In my project, I am looking to determine whether or not there is a tumor present in patient’s breast based on the data/image provided. I have decided to use softmax activation function with a categorical cross-entropy loss function. I chose this combination, rather than a sigmoid activation function for classification in conjunction with a binary cross-entropy loss function. Categorical cross-entropy is used when performing multi-class classification and binary cross-entropy for multilabel cross-entropy. Given that the output can only be “tumor” or “not tumor”, only one option for output is desired and with multi-label classification multiple items could be classified in a single image. In addition, I used two 2D convolution layers to reduce the 6x6 matrix of each dataset to 2x2 dataset. Then using a flatten layer I was able to create a 1D vector that could then be fed into multiple dense “fully connected” (FC) layers. The FC layers are connected to every neuron of its preceding layer to improve accuracy during training and validation. Also, as a note, the code is positioned to create a testing dataset that will be used in the last part of the assignment.

Below are the results of the neural network after 10 epochs:

**Epoch 1/10**

4476/4482 [============================>.] - ETA: 0s - loss: 0.0104 - accuracy: 0.9989

Epoch 1: val\_accuracy improved from -inf to 0.99862, saving model to brstTumor.h5

4482/4482 [==============================] - 16s 3ms/step - loss: 0.0104 - accuracy: 0.9989 - val\_loss: 0.0082 - val\_accuracy: 0.9986

**Epoch 2/10**

4468/4482 [============================>.] - ETA: 0s - loss: 0.0073 - accuracy: 0.9989

Epoch 2: val\_accuracy did not improve from 0.99862

4482/4482 [==============================] - 14s 3ms/step - loss: 0.0073 - accuracy: 0.9989 - val\_loss: 0.0082 - val\_accuracy: 0.9986

**Epoch 3/10**

4478/4482 [============================>.] - ETA: 0s - loss: 0.0071 - accuracy: 0.9989

Epoch 3: val\_accuracy did not improve from 0.99862

4482/4482 [==============================] - 17s 4ms/step - loss: 0.0071 - accuracy: 0.9989 - val\_loss: 0.0083 - val\_accuracy: 0.9986

**Epoch 4/10**

4480/4482 [============================>.] - ETA: 0s - loss: 0.0069 - accuracy: 0.9989

Epoch 4: val\_accuracy did not improve from 0.99862

4482/4482 [==============================] - 18s 4ms/step - loss: 0.0069 - accuracy: 0.9989 - val\_loss: 0.0079 - val\_accuracy: 0.9986

**Epoch 5/10**

4475/4482 [============================>.] - ETA: 0s - loss: 0.0069 - accuracy: 0.9989

Epoch 5: val\_accuracy did not improve from 0.99862

4482/4482 [==============================] - 23s 5ms/step - loss: 0.0069 - accuracy: 0.9989 - val\_loss: 0.0081 - val\_accuracy: 0.9986

**Epoch 6/10**

4482/4482 [==============================] - ETA: 0s - loss: 0.0068 - accuracy: 0.9989

Epoch 6: val\_accuracy did not improve from 0.99862

4482/4482 [==============================] - 31s 7ms/step - loss: 0.0068 - accuracy: 0.9989 - val\_loss: 0.0079 - val\_accuracy: 0.9986

**Epoch 7/10**

4481/4482 [============================>.] - ETA: 0s - loss: 0.0067 - accuracy: 0.9989

Epoch 7: val\_accuracy did not improve from 0.99862

4482/4482 [==============================] - 26s 6ms/step - loss: 0.0067 - accuracy: 0.9989 - val\_loss: 0.0073 - val\_accuracy: 0.9986

**Epoch 8/10**

4474/4482 [============================>.] - ETA: 0s - loss: 0.0067 - accuracy: 0.9989

Epoch 8: val\_accuracy did not improve from 0.99862

4482/4482 [==============================] - 30s 7ms/step - loss: 0.0067 - accuracy: 0.9989 - val\_loss: 0.0071 - val\_accuracy: 0.9986

**Epoch 9/10**

4472/4482 [============================>.] - ETA: 0s - loss: 0.0067 - accuracy: 0.9989

Epoch 9: val\_accuracy did not improve from 0.99862

4482/4482 [==============================] - 15s 3ms/step - loss: 0.0067 - accuracy: 0.9989 - val\_loss: 0.0078 - val\_accuracy: 0.9986

**Epoch 10/10**

4471/4482 [============================>.] - ETA: 0s - loss: 0.0065 - accuracy: 0.9989

Epoch 10: val\_accuracy did not improve from 0.99862

4482/4482 [==============================] - 15s 3ms/step - loss: 0.0065 - accuracy: 0.9989 - val\_loss: 0.0074 - val\_accuracy: 0.9986

I thought that a confusion matrix would be a great visualization to showcase the network’s performance. The true label shows what the expected value is (0 for no tumor and 1 for tumor) and the number of files for each class. The predicted label notes what the network predicted for each expected value. The goal is to achieve a high number of files in the top left and bottom right of the figure. See the figure below:

Chart

Description automatically generated

The high accuracy seen above is due to how the training and validation datasets were constructed prior to being fed into the NN. There are roughly 160,000 data files with about 350 of them being unmodified/not augmented. Only 175 of these images include a tumor. Of the 160,000 files, only 0.11% are real images with tumors. The remaining images are augmented with noise, brightness, contrast, saturation, sharpness, and gaussian blur. As a result, I have classified all but 175 images as “no tumor”. This has altered my neural network in a way that would result in a high accuracy for both training and validation because if it guesses “no tumor” then it’ll be correct 99.89% of the time. The best way to improve my work is to classify all tumor images that were augmented as “tumors” rather than “no tumors” to help the NN predict the classification for such images. After further discussions with Dr. Czajka, I realized that the data needs to be balanced, split 50/50 into tumor and no tumor datasets. This data will then need to be segmented into training, validation, and testing sets. This blunder was discovered too late and will be implemented in the next assignment.

Since my first attempt at using this network has a very high, and obviously skewed, accuracy I cannot say that adding additional convolutional (conv2D) layers to my model would improve accuracy. I do believe that when I make the aforementioned changes that implementing several more conv2D layers will increase the accuracy of the new dataset seen by the network. The reason for this is that introducing more conv2D layers will enhance the detail in the image /data that I am looking for.