Neural Network for Segmentation of Medical Images

by

Asif Ahmad

Supervised by

Dr. Noor Badshah

A thesis
submitted to the University of Engineering and Technology
in partial fulfillment for the degree requirement of
Master of Science
in
Mathematics

Session: 2018-2020

Peshawar, Khyber Pakhtunkhwa, Pakistan.

Dedication

To my parents, Mr. and Mrs. Gulsambar Khan, and my younger brother, Hanif Ahmad.

Acknowledgement

Almighty ALLAH has been very kind and has bestowed His blessings on me at every moment of my life. I am very much blessed to have parents whose quest for education and selfless prayers have brought me to where I am today; I owe my life to them. I would like to express my special thanks of gratitude to my supervisor, Dr Noor Badshah, for his invaluable support and guidance during my coursework and research at UET Peshawar; and specially for giving me the golden opportunity to work on this wonderful project. My heartfelt thanks to Dr Nasru Minallah, associate professor at computer system engineering department UET Peshawar, for giving an opportunity to work in a project during my coursework and for his financial assistance for the project. I am utterly thankful to Dr Nudrat Aamir, HOD Basic Sciences CECOS University of IT & Emerging Sciences Peshawar, for her support which helped me a lot in doing my research work with full concentration. I am deeply indebted to Mr. Irfan Ul Mulk for all his generous and fatherly support in the darkest period of my life. I would also like to thank my colleagues, Dr Asmat Ullah, Mr. Fahim Ullah (Ph.D), Mr. Mati Ullah (P.hD), Ms. Nasra Nadeem (P.hD), Ms. Hadia Atta (P.hD), Ms. Hena Rabbani (P.hD), Ms. Muniba Ashfaq (Ph.D) and Mahmood Ul Hassan (MS), at UET Peshawar with whom I shared my study time and came to know about many new things through useful discussions. At last, but not the least, I am immensely grateful to my friends, especially Abdur Raqeeb and Mujeeb Ullah, sisters, Samreena & Shehla, younger brothers and fiancee, Shaista Rahim, for their prayers, encouragement and love.

Abstract

The success of Deep Neural Networks (DNNs) is largely impacted by training of large networks on big data sets. The Greatest breakthrough was achieved in this field in 2012 by successful training of a large network, AlexNet, on ImageNet data set containing about 1 million images. U-Net architecture, a DNN based on Convolutional Neural Networks (CNNs), immensely advanced segmentation of medical images. In the present times, neural networks have outperformed other state-of-the-art approaches in segmentation of images. In this thesis, we present a neural network based on the CNNs for segmentation of medical images. The network, ResBCU-Net, is an extension of the U-Net which utilizes Residual blocks, Batch normalization and Bi-directional ConvLSTM. In addition, we present an extended form of ResBCU-Net, ResBCU-Net(d=3), which utilizes densely connected layers in its bottleneck section. The proposed neural network is trained and evaluated on ISIC 2018 data set, which is publicly available data set containing 2594 skin images, melanoma and non-melanoma. The network inferences segmentation of the images more accurately than other state-of-the-art alternatives.

Contents

List of Figures

List of Tables

List of Acronyms

DNN	Deep Neural Network.
CNN	Convolution Neural Network.
RNN	Recurrent Neural Network.
CAD	Computer Aided Diagnosis.
BN	Batch Normalization.
ReLU	Rectified Linear Unit.
LSTM	Long Short Term Memory.
ISIC	International Skin Imaging Collaboration.
GPU	Graphics Processing Unit.

Chapter 1

Introduction

Analysis of medical images is a vital part of diagnosis and treatment of diseases. Manual diagnosis of diseases depends on availability of advance appliances and high skilled doctors. Besides, it is costly, erroneous and tedious process. A computer aided diagnosis (CAD) system can be good choice here, as it can assist doctors in better diagnosis and treatment of diseases in large number of people in short time in a better way. CAD uses images processing techniques for analysis of medical images. Segmentation has a great value in the images processing. It divides the images into affected and unaffected region. The diagnostic process results are highly dependent on the accuracy of performed segmentation. Nowadays, many supervised and unsupervised techniques are used for the task of segmentation. Supervised techniques use active contours [1, 2], fuzzy sets [3, 4] and machine learning algorithms like k-mean clustering [5], morphological operations [6], etc. While, unsupervised techniques come under deep learning. Deep learning uses convolutional neural networks (ConvNet/CNNs)[7, 8, 9]. Over the couple of years CNNs have outperformed other approaches in this field.

Convolutional Neural Networks (ConvNet/CNNs) are one of the approaches in the field of computer vision, where the purpose is to enable machines to replicate function of humans' brain. CNNs have been designed for multiple tasks, like, image and video recognition, image classification and segmentation, detection, and solving inverse problem in image processing, imitating humans ability of perceiving them in a natural way. Architecture of CNN consists of layers of convolutions, max poolings, and activation functions, which mimic the pattern of neurons in a human brain.

Convolutional neural networks (CNNs) were being used long before 1990s [10], but they were limited due to size of then available networks and training data. The first breakthrough achieved by krizhevsky et al. [11] in 2012, named as AlexNet, using training of large network consists of 18 layers and millions of parameters on the imageNet data and 1 million of training images. After this a large and much deeper networks have been trained [12]. Since 2010 deep convolutional networks have outperformed the conventional state-of-the-art approaches towards visual recognition tasks, like, [13] and [11]. At first CNNs were used for classification of images. Ciresan et al. [14] presented a network for automatic segmentation of electron microscopy images. For the purpose, the authors have used CNN as a pixel classifier. A square window, centered at each pixel, is used for the prediction of the label of each pixel. Each window pixel is mapped into a neuron, which is followed by convolution and max-pooling layers.

A major breakthrough was achieved in this field in 2015 by U-Net architecture [7], an encoder-decoder based neural network proposed by Ronneberger et al. for segmentation of biomedical images. The network consists of two symmetric paths, encoding and decoding path. Convolutions, activation functions and max pooling operations are being used in the network. Nowadays, almost all deep neural networks presented for segmentation task of medical images are based on the U-Net and they have outperformed other state-of-the-art networks for the task [15, 8, 16, 9].

In this work, we propose a U-Net [7] based encoder-decoder neural network for segmentation of skin lesion. The network, ResBCU-Net, utilizes power of residual blocks [17], batch normalization [18] and BConvLSTM [19] network for the task. In the network, we enhance the encoding path with residual blocks and batch normalization, and the decoding path is enhanced by using BConvLSTM network in the path. In addition, we also present a densely connected form of ResBCU-Net, where we have made changes in the bottleneck section of our model like BCDU-Net [8]. More details of our work are given in chapter 4.

Thesis Outline

Other chapters of the thesis are arranged as follows:

Chapter 2: This chapter contains some basic terms which are used as building blocks for neural networks.

Chapter 3: In this chapter, we give an insight into background of neural networks and discuss related works in the literature.

Chapter 4: In this chapter, we explain our work in detail.

Chapter 5: This chapter concludes our work with a conclusion note.

Chapter 2

Basic Terms

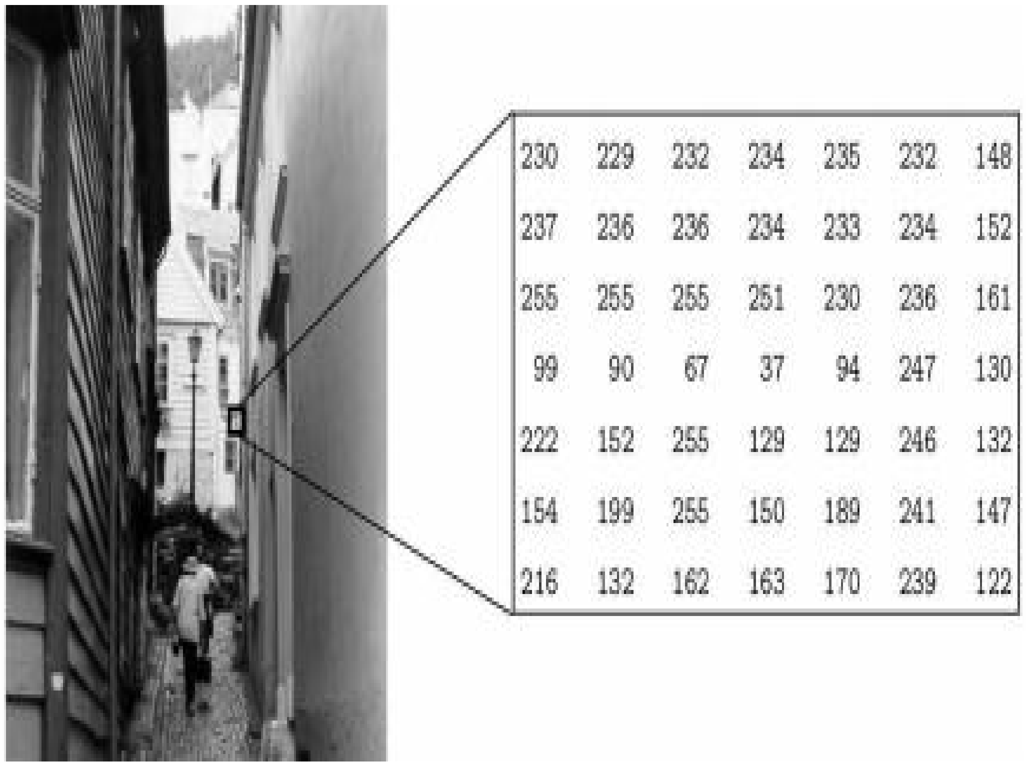
This chapter provides an insight into some basic terms which are very useful to understand the work.

2.1 Digital Image

An image may be defined as a two dimensional function, $f(x, y)$, where x and y are spatial co-ordinates and the amplitude of f at any co-ordinate (x, y) is called intensity or gray level of the image at that point. An image is said to be a digital image when x, y, and the amplitude of f are all finite and discrete quantities. In simple words, a digital image is composed of finite numbers, called elements or objects, arranged in rows or columns. The elements are referred to as picture elements or pixels. The pixel values lie between 0 and 255, these values represent the intensity value of black and white, respectively. According to these values digital images are classified into three main types, Grayscale image, Binary image, and RGB image. In this thesis, the word 'Image' refers to 'Digital image'.

2.1.1 Grayscale Image

An image is said to be a grayscale image if the range of the pixel value is 0 to 255, where black parts of the image are assigned 0's, and white parts of the image are assigned 255's while colors between black and white are assigned values between 0 and 255. An example of grayscale image is given in figure below 2.1. In more concise way, grayscale image is a matrix where the element of the matrix are numbers from 0 to 255.

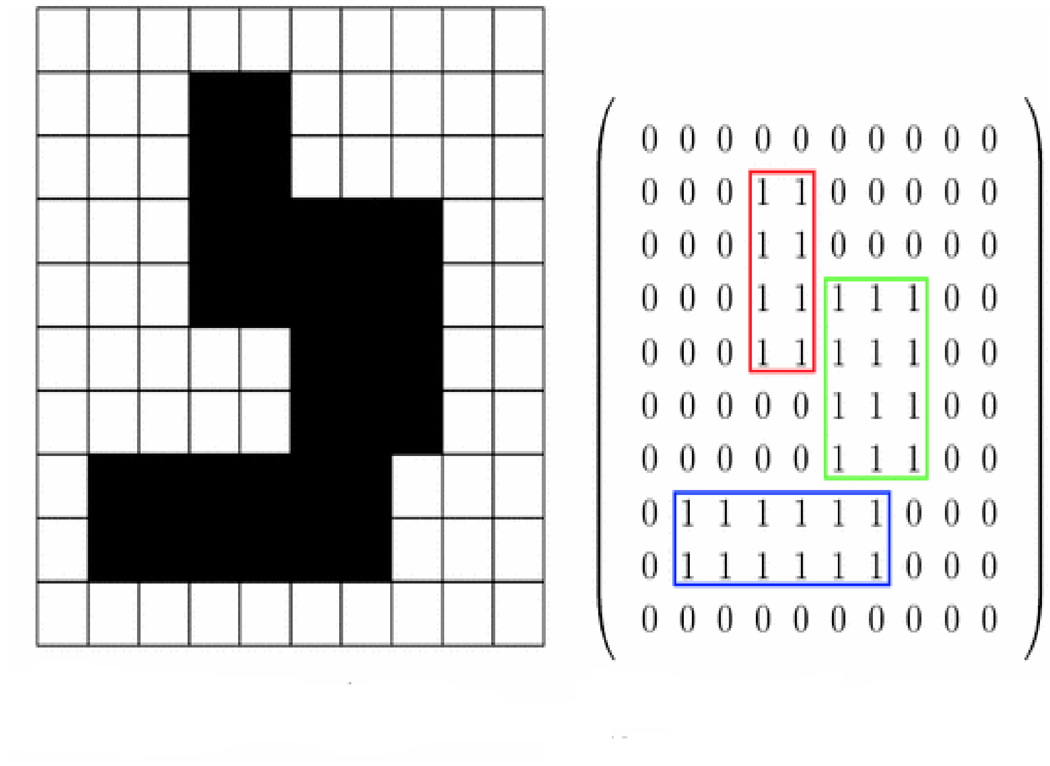


230	229	232	234	235	232	148
237	236	236	234	233	234	152
255	255	255	251	230	236	161
99	90	67	37	94	247	130
222	152	255	129	129	246	132
154	199	255	150	189	241	147
216	132	162	163	170	239	122

Figure 2.1: Grayscale image and its one minor part representation as rows and columns of numbers.

2.1.2 Binary Image

Binary image is an image composed of white and black colors only. Where 0's are assigned to black part and 1's are assigned to white part of the image but some assigns 1's to black part and 0's to white part, as shown in figure 2.2. Binary image is a matrix where the elements are only 0's and 1's.



$$\begin{pmatrix}
0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&0&0&0&0&0\\
0&0&0&1&1&0&0&0&0&0\\
0&0&0&1&1&1&1&1&0&0\\
0&0&0&1&1&1&1&1&0&0\\
0&0&0&0&0&1&1&1&0&0\\
0&0&0&0&0&1&1&1&0&0\\
0&1&1&1&1&1&1&0&0&0\\
0&1&1&1&1&1&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0
\end{pmatrix}$$

Figure 2.2: Binary image and its matrix form.

2.1.3 RGB Image

An RGB image is composed of three matrices which are stacked on one another, as shown in figure 2.3. The matrices are termed as Red (R), Green (G), and Blue (B). Each matrix elements range lie from 0 to 255.

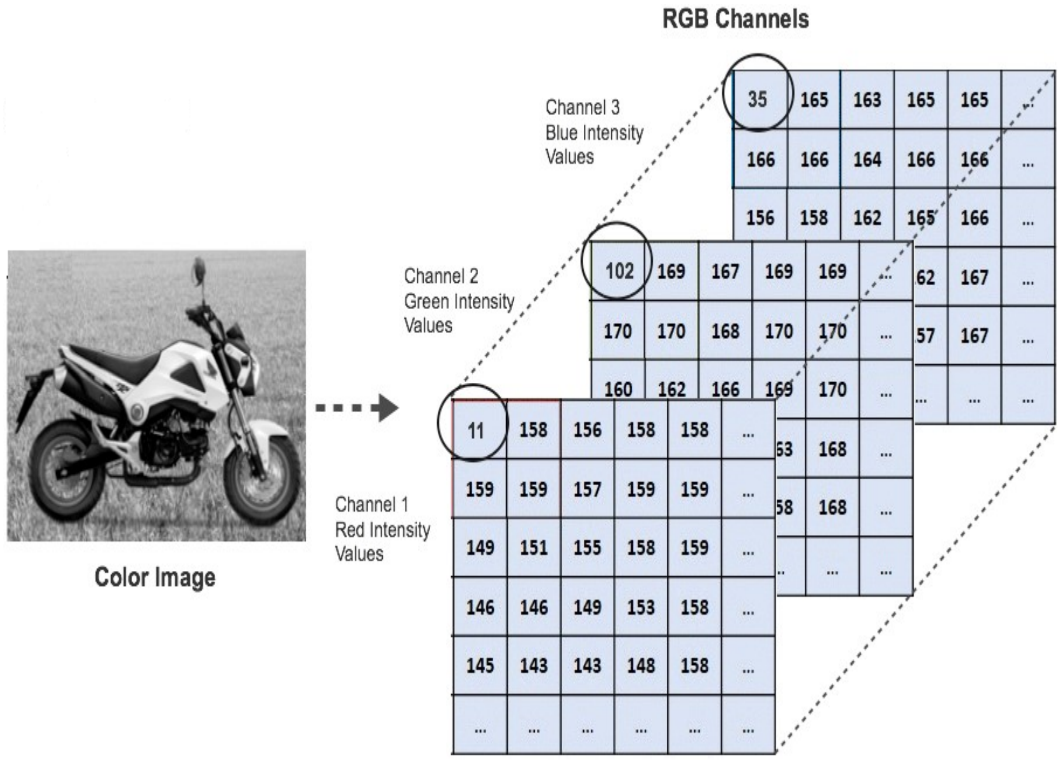


Figure 2.3: RGB image channel wise representation.

2.2 Image Processing

Image processing is a procedure of manipulating an image in order to enhance it or to extract some information from it. In the process, the input is an image and the output may be an image or some features and characteristics associated with the image. There are different forms of image processing, like, image classification, object detection, registration, pattern recognition, segmentation, etc.

2.2.1 Segmentation

In image processing, segmentation is a process of partitioning or dividing an image into parts which are called segments. Segmentation highlights or enhances a target in an image. An image is transformed into black and white parts, where the white part represents the targeted region of the image, while the black part is considered as a background of the target, as shown in the figure 2.4.

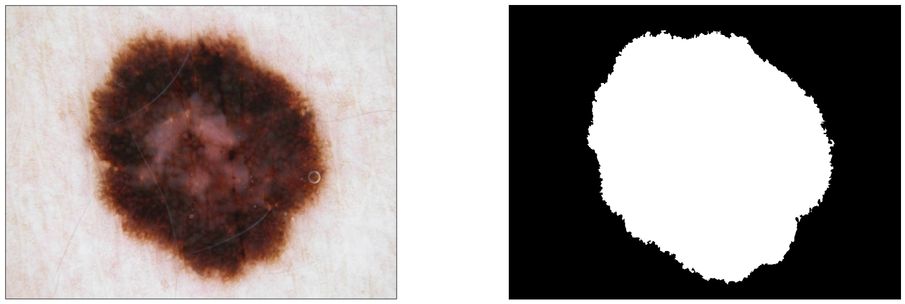


Figure 2.4: (a): An image, (b): Segmented version of the image.

Segmentation's goal is to simplify or change representation of an image into more meaningful and easier form to analyze. For segmentation of images, there are different approaches, supervised and unsupervised. Supervised techniques use active contours [1, 2], fuzzy sets [3, 4] and machine learning algorithms like k-mean clustering [5], morphological operations [6], etc. While, unsupervised techniques come under deep learning. Deep learning uses neural networks [7, 8, 9]. Over the couple of years neural networks have outperformed other approaches in this field.

2.3 Neural Network

Deep neural network or simply a neural network is a series of algorithms that attempts to recognize underlying relationships in a set of data through a process that imitates the way the human brain works. We can say, an artificial neural network which via an algorithm allows computers to learn by incorporating new data.

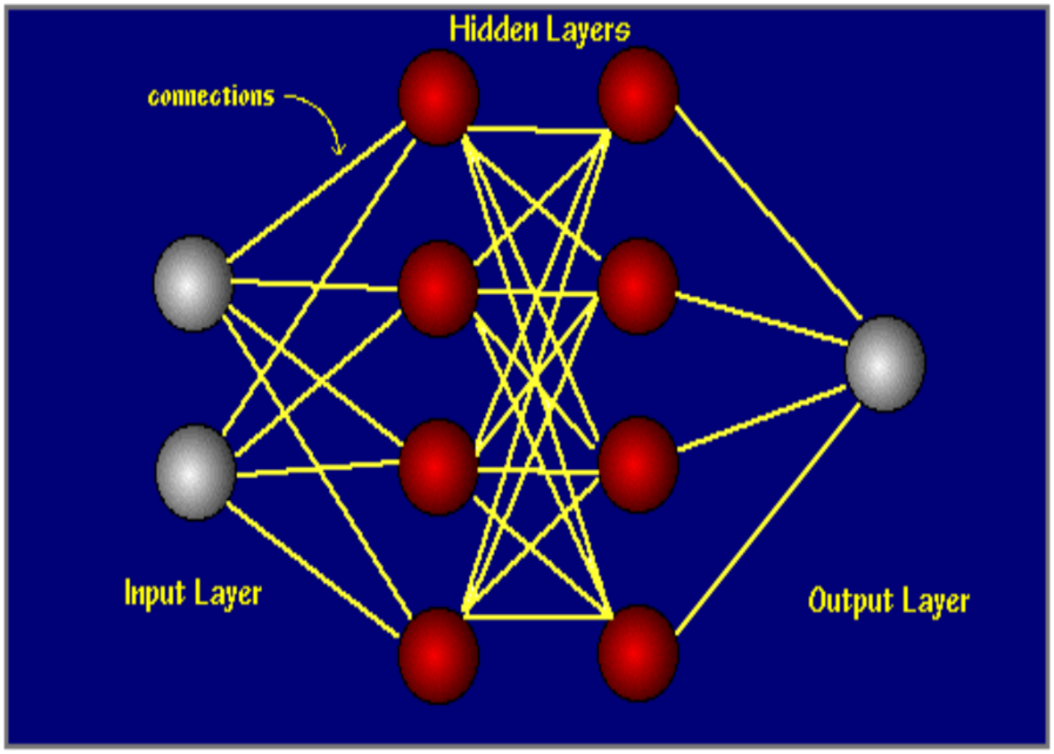


Figure 2.5: A simple neural network. Here, the red circular objects represent neurons in the hidden layers connected to the input via weights (connections)[1].

In simple words, neural network refers to a system of neurons, artificial in nature. There are different neural networks for different tasks, structure of a simple neural network is shown in figure 2.5. Among them Convolutional neural networks (CNNs) are used and considered best for segmentation task. Convolutional neural networks (CNNs), shown in figure 2.6, are neural networks which are developed by various layers of convolutional operations, pooling operations and activation functions.

[1]https://www.massey.ac.nz/~wwpapajl/evolution/assign2/MVD/Indexpg.html

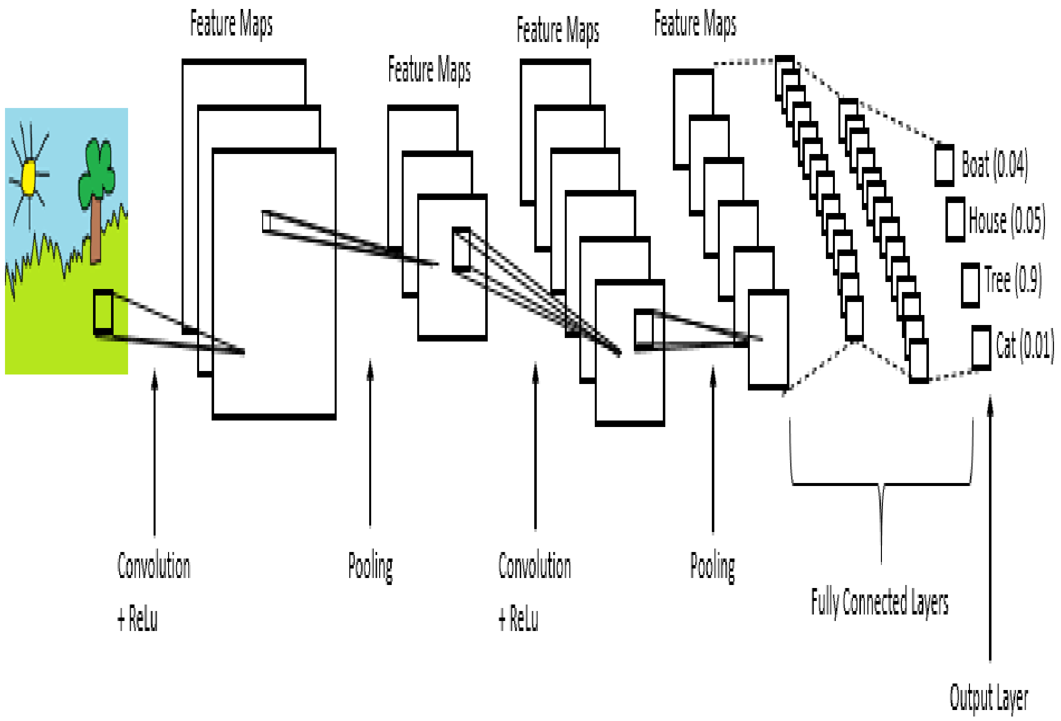


Figure 2.6: A convolutional neural network for classification task[2].

The working mechanism of a CNN can be represented in equations as: If x represents an input, f represents activation function in a layer, W^n represents a kernel/filter, used to convolve over an image in a convolution operation, in an nth layer, b^n represents bias term added in the nth layer, z^n represents an output of a convolution operation with bias term in an nth layer, and a^n represents an output of the f in an nth layer, then equations for n number of layers can be:

$$z^1 = W^1 x + b^1$$

$$a^1 = f(z^1)$$

$$a^1 = f(W^1 x + b^1)$$

[2]shorturl.at/IMN57

$$z^2 = W^2 f(W^1 x + b^1) + b^2$$

$$a^2 = f(W^2 f(W^1 x + b^1) + b^2)$$

Similarly, third layer activation function output can be represented by:

$$a^3 = f(W^3 f(W^2 f(W^1 x + b^1) + b^2) + b^3)$$

Likewise, nth layer activation function output can be represented as:

$$a^n = f(W^n f(W^{n-1} \ldots f(W^1 x + b^1) + \ldots + b^{n-1}) + b^n)$$

Convolutional neural network (CNN) based neural architectures are built up of convolution operation, pooling operation, activation function, etc. Below are some of the building blocks of CNNs.

2.3.1 Convolution

Convolution is a mathematical operation where we take a matrix, usually 3x3 square known as kernel/filter, and pass it over an image. Which in resultant transforms the image into another image known as feature map. The feature map is dependent on the elements/values of the filter. If an image I and kernel K are used and the position of rows and columns of the feature map are denoted by m and n, respectively, then the convolution operation of I and K can be represented as $I * K$ and the feature map G is given as:

$$G[m,n] = (I * K)[m,n] = \sum_i \sum_j K(i,j) I[m-i, n-j].$$

After placing the kernel over a selected region of the image, each value from the kernel is multiplied with the corresponding values in the region of the image. After multiplication of each corresponding element, all the results are summed up and put in the specified place of the feature map as shown in figure 2.7 below.

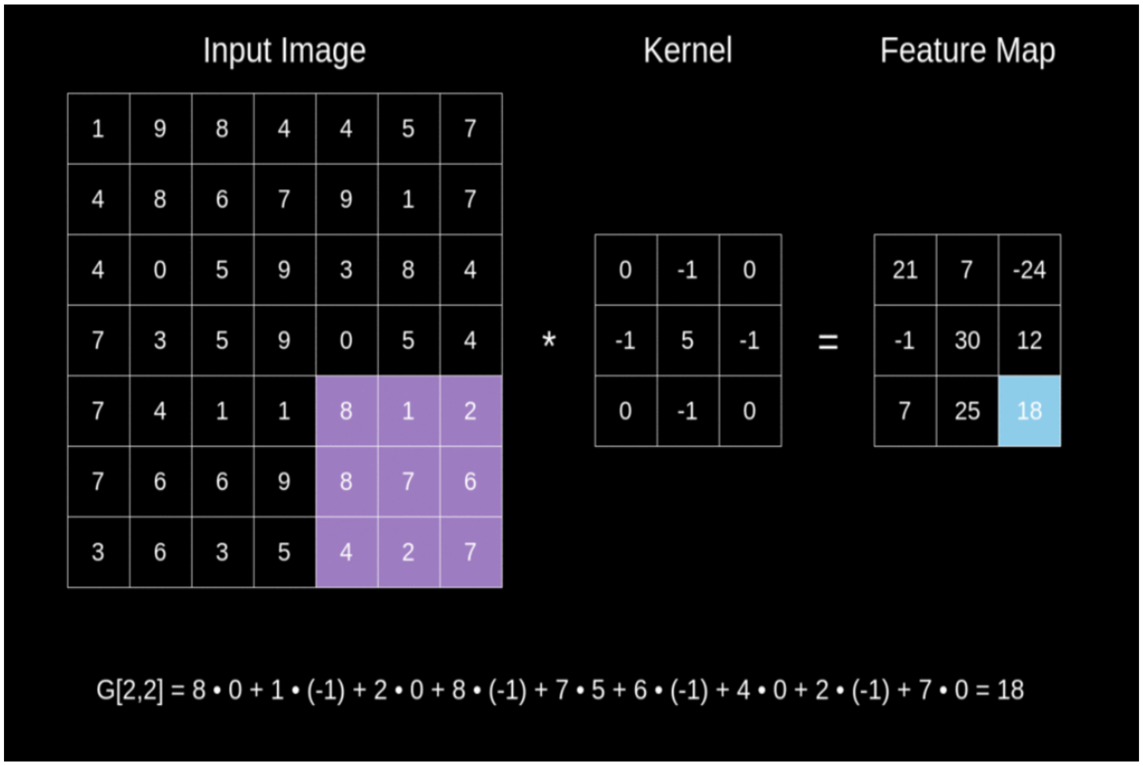


Figure 2.7: Convolution operation visualization[3].

Different kernels do different jobs i.e. some are used for sharpening of images, some are for blurring and some are used for edge detection.

2.3.2 Pooling

Pooling operations are used in convolutional neural networks to reduce size of a feature map and speed up calculations. Besides this use, pooling operations are used as regularizers to avoid overfitting. Unlike convolution operations, these use window which pass over an image. There are basically two types of pooling operations: Max pooling and Average pooling.

Max Pooling

In max pooling operation a window, usually 2x2, is used and pass over an image. The greatest value of the feature map among the values is selected which come under the window as shown in the figure 2.8 below.

[3]https://rb.gy/0wxnpb

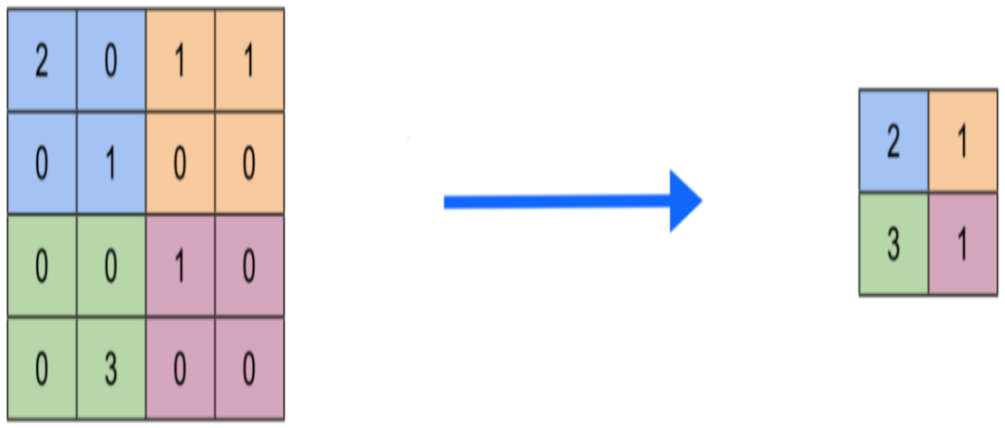


Figure 2.8: Max pooling operation visualization[4].

Average Pooling

In average pooling, unlike the max pooling, average value of all the values which come under the window is selected and placed in the new feature map as shown in figure 2.9 below.

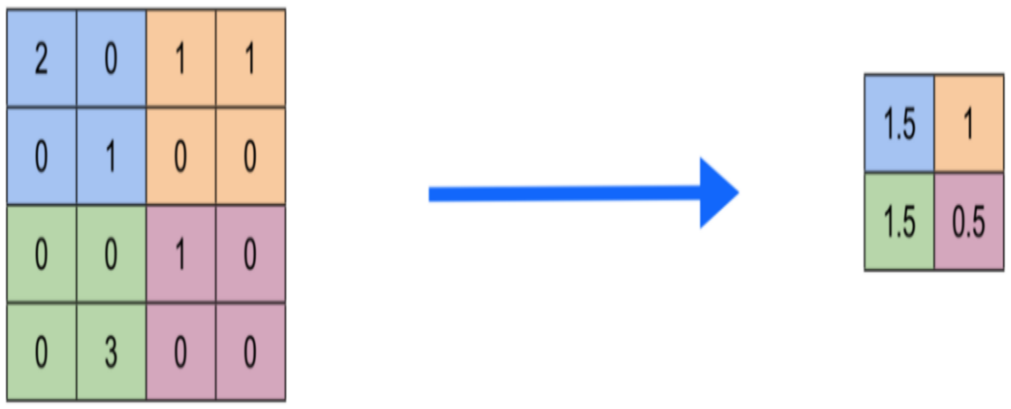


Figure 2.9: Average pooling operation visualization.

[4]http://cs231n.github.io/convolutional-networks/

2.3.3 Activation Function

Convolutional neural networks use activation function to add non-linearities into the network. Without this function neural networks would not be able to perform complex tasks like pattern recognition, classification, segmentation etc. Rectified Linear Unit (ReLU), Leaky ReLU, tangent hyperbolic (tanh) and sigmoid function are the most used activation functions.

ReLU

Rectified Linear Unit (ReLU) is the most used activation function in neural networks. The function is defined as:

$$f(x) = max(0, x).$$

This function allows all non-negative values in a feature map for further operations; and all negative values are blocked and are assigned zero values as shown in figure 2.10.

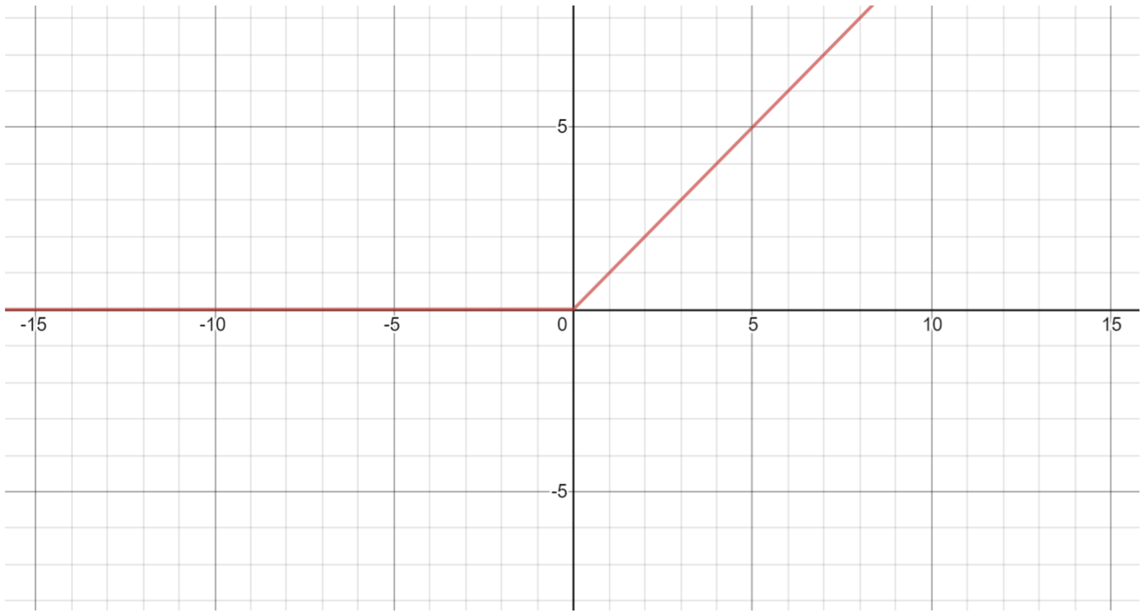


Figure 2.10: Graph of ReLU function.

Leaky ReLU

Sometimes, we come across dying ReLU problem, where all values in a feature map are blocked by the ReLU function. This problem leads to vanishing gradient and performance of neural networks is affected. To overcome this issue Leaky ReLU is proposed as an activation function, which is a modified form of ReLU. Leaky ReLU is defined as:

$$f(x) = \begin{cases} x, if x \geqslant 0 \\ ax, if x < 0 \end{cases}$$

Here, 'a' works as a parameter. If we take $a = 0.1$, then Leaky ReLU graph becomes, as shown in figure 2.11:

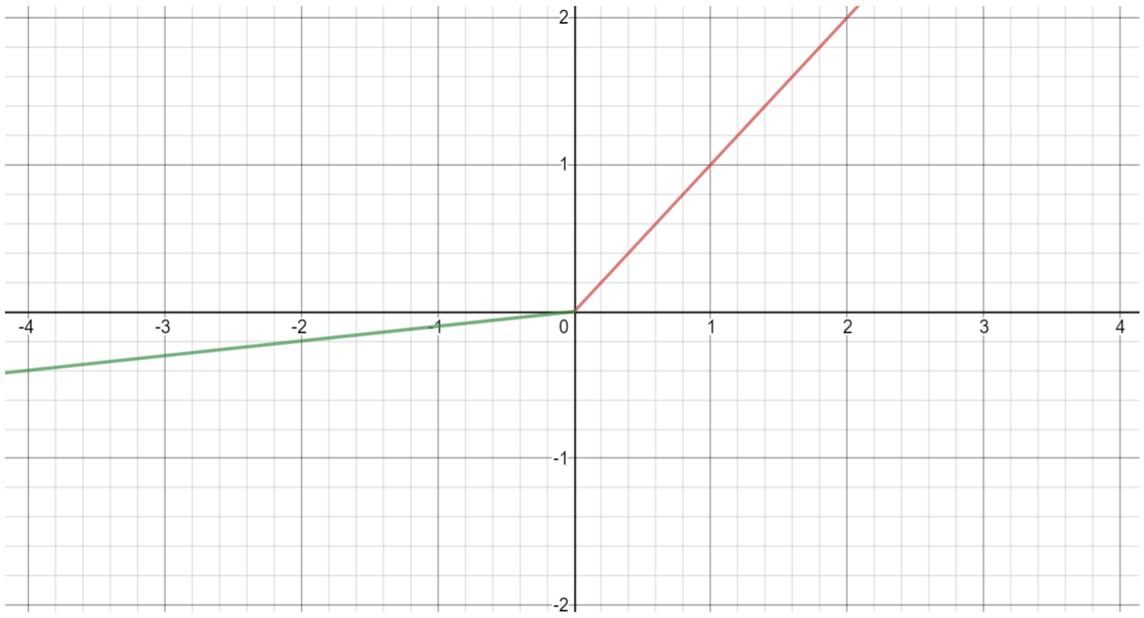


Figure 2.11: Graph of Leaky ReLU function.

This function passes all the values in a feature map to next level with adjusting negative values towards 0 and leaving others as they are.

Tanh

Tangent hyperbolic (tanh) function $f(x) = tanh(x)$ is also used as an activation function in neural networks. This function allows all values in a feature map to proceed further but by adjusting them between -1 and 1 as shown in figure 2.12.

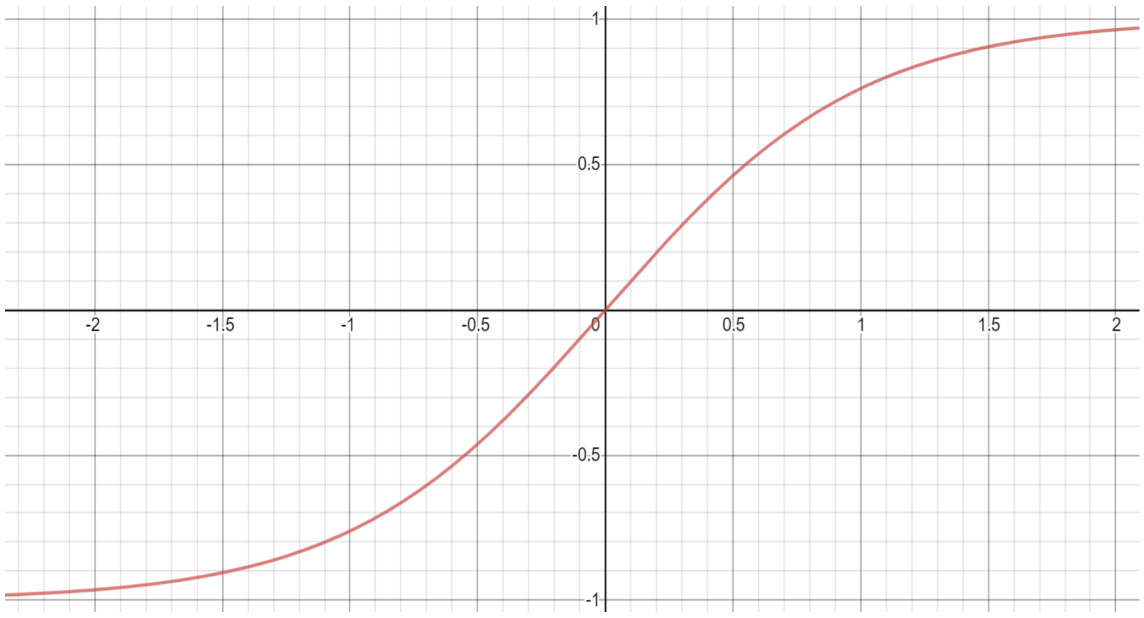


Figure 2.12: Graph of tangent hyperbolic function.

Sigmoid

Sigmoid function is defined as:

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

This function also allows all values in a feature for further operations but by adjusting them between 0 and 1, as shown in figure 2.13.

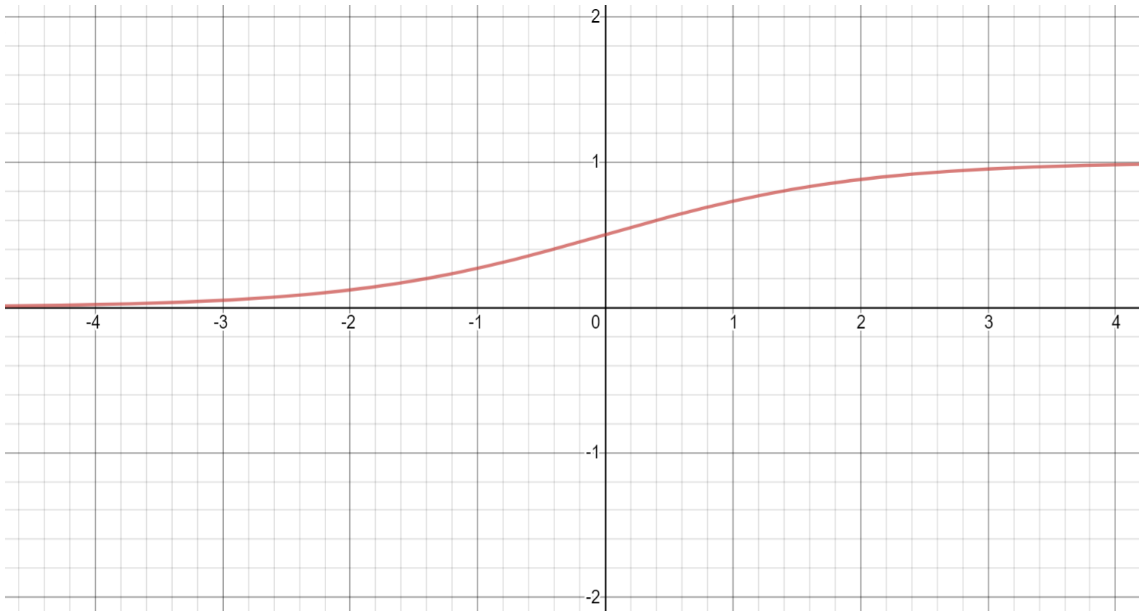


Figure 2.13: Graph of sigmoid function.

2.3.4 Batch Normalization

To avoid over-fitting and for acceleration of training process in neural networks batch normalization layers are used. Batch normalization layer controls variation in distribution by calculating mean and standard deviation values of the data set as a whole by adjusting the mean to 0 and variance to 1, the equation for batch normalization (BN) is given below as:

$$BN = \gamma_c \left[\frac{I_{n,c,h,w} - \mu_c}{\sqrt{\sigma_c^2 - \epsilon}} \right] + \beta_c.$$

Where, $I_{n,c,h,w}$ represents n-number of images provided to a neural network at a time with c channels, h heights and w widths. μ_c and σ_c^2 are channel wise global mean and variance of the images, respectively. β_c and γ_c are learnable mean and standard deviation, respectively, while ϵ is kept constant as 0.00001.

Chapter 3

Literature Review

3.1 Background

Convolutional neural networks (CNNs) were being used long before 1990s [10], but they were limited due to size of then available networks and training data. The first breakthrough achieved by krizhevsky et al. [11] in 2012, named as AlexNet, using training of large network consists of 18 layers and millions of parameters on the imageNet data and 1 million of training images. After this a large and much deeper networks have been trained [12]. Since 2010 deep convolutional networks have outperformed the conventional state-of-the-art approaches towards visual recognition tasks, like, [13] and [11]. At first CNNs were used for classification of images. Ciresan et al. [14] presented a network for automatic segmentation of electron microscopy images. For the purpose the authors have used CNN as a pixel classifier. A square window, centered at each pixel, is used for the prediction of the label of each pixel. Each window pixel is mapped into a neuron, which is followed by convolution and max-pooling layers. The network has also won the ISBI 2012 electron microscopy challenge by a huge margin.

3.1.1 U-Net Architecture

Ronneberger et al. [7] in 2015 proposed a state-of-the-art fast trained network based on fully convolutional network. The work is known as U-Net due to its architecture shape, a symmetric 'U' shape, as shown in figure 3.1.

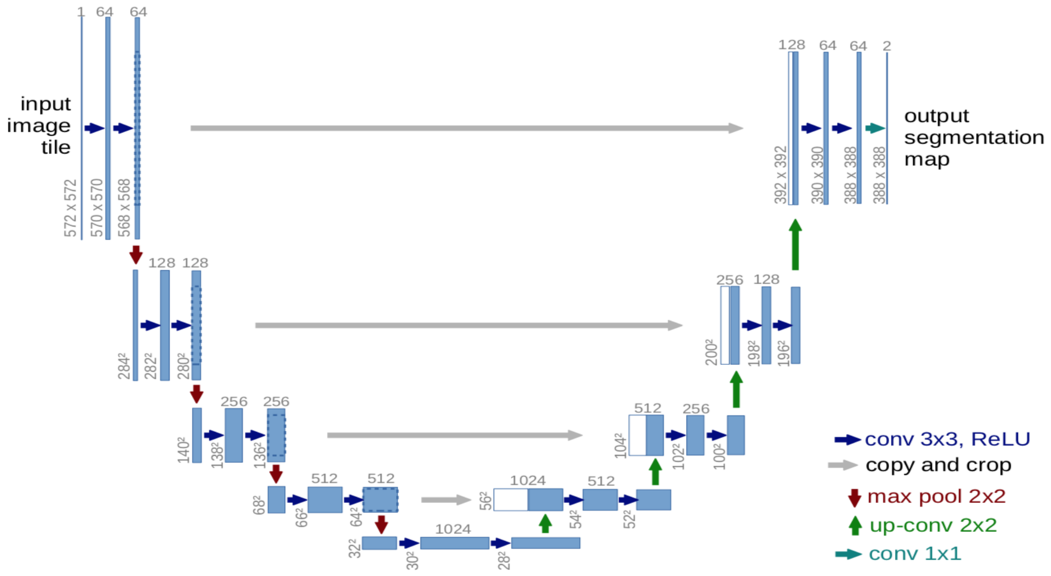


Figure 3.1: U-Net architecture.

The network consists of two paths: an encoding/contracting path and a decoding/expanding path, which are on the left and right side of the model, respectively. The encoding path captures context and a symmetric decoding path enables precise localization. The network consists of total 23 layers of convolutions. Each layer of 3x3 convolution in the contraction path is followed by a Rectified Linear Unit (ReLU) and a 2x2 max pooling operator with stride 2 for down sampling; and at each down-sampling the network makes the number of features doubled. While, in the decoding path up-sampling, a 2x2 convolution, is used which is followed by consecutive two 3x3 convolution layers, each layer is followed by a ReLU. At last, a final layer of 1x1 convolution is used to get the desired number of classes from each 64-component feature.

The authors of U-Net have applied the network in 2015 for segmentation task on data sets provided by ISBI cell tracking challenge, the challenge began in 2012 and still available online. At first, they applied the network on PhC-U373 data set which contains Glioblastoma-astrocytoma U373 cells. It obtained 92% of average value of intersection over union (IOU), and on DICHeLa data set they achieved an average 77.5% of IOU value. In both cases they got first position. Most of the neural networks presented recent years for the task of segmentation of medical images are based on the U-Net, which have outperformed other approaches for the task.

3.2 Related Work

Nowadays, most of the neural networks for medical images segmentation task are extensions of the U-net [7] architecture; and they have outperformed other approaches in the field.

3.2.1 Pyramid Dilated Res-U-Net

An end to end network based on ResNet and U-net presented in [20], as shown in figure 3.2. Each block of convolution in the encoder and decoder path of the U-net are converted to residual blocks by adding inputs of each block to its output before feeding the output to the next block of convolution.

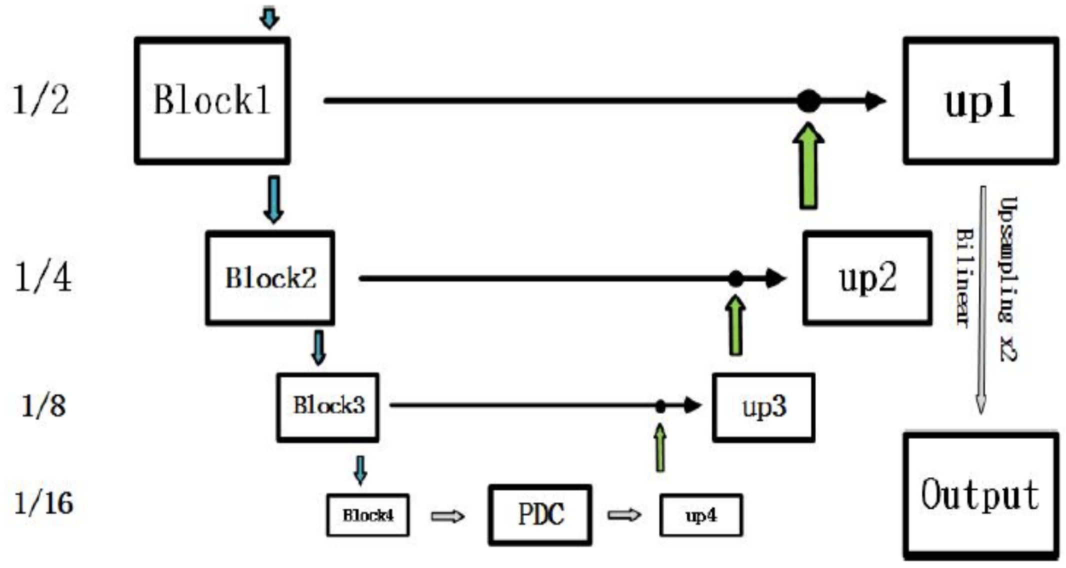


Figure 3.2: Pyramid Dilated Res-U-Net.

The network replaces pooling layers with convolution layers for possible reduction of information loss. To increase expressiveness of their model the authors have also introduced LeakyReLU instead of ReLU in the encoder path. The experimental results of the model have outperformed results of the basic U-net architecture.

3.2.2 RU-Net

Enhanced form of the U-net utilizing power of residual networks and recurrent convolutional neural networks have been presented by Alom et al. [21], as shown in figure 3.3 below.

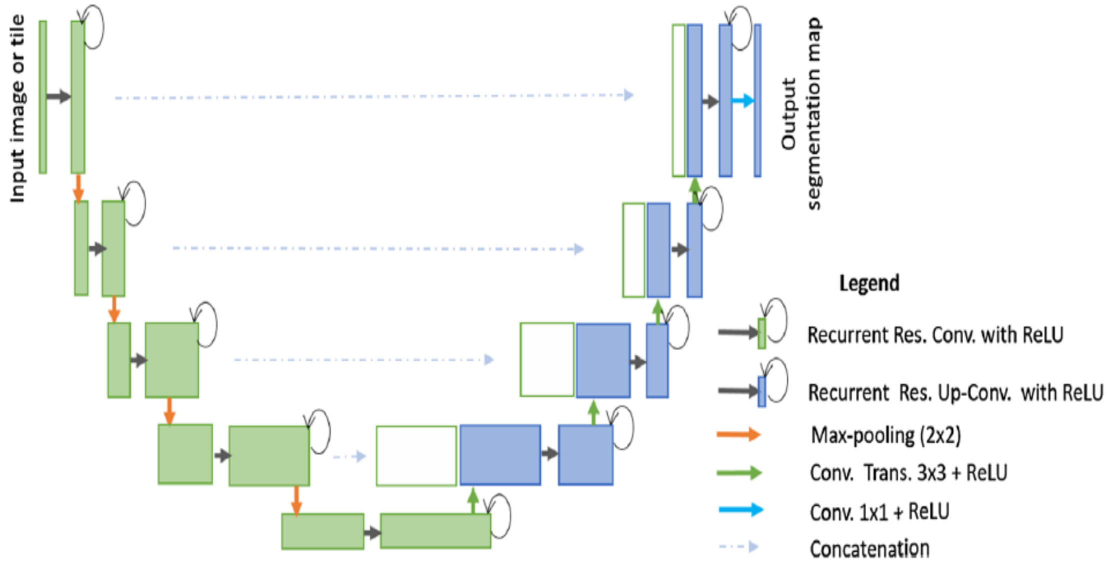


Figure 3.3: RU-Net architecture.

Residual units are selected as they are helpful in training deep neural architectures, where recurrent residual convolutional layers make sure better feature extraction in segmentation tasks. The authors have trained the model for three data sets, retinal images, skin cancer images and lung lesion segmentation achieving better segmentation results than SegNet, U-net and residual U-net.

3.2.3 MultiResUNet

In 2019, a MultiResUNet [22] named as DC-UNet has been presented, as shown in figure 3.4. In the architecture the authors have replaced convolutional blocks with MultiRes blocks and unlike to the simple skip connections in the U-Net the paper proposes Res paths. In the MultiRes block a residual connections between consecutive convolutional layers is being introduced.

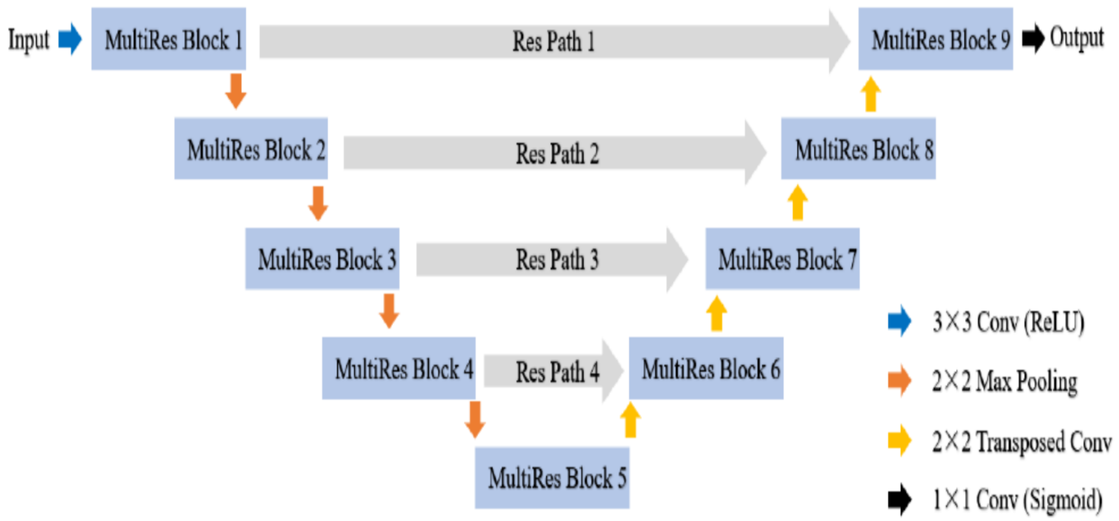


Figure 3.4: Architecture of MultiResUNet.

In addition, instead of equal number of filters the number of filters are also increasing from 1 to 3. While in the Res path, unlike to the simple concatenation from the encoder path to its corresponding decoder path, the feature maps are passed through series of residual blocks before concatenation of the features with the decoder features. The model is tested on five different data sets and has outperformed the baseline U-Net architecture.

3.2.4 UNet++

Zhou et al. [23] have presented UNet++ based on the U-Net architecture. The paper presents a densely connected skip path instead of simple skip connection in the classical U-Net and utilizes deep supervision. The architecture is shown in figure 3.5 below. The corresponding outputs of encoder and decoder paths are concatenated using densely connected layers. Deep supervision enables the model to work in two modes, a) accurate mode, and b) fast mode. In the accurate mode, output of all the segmentation branches are averaged. While in the fast mode, the best segmentation map is selected from one of the segmentation branches.

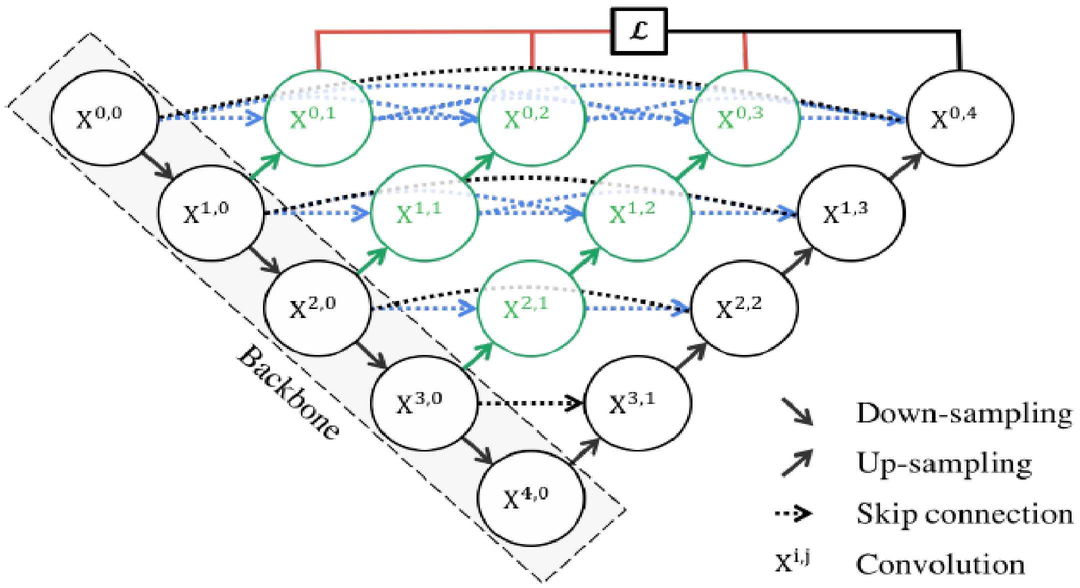


Figure 3.5: UNet++: A Nested U-Net Architecture.

The enhanced skip pathway reduces gap between corresponding feature maps of encoding and decoding path. The network is trained for segmentation of chest CT scans, microscopy images' nuclei, liver CT scans and for polyp segmentation in colonoscopy videos.

3.2.5 LSTM Multi-modal U-Net

A network consists of multiple encoding paths is presented by Fan Xu et al. [15]. The network is named as LSTM Multi-modal U-Net, shown in figure 3.6. Basically the network consists of two parts, multi modal u-net and convolutional LSTM. Multi modal u-net utilizes densely connected layers between four different encoder paths of the network for wholly exploitation of multi-modal data.

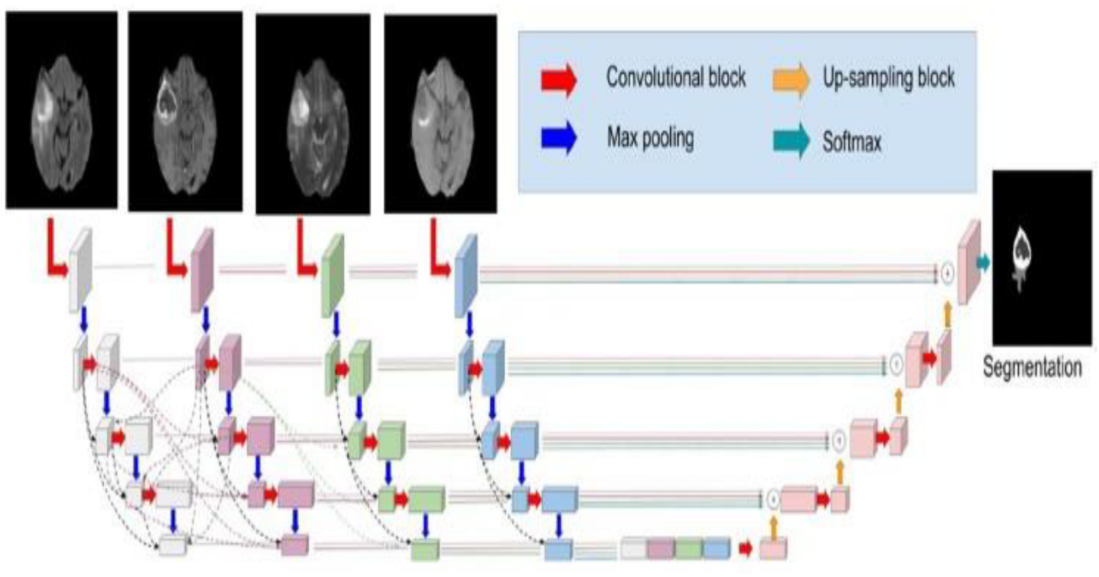


Figure 3.6: LSTM Multi-modal U-Net Architecture.

The encoder path of the U-Net is replaced by four different encoder paths according to modalities of brain tumor images used for the training process. They have regarded image depths as a sequence of slices. Convolutional LSTM is used for exploitation of sequential information between the consecutive slices of the images.

3.2.6 BCDU-Net

Another network utilizing BConvLSTM with densely connected layers (BCDU-Net) has been presented by Reza Asad et al. in 2019 [8]. The network is an extension of the U-net with utilization of power of densely connected convolutions and bi-directional ConvLSTM. The paper proposes usage of n number of densely connected convolutional layers in the bottleneck part of the classical U-net and Long Short Term Memory Convolutional layers (LSTM) after the concatenation process of corresponding output in the encoder and decoder path of the U-Net, as shown in figure 3.7.

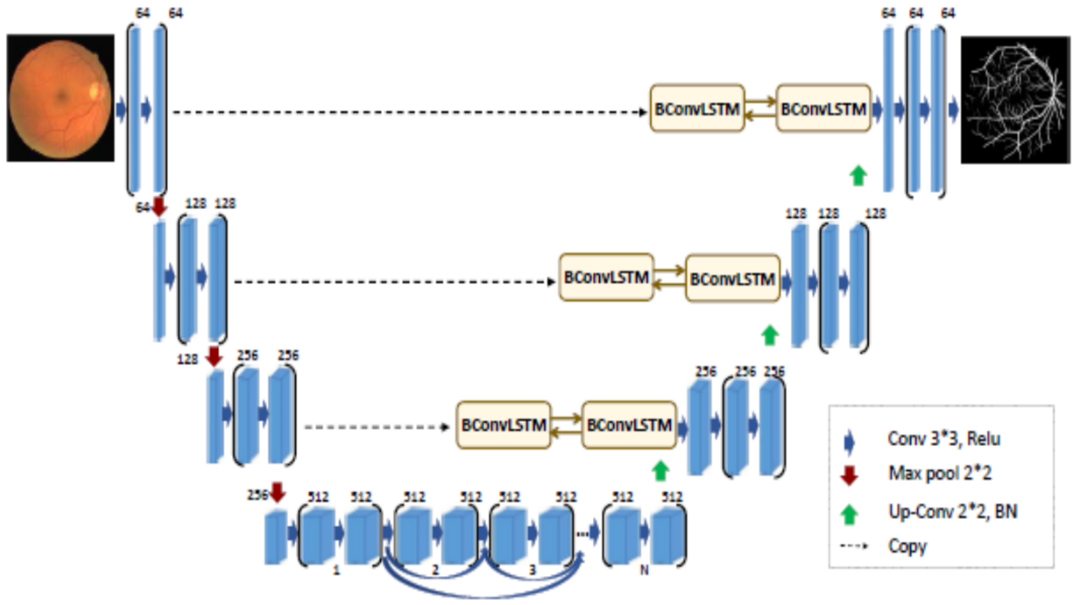


Figure 3.7: BCDU-Net with bi-directional ConvLSTM in the skip connections and densely connected convolution.

The network is tested for three data sets, retina images, skin images and lung images with and without three densely connected layers. Results show that the network with three densely connected layers has the best prediction.

3.2.7 DEFU-Net

A network, DEFU-Net [9], based on the U-Net presented recently for segmentation of chest X-ray, shown in figure 3.8. The paper proposes a dual encoder fusion U-net framework with densely connected recurrent convolutional neural network. The densely connected recurrent path of the network helps the network for extraction of feature in the images. Inception blocks with dilation are also used to increase width of the network and to enrich the network representation of features. Features from the densely connected recurrent path and the inception blocks are then summed up for decoder path.

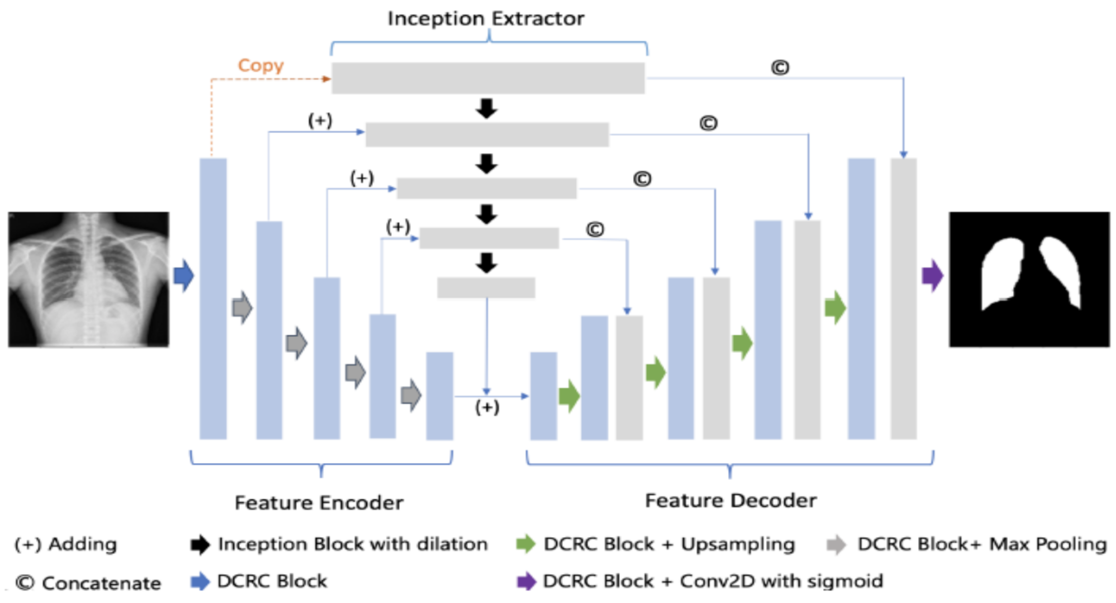


Figure 3.8: DEFU-Net with Inception dilation Convolution Blocks and Densely Connected Recurrent convolution (DCRC) Blocks.

The network is trained for chest X-ray data set achieving better performance than the basic U-net, BCDU-Net, Residual U-Net and R2U-Net.

BUSU-Net [16], a recently presented network based on the U-Net. The network is developed by combining two BCDU-Net with total of 108 layers, where one BCDU-Net is deeper than the original BCDU-Net. The network is trained for DRIVE data set, which contains retina images for blood vessel segmentation. The network has outperformed state-of-the-art networks for the data set.

In this thesis, we propose a U-Net [7] based encoder-decoder neural network for segmentation of skin lesion. The network, ResBCU-Net, utilizes power of residual blocks [17], batch normalization [18] and BConvLSTM [19] network for the task. In the network, we enhance the encoding path with residual blocks and batch normalization, and the decoding path is enhanced by using BConvLSTM network in the path. In addition, we also present a densely connected form of ResBCU-Net, where we have made changes in the bottleneck section of our model like BCDU-Net [8]. More details of our work are given in the next chapter, chapter 4.

Chapter 4

ResBCU-Net: an Encoder and Decoder Based Neural Network for Segmentation of Skin Images

Based on U-Net [7] and inspired by Residual blocks [17], Batch normalization [18] and Bi-directional Convolutional Long Short Term Memory (BConvLSTM) network [19], we present a neural network, named as ResBCU-Net, shown in the figure 4.1. We have made changes in the encoding path and decoding path of the classical U-Net, so we have categorized details of our network into sections: Encoding and Decoding.

4.1 Encoding

Unlike to the U-net [7], encoding/contracting path of ResBCU-Net consists of residual blocks [17] and batch normalization layers [18] with nine convolution layers. The path consists of three blocks, each block contains three convolution layers followed by a batch normalization layer. The output of first convolution layer in each block is added with the output of the batch normalization layer, which is then followed by a max pooling layer. At the same time, before the max pooling layer, the output of each block is passed for concatenation with the corresponding output of the decoding/expanding path.

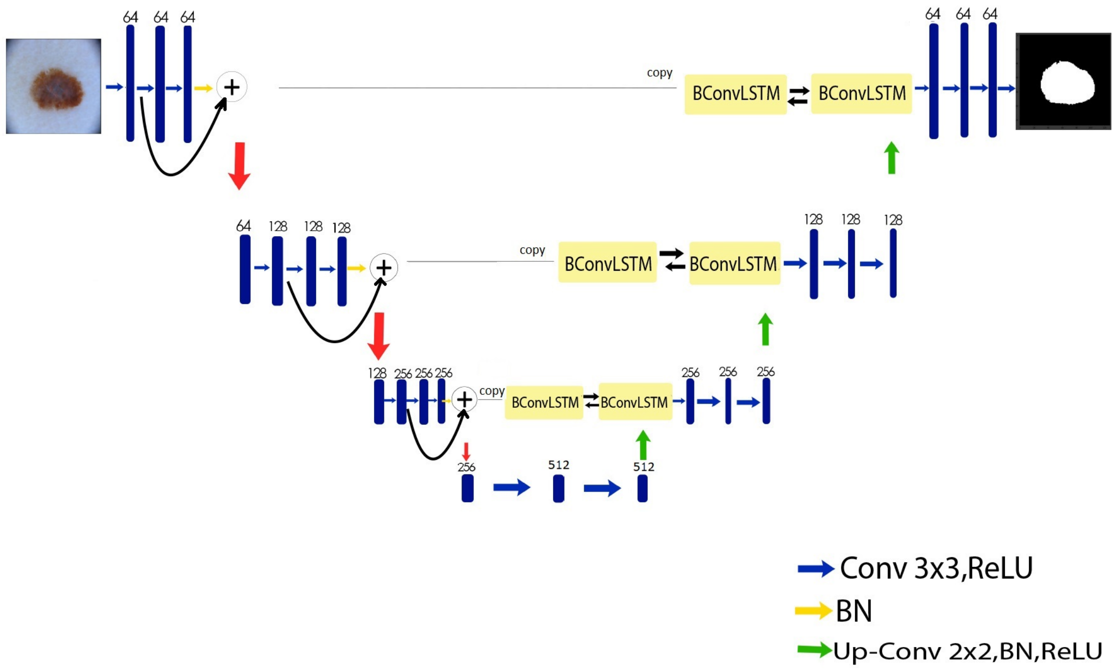


Figure 4.1: ResBCU-Net architecture with residual blocks in the encoding path and BConvLSTM in the decoding path. The numbers on top of the rectangles show number of channels.

4.1.1 Residual Blocks

Successive sequences of convolution layers lead to learning of different features, in some cases it may also lead to learning of redundant features; and adding more layers leads to higher training error. To solve this problem in such deeper models residual blocks are introduced in [17]. An input to some convolution layers is added to the output of the layers, the resultant is again feeded to the successive convolution layers, example of residual block is shown in the figure. 4.2 below.

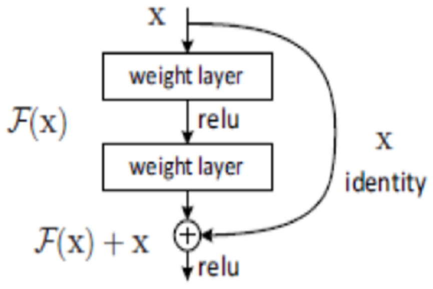


Figure 4.2: ResNet residual block.

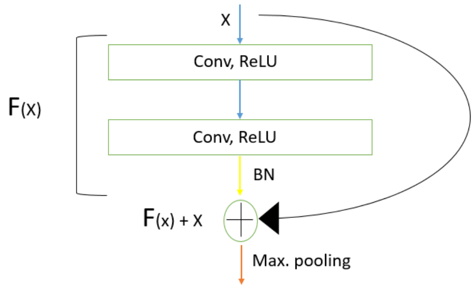


Figure 4.3: ResBCU-Net residual block.

We utilize this approach for ResBCU-Net encoding path. Instead of blocks of two convolution layers in the encoding path we introduce three convolutions blocks each followed by a batch normalization layer. Each block is then converted to residual blocks by adding output of the first convolution layer to the output of the batch normalization layer in the block, as shown in the figure. 4.3 above.

4.1.2 Batch Normalization

To avoid over-fitting and for acceleration of the training process we include batch normalization layers [18] in the encoding and decoding path of ResBCU-Net. The batch normalization layer controls variation in distribution by calculating mean and standard deviation values of the data set as a whole by adjusting the mean to 0 and variance to 1, the equation for batch normalization (BN) is given below as:

$$BN = \gamma_c \left[\frac{I_{n,c,h,w} - \mu_c}{\sqrt{\sigma_c^2 - \epsilon}} \right] + \beta_c.$$

Where, $I_{n,c,h,w}$ represents n-number of images provided to a neural network at a time with c channels, h heights and w widths. μ_c and σ_c^2 are channel wise global mean and variance of the images, respectively. β_c and γ_c are learnable mean and standard deviation, respectively, while ϵ is kept constant as 0.00001.

We introduce BN layers in each block of the encoding and decoding path. In the encoding path BN layers are used at the end of each block just before max-pooling layer. After max pooling layer of the third block of convolution layers bottleneck section of the network starts, where we use only two convolution layers each followed by an activation function, ReLU. While in the decoding path, we use batch normalization layers after each up-sampling layer, which are then followed by activation functions, ReLU, before proceeding to the next block.

4.2 Decoding

The decoding/expanding path of ResBCU-Net, inspired by BCDU-Net [8], contains convolution layers, up sampling layers, batch normalization layers [18] and Bi-directional LSTM convolutions (BConvLSTM) [19]. Right after the bottleneck portion of the network an up sampling convolution with 2x2 filter, followed by a batch normalization layer, is used which is then followed by two convolution layers block. Features from the corresponding blocks in the encoding path are passed into the BConvLSTM after concatenation with the outputs of the corresponding block of the decoding path. In each block, outputs of BConvLSTMs are passed into two convolutional layers. At the end of the decoding path, we use a convolution layer with 1x1 filter followed by a sigmoid function as an activation function.

4.2.1 BConvLSTM

In the decoding path, we take advantage of convolutional long short term memory (ConvLSTM) networks for ResBCU-Net inspired by [24, 8]. We use ConvLSTM to process features into two directions: forward and backward, known as BConvLSTM [19]. BConvLSTM have been implemented successively to enhance performance of neural networks [25, 8]. LSTMs are enhanced version of recurrent neural networks (RNNs) [26, 27, 28], which have been developed to overcome the gradient vanishing issue in long dependence of neural networks in training.

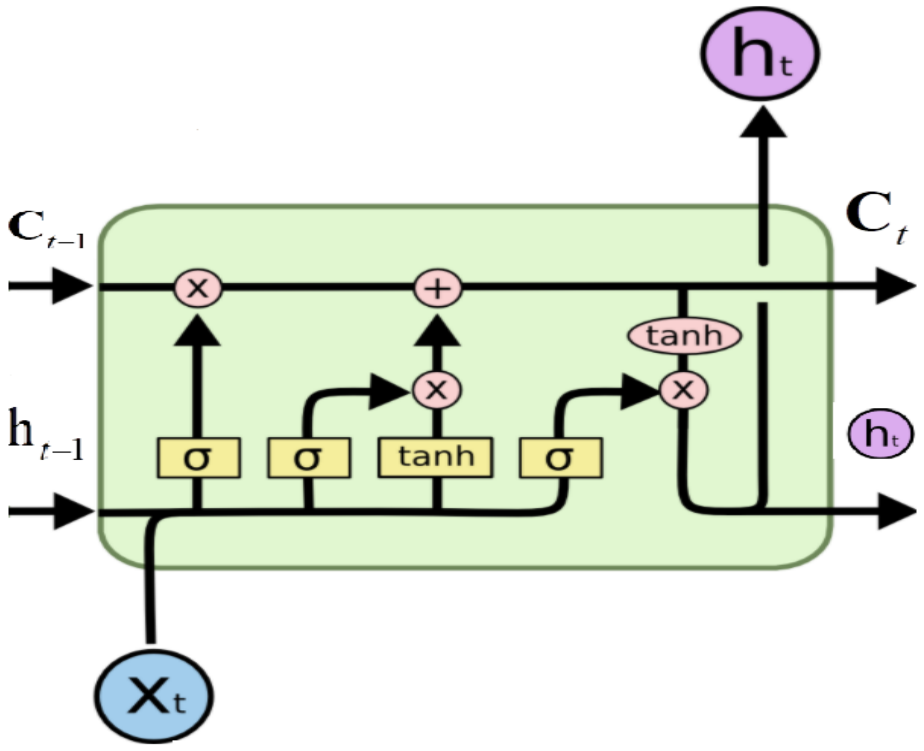


Figure 4.4: A single block of ConvLSTM network[1].

A single block of ConvLSTM consists of input gate, i_t, forget gate, f_t, and output gate, O_t. Here, $\chi_1, \chi_2, ..., \chi_t$ are inputs, $C_1, C_2, ..., C_t$ are cell state outputs and $h_1, h_2, ..., h_t$ represent hidden states. In the above figure 4.4, $\times$ and $+$ represent point wise multiplication and addition, respectively. Yellow colour blocks represent neural network layers followed by activation functions.

The above given LSTM network is a basic and simple LSTM network. We use variant of this network proposed by Gers & Schmidhuber (2000) [29]. The author

[1]http://colah.github.io/posts/2015-08-Understanding-LSTMs/

have introduced peephole connections between the cell state and the gates, the figure is given below .

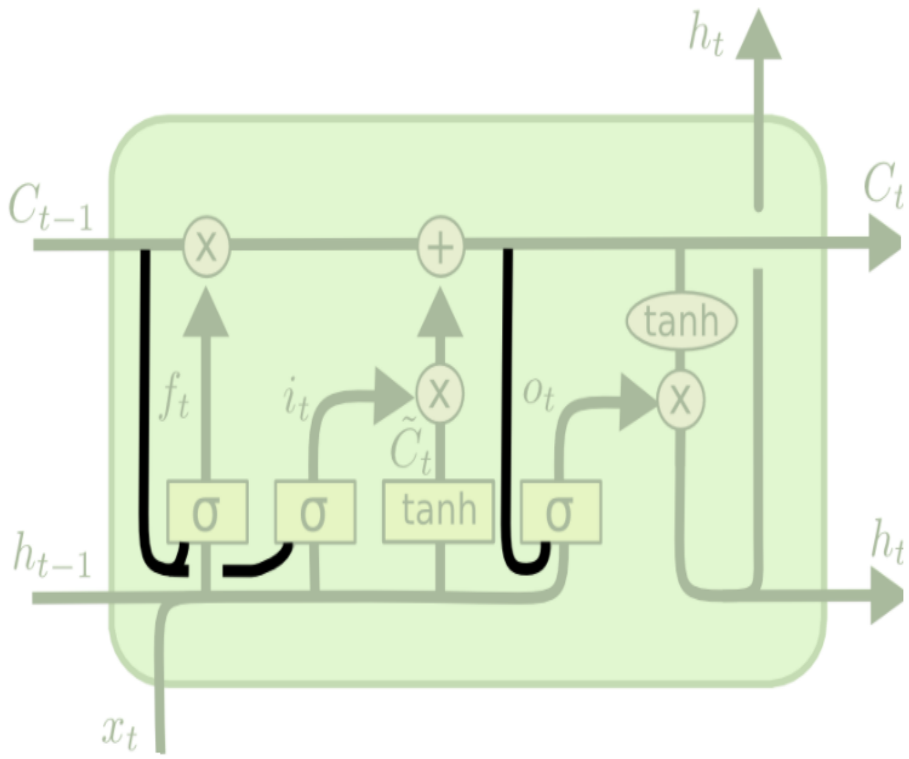


Figure 4.5: A modified ConvLSTM network[2]. Peepholes can be seen connecting the cell state with the mentioned three gates.

In the figure 4.5 above, the cell state keeps the information after all the processes inside the LSTM network which are then proceed for further operations through the hidden state. At first, the forget gate, f_t, decides which information has to be blocked from moving into the cell state, which is represented by equation 4.1 below. Where, $*$ and o are convolution operator and Hadamard product, respectively.

$$f_t = \sigma(\omega_{xf} * \chi_t + \omega_{hf} * h_{t-1} + \omega_{cf} o C_{t-1} + b_f) \tag{4.1}$$

The second phase is to letting of information to the cell state which is done by the input gate, i_t. The output of the input gate is represented by equation 4.2 below.

$$i_t = \sigma(\omega_{xi} * \chi_t + \omega_{hi} * h_{t-1} + \omega_{ci} o C_{t-1} + b_i) \tag{4.2}$$

The next phase is to update the cell state from C_{t-1} to C_t. For that, we first take Hadamard product of the old state with the output of the forget gate and also take Hadamard product of the output of the input gate and the activation function tanh. The whole process is represented by equation 4.3 below.

$$C_t = f_t o C_{t-1} + i_t o tanh(\omega_{xc} * \chi_t + \omega_{hc} * h_{t-1} + b_c) \tag{4.3}$$

Finally, the output of the network is gained using the new cell state in the output gate. The process is represented by equation 4.4.

$$O_t = \sigma(\omega_{xo} * \chi_t + \omega_{ho} * h_{t-1} + \omega_{co} o C_t + b_o) \tag{4.4}$$

The hidden state is then obtained by convolution of the output with the $\tanh(C_t)$. The final output of a single LSTM network is represented by the equation given below.

$$h_t = O_t o tanh(C_t)$$

The h_t is the hidden state, output, of the single ConvLSTM block. Now, in case of bi-directional ConvLSTM (BConvLSTM) the output can be represented as:

$$Y_t = tanh(\omega_y^{\overrightarrow{h}} * \overrightarrow{h}_t + \omega_y^{\overleftarrow{h}} * \overleftarrow{h}_t + b).$$

Here, $\overrightarrow{h}_t$ and $\overleftarrow{h}_t$ are output states of forward and backward directions features process; and the Y_t is the final output of a BConvLSTM block.

The copied features from encoding path are concatenated with corresponding outputs from decoding path and are then passed into BConvLSTM blocks. The output of these blocks are then proceed forward to the two convolution layers blocks.

4.3 Training and Results

We train ResBCU-Net in Google Colab [30] for ISIC 2018 challenge dataset [31], which contains 2594 skin images containing lesions. The data set was published by International Skin Imaging Collaboration (ISIC) for a challenge on lesion segmen-

[2] http://colah.github.io/posts/2015-08-Understanding-LSTMs/

tation, disease classification and dermoscopic feature detection. We distribute the data set as: 2300 images for training, 200 images for validation and 94 for testing. The images are of different sizes and are of high resolutions. To fit into the GPU we resize the images to 128x128 size of images. We utilize Keras with TensorFlow backened [32] for the implementation of our work; and take help from the strategy used by [8] for training our network.

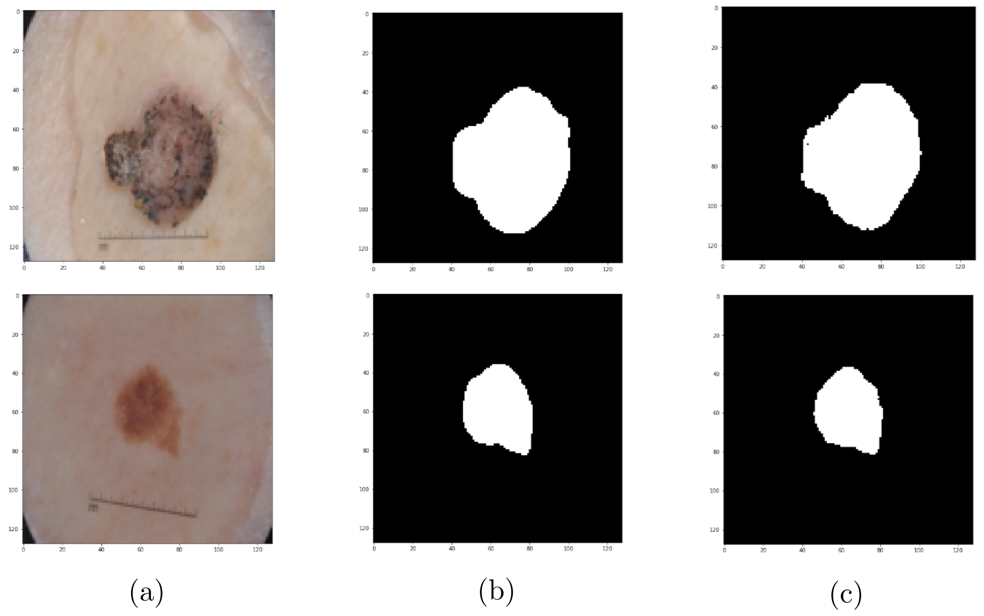


Figure 4.6: Visualization of segmentation results of ResBCU-Net. (a): Image, (b): Image mask, (c): Our network prediction

We train ResBCU-Net for 100 epochs with batch size 8. We choose 10^{-3} as starting learning rate and let the learning rate reduce to 10^{-4} by using Callback, ReduceLROnPlateau, in case of unimprovement of validation loss for seven epochs, segmentation results of ResBCU-Net are given in figure 4.6. During the training process visualization of losses and accuracies of training and validation data is given in the figures 4.7. The figures below show that the network was quite unstable at the start of training, this is because of the great variation between the images in the data set and random selection of the images for the training and validation.

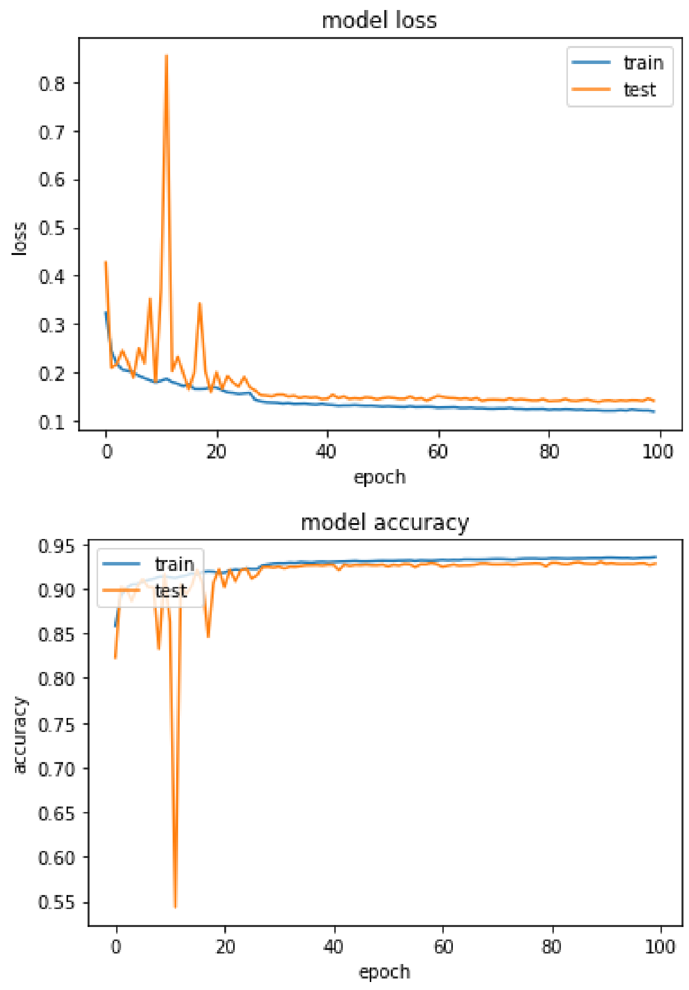


Figure 4.7: Training visualization of ResBCU-Net.

Neural networks need big data for training, in our case our data set contains only 2594 images which are not enough for training a neural network. Lack of data set causes overfitting; and initially we experienced it as expected. One option was to augment the data set and use the augmented data for training, which was not possible for us, as it requires high speed and high memory GPUs. The other option is to regularize the network. We go with the second option to avoid the overfitting, so we add 5% gaussian noise to the input images and use drop out layers in the third block and in the bottleneck of the network.

We also train the model with Densely connected layers in the bottleneck of our model just like BCDU-Net (d=3) [8] with and without batch normalization (BN) layers. Adding densely connected layers in the bottle increase the depth of the model which ultimately leads to overfitting with the same data set. Overfitting can be seen in the training visualization given in figure 4.8. The model with densely connected and BN layers shows little overfitting as compare to the model without

BN, shown in figure 4.9.

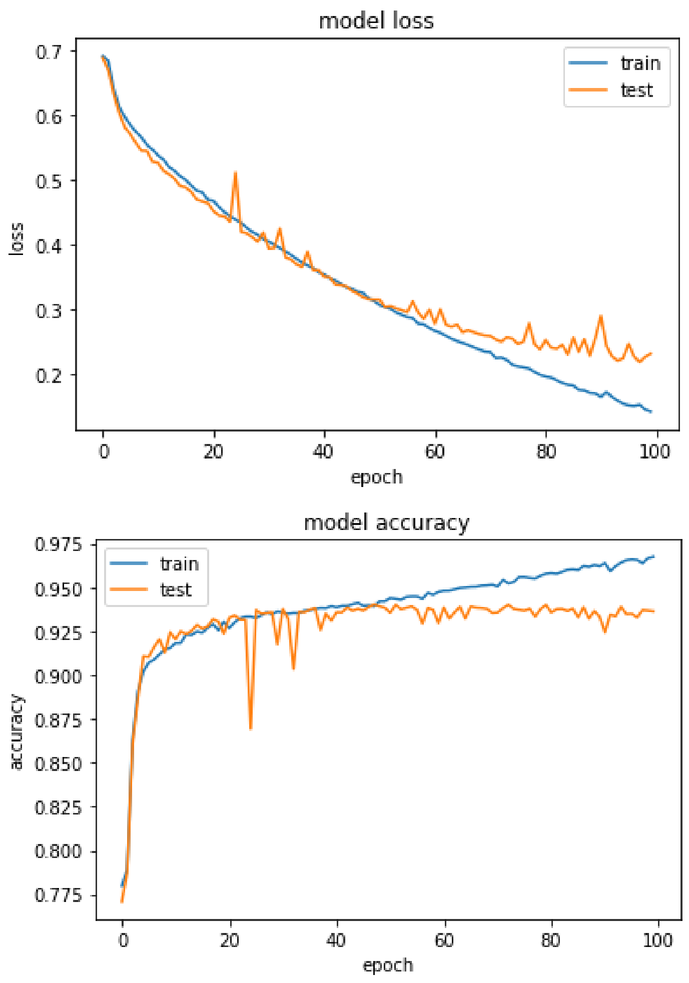


Figure 4.8: ResBCU-Net(d=3) training visualization with batch normalization layers.

In addition, we find that the model with densely connected and BN layers has more accurate results than the one without BN layers; and there is a huge difference between the training time, as the model without BN layers takes 143 seconds for an epoch while the model with BN layers only takes 51 seconds for one epoch.

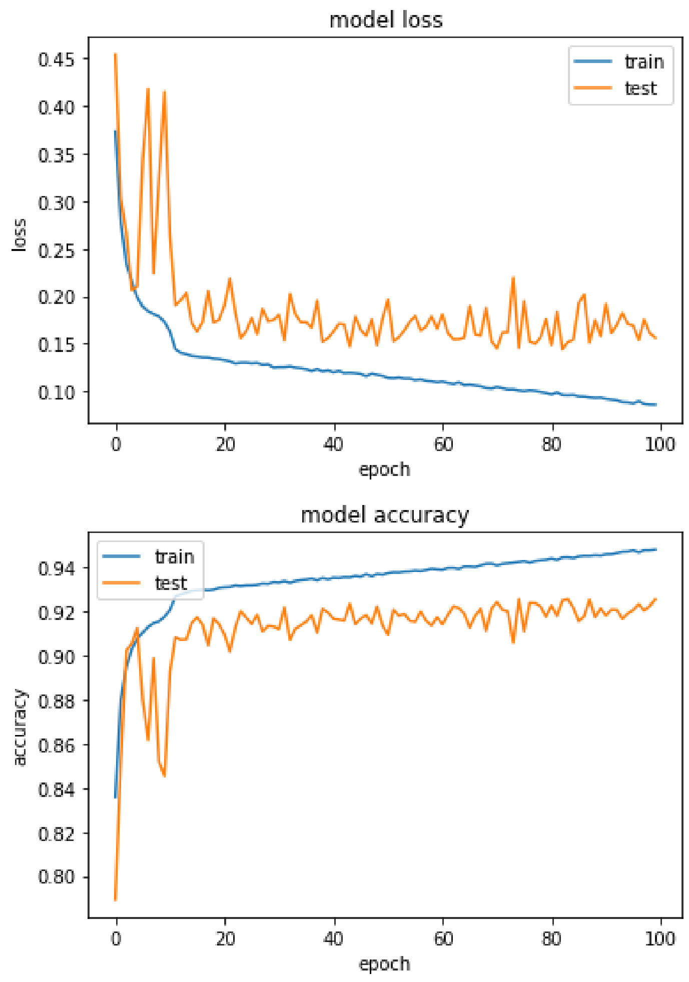


Figure 4.9: ResBCU(d=3) training visualization without batch normalization layers.

Comparative analysis gives us the significant of BN layers in accuracy, overfitting and speed up of neural networks. The model with no densely connected layers can be seen very well fit trained, shown in figure 4.7. The reason is that densely connected layers increase the depth of the network which leads to huge difference between the size of the network and size of the data set.

For quantitative analysis of results we use several metrics like, Jaccard Similarity Index (JS), specificity (SP), F1-score, and Sensitivity. Comparison, shown in the table 4.1, with other state-of-the-art networks shows that our network achieves the best results, so far, for the data set.

Table 4.1: Comparison of ResBCU-Net results with other state-of-the-art alternatives.

Models	**F1-score**	**Sensitivity**	**Specificity**	**JS**
U-Net	0.647	0.708	0.964	0.549
R2U-Net[33]	0.679	0.792	0.928	0.581
BCDU-Net(d=1)	0.847	0.783	0.980	0.936
BCDU-Net(d=3)	0.859	0.785	0.982	0.937
ResBCU-Net(d=3)	**0.864**	**0.795**	**0.987**	**0.945**
ResBCU-Net	0.845	**0.789**	0.980	**0.940**

BCDU-Net(d=3) and BCDU-Net(d=1) are networks presented by BCDU-Net [8] with dense layers and without dense layers, respectively. While, ResBCU-Net(d=3) is enhanced form of ResBCU-Net, which utilizes 3 densely connected convolutional layers in bottleneck section, as in BCDU-Net(d=3). The table shows that our network has achieved better performance as compared to the state-of-the-art neural networks. F1-score and specificity of BCDU-Net(d=3) are slightly higher than ResBCU-Net. If we look into number of parameters ResBCU-Net has 9.6 million parameters while BCDU-Net(d=3) has 20.6 million, which shows our network has less than 50% parameters than the parameters of BCDU-Net(d=3).

Chapter 5

Conclusion

In the work, we proposed an encoder-decoder neural network, named as ResBCU-Net, for segmentation of skin images containing lesions, which utilizes power of residual blocks, batch normalization and BConvLSTM network. We trained and evaluated the network on publicly available ISIC 2018 challenge dataset. The network results showed higher accuracy for our network than other state-of-the-art networks for the task. We also extended the network by utilizing densely convolutional connected layers in the bottleneck section; where we got much better results with some small amount of overfitting. We urge that if big size of data set is provided to ResBCU-Net then performance of the network will be much more better. In future, we intend to train the network for large data set or augmenting the available data set. Moreover, we want to extend our work for segmentation of brain tumor, retinal images using DRIVE data set, and chest X-rays of Covid-19 patients.

Bibliography

[1] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Transactions on image processing*, vol. 10, no. 2, pp. 266–277, 2001.

[2] L. Wang, L. He, A. Mishra, and C. Li, "Active contours driven by local gaussian distribution fitting energy," *Signal Processing*, vol. 89, no. 12, pp. 2435–2447, 2009.

[3] O. J. Tobias and R. Seara, "Image segmentation by histogram thresholding using fuzzy sets," *IEEE transactions on Image Processing*, vol. 11, no. 12, pp. 1457–1465, 2002.

[4] A. Ahmad, N. Badshah, and H. Ali, "A fuzzy variational model for segmentation of images having intensity inhomogeneity and slight texture," *Soft Computing*, pp. 1–16, 2020.

[5] C. W. Chen, J. Luo, and K. J. Parker, "Image segmentation via adaptive k-mean clustering and knowledge-based morphological operations with biomedical applications," *IEEE transactions on image processing*, vol. 7, no. 12, pp. 1673–1683, 1998.

[6] S. Z. Oo and A. S. Khaing, "Brain tumor detection and segmentation using watershed segmentation and morphological operation," *International Journal of Research in Engineering and Technology*, vol. 3, no. 03, pp. 367–374, 2014.

[7] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.

[8] R. Azad, M. Asadi-Aghbolaghi, M. Fathy, and S. Escalera, "Bi-directional convlstm u-net with densley connected convolutions," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 0–0, 2019.

[9] L. Zhang, A. Liu, J. Xiao, and P. Taylor, “Dual encoder fusion u-net (defu-net) for cross-manufacturer chest x-ray segmentation,” *arXiv preprint arXiv:2009.10608*, 2020.

[10] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, “Backpropagation applied to handwritten zip code recognition,” *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[12] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[13] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[14] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, “Deep neural networks segment neuronal membranes in electron microscopy images,” in *Advances in neural information processing systems*, pp. 2843–2851, 2012.

[15] F. Xu, H. Ma, J. Sun, R. Wu, X. Liu, and Y. Kong, “Lstm multi-modal unet for brain tumor segmentation,” in *2019 IEEE 4th International Conference on Image, Vision and Computing (ICIVC)*, pp. 236–240, IEEE, 2019.

[16] W. H. Khoong, “Busu-net: An ensemble u-net framework for medical image segmentation,” *arXiv preprint arXiv:2003.01581*, 2020.

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

[18] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.

[19] H. Song, W. Wang, S. Zhao, J. Shen, and K.-M. Lam, “Pyramid dilated deeper convlstm for video salient object detection,” in *Proceedings of the European conference on computer vision (ECCV)*, pp. 715–731, 2018.

[20] Q. Zhang, Z. Cui, X. Niu, S. Geng, and Y. Qiao, "Image segmentation with pyramid dilated convolution based on resnet and u-net," in *International Conference on Neural Information Processing*, pp. 364–372, Springer, 2017.

[21] M. Z. Alom, C. Yakopcic, M. Hasan, T. M. Taha, and V. K. Asari, "Recurrent residual u-net for medical image segmentation," *Journal of Medical Imaging*, vol. 6, no. 1, p. 014006, 2019.

[22] A. Lou, S. Guan, and M. Loew, "Dc-unet: Rethinking the u-net architecture with dual channel efficient cnn for medical images segmentation," *arXiv preprint arXiv:2006.00414*, 2020.

[23] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pp. 3–11, Springer, 2018.

[24] Y. Guo, J. Stein, G. Wu, and A. Krishnamurthy, "Sau-net: A universal deep network for cell counting," in *Proceedings of the 10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, pp. 299–306, 2019.

[25] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction," *arXiv preprint arXiv:1801.02143*, 2018.

[26] M. I. Jordan, "Attractor dynamics and parallelism in a connectionist sequential machine," in *Artificial neural networks: concept learning*, pp. 112–127, 1990.

[27] A. Cleeremans, D. Servan-Schreiber, and J. L. McClelland, "Finite state automata and simple recurrent networks," *Neural computation*, vol. 1, no. 3, pp. 372–381, 1989.

[28] B. A. Pearlmutter, "Learning state space trajectories in recurrent neural networks," *Neural Computation*, vol. 1, no. 2, pp. 263–269, 1989.

[29] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, vol. 3, pp. 189–194, IEEE, 2000.

[30] E. Bisong, “Google colaboratory,” in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, pp. 59–64, Springer, 2019.

[31] N. Codella, V. Rotemberg, P. Tschandl, M. E. Celebi, S. Dusza, D. Gutman, B. Helba, A. Kalloo, K. Liopyris, M. Marchetti, *et al.*, “Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic),” *arXiv preprint arXiv:1902.03368*, 2019.

[32] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, “Tensorflow: Large-scale machine learning on heterogeneous distributed systems,” *arXiv preprint arXiv:1603.04467*, 2016.

[33] M. Z. Alom, M. Hasan, C. Yakopcic, T. M. Taha, and V. K. Asari, “Recurrent residual convolutional neural network based on u-net (r2u-net) for medical image segmentation,” *arXiv preprint arXiv:1802.06955*, 2018.