Brain Hemorrhage Detection through ML

Kevin Ciardelli, Alicia Doung, Saskriti Neupane

A. Introduction

Intracranial Hemorrhage (ICH), or bleeding within the skull, is a severe health issue that accounts for approximately 10 percent of strokes in the United States. It is one of the top five causes of fatal health complications, requiring quick and accurate diagnosis to prevent further damage. ICH can vary greatly in severity and impact, depending on factors like the location, size, and type of hemorrhage. Rapid diagnosis is critical, especially in cases where patients show acute symptoms such as sudden loss of consciousness.

Given the complexity and time-consuming nature of traditional diagnostic methods, this project proposes an innovative solution to automate the preliminary assessment of intracranial hemorrhages. By using an advanced algorithm to identify the location, type, and size of the bleed, the diagnostic process can be significantly expedited, enabling doctors to make faster, more accurate decisions. This approach also reduces the likelihood of human error and can improve over time by learning from each scan analyzed. Ultimately, the project aims to build a robust algorithm that can reliably detect acute intracranial hemorrhage and its subtypes, providing a crucial tool for medical professionals and potentially saving lives.

B. Related Work

Previous research has investigated various methodologies for hemorrhage detection, including techniques such as K-means clustering and Neural Networks. For instance, one approach combined K-means clustering with the watershed algorithm to refine medical image segmentation [Ng, et al.]. Despite its enhancements, this method often struggled with over-segmentation and noise sensitivity, particularly in complex brain areas.

In another significant study, ”Detecting Intracranial Hemorrhage with Deep Learning,” convolutional neural networks (CNNs) were employed for automated hemorrhage detection from CT scans [Majumdar, et al.]. The study utilized a modified U-Net model, showcasing comparable sensitivity and significantly higher specificity. Although offering strengths like automated detection and improved identification of subtle hemorrhages, this approach faced challenges such as data limitations and computational demands.

Given the effectiveness of neural networks in addressing classification challenges, we opted for CNNs in our approach. Specifically, we chose ResNet due to its capability to handle deeper networks and enhance feature extraction, essential for accurate hemorrhage detection in CT scans.

C. Method

C.1. CNN Classifiers

Based on our findings on previous research, we decided to move forward with the methodology of CNN classifiers. Convolutional Neural Network classifiers work by sending an image through multiple layers of filters to detect spatial hierarchies such as edges, textures, and other patterns. After each layer an activator function known as ReLU introduces non-linearities into the model to further identify more complex patterns within the image itself. After this step, pooling the image by reduction of spatial dimensions take effect to reduce computational load, memory usage, and the detection of features invariant to scale and orientation changes.

Throughout the evolution of machine learning techniques within the realm of image recognition and classification, models such as ResNet, AlexNet, and DenseNet have emerged as front runners. Though different in techniques, these architectures have significantly advanced the field, setting new benchmarks for accuracy and efficiency in processing and analyzing visual data.

C.2. FastAi

Given the wide array of models available, we needed to identify a method that could leverage these diverse models efficiently both in accuracy and in a timely manner. FastAi is an open-source library design to simplify and accelerate the process of using deep learning models. The framework is built on top of the widely known pre-existing machine learning software PyTorch. The high level interface allows developers to quickly allocate specifications to their model with ease.
These programming lines allow the user to set batch size, transformations per convolution layer, percentage of training data versus validation and more. The danger with using this more template style of machine learning, is it is a high level of abstraction. In order to get the best use of the models you choose to proceed with, you must understand the models themselves and what the ideal circumstances are. This involves preprocessing the dataset in different ways, different batch sizes, numbers of epochs and more. Keeping this in mind, we decided to move to our first model. Core programming lines include:

```
ImageDataLoaders.from_df(labels, ’train_jpg/’, bs = 32)

vision_learner(dls, resnet18, metrics=accuracy_multi)

learn_baseline.fit_one_cycle(2, lr_rate)
```

C.3. Single Label vs Multi-label Classification

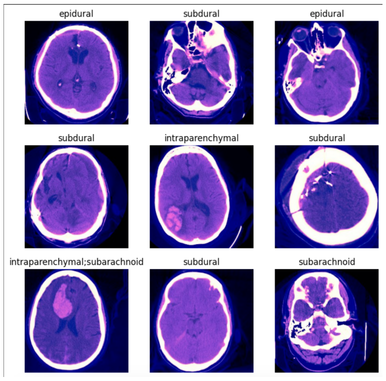
Due to our project encircling the task of classifications of five different brain hemorrhages, we decided to break it down into three main processes. Our first method would try testing whether or not the model could perform simple classification on whether or not a hemorrhage is present within an image. From there, we can move into a single-labeling classification where it can identify a single type of hemorrhage, and then ultimately identify multiple different classifications of hemorrhages in one image. This process would simplify the structure of our experiments, while also building a consistent way of testing across models.

Increasing technical complexities in these differences align with the escalating challenges as we approach the sophisticated objective of multi-label classification. For example, for single level classification we can use a categorical entropy loss function when determining accuracy which then switches to a binary entropy loss function that has to find the accuracy of each accuracy across categories and takes the average. Data management also becomes more difficult as we need to take into consideration the uneven amount of types of hemorrhages within our data. Keeping details such as these becomes essential when moving forward into our experimental phase.

D. Experiments

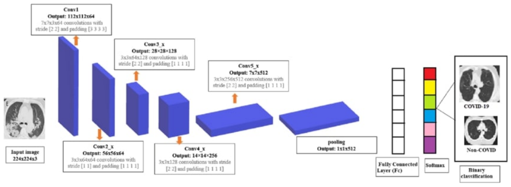
D.1. Data

The data we will be using is supplied by four different universities: Stanford, Thomas Jefferson, UHT, UNIFESP. This data consisted of 194,082 CT scans varying in the different types of classification. This was an uneven split between the five varying in proportions: aprox. 3,000 epidural, 24,000 intraventricular, 32,000 of subarachnoid and intraparenchymal, and 42,000 subdural. It is important to keep this disparity in mind as our process will take the averages of each classification to give an overall accuracy rate. We then proceeded to shrink this down proportionately to a batch of 1,500 for efficiency purposes. This set was broken down into 1238 files for testing and 262 files for validation.



D.2. ResNet18

Starting with ResNet18 is essential as it serves as a method that has a balance between performance and computational efficiency while serving as a foundation for further research.


Conv1
Output: 112x112x64
7x7x3x64 convolutions with stride [2 2] and padding [3 3 3 3]
Conv3_x
Output: 28×28×128
3x3x64x128 convolutions with stride [2 2] and padding [1 1 1 1]
Conv5_x
Output: 7x7x512
3x3x256x512 convolutions with stride [2 2] and padding [1 1 1 1]
Input image
224x224x3
Conv2_x
Output: 56x56x64
3x3x64x64 convolutions with stride [1 1] and padding [1 1 1 1]
Conv4_x
Output: 14×14×256
3x3x128 convolutions with stride [2 2] and padding [1 1 1 1]
pooling
Output: 1x1x512
Fully Connected Layer (Fc)
Softmax
COVID-19
Non-COVID
Binary classification

ResNet18 is a type of CNN classifier that features a distinctive architecture that addresses the vanishing gradient problem common in deep neural networks. It consists of 18 layers, including initial and final layers, and multiple residual blocks. Each block contains two convolutional layers with batch normalization and ReLU activation, connected by skip connections that add the input of the block to its output. These connections enable gradients to flow directly through the network, facilitating deeper model training without performance degradation. This structure not only improves training dynamics but also enhances generalization on image classification tasks.

By using this as our base method, we can use the results of our first experiment to see how different models and optimizations can improve these initial findings. Through two epochs we ended with a 67 percent accuracy rate on our multi label data. We will use this as our baseline to improve upon moving foward

D.3. Resnet36 Results

To better capture the complexities within hemorrhages, we transitioned from ResNet18 to ResNet36, a deeper architecture designed to better handle complex patterns. ResNet36's architecture comprises multiple stages of residual blocks, with each stage progressively deepening the network's capacity to capture intricate features. The initial experimentation with ResNet36 yielded promising results, showcasing notable reductions in both training and validation losses across epochs. Despite a slight increase in the error rate by the third epoch, ResNet36 demonstrated a substantial improvement in losses compared to ResNet18 over a similar timeframe. However, it's worth noting that the training times for ResNet36 were approximately doubled due to its heightened complexity and depth, necessitating additional computational resources and time.

Overall, ResNet36 exhibited enhanced performance in capturing complex patterns within hemorrhages, showcasing its potential as a robust architecture for our task. Despite the increased computational demands, the significant reductions in losses underscore its effectiveness in learning intricate features, highlighting its suitability for further experimentation and optimization.

D.4. Resnet36 with CeLu

To further enhance ResNet36, we introduced the SiLU (Sigmoid-Weighted Linear Unit) activation function, offering a smoother alternative to the conventional rectifier unit. SiLU combines the sigmoid function with input values, closely approximating ReLU behavior for large magnitude inputs. Unlike ReLU, however, SiLU exhibits a non-monotonic activation pattern, with a global minimum at approximately -0.28 for specific input values. Our experimentation revealed that ResNet36 with SiLU activation exhibited a lower initial training loss of 1.028587 compared to the traditional activation function. However, this came at the expense of a slightly elevated validation loss of 0.882539 and a notably higher error rate of 0.357500. Additionally, the training duration with SiLU activation was substantially prolonged, lasting 20 minutes and 12 seconds, nearly twice the time required for the traditional activation setup.

These initial observations highlight the potential benefits and challenges associated with SiLU activation. While it may enhance the network's initial learning efficiency by reducing the training loss, it also introduces complexities in terms of error rate and training duration. The elevated error rate and validation loss suggest potential difficulties in generalizing to the validation set during the early stages of training. Moreover, the significant increase in training time underscores the higher computational demands associated with SiLU activation. Thus, while SiLU activation offers promise in improving initial learning dynamics, achieving optimal generalization and stability may necessitate further adjustments and additional training epochs.

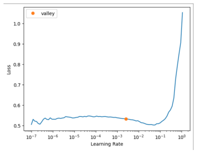
D.5. Alexnet Results

AlexNet is a convolutional neural network that was developed for image classification tasks. The architecture of AlexNet consists of five convolutional layers followed by three fully connected layers. Key innovations introduced with AlexNet include the use of ReLU as the activation function to speed up training by addressing the vanishing gradient problem common in deep networks. Additionally, AlexNet employs overlapping max pooling to reduce the size of the output from one layer to the next, enhancing the network's ability to capture features while preventing overfitting. Dropout is another critical technique used in the fully connected layers to further combat overfitting by randomly dropping units during training. The network also utilizes data augmentation methods like image translations, horizontal reflections, and alterations in intensity to diversify the training data and improve generalization.

Within our data Alexnet proved to be a very efficient model with a 73.94 accuracy rate over 4 epochs. The best takeaway from AlexNet was the time per epoch was very low, with a maximum of 2 minutes per epoch. Moving foward it was important to keep this in mind when thinking about future optimization practices and what model we would come to choose as the most optimal.

D.6. Alexnet with Optimized LR

To optimize, we chose to implement Leslie's Learning rate alongside AlexNet. The cyclical learning rate strategy involves oscillating the learning rate between a lower bound and an upper bound within a predefined range and cycle length. This approach helps in several ways: it allows the model to escape local minima in the loss landscape by periodically increasing the learning rate, which provides a form of automatic resetting and broad exploration. Then, as the learning rate decreases, the model can refine its adjustments to converge on more precise solutions.



By using this in tandem with AlexNet, we see a better time per epoch (1:20), and an accuracy rate of 74 percent. The take away from using this method was its applicability to any model. Moving forward with our models, we can apply this optimization to see further improvements.

D.7. DenseNet Results

Our exploration of DenseNet161, renowned for its dense connectivity structure, has yielded promising outcomes, positioning it as a formidable contender for our task. This architecture's dense blocks, characterized by intricate interconnections among layers, offer substantial advantages. DenseNet161 not only facilitates improved information flow throughout the network but also effectively mitigates the vanishing gradient problem, a common challenge in deep neural networks. Furthermore, its emphasis on dense connections enhances feature reuse, maximizing the utilization of learned representations across various layers. In our preliminary experiments with DenseNet161 on the provided dataset, we observed notable progress within just two epochs. From epoch 0 to epoch 1, we witnessed significant decreases in both training and validation losses, alongside a marked increase in accuracy. Specifically, the training loss decreased from 0.878568 to 0.641335, and the validation loss decreased from 0.614177 to 0.471099. Correspondingly, the accuracy improved from 0.734667 to 0.786667. These promising results underscore the efficacy of DenseNet161 in learning from data and generalizing effectively to unseen samples. As we continue to delve deeper into its capabilities, further optimization and fine-tuning hold the promise of unlocking even greater performance enhancements, solidifying DenseNet161 as a robust choice for various machine learning tasks.

D.8. DenseNet with Cutmix Results

The experimental results with DenseNet161 trained with CutMix did not exhibit the anticipated improvements in performance compared to the baseline results. Despite the expectation of enhanced robustness and generalization capabilities, the observed outcomes did not reflect a significant advancement. Specifically, over two epochs, we observed a moderate decrease in both training and validation losses, with a slight increase in accuracy. In epoch 0, the training loss was 0.743620, the validation loss was 0.510875, and the accuracy was 0.726667. By epoch 1, the training loss decreased to 0.540349, the validation loss slightly decreased to 0.478400, and the accuracy increased marginally to 0.753333.

Several factors might contribute to this unexpected outcome. One possibility is that the dataset may not have contained sufficient diversity or complexity to fully benefit from the CutMix augmentation. Additionally, while CutMix encourages the model to learn more general features, it might have inadvertently introduced noise or inconsistencies that hindered performance. Furthermore, the complexity of the task and model architecture could have influenced the effectiveness of the augmentation technique. Further analysis and experimentation are warranted to gain deeper insights into the interplay between CutMix and DenseNet161 and to explore potential adjustments or alternative augmentation strategies to maximize performance gains.

E. Conclusions

After evaluating all three models, we determined that AlexNet offers superior efficiency in achieving high accuracy. To enhance its performance, we identified several key strategies. Firstly, increasing the number of training epochs would enable the model to capture more intricate patterns within the data, thereby enhancing its predictive capabilities. Secondly, maintaining optimal learning rates is crucial to ensure effective convergence during training. Regarding data augmentation, we proposed several approaches to enrich the diversity of the training dataset. One such strategy involves experimenting with AlexNet using CutMix, a technique that blends images to encourage the model to learn from mixed samples. Additionally, we recommended further augmentations through transpositions to introduce variations in image orientation, facilitating better generalization to unseen data. Furthermore, leveraging a larger portion of the dataset is essential to capture a more comprehensive representation of underlying patterns. Implementation of these optimization and data augmentation techniques holds the potential to further improve AlexNet's performance, paving the way for more accurate and robust predictions in future applications.

F. Contributions

Kevin Ciardelli worked on sections C, D.1, D.2, D.3, D.6, and D.7. Alicia Doung worked on sections A, D.8, D.9, and E. Saskriti Neupane worked on sections B, D.4, and D.5.

https://github.com/KevinCiardelli/HemorrhageDetection

References

Ng, H. P., Ong, S. H., Foong, K. W. C., Goh, P. S., & Nowinski, W. L. (2006). Medical Image Segmentation Using K-Means Clustering and Improved Watershed Algorithm. *IEEE Southwest Symposium on Image Analysis and Interpretation* (pp. 61-65). Denver, CO, USA. doi: 10.1109/SSIAI.2006.1633722.

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