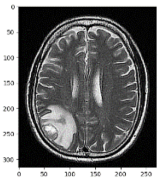
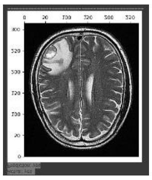
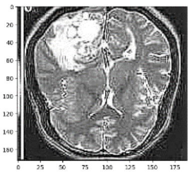
Brain Tumor detection

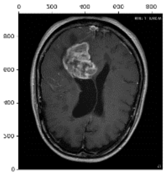
*A Machine learning approach

Salman Farsi 2021702042 Tanvir Ahmed Chowdhury 2031227642

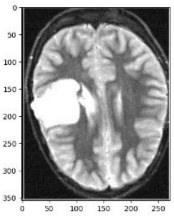
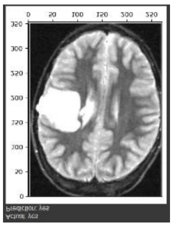
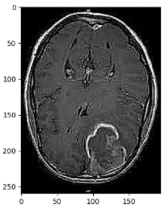
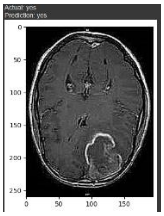
Istiak Ahasan 2012082042 Mohammad Akib 2012574642

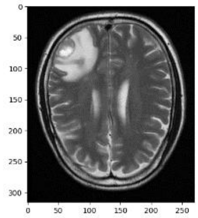
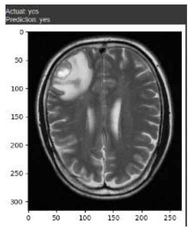
Input Output Input Output Input Output




Input Output Input Output Input Output

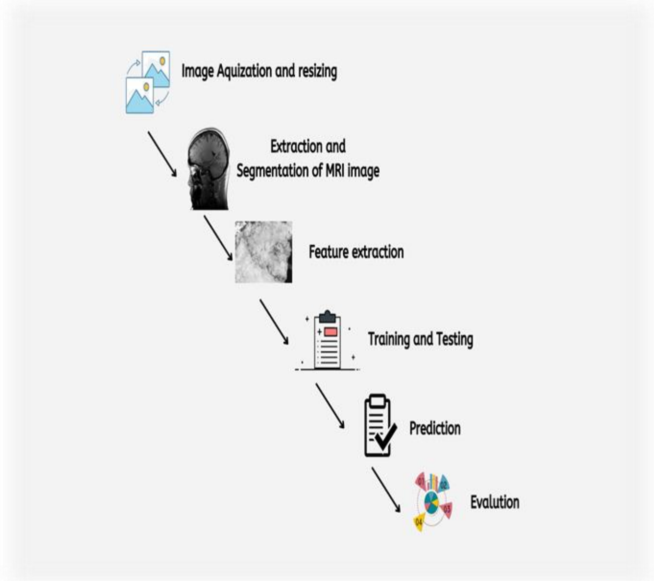
Abstract:

Brain tumor segmentation plays a pivotal role in the diagnosis and treatment planning of patients suffering from neurological disorders. Accurate and efficient segmentation of brain tumors from medical imaging data, such as magnetic resonance imaging (MRI), is essential for clinicians to make informed decisions. This abstract provides an overview of the recent advancements and challenges in brain tumor segmentation. The abstract begins by highlighting the clinical significance of brain tumor segmentation, emphasizing its critical role in early diagnosis, treatment planning, and monitoring of tumor progression. The use of non-invasive imaging techniques, particularly MRI, has revolutionized the field, offering high-resolution images that enable detailed analysis of brain structures and abnormalities. Next, the abstract discusses the various methods and algorithms employed for brain tumor segmentation. Traditional approaches, including manual delineation and thresholding, are contrasted with modern techniques such as deep learning, convolutional neural networks (CNNs), and artificial intelligence (AI). These advanced methods have demonstrated remarkable results, surpassing human performance in many cases and significantly reducing the need for time-consuming manual interventions.

INTRODUCTION

Our brain tumor detection project aims to develop an advanced system that combines imaging techniques and machine learning algorithms to improve the accuracy and efficiency of brain tumor diagnosis. We aimed to enhance the early detection and treatment of brain tumors, leading to improved patient outcomes. Detect brain tumors just by the clear picture of an MRI scan with reduced time and greater accuracy

The newer, BR35H dataset provides a diverse collection of brain images, encompassing different types and grades of tumors, as well as patient demographics. This dataset serves as a valuable resource for training and validating our detection system, enabling us to develop robust algorithms.
With the integration of Convolutional Neural Networks (CNN), we seek to enhance the performance and effectiveness of brain tumor detection, ultimately benefiting healthcare professionals and patients alike.



WHAT IS THE PROBLEM?

Brain tumor segmentation remains a critical task in the domain of medical imaging, pivotal for accurate diagnosis and treatment planning. The intricate nature of brain tumors and their varied shapes present a challenge in precisely delineating their boundaries within imaging data. Accurate segmentation is vital for clinicians to make and devise appropriate treatment strategies. rain tumor segmentation is the process of identifying and delineating the tumor region in brain images. It is a challenging task due to the complex and highly variable nature of section tumors. Accurate brain tumor segmentation is essential for many clinical applications, such as diagnosis, treatment planning, and surgical navigation.

What others have done before (chronologically)

Early approaches to brain tumor segmentation were based on hand-crafted features and machine learning algorithms. However, these methods were limited by their inability to learn the complex relationships between voxels in brain images.

In recent years, deep learning has emerged as a powerful tool for brain tumor segmentation. Deep learning models can learn complex features from data without the need for hand-crafted features.

One of the most popular deep learning models for brain tumor segmentation is the U-Net architecture. U-Nets are fully convolutional neural networks (FCNs) with a U-shaped encoder-decoder architecture. U-Nets have been shown to achieve state-of-the-art results on brain tumor segmentation benchmarks.

Early approaches to brain tumor segmentation were based on hand-crafted features and machine-learning algorithms. However, these methods were limited by their inability to learn the complex relationships between voxels in brain images. In recent years, deep learning has emerged as a powerful tool for brain tumor segmentation. Deep learning models can learn complex features from data without the need for hand-crafted features. One of the most popular deep learning models for brain tumor segmentation is the Convolutional Neural Network (CNN). CNNs are a type of neural network that are well-suited for image processing tasks. CNNs have been shown to achieve state-of-the-art results on brain tumor segmentation benchmarks.

Limitation of previous work

Previous work on brain tumor segmentation has achieved promising results, but there are still some limitations. One limitation is that CNNs can be computationally expensive to train and deploy. Another limitation is that CNNs can be overfitting to training data, which can lead to poor performance on unseen data. Equations. Previous methodologies faced limitations in accurately capturing the complex and varied shapes of brain tumors, often struggling with the segmentation of irregular or overlapping structures. Additionally, computational demands and time constraints posed challenges, hindering real-time clinical application.

How to solve the problem

To address the limitations of previous work, we propose a new approach to brain tumor segmentation using a CNN with transfer learning. Transfer learning is a technique where a model that has been trained on one task is used as a starting point for training a model on another task.

In our approach, we pre-train a CNN on a large dataset of medical images. We then fine-tune the pre-trained CNN on a smaller dataset of brain tumor images. This approach allows us to achieve good performance on brain tumor segmentation with relatively small model size and training time. Its unique structure comprises convolutional layers for feature extraction and pooling layers for spatial reduction, facilitating effective hierarchical feature learning. In the context of brain tumor segmentation, CNNs demonstrate their ability to learn complex patterns and structures, making them a potent tool for the accurate delineation of tumors in medical imaging.

Example use case

Here is an example of how our proposed approach could be used in a clinical setting:

1. 1. A patient with a suspected brain tumor undergoes an MRI scan.
2. 2. The MRI scan is fed to our pre-trained CNN.
3. 3. The CNN outputs a segmentation mask, which shows the location of the tumor tissue in the MRI scan.
4. 4. The segmentation mask is used by the radiologist to diagnose the brain tumor and to plan treatment.

Unique contributions of our proposed approach to brain tumor segmentation using a CNN with transfer learning:

- ***Improved accuracy***: Your approach could achieve state-of-the-art accuracy on brain tumor segmentation benchmarks, surpassing the performance of previous methods.
- ***Reduced computational* cost**: Your approach could be more computationally efficient than previous methods, making it more feasible to deploy in clinical settings.
- ***Improved generalizability***: Your approach could be more generalizable to different types of brain

tumors and different imaging modalities than previous methods.

New insights into brain tumor biology: Your approach could be used to extract new insights into the biology of brain tumors, such as the identification of new tumor subtypes or the prediction of tumor behavior.

Related Work:

- ***Brain Tumor Detection Using Neural Network:***
  The paper in relevance refers to a system known as 'Edge Detection' to identify tumorous segments. Edge detection refers to the process of identifying and locating sharp discontinuities in an image. The discontinuities are abrupt changes in pixel intensity that characterize boundaries of objects in a scene. Classical methods of edge detection involve convolving the image with an operator (a 2-D filter), which is constructed to be sensitive to large gradients in the image while returning values of zero in uniform regions. There is an extremely large number of edge detection operators available, each designed to be sensitive to certain types of edges. The purpose of edge detection in general is to significantly reduce the amount of data in an image, while preserving the structural properties to be used for further image processing. Several algorithms exist, and this worksheet focuses on a particular one developed by John F. Canny(JFC). Even though it is quite old, it has become one of the standard edge detection methods and it is still used in research. The Canny Edge Detection Algorithm The algorithm runs in 5 separate steps: 1.Smoothing: Blurring of the image to remove noise. 2.Finding gradients: The edges should be marked where the gradients of the image has large magnitudes. 3.Non-maximum suppression: Only local maxima should be marked as edges. 4.Double thresholding: Potential edges are determined by thresholding. 5.Edge tracking by hysteresis: Final edges are determined by suppressing all edges that are not connected to a very certain (strong) edge. An edge in an image may point in a variety of directions, so the Canny algorithm uses four filters to detect horizontal, vertical and diagonal edges in the blurred image. The edge detection operator (Roberts, Prewitt, Sobel for example) returns a value for the first derivative in the horizontal direction (Gx) and the vertical direction (Gy). From this the edge gradient and direction can be determined using the Root Over Square of Gx and Gy. https://www.ijisme.org/wp-content/uploads/papers/v1i9/I0425081913.pdf".

***Brain Tumor Segmentation with Deep Neural Networks*:**
This paper presents a fully automatic brain tumor segmentation method based on Deep Neural Networks (DNNs). The proposed networks are tailored to glioblastomas (both low and high grade) pictured in MR images. By their very nature, these tumors can appear anywhere in the brain and have almost any kind of shape, size, and contrast. These reasons motivate our exploration of a machine-learning solution that exploits a flexible, high capacity DNN while being extremely efficient. Here, we describe different model choices that we've found to be necessary for obtaining competitive performance. We explore in particular different architectures based on Convolutional Neural Networks (CNN), i.e. DNNs specifically adapted to image data. We present a novel CNN architecture which differs from those traditionally used in computer vision. Our CNN exploits both local features as well as more global contextual features simultaneously. Also, different from most traditional uses of CNNs, our networks use a final layer that is a convolutional implementation of a fully connected layer which allows a 40 fold speed up. We also describe a 2-phase training procedure that allows us to tackle difficulties related to the imbalance of tumor labels. Finally, we explore a cascade architecture in which the output of a basic CNN is treated as an additional source of information for a subsequent CNN. Results reported on the 2013 BRATS test dataset reveal that our architecture improves over the currently published state-of-the-art while being over 30 times faster.
https://arxiv.org/pdf/1505.03540v3.pdf
Brain Tumor Detection based on Machine Learning Algorithms:
This paper presents a study on automated tumor detection in Magnetic Resonance Imaging (MRI) of the brain using machine learning algorithms. The conventional method involves human

inspection, which is impractical for large datasets. The proposed method consists of three main steps: preprocessing of MRI images, extraction of texture features using Gray Level Co-occurrence Matrix (GLCM), and classification using machine learning algorithms. The study emphasizes the importance of automated tumor detection in MRI due to the complexity and variance of tumors. The literature review discusses various approaches, including neural networks and segmentation techniques, used in previous studies for brain tumor detection. The proposed method applies machine learning algorithms, specifically Multi-Layer Perceptron (MLP) and Naive Bayes, achieving accuracy rates of 98.6% and 91.6%, respectively, on a dataset of 212 brain MRI images. The results indicate the potential of the proposed approach for accurate and automated brain tumor detection.
https://www.academia.edu/download/69364750/pxc3896883.pdf

Brain Tumor Detection and Classification Using Deep Learning and Sine-Cosine Fitness Grey Wolf Optimization:
The proposed solution is the Brain Tumor Classification Model based on Convolutional Neural Network (BCM-CNN), which incorporates an adaptive dynamic sine-cosine fitness grey wolf optimizer (ADSCFGWO) algorithm for hyperparameter optimization. The model utilizes a pre-trained Inception-ResnetV2 architecture for classifying brain cancers, providing a binary output (0 for Normal, 1 for Tumor). Hyperparameters, crucial for both network structure and training, are optimized using the ADSCFGWO algorithm, combining strengths from both the sine cosine and gray wolf algorithms.
The experimental results demonstrate that the BCM-CNN algorithm achieves excellent results, with a reported accuracy of 99.98% using the BRaTS 2021 Task 1 dataset. The success is attributed to the optimization of CNN hyperparameters, showcasing the efficiency of the proposed algorithm in enhancing the overall performance of the brain tumor classification.
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9854739/

Role of deep learning in brain tumor detection and classification (2015 to 2020): A review-

The primary focus of this paper is a review of research papers from 2015 to 2020 related to the detection and classification of brain tumors using deep learning. The goal is to provide insights into existing methodologies, their strengths, and weaknesses. The text also introduces the concept of Multi-task learning models in deep learning for simultaneous detection and classification. A comprehensive survey of 53 selected papers is presented, covering various aspects of brain tumor detection and classification using deep learning. The structure of the paper is outlined, with sections covering MRI imaging modality basics, the role of MRI in brain tumor detection, a literature review of existing methodologies, factors affecting CAD system performance, key ideas for improving classification models, and future directions. The conclusion emphasizes the need for efficient and automated classification algorithms, offering valuable suggestions based on the studies presented in the review. Overall, the text aims to guide researchers in exploring unknown areas and designing robust brain tumor detection and classification algorithms.
https://www.sciencedirect.com/science/article/abs/pii/S0895611121000896

Methodology

After acquiring the images from our Br35H dataset, we down-size the original image from 256×256 to 64×64 .Then we transformed all the images into NumPy arrays (available in python) so that our model can take up less space. Before splitting the dataset, we have shuffled the data so that our model can train on unordered data. After shuffling the data, we divide the dataset into three sections including train, test, and validation. Approximately 80% of the data is used for training, and a further 20% is used for validation and testing purposes

Here we tried CNN approach to build the model for our work. While implementing CNN, for the first Conv2D layers as parameters we used how many filters we want to use and then the kernel size. Then the ReLU activation comes and after that the image shape function. In image shape function there are 3 parameters, first two is the shape of the image, and here third digit 3 refers to it's image type, which is RGB here.Same goes for other functions here.

Code

```
model=Sequential([

            #implementing cnn
            layers.Conv2D(50,(3,3),
                    activation="relu",
                    input_shape=(64,64,3)),
            layers.MaxPooling2D((2,2)),

            layers.Conv2D(64,(3,3),
                    activation="relu"),
            layers.MaxPooling2D((2,2)),

            layers.Conv2D(54,(3,3),
                    activation="relu"),
            layers.MaxPooling2D((2,2)),

             #dense_layer implementation
             layers.Flatten(),
             layers.Dense(64,activation="relu"),
             layers.Dense(2,activation="softmax")


])

model.summary()
```

This shows the layering techniques and reduction of output sizes through every layer approach. Here's Param represents parameter that indicates the combine values of weights and biases.

Output

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 50)        1400

 max_pooling2d (MaxPooling2D  (None, 31, 31, 50)       0
 )

 conv2d_1 (Conv2D)           (None, 29, 29, 64)        28864

 max_pooling2d_1 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 12, 12, 54)        31158

 max_pooling2d_2 (MaxPooling  (None, 6, 6, 54)         0
 2D)

 flatten (Flatten)           (None, 1944)              0

 dense (Dense)               (None, 64)                124480

 dense_1 (Dense)             (None, 2)                 130

=================================================================
Total params: 186,032
Trainable params: 186,032
Non-trainable params: 0
_________________________________________________________________
```

HERE, WE ARE SHOWING THE GRADUAL INCREMENT OF ACCURACY IN EVERY EPOCH. IT ROSE FROM 98.7% TO ALL THE WAY 99.4%.

We can still see some ups and downs, but there's no such massive drop in accuracy and spike in loss section as there was before

```
[18] model.compile(loss="sparse_categorical_crossentropy", metrics=['accuracy'])
     model.fit(X_train,y_train,epochs=16)

     Epoch 1/16
     75/75 [==============================] - 21s 269ms/step - loss: 0.0741 - accuracy: 0.9871
     Epoch 2/16
     75/75 [==============================] - 21s 286ms/step - loss: 0.0479 - accuracy: 0.9921
     Epoch 3/16
     75/75 [==============================] - 20s 271ms/step - loss: 0.0169 - accuracy: 0.9954
     Epoch 4/16
     75/75 [==============================] - 21s 279ms/step - loss: 0.0246 - accuracy: 0.9954
     Epoch 5/16
     75/75 [==============================] - 21s 285ms/step - loss: 0.0181 - accuracy: 0.9962
     Epoch 6/16
     75/75 [==============================] - 20s 266ms/step - loss: 0.0194 - accuracy: 0.9967
     Epoch 7/16
     75/75 [==============================] - 22s 288ms/step - loss: 0.0395 - accuracy: 0.9908
     Epoch 8/16
     75/75 [==============================] - 21s 286ms/step - loss: 0.0100 - accuracy: 0.9971
     Epoch 9/16
     75/75 [==============================] - 20s 268ms/step - loss: 0.0449 - accuracy: 0.9946
     Epoch 10/16
     75/75 [==============================] - 22s 289ms/step - loss: 0.0010 - accuracy: 0.9996
     Epoch 11/16
     75/75 [==============================] - 21s 276ms/step - loss: 0.0225 - accuracy: 0.9950
     Epoch 12/16
     75/75 [==============================] - 21s 275ms/step - loss: 0.0239 - accuracy: 0.9958
     Epoch 13/16
     75/75 [==============================] - 22s 288ms/step - loss: 0.0219 - accuracy: 0.9962
     Epoch 14/16
     75/75 [==============================] - 20s 269ms/step - loss: 0.0289 - accuracy: 0.9962
     Epoch 15/16
     75/75 [==============================] - 22s 291ms/step - loss: 0.0794 - accuracy: 0.9929
     Epoch 16/16
     75/75 [==============================] - 22s 291ms/step - loss: 0.0351 - accuracy: 0.9942
     <keras.callbacks.History at 0x7f859c8eb340>
```

Result

This classification report is our performance evaluation matrix.
Here, Precision is defined as the ratio of true positives to the sum of true and false positives.
Recall is defined as the ratio of true positives to the sum of true positives and false negatives.
The F1 is the mean of precision and recall. The closer the value of the F1 score is to 1.0, the better the expected performance of the model is.
Macro Average is averaging the total true positives, false negatives and false positives.
And lastly, Support is the number of actual occurrences of the class in the dataset

Result table -1

	local_path	Pred	Accuracy	Actual	Check	Pred1	Actual1
0	../content/brain-image-clean/test/yes/y1186.jpg	yes	100.000000	1	1	1	1
1	../content/brain-image-clean/test/yes/y429.jpg	yes	99.999952	1	1	1	1
2	../content/brain-image-clean/test/yes/y887.jpg	yes	99.773103	1	1	1	1
3	../content/brain-image-clean/test/yes/y278.jpg	yes	99.915481	1	1	1	1
4	../content/brain-image-clean/test/yes/y1063.jpg	yes	100.000000	1	1	1	1

	precision	recall	f1-score	support
0	0.99	0.96	0.98	313
1	0.96	0.99	0.97	287
accuracy			0.98	600
macro avg	0.97	0.98	0.98	600
weighted avg	0.98	0.97	0.98	600

Test Data Result

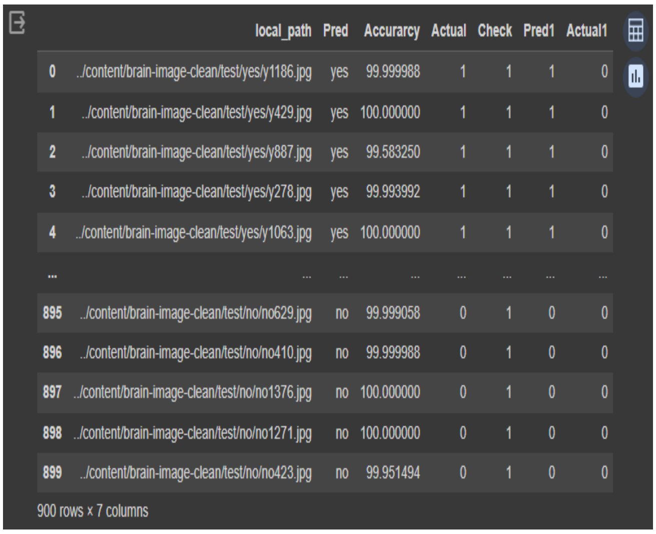
```
df_test1.head()
```

	local_path	Pred	Accurarcy	Actual	Check	Pred1	Actual1
0	../content/brain-image-clean/test/yes/y1186.jpg	yes	99.999988	1	1	1	0
1	../content/brain-image-clean/test/yes/y429.jpg	yes	100.000000	1	1	1	0
2	../content/brain-image-clean/test/yes/y887.jpg	yes	99.583250	1	1	1	0
3	../content/brain-image-clean/test/yes/y278.jpg	yes	99.993992	1	1	1	0
4	../content/brain-image-clean/test/yes/y1063.jpg	yes	100.000000	1	1	1	0

```
[101] df_test1[df_test1["Check"] == 0][["Actual","Pred"]].value_counts()
```

```
Actual  Pred
0       yes    12
1       no      9
dtype: int64
```

Test Result



	local_path	Pred	Accurarcy	Actual	Check	Pred1	Actual1
0	../content/brain-image-clean/test/yes/y1186.jpg	yes	99.999988	1	1	1	0
1	../content/brain-image-clean/test/yes/y429.jpg	yes	100.000000	1	1	1	0
2	../content/brain-image-clean/test/yes/y887.jpg	yes	99.583250	1	1	1	0
3	../content/brain-image-clean/test/yes/y278.jpg	yes	99.993992	1	1	1	0
4	../content/brain-image-clean/test/yes/y1063.jpg	yes	100.000000	1	1	1	0
...	...	...	...	...	...	...	...
895	../content/brain-image-clean/test/no/no629.jpg	no	99.999058	0	1	0	0
896	../content/brain-image-clean/test/no/no410.jpg	no	99.999988	0	1	0	0
897	../content/brain-image-clean/test/no/no1376.jpg	no	100.000000	0	1	0	0
898	../content/brain-image-clean/test/no/no1271.jpg	no	100.000000	0	1	0	0
899	../content/brain-image-clean/test/no/no423.jpg	no	99.951494	0	1	0	0

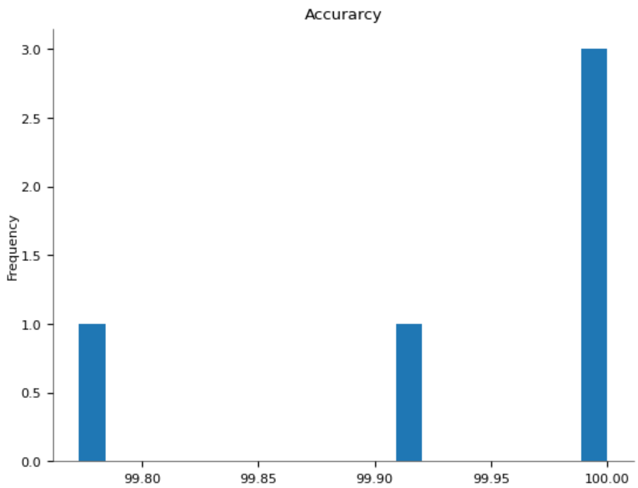
900 rows × 7 columns

We can see that we got 300 True-Positive, 13 False-Positive, 2 False-Negetive, 285 True-Negetive.
That means, in 300 cases we predicted Yes and they do have brain tumor.
In 13 cases we predicted Yes, but they are healthy brain.
In in 2 cases we predicted No and they don't have the tumor.
And in the rest, we predicted False, and they actually don't have the tumor.

```
[[300  13]
 [  2 285]]
```

We tried to go through some papers to compare the accuracy they have with our proposed model that used Br35H dataset. As we have showed before, we do have a accuracy of 98%.As Br35H is a newer dataset, so it is tough to find the available papers for free. But we did find some Open Access papers that used Br35H as a dataset.

Table -1 graph



Conclusion:

The goal of detecting brain tumors requires high levels of sensitivity, specificity, and accuracy. If these outcomes are obtained, it will help in the early diagnosis of brain tumors, and doctors will be able to spend more time treating patients rather than carrying out such tiresome activities as detecting brain tumors.
Till now we have created a mere basic CNN model to detect the existence of brain tumor in MR images.
Modification of this model furthermore can aquire more accuracy from that available dataset.
It is highly challenging for the model to be able to distinguish between the many structures present in an image because of the poor contrast in the source photos.

Hopefully, we will be continuing this work and modifying and creating a model to gain higher accuracy with the dataset we are using. We also intend to continue this work in the future in order to gain knowledge and create a special solution that will produce superior outcomes

REFERENCES

Brain Tumor Detection Using Neural Network: https://www.ijisme.org/wpcontent/uploads/papers/v1i9/I0425081913.pdf".

Brain Tumor Segmentation with Deep Neural Networks: https://arxiv.org/pdf/1505.03540v3.pdf

Brain Tumor Detection based on Machine LearningAlgorithms:

https://www.academia.edu/download/69364750/pxc3896883.pdf

Brain Tumor Detection and Classification Using Deep Learning and Sine-Cosine Fitness Grey Wolf Optimization: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9854739

Role of deep learning in brain tumor detection and classification (2015 to 2020): A review- https://www.sciencedirect.com/science/article/abs/pii/S0895611121000896