OCTAI: Smartphone-based Optical Coherence Tomography Image Analysis System

Adrit Rao
Frank S. Greene Jr. Middle School
Palo Alto, CA, USA
adrit.rao@gmail.com

Harvey A. Fishman
FishmanVision
Palo Alto, CA, USA
harvfishman@alumni.stanford.edu

***Abstract*—Identifying diseases in Optical Coherence Tomography (OCT) images using Deep Learning models and methods is emerging as a powerful technique to enhance clinical diagnosis. Identifying macular diseases in the eye at an early stage and preventing misdiagnosis is crucial. The current methods developed for OCT image analysis have not yet been integrated into an accessible form-factor that can be utilized in a real-life scenario by Ophthalmologists. Additionally, current methods do not employ robust multiple metric feedback. This paper proposes a highly accurate smartphone-based Deep Learning system, OCTAI, that allows a user to take an OCT picture and receive real-time feedback through on-device inference. OCTAI analyzes the input OCT image in three different ways: (1) full image analysis, (2) quadrant based analysis, and (3) disease detection based analysis. With these three analysis methods, along with an Ophthalmologist's interpretation, a robust diagnosis can potentially be made. The ultimate goal of OCTAI is to assist Ophthalmologists in making a diagnosis through a digital second opinion and enabling them to cross-check their diagnosis before making a decision based on purely manual analysis of OCT images. OCTAI has the potential to allow Ophthalmologists to improve their diagnosis and may reduce misdiagnosis rates, leading to faster treatment of diseases.**

***Index Terms*—Optical Coherence Tomography, Deep Learning, Macular Disease Detection**

I. Introduction

Optical Coherence Tomography (OCT) is a non-invasive imaging technique capable of scanning the structure of the retina [1]. It is widely utilized in Ophthalmology clinics to identify diseases in the eye [2]. Three common macular disease processes in the eye are Choroidal Neovascularization (CNV), Diabetic Macular Edema (DME), and Drusen [3]. These pathologies can lead to significant vision loss [3]. CNV involves the abnormal growth of new blood vessels within the retina [4]. DME is fluid accumulation within the macula [5]. Drusen are sub-retinal deposits that are risk factors for the development of CNV. Pathology within the retina is typically easy to identify by Retinal Specialists using OCT. However, these pathologies can be subtle or can co-exist with each other. Because of this, Ophthalmologists can have difficulty in diagnosis through manual assessment of the OCT image.

Multiple Deep Learning and Machine Learning (ML) based techniques and models have been introduced to classify different types of diseases in OCT images [6]. These models and methods may demonstrate high accuracy rates but do not lend themselves to a usable form-factor (analysis tool) that can be utilized by Ophthalmologists in a clinical setting. Additionally, most of these methods provide only single metrics. Utilizing only a single metric can lead to misdiagnosis due to the possibility of the model providing a wrong result or making a mistake in predicting the disease.

Given these challenges, there is a clear need for an accessible and robust analysis system for OCT images. This paper presents the use of a custom Deep Learning system (OCTAI) capable of integration into a smartphone application. OCTAI features multiple metric OCT image analysis using a custom Deep Learning pipeline and leads to a more robust diagnostic feedback than current methods of OCT image analysis. This type of infrastructure can be used by Ophthalmologists to derive a useful second opinion while assessing OCT images manually within a clinic through a smartphone.

II. Methods

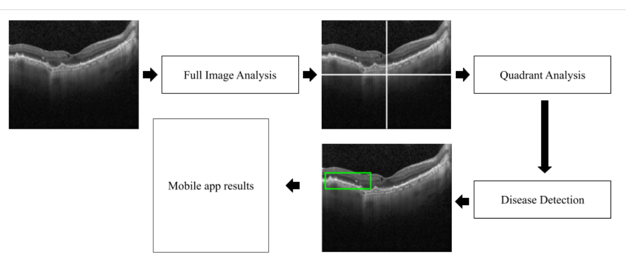

Fig. 1: OCTAI System Architecture

OCTAI is a smartphone-based end-to-end OCT image analysis system that uses multi-step Deep Learning. The mobile app allows the user to take a screenshot of the OCT machine display. The first step of OCTAI consists of a Convolutional Neural Network (CNN) that analyzes the OCT image as a whole. The second step splits the image into 4 quadrants and analyzes each quadrant using the same CNN. In the third step, OCTAI takes the whole image and sends it to a custom disease detection model capable of localizing disease within the OCT image. In the final step, the mobile app displays the results on the screen for the Ophthalmologist to interpret. Shown in Figure 1 is the complete OCTAI architecture. The goal of utilizing multiple Deep Learning techniques is to have multiple analyzable metrics.

The following order and methodology has been used to develop the full OCTAI system:

1) Data Collection
2) Data Preparation
3) Model Construction

A. Data Collection

The images used for training the model are from the "Large Dataset of Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images" dataset [7]. The dataset consists of an OCT image sub-directory which contains 108,309 images spanning the classes of CNV, DME, DRUSEN and NORMAL. Figure 2 shows the example images from the dataset in a 3x3 grid with their corresponding macular diseases.

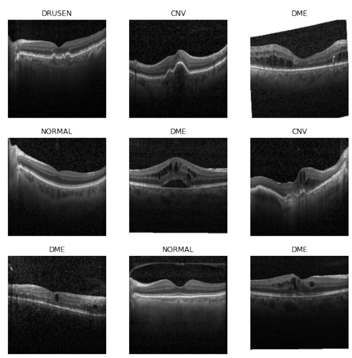

Fig. 2: Example images from dataset

Shown in Figure 3, is a bar chart that depicts the dataset distribution. The CNV class has 37,205 images, DME has 11,348 images, DRUSEN contains 8,616 images and NORMAL has 51,140 images. The percent distribution is as follows: CNV is 34.4% of the dataset, DME is 10.5%, DRUSEN is 8% and NORMAL is 47.2%. The dataset contained a substantially greater amount of NORMAL and CNV images.

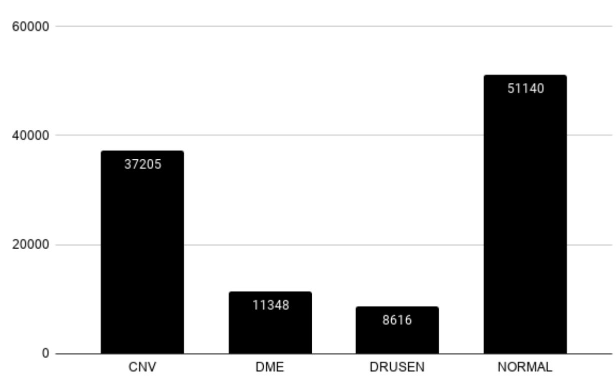

Fig. 3: Dataset distribution bar chart

B. Data Preparation

The dataset was split into an 80% training and 20% testing standard split. 99,644 files were used for training and 8,665 for validation/testing. The dataset was preprocessed/normalized by adding a rescaling function/layer transforming the RGB channels from [0, 255] to a more ideal binary Neural Network input of between [0, 1].

After normalization, data augmentations were added. This factor was added to not only reduce overfitting during model training but also to enable the model to become resilient to orientation, zoom and contrast. All augmentations were added as a layer at the top of the model architecture along with normalization. The following specific data augmentations were added.

- Random flips (y axis)
- Random rotation
- Random zoom
- Random contrast

Shown in Figure 4 is a single image multiplied nine times with the different data augmentations added randomly.

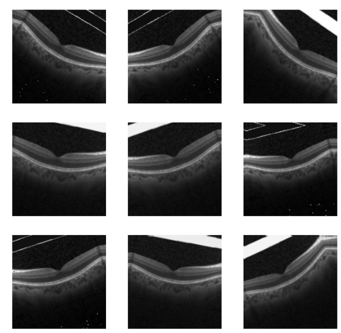

Fig. 4: Example of image after augmentation

Since OCTAI is a system integrated into a smartphone, users have the ability to take a screenshot of the OCT screen. These augmentations help prevent wrong predictions due to factors such as screen brightness and orientation.

C. Model Construction

All model development was done in the Google Colab Pro [8] Integrated Development Environment (IDE) with the NVIDIA Tesla P100 GPU [9]. Tensorflow [10] and Keras [11] were used to sequentially construct and train the models in the Python 3.7 programming language [12]. Throughout the model development process, multiple separate models with different architectures were developed with the goal of using the model that revealed the highest accuracy rate. The OCTAI system consists of two Deep Learning models. The first model is used for the first two steps of the system that include whole image analysis as well as quadrant based analysis. The second model is used for the disease detection analysis.

CNNs, a class of Deep Neural Networks (DNNs) have revealed high accuracy rates when analyzing visual imagery. The first model in OCTAI capable of identifying diseases in the whole image was developed using a custom CNN architecture. The first component of the CNN consists of the input layer with width and height parameters. After the input layer, the rescaling layer was added for the binary conversion. Subsequently, 4 sets of 2D Convolutional Layers and Max Pooling Layers were added for feature extraction as well as downsampling. After this, a Dropout layer of 20% was added to prevent overfitting during training. The last part of the CNN is the fully connected layer which consists of the Flatten Layer that converts the data into a one-dimensional representation before sending it to two Dense Layers for probability class prediction. For activation functions, the Rectified Linear Unit (ReLU) was utilized. The CNN model was trained across 10 epochs. The model compilation was done with the Adam optimizer [13]. Shown in Figure 5 is a diagram of the custom CNN architecture used in OCTAI.

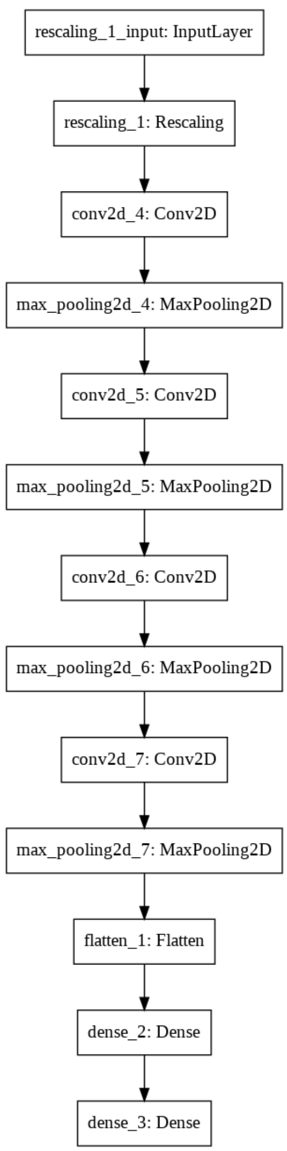

Fig. 5: CNN Architecture

The main goal for whole image analysis and using a CNN is to extract minor features that may not be clearly visible or recognizable by an Ophthalmologist. The model is learning based on a large number of features present throughout the whole image and is taking into account features down to the single-pixel level which may be difficult for an Ophthalmologist to recognize as well as analyze.

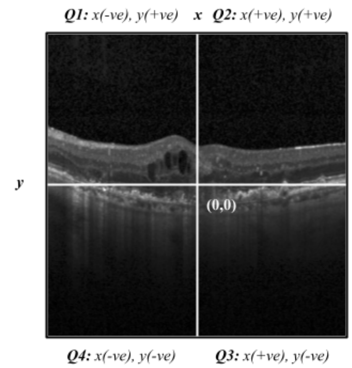

Fig. 6: Simple quadrant split

For the quadrant based analysis, the OCT input image is split into 4 quadrants as shown in Figure 6 using the simple X and Y values. Each quadrant is then fed into the same CNN model (Figure 5). When analyzing an OCT image, it is probable that there is more than one disease present. The goal for quadrant segmentation and analysis is to give the user another metric to interpret that can possibly help with the localization of the disease. Additionally, the model utilizes minor features as described above in making a prediction.

The second model that OCTAI uses is for disease detection. For the development of this model, the You Only Look Once (YOLO) [14] architecture was leveraged. YOLO is an architecture that allows for high-speed object detection in comparison to other methods. Before the YOLO model architecture was released, the R-CNN was used for object detection tasks. This model requires a high amount of computational power due to its underlying methodology.

At a high level, YOLO takes an image (SxS), applies bounding boxes (B), before using a CNN to analyze all of the bounding boxes and subtracting low confidence predictions based on a threshold. Shown in Figure 7 is the YOLO CNN architecture.

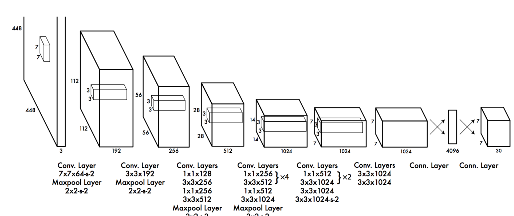

Fig. 7: YOLO Architecture [14]

Because the final goal is integration of the model in a mobile application, YOLOv4 tiny [15] was used for model development. This version of YOLO requires less computational power and thus is less in size and is ideal for on-device inference and deployment on multiple platforms.

For training this model, the dataset was consolidated into 1000 images per class. Because YOLO is an object detection model, it requires an annotated dataset. The dataset was annotated using the Roboflow Web App [16] that allows for annotations as well as storage of datasets in the Cloud. Figure 8 shows example images after annotation of the diseased portions.

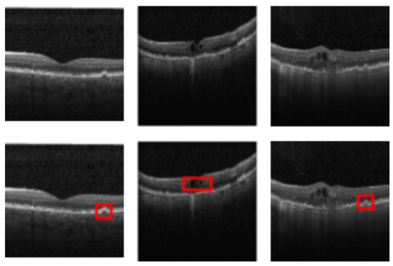

Fig. 8: Annotation examples for diseases

After the annotation process, text files containing the corresponding bounding box coordinates were generated for each image. This is the format that YOLO uses for model training. The model was then trained across 100 epochs using the Darknet framework [17] that allows for training of YOLO models using the Tensorflow and Keras libraries.

III. Results

For evaluating the models accuracy rates, both models were tested on their corresponding testing splits branched from the original full OCT image dataset.

In the 10th training epoch, the first CNN model received an accuracy of 98.71% on the training data and a validation accuracy of 95.13% on the testing dataset. It also received an F1 score of 92.64%. Shown in Figure 9 are two graphs that represent the training as well as validation accuracy and loss achieved over the 10 epochs.

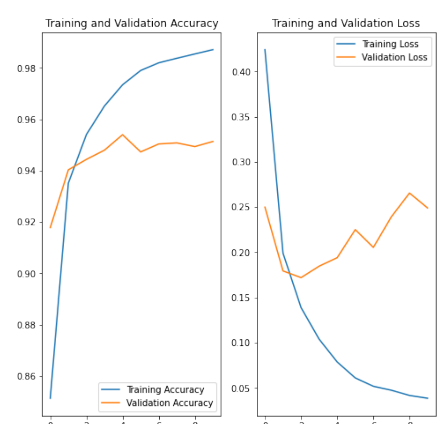

Fig. 9: CNN training and validation accuracy and loss

Equations 1 and 2 depict the formulas used for accuracy and F1 score calculations. These metrics are calculated for each of the training epochs. TP is the number of True Positives, TN is the number of True Negatives, FP is the number of False Positives and FN is the number of False Negatives.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{2}$$

The disease detection model performed with a 100% accuracy while tested on 30 images and successfully detected all the diseases present within the images. Example images are shown in Figure 10 from the testing dataset after inference was applied. All three images except the bottom right image (NORMAL) had diseases present. The model successfully did not detect disease in the normal image implying that no disease was present.

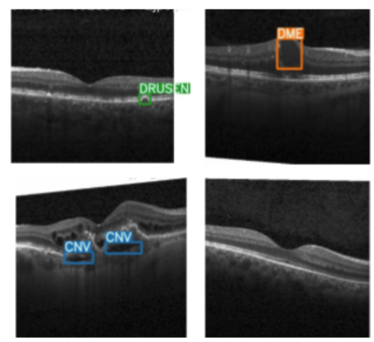

Fig. 10: Example images after disease detection

Both models in OCTAI were converted from their original format into an on-device Tensorflow Lite format. With this format, OCTAI has the ability to be integrated into iOS and Android mobile platforms. For this project, demonstration of integration was done within an iOS iPhone application developed with the Swift 5 programming language within the Xcode IDE. Shown below in Figure 11 is a screenshot taken of the iOS application. The app allows the user to take a picture or choose a photo from the photo library.

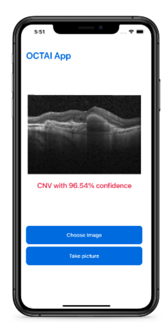

Fig. 11: OCTAI mobile app

IV. Conclusion

This paper demonstrates a novel and highly accurate Deep Learning system, OCTAI, that has the capability to run multi-step OCT image analysis within a smartphone application. With this proposed system, Ophthalmologists can have the ability to use a smartphone to derive a three-step predictive result with which they can interpret and assist them in making a final robust diagnosis of pathology. This system is agnostic to any commercial OCT device thus making this platform potentially a widespread and powerful tool for Ophthalmologists.

References

[1] D. Huang, E. A. Swanson, C. P. Lin, J. S. Schuman, W. G. Stinson, W. Chang, M. R. Hee, T. Flotte, K. Gregory, C. A. Puliafito *et al.*, "Optical coherence tomography," *science*, vol. 254, no. 5035, pp. 1178–1181, 1991.

[2] D. Turbert, "What is optical coherence tomography?" Apr 2020. [Online]. Available: https://www.aao.org/eye-health/treatments/what-is-optical-coherence-tomography

[3] M. A. Zarbin, "Age-related macular degeneration: Review of pathogenesis - m. a. zarbin, 1998." [Online]. Available: https://journals.sagepub.com/doi/abs/10.1177/112067219800800401

[4] A. M. J. A. A. CD;, "Age-related macular degeneration therapy: a review." [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/32205470/

[5] D. A. P. S;, "Diabetic macular edema: Pathophysiology and novel therapeutic targets." [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/25935789/

[6] R. Asaoka, H. Murata, K. Hirasawa, Y. Fujino, M. Matsuura, A. Miki, T. Kanamoto, Y. Ikeda, K. Mori, A. Iwase *et al.*, "Using deep learning and transfer learning to accurately diagnose early-onset glaucoma from macular optical coherence tomography images," *American journal of ophthalmology*, vol. 198, pp. 136–145, 2019.

[7] D. Kermany, K. Zhang, and M. Goldbaum, "Large dataset of labeled optical coherence tomography (oct) and chest x-ray images. mendeley data, v3," 2018.

[8] E. Bisong, "Google colaboratory," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*. Springer, 2019, pp. 59–64.

[9] "Nvidia tesla p100: The most advanced data center accelerator." [Online]. Available: https://www.nvidia.com/en-us/data-center/tesla-p100/

[10] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 2016, pp. 265–283. [Online]. Available: https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf

[11] K. Team, "Simple. flexible. powerful." [Online]. Available: https://keras.io/

[12] "Welcome to python.org." [Online]. Available: https://www.python.org/

[13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[15] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[16] "Everything you need to start building computer vision into your applications." [Online]. Available: https://roboflow.com/

[17] J. Redmon. [Online]. Available: https://pjreddie.com/darknet/