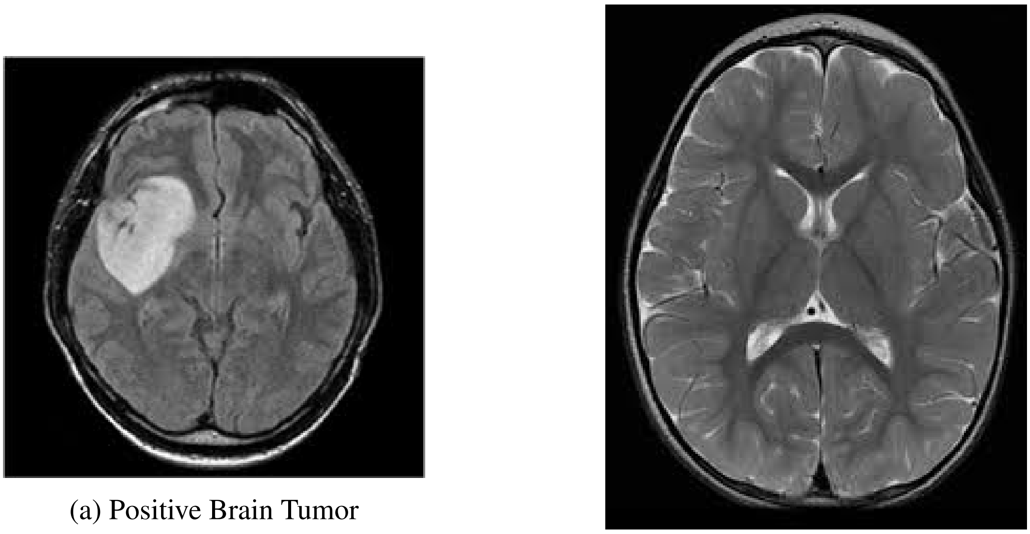
Detecting Brain Tumor with CNNs

John Kraft

May 2020

Brain tumors most commonly affect children and adolescents more than any other tumor [3] and approximately 700,000 Americans are currently living with a brain tumor today [1]. A brain tumor is a collection of abnormal cells. It can be classified as malignant (cancerous) or benign. Malignant tumors are the ones of utmost concern because if there is the primary tumor in the brain, it can spread throughout one's body and produce secondary tumors. It can cause a varies of symptoms including seizures, severe headaches, difficulty walking, and numerous concerning abnormalities in one's body inability one to live a normal life. It is important to understand the motivation behind researching and detecting brain tumors because it is something that can become life-threatening if it is not detected in the early stages of development. This analysis does not serve the purpose of detecting if a tumor is malignant or benign, but to extract and decided if there is a brain tumor present in the imagery. Once the carrier is notified of a brain tumor, other examinations are done to provide an accurate description if the tumor is malignant and has the potential to spread through their body. Other inspections may include analyzing the geometric shape of the tumor or surgically removing some of the tumor's tissue to analyze the tissue individually. Brain tumors are detected by a neurological exam with a computer tomography (CT scan) and/or a magnetic resonance imaging (MRI) test. It is important to use the technology at our disposal to get a highly accurate diagnosis. In this study, I used 253 MRI images to build a convolution neural network to detect if there is a tumor present from [2]. Figure 1.a displays an MRI image with a brain tumor, where figure 1.b exhibits an MRI image without a brain tumor.



(a) Positive Brain Tumor

(b) Negative Brain Tumor

Figure 1: MRI images with and without a brain tumor

When setting the two images side by side, there is a noticeable difference between the two and they can be easily identified as testing positive or negative for a brain tumor. According to the American Brain Tumor Association, nearly 80,000 people will be diagnosed with a brain tumor this year and around 16,000 people will die as a result of a brain tumor [1]. This is an important statistic to consider because the diagnosis process is extensive. With this being recognized, it can be valuable to acknowledge other technologies to ease and accelerate this process. The goal of researching into seeing if a convolution neural network can accurately diagnose a brain tumor is to assist the diagnosis process and not replace it.

One of the results I considered was the accuracy of my model which is defined as the total number of correctly predicted images divided by the total number of images. Also, I did examine other results such as precision and recall. The recall score is calculated by the total number of true positives divided by the addition of true positive and false negatives. It can also be described as the number of how many people are told they don't have a brain tumor when in reality they do. I put a higher importance on my recall compared to accuracy and precision because it can become deadly if a person is told they don't have a brain tumor and they move on without seeking necessary treatment.

The data set consisted of 253 MRI images, 155 images testing positive for a brain tumor, and the other 98 images testing negative. As shown in figure 2, the count difference between the number of positive and negative images is noticeable, but not enough to cause any concerns.

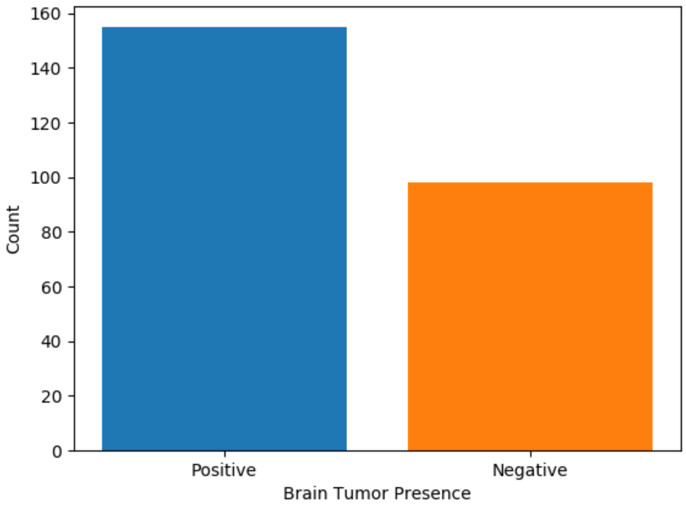


Figure 2: Size Difference

The first step was to set up my environment before proceeding and experimenting with the CNN model. I took all the images and converted them into a 32x32 pixel, red, green, and blue image. Next, I appended every image to a data array that could easily run through the CNN model. Then, I built up the labels array, consisted of ones and zeros where the ones represented a positive result for a brain tumor and the zeros representing a negative result. This is an essential part of setting up the environment because it is needed to assist the training process of the CNN model.

Next, I must set up the topology of my CNN that will consist of an input layer, a Conv2D layer, a max-pooling layer, a dropout layer, a flatten layer, a hidden layer, and finally, an output layer. Figure 3 represents my initial topology to attempt to get an understanding of the data I'm working with. Also, figure 3 represents the basis of what my topology will consist of throughout my entire analysis. This means I will only be adding new layers or changing parameters to this topology and not removing any existing layers.

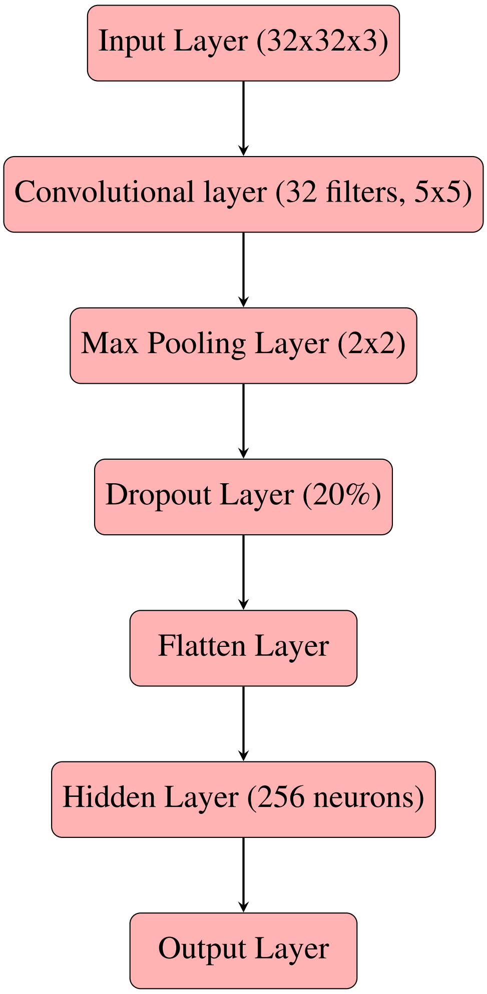
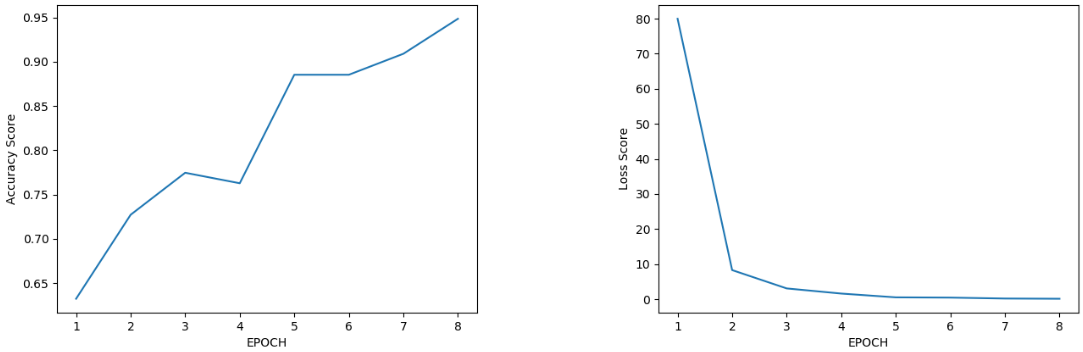


Figure 3: Initial CNN Topology

The input layer received the images in a 3-D space where the images are 32x32 pixels and the third dimension consists of 3 values representing red, green, and blue for each pixel. Then, it ran through the Conv2D layer with 32 feature maps and a 5x5 input patch using a relu activation. Next, the 5x5 input patch is converted to a 2x2 patch using max-pooling which downsamples the previous layer's feature map. A dropout layer is a form of ”regularization” which essentially randomly excludes 20 percent of the neurons. The goal of the dropout layer is to reduce over-fitting. Next, the flatten layer converts the 2-Dimensional matrix to a 1-Dimensional vector. It is important to flatten the matrix because it allows the output from the previous layers to run through fully connected layers and sets the data up nicely for the next layer which is the hidden layer. The hidden layer is where all the neurons are fully connected with a specific activation. For my first topology, I used a relu activation and 256 neurons. The final layer is the output layer which uses the number of classes, also known as the number of classes inside a one hot encoded vector, to minimize the loss value and to optimize the gradient descent. For my topology, it had 2 classes, true and false, trained with logarithmic loss, and used ADAM gradient decent, the same used in multilayered perceptrons.

Now, I will display the results of my CNN which used the topology in figure 3. Shown in figure 4, it displays the increase of the accuracy score as the number of epochs is increased, as well as the minimization of the loss score.



(a) Accuracy Score

(b) Loss Score

Figure 4: Accuracy and Loss Score on Initial CNN

The first feature I noticed was in figure 4b, the sudden drop in the loss score from the first epoch to the second epoch. This is an important result to consider because this shows the emphasis on running more than one epoch. Also, another feature I noticed was in figure 4a, the subtle drop inaccuracy on the fourth epoch, but on the fifth epoch, it jumped drastically, close to 13%. I gained some valuable insight into this data from this initial CNN. For instance, it displayed the positive effect of running numerous epochs. Also, I noticed on both plots, from epoch six to eight there wasn't much of an increase in accuracy or a decrease in loss. Even though I was to get up to an accuracy score of 95%, I decided it was important to research into how the model would react to unknown data and how well it would perform. Therefore, I decided to split my data into a training and testing data set. Figure 5 exhibits the difference in counts between the positive and negative brain tumor MRIs between the training and testing data.

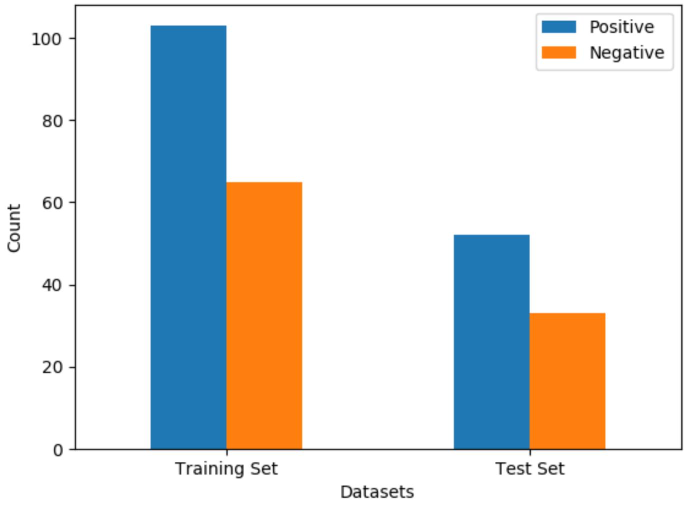


Figure 5: Plotting the Training and Test Datasets

I built the new CNN with the same topology as in figure 3, but this time with only the training data set and then tested the accuracy by running it on the test data. I ran this process eight times and recorded the accuracy and the recall percentage after each trial. Figure 6 displays the results of the eight trials in a table.

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6	Trial 7	Trial 8	AVG
Accuracy	0.75	0.774	.666	.702	.714	.75	.762	.726	.731
Recall	0.784	0.843	.529	.765	.647	.686	.745	.706	.713

Figure 6: Table Displaying Accuracy and Recall Results

The accuracy was not as promising as it seemed when we ran it with the entire data set. Also, I noticed the recall value is very volatile meaning it fluctuates frequently without a noticeable pattern. Another issue with these results is the recall averaged out to be roughly 71.3%. This is a considerable concern because that means roughly 3/10 will be diagnosed as negative for a brain tumor when in reality they have one. One solution I thought of was to convert the photos to 64x64 pixel images rather than 32x32 because the model doesn't take longer than 10 seconds to build and it would allow for fewer data to be lost in the conversion. Another solution I wanted to investigate ways to increase the epochs because my data set in general doesn't have a lot of observations, but after splitting the data set into training and testing data sets, the model is being built from a very limited amount of images. Therefore, increase the number of epochs will help increase the number of images the model is being built on. One concern to consider with this solution is being careful to not over-fit the data.

The first solution I tried was to convert the images to 64x64 pixel images rather than 32x32 pixel images. Then, I did the same analysis as before and stuck with a similar topology except the input layer will be 64x64x3. Shown below, figure 7 is the results of the new CNN.

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6	Trial 7	Trial 8	AVG
Accuracy	0.845	0.726	.798	.774	.762	.774	.810	.821	.789
Recall	0.922	0.725	.863	.745	.725	.824	.824	.902	.816

Figure 7: Table Displaying Accuracy and Recall Results

Converting the images to 64x64 pixel images rather than 32x32 pixel images shows great promise because the average accuracy jumped from 73.1% to 78.9% and the average recall score increased by over ten percent bring the score to an 81.6%. This is great news, however this model still misdiagnosis 1/5 people. The model still doesn't take very long to build, so I decided to adjust the number of epochs the model uses. I still want to consider the possibility of over-fitting. The way I plan to look for over-fitting is by keeping an eye for minuscule increases in accuracy after a few epochs. Also, it will be prominent when calculating the accuracy and recall score on the test data. Shown below in figure 8 displays the results of increases in the number of epoch to 15.

	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6	Trial 7	Trial 8	AVG
Accuracy	.810	.821	.786	.786	.821	.798	.798	.821	.805
Recall	.843	.784	.804	.784	.843	.824	.863	.863	.826

Figure 8: Table Displaying Accuracy and Recall Results

Increasing the number of epochs to 15 did not significantly increase the average of either the accuracy or recall score; however, there was something that did significantly improve the model. In previous models the accuracy and the recall score were inconsistent. The spread of the accuracy in figure 6 was 10.8% and the recall spread was 31.4%. With a fluctuation of 10.8% and 31.4%, it is impossible to rely on this model. The spread of the accuracy in figure 7 was 11.9% and the recall spread was 19.7%. The fluctuation in the accuracy was slightly worse and the recall improved by a fair amount, but still too significant for reliance. I bring those numbers up because the spread of both the accuracy and recall score dropped significantly when I increased the number of epochs to 15. The accuracy spread in figure 8 was 3.5% and the recall spread was 7.9%. The spread is important to consider because it shows the consistency of the model and how much we can rely on the model. It helps reassure the model being built wasn't just a fluke.

The final adjustments that can be made to the CNN is to improve the accuracy and recall score are manipulating the number of Conv2D layers, the number of feature maps within those Conv2D layers, the activation on both the Conv2D layers and the hidden layers, the number of hidden layers, and the total of neurons on each hidden layer. Therefore, there are numerous parameters I can adjust to improve my model, but one important aspect to consider is to only change a single adjustment at a time. This allows me to recognize which modifications improve the model and which ones to stay away.

The first adjustment I made to my CNN model was the activation's my Conv2D and hidden layers were using. I was initially using relu, but I decided to try a few others. The first one I tried was the activation tanh; however, the accuracy score stayed below 40% causing me to stay away from the tanh activation. Also, I tried the sigmoid activation to see if I could get better luck. Not to my surprise, the sigmoid activation results were better than the tanh activation but not as good as the relu activation. Therefore, I decided to stick with the relu activation and explore other options to improve my accuracy and recall score.

The second adjustment I made to my CNN model was the number of feature maps my Conv2D layer was using. Initially, I increased the number of maps from 32 to 64. After running multiple models with 64 feature maps, it is noticeable the recall score increases by a fair amount where the accuracy score still similar to what it was at with 32 feature maps. Therefore, I decided to increase the number of feature maps one more time to 128 because I put the highest value on the recall score. After all, it is significantly less dangerous to inform someone without a brain tumor that they have one compared to telling someone they don't have a brain tumor when they do. After changing the number of feature maps to 128, it did not increase either accuracy score or the recall score, therefore I will be sticking with 64 feature maps.

The third possible adjustment that can be made is the number of neurons in the hidden layer. Currently, I have 256 neurons but I plan to increase then to 512 to see if I can increase either the accuracy score or the recall score without sacrificing one for the other. After adjusting the neurons in the hidden layer to 512 it shows that the accuracy and recall score stays around 70% which is worse than my previous model's scores. Therefore, I decided to keep my hidden layer neuron count at 256.

The final adjustment I can make to my topology is the number of Conv2D layers and hidden layers. First, I choose to add a Conv2D layer similar to the first Conv2D layer. Sadly, I got no luck. Also, I tried to change the parameter on the second Conv2D, but again, I had no luck. Therefore, I went on to trying to add more hidden layers that were fully connected. I decided to add two more hidden layers with 64 neurons and both running on the relu activation. I did gain a little more luck with adding a hidden layer but still not significantly better. I was able to get about an average of 84% recall with an average accuracy staying around 80%.

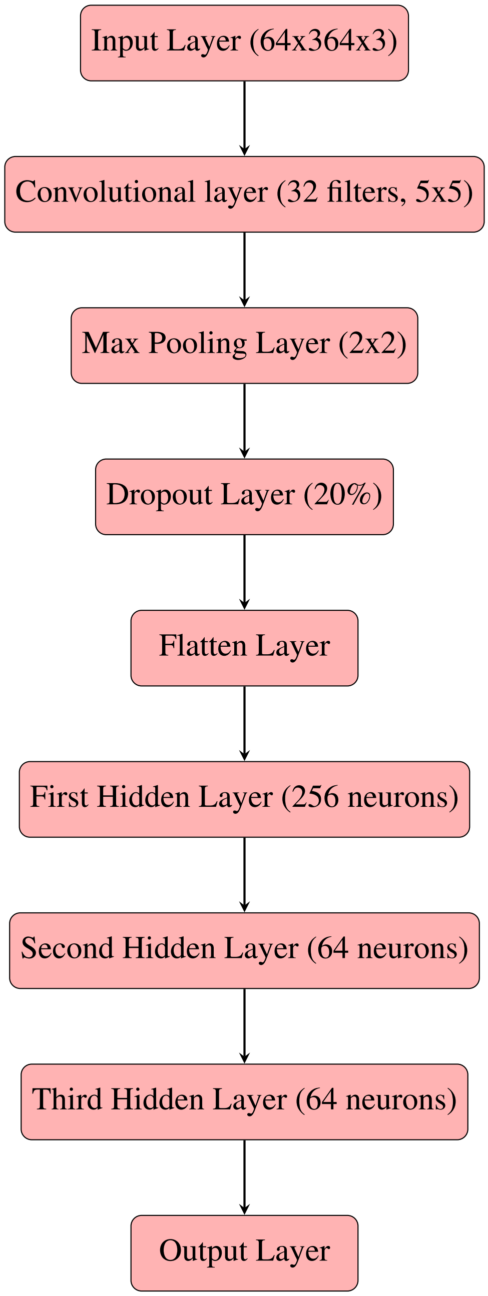

Input Layer (64x364x3)
Convolutional layer (32 filters, 5x5)
Max Pooling Layer (2x2)
Dropout Layer (20%)
Flatten Layer
First Hidden Layer (256 neurons)
Second Hidden Layer (64 neurons)
Third Hidden Layer (64 neurons)
Output Layer

Figure 9: Final CNN Topology

Figure 9 displays the final topology of my model. A few differences between the initial topology I built to my final topology are the input data size which was changed from 32x32 to 64x64 pixel images, along with the number of hidden layers. Another aspect I changed was the number of epochs I used to fit the data was changed to ten because after reviewing the score at 8 epochs and 15 epochs, it seemed to me eight epochs weren't enough, but 15 epochs were over-fitting the training data. Therefore, I landed at a median value between the two and decided to stick with 10 epochs which were the lowest epoch that avoided over-fitting the data. Figure 10 shows the confusion matrix of my final topology in figure 9.

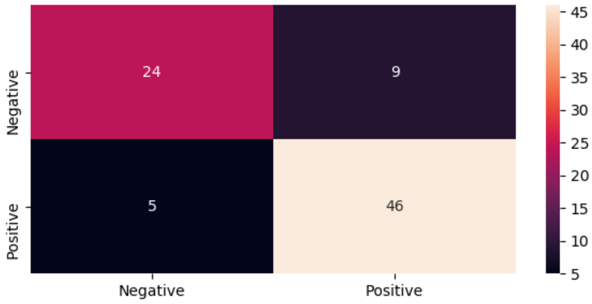


Figure 10: Confusion Matrix

In figure 10, the accuracy score was roughly 83.3% and the recall score was 90.2%. Even though these numbers seem promising to previous false-negative results, as well as a decent accuracy score, my topology is not very dependent because both my accuracy and recall score will still fluctuate a fair amount.

My goals were to achieve high accuracy and recall scores by building a CNN model and using existing MRI images containing a brain tumor and images that do not. Overall, I was happy with my results of developing a CNN that predicts brain tumors based on MRI images with an 83% accuracy score and a 90% recall score. It is still important to consider this model would never be allowed to be used in the medical field because 1/5 misdiagnosis is way too high, along with a 1/10 misdiagnosis that could end drastically poor. For now, this model should only be used to try to get a deeper understanding of how machine learning can have applications in the medical field especially diagnosis. If I had the opportunity to continue my research on MRI images with brain tumors, I would look into manipulating the images before running it through the CNN. One aspect I could change is cropping the images so it only contains the brain rather than the entire skull and the black background around a majority of the images. This would prevent the CNN to be wielded in the wrong direction. Therefore, it would eliminate outside noise and distractions thus only focusing on the brain. Another manipulation that can be done to the images is increasing the pixel size. My CNN showed promise when I increased the resize from 32x32 to 64x64. Therefore, it would be interesting to double the size again to see if would improve my CNN model by adding too much time complexity to the development of the CNN. There are numerous treatments to brain tumors, but someone has yet to discover a 100% survival rate treatment to all brain tumors and brain cancer. This is the sad reality 700,000 Americans and millions of people around the world live with everyday [1]. Even though there is no cure, there is an opportunity for machine learning to correctly diagnose patients with brain tumors in the early stages. Machine learning can create hope for the medical industry because if there is an algorithm that can correctly diagnosis patients at an extremely high rate, there may be an opportunity to save lives. This would allow doctors to focus on treatment and less on diagnosing because for some patients, time may be limited and every minute counts.

References

[1] American Brain Tumor Association. Brain tumor education, 2018.

[2] Navoneel Chakrabarty. Brain mri images for brain tumor detection, 2019.

[3] Stephanie Savelli and Pinki Prasad. Types of childhood and adolescent cancers, 2019.