JOURNAL OF CRITICAL REVIEWS

ISSN- 2394-5125 VOL 7, ISSUE 06, 2020

A CNN Based Efficient Brain Tumor Detection using MRI

Ankit Kumar, Amit Kumar Thakur, Prateek Kumar, Mansu Mandal, Sumana Kundu

Department of Computer Science and Engineering (Data Science), Haldia Institute of Technology, India

ABSTRACT

In this paper, the proposed system can detect tumors in human brains via MRI Images of brain with focus on high and low shades of the MRI. Convolutional Neural Network has been used as a learning algorithm, as it is best suited establishing correlation between different features. There are three main steps involved in the system. In the first step, preprocessing of MRI Images takes place and the images are cropped and resized. In this step Augmentation is also performed in order to generate multiple images from limited number of images by flipping, rotating, zooming and shifting image from its axis. In the next step model is built having several layers of Convolutional Neural Network. Various Layers are added to increase accuracy and prevent over-fitting as well as to prevent loss of data. In last step after model being trained, prediction is being done on a given image. This system also consists of a utility function to decrease model train time in order to provide better efficiency.

Keywords: CNN, MRI, Preprocessing, Tumor Detection.

1. INTRODUCTION

Cancer is one of the most deadly diseases that can be fatal to life. Most of the cases of cancer are caused due to tumors. The tumors in the brain can be more common than any other part of the body, hence leading to Cancer. The International Association of Cancer Registries (IARC) reported that there are over 28,000 cases of brain tumors reported in India each year and more than 24,000 people reportedly die due to brain tumors annually. A brain tumor is a serious condition and can be fatal if not detected early and treated. When benign or malignant tumors grow, they can cause the pressure inside your skull to increase. This can cause brain damage, and it can be life- threatening.

The exact cause of brain cancer is unknown. However, factors that can increase your risk of brain cancer include exposure to high doses of ionizing radiation and a family history of brain cancer.

The cases of brain tumors are increasing day by day hence diagnosis of them manually has become a tedious task and moreover the due to human error some cases remain unobserved hence untreated. To overcome this error, the proposed system have developed that could detect tumors in brain efficiently through MRI images. There are also several data available already of MRI of brain tumors, hence can be used for building the algorithm.

The conventional method for tumor detection in magnetic resonance brain images is human inspection. The observation from humans in predicting the tumor may mislead due to noise and distortions round in the image. This method is impractical for large amount of data and also very time consuming. So, automated tumor detection methods are developed as it would save radiologists. Tumor is identified in brain MRI using deep Learning and Machine Learning Algorithms.

These proposed works are divided into three sections.

1. Preprocessing steps are applied on the brain MRI images.
2. Texture features are extracted using Gray Level Co-occurrence Matrix (GLCM).
3. Classification is performed using Deep Learning Algorithm.

2. RELATED WORK

Now a day, automatic brain tumor detection is highly demanding in the realm of medical science. Researchers

ISSN- 2394-5125 VOL 7, ISSUE 06, 2020

already started to seek different effective brain tumor detection methods. In [1], the study of some types of brain tumors such as metastatic bronchogenic carcinoma tumors, glioblastoma and sarcoma were performed using brain MRI. The detection and classification of MRI brain tumors were implemented using different wavelet transforms and support vector machines. An automatic instance semantic segmentation method was proposed in [2] based on deep neural networks. This end-to-end multitask learning architecture comprising three stages, namely detection, segmentation, and classification. In this paper, the tumor detection was based on high-level extracted features from convolutional neural networks (CNNs) using the Hough transform technique. In [3], the brain tumor severity was estimated using Convolutional Neural Network algorithm. In [4], the authors discussed the Histogram segmentation method of the MRI Images. The scanned human brain MRI images were taken from the database. In [5], an automatic brain tumor detection method was proposed by using Convolutional Neural Networks (CNN) classification. The deeper architecture design was performed by using small kernels. The weight of the neuron was given as small. In [6,] the authors have concentrated on noise removal technique, extraction of gray-level co-occurrence matrix (GLCM) features, DWT-based brain tumor region growing segmentation to reduce the complexity and improve the performance. This was followed by morphological filtering which removes the noise that can be formed after segmentation. The probabilistic neural network classifier was used to train and test the performance accuracy in the detection of tumor location in brain MRI images. A fully automatic segmentation of brain tumour using convolutional neural network was proposed in [7]. Further, it used high grade gilomas brain image from BRATS 2015 database. A method was proposed in [8], to extract brain tumor from MRI images by Fuzzy C-Means clustering algorithm which was followed by traditional classifiers and CNN. The experimental study was carried on a real-time dataset with diverse tumor sizes, locations, shapes, and different image intensities. In traditional classifier part, six traditional classifiers were applied namely Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Multilayer Perceptron (MLP), Logistic Regression, Naïve Bayes and Random Forest which was implemented in scikit-learn. Afterward, CNN was implemented using Keras and Tensorflow. An automatic brain tumor diagnostic system from MR images was proposed in [9]. This system consists of three stages to detect and segment a brain tumor. In the first stage, MR image of brain was acquired and preprocessing was done to remove the noise and to sharpen the image. In the second stage, global threshold segmentation was done on the sharpened image to segment the brain tumor. In the third stage, the segmented image was post processed by morphological operations and tumor masking in order to remove the false segmented pixels. An image segmentation method was proposed in [10] to indentify or detect tumor from the brain magnetic resonance imaging (MRI). This paper proposed a set of image segmentation algorithms.

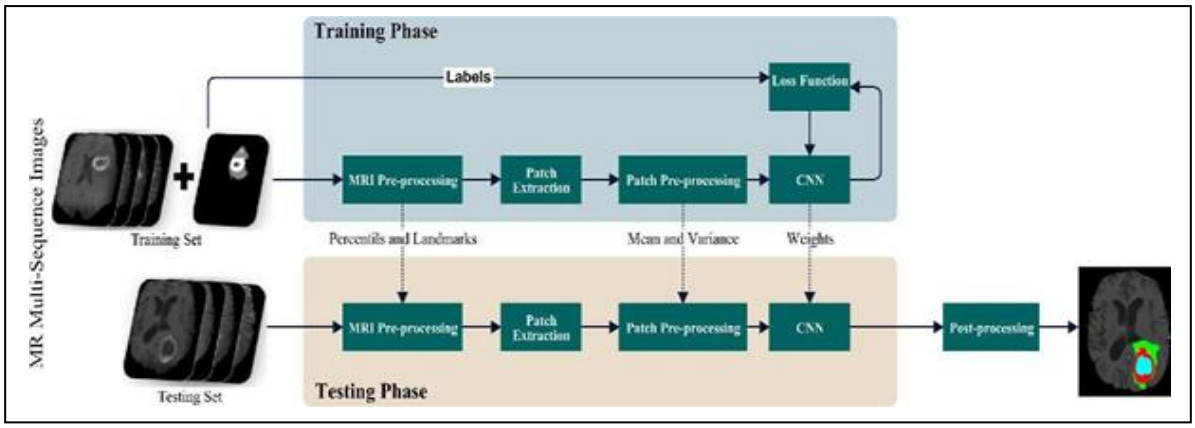
Considering the necessity of developing an efficient and automatic brain tumor detection system, which may ease the detection of the disease and fruitful for both doctors and patients, we developed an efficient brain tumor detection system.

3. SYSTEM OVERVIEW & APPROACH

The brain tumor is cancerous or maybe non-cancerous mass or abnormal cell growth in the brain. Abnormal cell growth in the brain results in brain tumor and affects a person's life. The early and accurate detection of such disease can help the patient in medical healing. Imaging is an important side of bioscience is to picturize the diagnosed structures or shape of the human body, which helps in medical diagnosis.

This proposed system is divided into two main parts. The first part deals with the detection of the tumor from MRI images, and the second part contains the process of classification of tumor type (Benign, Malignant or Normal). The given input MRI image will undergo a number of stages, which are pre-processing, segmentation and classification that contains the median filter, morphological operation, masking, feature extraction and CNN Model. The model that we have proposed is able to detect the affected region (tumor). The affected area will be separated using the morphological operation, which separates the affected and normal region from the given MRI image [Fig.1].

ISSN- 2394-5125 VOL 7, ISSUE 06, 2020

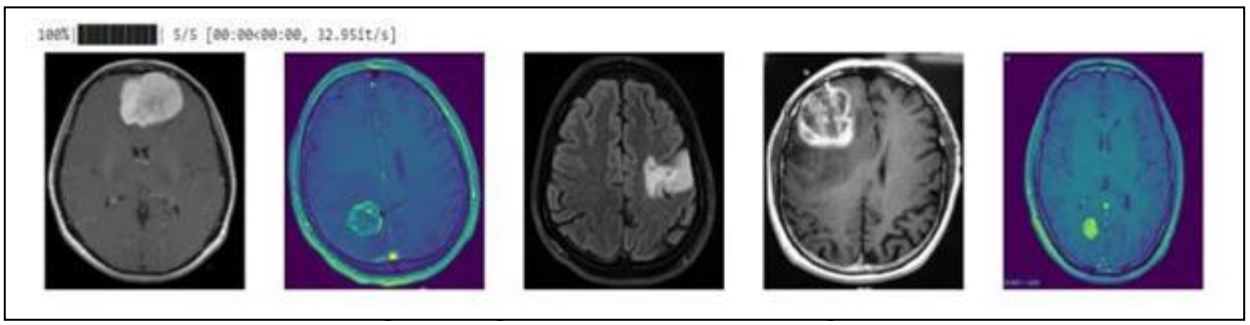


3.1. Preprocessing

The real-time images that have not been processed may contain some noise or distortion, and hence, the image has to be enhanced before applying the algorithm to get better performance further. The goal of the pre-processing is to improve theimagequalitythathelpsforfurtherprocessing.Pre-processinghelpsinreducingthe complexity and increases accuracy. There are many methods out there in order to clear the noise and to obtain the smooth image we have considered the median filter and Gaussian high pass filter approach.

3.1.1. Resizing and Cropping

The MRI images are resized and cropped to (224, 224, 3) size. Both tumorous and non-tumorous images are appended in an array and shuffled for better training. Moreover a label array is also made in order to determine which image is tumorous and which not. After preprocessing i.e. after resizing the images to (224, 224, 3) the tumors dataset MRI looks as follows [Fig. 2]:



After preprocessing i.e. after resizing the images to (224, 224, 3) the non tumors dataset MRI looks as follows [Fig. 3]:

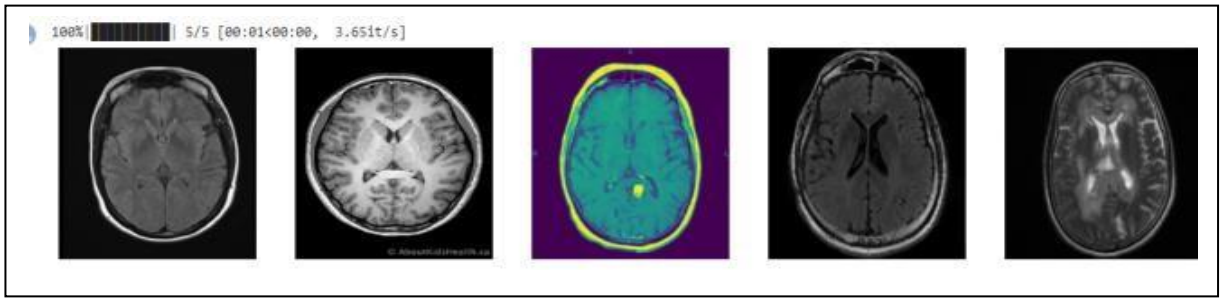


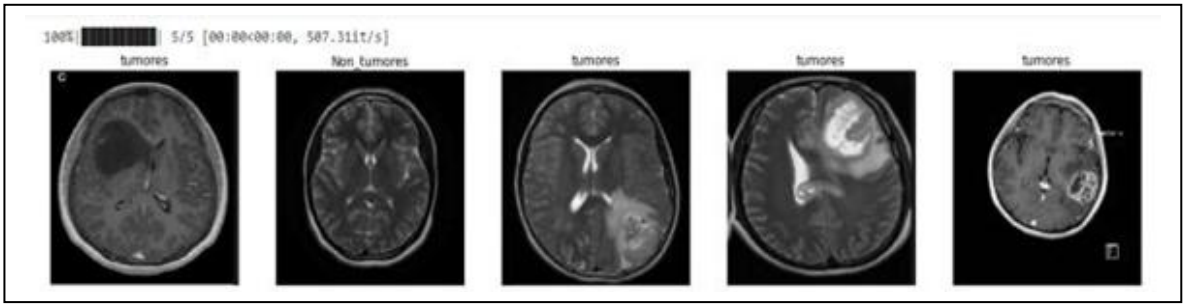
Fig3.Viewingdataset–NonTumorsafterresizing

ISSN- 2394-5125 VOL 7, ISSUE 06, 2020

3.1.2. Storing and labeling

The images after being resized and cropped are stored in an array by appending them one by one and a label is assigned corresponding to each image which makes it easier to distinguish tumors with non-tumors dataset or MRI images. After this step images are also shuffled so as to give a bit of randomness. After this Array values of each image are normalized.

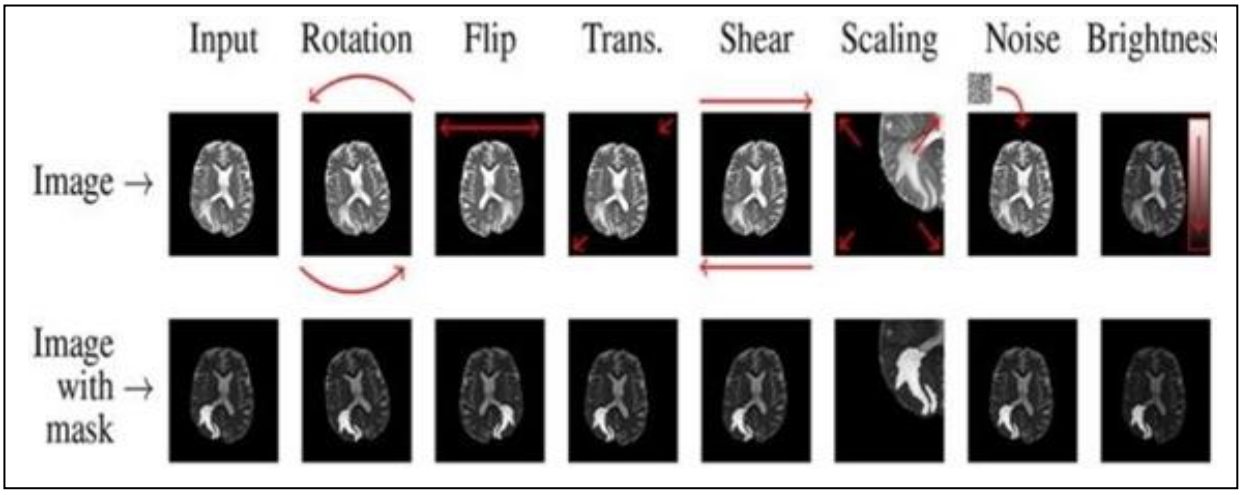
Given below are the images after performing above steps with label [Fig. 4]:



Data Augmentation is a method to generate multiple images from limited number of available datasets. Images are generated by performing various operations on image. In this system we have used ImageDataGenerator library to perform data augmentation.

Some of the most common data augmentation techniques used for images are [Fig. 5]:

(1) Position augmentation (Scaling, Cropping, Flipping, Padding, Rotation, Translation, Affine Transformation)
(2) Color augmentation (Brightness, Contrast, Saturation, Hue)



3.1.4. Splitting of dataset (Holdout Method)

The dataset is split for training and testing the model that has been built considering the Holdout method [Table 1]. i.e. images belongs to training data set are not used for testing. Similarly, images belongs to testing database are not used to train the system.

Table 1. Summary of dataset after being split

Parameter	**Training**	**Testing**
Percentage (%)	70	30
Shape	(177, 224, 224, 3)	(76, 224, 224, 3)

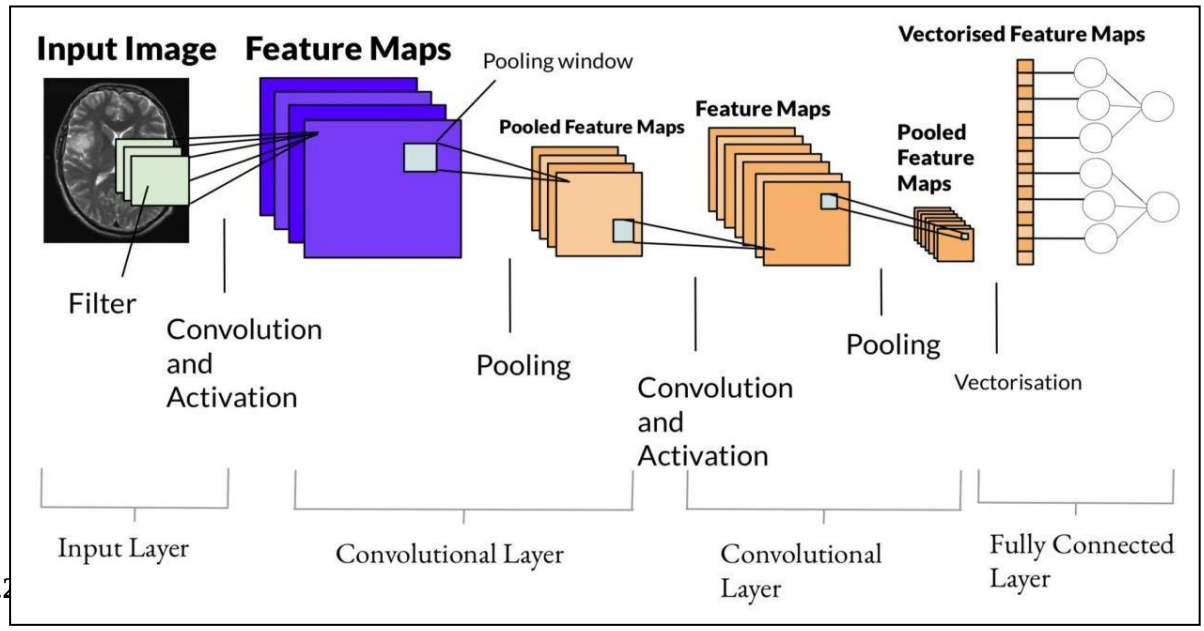
ISSN- 2394-5125 VOL 7, ISSUE 06, 2020

3.2. Model Architecture and Building

The Proposed system has been developed based on Convolutional Neural Network (CNN).

3.2.1. System Architecture

A Convolutional Neural Network (CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a Convolutional Network is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, Convolutional Network has the ability to learn these filters/characteristics [Fig. 6].



3.2

- **Input Layer:**
  The input layer of a neural network is composed of artificial input neurons, and brings the initial data into the system for further processing by subsequent layers of artificial neurons. Here, in the developed system all the MRI images of training dataset are fed in the input layer.
- **Zero Padding Layer:**
  Each convolutional layer has some number of filters that we define, and we also define the dimension of these filters as well. We also showed how these filters convolve image input. When a filter convolves a given input channel, it gives us an output channel. This output channel is a matrix of pixels with the values that were computed during the convolutions that occurred on the input channel. Zero padding is a technique that allows us to preserve the original input size. This is something that we specify on a per-convolutional layer basis. With each convolutional layer, just as we define how many filters to have and the size of the filters, we can also specify whether or not to use padding.
- **Batch-Normalization Layer:**
  Batch normalization applies a transformation that maintains the mean output close to 0 and the output standard deviation close to 1.
  Importantly, batch normalization works differently during training and during inference.

During training (i.e. when using fit() or when calling the layer/model with the argument training=True), the layer normalizes its output using the mean and standard deviation of the current batch of inputs. That is to say, for each channel being normalized, the layer returns gamma * (batch -

ISSN- 2394-5125 VOL 7, ISSUE 06, 2020

mean(batch)) / sqrt(var(batch) + epsilon) + beta, where:

- epsilon is small constant (configurable as part of the constructorarguments)
- gamma is a learned scaling factor (initialized as 1), which can bedisabled by passing scale=False to the constructor.
- beta is a learned offset factor (initialized as 0), which can be disabled bypassing center=False to the constructor.

During inference (i.e. when using evaluate() or predict() or when calling the layer/model with the argument training=False (which is the default), the layer normalizes its output using a moving average of the mean and standard deviation of the batches it has seen during training. That is to say, it returns gamma * (batch - self.moving_mean) / sqrt(self.moving_var + epsilon) + beta.

self.moving_mean and self.moving_var are non-trainable variables that are updated each time the layer in called in training mode, as such:

- moving_mean = moving_mean * momentum + mean(batch) * (1 -momentum)
- moving_var = moving_var * momentum + var(batch) * (1 - momentum)

As such, the layer will only normalize its inputs during inference after having been trained on data that has similar statistics as the inference data.

- **Max Pooling Layer:**
  Max pooling is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. Thus, the output after max-pooling layer would be a feature map containing the most prominent features of the previous feature map.

- **Drop Out Layer:**
  Dropout is implemented per-layer in a neural network. It can be used with most types of layers, such as dense fully connected layers, convolutional layers, and recurrent layers such as the long short-term memory network layer. This Layer prevents over fitting problem.
- **Flatten Layer:**
  Once the pooled featured map is obtained, the next step is to flatten it. Flattening involves transforming the entire pooled feature map matrix into a single column which is then fed to the neural network for processing.
- **Dense Layer:**
  Dense layer is the regular deeply connected neural network layer. It is most common and frequently used layer. Dense layer does the below operation onthe input and return the output.
  output = activation(dot(input, kernel) + bias).

- **Loss Function used:**

  In this system **Binary Cross Entropy or Log** Loss function has been used. Binarycross entropy compares each of the predicted probabilities to actual class output which can be either 0 or 1. It then calculates the score that penalizes the probabilities based on the distance from the expected value. That means how closeor far from the actual value.

$$\mathbf{Log\ loss} = \frac{\mathbf{1}}{\mathbf{N}}\sum_{\mathbf{i=1}}^{\mathbf{N}} -(\mathbf{y_i} * \mathbf{log}(\mathbf{p_i}) + (\mathbf{1} - \mathbf{y_i}) * \mathbf{log}(\mathbf{1} - \mathbf{p_i}))$$

  Here, p_i is the probability of class 1, and $(1-p_i)$ is the probability of class 0.

  When the observation belongs to class 1 the first part of the formula becomes active and the second part vanishes and vice versa in the case observation's actual class are 0. This is how we calculate

ISSN- 2394-5125 VOL 7, ISSUE 06, 2020

the Binary cross-entropy.

- **Activation Function used:**

  Activation function used is **Sigmoid.** The main reason why we use sigmoid function is because it exists between **0 to 1.** Therefore, it is especially used for models where we have to **predict the probability** as an output. Sinceprobability of anything exists only between the range of **0 and 1,** sigmoid isthe right choice.

3.2.3. Model Summary

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
zero_padding2d (ZeroPadding2 (None, 228, 228, 3)       0
_________________________________________________________________
conv2d (Conv2D)              (None, 113, 113, 32)      896
_________________________________________________________________
batch_normalization (BatchNo (None, 113, 113, 32)      128
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 56, 56, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 56, 56, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 27, 27, 64)        18496
_________________________________________________________________
batch_normalization_1 (Batch (None, 27, 27, 64)        256
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 13, 13, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 13, 13, 64)        0
_________________________________________________________________
```

ISSN- 2394-5125 VOL 7, ISSUE 06, 2020

```
 flatten (Flatten) (None, 10816)                                    0

 dense (Dense)     (None,                                           128)
1384576


batch_normalization_2   (Batch      (None,      128) 512

dropout_2 (Dropout)     (None, 128) 0

dense_1 (Dense)   (None, 1)
129
=================================================================
========
Total params: 1,404,993
Trainable params: 1,404,545
Non-trainable params: 448
```

4. RESULTS AND DISCUSSIONS

Table 2. Performance Evaluation of the proposed system

Training Accuracy	85.40 %
Validation Accuracy	84.00 %
Test Accuracy	83.74 %
Training Time	24696 ms

Model Prediction

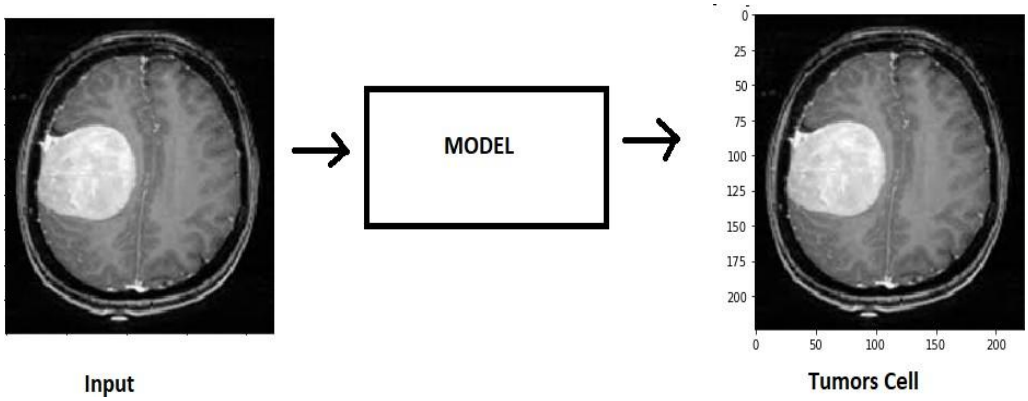


Fig. 7. Identification Testing

After the model is trained we can use it for prediction. Training of model takes less time comparatively as

ISSN- 2394-5125 VOL 7, ISSUE 06, 2020

we have used a utility function i.e. callback function that stops training the model when accuracy gets stagnated. When accuracy does not get improved further the training stops before the givenepoch iteration count. Hence the model takes less time to get ready for prediction. For prediction we can give any image (MRI) and the model then predicts its class and based on that labelled output is shown [Fig. 7]. Our model accuracy can further be enhanced by training with more number of datasets andperforming certain techniques like cross validation in order to find the best fit parameter for the model layers.

5. CONCLUSIONS

We made a classification model with the help of custom CNN layers to classify whether the patient has a brain tumor or not through MRI images. If we increase the training data may be by more MRI images of patients or perform data augmentation techniques we can achieve higher classification accuracy. Also, we can make use of pre-trained architectures like Vgg16 or Resnet 34 for improving the model performance.

REFERENCES

[1] M. Gurbin, M. Lascu, D. Lascu, "Tumor Detection and Classification of MRI Brain Image using Different Wavelet Transforms and Support Vector Machines", 2019 42nd International Conference on Telecommunications and Signal Processing (TSP), Budapest, Hungary, 2019, pp. 505-508.

[2] M. Rezaei, H. Yang, C. Meinel, "Instance Tumor Segmentation using Multitask Convolutional Neural Network", 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, 2018, pp. 1-8.

[3] H. Badisa, M. Polireddy, A. Mohammed, "CNN Based Brain Tumor Detection", International Journal of Engineering and Advanced Technology (IJEAT), Volume8 Issue-4, April 2019, pp. 1731 – 1734.

[4] MB Bramarambika, Seshashayee, "Brain Tumor Detection and Identification Using Histogram Method", International Journal of Innovative Technology and Exploring Engineering (IJITEE), Volume-8 Issue-10, August 2019, pp. 3517 – 3521.

[5] J. Seetha, S. Selvakumar Raja, "Brain Tumor Classification Using Convolutional Neural Networks", Biomedical & Pharmacology Journal, September 2018, Vol. 11(3), p. 1457-1461.

[6] N. Varuna Shree, T.N.R. Kumar, "Identification and classification of brain tumor MRI images with feature extraction using DWT and probabilistic neural network", Brain Inf. 5, pp. 23–30 (2018), https://doi.org/10.1007/s40708-017-0075-5.

[7] M Malathi, P Sinthia, "Brain Tumour Segmentation Using Convolutional Neural Network with TensorFlow", Asian Pacific journal of cancer prevention, APJCP (2019), DOI: 10.31557/APJCP.2019.20.7.2095

[8] T. Hossain, F. S. Shishir, M. Ashraf, MD A.l Nasim, F. M. Shah, "Brain Tumor Detection Using Convolutional Neural Network", 1st International Conference on Advances in Science, Engineering and Robotics Technology 2019 (ICASERT 2019), Dhaka, Bangladesh, 2019, pp. 1-6.

[9] M. U. Akram, A. Usma, "Computer Aided System for Brain Tumor Detection and Segmentation", International Conference on Computer Networks and Information Technology, Abbottabad, 2011, pp. 299-302.

[10] V. S. Shirsat, S. S. Kawathekar, "Brain Tumor Detection Using MRI Image Analysis", International Conference on Ubiquitous Computing and Multimedia Applications UCMA 2011: Ubiquitous Computing and Multimedia Applications pp 307-314.