Computer “Vision”: A Deep-Learning-Based Approach to Detecting Diabetic Retinopathy

Pranav Upadhyayula
Department of Computer Science
Stanford University
pranavu@stanford.edu

Sushil Upadhyayula
Department of Computer Science
Stanford University
sushilu@stanford.edu

Lauren Yang
Department of Computer Science
Stanford University
lauren11@stanford.edu

1 Abstract

Diabetic retinopathy occurs when retinal blood vessels begin to leak, resulting in blindness. Because early detection of DR is vital in preventing permanent damage, our goal was to develop a model that can accurately and quickly detect DR severity. In this paper, we propose a deep learning approach to automatically detect the stage of diabetic retinopathy in digital color fundus photographs. Specifically, this approach feeds augmented retinal photographs into a DenseNet CNN and predicts, on a scale from 0 to 4, the extent of DR present. Because a common problem in the medical technology field is lack of interpretability, our model also generates a Class Activation Map (CAM) to reveal a visualization of regions that were most relevant when predicting the given image's class, which could assist ophthalmologists in their diagnosis. Our results show that using a DenseNet CNN outperforms the baseline model significantly and is particularly strong at identifying early-stage DR, a significant issue in the medical world.

2 Introduction

Although relatively unknown, diabetic retinopathy (DR) is by and large the leading cause of blindness in adults of the developed world. Affecting over 93 million people, the disease causes significant vision impairment in individuals who have diabetes, but can be easily averted if detected at an early stage [1]. The current diagnostic process begins with an ophthalmoscopic examination in which a trained clinician examines digital color fundus photographs of the retina. Unfortunately, diagnostic results are only ready days after the image is taken. Additionally, factors like clinician fatigue and experience and image quality can heavily contribute to human errors. Our goal is to use deep learning to provide a solution that can assist eye specialists in making accurate diagnoses. While previous attempts have made some strides in the right direction, our goal was to extend far past that and create something with potential to be put into practice.

To achieve this, we built a model that classifies microscope images of the eye into one of 5 classes, each of which represents a different stage (0-4) of DR. The input to our algorithm is an image of a retina, oriented as if viewed through a microscope condensing lens. We then use a DenseNet CNN with a Softmax output layer to output a predicted intensity (0-4) of DR. The numeric scale scores refer to increasing levels of severity: 0 means no DR, 1 means mild DR, 2 means moderate DR, 3 means severe DR, and 4 means proliferative DR.

3 Related work

The majority of previous attempts to automatically detect DR utilize medically relevant feature extraction coupled with various classifiers, resulting in satisfactory performance.

3.1 Two-class detection

Gardner et al. employed retinal exudate detection as well as pixel intensity with a neural network to categorize DR severity, ultimately achieving 88.4% sensitivity and

CS230: Deep Learning, Winter 2019, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

83.5% specificity. However, this network struggled with low contrast images of the eye, as their model did not use data augmentation techniques to account for different image qualities [2]. Sinthanayothin et al. preprocessed images with contrast enhancements, and used intensity variation and Recursive Region-Growing Technique (RRGT), a region-based image segmentation technique, to detect blood vessels, hard exudates, and other retinal features. Using a NN with these features, their approach yielded 80.21% for sensitivity and 70.66% for specificity [3].

3.2 Three-class detection

Mookiah et al. extracted 13 features (including texture, microaneurysms, blood vessels, distance between various characteristics, and exudates) and used three classifiers (Probabilistic Neural Networks (PNNs), Decision Trees, and Support Vector Machines), to detect three stages of DR. Their PNN classifier obtained an average classification accuracy of 96.15% [4]. Nayak et al. had a similar approach, using fewer features, and obtained an average accuracy of 93%, specificity of 100%, and sensitivity of 90% [5].

3.3 General Shortcomings

These four papers cited that their models could be improved with larger data sets and higher quality images in each class for improved feature extraction. Because they all share a feature-based approach, performance is susceptible to image quality, noise, artifacts, and laser scars. Additionally, obtaining large datasets with the ground-truth is difficult because a trained clinician is required to label images.

Furthermore, these studies focused on extracting medically relevant features. While undeniably useful, these approaches failed to consider unintuitive features that a CNN could learn. Our work differs because we attempt five-class classification for a finer-grained categorization of DR intensity. Additionally, our dataset includes blurry and over/underexposed images with the intention of developing a robust model that can handle low-quality images. Our method is a different take on prior literature, and we addressed the weaknesses of these models by utilizing a CNN, which can stand alone without the need for handmade medical features.

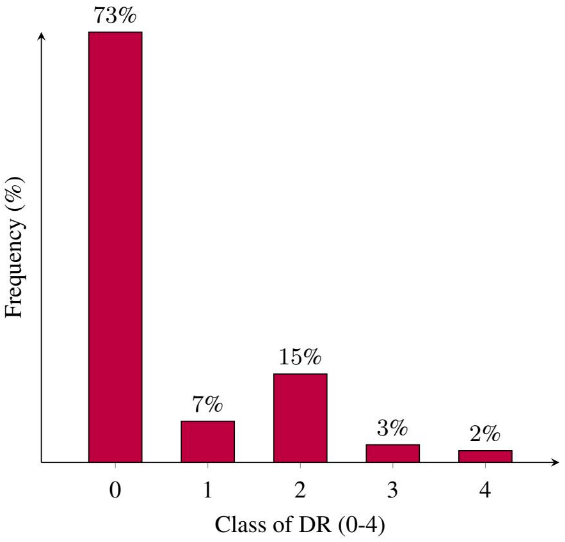
Finally, these models have low interpretability since they do not inform users what regions were most important to a specific classification. Our approach implements Class Activation Maps (CAMs), which allow clinicians to intuitively visualize which areas in the image were most essential to the diagnosis.

4 Dataset and Features

We used a rich, preprocessed dataset from Kaggle (https://www.kaggle.com/c/diabetic-retinopathy-detection/data) in order to perform our multi-class classification of the stage (0-4; 5 classes) of diabetic retinopathy present. This dataset contained 5,000 images, 1,000 of which we held out for the validation and test sets. Then, we used image data augmentation techniques to increase the size and strength of our train set (from 4,000 to 8,000 images). This is because the provided labeled data potentially included noise to accurately imitate real-world data - some images were blurry, had poor exposure, and varied in color scheme. So, having such additional data was germane to ensuring the robustness of our classifier. With data augmentation, we had 8,000 train examples, 500 validation examples, and 500 test examples (80/10/10 split).

4.1 Class Imbalance

From a descriptive analysis of our data, we realized that there was a significant class imbalance problem here: 73% of data belongs to class 0, 7% to class 1, 15% to class 2, 3% to class 3, and 2% to class 4, as shown in the histogram below. There were, however, an equal number of left and right eye images.



4.2 Resolution Pre-processing

The only pre-processing step that needed to be taken was modifying the resolution of the images. Excluding the RGB channel, the given images had resolution 4750 x 3160. However, after quick experimentation with our baseline model, we noticed that the size of these images caused learning to be quite slow. So, we shrunk each image's area to 475 x 316. Finally, since we used a CNN for our model, there was no need to perform any feature extraction steps; features were implicitly "created" by our convolutional layers.

4.3 Image Data Augmentation

We augmented every training data image with changes in blur, contrast, brightness, and color. To a given image, Gaussian blur with random σ between 1.0 and

2.0 was applied with probability 50%. Next, following the lead of literature in this field, an increase in contrast was also applied with probability 50% [6, 7]. Afterwards, we performed color shifting by multiplying the value of each color channel by a value between 0.75 to 1.25 with probability 30%. Finally, images were randomly slightly brightened or darkened using Keras's ImageDataGenerator class.

Rotation and cropping techniques - despite being popular for image classification - were not applied to the data. This is because having properly aligned and oriented images for DR diagnosis is crucial, and our images were all oriented the same way, as they all came from a microscope.

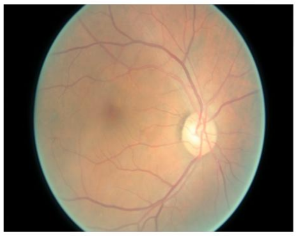


Figure 1: Original Image

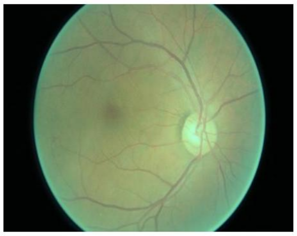


Figure 2: Augmented Image

5 Methods

We built two CNN's to perform this classification task: (1) a baseline CNN with two convolutional layers, and (2) a DenseNet CNN with 422 layers. Here is the link to the project's Github page: https://github.com/laurenyang/detect-diabet

5.1 Baseline Model

Convolutional Neural Networks (CNNs) are a class of deep learning model that uses convolutional filters to primarily analyze image data. We used Keras to create a CNN with 2 convolutional layers with 64 filters and ReLU activation. We then flattened the output of the second convolutional layer and fed that into a fully connected (dense) Softmax output layer with 5 classes.

5.2 DenseNet-121 Model

DenseNet is based off of the finding that CNNs are more accurate and efficient if they contain shorter connections between layers. While traditional CNNs only have connections between adjacent layers, DenseNet splits the CNN up into Dense Blocks, where every layer within a dense block is connected to every other layer within that block. By using the feature maps of all previous layers as input, DenseNet reduces the number of parameters by encouraging feature reuse. In addition, by being able to directly access all feature maps in its block, DenseNet solves the vanishing gradient problem. Literature shows that these advantages have allowed it to beat the state-of-the-art in benchmark object recognition tasks since its deep architecture allows it to understand the complex features central to advanced image classification [8].

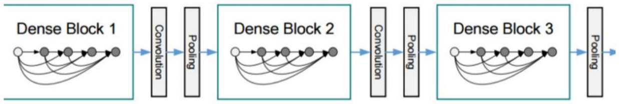


Figure 3: DenseNet Sample Architecture [9]

We used DenseNet-121, which contains 121 dense blocks, making a total of 121 batch normalization layers, 120 convolutional layers, 121 activation layers, 58 concatenation layers, and 1 global average pooling layer. We fed this last layer into a fully connected (dense) Softmax output layer with 5 classes. We used Stochastic Gradient Descent as our optimizer.

5.3 Sparse Categorical Cross-Entropy Loss

Cross-entropy loss is typically used for multi-class classification tasks. However, since our classes were encoded with integers 0-4 as opposed to one-hot encodings, we used a variation called sparse categorical cross-entropy loss. This is especially suitable for network with deep architecture like DenseNet since cross-entropy only depends on the output of a neuron rather than the gradient of the sigmoid function, which mitigates the vanishing gradient problem between dense blocks.

$$-\sum_{c=1}^{M} y_{o,c} log(p_{o,c})$$

Equation 1. Categorical Cross-Entropy Loss

5.4 Modifying the Loss Function

Because of the tremendous class imbalance problem (class 0 is far more common than the classes 1-4), we decided to modify our loss function so that in training, our model penalizes misclassifications of certain classes with a weight proportional to the log-inverse-frequency of that class. That way, the misclassification a rare class like 3 has a more significant effect on the weight updates than the misclassification of a common class like 0. To actually implement this, we set the class-weight argument in Keras's model.fit() function to class_weight = {0: 0.133841, 1: 1.15771, 2: 0.822009, 3: 1.60461, 4: 1.6956}.

This is also advantageous in a medical context. For precautionary reasons, it's much better to falsely put a patient on watch than to fail to diagnose someone. Since this system would be implemented to augment ophthalmologists' decision making, it behooves us to focus on what humans may miss and thereby give more weight to rarer classes.

6 Experiments/Results/Discussion

6.1 Hyperparameter Tuning

We used a learning rate of 0.01 because initial experimentation showed us that a learning rate of 0.001 was far too slow in training whereas a learning rate of 0.1 led to high bias. Experimenting with a standard batch size of 32 revealed that the difference between Bayes error and training error was significantly lower than the difference between training error and validation error. Thus, to decrease variance, we used a batch size of 64 instead. The model was trained for 20 epochs. We were careful not to train longer due to the inherent variance in eye shapes and colors; however, we didn't want to train for too less time given that detecting DR is quite a complex problem.

6.2 Results

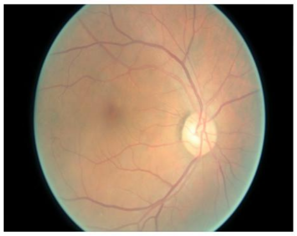


Figure 4: Original Image

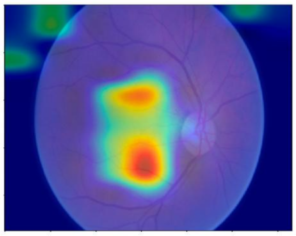


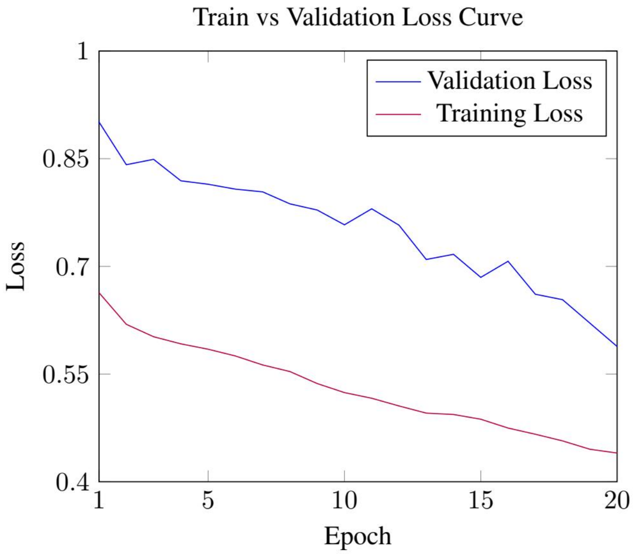
Figure 5: Heatmap of Image

Table 1: Confusion Matrix

	Predicted				
	0	1	2	3	4
True 0	671	2	42	0	0
True 1	5	66	8	0	0
True 2	18	5	119	13	0
True 3	1	4	10	9	4
True 4	1	0	7	4	11

Table 2: Results Table

	Sensitivity	Specificity	Accuracy
Baseline	0.674	0.925	0.702
Class 0	0.938	0.912	0.938
Class 1	0.857	0.988	0.835
Class 2	0.64	0.921	0.768
Class 3	0.346	0.983	0.321
Class 4	0.733	0.996	0.478
Overall (micro avg)	**0.876**	**0.969**	**0.876**



6.3 Discussion

In accordance with relevant medical literature, we chose to measure sensitivity, specificity, and accuracy. Sensitivity measures the algorithm's ability to correctly identify those with DR while specificity measures the algorithm's ability to correctly identify those without the disease. Sensitivity is of particular importance given the gravity of false negatives in diagnosis. Given the class imbalance, we use micro-averaged metrics to evaluate our overall model.

Qualitatively speaking, the results show that our model had sensitivity and specificity measures of 87.6% and 96.9%, both higher than the 67.4% and 92.5% of our baseline. In addition, DenseNet produced a micro-averaged accuracy of 87.6%, compared to our baseline model's accuracy of 70.6%.

The confusion matrix reveals that entries are primarily situated on the diagonal, indicating a strong true positive rate. Further analysis reveals that the DenseNet model was particularly adept at classifying stages 0 (no DR), 1 (mild DR), and 2 (moderate DR) accurately. Particularly, the accuracy was above 75% for all three classes, and Class 1 (mild DR) had a sensitivity of 85.7% and a specificity of 98.8% Since detecting these early stages of DR is most challenging for ophthalmologists, we know that our model has tremendous potential to assist doctors with early-stage detection.

That being said, a relative weakness of our model was with stages 3 (severe DR) and 4 (proliferative DR). We believe that this weakness was due to the severe class imbalance and lack of data for these classes; while we did take measures to prevent overfitting to the first 2 classes, we believe even more work in this area would alleviate this issue. However, manually diagnosing Stages 3 and 4 is not too difficult for specialists because the visual markers in the image are often quite apparent when DR is this severe. Hence, our model overall has great diagnostic prowess if used in conjunction with an eye specialist's diagnosis.

On the qualitative side, the graph of our loss function shows that as expected, the loss is decreasing over time. Perhaps most valuable feature of our model is our use of Class Activation Mapping (CAMs). As can be seen in results, CAMs use a heatmap to visualization the regions of the eye image the deep learning model is using to make classification decisions. For instance, the doctor would then know to take a closer look at the two highlighted regions. This is most useful as it allows our model to be interpretable, which is far more valuable in augmenting decision-making than slightly increased performance would be.

To prevent overfitting, a higher batch size was used, the model was only trained for 20 epochs, and we changed the model weights to be inversely proportional to the class imbalance. Even with these countermeasures, we believe we slightly overfit to the training set given that the difference between Bayes error and training error was significantly lower than the difference between training error and validation/test error.

7 Conclusion/Future Work

Our goal was to build a robust model capable of taking a retinal microscope image as input, and outputting the stage (0-4) of DR present. We built two models: (1) a baseline CNN with two convolutional layers, and (2) a DenseNet CNN with 422 layers.

7.1 Summary of Algorithms

Our DenseNet CNN significantly outperformed the baseline model because it was a deeper, more intricate model that allowed for more complex visual features to be learned. Furthermore, DenseNet requires fewer parameters due to extensive feature reuse and helps mitigate the vanishing gradients problem, which likely further contributed to its success.

Specifically, the DenseNet model was best at classifying early-stage (0-2) images, which are the most difficult diagnoses for ophthalmologists, so we know that our model has tremendous clinical potential. While our model's weakness was late-stage (3-4) images, ophthalmologists often have little difficulty making these diagnoses, so these misclassifications would have very little impact if checked by an ophthalmologist.

7.2 Future Work: Model

In the future, to refine our model, we could first try running our model on more data; no amount of model refinement will ever be a replacement for gathering more data. We have access to 44,000 more images, but computational resources limited our ability to use them, so using them in the future could yield a superior model. Furthermore, we could try freezing less layers on our pretrained model so that the trained weights are more specialized to our eye images.

7.3 Future Work: Computer-Aided Diagnosis

Beyond the model, we have actively been connecting with eye doctors to gather research on how our tool and CAM visualization could help improve DR screening. Per our conversations with specialists, they often have difficulty detecting DR at its early stages, and regular eye doctors are not even trained to diagnosis DR, so they expressed that a tool like ours would be immensely helpful for strengthening their diagnoses. In fact, one doctor indicated that they could not share much because high-profile companies had reached out to them about using such products and it would be conflict of interest, so we know that this is an important space. Ultimately, we are optimistic and confident that focusing on model interpretability (for example, by expounding upon our CAMs approach) will prove most useful in the real-world by facilitating eye doctors' diagnostic processes.

8 Contributions

The three of us worked on defining the problem to solve from the beginning. Pranav worked on getting the baseline model to work and did research into which architectures could be best for our final model (which ended up being DenseNet). Sushil worked on implementing the DenseNet model. Lauren worked on implementing the CAMs in order to achieve better interpretability of results. Pranav and Sushil worked on extracting and analyzing our evaluation metrics. We each focused on writing different sections of the paper (Lauren: Abstract, Introduction, Related Work; Pranav: Dataset and Features, Conclusion and Future Work; Sushil: Methods, Results), and afterwards, all three of us independently proofread the paper and made changes accordingly. We all worked together on putting together the images/diagrams/figures for the paper. Lauren worked on compiling our write-up into LaTeX, and Pranav and Sushil worked on transforming our write-up into the final poster.

References

[1] Lee, R., Wong, T. Y., Sabanayagam, C. (2015). Epidemiology of diabetic retinopathy, diabetic macular edema and related vision loss. Eye and vision (London, England), 2, 17. doi:10.1186/s40662-015-0026-2

[2] Gardner, G. G., Keating, D., Williamson, T. H., Elliott, A. T. (1996). Automatic detection of diabetic retinopathy using an artificial neural network: a screening tool. The British journal of ophthalmology, 80(11), 940-4.

[3] Sinthanayothin, C. , Boyce, J. F., Williamson, T. H., Cook, H. L., Mensah, E. , Lal, S. and Usher, D. (2002), Automated detection of diabetic retinopathy on digital fundus images. Diabetic Medicine, 19: 105-112. doi:10.1046/j.1464-5491.2002.00613.x

[4] M. R. K. Mookiah, U. Rajendra Acharya, Roshan Joy Martis, Chua Kuang Chua, C. M. Lim, E. Y. K. Ng, and Augustinus Laude. 2013. Evolutionary algorithm based classifier parameter tuning for automatic diabetic retinopathy grading: A hybrid feature extraction approach. Know.-Based Syst. 39 (February 2013), 9-22. DOI=http://dx.doi.org/10.1016/j.knosys.2012.09.008

[5] Nayak, J., Bhat, P. S., Acharya U, R., Lim, C. M., Kagathi, M. (2008). Automated identification of diabetic retinopathy stages using digital fundus images. Journal of Medical Systems, 32(2), 107-115. https://doi.org/10.1007/s10916-007-9113-9

[6] Valverde C, Garcia M, Hornero R, Lopez-Galvez MI. Automated detection of diabetic retinopathy in retinal images. Indian J Ophthalmol. 2016;64(1):26-32.

[7] Lam C, Yi D, Guo M, Lindsey T. Automated Detection of Diabetic Retinopathy using Deep Learning. AMIA Jt Summits Transl Sci Proc. 2018;2017:147-155. Published 2018 May 18.

[8] Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K. Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4700-4708).

[9] Chablani, M. (2017, August 24). DenseNet. Retrieved from https://towardsdatascience.com/densenet-2810936aeebb