Final Group Project Report: Diabetic Retinopathy

Fernando Zambrano, M.S.

Gaberial Campese, M.S.

Dr. Amir Jafari, Ph.D.

The George Washington University

Introduction

Diabetic retinopathy is the leading cause of blindness and it affects nearly 100 million individuals, 93 million to be exact. Identifying signs of this condition provides value to both medical professionals and victims of diabetic retinopathy. With this research, medical professionals can better identify signs of diabetic retinopathy in their patients through the use of image classification in neural networks. This study uses Keras to create a pretrained network architecture known as a ResNet50 Convolutional Neural Network (CNN). Diabetic retinopathy images are then fed into this network to ultimately train the network and create a model to recognize different magnitudes of diabetic retinopathy.

Dataset Description

The Diabetic Retinopathy Dataset was obtained from Kaggle and it was formerly used in a 2015 Kaggle Competition. The dataset is composed of 35,126 images and CSV file, all totaling 82GB. The images are of retina scans from both sides, left and right, of the eyes of patients. There are 5 classes, detailed in the CSV, representing the level of diabetic retinopathy for that patient ranging from 0 to 4. The classes are as follows: 0 signifies no diabetic retinopathy, 1 is for Mild, 2 for Moderate, 3 for Severe, and finally 4 is Proliferative diabetic retinopathy. Due to the overwhelming size of the dataset, this research effort opts to use a subset of the data totaling 7,920 images with relative distributions. Moreover, the use of the entire dataset was attempted, only to run into memory errors on the Google GPU node.

Deep Learning Network & Training Algorithm

As mentioned above, this project utilizes a pretrained Convolutional Neural Network in Keras known as ResNet50 as the base layer. ResNet50 is the winner of the 2015 ImageNet Challenge (Dwivedi, 2019). It features a 50 layer CNN architecture and it is extremely powerful in image

classification. A fully connected top layer was additionally created with regularization and the use of Dropout(0.2). The sigmoid, relu, and softmax activation functions each were tested in our model. Also, 3 different optimizers including adam, rmsprop, and SGD were compared in the deep learning network. To freeze layers, layer.trainable=False was experimented with to freeze the weights and reduce trainable parameters. The model was executed with and without this aspect.

Experimental Setup

The creation of the model is conducted by utilizing Python and Keras in PyCharm. A virtual environment is also created in a Google Cloud Platform Tesla K80 GPU instance to provide processing power for the data and deep learning architecture. The images sizes are set to 256 and are read in using the cv2 library. The CSV is read in and its labels are one hot encoded. The data is eventually converted to NumPy arrays and split into test/validation sets. ImageDataGenerator is used to augment our images by performing shifts, shears, flips, and rotations on the images. The data is then fed into the CNN architecture discussed in the previous section of this report. Accuracy, Cohen Kappa, and F1 scores are then measured.

Results

Our model produced good accuracy scores for both training and validation accuracy scores. For training, we were able to generate an accuracy score at 74.31% with validation accuracy being 82.88%. The accuracy score was consistent among just about every epoch. Cohen Kappa was also evaluated and found to be 0.0 for our model. F1 score was also measured and produced a score of 0.1812. Also, the freezing of layers methodology also had no effect on our model. Out of the 3 optimizers used, rmsprop was found to be the best optimizer for our model in terms of accuracy and F1 scores.

Summary & Conclusions

Although our model was able to produce great accuracy scores, we recognize that this does not always signify that the accuracy solely determines the overall success of the model. This sense of confidence in the success of a model is extremely important especially in the medical field. Those in the medical space must act with certainty being that human life is at stake. Because of the fact that we did not want to rest our confidence solely on the accuracy of our model, Cohen Kappa and F1 scores were calculated. The results of these scores are detailed above and they are rather unfavorable. The Cohen Kappa score essentially equates to "chance" in this case with the score of 0.0. Moreover, the fact that the accuracy is consistent among every epoch is another issue. It is almost as if our model is not training appropriately and this is to be further investigated. In future research, we would like to attempt to use higher GPU capabilities to leverage the entire dataset. We would also intend on focusing more on data augmentation, preprocessing, and potentially running networks in ensemble to produce more favorable results. Overall, our model does a good job at producing high accuracy scores, yet they are not what they appear to be at first glance. This research attempted to mitigate the issues previously discussed and we are better set up for success in the future because we have a better understanding in terms of direction moving forward.

References

Chollet, François. (2018). Deep learning with Python. Shelter Island, NY: Manning Publications Co.

Diabetic Retinopathy Detection (dataset). (2015). Retrieved from https://www.kaggle.com/c/diabetic-retinopathy-detection/.

Dwivedi, P. (2019, January 04). Understanding and Coding a ResNet in Keras. Retrieved from https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33.

Appendix

GitHub Repositories (code):

https://github.com/gcampese/FinalProject-Group7

https://github.com/nanditobandito/FinalProject-Group7

Our subset (hosted in Google Bucket):

wget https://storage.googleapis.com/finalproject-group7-ml2/train_subset.zip