# Machine Learning Engineer Nanodegree

## Brain Tumor Classification via GoogleNet Model

Padmaraj Bhat
January 15th, 2018

## I. Definition

**Project Overview:**

Nowadays, digital images in the medical field are increasingly being used for diagnosis. Early identification of brain tumors is important to treat the tumors effectively. Given the high soft-tissue contrast and zero exposure to ionizing radiation, MRI is the most popular technique for diagnosing human brain tumors. However, brain tumor classification is not a trivial task. Automatic classification of tissue types of region of interest (ROI) plays an important role in computer-aided diagnosis. In the current study, we focus on the classification of three types of brain tumors (i.e., meningioma, glioma, and pituitary tumor) in T1-weighted contrast-enhanced MRI (CE-MRI) images.

Following are the research/links related to the chosen machine learning problem domain:

1. http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0157112 - Retrieval of Brain Tumors by Adaptive Spatial Pooling and Fisher Vector Representation This paper actually works on same dataset; but extract the tumor with different technique.

2. https://www.kaggle.com/c/data-science-bowl-2017 - Lung cancer detection.

3. https://www.kaggle.com/jboysen/mri-and-alzheimers - To predict dementia / Alzheimer’s.

**Problem Statement:**

The conventional method for MRI brain tumor detection and classification is by human inspection, which depends strongly on the experience of radiologists who review and analyze the characteristics of images. Moreover, operator-assisted classification methods are impractical for large amounts of data and are also non-reproducible. Therefore, computer-aided diagnosis tools are highly desirable to address these problems. Applications of brain tumor classification can be mainly divided into two categories: (1) classifying brain images into normal and abnormal classes, i.e., whether or not the brain images contain tumors; (2) classification within abnormal brain images, in other words, discrimination between different types of brain tumors

The project focuses on classifying the tumors in MRI images (MAT files) through GoogleNet. To talk in technical terms; this is a multi-class classification problem for image recognition problem

**Metrics**:
**Accuracy**: The project can be easily expanded with n types of tumor detection just by training model with tumors with different labels. As it is multi-class classification problem; accuracy score best fits it. F1 score is usually used for binary classification and cross entropy with logits has been already used in the code to minimize loss in the model. Moreover, as the paper already compares accuracy of different exercises on the 'ring partitions'; it would be easier for the current model to have the same metric.

## II. Analysis

In clinical settings, usually only a certain number of slices of brain CE-MRI with a large slice gap, not 3D volume, are acquired and available. A 3D model is difficult to construct with such sparse data. Hence, the proposed method (below section) is based on 2D slices. The brain T1-weighed CE-MRI dataset was acquired from Nanfang Hospital, Guangzhou, China, and General Hospital, Tianjing Medical University, China, from 2005 to 2010. There are 3064 slices from 233 patients, containing 708 meningiomas, 1426 gliomas, and 930 pituitary tumors. The images have an in-plane resolution of 512×512 with pixel size 0.49×0.49 mm2. The slice thickness is 6 mm and the slice gap is 1 mm. The tumor border was manually delineated by three experienced radiologists.

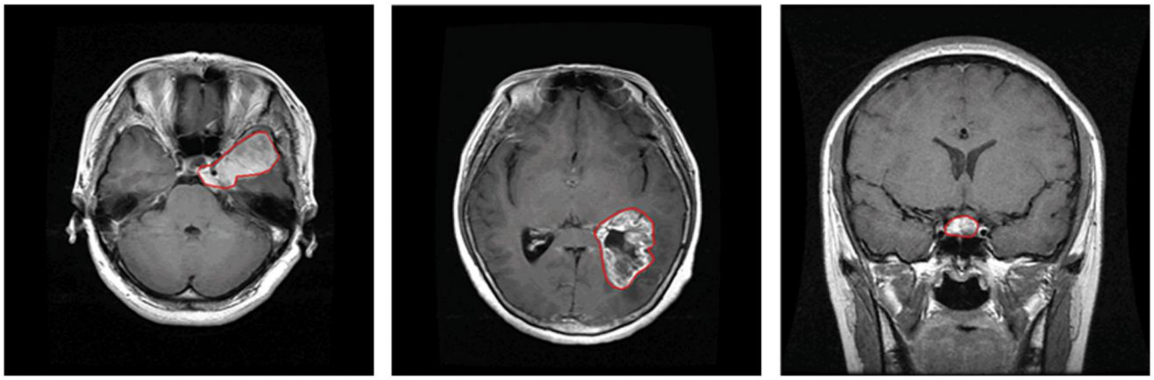
This data is organized in matlab data format (.mat file). Each file stores a structure containing the following fields for an image:

cjdata.label: 1 for meningioma, 2 for glioma, 3 for pituitary tumor
cjdata.PID: patient ID
cjdata.image: image data
cjdata.tumorBorder: a vector storing the coordinates of discrete points on tumor border.
        For example, [x1, y1, x2, y2,...] in which x1, y1 are planar coordinates on tumor border.
        It was generated by manually delineating the tumor border. So we can use it to generate
        binary image of tumor mask.
cjdata.tumorMask: a binary image with 1s indicating tumor region

*As Dataset has very limited number of samples of 3 tumor types; Tumor augmented regions were extracted from the original image and fed to the model for training.*

**Exploratory Visualization:**

Three types of Tumors that the project focus's on are:(Figure 1)



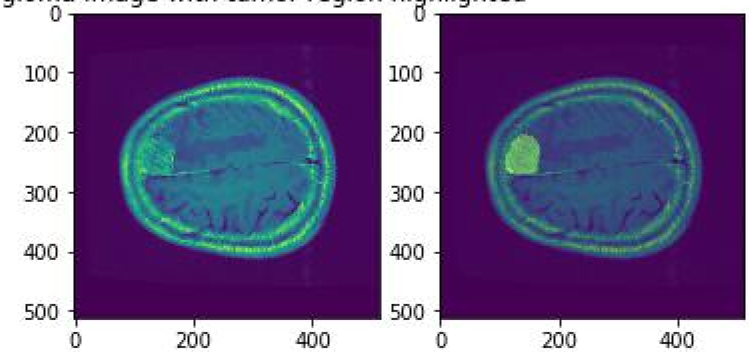
(a) (b) (c)

Illustrations of three typical brain tumors: (a) meningioma; (b) glioma; and (c) pituitary tumor.

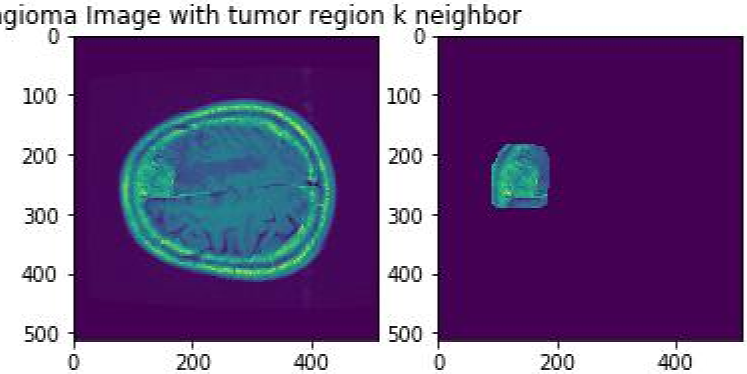
Below image gives you 3 tumor examples with tumor region highlighted (Figure 2)
Tumor region highlighted is through the information already availed in the dataset. “k” indicates the number pixels nearing to the tumor region and it is set to 20 in below images just for view ability. In execution, varied k neighbors were computed and fed to CNN (GoogleNet).

Please note that the augmentation of the tumor region is retained in 512x512 area. This enables varied size tumor to be availed to the model during training or testing.

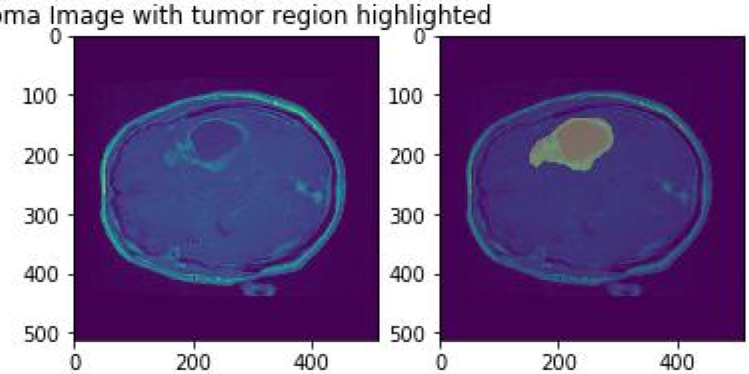
meningioma Image with tumor region highlighted



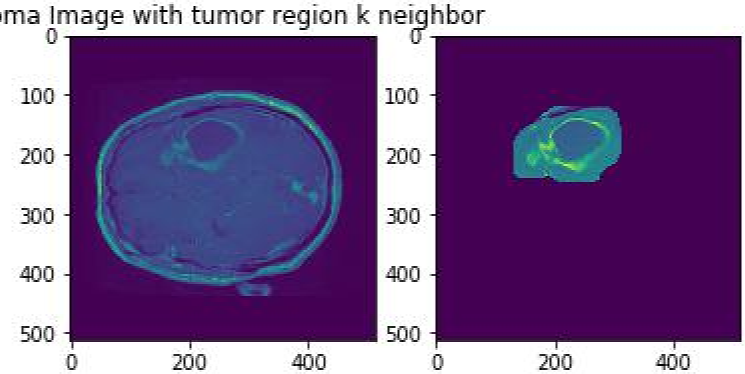

meningioma Image with tumor region k neighbor



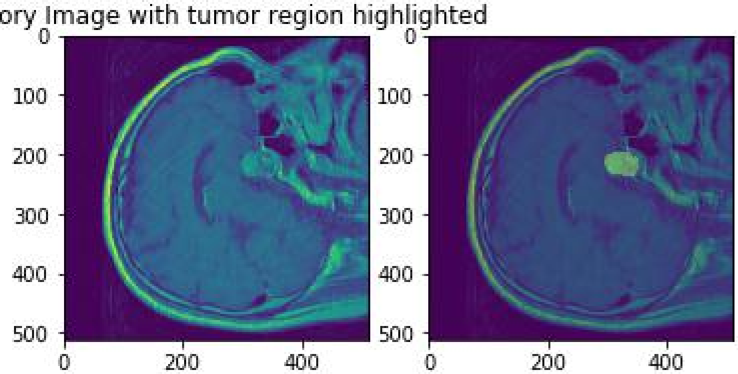

glioma Image with tumor region highlighted



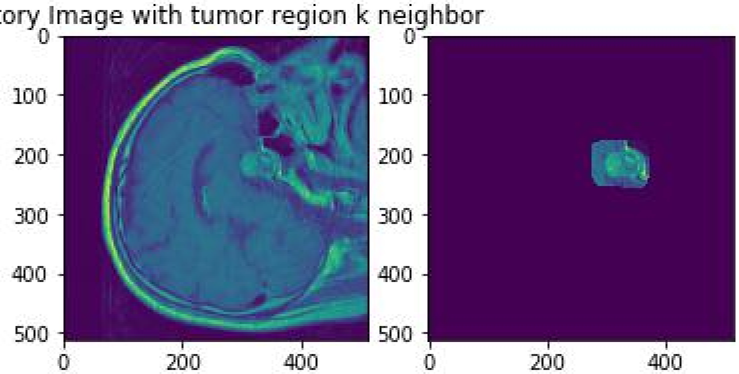

glioma Image with tumor region k neighbor



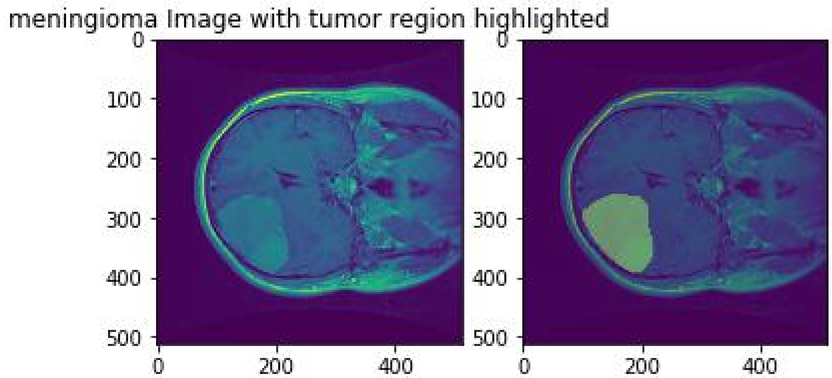
pitutory Image with tumor region highlighted



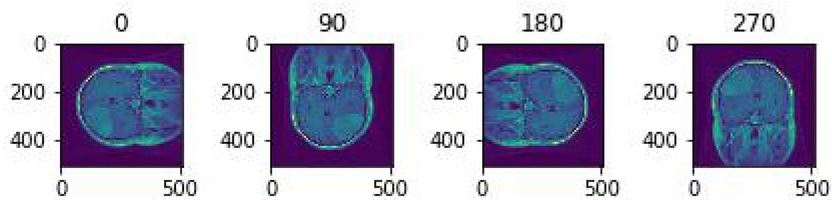

pitutory Image with tumor region k neighbor



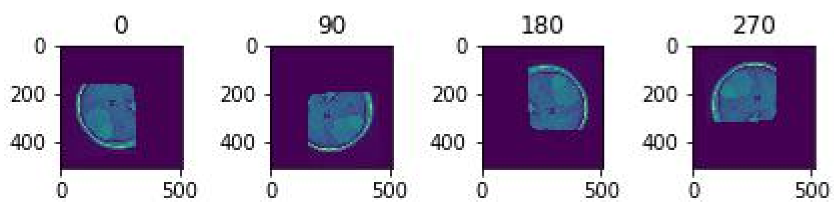
Please note that the images were cropped at 4 sides and rotated by 90 degrees to increase the input dataset size by 43 times as indicate below. (Figure 3)



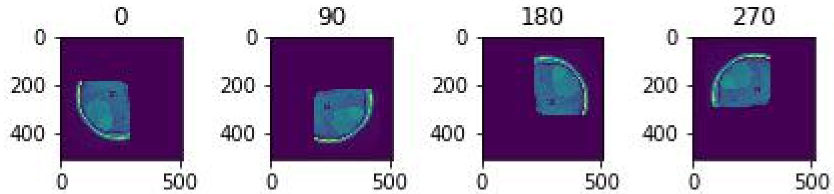

This is how the data is fed to CNN:
original image with the tumor and its 3 rotations



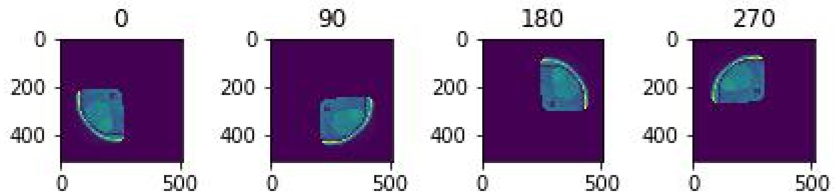

Tumor with 100 neighbors and its 3 rotations



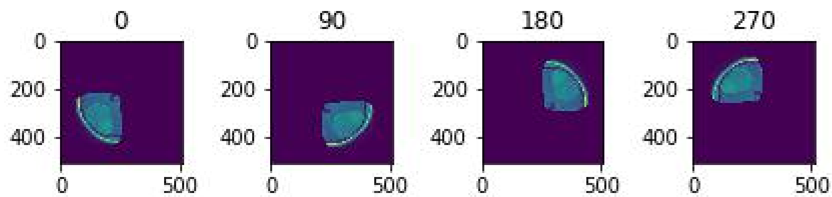
Tumor with 75 neighbors and its 3 rotations



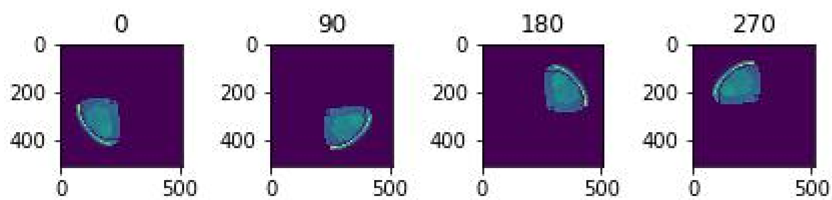

Tumor with 50 neighbors and its 3 rotations



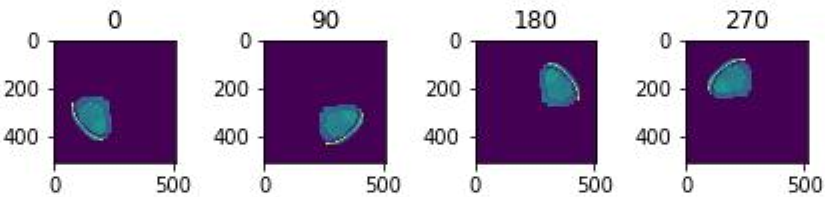

Tumor with 40 neighbors and its 3 rotations



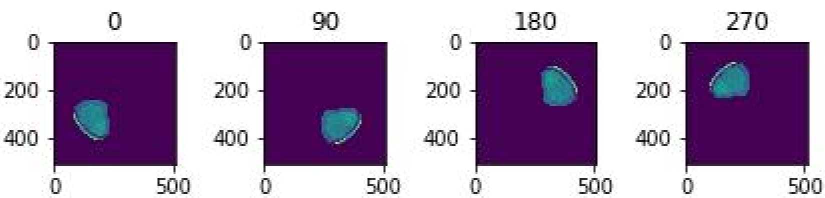

Tumor with 32 neighbors and its 3 rotations



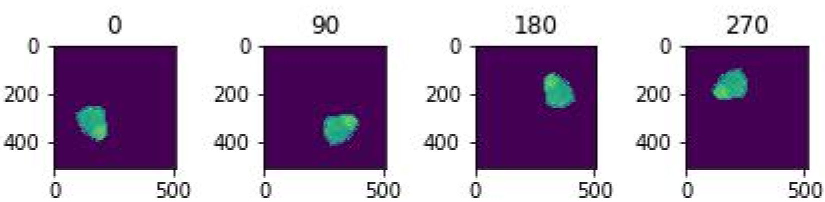
Tumor with 24 neighbors and its 3 rotations



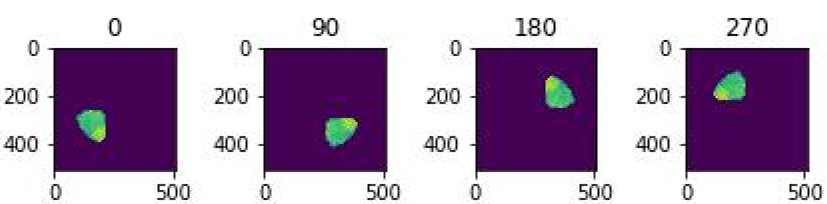

Tumor with 16 neighbors and its 3 rotations



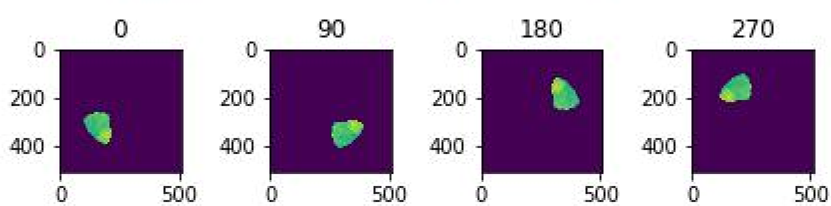

Tumor with 8 neighbors and its 3 rotations




Tumor with 4 neighbors and its 3 rotations




Tumor with 1 neighbor and its 3 rotations



**Algorithms and Techniques**

Idea behind the project is to classify MRI images (with Brain tumor) with the GoogleNet architecture. The paper below uses BOW (Bag Of Words) methodology to arrive at a accuracy of 88% with augmented tumor region.

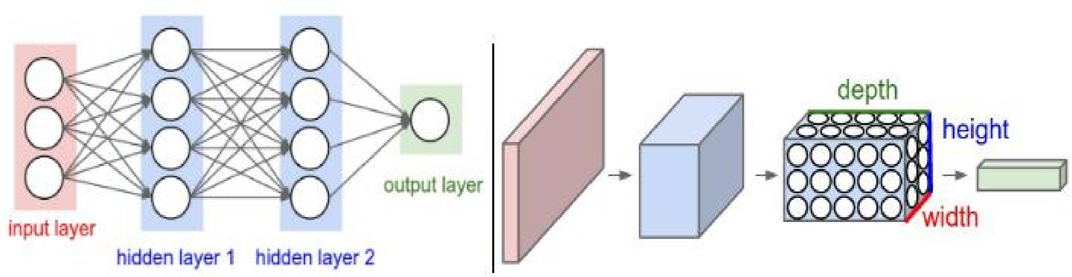
http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0140381

The Author indicates "The BoW method was originally used in the text retrieval domain, and it has been successfully adapted to the visual analysis domain".

However, in this capstone project, with the same dataset, it was attempted to reach the same accuracy through Convolution approach.

Inception modules with varied filter depth are stacked upon to form a the GoogleNet. Inception module computes and concatenates the 1x1, 3x3 and 5x5 convolutions with varied depth. Hence, convolutions are the basic fundamental block of the GoogleNet Architecture.

Convolution Networks or ConvNets is a neural network that share their parameter across space.




Left: A regular 3-layer Neural Network. Right: A ConvNet arranges its neurons in three dimensions (width, height, depth), as visualized in one of the layers. Every layer of a ConvNet transforms the 3D input volume to a 3D output volume of neuron activations. In this example, the red input layer holds the image, so its width and height would be the dimensions of the image, and the depth would be 3 (Red, Green, Blue channels).

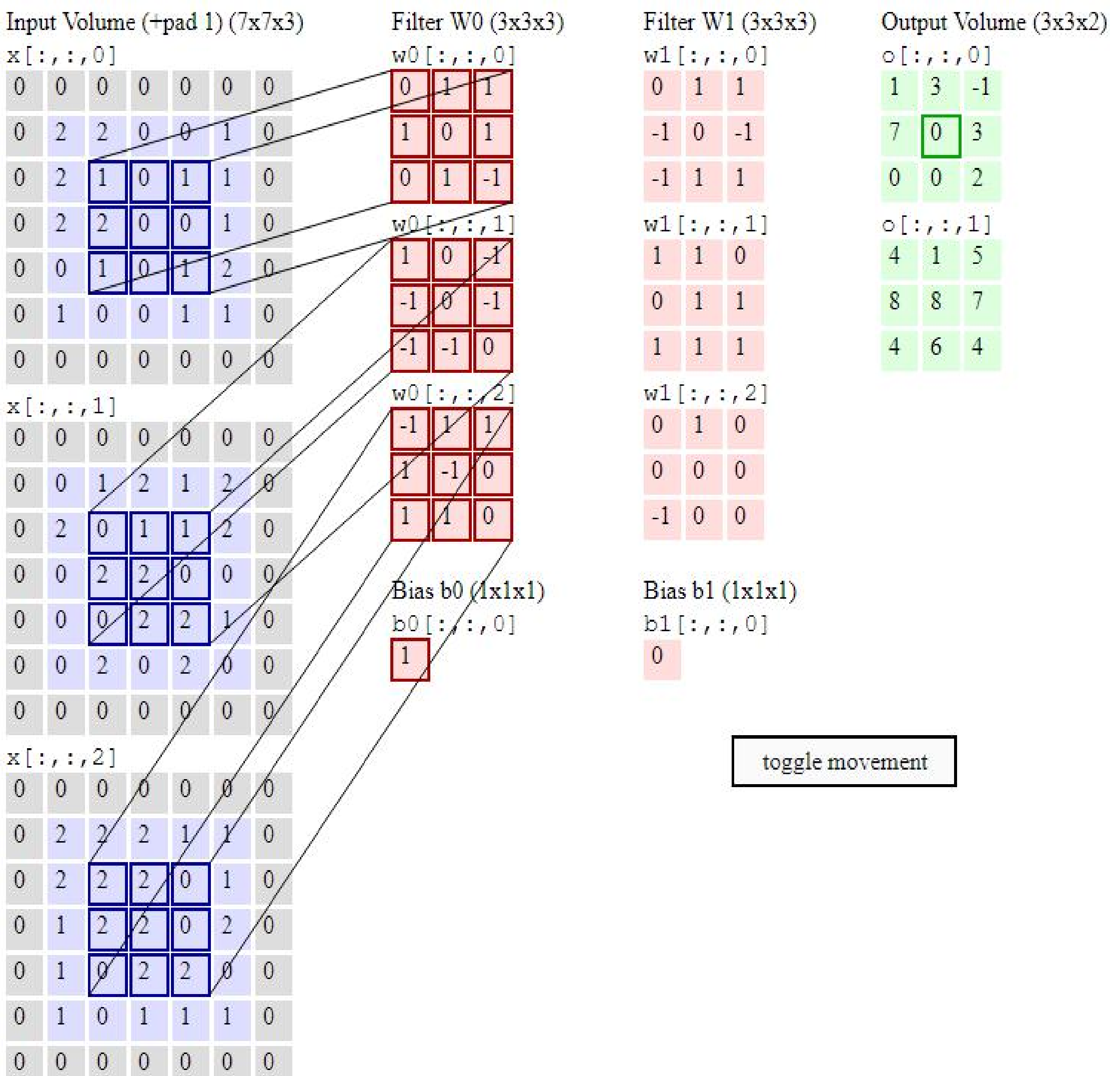
The above figure (Figure 4)clearly shows the difference between neural network and CNN. Unlike, a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions: width, height, depth. In case of CNN, small filter of increased depth (as compared to the input) is run over input data/image. Here, depth roughly balances the reduction of height and width dimension of the input image. During the run of filter over the input , the additional concept to be noted is 'stride'. It is the number of pixel that are shifted each time when patch is moved. Therefore, stride of 2 results in half of input's height and weight. Now, in order to reduce the loss of pixels at border of the image during the run of a filter; we may opt for 0 padding at each side. Output dimension at each layer of convolution can be calculated as per below formula:

Output width =(Input width – Filter width +2 * amount of 0 Padding )/(Stride) +1

Similarly, Output height = (In h - filter h + 2 * Padding)/ stride + 1

Depth = Number of filters

The "run" of the patch on the input image is best described in the below figure (Figure 5). Here, patch dimension of 3x3x3 is run on 5x5x3 image (with 1 layer of 0 padding at each side). Here, there are 2 filters used and stride = 2 and hence, the output is of the dimension 3x3x2.



**Benchmark**

The paper already compares accuracy for 3 methods ; intensity histogram, GLCM and BoW as indicated in below figure (Figure 6)

| R\methods | intensity histogram | GLCM-element | BoW |
|---|---|---|---|
| 0 | 71.39 | 78.18 | 83.54 |
| 8 | **82.31** | **84.75** | **88.19** |
| 16 | 80.39 | 82.79 | 86.86 |
| 24 | 79.67 | 82.60 | 87.58 |
| 32 | 78.68 | 82.41 | 87.26 |

doi:10.1371/journal.pone.0140381.t001

The author uses the patch size of 5x5 and average pooling; GoogleNet also has varied combination of patch size ,concatenation and pooling for the same.

## III. Methodology

**Data Preprocessing**

**Removal of Outliers:**
There were couple of 256x256 images in the Dataset which were removed as to keep the processing generalized.

**Image data Normalization:**
Each Brain MRI images has different intensity and was normalized between 0,1. Brain MRI Image is of the shape/dimension 512x512 ; however, the data was transformed to 512x512x1 to feed it to GoogleNet.

**Batch Creation:**

Batching algorithm takes input of batch size and carefully creates a batch with equi volume of 3 tumor images. Training and testing dataset ratio is 80:20. 20% of validation set is used to calculate training accuracy.

Note that a single Epoch will not cover all the MRI Images. However, they will be picked up in the subsequent run of Epochs. This is intended because we need to have uniform distribution of the 3 tumors in each batch and most importantly the Dataset do not have equal proportion of 3 types tumors. Random based selection is emphasized so as to avoid over fitting.

To Overcome memory issues, File Ids are stored against tumor lists and images fetched from the file during batch creation leaving the python enough space for GoogleNet layer computations. This has a side effect of 2 minutes delay per Epoch run but it is worth it. To minimize the frequent read delays Batch size is kept as high as 128*3=384 images/batch.

**Implementation**

To summarize the design in the simple words, the project focuses on using GoogleNet (winner of ILSVRC2014) for the tumor region detection and hence to make classification into 3 brain tumor categories based on the tumor region identified. It is like training a model with Apple and/or Orange images and expecting the model to identify the presence of apple or orange in a image of bowl of fruits.

What is ImageNet?

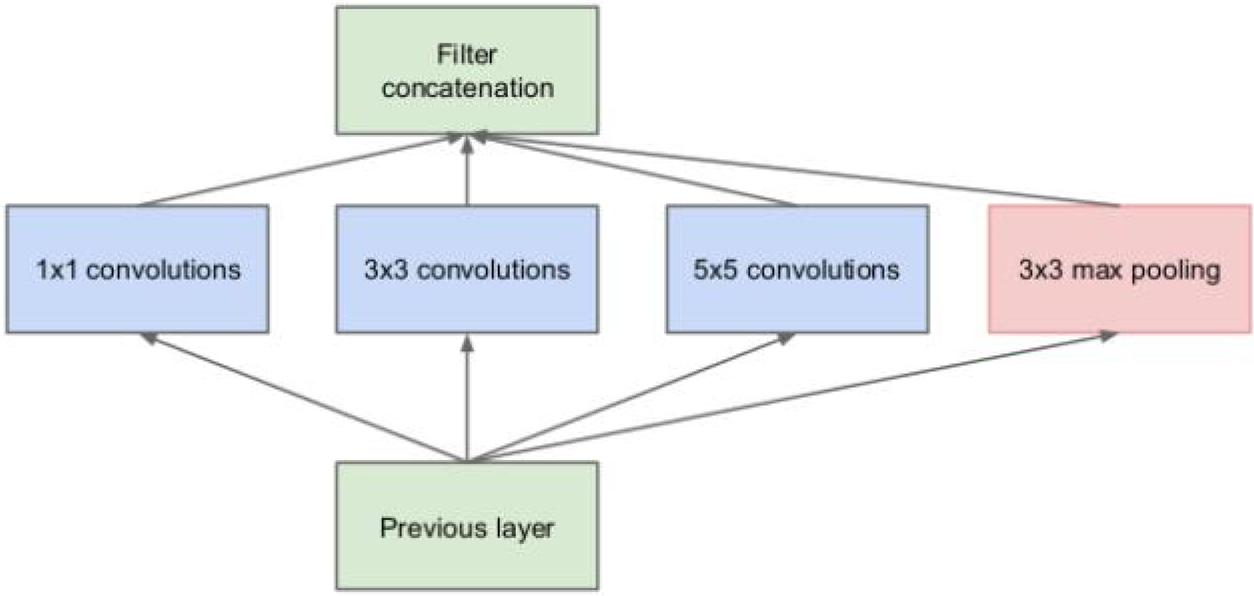
ImageNet is an image dataset organized according to the WordNet hierarchy. Each meaningful concept in WordNet, possibly described by multiple words or word phrases, is called a "synonym set" or "synset". There are more than 100,000 synsets in WordNet, majority of them are nouns (80,000+). In ImageNet, we aim to provide on average 1000 images to illustrate each synset. Images of each concept are quality-controlled and human-annotated. In its completion, we hope ImageNet will offer tens of millions of cleanly sorted images for most of the concepts in the WordNet hierarchy.

What is ILSVRC?

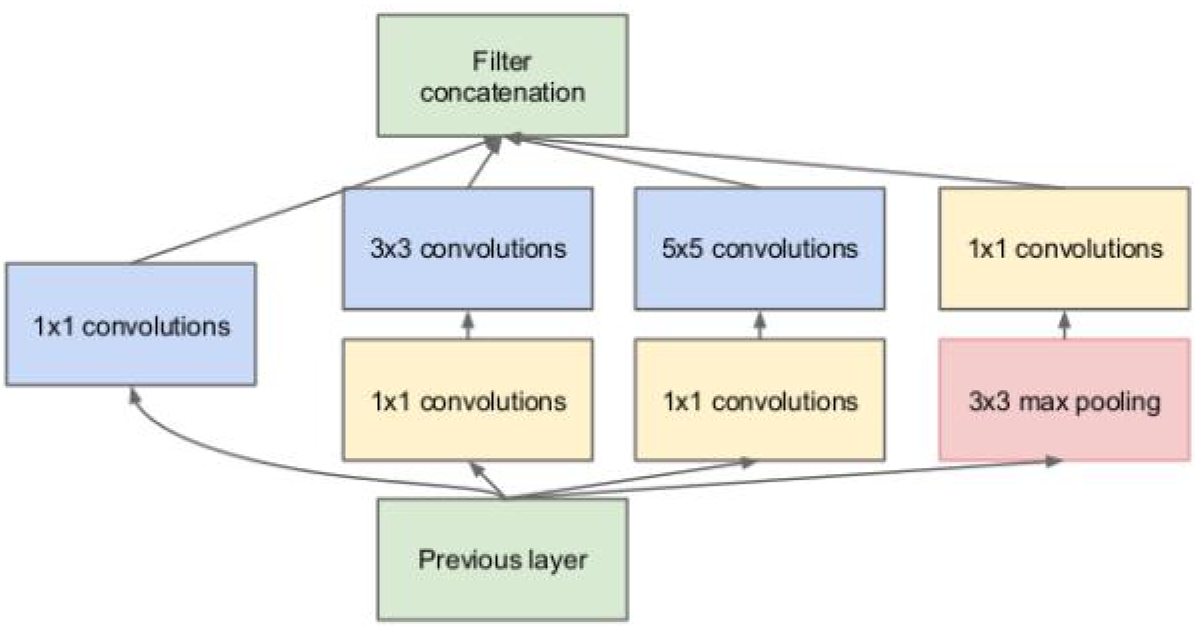
The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) evaluates algorithms for object detection and image classification at large scale. One high level motivation is to allow researchers to compare progress in detection across a wider variety of objects -- taking advantage of the quite expensive labeling effort. Another motivation is to measure the progress of computer vision for large scale image indexing for retrieval and annotation.

What is Inception Module?

As Explained in the below link, which actually restates the Udacity course material:
https://hacktilldawn.com/2016/09/25/inception-modules-explained-and-implemented/




The above picture (Figure 7)illustrates the naive inception module concept. However, in practice the 1x1 convolutions are used both computation efficiency and for filter depth increase as indicated below (Figure 8).




"
The inspiration comes from the idea that you need to make a decision as to what type of convolution you want to make at each layer: Do you want a 3×3? Or a 5×5? And this can go on for a while.

So why not use all of them and let the model decide? You do this by doing each convolution in parallel and concatenating the resulting feature maps before going to the next layer.

Now let’s say the next layer is also an Inception module. Then each of the convolution’s feature maps will be passes through the mixture of convolutions of the current layer. The idea is that you don’t need to know ahead of time if it was better to do, for example, a 3×3 then a 5×5. Instead, just do all the convolutions and let the model pick what’s best. Additionally, this architecture allows the model to recover both local feature via smaller convolutions and high abstracted features with larger convolutions.
"

Udacity course which talks about Inception:
https://www.youtube.com/watch?v=VxhSouuSZDY&list=PLAwxTw4SYaPn_OWPFT9ulXLuQrImzHfOV&index=41

What is GoogleNet?
GoogleNet has 22 Deep Network and relies on Inception Module for Network in Network approach. Please refer to https://www.youtube.com/watch?v=_XF7N6rp9Jw for more information of intro and advantages of GoogleNet.

GoogleNet was able to successfully able to not only recognize objects in a image but also could classify image. Below figure (Figure 9)

| type | patch size/ stride | output size | depth | #1×1 | #3×3 reduce | #3×3 | #5×5 reduce | #5×5 | pool proj | params | ops |
|---|---|---|---|---|---|---|---|---|---|---|---|
| convolution | 7×7/2 | 112×112×64 | 1 | | | | | | | 2.7K | 34M |
| max pool | 3×3/2 | 56×56×64 | 0 | | | | | | | | |
| convolution | 3×3/1 | 56×56×192 | 2 | | 64 | 192 | | | | 112K | 360M |
| max pool | 3×3/2 | 28×28×192 | 0 | | | | | | | | |
| inception (3a) | | 28×28×256 | 2 | 64 | 96 | 128 | 16 | 32 | 32 | 159K | 128M |
| inception (3b) | | 28×28×480 | 2 | 128 | 128 | 192 | 32 | 96 | 64 | 380K | 304M |
| max pool | 3×3/2 | 14×14×480 | 0 | | | | | | | | |
| inception (4a) | | 14×14×512 | 2 | 192 | 96 | 208 | 16 | 48 | 64 | 364K | 73M |
| inception (4b) | | 14×14×512 | 2 | 160 | 112 | 224 | 24 | 64 | 64 | 437K | 88M |
| inception (4c) | | 14×14×512 | 2 | 128 | 128 | 256 | 24 | 64 | 64 | 463K | 100M |
| inception (4d) | | 14×14×528 | 2 | 112 | 144 | 288 | 32 | 64 | 64 | 580K | 119M |
| inception (4e) | | 14×14×832 | 2 | 256 | 160 | 320 | 32 | 128 | 128 | 840K | 170M |
| max pool | 3×3/2 | 7×7×832 | 0 | | | | | | | | |
| inception (5a) | | 7×7×832 | 2 | 256 | 160 | 320 | 32 | 128 | 128 | 1072K | 54M |
| inception (5b) | | 7×7×1024 | 2 | 384 | 192 | 384 | 48 | 128 | 128 | 1388K | 71M |
| avg pool | 7×7/1 | 1×1×1024 | 0 | | | | | | | | |
| dropout (40%) | | 1×1×1024 | 0 | | | | | | | | |
| linear | | 1×1×1000 | 1 | | | | | | | 1000K | 1M |
| softmax | | 1×1×1000 | 0 | | | | | | | | |

GoogleNet Model has been implemented as in below paper based on Inception

Architecture.
https://arxiv.org/pdf/1409.4842.pdf

Following articles were studied to implement the GoogleNet.
https://github.com/Marsan-Ma/imgrec
https://gist.github.com/joelouismarino/a2ede9ab3928f999575423b9887abd14
https://hacktilldawn.com/2016/09/25/inception-modules-explained-and-implemented/
https://github.com/tflearn/tflearn/blob/master/examples/images/googlenet.py

### Refinement

Batching techniques were fine tuned to store the cropped images in memory so as to have the continuous supply of images to GPU while training. Along with tensorflow GPU compilation resulted considerable reduction in the training time taken.

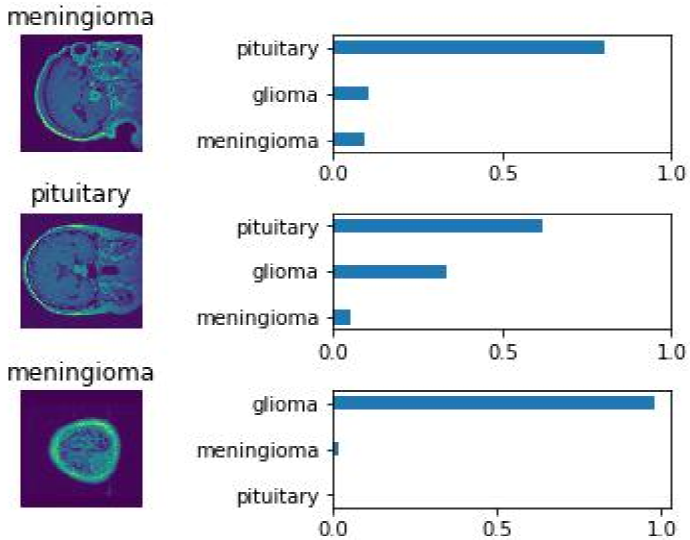
There were experiments with varied depth at each layer of inception module. As it could not give better result; its hyper parameters were unaltered in the final run.

## IV. Results

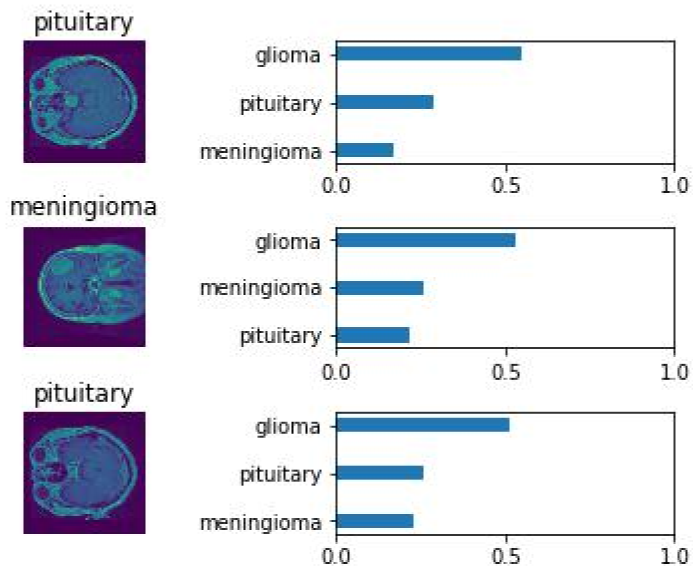
### Model Evaluation and Validation

Final model is in line to the solution expectancy. A comparative study of accuracy of the GoogleNet with conventional CNN architecture was done. GoogleNet not only showed better accuracy but also shown quicker stabilization of accurracy. Please note that the Conventional CNN architecture is derived from the model given in
http://cs231n.stanford.edu/

Few Sample predictions of GoogleNet Model with 56% accuracy is as below (Figure 10):




Sample predictions of Generic CNN Model with 47% accuracy (Figure 11):




**Justification**

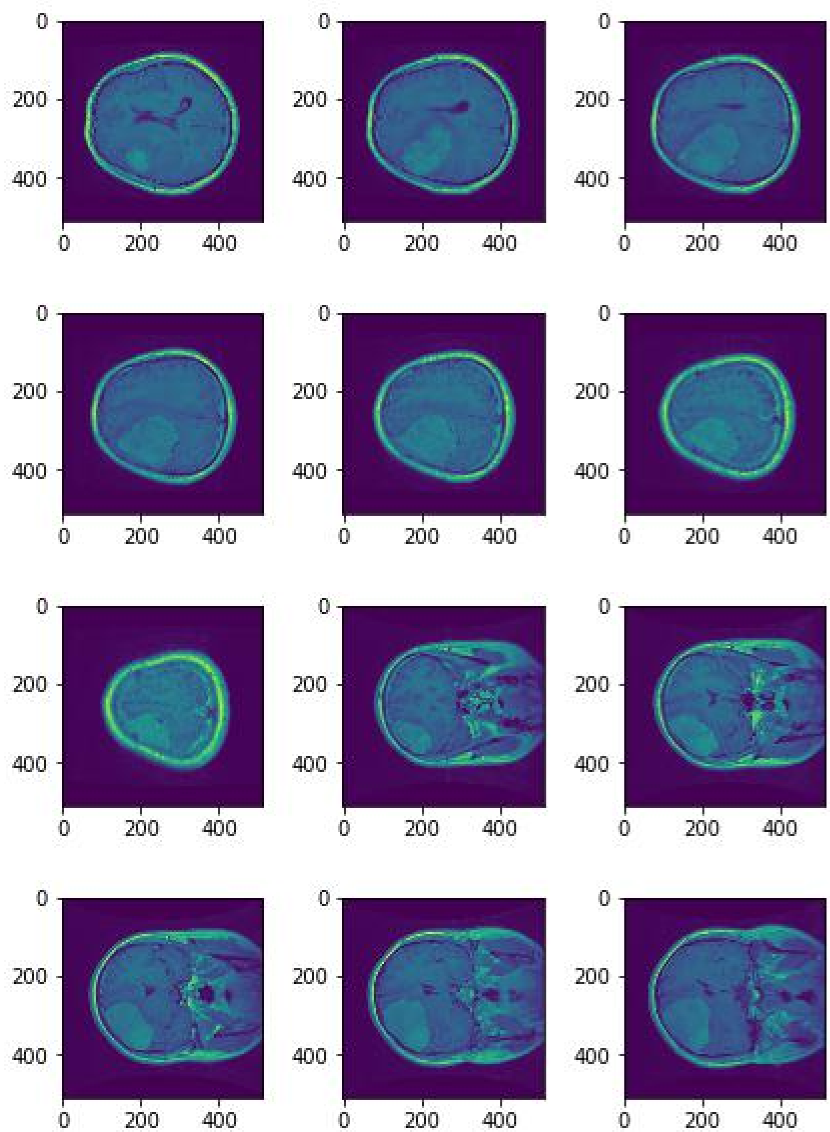
Clearly, multiple run of single batch indicates need of higher number of epoch runs as well. Benchmark results can be attainable if more GPU power is availed. The result proves that it is not random selection and also far better than the novice tumor analyst(like me).

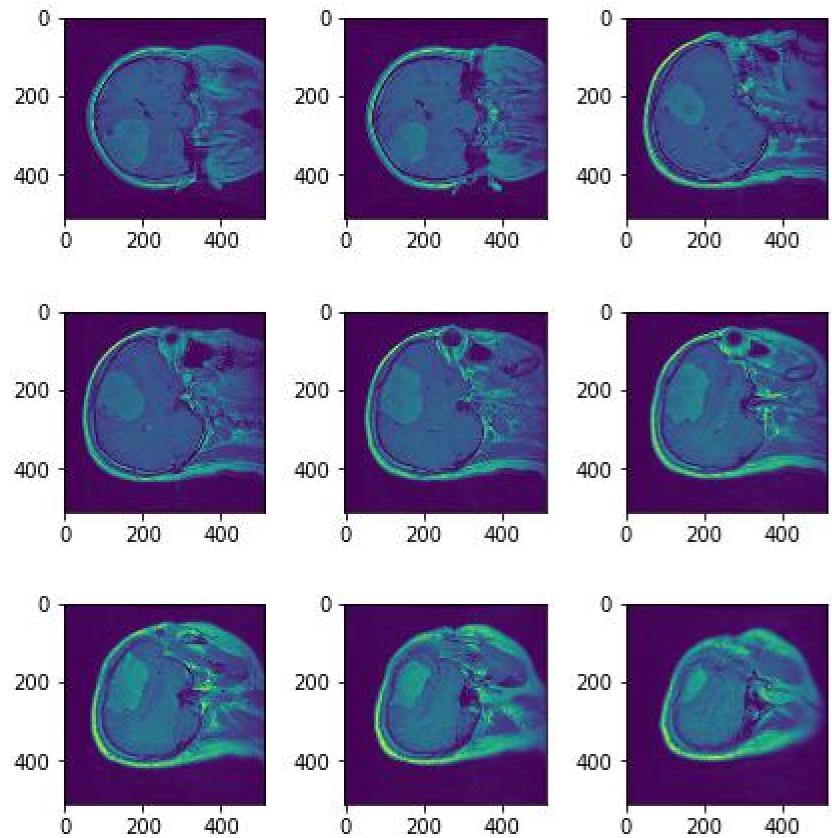
## V. Conclusion

### Free-Form Visualization

The MRI images per patient is not consistent. Count varies from 3 -20 images per patient(as indicated in the below figure for one patient of meningioma tumor). Number of images are at the discretion of the scanner who is collecting images. If there could have been automated scanning then the angles could be fixed and hence generic images can be given to CNN/GoogleNet architectures. May be latest scanning images like DIACOM might have the the angle/section information which can also be used clean the dataset. Please note that the images below are of the same patient as in figure 3 mentioned in “Exploratory Visualization” section.

For the Same Patient Diff MRI Images Available are as Below:





**Reflection**

End to End problem solution in machine learning involves following 4 stages;

**Dataset/Question:** we have gray scaled Brain mri images as our dataset and we try to address the problem of classifying them to 3 tumor types.

**Features (Extraction of information):** Definition of the 3 tumors are -
1. meningiomas are usually adjacent to skull, gray matter, and cerebrospinal fluid.
2. Gliomas typically involve white matter.
3. Pituitary tumors are adjacent to sphenoidal sinus, internal carotid arteries, and optic chiasma.

In simpler words the location of the tumor with respect brain is the criteria with which tumors are classified. Note that shape is not the criteria.

Tumor regions are second brightest elements in the T1-weighed CE-MRI dataset.

Argumentation of the tumor region does not only handles the above 2 features of the dataset which are fed to train the model but also helps in increasing the training samples.

**Algorithm:**For image classification various architectures have been developed over period of time. There are many architecture to start with LeNet to latest Squeeze-and-Excitation Networks. Generic CNN model was the base from which various of architectures started their research on. Hence, the same was used for the comparison with GoogleNet. GoogleNet was chosen in perticularly due to its popularity in image recognition field and its computational optimization.

**Evaluation:**accuracy metric was chosen for the multi class image classifier problem evaluation.

**Challenges Faced:**

1. Latest MAT libraries could not be used as the Dataset follows old mat structures.
2. Outliers came to light when batching logics were written. Hence, lot of time spent on revising Batching techniques.
3. Considerable time was spent on the compiling tensorflow on ubuntu 16.04 on GoogleCloud.This was required because current tensorflow GPU version is not compatible with cuda 9.1. Soon after the build ubuntu changed its version which actually impacted the NVIDIA drivers and hence could not work on the set up. Later had to move to ubuntu 17.04 and invested time on configuring the right combination for cuda and tensorflow.
4. Spent considerable time on various trails on different values for tumor knn
5. Experiments were also done GoogleNet Depth variation at various layers
   a) Doubled 4a layer series.
   b) Reduced initial strides and hand multiple convolution before 3a layer begins
   c) Played around at the end layers to match concluding layer output as that of GoogleNet imagenet output layer indicated in the previous section. Also tried with reducing FC layers.

## Improvement

1. Algorithm/Architecture is pretested and hence there need not have to be further investigation on the same. Accuracy can only be increased if more epochs are run over the input dataset.

2. Googlenet was implemented to identify the 1000 objects/classes in the given image . Hence convolutions depth in each inception modules (which are stacked) is gradually/carefully increased to 1024(please refer figure 9 in Implementation section). CNN Approach (as discussed in Result section with figure 11) indicates that the final depth required is much less than GoogleNet Archtecture depth of 1024. Many experiments were carried out with the depth at final layer as 40,60& 80 . Further study can be done with higher depth values.

3. Inception module architecture has been revised in the later paper (https://arxiv.org/pdf/1512.00567.pdf ). Even this may be studied on the given data set.