Brain Tumour Classification

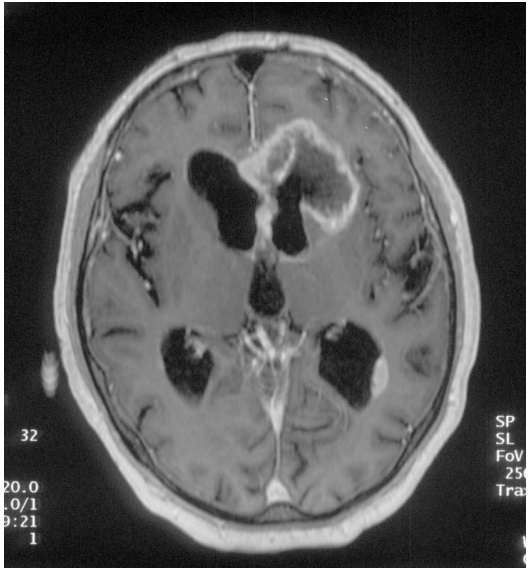
ECE 470

Andrew Rose - V00884894
Braidon Joe - V00822287
Ethan Janus- V00855202

Problem: Classifying Different Kinds of Brain Tumours

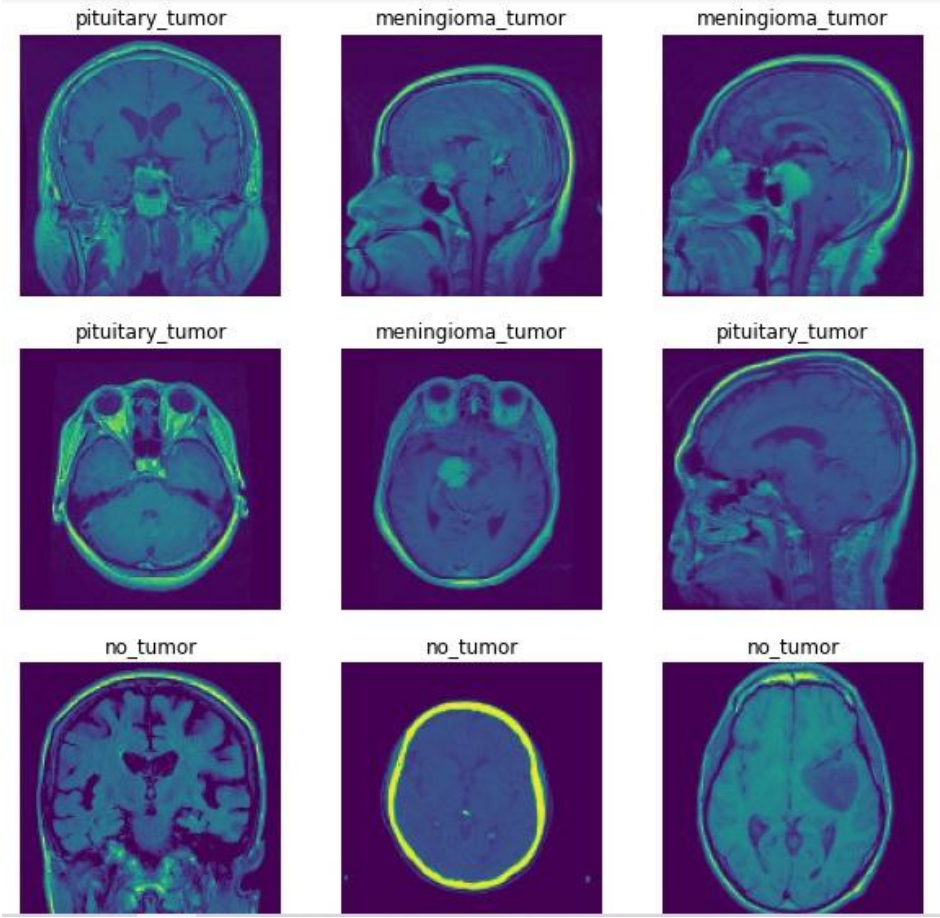
Why we selected this problem:

- Accurate identification and classification of brain tumours helps medical professionals treat patients
- To classify tumours into 3 different categories (plus “No Tumour”)
- To identify tumours in MRI scans that were taken at different angles of the brain
- To use a LeNet-5-style convolutional neural network (CNN) with optimized grid search to solve the classification problem using the TensorFlow in Google Colab



Data

- MRI scans of human brains
- 4 categories: Glioma Tumours, Meningioma Tumours, Pituitary Tumours, and No Tumour
- Using 2583 training images, 287 validation images, and 394 testing images
- Sagittal (side) views, coronal (front) views, and transverse (top-down) views of the brain are all present



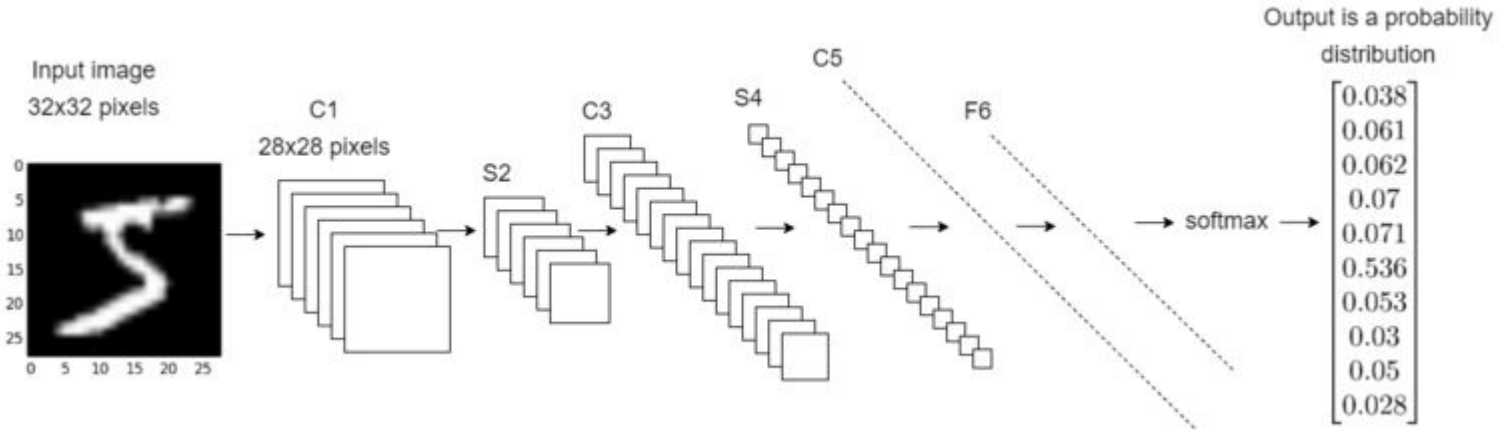
Approach: Preprocessing

Preparing the data for input into the CNN using TensorFlow's image preprocessing functionality.

1. Change all images from (usually 512x512) to 128x128 resolution for faster processing in the CNN
2. Crop and resize non-square images to size so that no stretching/squashing occurs
3. Change all images to grayscale from full colour RGB
4. Normalize all pixel values from an integer value between 0 and 255 to a floating point value between 0 and 1

Approach: Convolutional Neural Network

Inspired by the classic LeNet-5 image classification network layout, but adapted to larger images. We wanted to learn how to optimize the LeNet-5 concept for this new purpose.



Approach: Grid Search

- Trains multiple models based on different hyperparameter values
- Hyperparameters:
  - kernel size
  - activation function
- Fancy timesaving trick 1: After first pass, does an initial check of which activation functions have potential
- Trains remaining models for every possible combination of hyperparameters
- Chooses the best models based on lowest loss value
- Fancy timesaving trick 2: Stops training models whose loss values are much larger than the others

Approach: Optimizer and Loss Functions

- Optimizer used is called Adaptive Moment Estimator or Adam
- Adam is a first-order optimizer and works like an accelerated gradient descent

- Loss function used is Categorical Cross Entropy
- ML application is Multi-class classification
- Thus Categorical Cross Entropy is optimal for this task

Results

Activation Function: selu

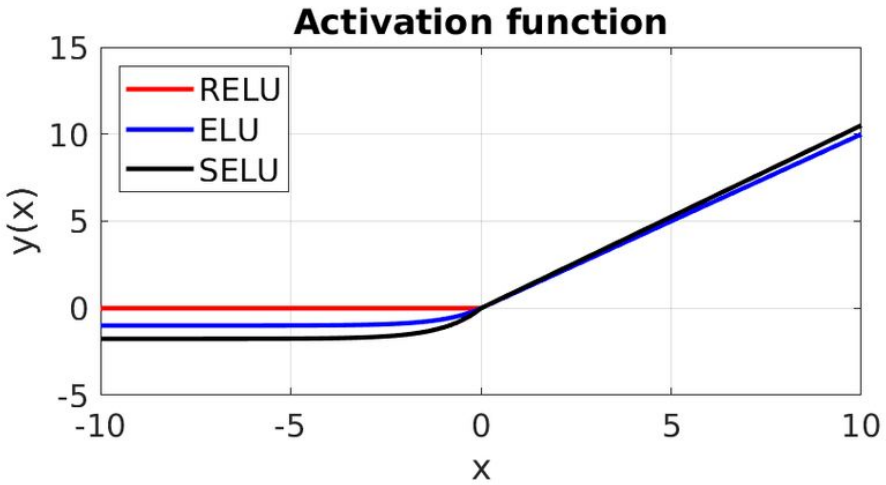
Kernel Size: 3

loss: 0.0353

accuracy: 0.9888

Test loss:
3.5515496730804443

Test accuracy:
0.7335025668144226



Limitations

- Some small chance that early drops dropped best hyperparameter values
- Varied angles, cropping, and scaling of photos

Future Work

- Consider splitting data up by viewing angle to improve model testing performance
- Fine tune number of filters and densely-connected nodes to see if performance improves
- Look into different loss functions that could improve performance
- Consider adding dropout layers to the network and data augmentation to help with overfitting
- Include an anti-overfitting loss improvement threshold for our grid search

Any questions?