samoliver3 / capstone-STRIP-AI Public

MIT license

0 stars 0 forks

Star Unwatch

Code Issues Pull requests Actions Projects Wiki Security Insig

main

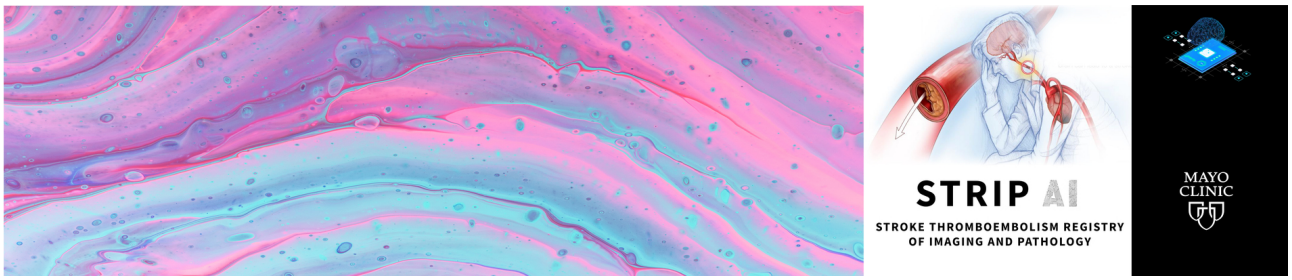
samoliver3 Updated presentation ... 23 minutes ago 89

View code

README.md

Predicting Acute Ischemic Stroke Etiology from Whole Slide Digital Pathology Imaging of Clots Retrieved via Mechanical Thrombectomy

Created by Sam Oliver

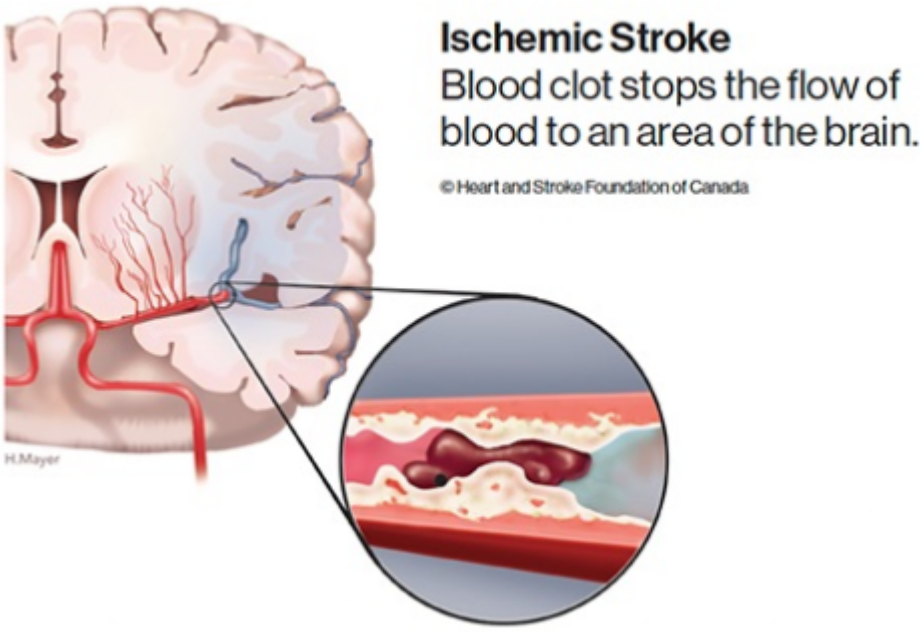


Project Overview

In the United States, over 700,000 people suffer an ischemic stroke every year, and stroke is the second most common cause of death in the world. Around 25% of people that have a stroke suffer one or more additional strokes. Identifying stroke etiology improves the likelihood of survival. Stroke Thromboembolism Registery of Imaging and Pathology (STRIP) is a project run by the Mayo Clinic Neurovascular Lab across multiple centers with the following goal: histopathologic characterization of thromboemboli of various etiologies and examining clot composition and its relation to mechanical thrombectomy revascularization. This project takes advantage of resources collected by STRIP- 1,000 Whole Slide pathology images of mechanical thrombectomized blood clots- to predict stroke etiology (the origin of the stroke). These images are high resolution (typical size of 25-50k width and 25-50k height pixels) and each image represents a blood clot from one of the two major stroke etiologies. The task of this project represents predictions as to whether a stroke ocurred as one or the other of the main two acute iscehmic stroke etiology subtypes: cardioembolic (CE) or large artery atherosclerosis (LAA). Ultimately, the best modeling results signify only slightly better predictions than randomly guessing the etiology. My model does not solve this problem well, and deployment of this model as a diagnostic function would not be recommended.

The predictions in this model were carried out using deep convolutional neural networks will input images of blood clots that have been extracted from the brain of the patients in this dataset.

At this point, it is helpful to define some of the terms noted in this introductory paragraph.



Acute ischemic stroke (AIS): An ischemic stroke refers to a blood clot that blocks or narrows an artery leading to the brain. Acute, in this context, refers to a sudden medical episode related to a pre-existing condition. Cardioembolic AIS: commonly defined as cerebral vessel occlusion by distant embolization arising from thrombus formation in the heart. CE is essentially the blocking of blood flow in the brain caused by a blood clot in the heart (coronary thrombosis). Large Artery Atherosclerosis AIS: Atherosclerosis - changes in the walls of blood vessels that occur as a result of inflammation and the accumulation of fatty deposits. Atherosclerosis causes arteries to narrow, which impedes blood flow. Atherosclerotic plaque buildup can then rupture and cause the formation of blood clots that can block arteries in the brain, which can then cause stroke.

To reiterate, the goal of this project is to create useful predictions as to whether a stroke occurred from a clot in the heart or froom large artery atherosclerosis from an image of the removed blood clot from the patient's brain. The point of solving this project is to utizile beneficial prediction methods to then appropriately prescribe therapeutic treatmeant for patients that have suffered one of these kinds of strokes, which allows for more specialized treatment depending on which kind of stroke the patient had and will increase survival odds for the patient.

The Stakeholder

The stakeholder is the Mayo Clinic, which is offering cash rewards to the creators of the top five best-scoring entries according to a scoring algorithm. The Mayo Clinic is asking for entries to classify images into one of the two major AIS categories for each image in a test set. These two major categories include Cardioembolic (CE) strokes and Large Artery Atherosclerosis (LAA) strokes. The Mayo Clinic would ultimately work with and provide other incentives (contracts, paid positions, etc) for the individuals or teams with the best scoring entries.

Navigation Instructions

This project utilizes two Jupyter Notebooks for all the code and data exploration used throughout the project. One of these notebooks was hosted on Google Colab, and this notebook created an allowance of more computational resources that could not be sourced from my own computer. Google Colab allowed for the use of more advanced GPUs and TPUs as well as approximately 32GB of RAM. These resources represent much greater computational capacities than my own machine. The other notebook utilized my own machine. This project also includes some PDFs for the Github repository, the notebooks, and a slideshow of the main findings for this project. This project utilized data from a Kaggle competition run by the Mayo Clinic. This project utilizes the original data provided by the Mayo Clinic as well as other datasets compiled by Kaggle users that were used in order to avoid OOM issues and other issues encountered by the large file sizes of the original images. Links to these datasets and other resources are below.

Links to Important Resources

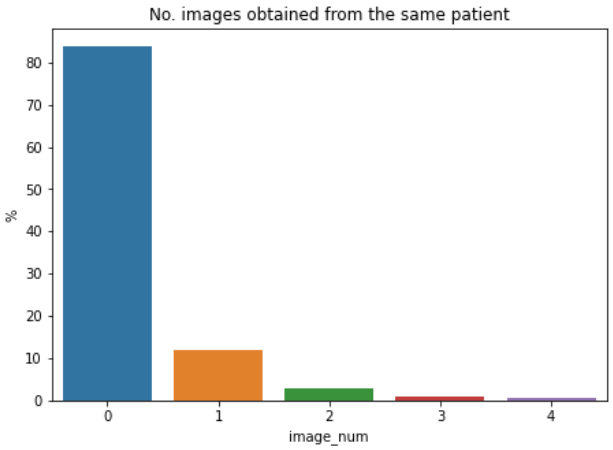
1. Project Presentation
2. Modeling and data exploration carried out on Google Colab
3. Modeling and data exploration carried out by local machine
4. Original dataset provided by the Mayo Clinic on Kaggle
5. Downsized png dataset
6. Tiled dataset

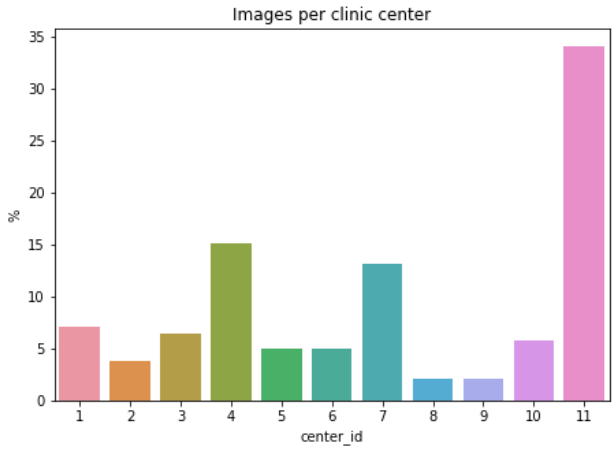
Replication Instructions

An important note - the environment used for this project can be found in the requirements.txt file in this repository. All other resources used should be documented within the modeling notebooks.

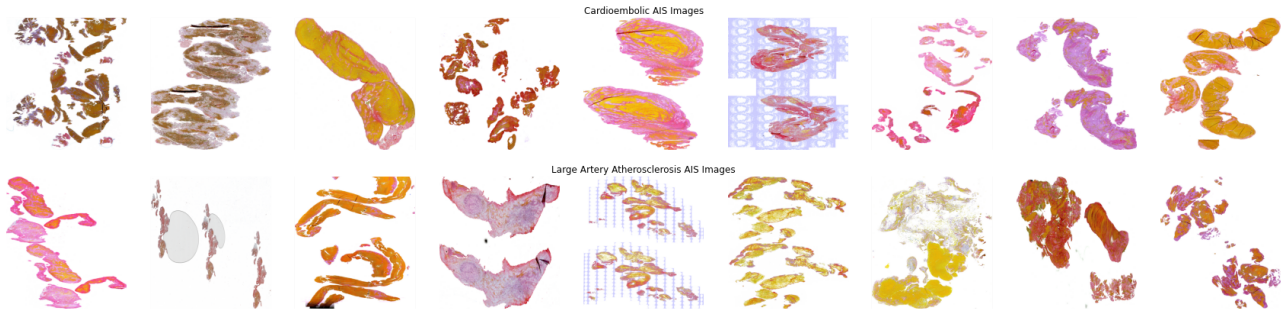
The Data

The dataset provided by the Mayo Clinic contains over a thousand high-resolution whole-slide digital pathology images, and the whole dataset represents about 400 GB of data. Each slide depicts a blood clot from a patient that had experienced an acute ischemic stroke. The dataset contains roughly a 2:1 ratio of CE to LAA blood clots. Some patients have multiple representations of blood clots in the dataset, and there are 11 different clinics that have contributed to the image registry.

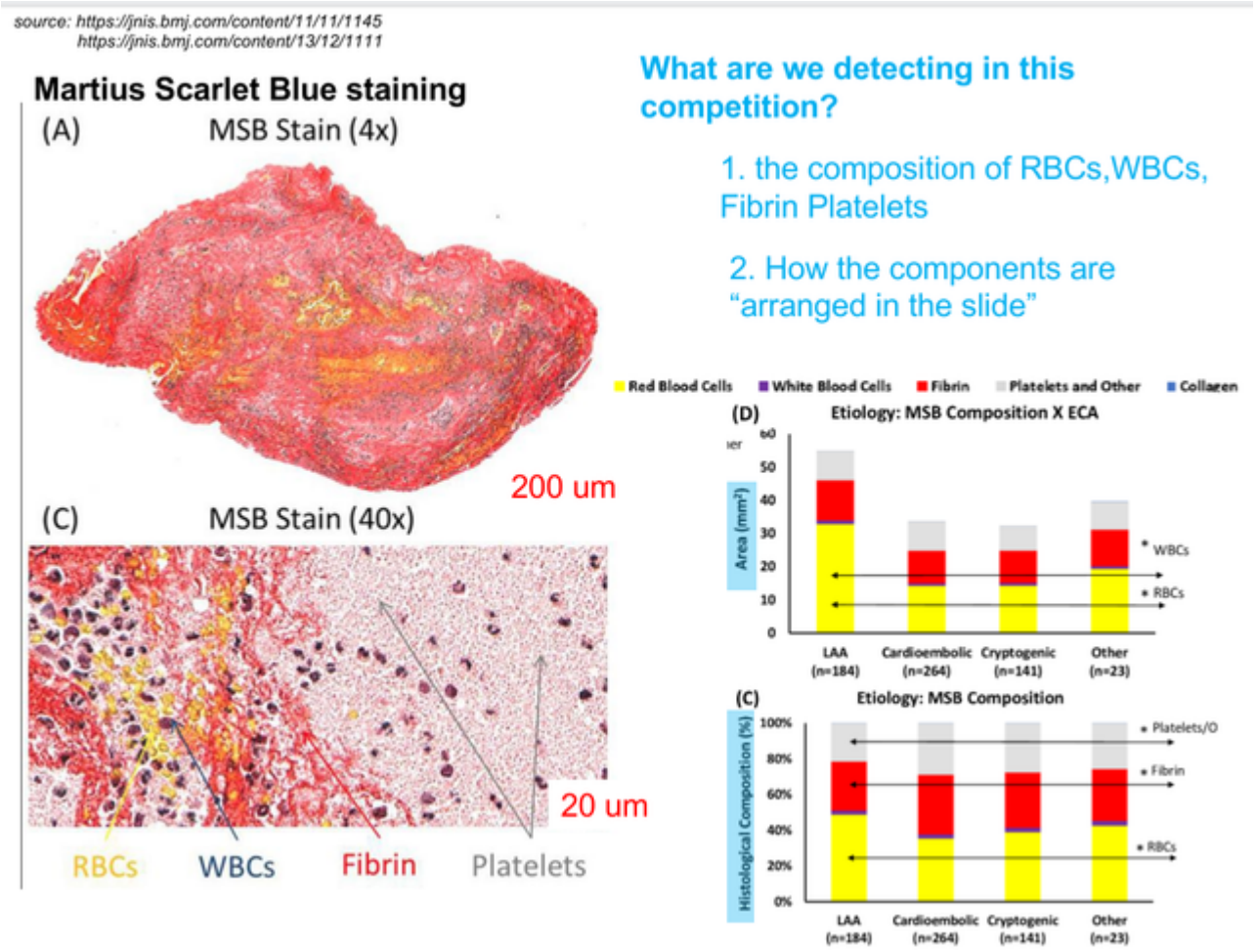




A preview of some of the images from each category (CE and LAA) can be seen below.

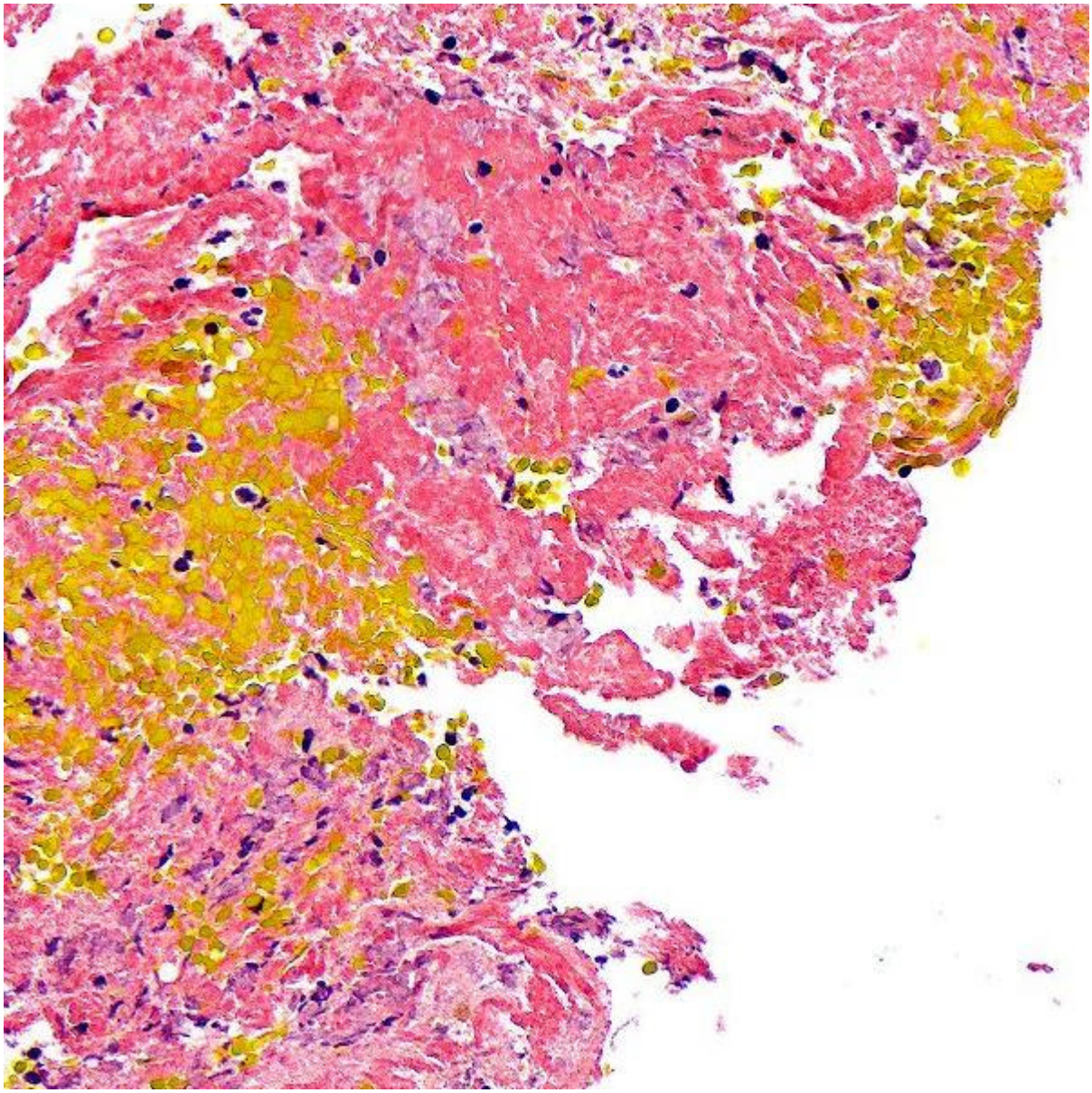


Spot any differences between the different categories? Differences are not detectable by the human eye, and the medical literature dealing with this subject points to techniques that could potentially unlock feature detection techniques to differentiate between the two; but it is not entirely clear what makes these two major types of strokes different from each other when given an image of the blood clot. However, perhaps the most remarkable work on feature differentiation between the two categories indicates that there might be a significant difference in composition between LAA and CE, especially in % composition of red blood cells. This study uncovers potential differences, and a summary of these findings can be viewed below.



It seems like the medical literature on this subject is guiding methodology for discovering feature detection techniques, but differentiation between these two categories is a task for machine and deep learning neural networks. According to the Kaggle competition host, trained medical practicioners are not able to detect differences between these two, and the hope for this project is to facillitate computational learning to create useful predicitons to differentiate the two categories. This notion makes it difficult to guide modeling techniques for feature detection, but it is possible that there are important differences such as the composition of RBCs, dWBCs, fibrin, and other blood components.

The images in the original dataset, the Whole Slide digital pathology dataset, are very large images with sizes that typically range from around .75 GB to 1.5 GB. This size makes it difficult to handle these images and storing just a few of them in memory can easily crash memory. For this reason, I used a variety of techniques including downsizing these images after reading them into memory, using downsized image datasets, and using tiled datasets. The disadvantages of using downsizing techniques is that much of the resolution is lost, which is theoretically very problematic becuase without a direct approach to feature detection, it can be assumed that it is quite possible that signal differentiating the two categories can be easily lost when decreasing the resolution. For this reason, I turned to using a tiled dataset that maintains high resolution and even deletes tiles when the majority of its space is empty. An example of a tile found in this dataset is below.

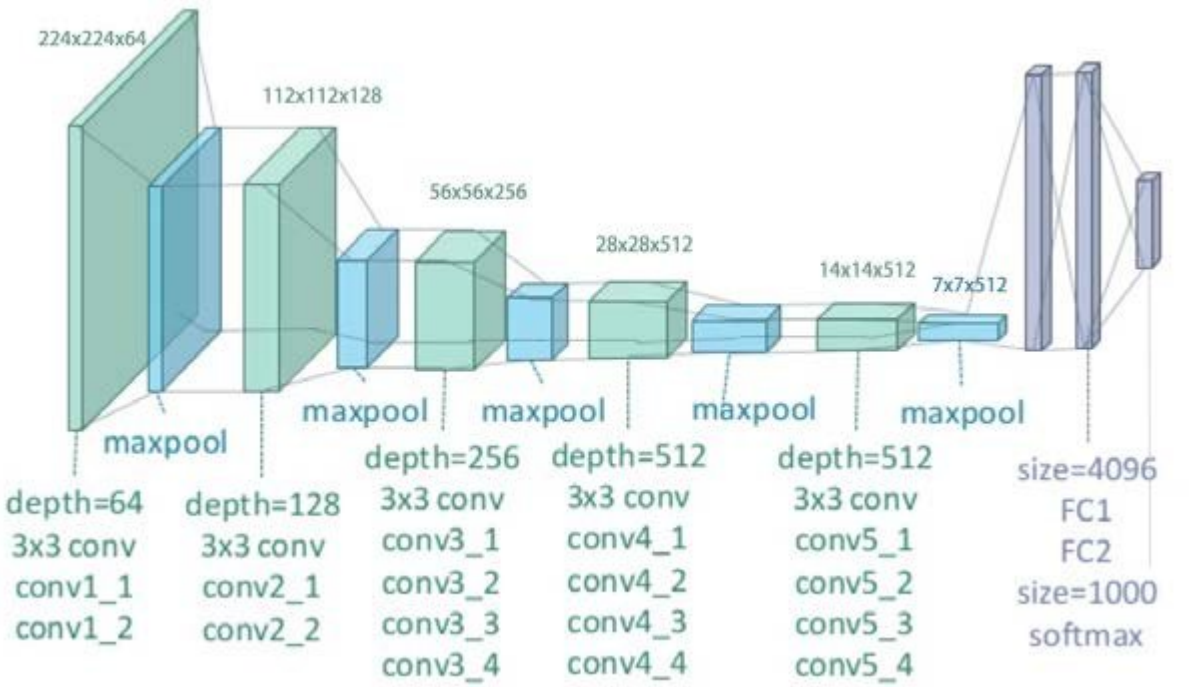


The entire tiled dataset from this source contains around 120,000 images, and as observed from this example tile, the high-resolution of the original WSI is maintained.

Methods

First, the data from the original dataset and the downsized png dataset were evaluated and previewed along with various metadata details such as number of clinics contributing images to the dataset, number of images per patient, etc. A simple convolutional neural network was created as a baseline model, and the model simply guessed that each image represented the CE category. This initial model was created with the downsized png dataset for the Colab notebook and with the original WSI dataset for the local notebook. The original basis for this architecture is adapted from Francois Chollet and his blog post titled "Building powerful image classification models using very little data".

After this model, VGG-19 was utilized for the next model's architecture. The architecture of this model can be observed below.

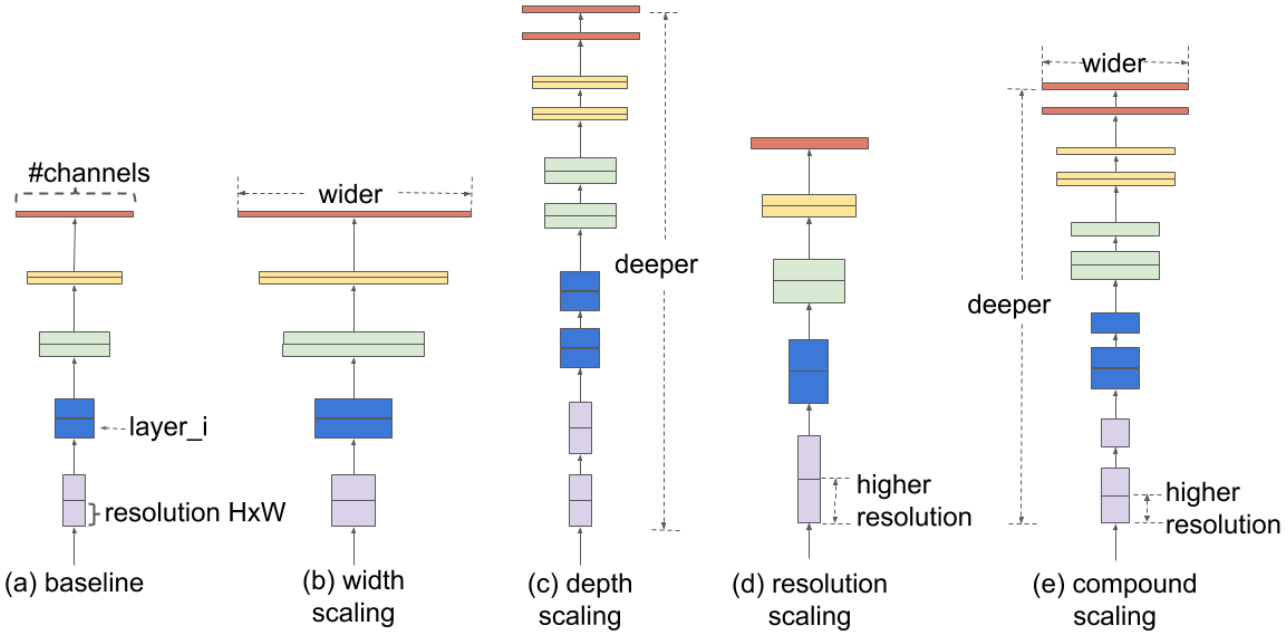


I tweaked some parameters for each model, and I found that an Adam optimizer typically worked well with a moderate to low (1e-6) learning rate. I also used a binary crossentropy loss function for my models as this loss function was fairly compatible with how the Kaggle competition results were scored. This model did predict that some images belonged to the LAA category, but it's false positive rate was high and resulted in worse predictive results that simply randomly guessing the CE or LAA category for each image.

After this model, I introduced Keras's ImageDataGenerator to permute the images in hopes that these permutations would allow the model to pick up on feature importances. I also allowed for higher resolution of the images (512x512 pixels) as opposed to a smaller resolution in the first VGG-19 model instance that used 256x256 image size to allow the model to read in better quality of images. However, this image quality still is not comparable to the extemely resolute images found in the original WSI imaga dataset, which has a typical image size of around 20k-50k pixels in both height and width parameters. I could not get around losing much of the resolution of these images due to resource constraint. This iteration of VGG-19 indicated that each image in the validation set was CE, which is what the baseline model indicated.

I then turned to the tiled dataset in the Google Colab notebook. I created a basline model with similar architecture as that of my original baseline model, and I received similar results as my original baseline with the model simply guessing that each validation tile belonged to the CE category.

I saw on Kaggle that some users were supposedly having some small successes with EfficientNet, so I gave this model a try. EfficientNet is a CNN that uniformly scales all dimensions of an image using a compound coefficient. Its architecture can be seen below.



As seen here, the EfficientNet architecture uniformly scales the dimensions of the images fed in using a fixed ratio. To avoid memory overflow, I used EfficientNetB0, which reads in images as 224x224 size. It would have been nice to read in images at higher resolution, but I kept having issues with OOM, so I stuck with version B0. This model did predict some validations as belonging to the LAA category, but ultimately, it still showed worse results than simply guessing one or the other category or randomly guessing.

I also tried to use this same model architecture on the Whole Slide images in my local modeling notebook, but I did not have success with this model either.

I heard some Kaggle users had promising results with Multiple Instance Learning (MIL) modeling, so I also tried this trategy. I implemented this strategy with Keras using this article. MIL is a commonly used strategy for WSI classification problems because WSIs have very high resolutions and typically do not have data specific or compositional-based data labeling within the image. In the context of this problem, it is unclear where the signal is for differentiating between the two categories, and using MIL accompanied with tiled data aims to solve exactly this issue. This model first extracts feature embeddings, feeds these into an attentino layer, and then multiplies features and attention scores together to then create a prediction. More information on the specifics can be found in the Colab notebook or the MIL Keras tutorial. My MIL model predicted that each tiled image in the validation set belonged to one of the classes, which is consistent with much of the other poor results in my other models. There is definitely a lot of room for improvement within how I carried out MIL modeling in this project, and it seems like some of the successful models in the Kaggle competition were able to successfully utilize MIL.

I also tried to use the Xception deep learning architecture. I have not tried this architecture ever before, and admittedly, I only tried this kind of model in this project becuase my instructor suggested doing so. This kind of CNN represents an "intermediate step in-between regular convolution and the depthwise separable convolution operation (a depthwise convolution followed by a pointwise convolution)." This quote is from a Francois Chollet article describing this model. I experimented with different parameters with this model and tried it several times, but I was not able to find any success with it.

Ultimately, I did not have much success with my models, and the Kaggle competition resulted in only a few entries scoring above the sample submission, which is a submission that represented randomly guessing (equally weighting probability for the two categories) whether each image in the test set belonged to CE or LAA. The sample submission got a bronze medal (top 100 placement) for this competition on Kaggle at 60th place overall. The difference in score between the best entries and the sample submission is quite small. I think these results simply show how difficult this problem is.

Evaluating the models

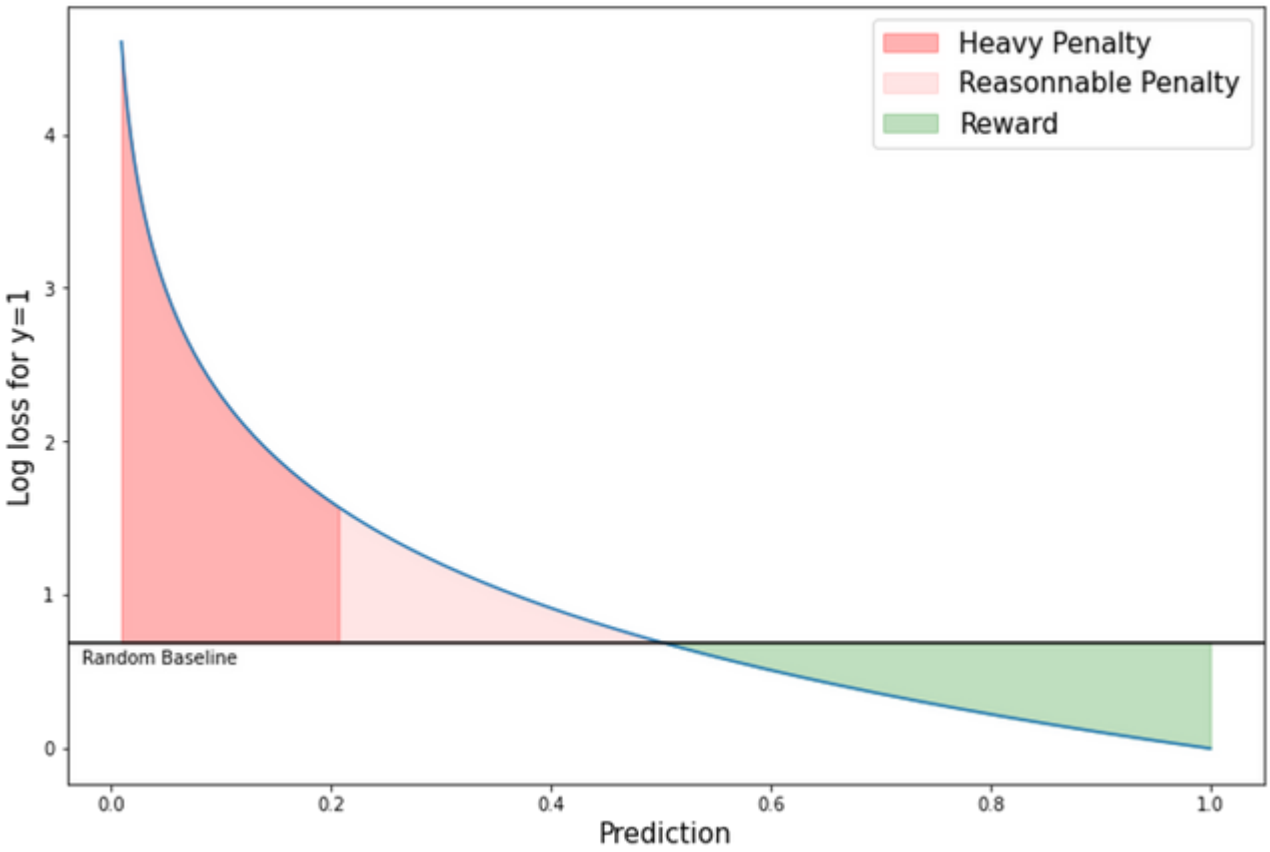
The Kaggle competition scored submissions using a weighted multi-class logarithmic loss function. This function can be seen below.

$$\text{Log Loss} = -\left(\frac{\sum_{i=1}^{M} w_i \cdot \sum_{j=1}^{N_i} \frac{y_{ij}}{N_i} \cdot \ln p_{ij}}{\sum_{i=1}^{M} w_i} \right)$$

This formula has the effect of placing equal importance on each category for the final score. This function is quite similar to the binary cross entropy loss function. I used the binary cross entropy loss for evaluating my models along with rmse, mae, and accuracy. I used all these metrics to help understand what my model was doing, but the most important metric for me to keep track of was binary cross entropy. I used this formula because it can be substituted for the Kaggle competition formula as it essentially carries out the same calculation, and binary cross entropy is easily built in as a loss formula for the kinds of models I created through Tensorflow and Keras. The formula for binary cross entropy is depicted as follows:

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

This metric is useful for this project because it penalizes bad predictions and rewards good predictions- intuitive, right? It's actually not the *most* intuitive metric, and I often refer back to this article to make sure that I am interpreting and using it correctly. Essentially, if you make a prediction with high certainty that an image belongs to one category, but it belongs to another (or vice-versa), then you are penalized heavily. The following image is a pretty good visualization of how predictions are penalized and rewarded.

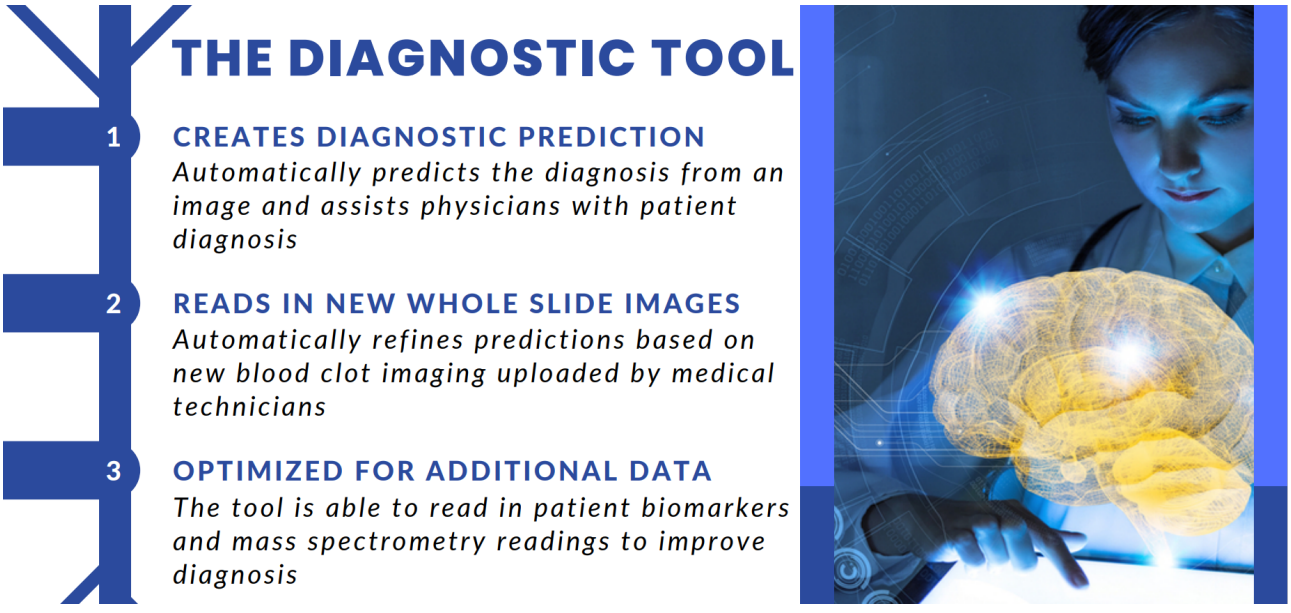


Evaluating the success of baseline vs. best model

In terms of binary cross entropy loss, my baseline model for my Google Colab notebook scored approximately 0.6625. My first Xception model (local notebook) had the lowest binary cross entropy loss score of 0.604. In comparison to the Kaggle competition, the sample submission (randomly guessing) scored 0.693, and the top score was 0.660. I can't gurantee that my losses are comparable to the Kaggle competition, especially because the formulas actually are slightly different, but I do wish I would have contributed an entry to this competition. My highest scoring model (and most of my models at that) score the images very conservatively with predictions around 50% likelihood. It's not that my models are any good, it's just that I'm not getting penalized terribly for bad predictions because my models simply aren't confident in their predictions. Some of the Kaggle users purport making their predictions more conservative by intentionally casting very high or very low percentage predictions to values closer to 0.5 so that they are not penalized by the scoring metric. Considering that the sample submission and the top submission purport such similar results, I am not so convinced that this competition resulted in a model that in itself will be able to complete this task, but hopefully the competition as a whole furthered or redirected solutions for this problem.

Conclusions and further recommendations

The results of my modeling and the Kaggle competition are almost certainly not satisfactory for deployment to real patients. There is certainly still more work to be done to create a model that can carry out this task in an effective way and save lives. My *actionable* recommendations for this project include creating a diagnostic model that is optimized to perform well with more data including more Whole Slide images as well as other types of data such as patient biomarkers. This tool can be summarized below:



Further recommendations:

My further recommendations for tackling this problem more effectively include:

1. Utilize an abundance of computational resources. I'm talking super-computer level computational power. These images are huge. The whole dataset that the Mayo Clinic provided was about 400 GB and loading just a few of these into memory for most machines will crash them. There needs to be computational tools available in order to maintain high resolution of the images and feed in high-resolution images into deep learning technologies.
2. Obtain biomarkers for patients. What other information about a patient could help inform a prediction about whether the stroke that occurred was one type or the other?
3. Obtain other information on the blood clot. For example, a mass spectrometer could be used to obtain the protein content of an individual clot fragment.
4. Normalize how data is collected and collect data from many more sources. There were some problems with the data such as the inclusion of some blurry images, images with tons of whitespace (or blue or grey empty space). It would be more useful if empty space could be removed or not scanned in to avoid expensive computational methods for handling these images or preprocessing them. It would also be useful to have a system in which many hospitals and labs across the country could follow a standard procedure for extracting and imaging these blood clots and sending them to a collective repository of images.
5. Data labeling - it could be extremely beneficial if technicians could label the components within the clot such as red and white blood cells, platelets, fibrin, etc. With these labels, feature extraction would be much easier and could allow for greater possibilities for differentiating between the two categories.
6. Establish trust with models - use methods such as LIME in order to extract features and recognize whether or not a model can be trusted. An easy way to pick up on an untrustworthy model, especially for this task, is if the model is placing importance on empty space or the background of an image.
7. Stepping outside the computational box: Investigate prevantative healthcare measures. Are there techniques that could allow for identification and screening of potential clots before they happen?

Repository Structure

```
├── PDFs
├── images
├── README.md
```

```
    ├── STRIP.ipynb
    ├── og_data_modeling.ipynb
    └── requirements.txt
```

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

- **Jupyter Notebook** 100.0%