Identifying Hemorrhages with Machine Learning

Sumit De, James Duce, Maximilian Goott, Pranav Olety,
Raymond Valenzuela

December 12, 2021

Abstract

Image classification is a subset of machine learning and computer vision that aims to assign an image to a specific class based on pattern learned across a large set of data. There are several different models that can be used for image classification, each with varying levels of effectiveness for certain problems. In this project, we use logistic regression, support vector-machines, artificial neural networks, and convolutional neural networks to build models that identify if a CT scan of a brain has a hemorrhage, and identify the type of hemorrhage if so.

1 Introduction

Zeta Surgical is a company that aims to provide better insight during intensive brain surgeries with their bedside robotics platform. In order for the robotics platform to be able to perform surgeries, it needs to be able to classify a brain hemorrhage and create a 3-dimensional structure of that brain based on the segmentation of the hemorrhage. Based on this information, a robotic assistant like the Zeta Nexus can create a surgical plan and perform a procedure. We will be developing machine learning models to achieve the first part of this process which is accurately detecting the presence of a hemorrhage and what classifications the hemorrhage falls under. Machine learning is a suitable technique to solve this type of problem, as we are attempting to extract patterns about how different types of hemorrhages look from large amounts of image data.

2 Method Description

2.1 Problem Description

We were given labels for each image in the dataset in a csv file. The six columns were any, epidural, intraparenchymal, intraventricular, subarachnoid, and subdural. The last five columns are all different types of hemorrhages. The any column has a 1 if the image has any hemorrhage at all and 0 otherwise. If the image does have a hemorrhage, the next 5 columns either have a 1 if the hemorrhage is of given type and 0 otherwise. If the brain has a hemorrhage, then the hemorrhage can be of exactly 1 type or as many as 5 types. Hence, our approach for this problem will be to build a separate binary classification model for each column in the labels file, as the columns are not mutually exclusive and a hemorrhage can be of multiple. (There are 23 examples of hemorrhages that fall under every category of hemorrhage.) With this approach, we can identify multiple types of hemorrhages in a given image instead of grouping all of these images in a "multi" class.

2.2 Data Cleaning

The csv file of labeled images we were given had labels for 752,803 images. Of these labels, we had access to 116,532 images, as the initial sample set was dense with brains that had no hemorrhages and many of these examples were omitted. There were at least 3,145 examples for each type of hemorrhage, so we built a sample of the initial set of images that had 3,145 images of each type of hemorrhage. This sample had a total of 18,301 images (since an image can have multiple types of hemorrhage).

Every image we read was of size 512 x 512 pixels and in RGB format, so each image initially was associated with an array of 786,432 features. However, attempting to read all of these features was extremely slow, and also led to having far more parameters than training examples. So, we downscaled each image from size 512 x 512 to 128 x 128 and converted each image from RGB to grayscale. This process reduced the number of features for each image by a magnitude for 48, from 786,432 to 16,384 pixel values, and made it more feasible to run our models in a reasonable amount of time.

2.2.1 Model Testing

We create a customized subset of data to train and test every model based on what type of hemorrhage we are trying to classify. Each model acts as a binary classifier for a hemorrhage type that determines if the given image has that type of hemorrhage. If a simple random sample of our data is taken to train and test every model, then only about 17% of the data will belong to the positive class. Training a model with this sample will create an unbalanced model. So, we instead create a sample data set for each model where 50% of the data belongs to the positive class (images containing the type of hemorrhage we are trying to classify) and 50% of the data to negative class (any image that does not have the targeted hemorrhage). The negative class would contain both images of different types of hemorrhages and images without any hemorrhage.

Hence, the data set for each model contains 6,290 images, where 3,145 belong to the positive class and 3,145 to the negative. For each model, we split the dataset into a training set and a test set where 80% of the data was used for training the model and 20% for testing the model's accuracy. Using this training data, we attempt to run each model until it converges to an estimated minimum loss without overfitting to the training data. By comparing only converged models, we look to compare every model in its optimized version. We test the accuracy of every model on the same amount of images, 1258, to provide consistency when comparing performance across models.

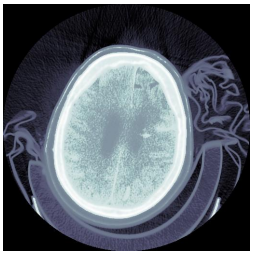


Image 1: Example Image from Dataset

2.3 Models Used

We will use 4 different types of machine learning models to build classifiers for hemorrhages: Logistic Regression, Support Vector Machine, Artificial Neural Network, and Convolutional Neural Network.

2.3.1 Logistic Regression

We use the standard LogisticRegression model from scikit-learn library to build classifiers for each type of hemorrhage. We attempt to run each logistic regression model until convergence, and we try different solvers to determine which solver works most efficiently for our data.

2.3.2 Support Vector-Machine

We use two different Support Vector Machine models from the scikit-learn library to build classifiers for each type of hemorrhage. The first type of SVM we use was the LinearSVC model. The standard SVC implementation scales quadratically with the number of training examples, so we use LinearSVC which is implemented in terms of liblinear rather than libsvm and is intended to run faster for large sample sizes. second type of SVM we use was the SGDClassifier model. The model uses a regularized linear model with stochastic gradient descent to fit a linear support vector machine.

2.3.3 Artificial Neural Networks

We implemented an Artificial Neural Network using the TensorFlow and Keras libraries. Of the 5032 examples in the training set for each model, we use 4528 examples to train the neural network and 504 examples as a validation set to prevent the neural network from overfitting on the training data.

Each example in the neural network is inputted as an array of 16,384 pixels. The first layer of our neural network has 4,096 neurons, each with a ReLU activation. Each successive layer decreases the number of neurons by a magnitude of 4, as we attempt to compress the information from the images. The final layer has a neuron with sigmoid activation and outputs the probability that the hemorrhage belongs to the given class. See *Image 2* for the visualization of this architecture.

Layer (type)	Output Shape	Param #
dense_28 (Dense)	(None, 4096)	67112960
dense_29 (Dense)	(None, 1024)	4195328
dense_30 (Dense)	(None, 256)	262400
dense_31 (Dense)	(None, 64)	16448
dense_32 (Dense)	(None, 16)	1040
dense_33 (Dense)	(None, 4)	68
dense_34 (Dense)	(None, 1)	5

Total params: 71,588,249
Trainable params: 71,588,249
Non-trainable params: 0

Image 2: Artificial Neural Network Architecture

We use a binary crossentropy loss as the metric that network tries to minimize, and use stochastic gradient descent as an optimizer to calculate the new weights of our parameters for every epoch.

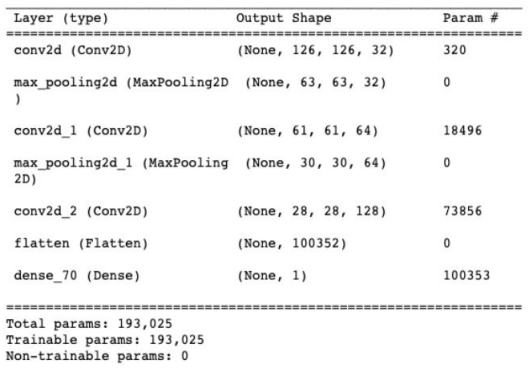
We also use early stopping as a mechanism to prevent the neural network from overfitting. If the validation loss does not improve after 3 epochs, we assume the model has converged and stop training the model.

2.3.4 Convolutional Neural Networks

We implemented a Convolutional Network using the TensorFlow and Keras libraries. Similar to the ANN, we use 4528 examples to train the neural network and 504 examples as a validation set.

Each example in the neural network is inputted as a 128 x 128 x 1 array of pixels. The first layer is a 2D convolution layer that move a 3 x 3 filter across the image. Each neuron in the convolution layer attempts to learn a high-level feature or pattern from the image. The output of each neuron in the convolutional layer is a feature map. After the convolutional layer, we add a pooling layer to compress some of the information in these outputted feature maps. We alternate between Conv2D and MaxPooling2D a few more times, as shown in the architecture below, below flattening all of the learned feature maps into 1 dimensional array. Finally, we attempt to calculate the probability that the image has the given type of hemorrhage using a layer with one neuron that has a sigmoid activation. See *Image 3* for the visualization of this architecture

Similar to the ANN, we attempt to minimize binary crossentropy loss, use stochastic gradient descent to calculate weights, and include early stopping to prevent overfitting.



Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 126, 126, 32)	320
max_pooling2d (MaxPooling2D)	(None, 63, 63, 32)	0
conv2d_1 (Conv2D)	(None, 61, 61, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 30, 30, 64)	0
conv2d_2 (Conv2D)	(None, 28, 28, 128)	73856
flatten (Flatten)	(None, 100352)	0
dense_70 (Dense)	(None, 1)	100353

Total params: 193,025
Trainable params: 193,025
Non-trainable params: 0

Image 3: Convolutional Neural Network Architecture

3 Results

We value recall as a slightly more important metric than accuracy and precision in the context of our project. The accuracy of one of our model describes the percentage of images classified properly. Precision measures the percentage of labeled hemorrhages that are actually. Recall describes the percentage of hemorrhages that were detected by our classifier. Since the goal of our project is to identify hemorrhages when they exist in a scan, we believe it is more important to detect as many hemorrhages as possible while still maintaining a reasonable accuracy. In a high stakes medical environment, false negatives are more costly than false positives.

3.1 Logistic Regression

Classifier	Score	Precision	Recall
Any Hemorrhages	0.655	0.647	0.684
Epidural	0.729	0.705	0.787
Intraparenchymal	0.569	0.568	0.576
Intraventricular	0.634	0.630	0.647
Subarachnoid	0.557	0.555	0.582
Subdural	0.599	0.596	0.615
Averages	0.624	0.617	0.649

Table 1: Logistic Regression Results

The Logistic Regression model worked surprisingly well despite being the most rudimentary model used during this project. We tried different solvers provided by scikit-learn for logistic regression to optimize for speed and accuracy, including batch gradient descent, stochastic gradient descent, and Newton's method. All methods performed similarly in test accuracy, but Newton's method converged far faster, in less than 2 minutes, than the gradient descent implementations, which took up to 5 minutes.

Comparing accuracies across classifying different types of hemorrhages, epidural hemorrhages were easier to classify than other types of hemorrhages. The complete table of results for all logistic regression models can be seen in *Table 1.*

3.2 Support Vector Machines

Classifier	Score	Precision	Recall
Any Hemorrhages	0.631	0.639	0.625
Epidural	0.627	.627	.653
Intraparenchymal	0.552	0.563	0.512
Intraventricular	0.594	0.611	0.545
Subarachnoid	0.533	0.538	0.540
Subdural	0.570	0.574	0.582
Averages	0.585	0.592	0.528

Table 2: LinearSVC Results

Classifier	Score	Precision	Recall
Any Hemorrhages	0.642	0.670	0.561
Epidural	0.665	0.605	0.954
Intraparenchymal	0.591	0.595	0.571
Intraventricular	0.610	0.686	0.404
Subarachnoid	0.534	0.519	0.952
Subdural	0.603	0.621	0.528
Averages	0.608	0.616	0.612

Table 3: SGDClassifier Results

The two SVM models performed quite differently. The LinearSVC model was quite slow, and we could only run it with a training set of 742 images until convergence, which took approximately 5-6 minutes. On the other hand, the SGDClassifier was trained on the full of 5032 images and converged after approximately 30 seconds. While both yielded similar results in terms of accuracy and precision, the stochastic model exhibited more variation in recall. For example, the stochastic model recalled about 95% of the epidural and subarachnoid Hemorrhages, compared to the 40% recall for Intraventricular Hemorrhages. The LinearSVC model had recall between 50%-65% for all models. The results from building these models are shown in *Table 2* and *Table 3*.

3.3 Artificial Neural Networks

Classifier	Score	Precision	Recall
Any Hemorrhages	0.683	0.620	0.948
Epidural	0.616	0.571	0.925
Intraparenchymal	0.596	0.578	0.698
Intraventricular	0.647	0.613	0.795
Subarachnoid	0.540	0.610	0.230
Subdural	0.619	0.691	0.431
Averages	0.617	0.615	0.671

Table 4: Artificial Neural Network Results

The artificial neural network also took longer to run than the logistic regression model, with each epoch running for 25-40 seconds. We tried different optimizers for the neural network, but found that the stochastic gradient descent performed far better than the adam optimizer. When trying out different network architectures, we found that the last neuron needed to have a sigmoid activation in order to accurately achieve the task of binary classification.

Since we implemented early stopping to prevent overfitting, each model took about 7-10 minutes to converge. Lastly, we attempted to implement L1 and L2 regularization to prevent overfitting, but it backfired as altering the weights made the model much more inaccurate. We determined that early stopping sufficiently prevented overfitting, as the difference between training and test accuracy was marginal. The results for the Artificial Neural Network are shown in *Table 4*.

3.4 Convolutional Neural Networks

Classifier	Score	Precision	Recall
Any Hemorrhages	0.680	0.617	0.944
Epidural	0.688	0.677	0.722
Intraparenchymal	0.572	0.588	0.483
Intraventricular	0.616	0.569	0.959
Subarachnoid	0.564	0.539	0.901
Subdural	0.599	0.570	0.806
Averages	0.620	0.593	0.803

Table 5: Convolutional Neural Network Results

The Convolutional Neural Network exhibited the best recall, identifying hemorrhages with an average recall of above 80%. This was considerably greater than the next highest average recall, which measured at 67.1% for the artificial neural network. The CNN was more consistent in terms of accuracy compared to previous models. The trade off for the CNN's higher accuracy was a longer time to convergence, with each model taking up to 12 minutes. We made similar observations for training the CNN as we did when training the ANN, where L1 and L2 regularization hurt the performance and a final layer of one neuron with sigmoid activation was needed. The results for the Convolutional Neural Network are shown in *Table 5*.

4 Conclusions

Across the course of our project, we found that each model uniquely performed well at classifying some types of hemorrhages but not at all types. While we weren't able to create a single model that worked the most effectively, a synthesis of models across different types like SVM and CNN with provides a set of models that can accurately identify every hemorrhage class. The highest recall values and the models that achieved these values are listed for each hemorrhage class in *Table 6*.

Classifier	Model	Recall
Any Hemorrhages	ANN	0.948
Epidural	SVM	0.954
Intraparenchymal	ANN	0.698
Intraventricular	CNN	0.959
Subarachnoid	SVM	0.952
Subdural	CNN	0.806

Table 6: Most Effective Models for Hemorrhage Classes

5 Future Work

5.1 Transfer Learning

Transfer Learning is a powerful technique that attempts to use pre-existing information stored in one machine learning model in order to classify a separate set of data.

Some benefits of applying transfer learning to the current problem would be leveraging models that have had hundreds of thousands of images already inputted in and tested against. With the sheer number of images already used in a model, the accuracy of the new model would be much higher without having to put in a lot of new training and test data. In fact, one should expect not only a higher accuracy, but a higher learning rate as well as faster training. Clearly, applying transfer learning to any model would only be highly beneficial.

Given that we were trying to use image data to classify hemorrhages it made sense to use popular computer vision and image processing pretrained models such as ImageNet and ResNet50. We specifically attempted to use ResNet50 because of the similarities it shared with the architecture of CNN and ANN models. ResNet50 used input images represented in the form 128 * 128 * 3, while our previous CNN model has an input of 128 * 128 * 1.

At the moment there is not a specific pretrained model for grayscale or medical images. Therefore, the only options to use transfer learning on our dataset is to find a way to reasonably store our 3 RGB values or build our own transfer learning model that uses grayscale images. Unfortunately we did not have enough resources to do either, however given more time with this dataset, it would be interesting to see how much more accurate a transfer model would be.

5.2 High Performance Computing

When developing machine learning models for all the specified techniques, it was all run on one Jupyter Notebook with the limitations of a Macbook CPU and GPU. We were constrained by this computing capacity and hence had to use a smaller subset of images to train and test our models. While doing work like this in the future using a machine with more computing power or a stronger GPU would be helpful in processing the large data set. Northeastern's Discovery High Performance Computing cluster would have been optimal for training and testing our models on a much larger dataset of images, perhaps the entire cache sent by Zeta Surgical. While there would be have been significant advantages to using high performance computing resources, it would have still taken a lot of time with epochs potentially taking hours to converge. With the given time limitations, we chose to not pursue using these resources.

6 Acknowledgements

We would like to thank Zeta Surgical for providing the data used in building these models.

7 Bibliography

1. A. Majumdar, L. Brattain, B. Telfer, C. Farris and J. Scalera, "Detecting Intracranial Hemorrhage with Deep Learning," 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2018, pp. 583-587, doi: 10.1109/EMBC.2018.8512336.

2. Al-Ayyoub, Mahmoud Alawad, Duaa Al-Darabsah, Khaldun Aljarrah, Inad. (2013). Automatic Detection and Classification of Brain Hemorrhages. WSEAS Transactions on Computers. 12. 395-405.

3. Arab, A., Chinda, B., Medvedev, G. et al. A fast and fully-automated deep-learning approach for accurate hemorrhage segmentation and volume quantification in non-contrast whole-head CT. Sci Rep 10, 19389 (2020). https://doi.org/10.1038/s41598-020-76459-7

4. Hang Chen, Sulaiman Khan, Bo Kou, Shah Nazir, Wei Liu, Anwar Hussain, "A Smart Machine Learning Model for the Detection of Brain Hemorrhage Diagnosis Based Internet of Things in Smart Cities", Complexity, vol. 2020, Article ID 3047869, 10 pages, 2020. https://doi.org/10.1155/2020/3047869

5. Jeremy Jordan. (2018, October 20). Common architectures in convolutional neural networks. Jeremy Jordan. Retrieved December 11, 2021,

from https://www.jeremyjordan.me/convnet-architectures/.

6. Sage, A.; Badura, P. Intracranial Hemorrhage Detection in Head CT Using Double-Branch Convolutional Neural Network, Support Vector Machine, and Random Forest. Appl. Sci. 2020, 10, 7577. https://doi.org/10.3390/app10217577

7. Sha, Raahil, RS. (2021, October 18) Zeta Company Introduction [Powerpoint]. Zeta Surgical file:///C:/Users/18189/Downloads/Deck%20%E2%80%93%20Zeta.pdf

8. Wu, Y., Supanich, M. P., Deng, J. (2021, June 23). Ensembled deep neural network for intracranial hemorrhage detection and subtype classification on noncontrast CT images. Journal of Artificial Intelligence for Medical Sciences. Retrieved December 5, 2021, from https://www.atlantis-press.com/journals/jaims/125958274/view.