Classification and Segmentation of Hemorrhages

Haoyuan Cheng
Department of Mathematics, Northeastern University

December 08, 2022

1 Abstract

In order to obtain a machine learning algorithm capable of classifying hemorrhages from different scanning windows, three different methods were developed and compared. These included logistic regression, transfer learning CNN, tree-based models. Additionally, two methods were used in order to segment the images; the first of which being an original method that allows for the use of linear regression in segmentation, and the second being a conventional method. These are all elaborated upon in the Models section of this paper below. Overall, the Neural Network produced the best results for classification.

2 Introduction

Brain hemorrhages, or cerebral hemorrhages, are an emergency condition in which a blood vessel within the brain ruptures causing internal bleeding [1]. They are most commonly the result of trauma and elevated blood pressure, and can lead to strokes, loss of brain function, and death [2]. For this reason, it is incredibly important to be able to classify and locate hemorrhages in the brain in order to provide proper treatment as fast as possible. One such avenue for this is through the use of image classification and segmentation modeling, with hopes of producing high-accuracy models that can assist a doctor in diagnosing a hemorrhage. As such, during this report we will go over different types of machine learning (ML) models that we developed and their accuracy scores, including linear and logistic regression, convolutional neural networks (CNNs), decision trees, and image segmentation.

3 Linear Logistic Regression

To begin, linear and logistic regression were used as baseline models to use in comparison with more advanced models such as CNNs and trees. Logistic regression was used in order to build a classification model for the hemorrhages, whereas linear regression was used to build a region of interest model based on the classification.

For the logistic regression, two different models were developed and compared. The first one was rather a collection of models as opposed to a single model; six different models were trained to identify solely whether or not the image inputted contained each specific type of hemorrhage, theoretically at a higher score. After this, an inputted image would be put through all six models, and would produce an output with a 1 for every model that predicted true and 0 for the others, recreating the format of the labels. This model was predicted to be especially effective at images with multiple hemorrhages, as it would scan for each type individually. Additionally, the model would reveal which specific image window worked best for identifying each type of hemorrhage. However, a theoretical limitation of this model was that even if the scores for each individual model was relatively high, when applied side by side, the probability of getting all correct will rapidly decrease.

When the algorithms were trained, the average score for each method across all windows was within the range of 0.59 to 0.68. Even if the highest possible score was taken for each model, when raised to the power of 6 for each type of hemorrhage being checked for, the results very rapidly approach a score close to 0. From these results, it is clear that the model would be highly ineffective at accurately predicting which type of hemorrhage was present in each image, as it contained a large margin for error in the individual scores, and would therefore likely predict multiple hemorrhages that do not actually exist. However, it did reveal that all windows performed similarly when undergoing regression.

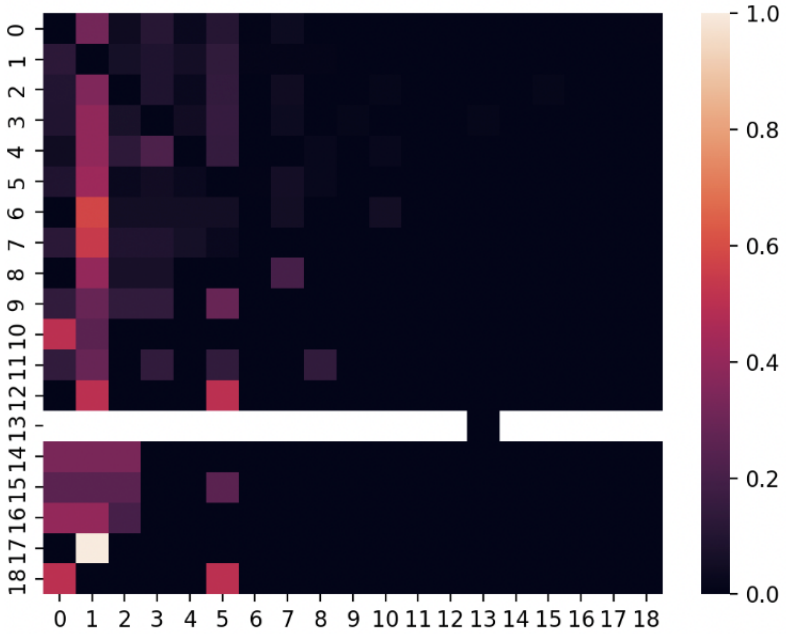
Because of this, the second model was created, this one being a multi-class regression, with 18 classes total representing different combinations of hemorrhages present. The label space for this model was set up such that it would be more easily

readable when the results were projected onto a confusion matrix. To begin labeling, the '1' or '0' value representing which hemorrhages were in the image were all appended into a single binary string. The 'any' column was removed, as it did not provide any extra information. This string was then converted into decimal in order to produce short, numerical labels ranging from 0-31 that would be used to distinguish between different types of combinations. As apparent in the confusion matrix, not all possible combinations of hemorrhages ended up being present in the image database. With the label space set, logistics regression was performed, obtaining the following score and confusion matrix:

Samples: 5000

Score: 0.3276

Confusion matrix:

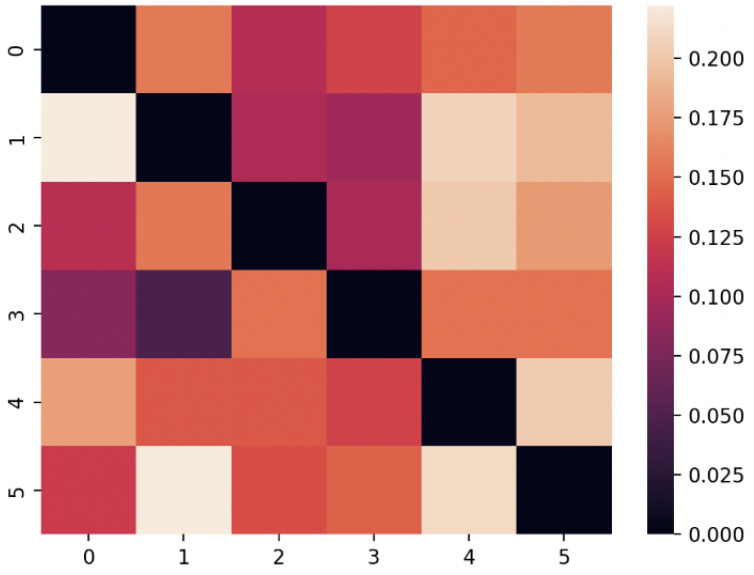


While this result was stronger than that of the earlier method, it is apparent that logistic regression performed with every different combination of hemorrhages would perhaps not be the most effective method in general. Therefore, a second multiclass regression model was created, this one using only six classes representing the six types of hemorrhages. The results for this model were as follows:

Samples: 3000

Score: 0.25

Confusion Matrix:



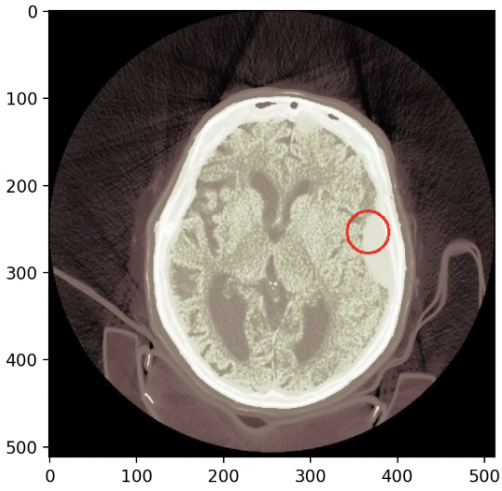
Once again, the score for the logistic regression model was quite low, solidifying the notion that a CNN or decision tree could potentially be a better choice.

For the linear regression model used in region of interest identification, the first step was creating a more useful label space from the given csv data. Because the labels were initially a large number of points outlining a polygon around the region of interest, and the number of points were inconsistent, measures were taken to simplify this. To begin, all of the coordinates were averaged in order to find an x and y in the middle of the region. Next, the size of the shape of the region was found and mapped to a circle, whose radius was then stored. Using this method, the region of interest could be identified with a center (x,y) coordinate, and a radius for the size of the circle. Next, a linear regression model with three columns was trained with the images; one to estimate x, one to estimate y, and one to estimate r.

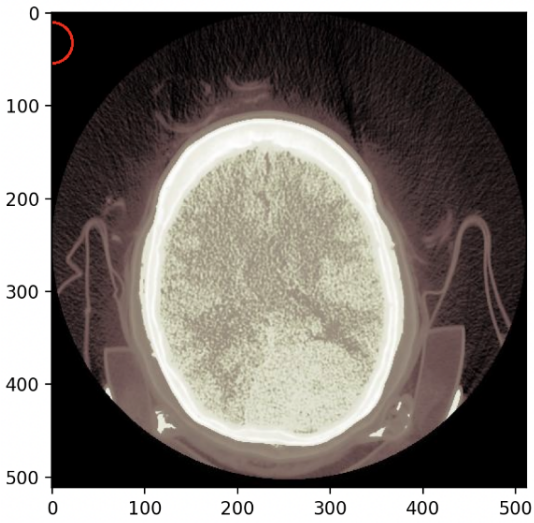
The model used is an unorthodox model, however it allows for image segmentation to be performed faster and with simpler code using basic linear regression. The purpose of this type of segmentation is not to precisely identify an entire region of interest, but rather to accurately pinpoint a relevant location on the image upon which a closer inspection could prove useful.

Because an exact point is estimated for the center but all that is needed for this method to cover a general area, low scores for the predicted point do not accurately reflect the usefulness of the model as long as the predicted point is within the distance of the radius of the circle from the actual center. As such, the conventional score for linear regression for this model was negligible. However, as apparent from the success case seen below, this model has positive potential if developed further.

Success:



Failure:



The first next step could be to redefine the scoring method such that it returns a 1 if the drawn circle contains the center of the tumor and 0 otherwise. Additionally, the model itself could be improved by transitioning from basic linear regression towards a new method that minimizes a cost function defined by the distance between the predicted center and the actual center as well as the size of the drawn and actual radius.

4 Neural Network for Classification

To apply CNNs to the classification problem, a transfer learning method was used by taking pytorch's resnet-18 model and retraining it on our dataset. This enabled the model to train on a relatively small dataset with relatively good performance. Both methods were trained on the max contrast image type.

The first method that was used involved the last layer of the resnet outputting a 7 dimensional tensor, with each value corresponding to a different hemorrhage type. An example output of the network would be:

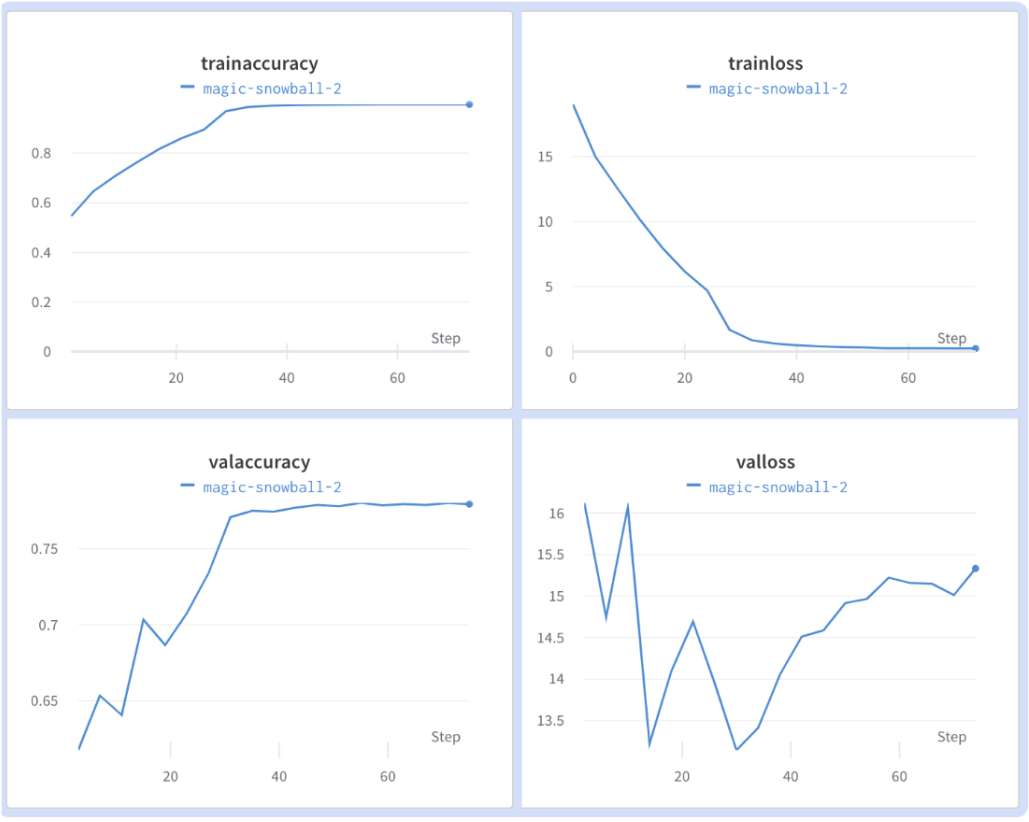
[0.154, 0.051, 0.313, 0.098, 0.113, 0.226, 0.045]

with the values corresponding to: ['normal', 'epidural', 'intraparenchymal', 'intraventricular', 'subarachnoid', 'subdural'. 'multi'] respectively.

The values in the tensor were then maxed over, providing the index of the highest value within the tensor that would correspond to the resulting hemorrhage type. With regards to the example above, the hemorrhage type would be classified as intraparenchymal.

The network was trained on the full dataset of around 100,000 images over 20 epochs. The hyperparameters have not been fully tested, but we used a batch size of 16, a decaying learning rate via a scheduler, and a cross entropy loss function. The results of the trained network can be seen below.

Output Network, Maxed over indices

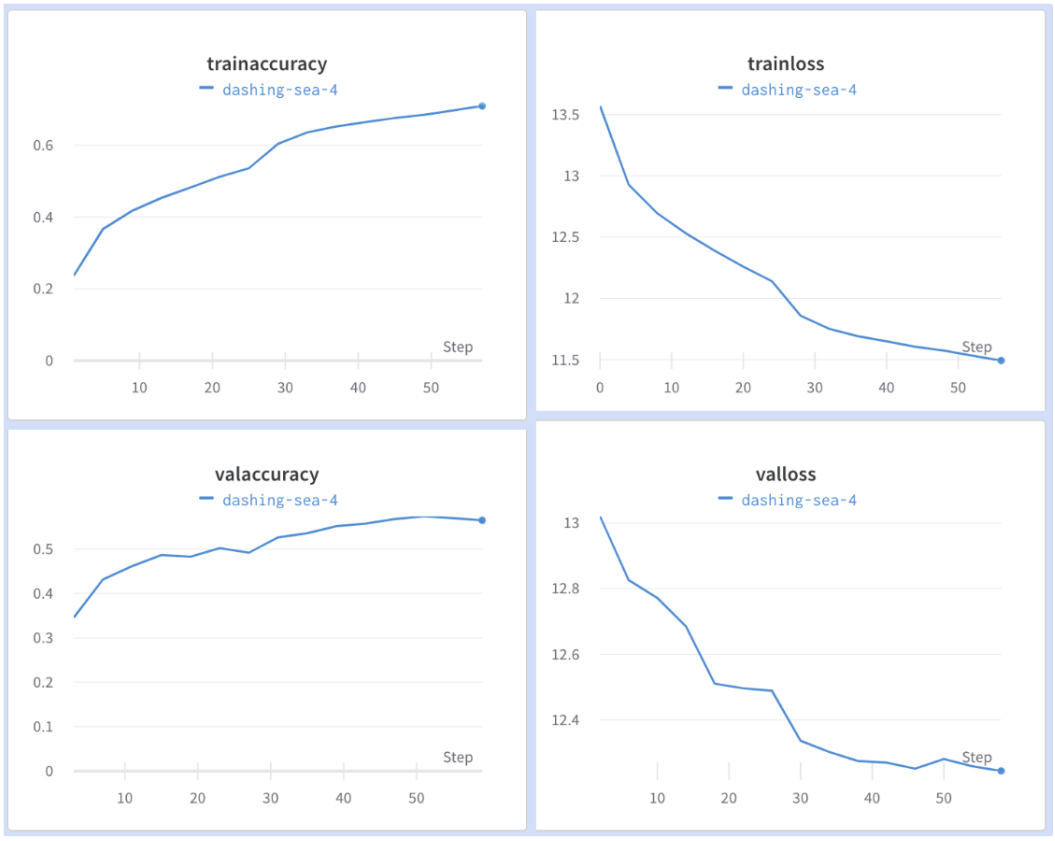


After 20 epochs, the model achieved a training accuracy of 0.997 and a validation accuracy of 0.78. The difference in performance is likely due to not enough data, an incorrect selection of hyperparameters, or an incorrect loss function. However, these results show potential for the use of CNNs in hemorrhage classification.

A drawback of our first method was that it failed to classify multi-type hemorrhages appropriately. Considering the fact that multi hemorrhages consist of almost a third of the overall dataset, more specific classification techniques for the multi hemorrhages are needed. Therefore, we developed a second method that involved changing the last layer of the resnet to output a 5 value tensor. This method used a binary classification technique for each different type of hemorrhage. The values were

then run through a sigmoid function in order to achieve binary values, then rounded to the nearest integer. An example output of this model would be: [0, 1, 0, 0, 1], with each value corresponding to: ['epidural', 'intraparenchymal', 'intraventricular', 'subarachnoid', 'subdural'.], respectively. The model was trained with a batch size of 8, using a BCE Loss function. The results of the second model can be seen below

Output Network with Binary Classification

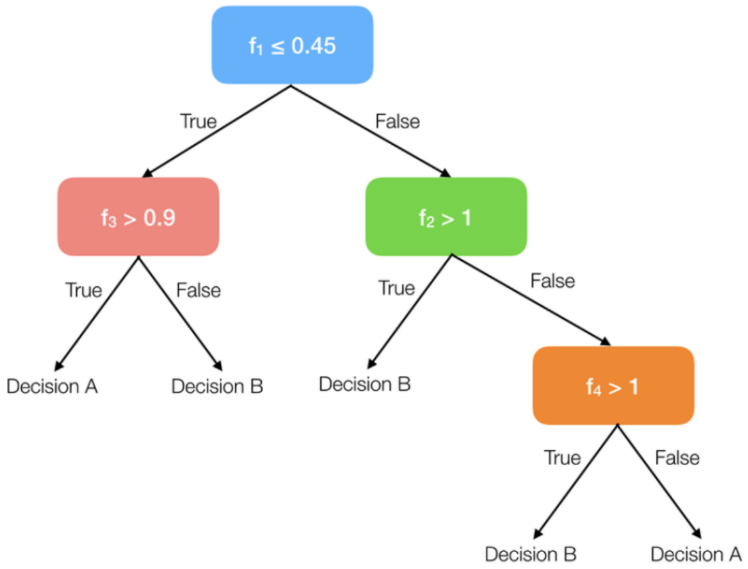


Our second method achieved a training accuracy of 0.70, and a validation accuracy of 0.57. The significant decrease in model accuracy is due to the fact that a single inaccurate value in the output tensor results in a failure of classification.However despite this, the results look promising. We were forced to end training early due to technical issues, but we can see that the model was not fully done training.

5 Tree-based Models for Classification

Tree-based models are extremely intuitive and mimic human-level thinking. Unlike neural networks, it is a white-box model. Tree-models are also different from logistic regression since the decision-making process does not depend upon probability distribution assumptions. Furthermore, since the tree structure can span infinitely, tree-based models are capable of handling high dimensional data with good accuracy.

To begin with, a baseline simple decision tree classifier is created. A decision tree is a flowchart-like tree structure where each internal node represents a feature or an attribute. The branches represent a rule to decide and each leaf node following each branch represents the outcome. The node at the very top is the root node. The algorithm starts from there and learns to partition into each branch on the basis of the attribute value. It then partitions throughout the rest of the tree following a recursive manner. A simple illustration of decision tree algorithm is shown below:



After pre-processed the data by reshaping each 512x512 image into a 262144 array, and aggregating with the size of the data, the resulting shape of the whole datasets becomes (282306, 262144). Then, the datasets is split into training and testing and input into the scikit-learn DecisionTreeClassifier() library. The resulting accuracy is 0.27.

To increase the accuracy, an ensemble method was implemented using a random forest classifier. A random forest is a meta estimator that fits several decision tree classifiers on different segments of the large datasets. It then uses an averaging method to increase the predictability and at the same time control overfitting.

Implementing such a method increased the overall accuracy to 0.46. Another ensemble method to increase the accuracy of a tree-based model is boosting. For this purpose, an Adaboost classifier was implemented. It started with a single decision tree classifier to capture the whole datasets. Then, for each part where the weight is incorrectly classified, an additional copy of the classifier is created to adjust the overall accuracy. Such a method arrived at an accuracy of 0.33.

To conclude, using tree-based models to classify brain hemorrhages did not produce satisfying results though the performance may enhance after hyperparameter tuning.

6 Image Segmentation

Image segmentation is a combination of both classification and localization, and can be classified into three main types: Semantic segmentation, Instance segmentation, and Panoptic segmentation.

For the interest of this paper, the segmentation performed on the brain hemorrhage scans is semantic segmentation, where the pixels of each image of the scan are classified as either belonging to the region of interest (ROI) which is the region of the hemorrhage, or belonging to any other region.

The goal of this segmentation is to help doctors identify areas of importance/ areas which need to be operated automatically and quickly, which saves time from manual human analysis. Both supervised and unsupervised learning can be employed for semantic segmentation, however, supervised learning has a higher chance of achieving accurate results with a smaller data set. Supervised learning was employed for this project since a sufficiently large labeled dataset was available.

Data cleaning and organization Before beginning segmentation, the provided labeled data first had to be cleaned, and converted to labels usable by the learning algorithm.

The data set provided for the hemorrhage scans consists of around 2000 usable labels, which comprises labels for epidural, intraparenchymal, subarachnoid, subdural images.

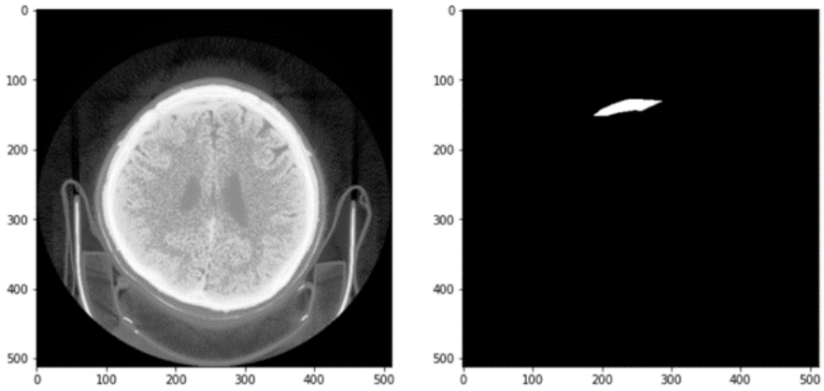
A set of 'flagged' labels was provided which were removed from the data set.

Labeling was done through classifying Regions of Interest using a technique called Polygon Annotation. Thus, every ROI label consists of a set of polygon vertices.

A large portion of the images had not yet been labeled, or had a low accuracy label, thus only the "Correct Label" of polygon vertices was used while the other data points were filtered(this is considered the gold standard of accuracy), however, in order to increase the size of the data set at the cost of accuracy, data from "Majority Label" can also be used.

Furthermore, due to the high number of polygon vertices per each label, several labels were removed as they contained inconsistent or empty polygon vertices.

The annotated polygons are composed of x and y coordinates between 0 and 1. Thus in order to create a mask using the coordinates, they were scaled by a factor of 512 and converted to a whole number, thus denoting the pixels enclosing the hemorrhage on the 512*512 image. All pixels within the enclosed region of interest were given the RGB label (255, 255, 255), and all pixels outside were given the RGB label (0,0,0). A new 512*512 pixel mask was created for each respective brain scan image.



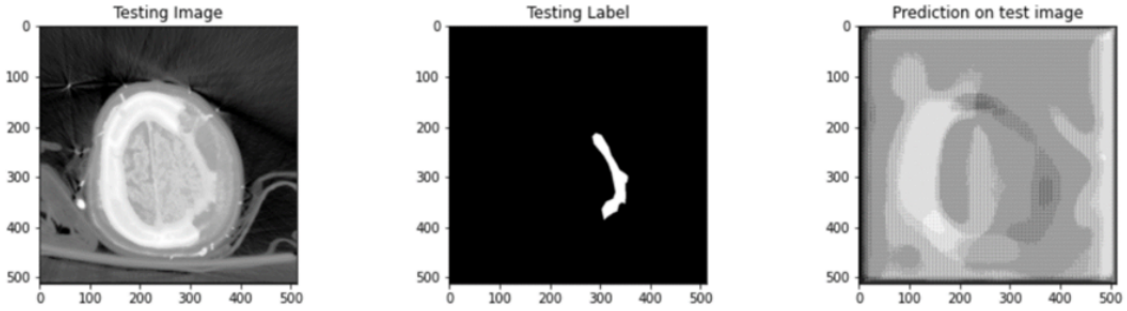
The pixel values were then normalized and the images of both the scans and masks were converted into a numpy array which is later to be used in the semantic segmentation training. One issue was that within the array the pixel values of the mask were not strictly 1 or 0, but rather some were approaching 0 or 1 (possibly due to an error in conversion to the mask). Thus the array values were iterated over and converted to 0 or 1, whichever was closest.

Learning algorithm The pixels within the polygon enclosed region are the training points. Thus, every enclosed pixel that belongs to the region of interest is classified as – hemorrhage, and every pixel that is not enclosed is classified as non-haemorrhage. The image that is produced from this classification is called the mask image. The training algorithm takes in the original image as training data and the semantic segmentation mask image as the label. The most commonly used types of

segmentation training algorithms include: · CNN – Convolutional Neural Network · FPN – Feature pyramid network · FastFCN – Fast fully connected network

For the purposes of this project, the model of choice is a type of convolutional neural network called U-Net[4], a model that was originally developed for biomedical image segmentation. See the following reference for the U-Net implementation used to help construct the model[5].

The training dataset was obtained from the one of the view windows of the brain scans, however, due to the image pixel clarity, "the brain bone window" was used to obtain the best results. The softmax function was used for activation. This function is used because there are two classes, one hemorrhage class and one non-haemorrhage class. The function outputs a probability indicating the likelihood that a pixel belongs to a particular class. The higher the probability, the darker the pixel is. An example is given below.



The U-Net uses convolution and up-convolution on the training and label images, producing new arrays called feature maps from the original pixel arrays of the images. The sizes of feature maps used for the interest of this project were, 1024*1024, 512*512, 256*256, 128*128.

The accuracy of the algorithm is evaluated by the IoU metric (intersection over union – the area enclosed by two overlapping regions, divided by the total area of the images). It indicates how well the prediction label overlaps with the testing label. The IoU values produced largely fell between 0.3 and 0.6.

```
n_classes = 2
IOU_keras = MeanIoU(num_classes=n_classes)
IOU_keras.update_state(y_pred_thresholded, y_test)
print("Mean IoU =", IOU_keras.result().numpy())
```

```
Mean IoU = 0.49425697
```

In conclusion, the prediction label is reasonably accurate, however, in order to increase the mean IoU (¿0.9 is regarded as an excellent value) and additionally output prediction labels which give clearer probability predictions, the training data needs to be increased. One way would be to use "majority labels" in addition to "correct labels" so as to increase training data size at the expense of accuracy of the polygon labels. Another way would be through image augmentation, such as flipping the image or rotating it. This can be performed on the "correct label" images so the accuracy of the annotations wouldn't be affected. Furthermore, augmenting the image would decrease the bias of the prediction labels. For example, if more hemorrhages were found on the right side, then the algorithm would be more inclined to label right side pixels, which resemble a hemorrhage. Then, flipping or mirroring the image, would minimize this bias.

7 Results

After comparing all four approaches, we found our that neural networks is the best approach in classifying brain hemorrhages. The result accuracy for training is 0.997 and the validation accuracy is 0.78.

7.1 Future Research

To enhance the classifying accuracy even further, some possible directions may include:
1) implementing a PCA methods to reduce the dimension of photos while keeping mamimum information
2) sort all images by contrast and filter out images that are unclear
3) use hyperparameter tuning with cross-validation to enhance the model performance

8 References

References

1. https://www.mayoclinic.org/diseases-conditions/subarachnoid-hemorrhage/symptoms-causes/syc-20361009
2. https://www.webmd.com/brain/brain-hemorrhage-bleeding-causes-symptoms-treatments
3.https://pytorch.org/docs/stable/generated/torch.nn.BCELoss.htmltorch.nn.BCELoss
4. https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/
5. https://github.com/bnsreenu/python$_{f}or_{m}icroscopists$