Final Year Project Report

Classification of Diabetic Retinopathy Using Retinal Images

Prepared by: **Mark Dhruba Sikder (Student ID- 26529548)**

&

Asif Rana (Student ID- 27158632)

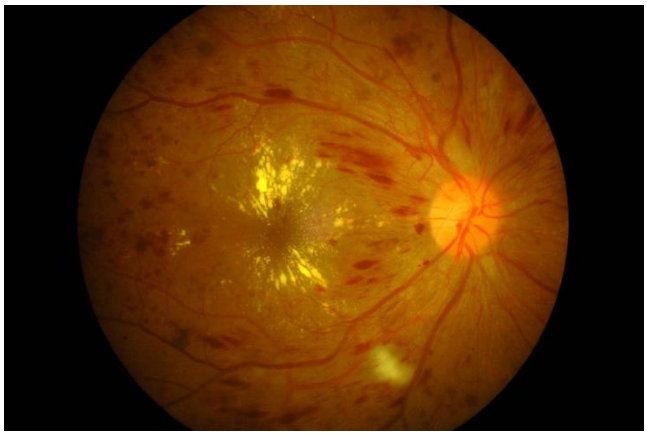
Course: Bachelor's in Computer Science, C2001

Date: 19 October 2018

Supervised by: **Dr. Muhammad Fermi Pasha**

Contents

Introduction



It is predicted that by the year 2035, the count of diabetic patient will rise to 562 million(Zhou, Zhao, Yang, Yu, & Xu, 2018) with Diabetic Retinopathy being one of the many diseases associated with diabetics(Pang, Luo, & Wang, 2018). Along with many complications a diabetic patient has a high chance to suffer from critical level vision loss and in worst case permanent blindness due to DR. 40-50% of the global population is in the treat of becoming a victim of DR(Bravo & Arbeláez, 2017). This catastrophic disease is claimed to be the major cause of blindness and with time the number of patients losing their eyesight is increasing rapidly(Bravo & Arbeláez, 2017). Rate of DR can be reduced significantly if the disease can be addressed in the early stages (Bravo & Arbeláez, 2017); detecting DR in the early stages is a challenge to modern science. Since, it has no visual indication of this disease in its preliminary stage(Bravo & Arbeláez, 2017). The necessity of detecting (DR) in early stage becomes an important task to accomplish in the health sector. In the past DR was classified as a disease which cannot be cured (Bravo & Arbeláez, 2017). Currently, there have been some proposed DR classifier models but there is a lot of room to improve in terms of efficiency and accuracy. Deep learning has shown some promising outcomes which has made it the to go choice for image feature extraction and classification in the medical domain(Quellec, Charrière, Boudi, Cochener, & Lamard, 2017). Despite having strong computational power, current deep learning algorithm is not able to gain the trust of the medical experts in classifying DR(Pang et al., 2018).

In the field of medical image analysis deep machine learning is playing a vital role (Quellec et al., 2017). Some research articles suggest that the detection of Diabetic Retinopathy can be made by applying the CNN (Convolutional Neural Network) and image fusion combined (Liu, Guo, Georgiou, & Lew, 2018). The use of the CNN model is recognized as a better alternative to the conventional methods used for visual learning (Girshick, Donahue, Darrell, & Malik, 2014).

In this project we investigated the possibility of classifying DR using deep learning with CNN and image fusion as the classifying tool. The staggering number of people suffering from DR has

encouraged us to work on this project of classification of DR. Moreover, during the preparation for the literature review we came to realize that in the current time the machine learning algorithms can play a significant role in classifying DR. Second, not that much work has been done previously on DR. Recently, the researchers are focusing on the task of classifying DR. The main goal for our project is to put some light on the importance of using machine learning to classify DR. This claim is backed up by an implementation of the classifier model for DR.

In the first stage of the implementation we prepare for the preprocessing of the retinal images so that we can applying the learning model for building the classifier model using deep learning. Then we focused on the extraction of features from the retinal images by abiding some of the key features noted relating to extraction and learning of exudates from the training images followed by the principals of the features used in classification. In the process of implementation of the proposed classifier model we will considered the problem statements stated by the researchers and provided a solution to those problems.

Background

(Quellec et al., 2017) implemented a CNN classifier model to detect the lesion in the retina for DR prediction. The researchers also proposed the use of heat maps to eliminate the artifacts which are being included during feature extraction. (Quellec et al., 2017) claim that their implementation of tweaking the configuration of CNN with pixel level supervision produced a better heatmap compared to the general algorithms used for generating heat map. While testing the model in the Kaggle data set it received an impressive AUC of 0.954 after training the model with 108,000 images. However, the validity of the model is questioned when we consider the fact that the model was created with a testing size of 89 images from DIARETDB1 dataset.

Data set normalization was preferred by (Pratt, Coenen, Broadbent, Harding, & Zheng, 2016) before moving forward with any form of classification. This method in preprocessing is quite common when classifying models in the medical domain is considered. Since in the medical domain the training images are more likely to have poor resolution and various color intensities. CNN has been proven as a best approach to classify model using deep learning (Pratt et al., 2016) has proposed to implement a CNN model along with fine tuning. Fine tuning relates to tweaking with the parameters of the CNN model to improve its efficiency. The first layer of the CNN is given the task to learn the edges of the training images; whereas, the last layer is being used to learn the features contained in the training images such as exudates. In the middle sub portion increased convolution layer learns the deeper features these layers are grouped together to form a convolution block. Basically, (Pratt et al., 2016) is trying to divide a complex CNN architecture into three sections and assigning each section an individual task of extracting features from the training image. This is an efficient approach as training 3 separate CNN to do the same task will require a lot of computational power and time. After the partition several of these blocks are grouped together to result into several convolution blocks. Batch normalization with max pooling is being applied to each of the convolution block. CNN with a smaller training data size have a high tendency overfit the model. In order, to prevent the model from overfitting, the convolution block is flattened to one-dimension. The final convolution layer is then

associated with dropout dense layer and rectified linear unit in the end. The research article(Pratt et al., 2016) gave importance to pre-processing rather than carrying out building the classifier with the raw training images for DR. They addressed the shortcomings of raw data set and proposed normalization to be the solution to counter attack the drawback. Further, (Pratt et al., 2016) addressed the possibility of over fitting the model due to the presence of skewed data sets which is common in the medical domain training images. Implementing real-time class weights in the CNN with back propagation can be considered as the solution(Pratt et al., 2016). The results from the trained model states that the model achieved an accuracy of 75%, specificity- 95% and sensitivity- 30% (Pratt et al., 2016). Achieving a sensitivity of 30% and specificity of 95% suggests that the model proposed is biased towards classifying the output as not having DR. This means even after considering the sewed test data set their applied procedure was not able to overcome the drawback. Moreover, 10% of images in their training datasets was not gradable by the professional.

Fine tuning was also used by (al., 2016) to build a CNN classifier model. In this paper the researchers went with a different approach to fine tuning rather than going with the common practices as in (Pratt et al., 2016). The common implementation of fine tuning CNN which only creates features from the training images and some fine tunes all of the layers of CNN(al., 2016). AlexNet (Gao, 2017)was used as their preferred of CNN. They generated a set of images by applying data augmentation which will be used for fine tuning the CNN classifier model. All the training images were given as an input to the trained CNN. The probabilistic output for the test exudates was averaged and FROC was as a measure of performance(al., 2016). The authors claimed that by incorporating the CNN model with handcrafted features will output an improved classifier model(al., 2016). The generated result shows that fine-tuned CNN had a slight better performance compared to the handcrafted features however fine tuning all the layers of CNN can improve the accuracy significantly(al., 2016). In this article, there was no specific method about the classification of DR. This article mainly focuses on the prediction difference between fine-tuning CNN compared to pretrained CNN and gave a brief idea about the level of accuracy using handcrafted features. Although the images for training the model was not used from a renowned data banks for medical images. They used a single video and used it to source the images for training the classifier this made us less confident in acknowledging their work as a better implementation to classify DR.

To obtain a good prediction, (Pang et al., 2018) planned in using two fundus images of each patient. Hence, this required them to implement feature fusion for two eyes. While creating the classifier model for DR (Pang et al., 2018) constructed a CNN model with two levels. Inception-V3 which is an implementation of a CNN architecture is used for feature extraction and VGG model. Final prediction was derived using both of these two procedures. Thus, the model was given two input images at a time for more accurate classification. The last convolution output was used for visualization. After building the model, the authors created a series of process which will guide the medical professionals for an easy access to the services provided by the model. Even though, models like CNN and deep MIL have been used in the model while building the classifier model for DR; (Pang et al., 2018) did not mention any specification of their model which we could take note of. In addition, they did not mention the data set being used not did they reported any form of pre-processing steps before implementing the model.

In a research article (Bravo & Arbeláez, 2017) approached on DR classification. Unlike (Zhou et al., 2018) and (Mansour, 2018) where both of them preferred to use AlexNet as their choice of CNN architecture (Bravo & Arbeláez, 2017) preferred to use VGG16 as the CNN architecture. The main goal of their proposed model was to detect DR in its early stage. Use of VGG16 as a classification architecture had an advantage over AlexNet which is VGG16 was designed for the implementation of large scale natural image classification. As a form of pre-processing the training data set they went with removing the background of the images which was not mentioned in any of the papers we reviewed and as a compromise they did not consider applying and form of filtration to eliminate noise in the training data set. This taught us how can we deal with smaller data set and come up with a better classification model. The researchers went with four different CNN models rather than sticking with a single model.

In image fusion the combination of modalities used to classify a model depends on the presence of noise in the training data set. There is direct relationship between the performance of classifier model and the quality of the data set in the model. The performance of a modality in medical domain is dependent on the structure of the organ and the tissues(James & Dasarathy, 2014). The authors gave an overview of the application of image fusion in the medical domain and a brief idea about how image fusion can be applied in the medical domain. Since it was a review article there were no specific proposed methodology for applying image fusion.

The application for using image fusion on the test images allows to improve the strength of the prediction of each individual layer and result into an improved prediction(Liu et al., 2018). As a preprocessing stage for the convolution fusion network they followed the conventional pre-processing strategy. In this research paper, (Liu et al., 2018) proposed a new fusion architecture from the ground up which combines the intermediate layers into changing weights. It is suggested that using early fusion accompanied by late prediction can be able to provide a high-resolution interpretation and decreases the number of parameters; thus, reducing the computational complexity. Locally connected layer can be implemented as a fusion module which can learn the weights of each features and produce a better representation. The first step of forming a CFN (Convolution Fusion Network) is to fuse the multi- level features in CNN which is then merged with several locally connected modules. The fusion results into a fusion module which can address image level classification; after that more training is carried out using the training images called the trained CFN module. There are three sub sections in the trained CFN module scene recognition, fine grained recognition and image retrieval. More development is being carried out for each of these dimensions until they reach a certain amount of accuracy. The final resulting model is called the CFN(Liu et al., 2018). This proposed methodology is claimed to be applicable for different visual recognition tasks (Liu et al., 2018). Using input specific weights can reduce the variance between images. This is the one of the first research paper to use locally connected layer as a fusion module. They proved that incorporating CNN with image fusion can output better result compared to CNN.

Methodology

The methodology to construct a perfect diabetic retinopathy classifier is considered to be one of the most critical aspect in current times. Several methodologies have been proposed during the past few years, yet no such methodology has proven to reach an accuracy over 82%. We illustrated the ideology of pre-processing the images similar to the other proposed ideologies. However, the approach to making a prediction over a certain retinal image is slightly different. The following subparts illustrates an overview of the implementation techniques and ideologies used to classify a diabetic retinopathy retinal image, how the issue of the data was handled and what is the outcome.

Data Split

The training and testing data provided was split before the model was trained. It is an industry norm to split the data set into a 70/ 30 split. 30% split of the data is kept for validating the data set after the model has been built. Rest 70% of the data is used for training the model.

While training the model, the training dataset was further split to 70/ 30 split. 30% of the training dataset was used for validating the learning process of the model. Exactly after this point is where we assured that data augmentation was a key aspect to achieve higher accuracy.

Image Pre-processing and Data Augmentation

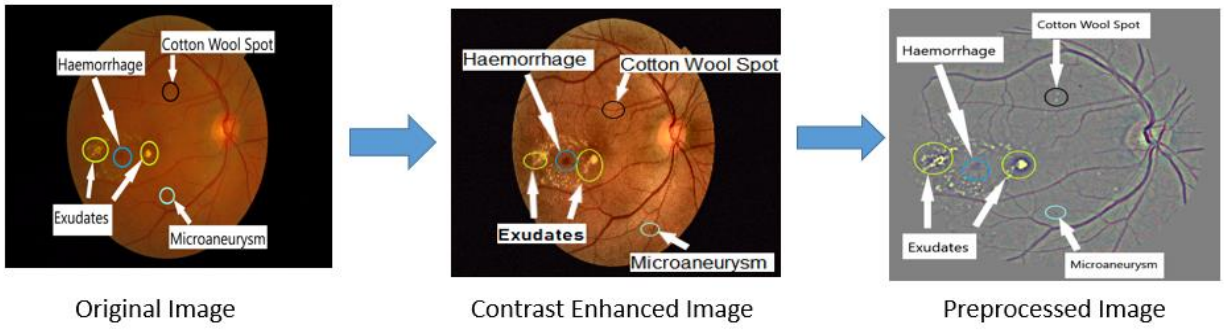


Fig:2 Preprocessing stages

After the split was made, while analyzing the dataset, the first thing we noticed is that the images provided does not have equal distribution for the classes. This means that we are required to provide a solution to this issue or else the model which we will be creating might be biased. Second, the total number of images in our training set was only 650. During the stage of crating

the literature review it was mentioned in several research papers that the size of the dataset for machine learning play a vital role in creating a reliable machine learning model.

To overcome the first obstacle of unbalanced images for training class we implemented batch normalization which was carried out by taking equal number of images for all of the classes in the training set for the classifier. This step for preprocessing will ensure that there is less chance for the classifier model to be biased towards a certain class.

In the next stage we figured out a solution for the small dataset size. Since, it would be very challenging to a significant model for classifying with a small dataset we had to come up with a solution which will deal with the constraint of small dataset size. The selected solution was to implement dataset augmentation. Augmenting a dataset is a common technique to deal with a small dataset. Almost all of the research papers we have read across during the literature review has provided some form of dataset augmentation. During the implementation we have carried out several augmentation techniques which includes rotation of the images into several degrees; 180 and 270 degrees.

The images were divided into three categories, where each category represents different variations of diabetic retinopathy. So, the most important stage before creating the classifier is to traverse the images through a series of some preprocessing techniques. The raw images provided were of very poor quality and so, configuring out a perfect processing technique which can fix the issue was strenuous. Images varied in resolutions, illuminations and contrast. A normalizing technique was performed which lead all the images to have the same resolution. Here, the images were resized to 512 pixels and cropped based on the radius of the retinal image which allowed the images to have the same radius of the field of view. In order to fix the illumination and contrast, we applied a robust contrast enhancement technique known as the histogram equalization which enhances the contrast of the image. This enhanced valuable features such as microaneurysms, exudates and thick blood vessels which are initiations of diabetic retinopathy. Furthermore, using the enhanced contrast resized image we applied Gaussian smoothing kernel, with standard deviation in order to estimate the background illumination. This technique transformed the image in a way such that the obvious features such as the color of the retina was made translucent. Hence, only the important features such as the blood vessels, microaneurysms and the exudates were immensely visible. In contrary, we just made it easier for the model to learn as it is only going to learn the factors we want the model to learn. Lastly, using the contour technique the outer circular border was removed. The rightmost picture in Fig.2 displays the final pre-processed image of three different classes.

CNN Architecture

Several architectures have already been tested for DR classification. However, the difference lies in how the pretrained model is fine-tuned which best fits the data specified. The most commonly used CNN used so far is the VGG16 architecture which was originally designed to classify 1000 different classes. It consists of thirteen convolution layers coupled with ReLu. Hence, we used

VGG16 as our prior model and alongside we used InceptionV3 which has a much more deeper architecture than VGG16. They use parallel processing blocks in the layers to improve its performance. The weights however are pre-set as ImageNet and thus for no complications in the model we used the weights of ImageNet.

The learning rate for Inception however was very critical to fix. Hence, we took the advantageous functionality of Keras, which allows us to use the Stochastic Gradient Descent (SGD) optimizer which changes the learning rate based on what it is learning during each epoch. To train the network we used a multinomial logistic loss function and implemented it in the same layer as the softmax classifier. The combination of the softmax layer and the log-loss is useful for numerical stability. The training was done on a virtual supercomputer constituting of 32GB RAM including a graphics processor of Intel(R) Xeon(R) of 2.20GHz.

InceptionV3 as known to be can perform better with images of size 299. Hence, when loading the images into the model we resize the image to 299. The model was tested for at most 60 epochs and a graph was plotted based on the training and validation accuracy for every epoch to analyze the training phase of the model. In order to reduce overfitting, dropout layers were added after each fully connected layer and lastly the functionality for the model to predict only 3 classes was initialized after the fully connected layer. Moreover, while tuning the model, some of the layers were frozen as those layers learn the basic features of an image such as shapes and objects. Hence, there was no specific requirement in tuning the first few layers of the model.

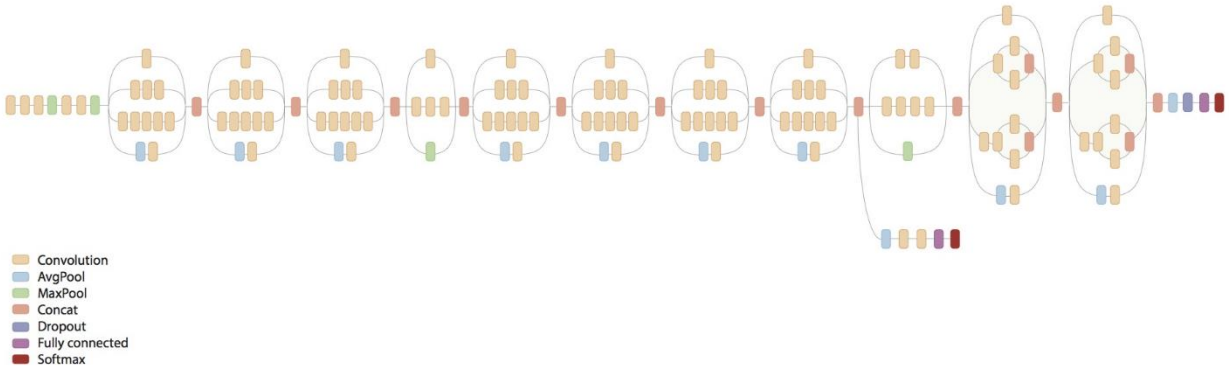


Fig3 - InceptionV3 (Shai, 24 July 2018)

VGG16 however does not have such issues in tuning as the internal architecture of the model is not as deep as InceptionV3. So, it is less likely to learn the complete deep features of the retinal image where we have to detect several features such as microaneurysms, hemorrhages and internal blood clots. Similar to InceptionV3, we provided same weights to VGG16 as initially this model was also trained using ImageNet weights. In contrary, one flatten layer was used which generalizes the input neurons into one specific shape. Followingly, two dense layers were used with filter size 4096 alongside with softmax function as the activation function and two

dropout layers of 0.2; 20% of the neurons which have no contribution are dropped after training. Furthermore, during compilation, once again SGD optimizer was used as it is an incremental gradient descent which uses an iterative approach for optimizing a differentiative objective function. Individual optimizers tend to have different learning rate computations. However, SGD seems to outperform the other well-known optimizers such as ADAM and RMSPROP.

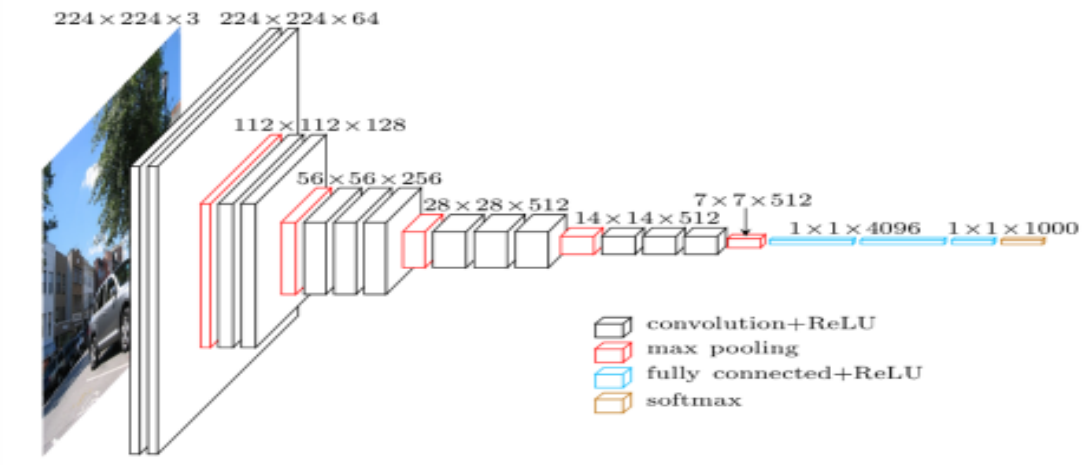


Fig4 – VGG16 Architecture ("VGG in Tensor Flow," 17 June 2016)

Insufficient data plays a vital role in the training phase of the model. No matter how detailed the images are or how many classes we have, due to a compact dataset the model cannot learn as many features as possible and hence there is an intensive requirement for data augmentation. Due to such compact data we decided to use 32 batches of images for each epoch during training. In a sense, the training batch size depends on the number of total training images. Therefore, the training batch size should be a factor of the total training images so that the batches are evenly divided among every epoch. Similar to this concept the validation batch size and the test batch size was set using the total number of images.

As two models are being used for the classification of DR, we have two sets of predictions; one set for each model. But the final prediction has to base on one single set of predictions and in order to get one single set of predictions we introduced a fusion approach where we fuse two predicted values and generate one single prediction for each class. This concept was taken from a technique known as fusion technique. Overall, we base our test accuracy on the final fused predicted values. This fusion was done via the concept of averaging technique which takes the average of the two sets of predictions for every image. However, this method does have limitations as well. One limitation can be the fact that if one classifier generates a very poor prediction and the other predicts a quite well prediction, then the overall prediction can turn out to be low and eventually lead to lower accuracy. However, in our case we do not aim for overall accuracy but aim for how much well the model predicts what class the DR image belongs to.

Validation

While building the model we need to ensure that the model was able to learn proper features of the model. This was used as the sanity check for the learning process for the creating the classifier model. As mentioned above there are several standard ways for building a DR model. In terms of our case, the selection for the validation process for building the model was mainly centralized on the validation processes which can be visualized. This includes training vs validation graph, training vs validation loss and confusion matrix. All the mentioned methods for validation helped us to make sure that after each change made to the model, the model was able to detect the features of the DR images appropriately with minimal loss.

Project Management and Process

Since this project was a team contribution we had to implement the general principles of the PIMBOK guide to ensure that the project was being carried out in a professional way. At every stage of the project we tried to implement the objectives of project management we have learned in the semester 1 of project management.

The project was managed on the basis of a log where we noted the progress of the project and noted the improvements made to the project during the entire project continuation period. In the log file we mentioned the challenges which emerged during the development of the classifier model. The progress of the project was closely monitored in conjunction with the gunch chart developed in the first semester of the final year project. The gunch chart helped us to closely monitor the progress of the project. This means that every week we looked up to the schedule created in the previous and semester and this made us understand whether we are on schedule or not. In some weeks we found out that we missed some of the deadlines according to the schedule. To catch up with the schedule we escalated the priority for the project by allocating more amount of time to the project and this resulted in catching up with the deadline. During the continuation of the project we have faced some complication where we had to fix some bugs in the code or some sort of brain storming was required to get the break though. In the process of brain storming we booked a meeting room where we used a marker and wrote on the glass with a marker to make sure that the ideas which were being proposed were noted down to give us some time to look back and this. This methodology worked well for our project. We have noticed that after generating the ideas breakthrough came very quickly.

All of the aforementioned details were carefully being jotted down in the progress log. In case of the source code for building the classifier model we used git hub as the form of version control. In addition the git hub platform allowed us to work on the model seamlessly as both us were able to work on the model simultaneously and at the same time we were able to see what were the changes being made by the team mate and if the code broke at any point in the development

point we could easily revert back to the place where we stared. Moreover, we were able to access the codes regardless of our location.

The project involved supervised machine learning which required classified training mages for training the model. The set of training images was provided by our supervisor. Then we moved on to the step of training the model which required access to a powerful computer with GPU support for faster learning process.

During the entire project continuation project, we came across some unforeseeable events which ranged from not being able to install the packages we wanted to install, certain bug in the program which was preventing the model learning process, preprocess stages did not improve the accuracy of the model, to the model going to an infinite loop during execution. These unforeseeable events were handled with patience and good coordination within the team. Before we started troubleshooting the problem we backup the codes in a repository so that we can revert all the changes if the fix did not work or completely broke the code. The bugs in the codes were fixed by using reverse engineering and carefully tracing each line of the code.

However, there were some limitations which were not within our control. First, each two teams were handed over a single account to the virtual machine provided by ITS. This means we had to share the time slot accessible to the virtual machine with another team which required coordination with another team. This required us to negotiate over the feasible time for accessing the machine. To make matter worse we had to work within the Monash premises as the access to the virtual machine was only possible within the computers in the labs of Monash campus which were accessible until a certain period and in the weekends the accessible time were much shorter. This constraint of time limit made working with a piece of mind much challenging. Since, training the model requires a peace of mind.

Outcomes

Among the two models we implemented, InceptionV3 which is under the domain of CNN architecture seemed to perform better. After multiple tuning and testing we achieved an average training accuracy of 73.5 % with a validation accuracy of 68.7%. However, the highest we have achieved for training is 79%. In the case of VGG16 model, we were able to achieve an average training accuracy of 72.4% and validation accuracy of 62.5%. Highest ever reached training accuracy for VGG16 was 83.2%. By observing the accuracy graph for InceptionV3, we found out that there was a minor sign of overfitting which is an achievement on its own since with a small data set there was a high chance for the model to overfit. In addition, when running the model several times for further clarification about the reliability of the model we noticed that the variance of the results for the accuracy was less than 2%. Another metric to consider for analyzing the performance of the model is validation loss. In our implementation the validation loss was less than 1.05% for InceptionV3 and on the other hand the variance for validation accuracy for VGG16 is about 5.0%. We found an average of 72.21% training accuracy for InceptionV3 and a validation accuracy of 68.13%. From the following result we can infer that the model is overfitting. However, reducing this overfit is very strenuous in a sense that we do

not have full access to Keras application for VGG16 and hence finetuning the model in depth is very difficult.

Considering the two validation accuracies we can comment that due to the deeper architecture of InceptionV3, the model outperformed VGG16 and further tweaking can lead to higher accuracy. However, it is a fact that VGG16 requires a tremendous amount of data to perform well whereas Inception can perform better than VGG16 with a much less dataset.

Overall after fusing the model's predictions, we achieved an accuracy of 66.6% on 52 images from 3 different classes. Fig.5 and Fig.6 shows the learning curve of InceptionV3 and Fig.7 and Fig.8 displays the learning curve of VGG16.

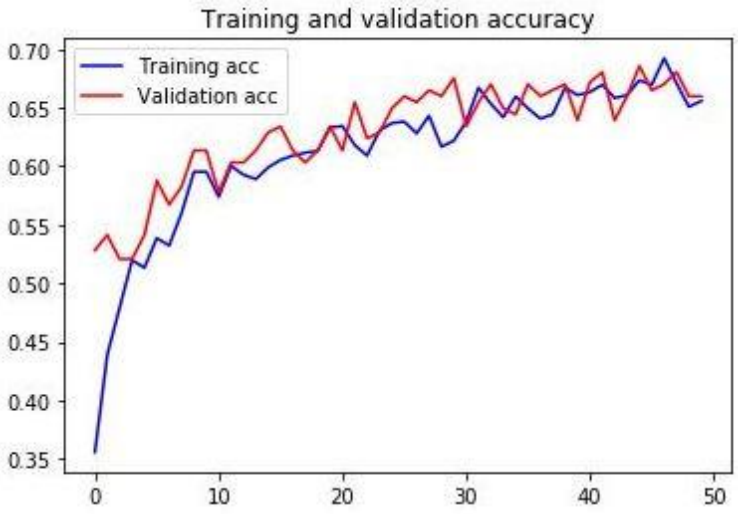


Fig.5 – Training accuracy Vs. Validation accuracy

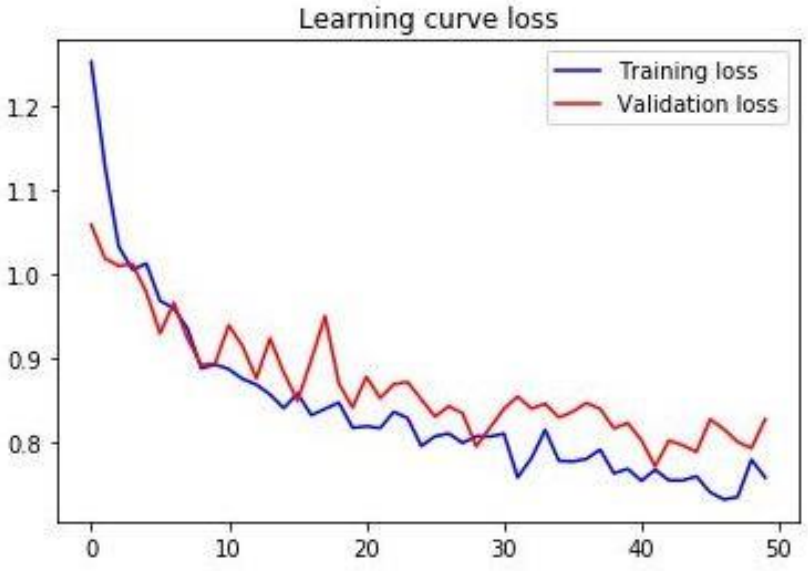


Fig.6 – Training loss Vs. Validation loss

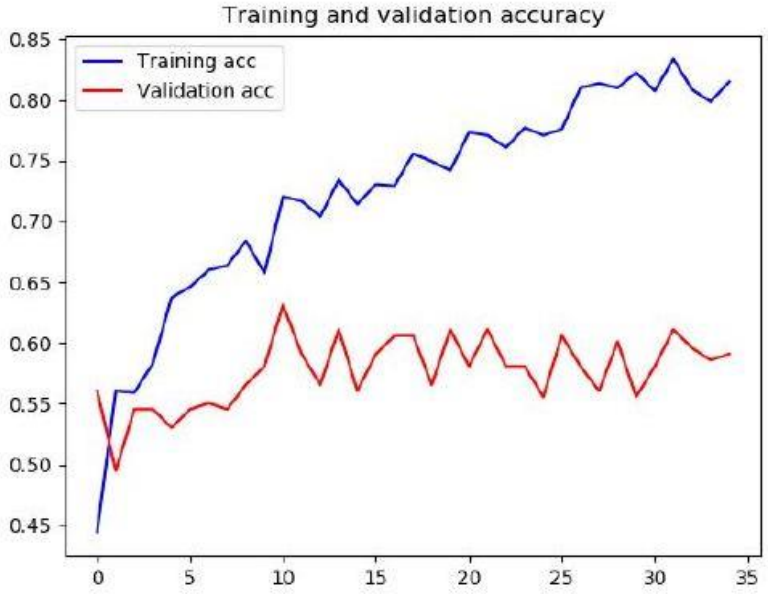


Fig.7 – Training loss Vs. Validation loss

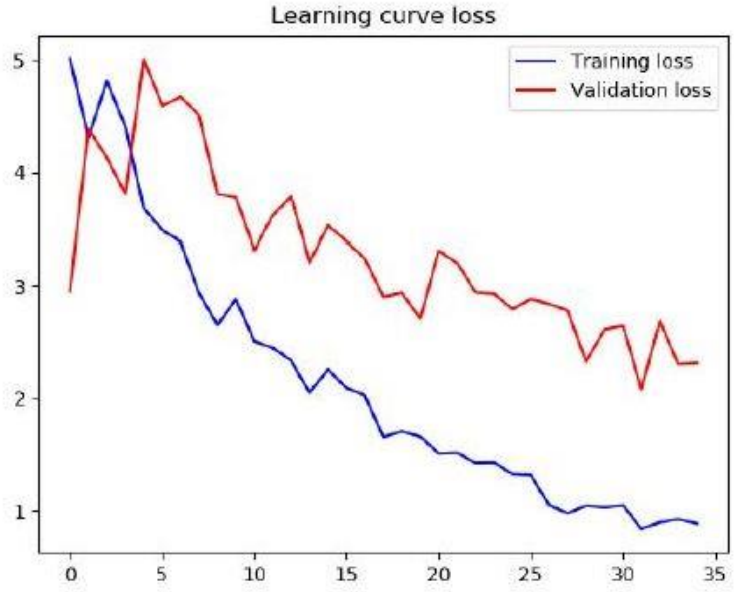


Fig.6 – Training loss Vs. Validation loss

Limitations

In current times DR can be classified into many different stages. However, we decided to categorize our data into 3 parts as our data is very limited and the classification of several classes can ultimately lead us to lower accuracy as the model then has less learning data compared to validation. In contrast, the dataset was densely imbalanced which caused a problem to split the data. This prevented us in achieving a much higher accuracy. Furthermore, from the point of view of an IT student, it is very difficult for us to clean the data as we do not have the complete knowledge of what exact symptoms can be defined as a DR. Still with lack of complete knowledge, many images had the necessity to be transferred from one class to another as many images belonged to the inappropriate class. The judgment for such transfer was made using the basic knowledge taken from Google and literature review of the research papers. Another such limitation which can be pointed out is the access to the supercomputers. The access to the supercomputers was given very late at time. Hence, the run time of the program was very slow which reduced the efficiency of working. One more limitation is the use of Keras libraries and Anaconda which was very difficult to install. Moreover, due to network errors and restrictions many libraries were difficult to install and use. Alternatively, issues were faced when running the model as sometimes the program crashed due to overloading which created a very big problem.

Conclusion

There is no doubt that deep learning has the capability to classify DR. In the medical domain it is very common to get a skewed data set for train the classifier; this is accompanied with variable resolution, brightness and contrast level. Common filtering approaches were – normalization of the data set, weighted sampling, eliminating skewed classifier. After going through all the research paper and considering several modalities it can be concluded that using a CNN classifier with fine tuning and the incorporation of image fusion with the classifier is more like to output a better classifier model for DR. The classifier model was built on the basis of standard industrial procedures which made the model up to the standard of professional implementation. All of the features added to the model was based on the literature review which was made based on the state of the art in the first place. The main goal of continuing this project was to show the importance of detecting DR in the primary stage and that machine learning can play a significant role in providing a solution to detecting the presence of DR in an early stage thus providing the medical experts valuable time to cure the disease. This is just a pinnacle of how machine learning can classify DR. There is a scope of improvement to our existing model these scopes were left to explored due to the time constraint. After the final semester we have decided to keep working on this classifier model and validate it with a larger data set which will allow the model more sensitive to classifying the classes of DR.

References

al., N. T. e. (2016). Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning? *EEE Transactions on Medical Imaging, vol. 35*(no. 5), pp. 1299-1312. doi:10.1109

Bravo, M. A., & Arbeláez, P. A. (2017). *Automatic diabetic retinopathy classification.* Paper presented at the 13th International Symposium on Medical Information Processing and Analysis.

Gao, H. (2017). A Walk-through of AlexNet. Retrieved from https://medium.com/@smallfishbigsea/a-walk-through-of-alexnet-6cbd137a5637

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014, 23-28 June 2014). *Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation.* Paper presented at the 2014 IEEE Conference on Computer Vision and Pattern Recognition.

James, A. P., & Dasarathy, B. V. (2014). Medical image fusion: A survey of the state of the art. *Information Fusion, 19*, 4-19. doi:https://doi.org/10.1016/j.inffus.2013.12.002

Liu, Y., Guo, Y., Georgiou, T., & Lew, M. S. (2018). Fusion that matters: convolutional fusion networks for visual recognition. *Multimedia Tools and Applications*. doi:10.1007/s11042-018-5691-4

Mansour, R. F. (2018). Deep-learning-based automatic computer-aided diagnosis system for diabetic retinopathy. *Biomedical Engineering Letters, 8*(1), 41-57. doi:10.1007/s13534-017-0047-y

Pang, H., Luo, C., & Wang, C. (2018). Improvement of the Application of Diabetic Retinopathy Detection Model. *Wireless Personal Communications*. doi:10.1007/s11277-018-5465-3

Pratt, H., Coenen, F., Broadbent, D. M., Harding, S. P., & Zheng, Y. (2016). Convolutional Neural Networks for Diabetic Retinopathy. *Procedia Computer Science, 90*, 200-205. doi:https://doi.org/10.1016/j.procs.2016.07.014

Quellec, G., Charrière, K., Boudi, Y., Cochener, B., & Lamard, M. (2017). Deep image mining for diabetic retinopathy screening. *Medical Image Analysis, 39*, 178-193. doi:https://doi.org/10.1016/j.media.2017.04.012

Shai. (24 July 2018). How do I interpret this Inceptionv3 model graph.

VGG in Tensor Flow. (17 June 2016).

Zhou, L., Zhao, Y., Yang, J., Yu, Q., & Xu, X. (2018). Deep multiple instance learning for automatic detection of diabetic retinopathy in retinal images. *IET Image Processing, 12*(4), 563-571. Retrieved from http://digital-library.theiet.org/content/journals/10.1049/iet-ipr.2017.0636