Tumor Segmentation using deep learning architecture

Gaurav Kanwat

(2017BCS0020)

Faculty:

Dr. Bibal Benifa

Abstract

Healthy brains are typically made of 3 types of tissues: the white matter, the gray matter, and the cerebrospinal fluid. The goal of brain tumor segmentation is to detect the location and extension of the tumor regions, namely active tumorous tissue (vascularized or not), necrotic tissue, and edema (swelling near the tumor). A tumour is a mass of tissue that's formed by an accumulation of abnormal cells. This is done by identifying abnormal areas when compared to normal tissue. Since glioblastomas are infiltrative tumors, their borders are often fuzzy and hard to distinguish from healthy tissues. Tumour cells grow, even though the body does not need them, and unlike normal old cells, they don't die. Brain tumour means the aggregation of abnormal cells in some tissues of the brain. Brain tumours can be cancerous or noncancerous. So, the early detection of tumour cells plays a major role in treatment and recovery of patients. Diagnosing a brain tumour usually undergoes a very complicated and time consuming process. The MRI images of various patients at various stages can be used for the detection of tumours. There are various types of feature extraction and classification methods which are used for detection of brain tumour from MRI images. Convolutional Neural Network image classification algorithm helps in detecting the tumour at early stage with high accuracy. We proposed a Recurrent Neural Network architecture for detection of tumour cells which will give more accuracy. A recurrent neural network (RNN) is a type of artificial neural network in that connections between nodes form a directed graph along a temporal sequence. Here we have used CNN (Convolutional Neural Network) and RNN (Recurrent Neural Network) to compare which one will give more accuracy.

INTRODUCTION

A brain tumor occurs when abnormal cells form within the brain. Our brain is enclosed by a skull which is very rigid. Any growth inside such a restricted space can cause many problems for humans. The cause of most brain tumors is unknown, Uncommon risk factors include exposure to vinyl chloride, Epstein–Barr virus, ionizing radiation, and inherited syndromes such as neurofibromatosis, tuberous sclerosis, and von Hippel-Lindau Disease. Brain tumours can be both cancerous (malignant) or noncancerous (benign). The pressure inside the skull increases when benign or malignant tumours grow. This will result in brain damage, and it can be life-threatening. Brain tumours usually appear in various locations with different dimensions and shapes. Brain tumours are categorized as primary or secondary. A primary brain tumour originates in our brain. Many primary brain tumours are benign. A secondary brain tumour, which is also known as a metastatic brain tumour, occurs when cancer cells spread to our brain from another organ, such as lung or breast. Early detection of tumour cells can save a large number of human lives. Detecting the brain tumour and its stage undergoes a very complicating and time consuming process. The patient refers to MRI when some symptoms related to tumours have appeared. After examining the brain images, if tumour existence is suspected, the patient's brain biopsy comes into action. Biopsy is an invasive procedure and in some cases it may even take up to a month for a definite answer.

MODULES INVOLVED IN PREDICTION MODEL

The data samples are first pre-processed. Jupyter notebook is used to run and compile the code. The modules involved in prediction model are:

Pre-processing and normalization. The dataset can be splitted into training dataset and testing dataset, Data Augmentation, Design of Prediction model (CNN based and RNN based), Training the Network model with the help of training dataset, Testing the Network model with the help of testing dataset.

Pre-processing

In MR imaging technique, specific image appearances are given by setting some parameters such as radio frequency pulses and gradients. T1 and T2 are the most common MRI sequences and each provides particular details about tissues that occurred in the brain. T1 images are employed in this study, for normal cases. In order to reduce the number of normal brain images, six sections with approximately equal intervals have been selected from MR images of each normal subject (person). In the gadolinium enhanced T1 MR images, due to the injection of a contrast agent called Gadolinium, tumour boundary can be better identified. The above mentioned images are used to classify tumour grades in this study. In addition, in the proposed method one or several slices of an MRI image are used and it is not necessary to use all slices or 3D images.

Dataset Split-Up

In this work, we have taken images from a dataset. 80% of the images are used for Training for the model and 20% of the images are used for testing the proposed prediction model.

Data augmentation

Data augmentation is a strategy that enables practitioners to significantly increase the diversity of data available for training models, without actually collecting new data. Creating fake data and adding to the dataset is a very simple and straight way that it is done in the data augmentation step. In the proposed method, some manipulated images were added to the original data set by applying random changes to the already existed data set.

CONVOLUTIONAL NEURAL NETWORK AND RECURRENT NEURAL NETWORK BASED PREDICTION MODEL (RNN and CNN)

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various objects in the image and be able to differentiate one from the other. It contains an input and an output layer, as well as multiple hidden layers. The hidden layers of a CNN typically consist of a series of convolutional layers that describes a multiplication or other dot product. The activation function is generally known as a RELU layer, and is a subsequent function followed by additional convolutions referred to as hidden layers such as pooling layers, fully connected layers and normalization layers, because their inputs and outputs are enclosed by the activation function and final convolution.

CNNs are efficient supervised methods of deep learning which have made specific improvements in the image processing field. Generally, a convolutional network architecture contains convolutional, pooling, and fully connected are three main layers. In convolutional layers, the network uses different kernels to represent the input image to create various feature maps. Applying this layer method will significantly reduce the number of parameters (weight sharing) of the network and the network learns the correlation between the neighbour pixels (local connectivity. There are two types of training in every convolutional neural network: Feed forward and Back propagation. In feedforward, input images are fed to the network. In each layer, the convolution operation on input with convolution filter, followed by pooling operation is performed.

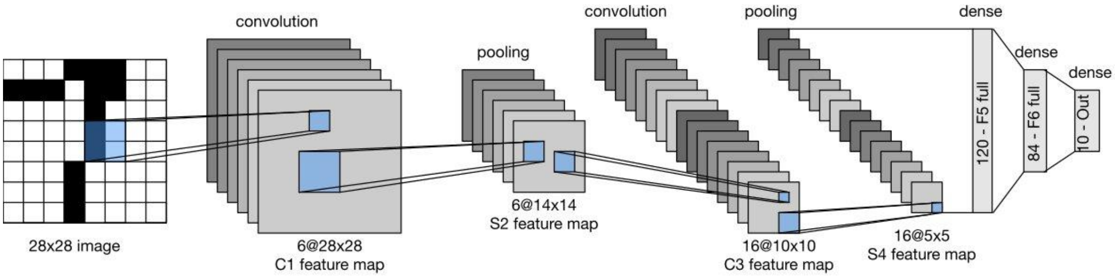


Figure 1: LeNet convolutional neural network structure

Then finally, the classification output is computed by applying sigmoid activation function. By using a loss function, the network output is compared with the desired output (correct answers) and the error rate is computed then, based on the error, the back propagation stage begins. Calculation of the gradient of each parameter is done in this step using the chain rule and finally all the parameters are updated. This is repeated for an adequate number of iterations.

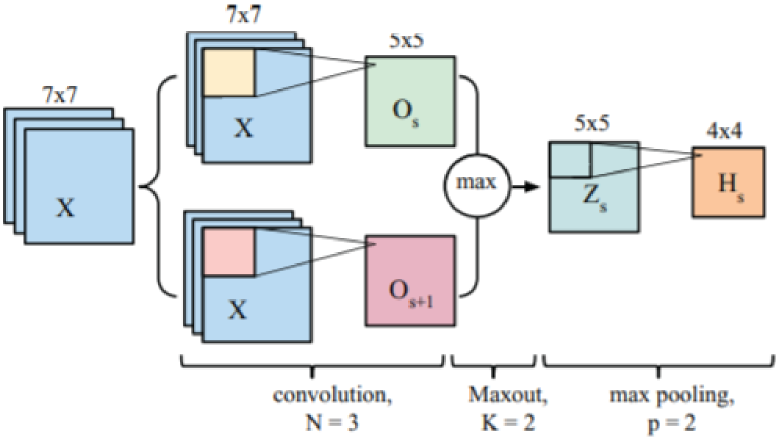


Figure 2: A single convolution layer block showing computations for a single feature map. The input patch (7 x 7) is convolved with a series of kernels (3 x 3) followed by Max-out and max-pooling.

CNNs have unique layers called convolutional layers which separate them from RNNs and other neural networks. Recurrent neural networks (RNNs) are a class of neural networks that allow previous outputs to be used as inputs while having hidden states. These are designed to interpret temporal or sequential information. RNNs use other data points in a sequence to make better predictions. They do this by taking in input and reusing the activations of previous nodes or later nodes in the sequence to influence the output. RNN has a recurrent connection on the hidden state. This looping constraint ensures that sequential information is captured in the input data.

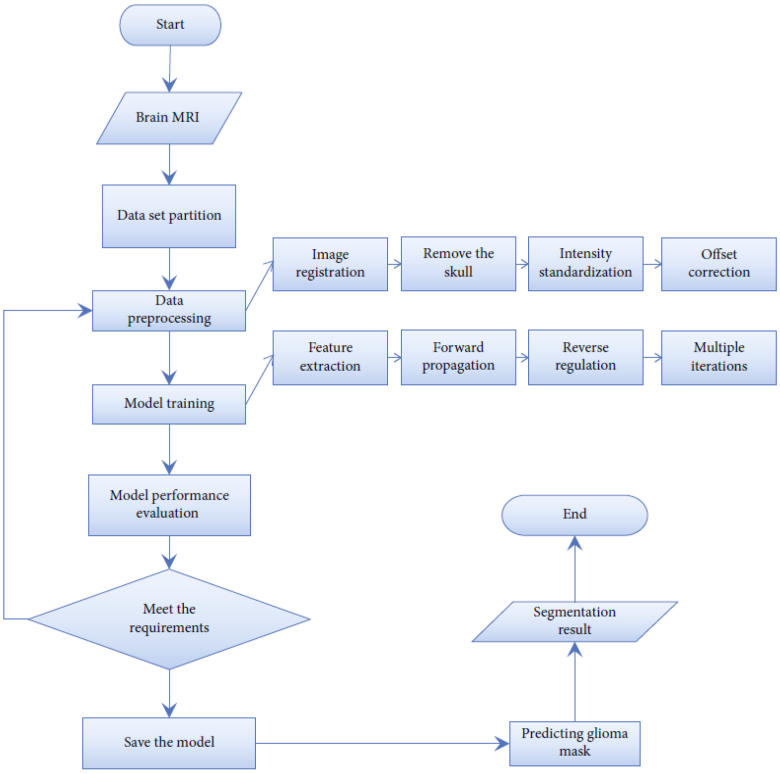


Figure 3: Flow Chart representation of the Implementation

Weight initialization for Neural Networks

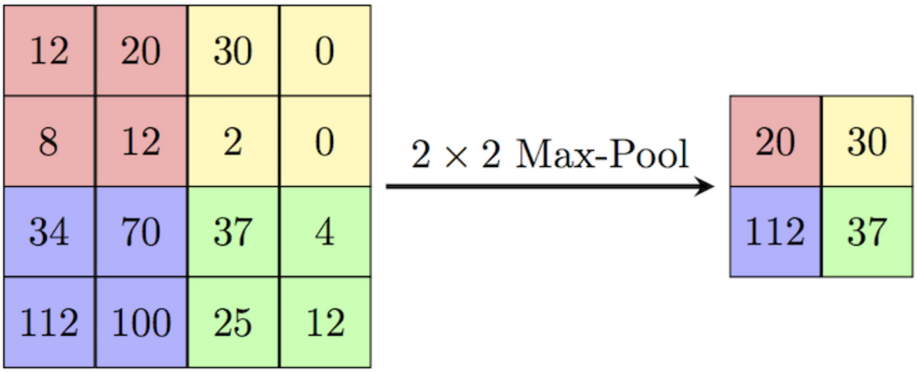
The aim of weight initialization is to prevent layer activation outputs from exploding or vanishing during the course of a forward pass through a deep neural network. If either occurs, loss gradients can flow backwards either be too small or too large beneficially, and the network will take longer to converge, if it is even able to do so at all. The product of this multiplication at one layer can be the inputs of the subsequent layer, and so on.

Activation function

The product of the multiplication at one layer can be the inputs of the subsequent layer, and so on. The performance of a deep learning model is determined by number of input images, quality of input, activation functions, number of convolution layers, pooling layers, dropout layers, number of nodes in each layer of model. In this proposed model, the training speed of the deep networks is increased by applying the Rectified Linear unit (ReLU) activation function. relu(x) = x if x ≥ 0 , 0 if x < 0

Pooling

The pooling operation involves sliding a two-dimensional filter over each channel of the feature map and summarising the features lying within the region covered by the filter. A pooling layer generally comes after a convolutional layer and which reduces the number of parameters of the network and the size of the feature maps which cause a decrease in computational costs. Max-pooling is one of the widely used pooling methods.



12	20	30	0
8	12	2	0
34	70	37	4
112	100	25	12

20	30
112	37

Figure 4: An example of Max-pooling (type of pooling)

RECURRENT NEURAL NETWORK BASED PREDICTION MODEL

Recurrent Neural Networks (RNNs) are neural networks that can deal with time series input data. Some of the applications of this network are Speech Recognition, Sentiment analysis, Machine. The data can be scanned from left to right in recurrent neural networks. The error is back propagated from the last time step to the first time step. The error at each time step is calculated and this allows us to update the weights. RNN converts the independent activations into dependent activations by providing the same weights and biases to all the layers, thus reducing the complexity of increasing parameters and giving input to the next hidden layer by memorizing each previous output. Hence all these three layers can be joined together such that the bias and weights of all the hidden layers can be same and formed into a single recurrent layer.

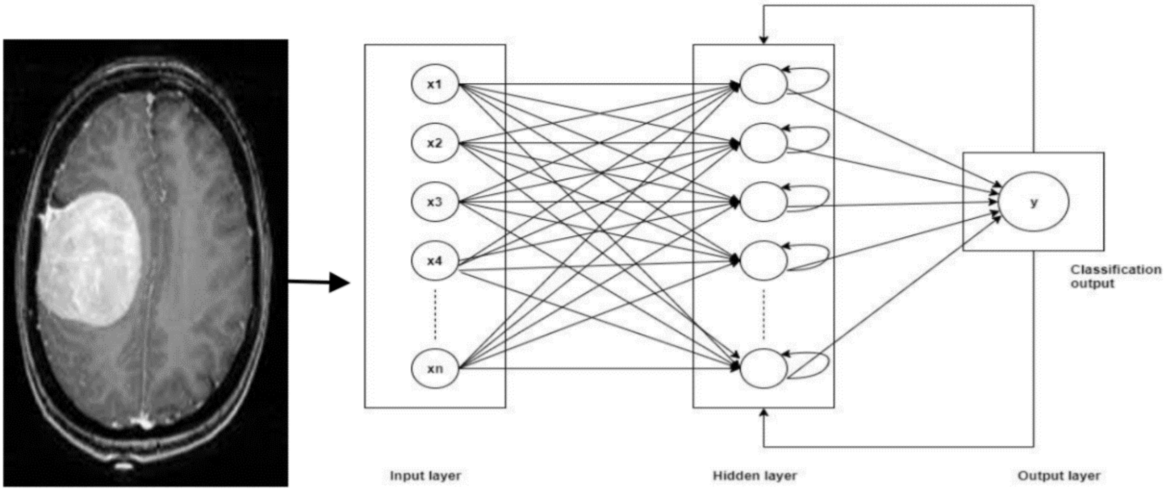


Figure 5: Implementation of functions on different layers

PERFORMANCE ANALYSIS

In this section we have discussed performance metrics, how accurate models can be achieved by parameter tuning and comparison between CNN based models and RNN based models.

Performance Metrics

A performance metric is used to measure the performance of the implemented system. Here, we have taken loss and accuracy for analyzing the performance of the proposed system.

Accuracy

Accuracy is one metric for evaluating classification models. It is the proportion of true results among the total number of cases examined. Accuracy of the system can be calculated by creating a confusion matrix and using the following formula:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

where,

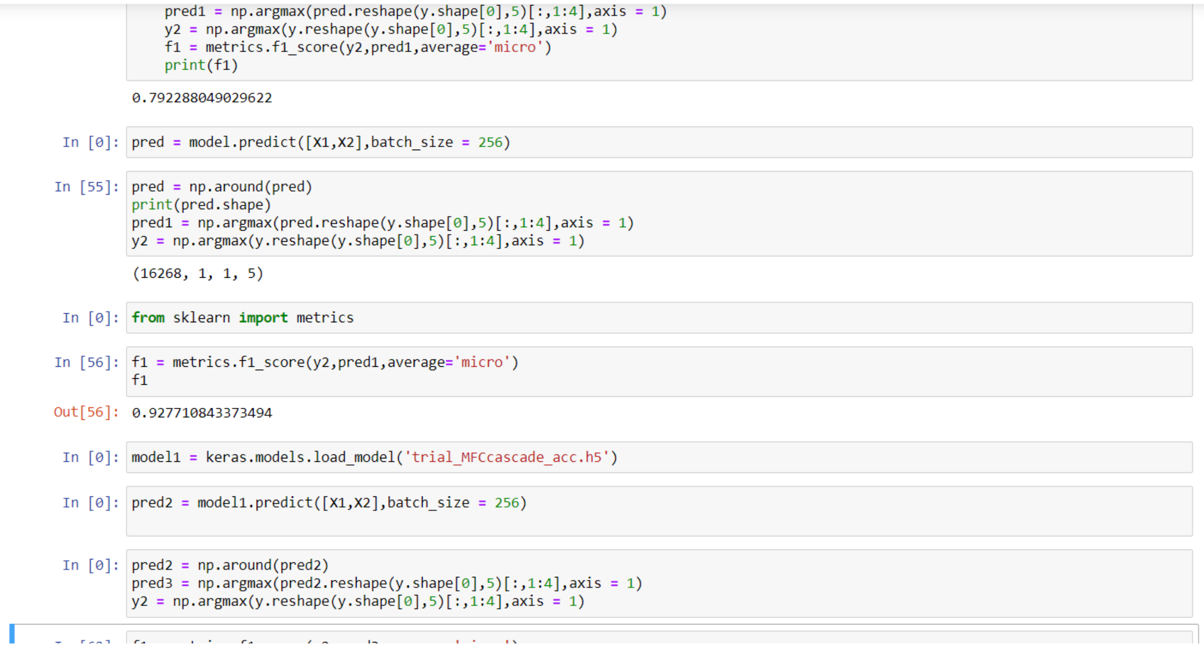
- TP: True Positive: Predicted values correctly predicted as actual positive
- FP: Predicted values incorrectly predicted an actual positive. i.e., Negative values predicted as positive
- FN: False Negative: Positive values predicted as negative
- TN: True Negative: Predicted values correctly predicted as an actual negative

Loss

The loss functions are used for updating the weight vector by using labelled output and calculated output of the model. This work uses two commonly used methods of finding the loss is “Gradient Descent” and Mean-Square-Error functions. According to mathematical decision and optimization theory, a loss function or cost function is a function that maps an event of one or more variables onto a real number intuitively representing some "cost" associated with the event.

Training

Training a model simply means learning values for all the weights and the bias from labeled examples. In supervised learning, builds a model by examining many sample images and attempting to find a model that minimizes loss. This process is called empirical risk minimization. The goal of training a model is to find a set of weights and biases that leads to minimization of loss, on average across all examples. To get an efficient prediction model, we have tested the model with different learning rate, number of hidden layers, number of nodes in each layer, different optimizers, number of epochs. The model is evaluated by their accuracy and loss.



```
    pred1 = np.argmax(pred.reshape(y.shape[0],5)[:,1:4],axis = 1)
    y2 = np.argmax(y.reshape(y.shape[0],5)[:,1:4],axis = 1)
    f1 = metrics.f1_score(y2,pred1,average='micro')
    print(f1)

0.792288049029622

In [0]: pred = model.predict([X1,X2],batch_size = 256)

In [55]: pred = np.around(pred)
         print(pred.shape)
         pred1 = np.argmax(pred.reshape(y.shape[0],5)[:,1:4],axis = 1)
         y2 = np.argmax(y.reshape(y.shape[0],5)[:,1:4],axis = 1)

         (16268, 1, 1, 5)

In [0]: from sklearn import metrics

In [56]: f1 = metrics.f1_score(y2,pred1,average='micro')
         f1

Out[56]: 0.927710843373494

In [0]: model1 = keras.models.load_model('trial_MFCcascade_acc.h5')

In [0]: pred2 = model1.predict([X1,X2],batch_size = 256)

In [0]: pred2 = np.around(pred2)
        pred3 = np.argmax(pred2.reshape(y.shape[0],5)[:,1:4],axis = 1)
        y2 = np.argmax(y.reshape(y.shape[0],5)[:,1:4],axis = 1)
```

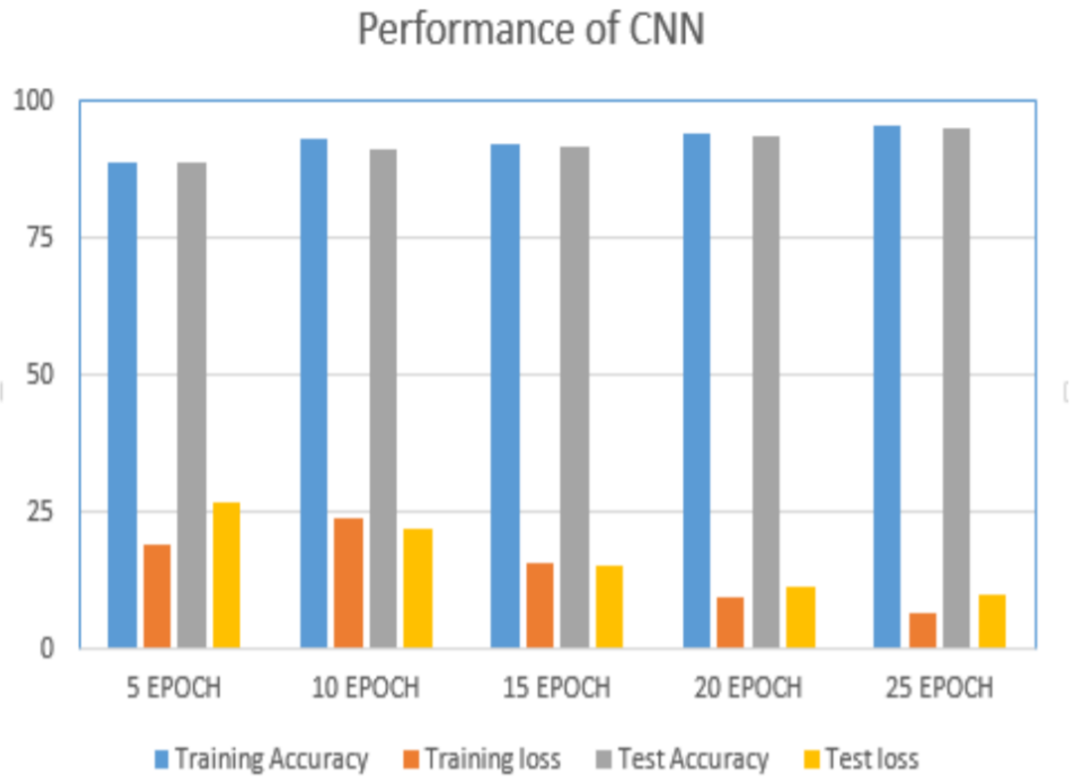
FIgure 6: Testing and finding the accuracy of implemented model

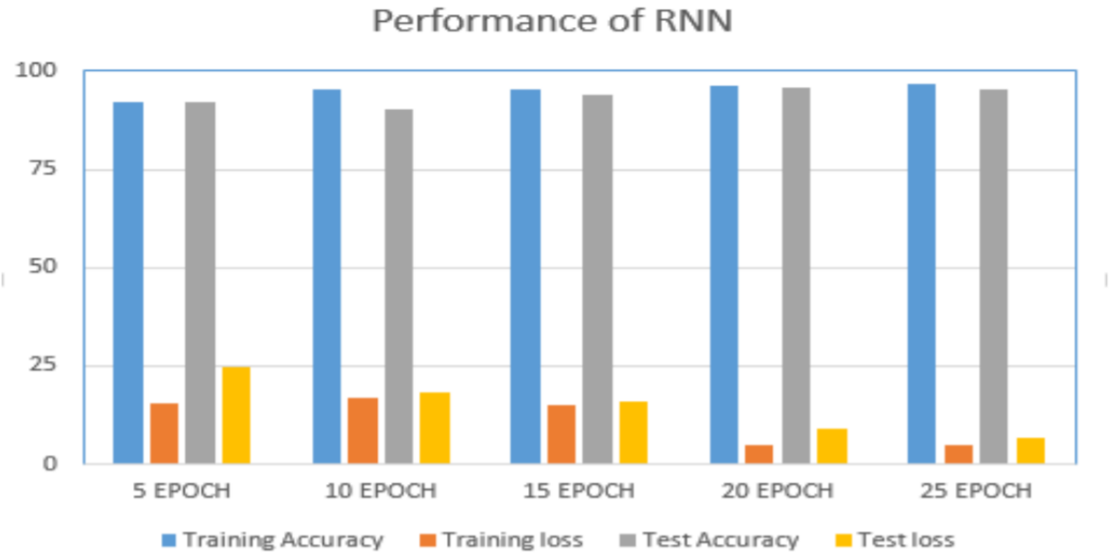
Transfer learning

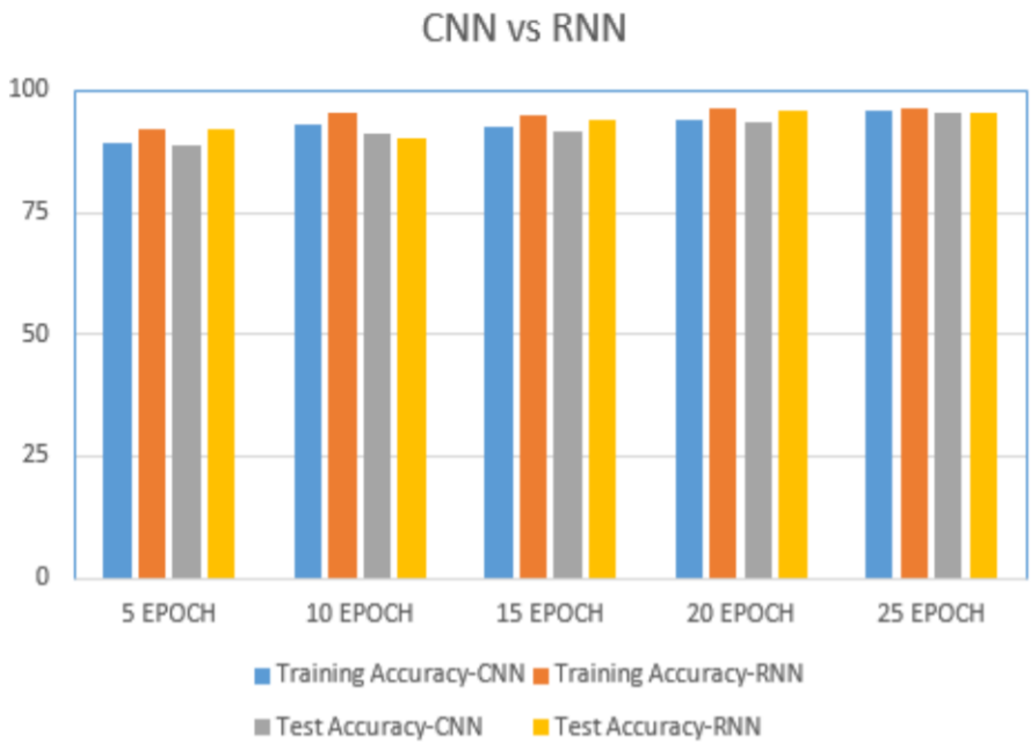
In deep learning, sometimes we use a transfer learning approach in which instead of making a scratched CNN model for the image classification problem, a pre-trained CNN model that is already modeled on a huge benchmark datasets.

Result Analysis

The performance measures such as training accuracy, training loss, test accuracy and test loss while implementing CNN architecture and RNN architecture is given it illustrates the comparison of accuracy and loss between CNN based model and RNN based model. By analysing the CNN and RNN architecture, RNN gives more accuracy when compared to CNN, but training time taken by RNN based model is higher than CNN based model.


Performance of CNN
100
75
50
25
0
5 EPOCH
10 EPOCH
15 EPOCH
20 EPOCH
25 EPOCH
Training Accuracy
Training loss
Test Accuracy
Test loss


Performance of RNN
100
75
50
25
0
5 EPOCH
10 EPOCH
15 EPOCH
20 EPOCH
25 EPOCH
Training Accuracy
Training loss
Test Accuracy
Test loss



Evaluation indexes of the segmentation results of the Two models:

Method	**DSC**	**Sensitivity**	**Specificity**
CNN	0.8869	0.9152	0.9657
RNN	0.8705	0.9001	0.9586

CONCLUSION

Deep learning based algorithms are proposed to automate feature extraction for brain tumor detection from MRI images. The proposed work increased the accuracy and reduced the loss when compared to the existing system. Network that resulted in the highest accuracy during testing has been selected and used as the classifier for brain tumour detection.