CS168 Team 28 Project Report

Topic 25: Automatic Collateral Score for Acute Ischemic Stroke from CTA images

Benjamin Dang
benjamin.d.dang@gmail.com

Manas Kumar
manask6@gmail.com

Sahil Jayaram
sahil.jayaram@gmail.com

1 INTRODUCTION

A stroke is a medical condition in which cells in the brain die from a lack of blood flow. The most common type, the ischemic stroke, is caused by a blood clot, or blockage in the arteries leading to the brain. The danger posed by these clots can be mitigated through collateral circulation that allows blood to circumvent the blockage. Thus, the mapping of these collateral pathways is an important topic in neurology [1].

The Circle of Willis is a circulatory structure in the brain that acts as a primary collateral, delivering blood coming from the heart to several internal carotid arteries. If a person's Circle of Willis allows for "moderate-to-good" collateral circulation, their chances of surviving ischemia from a blockage are improved. More specifically, the time window in which endovascular treatment remains viable lengthens, the potential effectiveness of said treatment increases, and the chance of a post-procedure hemorrhage decreases. As such the specific anatomy of the Circle of Willis, which often deviates from the classic structure due to hypoplasia or underdevelopment, is said to predict one's ischemic stroke survivability [2].

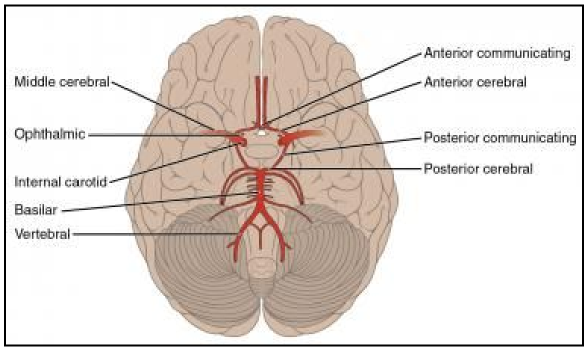

Figure 1. The Circle of Willis (from [3])

Herein lies the problem with evaluating such collateral pathways; because most aspects of the fluid dynamics of blood flow cannot be directly measured, and because the overall quality of flow is not directly proportional to any one of these measures, determining what is "good" or "bad" collateral circulation is a complicated process. Before the common evaluation methods can be understood however, the primary source of information on collateral structure, imaging, must be discussed [2][4].

A Computed Tomography Angiography (CTA) is a medical imaging technique whereby a contrast substance is injected into certain blood vessels and scanned via x-rays to reveal vessel wall structure. CTAs can reveal blockages, tearing, and narrowing of the walls, but only as long as the contrast is sufficiently present. Digital Subtraction Angiography (DSA) is an imaging technique that follows similar principles, in which the end result is obtained from subtracting a pre-contrast image from the rest of the visualizations, and is typically used for bony or otherwise dense areas of the body. Because CTAs and DSAs taken of collateral circulation require contrast to flow through the scanned vessels, and because the substance cannot be injected at all places at once and must flow through all the areas of interest over time, complete visualization occurs in phases. When imaging the Circle of Willis, these phases are peak arterial, peak venous, and late venous, so named for where the contrast is most concentrated (the arteries or veins) [4].

Figure 2. Circle of Willis CTA scan (from [3])

Given the variety and complexity of image sets that can be used to visualize blood vessels, there are several methods for evaluating collateral circulation in the brain. Typically used is a qualitative numbered grading scale, where grade indicates the availability of collateral pathways, whether they reach the ischemic site itself or just the periphery, and their quality of blood flow. Often, these factors are determined by comparing the site in question with the corresponding area in the opposite, asymptomatic hemisphere [5].

The qualitative nature of these scales means that historically, assigning a grade could not be done automatically; a human doctor would have to make the evaluation themselves after inspecting provided images. However, with the advent of neural networks and deep learning in the field of computer science, creating automated programs to make such qualitative distinctions based on visual inputs is becoming more and more feasible. Thus, in this paper we aim to explore and evaluate different possible methods for machine learning-based collateral grading, implement the most feasible one, and maximize its accuracy (based on how well its results agree with evaluations by human neurologists following the same scale) [5].

2 METHODS

The first step in creating our machine learning-based collateral scoring program was to obtain image data to train our model. This data was graciously provided to us by Dr. Sophie Pu from the Beijing Tiantan Hospital, who is currently working out of the University of California Semel Institute for Neuroscience and Human Behaviour. Together, we determined that the most convenient data set to process and work off of would be a database of multi-phase CTA images. These scans were previously graded by Dr. Pu and her team on a zero-to-five mCTA Collateral Score scale, enabling us to work with a fully labeled dataset. Because the scans came from real patients and not theoretical examples, only scores of 3, 4, and 5 are present in the set. At first the database was quite large as the CTA image files were in the Digital Imaging and Communications in Medicine (DICOM) format, showing a continuous brain scan in three dimensions. Training a model off such rich data is certainly possible in theory, but in practice; discrete, two-dimensional pictures are of greater utility. The significant loss of raw data was not of concern, as machine learning algorithms typically excel at making accurate determinations from a breadth of simplified data, as opposed to a small amount of highly detailed examples. Broadly speaking, because of the coarse, qualitative nature of our learning task, general accuracy is far more consequential to success than precision. In keeping with this our first step was to convert the scans to Maximum Intensity Projections (MIP), making it easier for an algorithm to "pick up" the most important features while learning, and then converting the resulting objects to a discrete format. We accomplished this conversion with OsiriX, a DICOM processing application, which we used to separate the scans at 24mm intervals along the z-axis (the direction of the brain stem) with each image containing data from 4mm worth of scanning data [6].

After this step, we had a modified database of two-dimensional .jpg files representing mCTAs of 46 patients, plus reference templates for each of the three scanning phases, adding up to a total of 3,423 images. The most pertinent change to be made from here was cropping out parts of the images that did not include the brain, so that the model did not take into account unnecessary data. Because we wanted the data being inputted into our model to have a consistent format and because the sizes of the brains themselves varied, we could not manually reduce each .jpg to its essentials and end up with differently proportioned images. Instead, we wrote Python code utilizing the OpenCV library to simply crop every 500x500 pixel file down to the 299x299 pixel square in their centers.

At this point our database contained only pertinent visual data, however there were still variations in the scans themselves (e.g. orientation of the brain and its translational location on the image) that we knew had no practical impact on collateral score, but could still be features "picked up" by the algorithm. The solution to this is registering each image to a template image determined by which portion of the brain is represented. There are several options available for implementing this registration process in Python. Given our aforementioned restraints and plans for the model, the Scikit-Image library was a convenient possibility; we knew we would already be using this package later in our code so overall there would be less required compilation and installing. The main issue when testing this approach was that the library's transformation framework is mostly suited to "warping" one image into another, meaning our transformed images often had the shape of their blood vessels altered to more exactly match the template [7]. This would almost definitely change the resulting collateral score. Another library we tested, likely the most robust of our options, was SimpleElastix. This framework is built off of the SimpleITK C++ library, but also utilizes Elastix, a registration toolbox developed specifically for medical imaging [8]. The problem with this method for our purposes was, ironically, the robustness that drew us to it in the first place. SimpleElastix takes up to eight hours to build locally on our ultrabook-style computers (which is necessary for it to make full use of Elastix) because of its myriad of tools and options, a process we had to repeat multiple times due to various small incompatibility issues. The registered images produced by the framework appeared to be quite accurate, however due to time constraints and considerations for performance we searched for alternatives. The third library we investigated was python-register, which is built on top of scipy and numpy. This fact gives python-register the same draw as Scikit-Image, with the added benefit of offering affine transformation-based registration instead of only image warping [9]. Unfortunately, its codebase seems to have been abandoned some time ago, and so we were unable to build and integrate it with our operating system and Python versions. The last option we tested, which we ended up using for our final results, was PyElastix, essentially a lightweight wrapper for Elastix. Briefly put, this framework offers the "best of all worlds" of the previous three: integrating it is simple because it can be built and installed separately from the Elastix executable while still utilizing its core features, and each invocation in our code opens a separate instance of said executable [10]. The latter point became especially important because it allowed us to easily parallelize the registration code with multithreading. After setting the number of scales to apply registration at (or resolution) to two and implementing a triple-threaded approach, the successful affine transformation of all 3,423 images took under six hours to complete, a highly optimal result given our time restraints and low computing power.

The architecture of our sequential model consists of a single-node hidden LSTM layer stacked atop a pre-trained, time-distributed instance of Inception-v3, all implemented in Keras [11]. For each patient, the three image sequences corresponding to different phases are treated as channels, in accordance with the three-channel input format recommended for Inception-v3. Data points are zero-padded with all-black images to achieve a constant sequence length of 30. We used a train-eval split of 30-16 and trained our model for 15 epochs via Adam optimization with a learning rate of .001 and batch size of 6, selecting mean squared error (MSE) as our loss function.

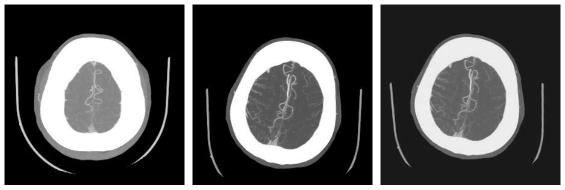

Figure 3. Template, original scan, and registered scan

3 RESULTS

Training took just under seven hours. Our model achieved an MSE of about 0.185 (Fig. 4) and accuracy of 80% (Fig. 5) on the validation set.

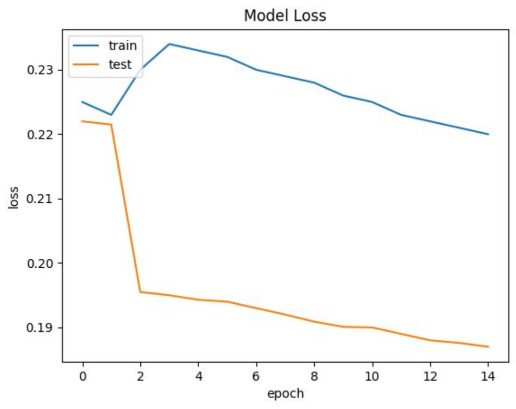

Figure 4. Loss

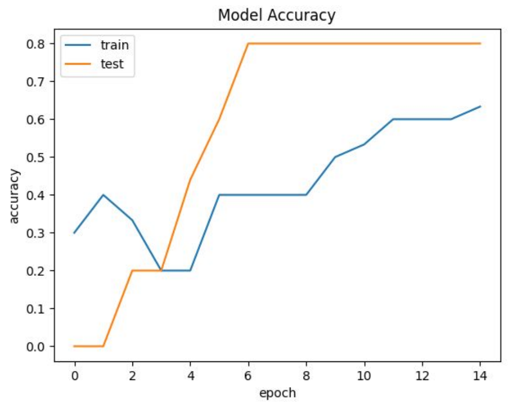

Figure 5. Accuracy

Later inspection of the model yielded the following approximately-diagonal confusion matrix:

$$\begin{bmatrix} 4 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 0 & 5 \end{bmatrix}$$

4 DISCUSSION

Considering that our data exhibits a roughly uniform distribution across the three labels represented and that our classifier achieved 80% accuracy, it is fair to conclude that our model does not suffer from mode collapse. Our confusion matrix confirms this deduction; generally speaking, our algorithm's predictions do not exclude any of the three classes, as none of the rows are disproportionately populated to a considerable degree. Furthermore, the matrix's near-symmetry suggests that our learned model is free of any conspicuous bias—a highly desirable quality in medical image analysis, where both overtreatment and undertreatment are highly unfavorable.

By nature of the Inception-v3 architecture, our model contains an enormous number of parameters. Given the small size of our training set (30 patients), we were concerned that overfitting was inevitable, i.e. that our classifier would continually exhibit poor test set performance while train set performance improved. Conversely, Figures 4 and 5 demonstrate that test set performance constantly improved throughout the training process and may have continued to improve for several epochs if additional iterations of Adam optimization had been performed. Thus, if equipped with a different set of hyperparameters, our algorithm may have the potential to achieve accuracy considerably greater than 80%.

Another possible avenue of improvement is tuning the pre-processing parameters, specifically those for registration. As stated, we used a resolution number of two and got fairly uniform resulting images without alterations to vessel shape, all within a feasible processing time frame thanks to the multithreading optimizations. However, for medical imaging purposes "fairly uniform" may not be sufficient; while the translation aspect of the affine transformations appears to have been applied effectively, the rotation of some images was short of perfect alignment and the scaling was still inconsistent. It is still unclear to us whether that last point is truly a detriment (perhaps maintaining the relative sizes of the patients' blood vessels helps the algorithm analyze collaterals more precisely) but being able to at least test this aspect could have improved the results. In individual registration tests we noticed small improvements to alignment quality when we increased the number of resolutions to eight (albeit at exponential cost to processing time), and we believe the macro-level benefits to scan uniformity would produce notably more accurate results. Finally, we were unable to modify the number of iterations at each resolution level. While the default of 500 seems more than sufficient given our learning bottlenecks in other areas, tuning this parameter could also improve accuracy.

5 CONCLUSION

Our model serves as a good baseline for the problem it is meant to solve; although 80% accuracy is far from ideal, it may be sufficient for some applications of collateral scoring. If we had access to a much larger dataset and more powerful computing machinery, considerably better performance could be achieved by using a larger hidden LSTM layer and feeding our learning algorithm a greater number of training batches. Another factor that may improve performance in future attempts is feature annotation; a model that is "taught" to identify certain relevant brain features, particularly the Circle of Willis, may converge faster than our classifier and outperform it as well.

In order for Computer Vision to assume an assistive role in the realm of radiology, we believe it is necessary that the aforementioned future directions be explored. Nonetheless, our work demonstrates that CV is a viable approach to the task of assigning collateral scores from CTA imagery.

REFERENCES

[1] Anon. 2019. "Stroke". [Online]. Retrieved April 30, 2019 from MayoClinic.org.

[2] Z. Vrselja, H. Brkic, S. Mrdenovic, R. Radic, and G Curic. "Function of circle of Willis," *National Center for Biotechnology Information*. Jan 2014. Retrieved April 30, 2019 from NIH.gov.

[3] Gupta, G. (2018, July 30). Circle of Willis Anatomy. Retrieved May 1, 2019 from https://emedicine.medscape.com/article/1877617-overview

[4] G. Anderson, D. Steinke, K. Petruk, R. Ashforth, and J. Findlay. “Computed tomographic angiography versus digital subtraction angiography for the diagnosis and early treatment of ruptured intracranial aneurysms,” *National Center for Biotechnology Information*. Dec 1999. Retrieved April 30, 2019 from NIH.gov.

[5] L. Liu, J. Ding, X. Leng, Y. Pu, L. Huang, A. Xu, K. Wong, X. Wang, and Y. Wang. “Guidelines for evaluation and management of cerebral collateral circulation in ischaemic stroke,” *Boston Medical Journal,* vol. 3*,* issue 3. Sept 2018. Retrieved April 30, 2019 from BMJ.com.

[6] A. Rosset, L. Spadola, and O. Ratib. “OsiriX: An Open-Source Software for Navigating in Multidimensional DICOM Images,” *Journal of Digital Imaging,* Volume 17, issue 3. June 29, 2004. Retrieved June 8, 2019 from Springer.com

[7] S. Van Der Walt. “Scikit-Image”. [Online]. Feb 8, 2019. Retrieved June 8, 2019 from Scikit-Image.org.

[8] K. Marstal. “SimpleElastix Documentation”. [Online]. March 10, 2015. Retrieved June 8, 2019 from ReadTheDocs.io.

[9] S. Van Der Walt, N Faggian, and R Van Den Dool. “About”. [Online]. Jan 8, 2013. Retrieved on June 8, 2019 from Github.com.

[10] A. Klein. “PyElastix - Python wrapper for the Elastix nonrigid registration toolkit”. [Online]. Jan 13, 2017. Retrieved on June 8, 2019 from Github.com.

[11] Keras: The Python Deep Learning library. Retrieved on June 8, 2019, from https://keras.io/