Deep CNNs for Diabetic Retinopathy Detection

Alex Tamkin
Stanford University
atamkin@stanford.edu

Iain Usiri
Stanford University
iusiri@stanford.edu

Chala Fufa
Stanford University
cfufa@stanford.edu

1 Introduction

Diabetic retinopathy is an eye disease caused by diabetes that can lead to loss of vision or even complete blindness. Diabetic retinopathy accounts for 12% of all new cases of blindness in the United States, and is the leading cause of blindness for people aged 20 to 64 years.7 If caught early enough, progression to vision impairment can be slowed if not altogether stopped, however, this is often difficult because symptoms may appear too late to provide effective treatment. Diabetic retinopathy (DR) has been estimated to affect about 93 million people globally, though only half are aware of it. There are four main stages of Diabetic Retinopathy; in its most advanced stage, abnormal blood vessels propagate on the surface of the retina, which can lead to scarring and cell loss in the retina.

Currently, diagnosing DR is an slow and arduous process that requires trained doctors to analyze color photographs of retinas. They then classify the level of deterioration the patient's eye has experienced into one of four categories. While this process is effective, it is very slow. It takes about 2 days to get back results and after that time it may be harder to reach the patient. Furthermore, in areas where access to trained clinicians or suitable equipment is limited, individuals are left without any support. As the number of people with diabetes increases this system will become even more insufficient.

We propose a model for classifying retina images as having DR using convolutional neural networks trained with transfer learning. The input to the model is a pre-processed 256px x 256px retina image, and the output is a class label indicating whether or not the retina has DR.

2 Related Work

Historically, image analysis and classification has mostly focused on "low level image analysis tasks" like feature extraction and basic color normalization coupled with classical machine learning classification models like regression, SVMs and random forests. These algorithms would usually manage a small set of manually identified features ("in the order of tens") which tend to limit their classification ability. Progress was made by the introduction of automated extraction of high dimensional sets of image features (on order of thousands). Dimensionality reduction techniques (e.g sparse regression) were then used to supply the construction of simple linear classifiers for the data.[1]

More recently, leveraging Convolutional Neural Networks to perform image classification has become a very popular technique, particularly in the biomedical field. Their invariance to noise, orientation, image quality, lighting and intra-class variation offers a critical robustness that makes them especially suitable for biomedicine. Early work from the research group of Jurgen Schmidhuber won the ICPR best paper awards in 2012 and 2013 by focusing on algorithmic work for mitotic figure detection using neural networks [2]. Furthermore, neural networks have been used in identifying metastic breast cancer,[1] brain lesion segmentation[4], cancer diagnosis [5] and other areas where humans have previously been forced to manually classify a test result.

A joint study [1] by Harvard Medical School and MIT worked on furthering techniques in histopathological image analysis for metastic breast cancer. Their approach obtained a near human-level classification performance using a 27-layer neural network architecture. Similarly, researchers from Cambridge University, Imperial College London, and others, [6] intruduced an 11-layer 3D convolu-

tional neural network to offer a more computationally efficient approach to brain lesion segmentation. One of the challenges they managed to overcome is the "vanishing gradient problem," where the signal of whether the prediction was correct or not becomes greatly attenuated as it propagates through the layers of the neural network. Batch Normalization, coupled with other heuristics, was leveraged to preserve this signal and offered a significant performance improvement.

These are still early days for neural network applications, and methods are still being developed in areas as diverse as systems architecture and data augmentation to improve these models' predictive power.

3 Dataset and Preprocessing

We use a dataset of retina images from a recent Kaggle competition [1].These are a set of high resolution retina images taken in a variety of conditions, including different cameras, colors, lighting and orientations. For each person we have an image of their left and right eye, along with a DR classification diagnosed by a clinician. There is considerable noise and variation in the data set due to these differing conditions. We describe our data preprocessing in the following paragraphs, and the number of samples we use for each data partition in the experiments section.

A key part of setting up our pipeline was preprocessing our data, the color retina images. Despite coming from Kaggle, which has a reputation for having clean data, these images required a hefty amount of preprocessing before we could use them in our neural network. The provided retina images were of different dimensions and resolutions, were taken by different cameras, were in different orientations, and were sometimes not even aligned or cropped similarly. The size of the dataset was also more than 38 gigabytes (35,126 images), which was intractable to handle with our computational resources.

To start, we had to transform the images in such a way that it would be feasible for a neural network or any learning algorithm to converge in a reasonable time. This consisted of resizing each image 256px by 256px. While this helped in making the necessary computation less intensive, it did not help with the fact that the lighting, orientation, and alignment were not similar across images.

To reduce this variation in the images, as has been the standard in other high-performing papers according to Andrej Karpathy, each image was rescaled to have the same radius (the eyeball) and each pixel had its color subtracted by the local average, mapping the average to 50% gray [2]. A local average was used to account for the varying lighting conditions of the images, given that these images are taken by illuminating the retina, and an ill-aligned lighting source creates a gradient of illumination across the image that a local average can largely remove. The edges of the images were also clipped since there is a great variation on the boundaries or edges of the images.

We then trained the convolutional neural networks with these preprocessed images.

4 Model

Because we have a large and well defined data set to learn from, this is a supervised learning problem and more specifically a classification problem. Our objective is to create accurate binary classification on our data; we wish to classify a retina image as having or not having various stages of diabetic retinopathy. There are 5 stages (0-4) and our first challenge was deciding how best to bucket the stages. We decided to use a 0 vs 1-4 bucketing and a 0-1 vs 2-4 bucketing, as described in the next section. We used convolutional neural networks, or CNNs, which are currently state of the art for image classification, as our learning model.

For some context, neural networks are a computational approach modeled off of the neural structure of the mind. They are structured as a large collection of "neural units" which are clustered in layers. Each neural unit is connected to others and can contain an activation function that combines its input values together. Lower layers pass a signal up to higher layers, with each layer learning higher-level representations of the data. Images are fed into the model during training, and the parameters are tuned to bring the output closer to the desired output.

[1] https://www.kaggle.com/c/diabetic-retinopathy-detection

[2] cs231n.github.io/neural-networks-2

[9] We used transfer learning to help simplify our learning time. Transfer learning is when knowledge gained from learning to complete one task is leveraged to complete another task of similar structure. For example, if you build a model to learn to classify cars it follows that you can use parts of that model to classify trucks. In our case, though the domains are different, we still had good results with transfer learning. Transfer learning is shown to be good because you gain better performance with your learning model very quickly, much earlier than when having to train a model from scratch. This means that you can gain powerful predictive power with much fewer training examples in cases where training data is hard to get. You also can re-use lower-level features across domains.

[13] For our transfer learning architecture, we used the Inception V3 model. We used the ImageNet weights and added 2 dense layers on top of the model which we trained. We then inserted dropout layers between the dense models to reduce overfitting [11]

We then just trained those two dense layers using the Adam optimizer [12] The Adam optimizer works as follows:

$$m(w,t) = \gamma_1 m(w,t-1) + (1-\gamma_1)\nabla Q_i(w)$$
$$v(w,t) = \gamma_2 v(w,t-1) + (1-\gamma_2)(\nabla Q_i(w))^2 v(w,t) = \gamma_2 v(w,t-1) + (1-\gamma_2)(\nabla Q_i(w))^2$$
$$\hat{m}(w,t) = \frac{1}{1-\gamma_1^t} m(w,t) \hat{m}(w,t) = \frac{1}{1-\gamma_1^t} m(w,t)$$
$$\hat{v}(w,t) = \frac{1}{1-\gamma_2^t} v(w,t) \hat{v}(w,t) = \frac{1}{1-\gamma_2^t} v(w,t)$$
$$w(t+1) = w(t) - \frac{\eta}{\sqrt{\hat{v}(w,t)} + \epsilon} \hat{m}(w,t) w(t+1) = w(t) - \frac{\eta}{\sqrt{\hat{v}(w,t)} + \epsilon} \hat{m}(w,t)$$

(Wikipedia: Stochastic gradient descent.)

Finally, we trained the top two blocks of the V3 architecture using Adam as well.

5 Experiments, Results, and Discussion

We considered a variety of different tasks, architectures, data augmentations and preprocessing strategies, to see the conditions under which different types of tasks could fare well. We built and evaluated each model in Keras[3], using a TensorFlow[4] GPU backend, running each model until convergence, defined as no improvement in validation accuracy for five consecutive epochs. An epoch was set to 2000 training examples, generated via our preprocessing system above, and the batch size was kept at the default of 32. We optimized our CNNs using Adam, using the original parameters in the paper that proposed it. The cost of computational resources and time constraints precluded a systematic hyperparameter search.

Since our number of positive and negative examples are equal, accuracy, $\frac{\text{\# Correct}}{\text{\# Of Examples}}$, is a good measure of our model's performance, and we report the respective accuracy of each model in Tables 1 and 2.

For our best performing models of each task, We also report sensitivity, $\frac{\text{\# True Positives}}{\text{\# Positives}}$ and specificity, $\frac{\text{\# True Negatives}}{\text{\# Negatives}}$. These two statistics are commonly used in medical diagnosis to evaluate tests. [3] A high sensitivity (also known as "recall") is important to identify all subjects who have a given condition, and a high specificity (also known as "true negative rate") is important to avoid false positives and run unnecessary procedures on healthy patients. We also include a plot of the Receiver Operating Characteristic (ROC), which is a curve that plots specificity against (1-sensitivity) as the discriminating threshold of a model is varied. The closer the area under this curve is to 1, the better the performance of the model.

As a first, simpler task, we attempted to differentiate between healthy eyes (graded in our dataset as a 0) and eyes with the worst form of diabetic retinopathy (graded in our dataset with a 4). We extracted a set of retina images from our dataset that matched these criteria (totaling 1665 images)

[3] https://keras.io

[4] https://www.tensorflow.org

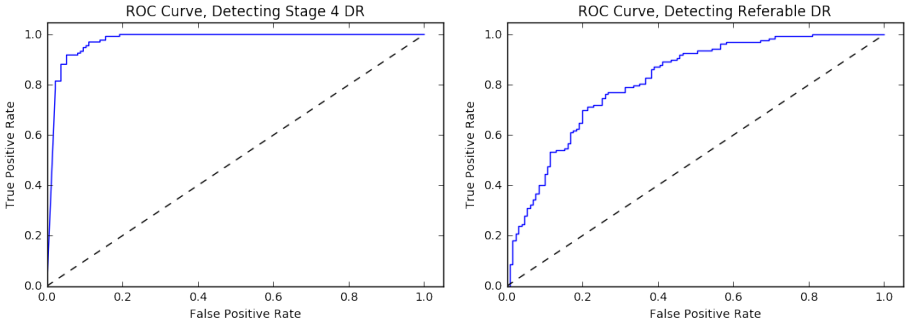
and used 80% of them as a training set, with the remaining 20% split evenly between validation sets and test sets. Statistics measuring performance on the validation sets were used for tuning hyperparameters and evaluating model progress. Statistics measuring performance on the test sets, which are the statistics included in this report, were only computed after all tuning of the model had been completed.

We initially evaluated two models for this simpler task, both using the Inception V3 model with ImageNet pre-trained weights and two additional fully-connected layers. The first model involved freezing the base model and training just the final two layers, and the second involved additionally training the top two blocks (layers 172 to 217) of the base model.

A large gap between the training and validation accuracy of the latter model ($\approx 10\%$) indicated that the model was overfitting to the training data. To compensate for this, we added data augmentation consisting of random vertical and horizontal reflections, as well as Gaussian noise, random crops (up to 1.2x magnification) of the full images, and random shear. We than re-ran the two models using the augmented datasets, and found that the worst-performing model trained on augmented data performed better than the best-performing model trained on non-augmented data. The results of all four tests are listed in Table 1.

To evaluate our model on a more useful task, we attempted to discriminate between eyes that had referable DR[8] and eyes that did not. Referable DR is defined as moderate or worse DR (which excludes mild DR), and corresponds to a grading of 2-4 in our dataset. Thus, in this task we divided our dataset into two classes, one with images graded 0 or 1, and one with images graded 2, 3, or 4. We report the performance of these models, trained with data augmentation, in Table 2.

The Receiver Operating Characteristics of the best performing models for each task (training top two blocks, with data augmentation) are located in Figure 1, along with their associated AUCs (Area Under Curve).



(a) AUC = 0.9764 (b) AUC = 0.8189

Figure 1: Receiver Operating Characteristic Curves for (a) Detecting Stage 4 DR, and (b) Detecting Referable DR

These results show enormous promise for diagnosing and detecting diabetic retinopathy. Family doctors have been shown to have a specificity of 87% and a sensitivity of 54% in detecting referable diabetic retinopathy [3] our model achieves both higher specificity and sensitivity. This is highly encouraging, because it shows that our model could be used to assist family doctors and general practitioners in diagnosing this disease, helping to stop the proliferation of diabetic retinopathy early and saving people's vision. Moreover, these results have promise for nations and populations where medical knowledge, professionals, and care are less readily available through the distribution of standalone cameras packaged with DR-detection software. Since DR is such a large cause of blindness across the world, these findings have the potential to save millions of people's eyesight each year.

Table 1: Accuracy of different models for detecting stage 4 DR

Model	Data Augmentation	Accuracy
Last Two Layers	No	0.8517
Last Two Layers + Top Two Inception V3 Blocks	No	0.8733
Last Two Layers	Yes	0.9037
Last Two Layers + Top Two Inception V3 Blocks	Yes	**0.9259**

Table 2: Accuracy of different models for detecting referable DR

Model	Accuracy
Last Two Layers	0.7185
Last Two Layers + Top Two Inception V3 Blocks	**0.7296**

6 Conclusions and Future Work

We present accurate deep convolutional network models for detecting both stage 4 diabetic retinopathy and referable diabetic retinopathy. Our model for detecting referable diabetic retinopathy yields better sensitivity and specificity than family doctors.

Across all models, using data augmentation improved the resulting accuracy of the model. This is likely due to its ability to reduce overfitting by artificially increasing the size of the training set, given that smaller test sets are easier to overfit. Additionally, training the top two blocks of the Inception V3 network in addition to the final two fully-connected layers improved performance relative to just training the final two layers. This is likely because the top blocks in the base model represent higher-order features that are more specialized to the ImageNet task the model was originally trained for. So, re-training these layers likely allows us develop higher-level features specific to this task, increasing performance.

We obtained our best results using the Inception V3 model and ImageNet pre-trained weights with two additional fully-connected layers, training the two additional layers and the top two blocks of the the base model. Given the degree to which data augmentation improved our results, it might be promising to explore other means of augmentation. One option to consider would be adding small random rotations to our training images, in addition to the flipping, noise, zoom, and shear transformations we applied.

One of the challenges of the project was the limitations of the computational resources we had available. Deep CNN models take a long time to train, and the resources necessary to train them become expensive over time. There are multiple network architectures and preprocessing/data augmentation techniques that could have been explored had there been more computational resources available. Because of these time and computational constraints, there was a tradeoff between training time and accuracy. Training time could have been sped up by shrinking the image size, but cursory experiments indicated that this would result in markedly lower accuracy. Furthermore, higher accuracy could have been achieved through randomized search on the hyperparameter space had these resources been available.

The next logical step for this project would be attempting to determine the presence of diabetic retinopathy at any stage. I.e. this would mean differentiating between Stages 0 and Stages 1,2,3 and 4. Domain knowledge could also help in developing better preprocessing methods that would allow the model to more easily achieve a higher accuracy. Even further down the road, the holy grail of DR-detection would be to develop a multiclass classification model to not only determine whether an eye has diabetic retinopathy, but what specific stage it is in as well.

Table 3: Sensitivity and Specificity Per Task of Best Performing Models

Model	Sensitivity	Specificity
Detecting Stage 4 DR	0.9333	0.9111
Detecting Referable DR	0.8920	0.5878

References

[1] Dayong Wang Aditya Khosla? Rishab Gargeya Humayun Irshad Andrew H Beck Beth Israel Deaconess Medical Center, Harvard Medical School CSAIL, Massachusetts Institute of Technology,
Deep Learning for Identifying Metastatic Breast Cancer. `https://arxiv.org/pdf/1606.05718v1.pdf`.

[2] D. C. Cires¸an, A. Giusti, L. M. Gambardella, and J. Schmidhuber,
Mitosis detection in breast cancer histology images with deep neural networks. In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013, pages 411–418. `https://arxiv.org/pdf/1603.05959v2.pdf`.

[3] Gill, James M., et al. "Accuracy of screening for diabetic retinopathy by family physicians." The Annals of Family Medicine 2.3 (2004): 218-220.

[4] Havaei, Mohammad, et al. "Brain tumor segmentation with deep neural networks." Medical Image Analysis (2016).

[5] Fakoor, Rasool, et al. "Using deep learning to enhance cancer diagnosis and classification." Proceedings of the International Conference on Machine Learning. 2013.

[6] Konstantinos Kamnitsasa , Christian Lediga , Virginia F.J. Newcombeb,c , Joanna P. Simpsonb , Andrew D. Kaneb , David K. Menonb,c, Daniel Rueckerta , Ben Glockera,
Efficient Multi-Scale 3D CNN with fully connected CRF for Accurate Brain Lesion Segmentation. `https://arxiv.org/pdf/1603.05959v2.pdf`.

[7] Oquab, Maxime, et al *"Learning and transferring mid-level image representations using convolutional neural networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.*

[8] Abràmoff, Michael D., et al. "Automated analysis of retinal images for detection of referable diabetic retinopathy." JAMA ophthalmology 131.3 (2013): 351-357.

[9] Oquab, Maxime, et al. "Learning and transferring mid-level image representations using convolutional neural networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

[10] Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." arXiv preprint arXiv:1512.00567 (2015)

[11] Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." Journal of Machine Learning Research 15.1 (2014): 1929-1958.

[12] Kingma, Diederik, and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

[13] Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." arXiv preprint arXiv:1512.00567 (2015).)

[14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov

"Department of Computer Scienc Dropout: A Simple Way to Prevent Neural Networks from Overfitting"