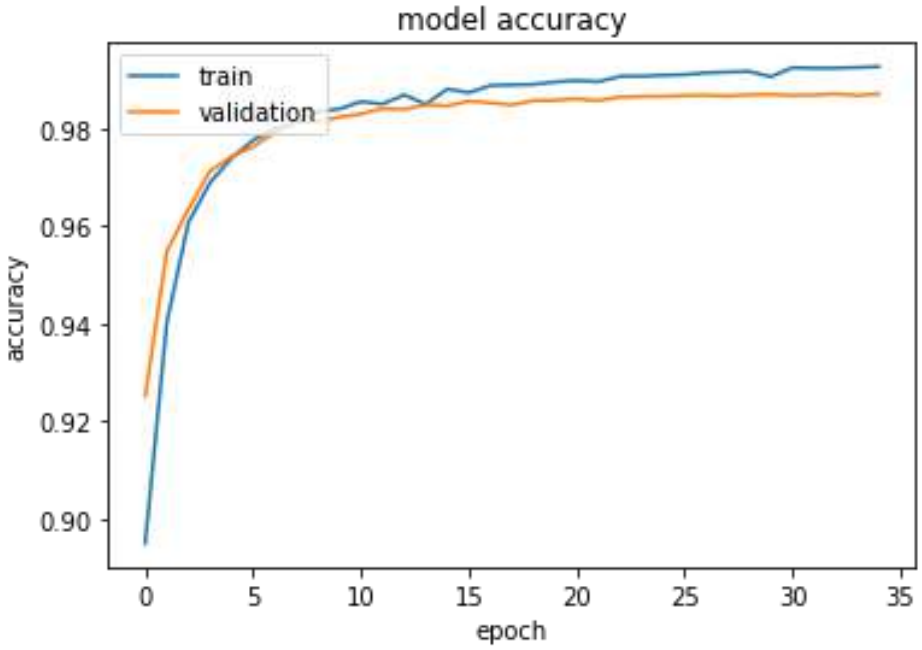
The network is trained with the Binary Cross-Entropy loss function. The Adam optimizer [Kamnitsas et al., 2015], which is a variant of Stochastic Gradient Descent, is used to update the weights. The default hyper parameters are used for this optimizer: β_1 = 0.9, β_2 = 0.999, $\epsilon = 1e-08$. The learning rate is empirically chosen as 1e-4. Initial network weights are assigned according to the He normal initializer. No pretrained weights were used in the process. The network is trained from scratch completely on the train set.

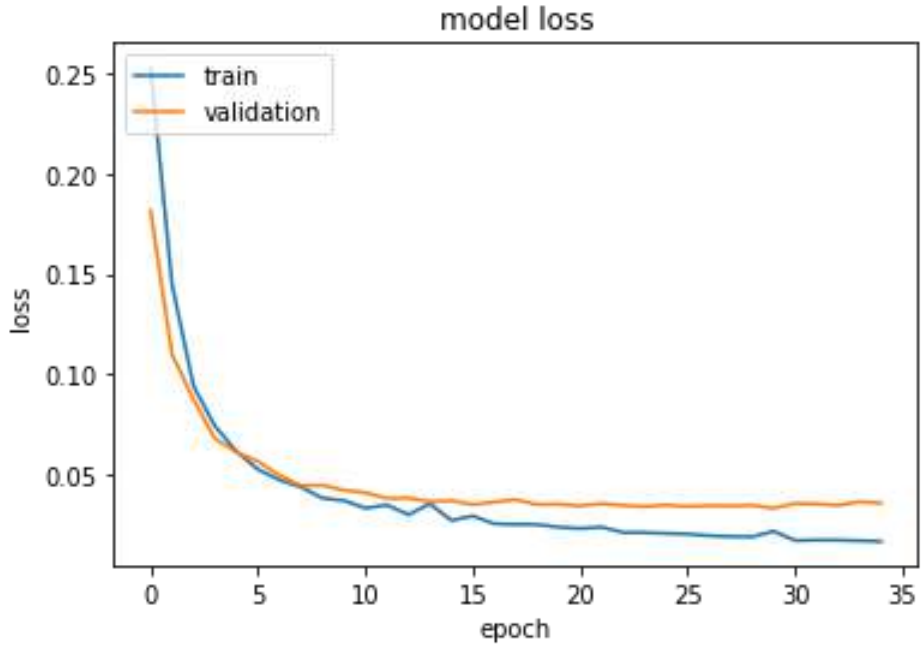
Networks of depth 1 and depth 2 exhibited limited learning capabilities and the training loss did not reduce to an acceptable value. Finally a depth 3 architecture is chosen as discussed in Section 3.2.7. Experiments are done with the size of the starting filter. Two sizes are mainly experimentaed with, 16 and 32. Model with initial filter size 32 produced much more lower training loss than that of 16 in fewer epochs. With starting filter size 16, after 60 epochs, a training loss of 0.02939 is achieved on the homogeneous training set. With starting filter size 32, after only 35 epochs, a much lower training loss of 0.01699 is achieved on the heterogeneous training set, and, a training loss of 0.01857 is achieved on the homogeneous (lung) training set. However, there is a huge difference between the number of learnable parameters for the two different sizes of initial filters. There are 481,745 trainable parameters for 16. For 32, the number is 1,925,025. The model of depth 3 and starting filter size 32 is chosen as the final model of this project.



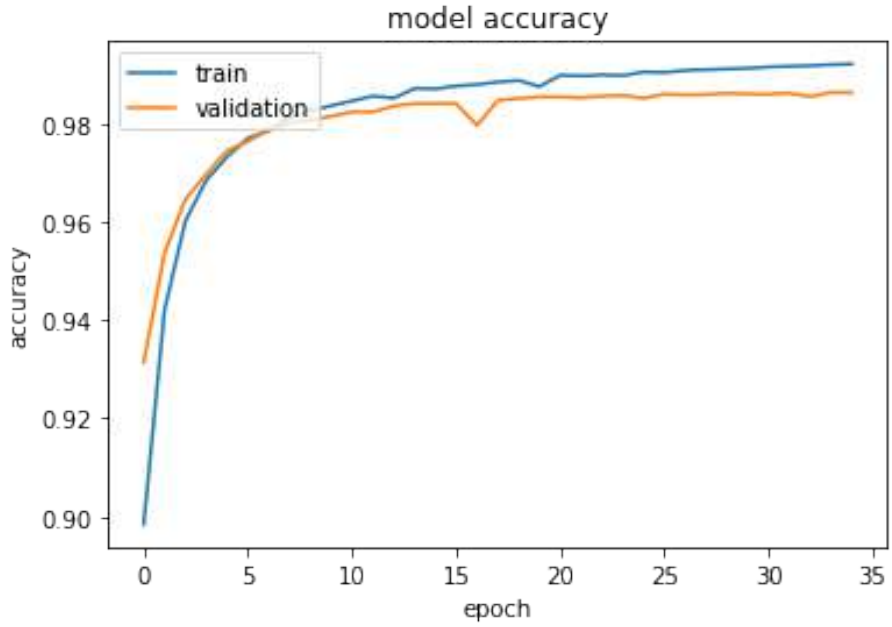


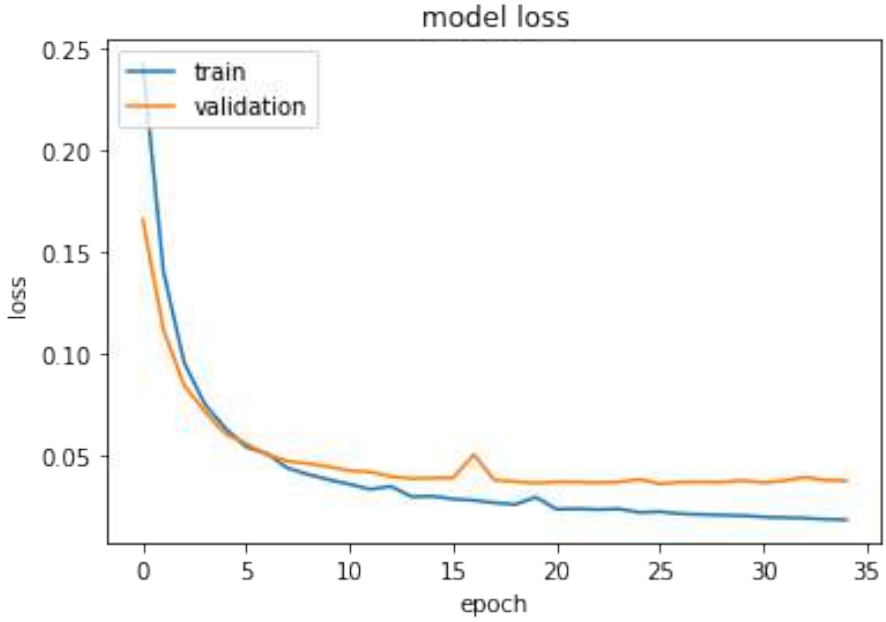
(a) Number of Layers=14, Depth=3, Starting Filter Size=16, Homogeneous (Lung) training set





(b) Number of Layers=14, Depth=3, Starting Filter Size=32, Heterogeneous training set





(c) Number of Layers=14, Depth=3, Starting Filter Size=32, Homogeneous (Lung) training set

4.2.1 Model Accuracy Vs Epoch and Model Loss Vs Epoch Graphs for Several Starting Filter Sizes for Depth Three Architecture

4.3 Results and Analysis

Segmentation results of the proposed method is compared with ground truth provided with dataset images. To compute the shape concordance between a ground truth object G_i and its associated segmented object S_j , Dice's coefficient [18] is often used. Dice's coefficient is a metric which measures the relative area of overlap between the two sets. Dice coefficient is defined as,

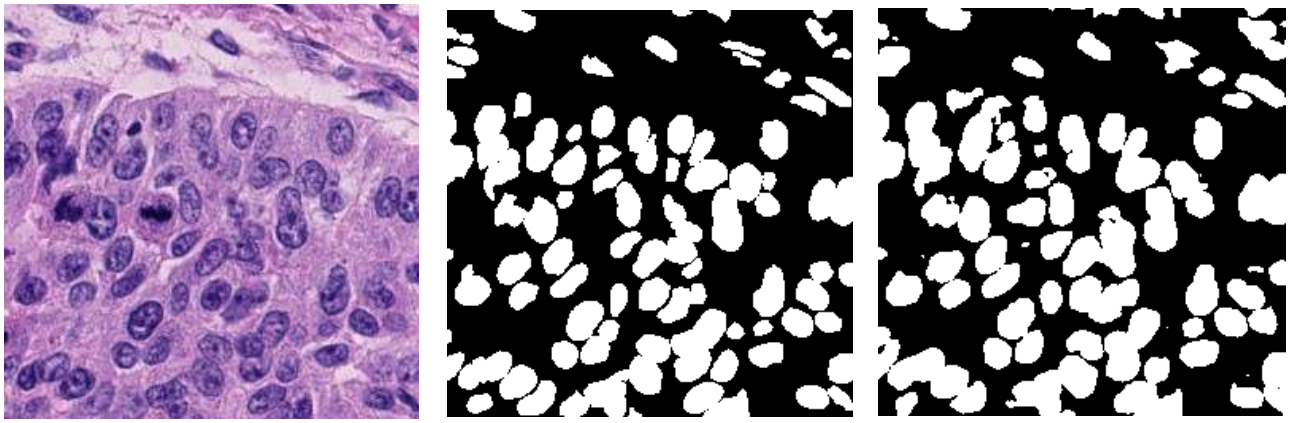
$$\text{Dice Coefficient}\,(G_i, S_j) = 2\,\frac{|G_i \cap S_j|}{|G_i \cup S_j|} \qquad (4.3.1)$$

The Table 4.3.1 below shows the Dice Coefficient values of the proposed method for the heterogeneous training set. The mean value of all the Dice Coefficient values is 0.7900695389088697.

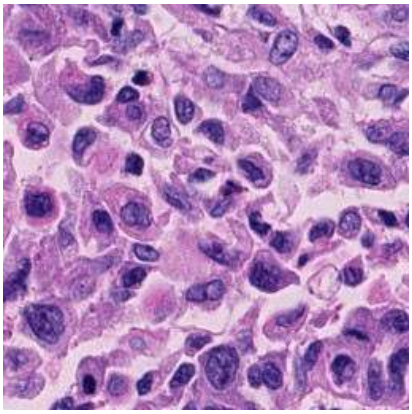
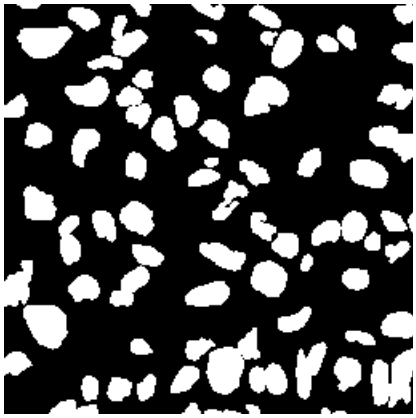
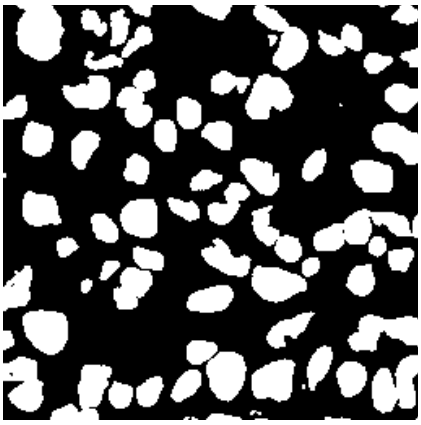
Image	Dice Coeffcient Value
img1	0.866974
img2	0.80075256
img3	0.77516643
img4	0.82509755
img5	0.80043036
img6	0.73734305
img7	0.78873671
img8	0.72605565

4.3.1 Dice Coefficient Value for Images in Heterogeneous Test Set

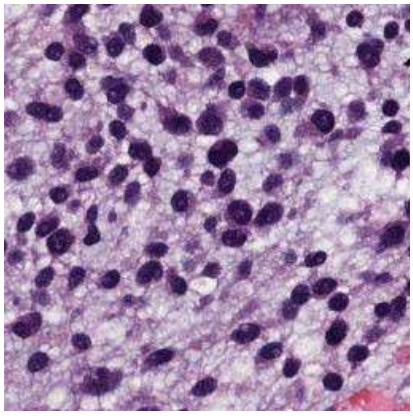
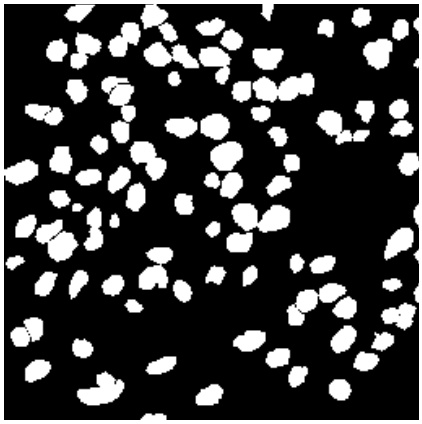
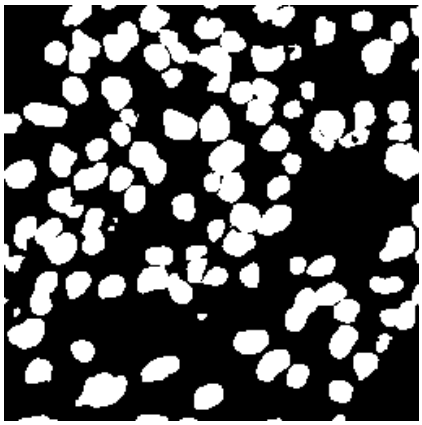
Input Image **Ground Truth** **Segmented Image**



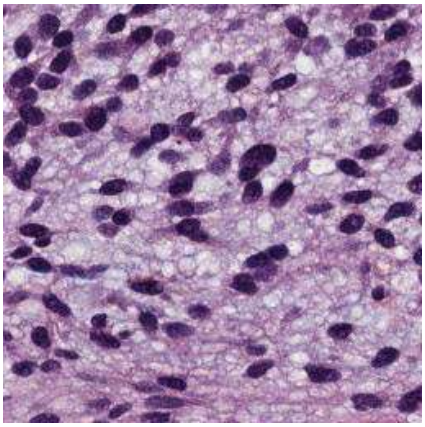
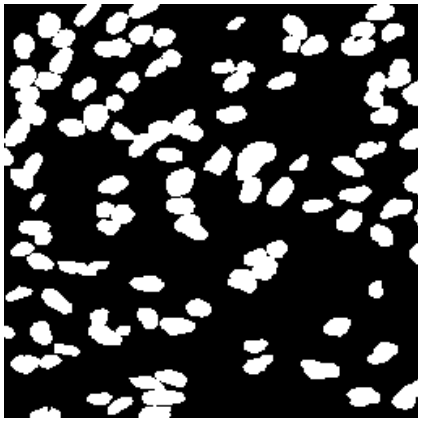
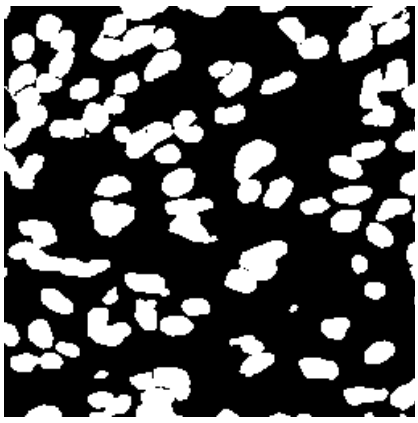
img1 (Lung)

Input Image	Ground Truth	Segmented Image
		

img2 (Lung)

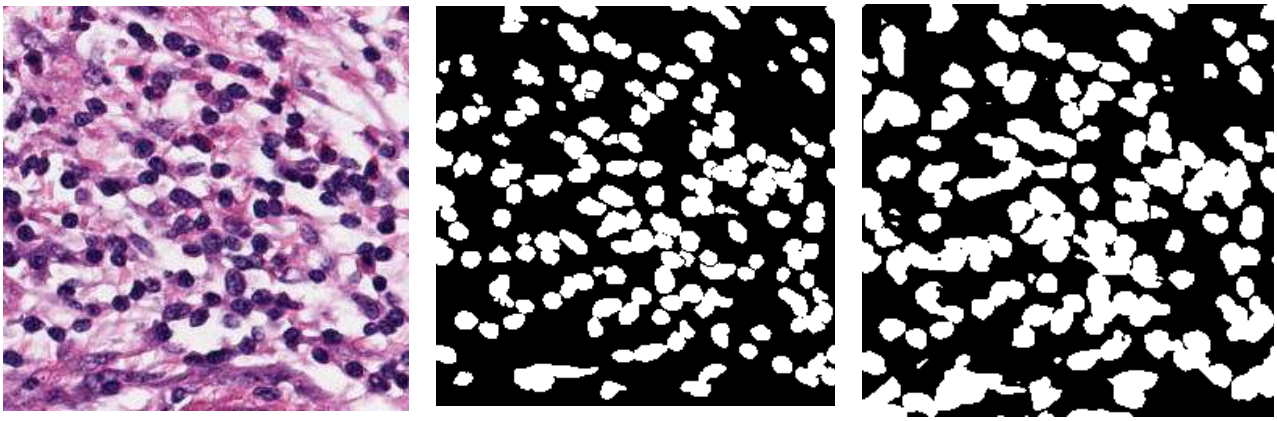
		

img3 (GBM)

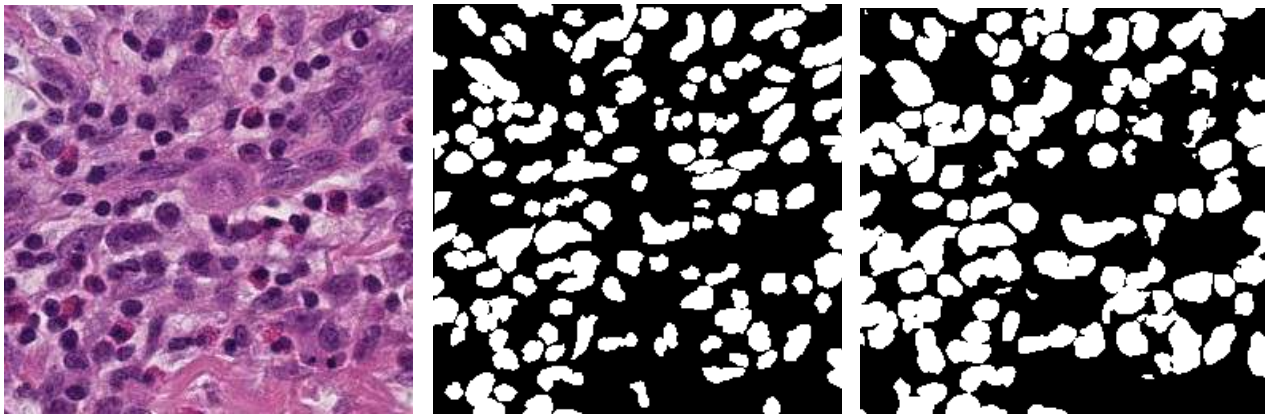
		

img4 (GBM)

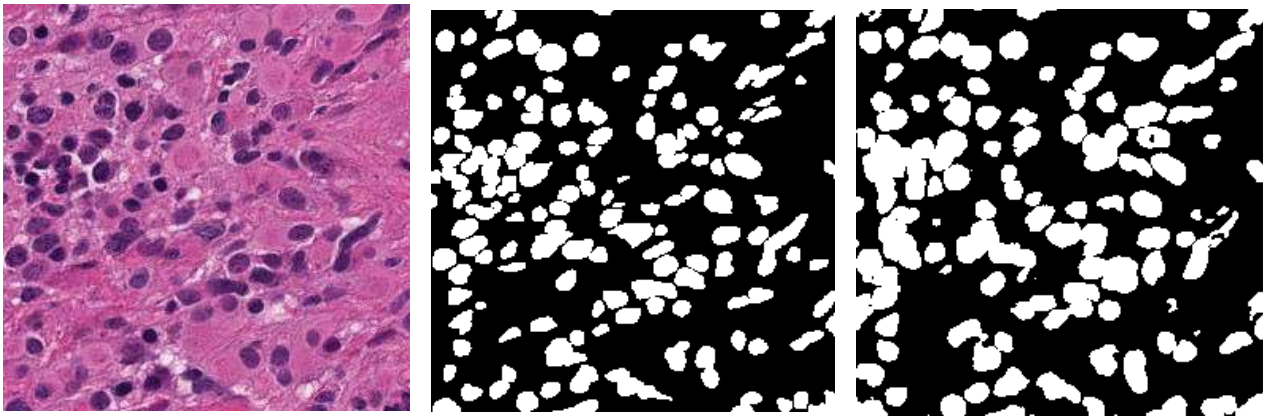
Input Image **Ground Truth** **Segmented Image**



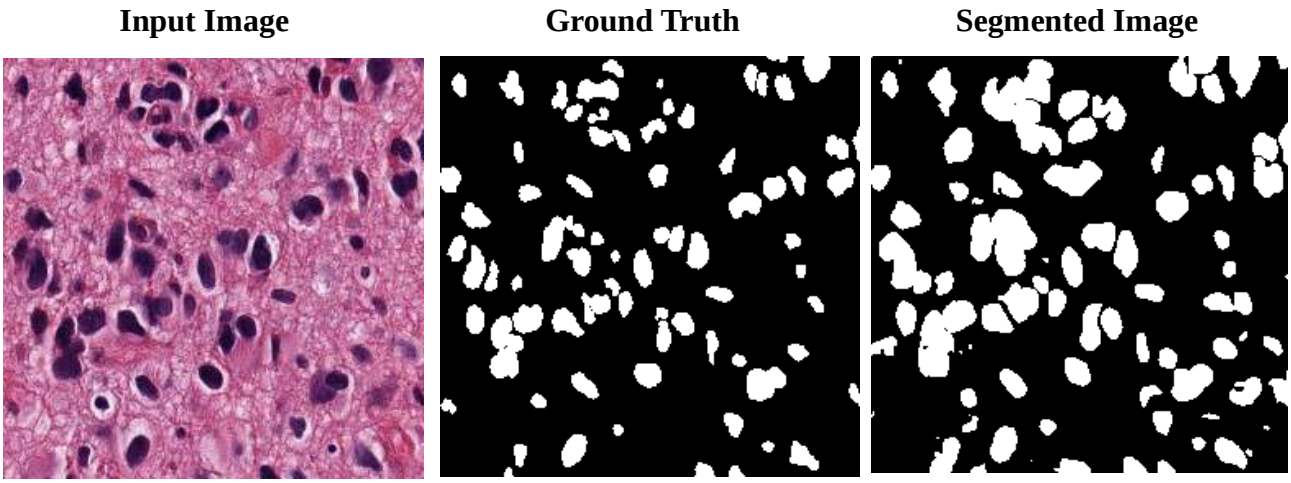
img5 (HNSC)



img6 (HNSC)



img7 (LGG)



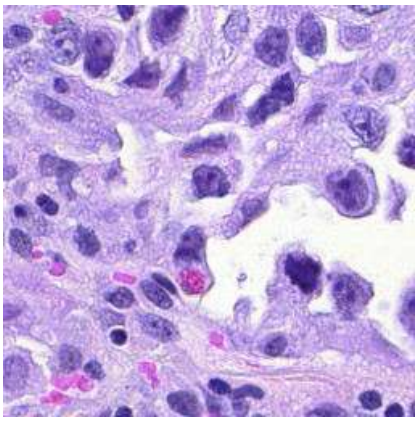
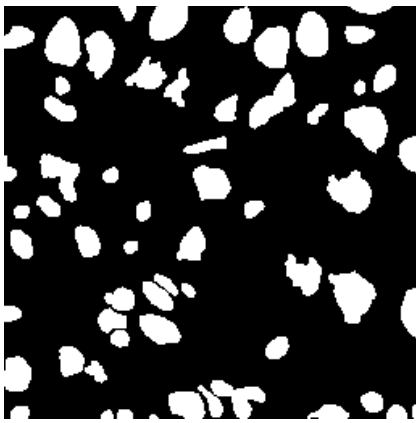
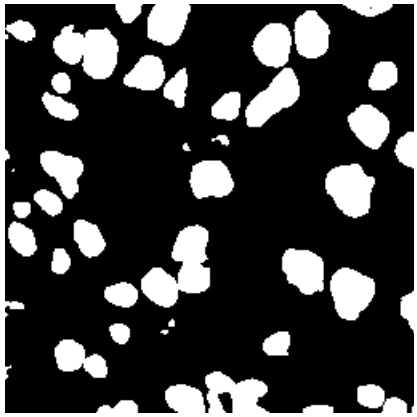
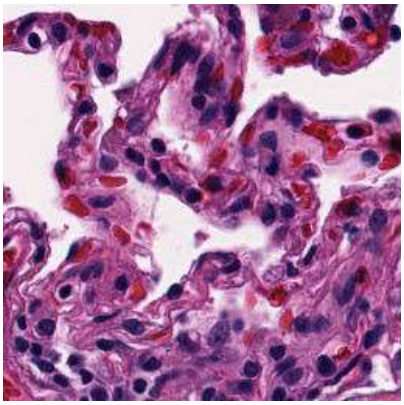
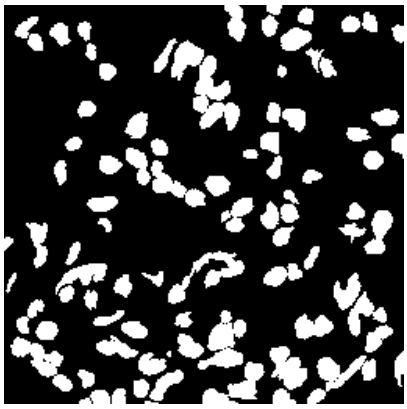
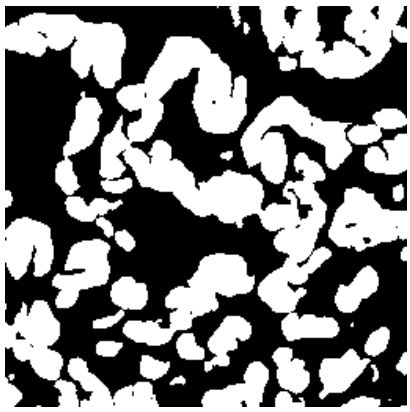
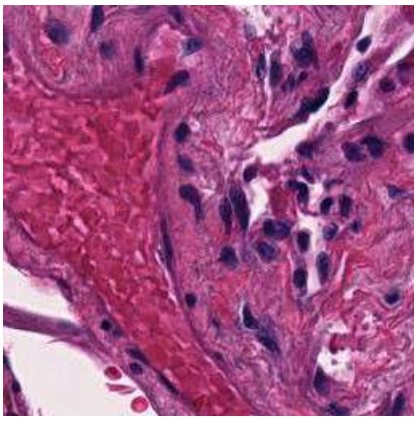
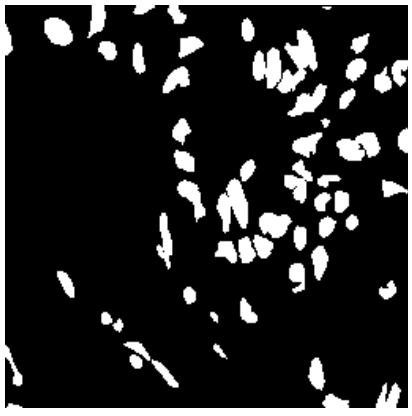
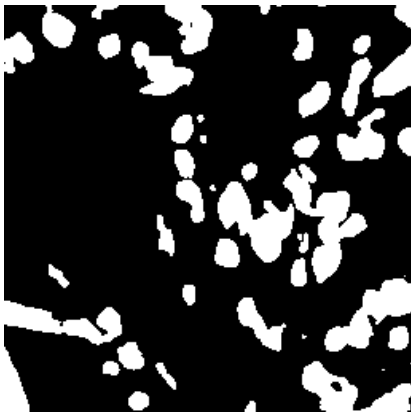
img8 (LGG)

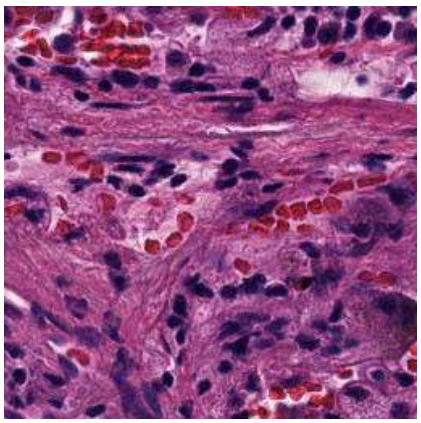
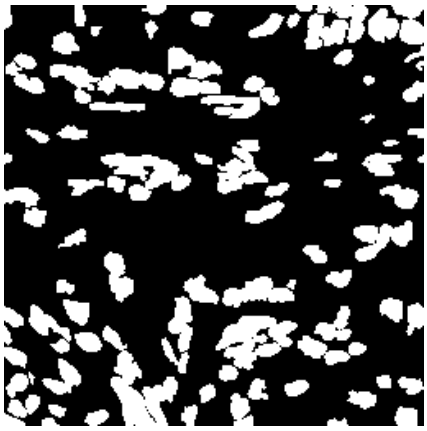
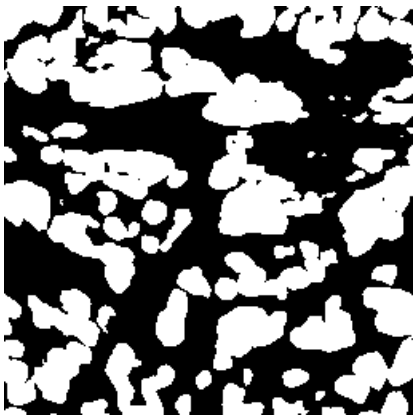
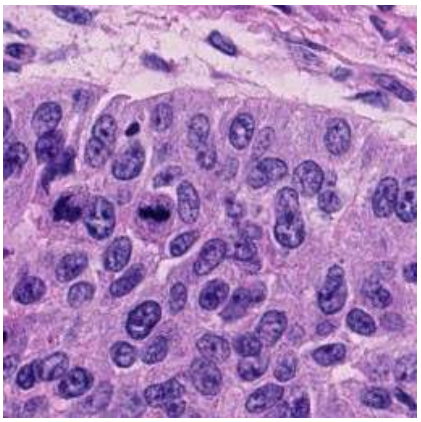
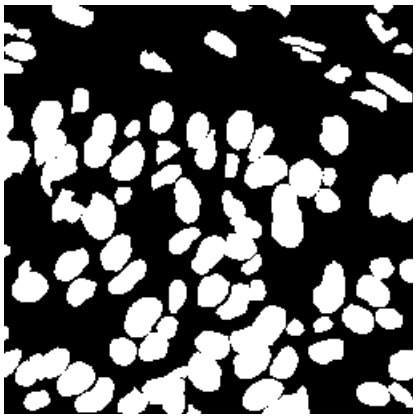
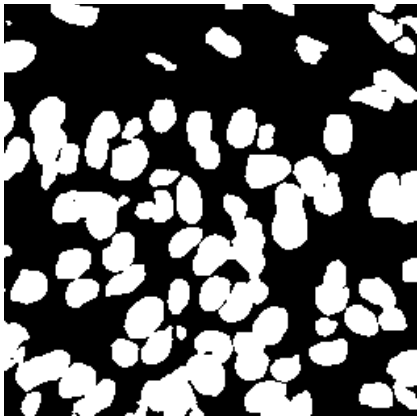
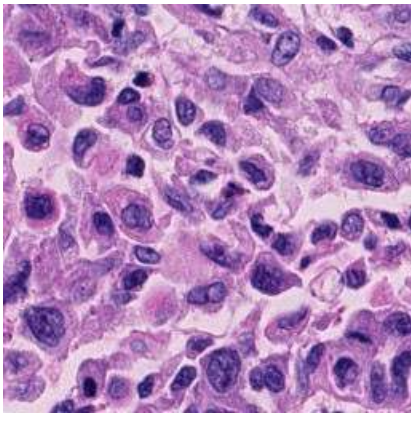
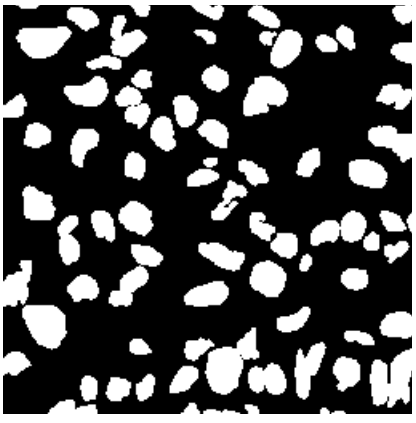
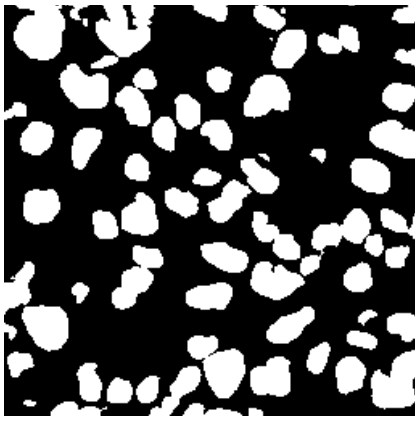
4.3.1 Inputs, Ground Truths & Segmented Images Produced by the Proposed Method

The Table 4.3.2 below shows the Dice Coefficient values of the proposed method for the homogeneous training set (lung). The mean value of all the Dice Coefficient values is 0.6844246242424419.

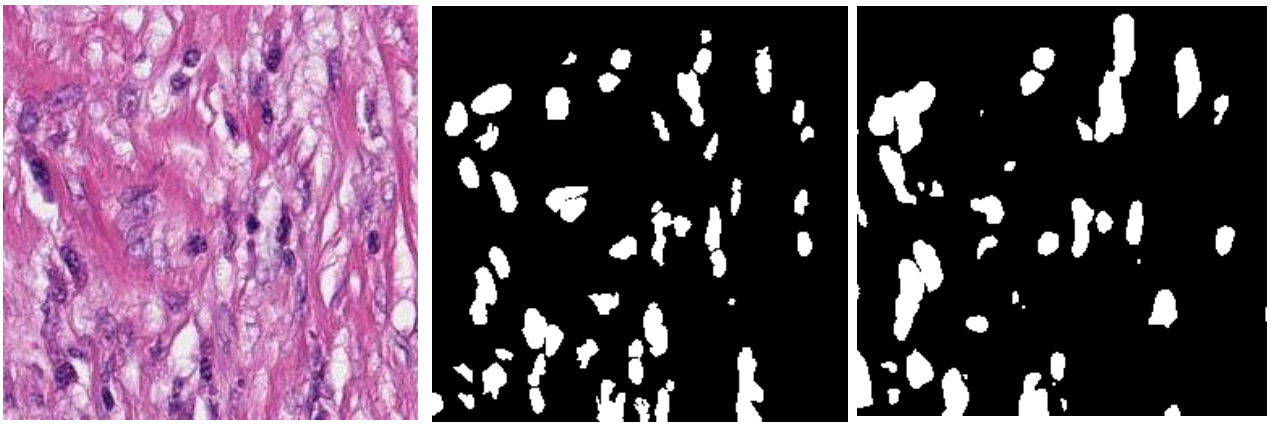
Image	Dice Coeffcient Value
img1	0.75577183
img2	0.56661485
img3	0.54975499
img4	0.62445367
img5	0.86760874
img6	0.79962292
img7	0.56655162
img8	0.74501837

4.3.2 Dice Coefficient Value for Images in Homogeneous Test Set

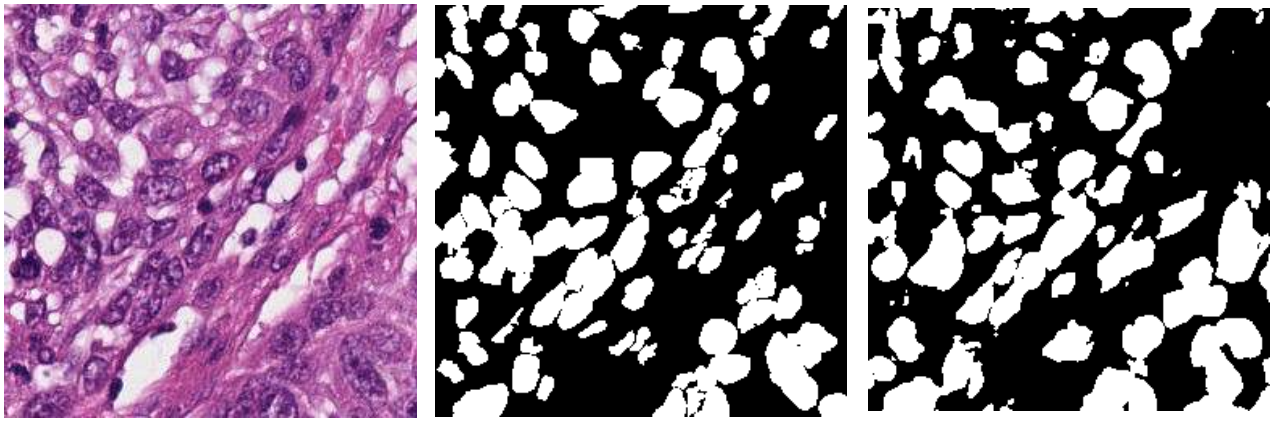
Input Image	Ground Truth	Segmented Image
		
img 1		
		
img2		
		
img3		

Input Image	Ground Truth	Segmented Image
		
img4		
		
img5		
		
img6		

Input Image **Ground Truth** **Segmented Image**



img7



img8

4.3.2 Inputs, Ground Truths & Segmented Images Produced by the Proposed Method on the Lung images

Chapter 5

CONCLUSION & FUTURE WORK

This work presents a CNN based automated nuclear segmentation technique. A diverse dataset of four different types of cancerous human cell images with annotated ground truths collected from a varied set of patients and organs. The technique gives a Dice coefficient value of more than 0.79 on a test set which is evenly distributed by all the four types of cells. It gives a Dice coefficient value of more than 0.68 on a test set consists of only lung cells which is not used to train the CNN.

Proposed method of binary segmentation can be improved by experimenting with newer and more advanced CNN architectures. A deeper model with more number of layers can be constructed if significant amount of training data is present.

This method can further be extended to a ternary segmentation problem to detect nuclear boundary pixels, including those between two touching nuclei, by explicitly introducing a third class of pixels in addition to the usual binary of foreground (inside aany nucleus) and background (outside every nucleus).

REFERENCES

[1] Andrew H. Beck, Ankur R. Sangoi, Samuel Leung, Robert J. Marinelli, Torsten O. Nielsen, Marc J. van de Vijver, Robert B. West, Matt van de Rijn, and Daphne Koller, "Systematic analysis of breast cancer morphology uncovers stromal features associated with survival," *Science Translational Medicine,* vol. 3, no. 108, pp. 108ra113–108ra113, 2011.

[2] H. Chang, J. Han, A. Borowsky, L. Loss, J. W. Gray, P. T. Spellman, and B. Parvin, "Invariant delineation of nuclear architecture in glioblastoma multiforme for clinical and molecular association," *IEEE Transactions on Medical Imaging,* vol. 32, no. 4, pp. 670–682, April 2013.

[3] S. Naik, S. Doyle, S. Agner, A. Madabhushi, M. Feldman, and J. Tomaszewski, "Automated gland and nuclei segmentation for grading of prostate and breast cancer histopathology," *5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro,* May 2008, pp. 284–287.

[4] Mina Khoshdeli, Richard Cong, and Bahram Parvin, "Detection of Nuclei in H&E Stained Sections Using Convolutional Neural Networks", *IEEE EMBS Int Conf Biomed Health Inform*. 2017 February.

[5] Laith Alzubaidi1, Raja Daami Resan, Huda Abdul_hussain, Haider A. Al-Wzwazy, Hayder Albehadili, "A Robust Deep Learning Approachto Detect Nuclei in Histopathological Images", *International Journal of Innovative Research in Computer and Communication Engineering,* Vol. 5, Issue 3, March 2017.

[6] Neeraj Kumar, Ruchika Verma, Ashish Arora, Abhay Kumar, Sanchit Gupta, Amit Sethi, and Peter H. Gann, "Convolutional Neural Networks for Prostate Cancer Recurrence Prediction", 2017.

[7] Eliezer Farrant Braz and Roberto de Alencar Lotufo, "Nuclei Detection Using Deep Learning", *XXXV Simposio Brasileiro de Telecomunicacoes E Processamento de Sinais SBrT2017,* 36 de Setembro de 2017.

[8] F. Xing, Y. Xie, and L. Yang, "An automatic learning-based framework for robust nucleus segmentation," *IEEE Transactions on Medical Imaging,* vol. 35, no. 2, pp. 550–566, Feb 2016.

[9] Kaur Prabhjot, "Mammogram Image Nucleus Segmentation and Classification using Convolution Neural Network Classifier", *International Journal of Advance research , Ideas and Innovations in Technology,* Vol. 2, Issue5, 2 016.

[10] Mohammad Havaei, Francis Dutil, Chris Pal, Hugo Larochelle, Pierre-Marc Jodoin, "A Convolutional Neural Network Approach to Brain Tumor Segmentation", *Multimodal Brain Tumor Image Segmentation Challenge,* 2015.

[11] Angel Cruz-Roa, Ajay Basavanhally, Fabio Gonz_alez, Hannah Gilmore, Michael Feldman, Shridar Ganesan, Natalie Shih, John Tomaszewski and Anant Madabhushi, “Automatic detection of invasive ductal carcinoma in whole slide images with Convolutional Neural Networks”, *Medical Imaging 2014: Digital Pathology,* 2014.

[12] Neeraj Kumar, Ruchika Verma, Sanuj Sharma, Surabhi Bhargava, Abhishek Vahadane, and Amit Sethi, “A Dataset and a Technique for Generalized Nuclear Segmentation for Computational Pathology”, *IEEE Transactions On Medical Imaging,* February, 2017.

[13] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation”, *International Conference on Medical Image Computing and Computer-Assisted Intervention,* pp. 234–241, Springer, May, 2015.

[14] Goodfellow, I., Bengio, Y., and Courville, A., “Deep learning”, *MIT Press, 2016.*

[15] Li, F. F., Karpathy, A., and Johnson, J., “Cs231n: Convolutional neural networks for visual recognition”, *URL: http://cs231n.github.io.*

[16] Nielsen, M. A., “Neural networks and deep learning”, *URL: http://neuralnetworksanddeeplear -ning.com/,* 2015.

[17] Long, J., Shelhamer, E., and Darrell, T., “Fully convolutional networks for semantic segmentation”, *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* pages 3431–3440, 2015.

[18] L. R. Dice, “Measures of the amount of ecologic association between species,” *Ecology,* vol. 26, no. 3, pp. 297–302, 1945.