Diabetic Retinopathy Classification using Machine Learning Techniques

Debanshu Das
Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
debanshd@andrew.cmu.edu

Subhagato Dutta
Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
subhagad@andrew.cmu.edu

Anoop Hallur
Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
ahallur@andrew.cmu.edu

Surbhi Motghare
Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
smotghar@andrew.cmu.edu

Abstract

Diabetic Retinopathy (DR) is an eye disease associated with long-standing diabetes and its detection by a trained clinician is a tedious task. In this project, we aim to detect Diabetic Retinopathy, using high resolution images of human retina called fundus image. We have used multiple approaches to solve the problem. Initially, we were using the "bag of visual words" approach to obtain histogram features and used this training dataset to train our classifiers. Then we proceeded to use Convolutional Neural Networks(CNN) to detect the level of Diabetic Retinopathy on our extracted features. To improve our results, we used the raw images directly with minimum preprocessing to train convolutional neural networks.

1 Introduction

Diabetic retinopathy, a retinal vascular disorder that occurs as a complication of diabetes mellitus (DM), is a leading cause of blindness in the United States, often affecting working-aged adults.It is characterized by signs of retinal ischemia (microaneurysms, hemorrhages, cotton-woolspots, intraretinal microvascular abnormalities, venous caliber abnormalities, and neovascularization) and/or signs of increased retinal vascular permeability. Vision loss can result from several mechanisms, including neovascularization leading to vitreous hemorrhage and/or retinal detachment, macular edema, and retinal capillary nonperfusion. Retinopathy occurs in most persons with long-standing DM, but its incidence rate can be reduced by aggressive control of hyperglycemia and hypertension.

The estimated US general population prevalence rates for retinopathy and vision-threatening retinopathy were 3.4% (4.1 million persons) and 0.75% (899000 persons). Approximately 4.1 million US adults 40 years and older have diabetic retinopathy; Detecting DR in early stage helps surviving from severe vision loss.

1.1 Related Work

The idea behind classifying Diabetic Retinopathy is that patients who are marginal cases on having DR must go for further medical examination. That is why the classification provided by using machine learning techniques must avoid falsely classifying the images as far as possible. To address

this problem, Anderson Rocha, Tiago Carvalho,Siome Goldenstein, Jacques Wainer, have proposed a solution in their technical report on Points of Interest and Visual Dictionary for Retina Pathology Detection. [5] They constructed a visual dictionary of the important features and classified if an ocular fundus image is of a patient with DR or non-DR. The important features they have used are exudates, haemorrhages, and microaneurysms and tested various paramter configurations.However, they obtained the points of interest to create a visual dictionary of 100 words which were selected by a trained specialist. Whereas, in our case, we aim to create a learning model which is fully automated and does not involve any preliminary dependencies on a specialist or ophthalmologist.

Sopharak et al. have used 15 per-pixel features as potential indicators of exudates. They used naive Bayes and SVMs for feature selection and pixel classification. They filtered images for noise, enhanced image contrast, detected and removed the optic disc, extracted local features describing pixels or regions and then classified those features using a model built from a training set. Their naive Bayes classifier, after feature selection, achieves an overall per-pixel sensitivity of 93.38%, specificity of 98.14%, precision of 47.51%, PR of 70.45% and an overall accuracy of 98.05% on a test set not used during training. The SVMs classifier achieves an overall per-pixel sensitivity of 92.28%, specificity of 98.52%, precision of 53.05%, PR of 76.27% and an overall accuracy of 98.41% [6].

However, in their approach they used only 39 training and test images. The claims made could be left unsubstantiated if the resulting classifier is tested on a bigger data-set. In addition,they have images having uniform lighting conditions and image resolution while our dataset is huge where image features tend to vary across images.

R.Priya et al have approached the same problem using Probabilistic Neural network (PNN), Bayesian Classification and Support vector machine (SVM). They have extracted features like blood vessels, hemorrhages of NPDR image and exudates of PDR image for classification using a different image processing sequence (as compared to Sopharak et al) for 350 fundus images, out of which 100 were used for training and 250 images were used for testing. They have found that PNN has an accuracy of 89.6 %, Bayes Classifier has an accuracy of 94.4% and SVM has an accuracy of 97.6%. [4]

This approach deals with the problem of classifying Diabetic Retinopathy as a binary problem i.e if DR exists or does not. However, we need an approach that helps us further classify DR as one of the 4 types that have been outlined. We also need something that will work well with different quality images.

Finally, deep learning is a promising method of learning representations of data and can be extended to improve diabetic retinopathy classification. Lilly Oetting, in her report- Deep-learning Techniques for Diabetic Retinopathy Classification, has performed deep-learning experiments whoich centered on stacked sparse autoencoders.The performance of a Random Forest classifier was taken as the baseline result. The best result she obtained had the receiver operating characteristic (ROC) of 0.86595.

These relevant pieces of work gave us an insight into the intricacies of the problem at hand.In our project, we started with addressing the binary classifcation problem of a patient having DR or not. Finally, we proceeded to address a multiclass DR classification problem.

2 Methodology

2.1 Preliminary Work (Midway)

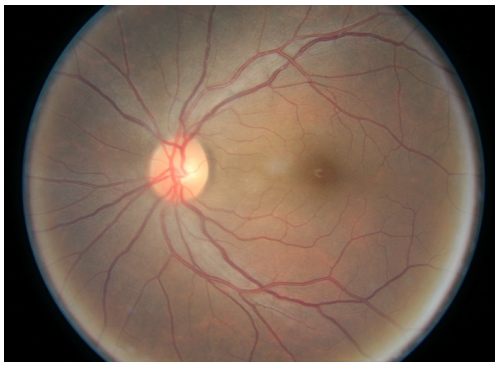
We wanted to apply some techniques to classify whether an eye has diabetic retinopathy i.e We wanted to build a binary classifier instead of predicting the multi-class 5 levels of diabetic retinopathy for the kaggle competition requirement.

The first step of our pipeline is resizing the images so that all have similar dimensions. So we resized each image to have a width of 1000 pixels (adjusting height to maintain the original aspect ratio).

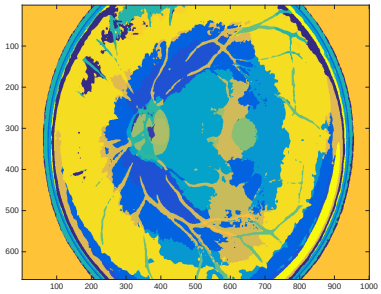
For our midway work, we used the technique called Bag of visual words. We believe that the retina images are just a collection of visual words. The occurrences of some specific visual words (blood vessel haemorrhages, exudates) at different frequency determines if they eyes are affected by

diabetic retinopathy. In the second step we converted the images from RGB space to LAB space. Then we applied different filters (Gaussian, LoG, HoG, Sin Gabor, Cos Gabor) at different scales, so a total of 30 filters are applied. So for the 3 channels total $30 \times 3 = 90$ filter response images are produced. In the next step we picked α (500 in our case) random points from each image (with a validity check, to ensure the points are picked from within the eye region, not the black region outside), with the belief that we will uniformly get all types of pixels from the images. Now each point has 90 features because of 90 filter responses. These points are fed to k-means clustering algorithm to cluster into K (100 in our case) different clusters. We call them the dictionary of visual words. They represent all types of pixels in the eye. Next for all the images in the our data set (train and test), for each pixel, we compute the 90 filter responses and replace it with the nearest visual word (by using 1 nearest neighbor algorithm from the dictionary). By this step we are ensuring that all pixel values have only 1-K discrete values. This process of converting to visual word is needed because it drastically reduce the size of data we feed into the machine learning algorithm and hence reduce complexity of the algorithm, while preserving most of the information of the images.

A sample image and its corresponding visual word can be visualized as follows.



Image



Visual Words

In the last step of preprocessing we generated a 1-D histogram of all the words in every image. We normalized the histogram on the scale of 1. As our $K = 100$ the histogram vector length of each image is 100. We believe that histogram of word vectors captures all the relevant information because we are interested only in the presence of specific features in the image (hemorrhages, exudates and other interest points usually present in eye which has diabetic retinopathy). At this point, we have successfully extracted the features of the image i.e each image is represented by its corresponding feature vector.

For the classification task, we started using Logistic Regression as the classifier since it's the simplest classifier familiar to us. However, later we realized that for difficult image classification problems, Support Vector Machines or simply SVM, can generalize well for high-dimensional histogram features. Thus, SVM is considered a good candidate because of its high generalization performance without the need to add a priori knowledge, even when the dimension of the input space is very high [1]. In addition, SVM has a faster training speed. As such, we opted for SVM to be applied to our histogram of word vectors.

In our technique for training a model for classification, we initially used the Logistic Regression approach. To begin with, we used 500 images as the training dataset to train the classifier. We then used 160 images for cross-validation to measure the correctness of our classifier. The error rate obtained over this cross-validation dataset was huge about 51.88 %, same as random guessing. So the classifier is not able to learn the target function at-all. We observed with less number of images it could separate the data linearly but failed with more data. The training error with 8000 data points is 39.51%. We then proceeded with Support Vector Machines (SVM) for classification (with an idea that it can be easily extended to multiclass classifier). Initially we trained SVM with a linear kernel. When we trained the model and applied the model to our cross validation data set the error was about 46 %. Then we used the 9000 images as training dataset and used 1000 images to test the performance of the classifier. Surprisingly, the error rate remained almost same to 44%.

As this larger dataset was prone to noise, all of the data could not be fit correctly and the accuracy deteriorated. These results needed a significant improvement. From this we arrived at the conclusion that the decision boundary might not be linear.

As a next step, we used SVM with Radial Basis Function (RBF) kernel and the cross-validation error dropped to 36 %. However, its performance on the training data, i.e. training accuracy, was about 77% indicating the the classifier could not fit all the data accurately.

Finally we used the Neural Network methodology. For 100 hidden neurons, we got an accuracy of 60% . For 500 hidden neurons, the accuracy on cross-validation dataset was 65% and the training accuracy was about 72 %.

The results at each step of our research are tabulated as follows:

Method	Training Data	Testing data	Training Error(%)	Testing Error(%)
Logistic Regression	500 images	160 images	29.40 %	51.88 %
	9000 images	1000 images	39.51 %	41.80 %
SVM	500 images	160 images	25.20 %	45.62 %
	9000 images	1000 images	40.27 %	44.30 %
SVM with RBF Kernel	500 images	160 images	10.80 %	52.50 %
	9000 images	1000 images	23.70 %	36.6 %
Neural Network	(i) 100 hidden neurons	1000 images	45.22%	46.01%
	(ii) 500 hidden neurons	1000 images	47.19%	47.76%

Moreover, as shown in the preliminary results, the error in using logistic regression was higher than expected. Later, as we started using default Neural networks toolbox in MATLAB,we got initial results, and it looked promising, and hence we decided to build on this in the later part of the project.

2.2 Feature Extraction

In the second approach, to the problem convolutional neural networks for classification(CNN) have been used. Since CNN's approach suggests against using complex image processing techniques for extracting features, the minimum processing of the images was done. The images were resized to 256 x 256 pixels as the CNN's we employed , expected the input images to be of this size. The details of image preprocessing done arejnk summarized as follows

- The color image is converted to grayscale.
- Average intensity multiplied by a factor of 0.2 is used as a threshold to generate a binary image. The output of this step is a central image with noise in the periphery
- Contour finding is used on the binary image to get the contours that have data. However, these contours can also include noise. Iteratively cycling through all the contours helps find the biggest contour so that noise can be eliminated from the background portion of the retina image.
- Now there are two types of images based on the aspect ratios in which the images were taken. The first type is when the retina is a complete circle and the second type is when the retina image is cropped due to the way the image was taken. It is necessary to preprocess these so that the features of the images are in the form of a circle. This is needed to match it with the features of those images where the contour is already in the form of a perfect circle.
- From the biggest contour points which are at the boundary with a 20 pixel tolerance are removed.
- A circle fit algorithm called Tobin's method is run on the 2D vector of points to get the center and the radius of the equivalent circle. This circle is used to crop the color image and resize the images so that the diameter of the circle is exactly 256 pixels.
- At this point 256 x 256 representations of the retina images have been created to feed to a CNN. This size specification is required by Alexnet to function properly.

- Histogram Equalization is perfomed on the images that we have obtained from the previous steps which helps in removing the effects of different lighting conditions in the images in the dataset.Now there are images where all intensities have the similar number of pixels.
- Color Histogram Equalization is a predominant technique in Computer Vision where images are converted from RGB to the YCbCr scale where the Y components are sensitive to the rods in the eye and the CbCr components are sensitive to the cones in the retina. Histogram Equalization makes sure that only the Y component is transformed to a modified Y. Then this newly modified image in the YCbCr scale is transformed back to the RCB scale as that is required as input to the Convolutional Neural Network.

Table 1: Pre-processing images

Original Image

Pre-processed image

2.3 Machine Learning

Various implementations of Convolutional Neural Networks were used for the next stage of the project. However it was necessary to balance the distribution of labels i.e for each of the 5 classes, to make their representations uniform, otherwise the network tended to be biased towards the most frequently seen classes in the training data set. The precise distribution can be seen below

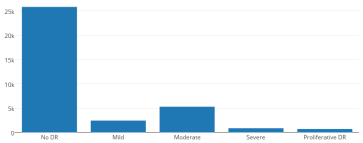


Figure 1: Distribution in Training Data

75% of the images had a label of "No Diabetic Retinopathy" and hence the images in other labels were duplicated. In one of the initial CNN's training was performed without balancing the classes, and the model predicted "No DR" as the label for 75% of the test images too, irrespective of the data set used. Hence, it was necessary to balance the distribution of the labels.

Three implementations of CNN were tried. Googlenet [7], AlexNet [3] and ImageNet Feature Extraction combined with Logistic Regression were used.

AlexNet [3] is a Convolutional Neural Network designed to work with ImageNet images. The system is composed of a mixture of what is called convolutional layers, Max pooling layers and fully connected layers. Every layer is composed of multiple neurons. Each neuron has a randomly initialized k x k matrix of weights. A generic CNN is explained in detail in the coming sections

- A k x k matrix of initial random weights was used to perform a 2D convolution with the input image having dimensions 256 x 256. In theory this is done by taking a sliding window of k x k pixels in the image, taking a dot product with matrix of weights while moving the sliding window one pixel row at a time in the horizontal direction.
- This sliding window is shifted from the left to right and from top to bottom in sequence thereby getting an image output of 256-k/2 pixels by 256-k/2 pixels (as in the technique extra iterations that take place at the edges are omitted)

- The image is then fed to the max polling layer which divides the image into a grid of l x l matrices and takes the maximum value of the pixel in a given l x l division. It thus reduces the image to a 1/l in height and width. It has been empirically found that the max value gives the best feature while a parameter like the average value still retains the original features.
- This process continues until all the convolutional layers are exhausted. After which the fully connected layers (the image size could be 64 x 64) are reached. Here the 2D array of points are converted into a 1D array of points having 4096 elements.
- Now a dot product of the 1D array of elements in the first fully connected layer is performed with a set of randomly initiated weights to derive each element of the array of the next fully convoluted layer one point at a time.
- Ultimately an output of 5 output neurons which correspond to the classification of the images based on our 5 levels of diabetic retinopathy is obtained. This is compared with that of the true labels and the loss function is computed which is the difference between the squares of the obtained values and the true values.
- Back propagation algorithm is used from the layer representing the output of these 5 classifications to retrain the randomly initialized weights for each neuron in each layer.
- Thus all our 35000 images are trained in this way. Each training of the dataset is called an epoch. We then use many such epochs to train our neural network

The structure of AlexNet is similar to what is described above. But googLeNet and FeatureExtrcation neural nets are similar, but more complex. In FeatureExtraction Net, the initial layers are same as AlexNet, i.e succesive paris of convolutional layers, pooling layers and normalization layers. However the last few layers are omitted, instead a multiclass logistic regression classifier with Softmax classifier is used, eliminating few final layers. In GoogleNet, the complexity is increased exponentially. It uses a 22 layer deep neural network, each layer specifically designed to extract specific features of the image. The results have been analyzed in the next section.

2.4 Software organization

For the implementation we are using Matlab as a platform. For the image filters we have used, we used matlab parallel computing toolbox to utilize Nvidia CUDA cores. For generating the visual word maps of each image, we require to match each pixel with the closest visual word from the visual word dictionary. That is a computationally intensive task. So we decided to write cuda optimized kernel to do this for every image. A matlab mex interface is used to create a bridge between the cuda code and matlab. This mex file is called every time to compute the visual words for every image. That make the code 2.5x faster than CPU implementation.

We used the logistic regression code we wrote for class assignment to build a classifier. But for SVM and Neural Networks we used Matlab libraries.

In the next stage, for the convolutional neural networks, we implemented the preprocessing tasks in python and used opencv for some parts of pre processing images. We also used imagemegick tools for quick processing. For training the CNN, we used a framework of caffe and digits(develeoped by UC Berkeley) [2], where in we have only have to specify the architecture of the CNN and need not worry of correctness the code. We customized the available networks and obtained the results.

We used NIVIDIA GPU GTX Titan with 2688 cuda cores, and 6 GB of GDDR5 and rented cloud based GPU instances (g2.2xlarge) from Amazon Web Services. Since training each neural network, needs almost an entire day, we could try only limited number of configurations within out budget in this phase of the project.

3 Experiment

3.1 Dataset

We obtained our dataset from the Kaggle Diabetic Retinopathy Detection which is an on-going international competition this year. The dataset contains a large set of high resolution retina images

taken under different imaging conditions. The left and right eye retina images of the subjects are available. The training set provided has images labeled with subject IDs and indicates if it is a left or right eye retina image. For eg., 5_right.jpeg is the right eye retina image of patient number 5.

These images were obtained from different types of cameras thus the dataset had non-uniform resolution. This also affected the visual appearances of the left-right eye images. For instance, some images resembled the way we would look at the retina anatomically, i.e., macula on the left, optic nerve on the right for the right eye. Whereas, some other images looked as if observed through a microscope, i.e., as one would see in a live eye examination. Some images were affected by the presence of noise. These images were either out of focus, underexposed, or overexposed.

A clinician has certified the presence of Diabetic Retinopathy(DR) in these patients. The scale ranges from 0 to 4 as follows:

Scale	0	1	2	3	4
Level Of DR	No DR	Mild	Moderate	Severe	Proliferative

The entire dataset consisted of 35,126 labeled images and approximately 53,000 unlabeled images, which depicted the different levels of Diabetic Retinopathy. For the shallow networks, we used out of these labeled images, we used 9000 images as the training data and another 160 images to form the cross-validation dataset as summarized in the table.

However, for the CNN, we approached it as a multi class classification problem , and did not modify the labels. We used only the labeled images for training and cross validation, and the the unlabeled images as test images(We could get the evaluations by submitting the labels though).

Before we proceed to describe the techniques, we want to establish the performance metric. For the midway report, we had established the performance technique to be percentage of correctly classified images in the test data set since we were treating it as a binary classification problem. In the next phase, we used a weighted quadratic kappa score as our metric for evaluating our model since it penalizes large offsets more than small offsets. i.e if true label was 2, our prediction of 4 would be penalized more than the prediction of label 3 and so on.

This metric typically varies from 0 (random agreement between raters) to 1 (complete agreement between raters). In the event that there is less agreement between the raters than expected by chance, this metric may go below 0. The quadratic weighted kappa is calculated between the scores assigned by the human rater(true label) and the predicted scores.

Images have five possible ratings, 0,1,2,3,4. Each image is characterized by a tuple (e_a, e_b), which corresponds to its scores by Rater A (human) and Rater B (predicted). The quadratic weighted kappa is calculated as follows. First, an N x N histogram matrix O is constructed, such that Oi,j corresponds to the number of images that received a rating i by A and a rating j by B. An N-by-N matrix of weights, w, is calculated based on the difference between raters' scores:

$$w_{i,j} = \frac{(i-j)^2}{(N-1)^2}$$

An N-by-N histogram matrix of expected ratings, E, is calculated, assuming that there is no correlation between rating scores. This is calculated as the outer product between each rater's histogram vector of ratings, normalized such that E and O have the same sum.

From these three matrices, the quadratic weighted kappa is calculated as:

$$\kappa = 1 - \frac{\sum_{i,j} w_{i,j} O_{i,j}}{\sum_{i,j} w_{i,j} E_{i,j}}$$

Results

Model I

First we trained the images with a modified AlexNet structure, in which we kept the same weights for the first 5 layers (Convolution and Max Pooling) as of a learnt ImageNet, assuming that the low level features learnt from the imageNet database are good enough. Then we learnt only the high

level layers (Fully connected layers) to get the features relent to eye retina which are responsible for diabetic retinopathy.

The confusion matrix for this model when trained on all 32K images and tested on 3K validation images are

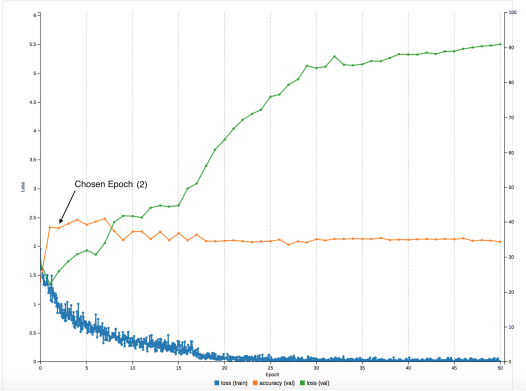


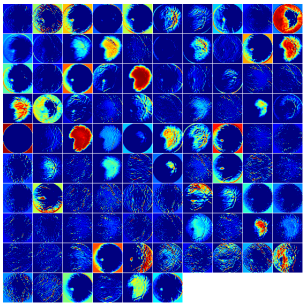
Figure 2: Learning curve on multiple epochs

0	predicted				
actual	0	1	2	3	4
0	2137	42	214	13	122
1	221	8	14	3	11
2	379	8	109	6	28
3	28	2	17	10	13
4	15	0	15	1	46

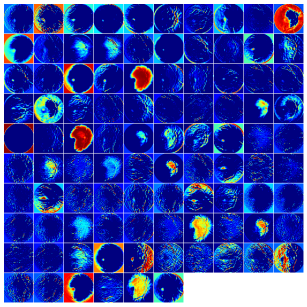
Table 2: Confusion Matrix

This gave a Quadratic Weighted Kappa score of 0.28 and accuracy of 67%. The features extracted from this are presented below

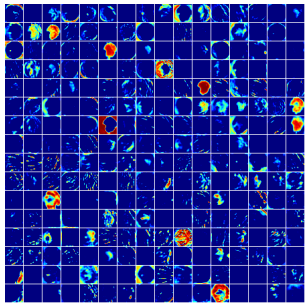
Table 3: Activation on low level features



Layer 1



Max Pooled from Layer 1



Layer 2

Model II

The kappa score we got training that system was not satisfactory in the Kaggle competition. So we moved on doing more experiments trying GoogLeNet. The confusion matrix for this model when trained on all 32K images and tested on 3K validation images are

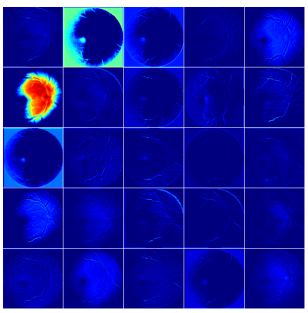
Figure 3: Learning curve on multiple epochs

	predicted				
actual	0	1	2	3	4
0	1966	8	383	22	149
1	194	0	47	6	10
2	250	1	166	51	62
3	12	0	24	23	11
4	7	0	18	10	42

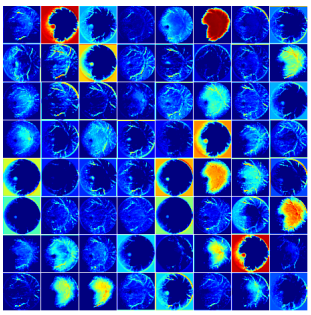
Table 4: Confusion Matrix

This gave a Quadratic Weighted Kappa score of 0.36 and accuracy of 71%. We submitted this result to Kaggle and ranked 52. The features extracted from this are presented below

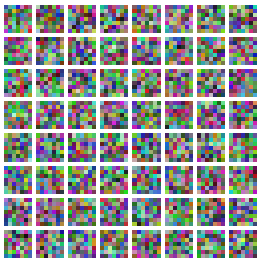
Table 5: GoogLeNet Learnt Features Visualization



Layer 1



Max Pooled from Layer 1

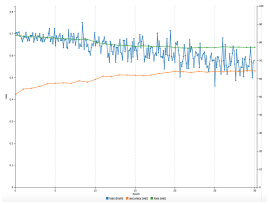


High Level learnt features

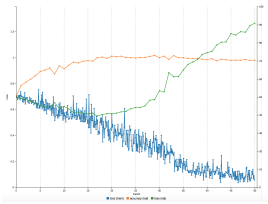
Model III

In this we tried to learn the partitions between classes by learning 4 binary classifiers using AlexNet, they are 0 vs 1-4, 0-1 vs. 2-4, 0-2 vs. 3-4, and 0-3 vs. 4. We observed that AlexNet worked better learning this binary features than the GoogLeNet.

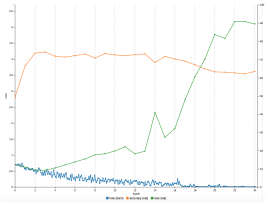
Table 6: Learning multiple binary classifiers



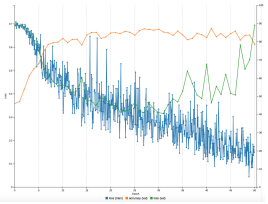
0 vs 1,2,3,4 (64.5 %)



0,1 vs 2,3,4 (72.14 %)



0,1,2 vs 3,4 (73.12 %)



0,1,2,3 vs 4 (85.65 %)

4 Future work

So far the network of GoogLeNet has given us the best kappa score of 0.36 %. Which is decent, but we believe that we can do better by combining multiple classifiers such as the binary classifiers trying to find boundary and the multi-class classifiers. We identified some drawbacks of our algorithm. They are listed below

1. We used multi-class classification neural networks to directly do multi-class classification. Instead, a better approach is to use binary classification neural network i.e detect whether level of DR is above threshold or below threshold, for threshold = 1,2,3,4 and use a boosting technique to obtain a multi-class classifier. Based on our research, we found that using boosted regression trees with the technique we have used, gives a better kappa score than what we had achieved.
2. The neural networks we had used , all worked on images of size 256 x 256 pixels. We believe that if we could build a neural net architecture which takes in images of 512 x 512 or even higher, we could achieve better results as the images were of much higher resolution 3000 x 3000 on average. In compression, a lot of valuable information might have been lost. In class 1 (Mild DR) micro-aneurism can only be detected in a high resolution image. We are planning to add one more layer to AlexNet to address the resolution issue.
3. We have a large number of unlabeled images. We can perform unsupervised learning by doing a Restricted Bolzman Machine(RBN) Deep Neural Network and then extract all the features from that. There is a lot of scope doing semi-supervised learning in this project. We will explore all those in future.

5 Conclusion

At the midway phase, we had seen that the results were not satisfactory yet. In the next phase of our work, we achieved reasonable results, but we still plan to continue working on the project with convolutional neural networks as this competition is open to public till it ends in July 2015. So far, the approach of basic CNN's helped us to get a score of 0.36 and a rank of 54 in the Kaggle competition that provided the dataset.

Acknowledgments

We take this opportunity to extend our gratitude towards Prof. Maria-Florina Balcan and our mentor Micol Marchetti-Bowick for their guidance and support. We also thank Prof. Tom Mitchell for enriching our knowledge throughout the course of the class.

References

[1] Olivier Chapelle, Patrick Haffner, and Vladimir N Vapnik. Support vector machines for histogram-based image classification. *Neural Networks, IEEE Transactions on*, 10(5):1055–1064, 1999.

[2] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[4] R Priya and P Aruna. Svm and neural network based diagnosis of diabetic retinopathy. *Int J Comput Appl*, 41(1):6–12, 2012.

[5] Anderson Rocha, Tiago Carvalho, Siome Goldenstein, and Jacques Wainer. Points of interest and visual dictionary for retina pathology detection. *Institute of Computing, University of Campinas, Tech. Rep. IC-11-07*, 2011.

[6] Akara Sopharak, Matthew N Dailey, Bunyarit Uyyanonvara, Sarah Barman, Tom Williamson, Khine Thet Nwe, and Yin Aye Moe. Machine learning approach to automatic exudate detection in retinal images from diabetic patients. *Journal of Modern optics*, 57(2):124–135, 2010.

[7] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.