MAIA

RETINAL LESION SEGMENTATION
Advanced Image Analysis (AIA)
Machine and Deep Learning (ML/DL)

Alexandra Albu (alexandra.albu@studentmail.unicas.it)
Anwai Archit (anwai.archita@studentmail.unicas.it)
Ebaneo Enrique Valdez Kao (ebaneoenrique.valdezkao@studentmail.unicas.it)
Sheikh Adilina (sheikh.adilina@studentmail.unicas.it)
Università degli Studi di Cassino e del Lazio Meridionale

1 Introduction

Diabetic Retinopathy (DR) [1] is a serious microvascular disorder that results in loss of vision and is the leading cause of blindness. It seriously damages and reduces the light-sensitive inner layer of the eye. The manual inspection of Retinal Fundus Images on DR to detect the morphological abnormalities in Microaneurysms (MAs), Haemorrhages (HEs), Hard Exudates (EXs) and Soft Exudates (SEs) is very difficult and time-consuming process. In order to avoid this, the regular follow-up screening process and early Automatic Computer-Aided DR Diagnosis is necessary. Direct segmentation techniques [9] generate unsatisfactory results in most cases. This is because of the fact that the texture of unhealthy areas such as DR is not homogeneous. We propose a complete lesion segmentation system as Multi-Class Hierarchical Classification Algorithm for the Image Processing and Machine Learning pipeline and Lesion-Based Segmentation for the Deep Learning Pipeline. We sincerely hope the presented work with requisite developments to come will be helpful for the radiologists and researchers who want to focus on enhancing the diagnosis of a powerful system in real life.

Figure 1: Retinal Fundus Image

2 Materials Used

We have been provided with the IDRiD Challenge Dataset which contains 81 images (54 for training and 27 for testing). The dataset also contains the ground truths of optic disc as well as the 4 types of retinal lesions. For these project, we only used the data for the lesions: Soft Exudates, Hard Exudates, Haemorrhages and Microaneurysms.

3 Image Processing

The Image Processing portion of the **Retinal Lesion Segmentation** was not counter-intuitive right from the inception. Suggestions have been inherited throughout the period; the pre-processing stage for the Machine Learning pipeline saw relentless developments. Right from the very beginning, our motive of obtaining probable candidate patches shifted from precise segmentation to encouraging maximum possible True Positives within a lot of available candidates. The core idea of the image processing reflects the building of a reasonable set of candidates with a requisite trade-off between High Sensitivity and Candidates.

➔ *Why High Sensitivity?* – For Lesion Detection/Segmentation associated with medical data, our considerations should be to never miss a True Positive (TP). Missing TPs might risk life and death situations. The higher the sensitivity, the more suitable the lesion segmentation.

For instance, the binary thresholding of the original image with [0,255] would fetch us 100% sensitivity, but would it be good for the Machine Learning (with respect to obtaining candidates)? We are herewith mentioning the challenges to address whilst the Image Processing pipeline:

- Structural Complexity of Lesions
- Lesion Detection amidst Tessellated Images and Noisy Images (Reflections - Optical and Bright Borders, Impulsive Noise)
- Strong Inter-Class Similarity (Red/Dark Lesions: MA-HE and Bright/Yellow Lesions: SE-EX)
- Non-Lesion Appearances (Deposits like Drusen, Vessel and Nerve Fiber Reflections)

The trivial conception of the framework through handcrafted features based multi-staged approach typically involves a pre-processing stage for image enhancement and/or image smoothing, image segmentation, feature extraction (for DR lesion detection) and multi-class classification. Our majority attempts of classical image processing techniques involved for candidate's generation are mathematical morphology, template matching, entropy thresholding, adaptive thresholding, etc. Henceforth, using (IDRiD: Diabetic Retinopathy – Segmentation and Grading Challenge – Literature) as the preamble, we came up step-by-step with our algorithm.

- The environment used for Image Processing is Google Colaboratory. The first step begins with importing the requisite libraries and leaving this section open to additions. Simultaneously, we import the data folders of the training and test images and respective ground truths (we perform the *b, g, r split* and operate on the 'Green Channel'). We create a function *evaluation()* to monitor the sensitivity of the candidates and another function *imshow()* to visualise our intermediate processing. After autonomous processing of all dark and bright lesions, we try to avoid over/under processing by keeping the Sensitivity vs. Candidates trade-off in our mind.
- After the optimal set of candidates are obtained, we match the detection and labels, and evaluate the Sensitivity ($Recall = \frac{TP}{TP+FN}$). Furthermore, we save the images to pace with the next stage.
- *Why Green Channel?* – The retinal images are usually low contrast images. Lesions are visible in the green channel due to high contrast. Hence, we adopt extraction of green channel for all lesions. A sample of green channel with dominant lesions is shown in Figure 2

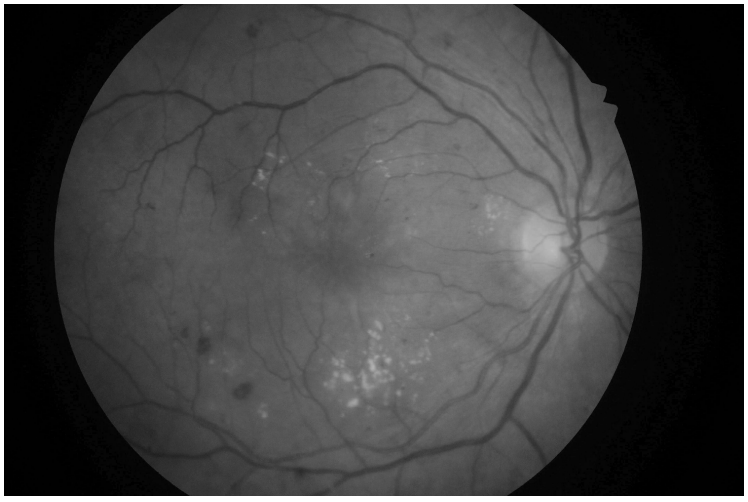


Figure 2: Green Channel with Dominant Lesions

3.1 Microaneurysms (MAs)

We begin with reading the ground truth and the fundus images. As discussed, the green channel is applied Adaptive Histogram Equalization (CLAHE) after resizing. We use the Canny Edge Detection and henceforth an additional smoothing with the Gaussian Blur. To conclude with this stage, we apply Morphological Opening Operation (Erosion and then Dilation) and hence subtract artefacts from the MA candidates. We resize our candidate image to the original dimension and evaluate (aiming at higher sensitivity) the saved candidates. A sample of the segmentation is shown in Figure 3 and the whole process is illustrated in Figure 7a.

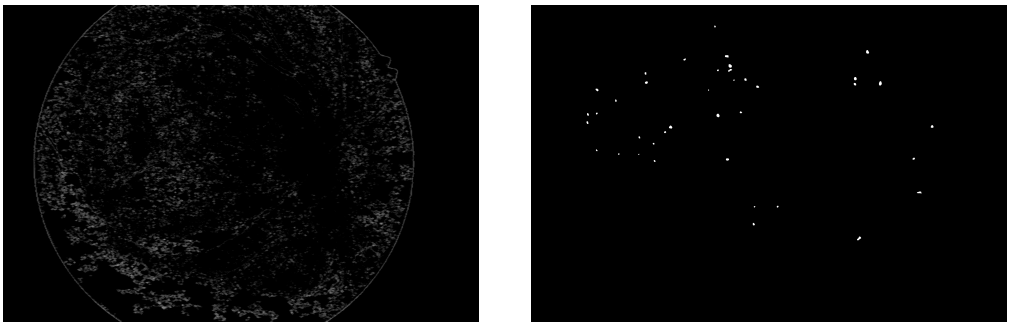


Figure 3: Patient 17 (Left: Segmented Candidate Patches, Right: Ground Truth – MAs)

3.2 Haemorrhages (HEs)

Our first step again starts with reading the images and respective ground truths. We handle the exception of absence of HEs in Patient 43. We apply Adaptive Histogram Equalization (CLAHE) on the resized green channel. We apply a pipeline of median blur with ksize=81, smooth the regions, apply a morphological opening on a threshold section. We repeat the above pipeline with ksize of the median blur as 131. We use the bitwise operation in both pipelines, and hence resizing our candidate image to the original dimension, evaluate (aiming at higher sensitivity) the saved candidates. A sample of the segmentation is shown in Figure 4 and the whole process is illustrated in Figure 7b.

3.3 Hard Exudates (EXs)

Our processing begins with storing transformation to LAB colour space and applying erosion on the lightness channel (for, bright lesions) on the read masks and images. We use a pipeline on the green channel by applying median blur, Adaptive Histogram Equalization (CLAHE), subsequent tophat

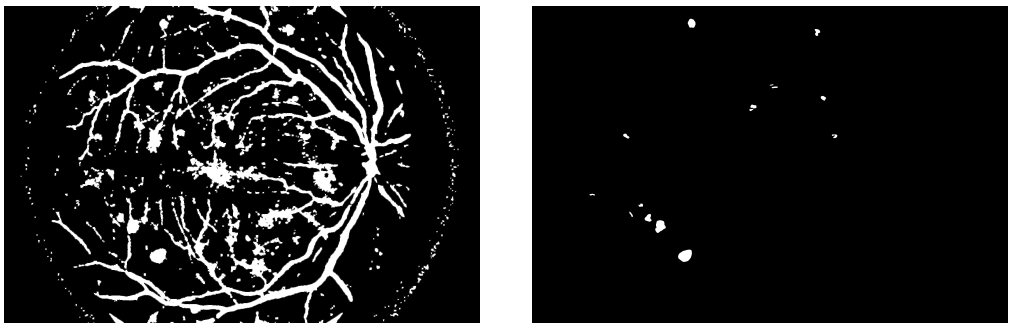


Figure 4: Patient 1 (Left: Segmented Candidate Patches, Right: Ground Truth – HEs)

and blackhat operations and threshold their respective subtraction. We repeat the above pipeline by halving the dimensions of the green channel. We perform the nearest neighbour resize on the immediate halved operated output with the scales of the penultimate pipeline, and conclude with bitwise operations on both the pipelines, and henceforth their output with the lightness channel output. We resize our candidate image to the original dimension and evaluate (aiming at higher sensitivity) the saved candidates. A sample of the segmentation is shown in Figure 5 and the whole process is illustrated in Figure 7c.

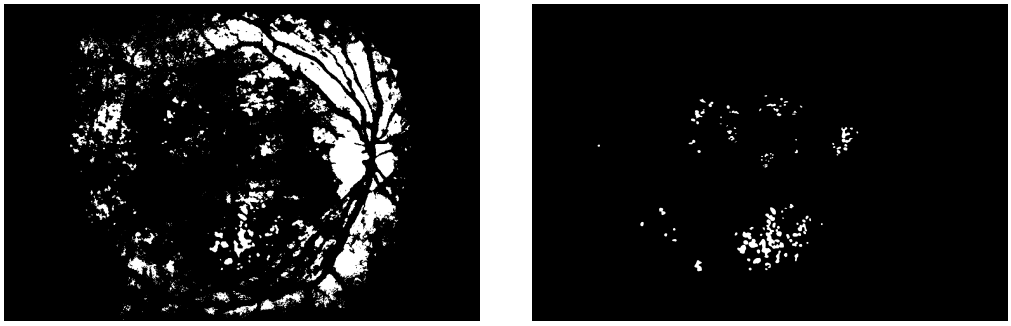


Figure 5: Patient 1 (Left: Segmented Candidate Patches, Right: Ground Truth – EXs)

3.4 Soft Exudates (SEs)

The cotton wool spots are quite an exception to be handled in more than half of the patients. We apply contrast stretching on the smoothened images. We proceed with apply gradient operations on both axes (Sobel Operators), perform morphological opening and subsequent distance transform to find an optimal threshold to the candidates. We repeat a morphological opening to remove noise. We resize our candidate image to the original dimension and evaluate (aiming at higher sensitivity) the saved candidates. A sample of the segmentation is shown in Figure 6 and the whole process is illustrated in Figure 7d.

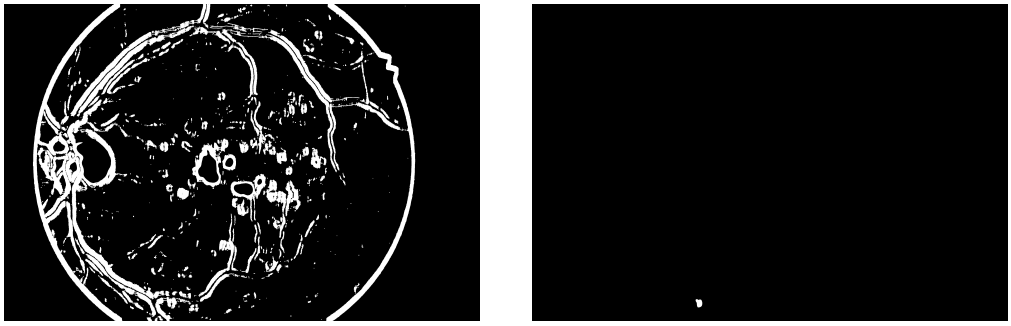
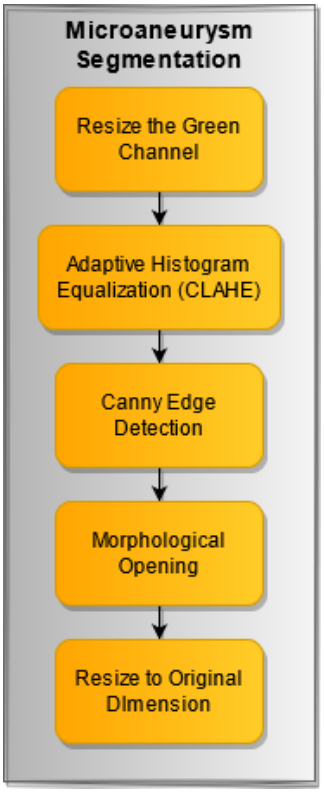
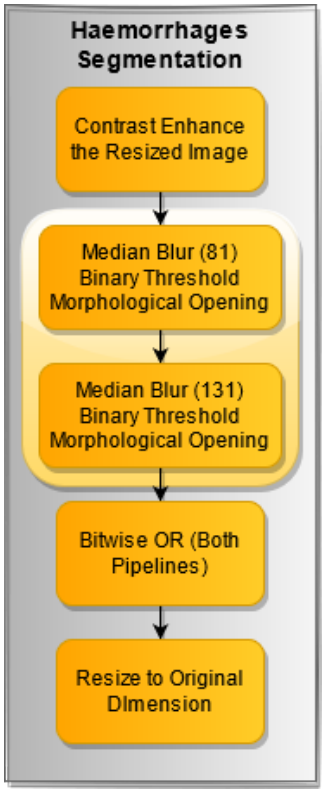


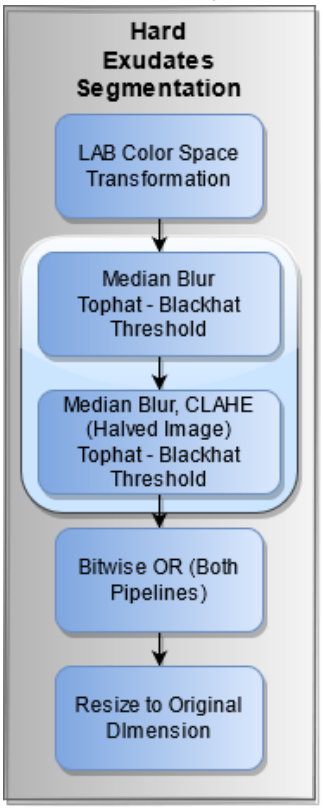
Figure 6: Patient 3 (Left: Segmented Candidate Patches, Right: Ground Truth – SEs)



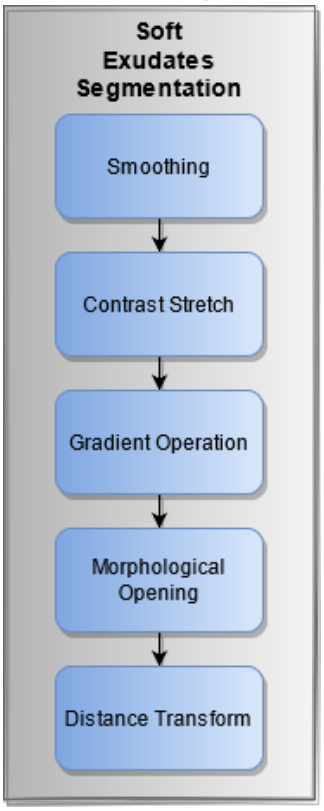
(a) MA Candidate Segmentation



(b) HE Candidate Segmentation



(c) EX Candidate Segmentation



(d) SE Candidate Segmentation

4 Machine Learning

The Machine Learning part of the project relies completely on the Image Processing part of the project. We already have the segmented images of the four lesions: Soft Exudates, Hard Exudates, Haemorrhages and Microaneurysms. In this phase, we used a hierarchical classification technique to achieve the multi-class classification of the DR Lesions. The overview of this section is given in Figure 8. In the first part, lesions are classified from non-lesions and sent over to the second part. In the second part, the multi-class classification of the four lesions is done.

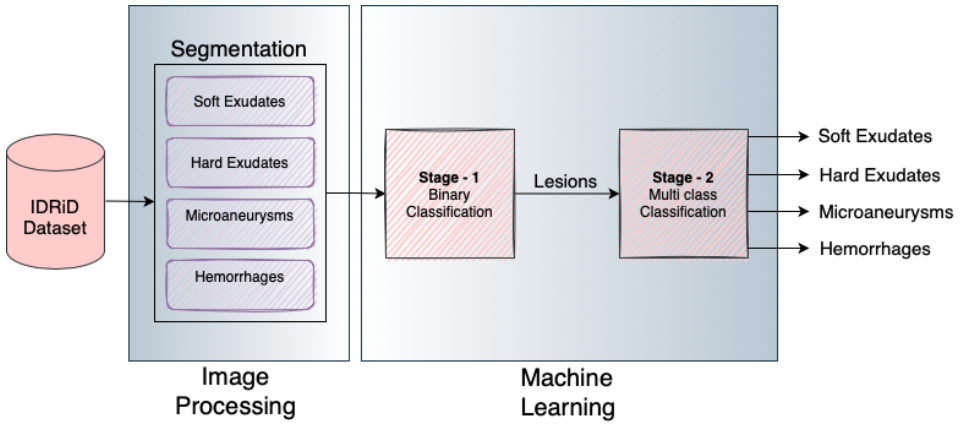


Figure 8: Overview of the Image Processing and Machine Learning pipeline

4.1 Preprocessing and Feature Extraction

Initially, we created a pixel based classification model where each pixel was labelled as one of the four lesions or as non lesion. This technique was computationally extremely expensive. In addition to that, this model can itself be an independent model and would not even require a Image Processing pipeline. So we shifted to the region based technique.

In the Region Based technique, for each image we obtained all the possible regions and cropped them as rectangle images and compared them with the same location in the ground truth. If the white pixels in the region matched by 30% in the ground truth, the region was labelled as a true lesion otherwise it was labelled as false positive. We experimented with various thresholds ranging from 30% to 70% and the performance of 30% matching was the best. For each of the four lesions a number was assigned (given in Table 1) and a label of 0 was given for the false positive regions. Example of a candidate region, its ground truth and the corresponding original image is shown inf Figure 9.

Feature Extraction was done from the green channel [4] of the fundus image as it contains the least noise and hence is the most informative among the 3 channels. Features like area, solidity, maximum intensity, minimum intensity and mean intensity were extracted. Then 48 GLCM features [8] [5] were obtained. GLCM contrast, energy, homogeneity and correlation were obtained at distances 1, 2 and 3 and at angles 0°, 45°, 90°and 135°. The features are summarized in Table 2.

Table 1: Label of Each Lesion

Lesions	**Label**
Soft Exudates	1
Hard Exudates	2
Microaneurysms	3
Haemorrhages	4

4.2 Stage 1 - Binary Classification

At the beginning of this stage, we already have a huge number of regions that are false positives (non lesions). Moreover, we don't have 100% of the actual lesion regions as we have lost a fraction of

Table 2: Types of Features Extracted

Serial	Feature	Total Number
1	Area	1
2	Solidity	1
3	Maximum Intensity	1
4	Minimum Intensity	1
5	Mean Intensity	1
6	GLCM Contrast	12
7	GLCM Energy	12
8	GLCM Homogenity	12
9	GLCM Correlation	12
	Total Number of Features	**53**

them during the segmentation process. Hence, the main goal of this stage was to retrieve as many actual regions as possible and reject as many false positive regions as possible. To achieve this, we focused on sensitivity of both the classes.

First, the whole dataset was converted into a binary classification problem for this stage, where the non lesion were labelled 0 and lesions are labelled 1. Then we performed standard normalization from the scikit-learn library on the dataset. Then we experimented with two different methods to achieve the desired performance on the training dataset and selected the best method. Below we have explained the two methods that we have tried.

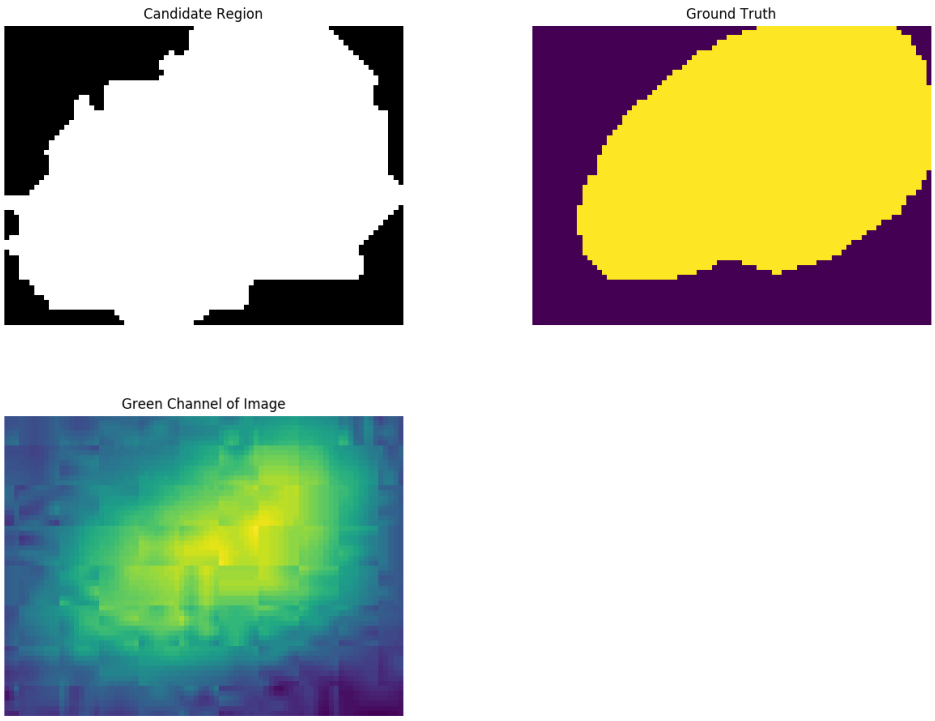


Figure 9: Hard Exudate region taken from the Patient 1 image

4.2.1 Mini Batch Method

Since our dataset was massive in size, instead of fitting the whole dataset into the model at once we applied the concept of mini batch to reduce the computational time. We divided the dataset into small chunks of 16000. Each chunk was highly imbalanced so we carried out sampling to address the class

imbalance problem. After experimenting using SMOTE and Random Under Sampling, we found out that random under sampling of the false positives is more suitable. The ratio of lesion and non lesion was kept at 1:2.

We fitted the model using the chunk in each loop. The partially fitted model is carried forward to the next loop and the next chunk is fitted. So, after the whole dataset is iterated, we have a model which has been fitted on the whole dataset and has the knowledge from the whole data. We then got the predictions and evaluated the model on the dataset. The whole process is summarized in Algorithm 2.

We used the SGDClassifier in the scikit-learn [7] library of python as the model. The SGDClassifier implements linear models with stochastic gradient descent (SGD) learning. The loss parameter indicates which linear model is being used. For example: the hinge loss indicates it is a Support Vector Machine and the log loss indicates that it is a Logistic Regression. We experimented using all the 9 loss functions that are available. For each loss function, we also tried the three regularization techniques (l2', 'l1', 'elasticnet') that were available.

Algorithm 1 Mini Batch Method

1: Convert the multi-class dataset into binary class, where the non lesion are labelled 0 and lesions are labelled 1
2: **while** instances left in the dataset **do**
3: Create chunk of 16000 instances
4: Counting occurrences of the minority class (lesions)
5: Undersample the classes to 2:1 ratio for non lesions and lesions
6: Partial fit the chunk into the model
7: **end while**
8: Get prediction on the entire dataset using the fitted model
9: Get the sensitivity for both lesions and non lesions

4.2.2 Decision Tree Method

Later, we tried to create a cascade classifier inspired from the Haar Cascade Classifier taught in our Image Processing course. After experimenting using SMOTE and Random Under Sampling, we decided to apply Random Under Sampling on the majority class (non lesions). The ratio of lesion and non lesion was kept at 1:2. We used Decision Tree as the model in this method. We did not carry out any feature selection as Decision Tree creates the tree using only the features that have the highest information gain which is in other words can be referred to as feature selection. And we also did not need to handle the class imbalance problem in this method, in fact, performing under sampling reduced the performance of the model.

Algorithm 2 Decision Tree Method

1: Convert the multi-class dataset into binary class, where the non lesion are labelled 0 and lesions are labelled 1
2: Counting occurrences of the minority class (lesions)
3: Undersample the classes to 2:1 ratio for non lesions and lesions
4: Fit the model on the sampled dataset
5: Get prediction on the entire dataset using the fitted model
6: Get the sensitivity for both lesions and non lesions

4.3 Stage 2 - Multiclass Classification

We applied the multi class classification using several classifiers: Gradient Boost, Decision Tree, Gaussian Naive Bayes, K Nearest Neighbour, Random Forest, Extra Trees Classifier, Logistic Regression, AdaBoost and Support Vector Machine (with linear kernal). Since most of these classifiers are only designed for binary class classification task, we had to use meta-strategiess to achieve the multi class classification. We experimented using One-vs-One and One-vs-Rest and found that One-vs-Rest was more suitable for this problem. The One-vs-Rest strategy splits a multi-class classification of four leasion into four different binary classification problem.

4.4 Performance Evaluation

For Stage-1, we selected the Decision Tree Method as it had higher sensitivity on both the lesions and non lesions. Based on the predictions of this model we carried forward on the regions which were predicted as lesions to the next stage. We needed to focus on sensitivity of both the classes because we wanted to carry forward all the true regions and at the same time we also had to make sure that not too many false regions go to the next stage. This was the main reason why we did not use the mini batch method as even though it gave us a high sensitivity for the true lesions, it was also predicting a lot of false regions as true regions.

For stage-2, we used 10 fold cross validation to achieve the performance on the training set. Furthermore, at each fold we applied random under sampling of the hard exudates class as it was dominating all the other classes. The under sample value at each fold was assigned as the average value of all the lesions in that fold. We did not apply feature selection in any of the stages because we only have 53 features which is a very small number compared to the huge number of instances we have. The model of the classifiers were saved and was later used on the test set to achieve the testing performance. All the results are shown in Table 3 and Table 4.

Based on the performance on the training dataset (Figure 15), Gradient Boost is the best classifier for stage-2 (Figure 10). However, during testing we can clearly see that Decision Tree performs noticeably better (Figure **??**).

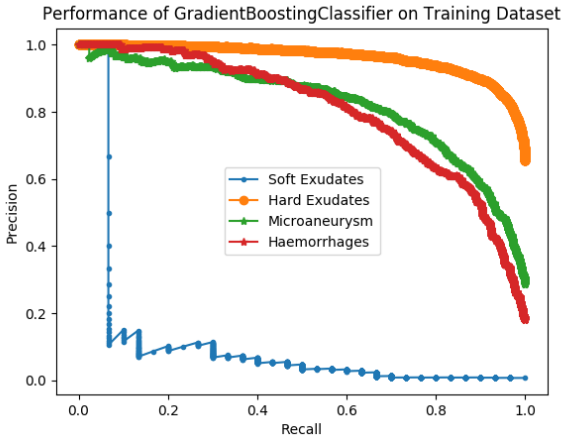
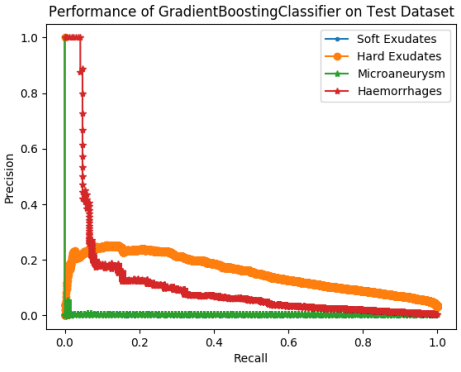
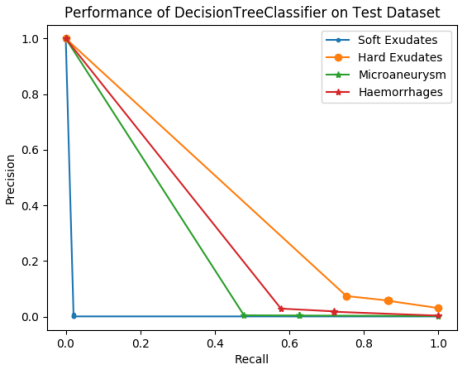


Figure 10: Best Performing Classifier on the Training Set



(a) Best Performing Classifier based on the Training Performance



(b) Best Performing Classifier on the Test Set

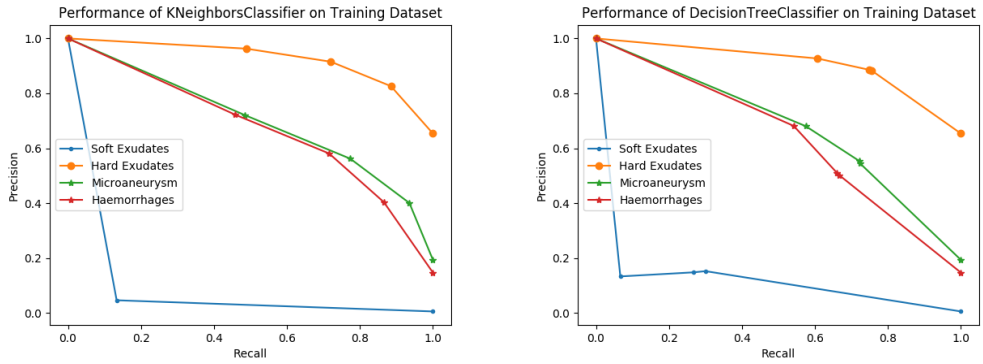


Figure 12: Performances of K-Nearest Neighbour (left) and Decision Trees (right) on the training set

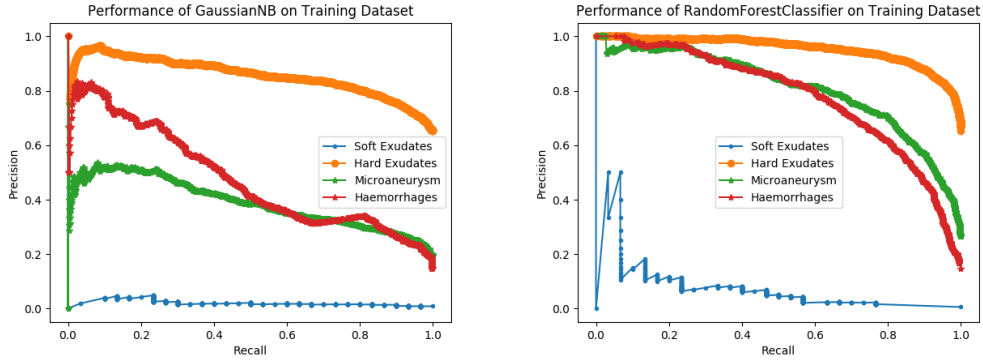


Figure 13: Performances of Naive Bayes (left) and Random Forest (right) on the training set

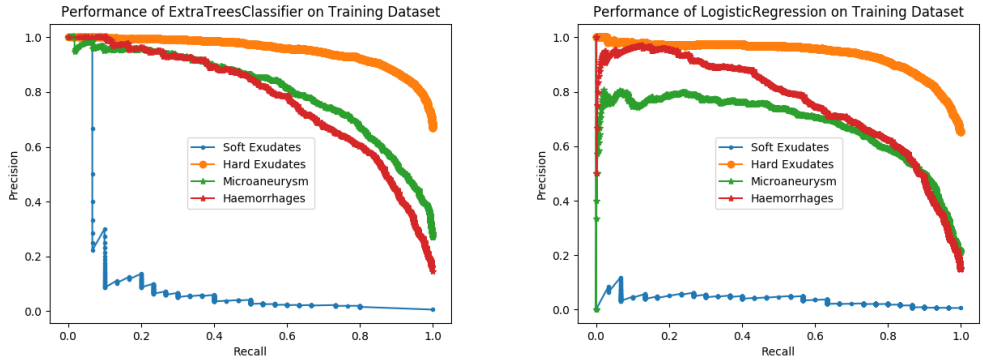


Figure 14: Performances of Extra Trees Classifier (left) and Logistic Regression (right) on the training set

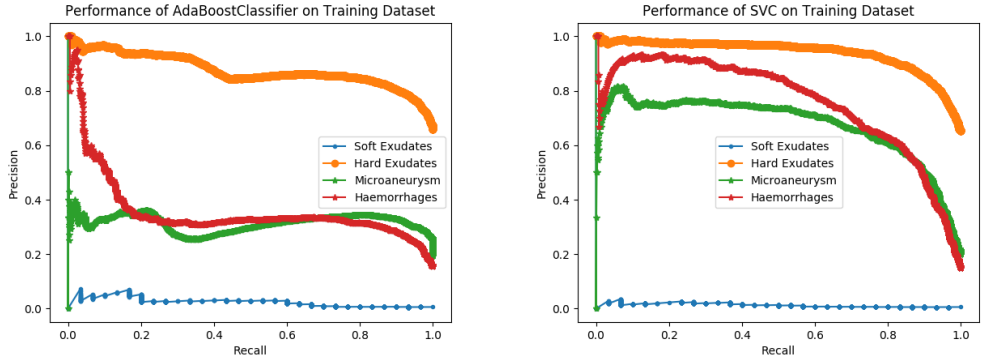


Figure 15: Performances of AdaBoost (left) and SVM (right) on the training set

Table 3: auPR Score of the Classifiers on Training Dataset (in %)

Classifier	Soft Exudates	Hard Exudates	Microaneurysms	Haemorrhages
Gradient Boost	11.11	**96.21**	**82.29**	**80.44**
Decision Tree	**12.63**	90.72	67.67	63.75
Gaussian Naive Bayes	2.022	85.63	38.97	47.65
K Nearest Neighbour	11.48	92.41	70.71	67.97
Random Forest	7.383	95.38	81.59	77.84
Extra Trees Classifier	11.25	95.31	80.73	77.3
Logistic Regression	3.621	93.76	67.79	76.02
AdaBoost Classifier	2.408	87.55	31.53	35.81
Support Vector Machine	1.256	93.87	67.31	75.78

Table 4: auPR Score of the Classifiers on Test Dataset (in %)

Classifier	Soft Exudates	Hard Exudates	Microaneurysms	Haemorrhages
Gradient Boost	0.2221	15.44	0.4778	11.4
Decision Tree	**1.166**	**41.79**	**24.24**	**30.37**
Gaussian Naive Bayes	0.2004	9.463	0.2998	3.162
K Nearest Neighbour	0.2701	40.42	19.04	23.88
Random Forest	0.3585	28.13	1.034	8.936
Extra Trees Classifier	0.2969	26.32	0.5149	6.467
Logistic Regression	0.6017	9.372	0.3709	4.492
AdaBoost Classifier	0.1742	12.41	0.2637	2.2
Support Vector Machine	0.2689	8.233	0.3733	3.886

4.5 Future Work for Machine Learning

The significant drop in the result on the test set is aclear indication that the model overfitted on the training dataset. The main cause of overfitting is the few number of features. With huge number of features, there would have been a less chance of overfitting. While searching for more features we came across several publications citegabor-1 [11] [6] [2] where Gabor filter was used. We wrote a code to extract the Gabor filter and we added the code to obtain GLCM features from the Gabor filtered image. However, it took around 20 hours for the features to be extracted for Hard Exudates and Haemorrhages. It took around 10 hours for th feature extraction of Soft Exudates. However, for Microaneurysm, after 36 hours it was only at the 17th image, so we had to terminate the code as we did not have enough time to extract features for the training set let alone the test dataset. We also wanted to extract Haar features but was unable due to the time constraint. With a lot of features, we would also have had the scope to apply feature selection algorithms which we could not do with our current feature set.

5 Deep Learning

To complete the challenge of this project, we wanted to compare the performance of the results of Deep Learning with the Machine Learning Segmentation.

However, in the case of Deep Learning, the goal was to segment the lesions individually, which led to a binary class segmentation. Thus, the upcoming approaches were chosen and implemented to segment each lesion separately.

5.1 Image Augmentation

The dataset which we used contains 54 training images and 27 test images with their coresponding groundtruths for the following lesions: Hard Exudates, Soft Exudates, Microaneurysms and Haemorrhages.

The nature of the Deep Learning algorithm is that it generalize better and its performance increases when provided large sets of data. With that in mind we have decided to employ to the fullest extent the given dataset and slice the images as a form of data augmentation.

The size of the original image from the dataset is 4288 x 2848 pixels. For the first UNET that we have used, we sliced the image into 5 slices such that the resulting slice will have 857 x 569 pixels and 6 slices each having the size of 714 x 474 pixels for the second UNET. The change in the number of slices is a need that had to be fulfilled with the increase of the depth of the UNET for the second network. Otherwise, our GPU memory would not be enough.

In order to proceed with the data augmentation, we have used the albumentations library. Firstly, the training image is randomly rotated by up to 35 degrees. Afterwards, the image is flipped horizontally with a probability of 50%, which means that half of the images will be flipped and then it is vertically flipped with a probability of 10%. Then we use a shift-scale-rotate transformation with 50% of probability, grid distortion with probability 30% and channels shuffle. Both the training and test images are normalized in this process.

5.2 Loss Functions

The loss function is meant to boost the effectiveness and efficiency of the learning algorithm of the deep neural network in discussion. In order to compute the error of the model during the optimization process given the complexity of the dataset that contains retinal lesions and due to the input class imbalance we tried multiple loss functions that will be discussed in this subsection.

5.2.1 Combo Loss

The Combo Loss function [3] is meant to control the trade-off between false positives and false negatives. In the original paper it was used on organ detection and was able to handle the lack of data in the dataset by penalizing the false positives more.

This loss function uses the Dice function to handle input class-imbalance problem, which is the case in our dataset too given that the retinal lesions are very small compared to the rest of the eye representation. In addition, it enforces a smooth training using cross entropy. The loss L is the weighted sum of two terms: A Dice loss and a modified cross entropy as a form of encoding the gradual learning.

The Combo Loss introduces the alpha α parameter which controls the amount of Dice term contribution and the beta β parameter, $\beta \in [0, 1]$, which controls how the model penalizes false positives and false negatives, namely when β<0.5 false positives are penalized more.

5.2.2 Focal Loss

The Focal Loss [**focalLoss**] focuses training on a sparse set of hard examples and avoids the scenario where the "easy" negatives overwhelm the detector in the training phase.

This function also uses the α hyperparameter, but it does not differentiate between easy and hard examples. Thus, the standard cross entropy loss is reshaped to down-weight the well-classified examples (easy examples) hence focus on training the hard negatives.

The Focal Loss attaches a modulating factor to the cross entropy loss

$$(1 - p_t)^\gamma$$

Furthermore, it introduces the focusing parameter gamma, $\gamma \geq 0$ that take values up to 5 and such the formal definition of the Focal Loss including the alpha hyperparameter is:

$$FL(p_t) = -\alpha_t(1 - p_t)^\gamma log(p_t)$$

It is straightforward to see that for γ = 0, the Focal Loss is in fact the α_t-balanced cross-entropy loss. For our architectures we have used the focusing parameter gamma = 2. Moreover, we can observe that as the probability tends to 1, the modulating factor tends to 0, thus the loss for true positives and true negatives is down-weighted.

5.2.3 Tversky Loss

The Tversky Loss function [10] is based on the Tversky similarity index which is a generalization of the Dice similarity coeffcient and the Fβ scores.

The formula for the Tversky loss as described in the cited paper is the following:

$$T(\alpha, \beta) = \frac{\sum_{i=1}^{N} p_{0_i} g_{0_i}}{\sum_{i=1}^{N} p_{0_i} g_{0_i} + \alpha \sum_{i=1}^{N} p_{0_i} g_{1_i} + \beta \sum_{i=1}^{N} p_{1_i} g_{0_i}}$$

In the original paper they have used 3D images and thus the probabilities are the probabilities of voxels, not pixels. Yet, our images are 2D represented and as a result, we have translated the probabilities into pixel probabilities as follows: p_{0_i} is the probability that a pixel is a lesion, p_{1_i} probability that a pixel is non-lesion, g_{0_i} is 1 for lesion and 0 for non-lesion.

In this setting, by imposing the condition of $\alpha+\beta = 1$), we obtain the F_β scores. When $\alpha = \beta = 0.5$ the Dice coefficient is obtained which is equivalent to the F_1 score.

5.2.4 Focal Tversky Loss

The Focal Tversky Loss is a solution that aims to improve the imbalance between precision and recall due to the class imbalance and non-linearly focuses training on hard examples (where Tversky < 0.5). Moreover, it suppresses easy examples from contributing to the loss function.

$$FTL_c = \sum_c (1 - TI_c)^{1/\gamma}$$

TI_c is a generalization of the Tversky Index function previously shown in 4.2.3. In the above formula, c denotes the class in the segmentation problem. The gamma is the focal parameter, $\gamma \in [1, 3]$

Alpha, beta and gamma are hyperparameters that will dictate the performance of the loss function. A higher value for α will improve the model convergence by shifting the focus to minimize false negative predictions. We notice that in this loss function too, choosing $\alpha = \beta = 0.5$ will lead to the Dice Score. By setting gamma $\gamma = 1$, the Focal Tversky Loss Function is reduced to the Tversky Loss Function.

5.3 Architectures

Our dataset consists of medical data that illustrate Retinal Fundus with four lesions. The lesions are very small compared to the background and thus segmenting them is a very challenging task even for a deep learning network. Considering the complexity of the task, we decided to implement a special type of Convolutional Neural Network that has been specifically designed for medical tasks: **UNet**.

In order to implement the architectures that we analysed, we have used the PyTorch library to build the Deep Neural Networks. Each of the mentioned architectures have been trained and tested individually on the four lesions.

5.3.1 UNET Model with 4 layers depth

Initially we have tried to solve the task with a UNet that has the depth of four layers.

On the basis of this, the training started with the most promising lesion in terms of detection, namely the hard exudates. We have used the Binary Cross Entropy loss function, the Combo Loss, Tversky Loss and Focal Tversky Loss to observe the behaviour of the network.

	Hard Exudates	Hemorrhages
Combo Loss	(-0.0334, -0.4040)	(-0.0127, -0.1012)
Focal Loss	(0.0005, 0.0007)	(0.0013, 0.0010)
Tversky Loss	(0.8634, 0.8684)	(0.9368, 0.9182)
Focal Tversky Loss	(0.8621,0.8684)	(0.8477, 0.7980)

Table 5: Loss functions 4 layers UNet

The previous table presents the tuple with (training _loss, validation _loss.The losses were computed separately for the lesions and the results have been selected after running the same number of epochs. For the haemorrhages, the Focal Loss Tversky has reached (0.6434, 0.6164) loss tuple after 120 epochs.

Binary Cross Entropy although a rather popular choice in general did not perform well for our dataset. For instance, for the haemorrhages, the loss tuple was (0.0512, 0.0552).

Considering all the beforehand mentioned loss functions, we have come to the conclusion that the best results would be obtained using the Focal Tversky loss function.

As a result of the previous training we have acknowledged that 4 layers were insufficient for the complexity of the goal. Thus, we increased the depth of the network by one layer.

5.3.2 UNET Model with 5 layers depth

The network uses 2D convolutions with kernel size = 3, followed by the ReLU activation function followed by batch normalization then again 2D convolutions followed by ReLU and batch normalization.

Once we reached the bottleneck, the network will go up the U shape by 2D transposed convolution given that the input is composed of multiple planes or simply by upsampling followed by a 2D convolution.

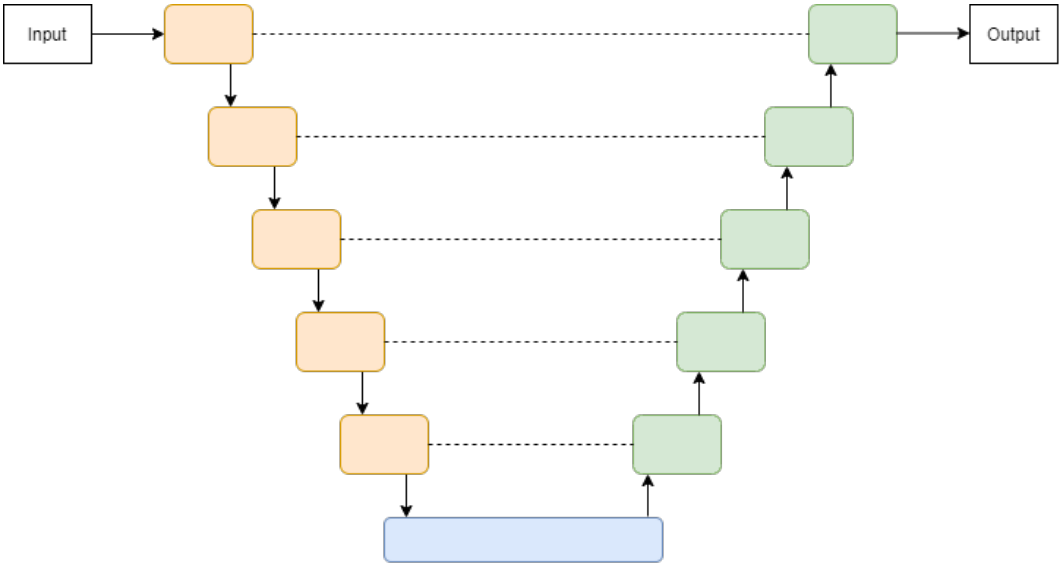


Figure 16: General architecture of the 5 layers UNet

The summary from the Google Colab notebook for the implemented architecture is illustrated in Figure 17.

5.3.3 UNET Model with 50 layers depth and ResNet Transfer Learning

The purpose of this attempt of enhancement is to explore the possibilities of transfer learning for our model given the dataset.

For this part we have taken a pretrained ResNet with 50 layers and combined it with our 5-layers UNet such to improve the overall performance of the network.

5.4 Area Under Precision Recall Curve

Class imbalance is a major challenge that we have faced in the given dataset. In order to properly measure the performance of our models, we have have used the Area Under Precision Recall Curve (AUPRC) as the final indicator.

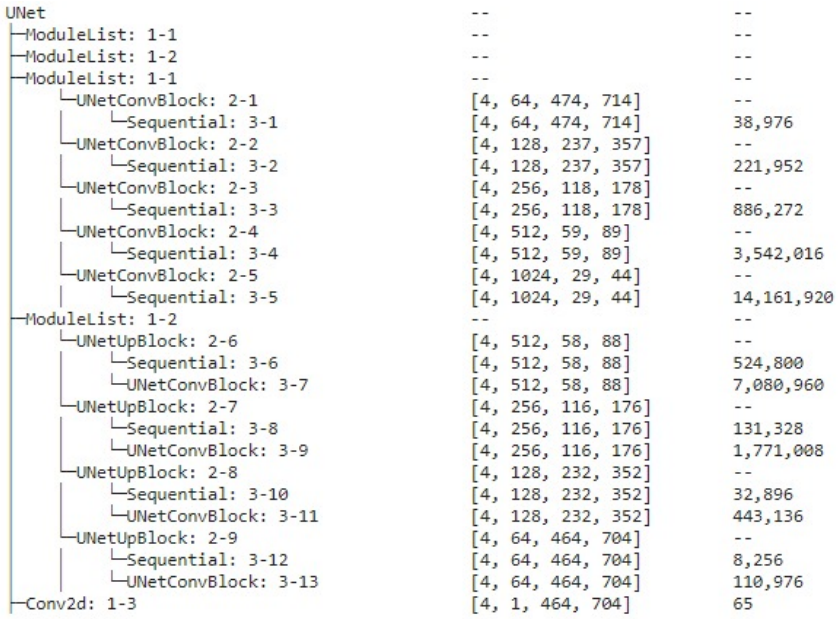


Figure 17: Summary of the 5-layers UNet

The precision is the ability of the classifier not to label as positive a sample that is negative and is computed as the ratio between true positives over the sum of true positives and false positives.

$$Precision = \frac{TP}{TP + FP}$$

The recall is the ability of the classifier to find all positive samples and is computed as the ratio between true positives over the sum of true positives and false negatives.

$$Recall = \frac{TP}{TP + FN}$$

With the purpose of computing the AUPRC, we have used the scikit-learn [7] library.

We have created a function that loads the prediction saved during the training phase and feeds the predictions and the actual groundtruths to the precision_recall_curve function.

	Hard Exudates	Soft Exudates	Hemorrhages	Microaneurysm
AUPRC	0.7005	0.4355	0.2931	0.3660

Table 6: AUPRC 5 layers UNET

The results of the AUPRC for the transfer learning through 50-layer ResNet enhanced UNet are as follow:

	Hard Exudates	Soft Exudates	Hemorrhages	Microaneurysm
AUPRC	0.5054	0.1875	0.1613	0.3968

Table 7: AUPRC UNET with Resnet

Figure 25 show the plots of the Area Under Precision Recall Curve for the four lesions in the case of using the UNet combined with ResNet architecture.

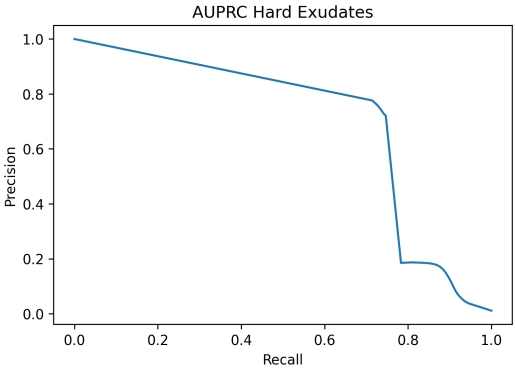


Figure 18: AUPRC HEx 5 layers UNet

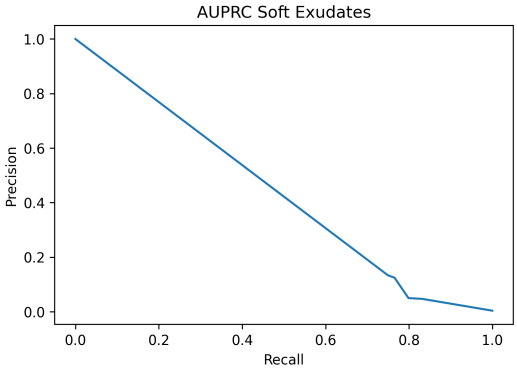


Figure 19: AUPRC SE 5 layers UNet

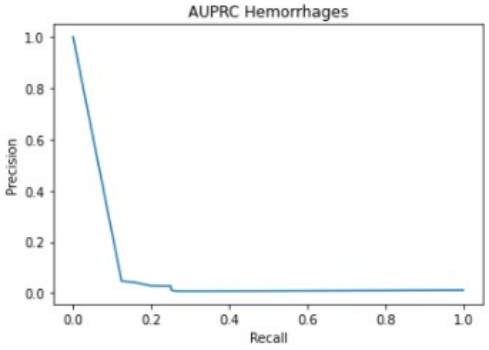


Figure 20: AUPRC HEM 5 layers UNet

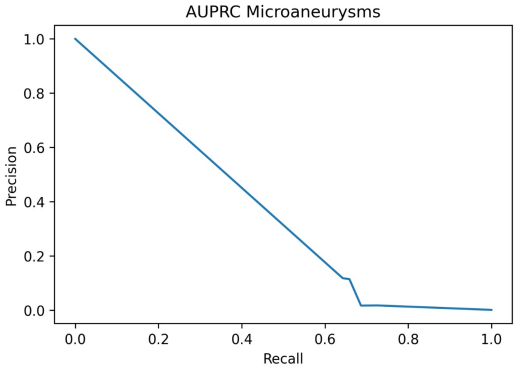


Figure 21: AUPRC MA 5 layers UNet

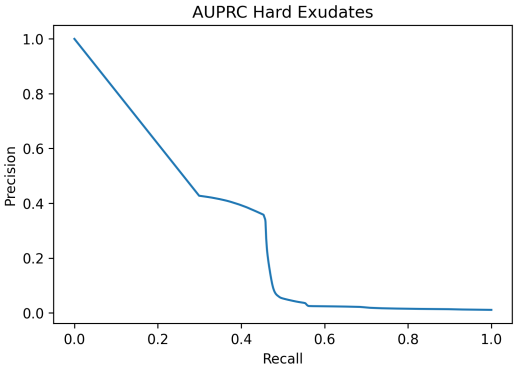


Figure 22: AUPRC HEx 50 layers UNet

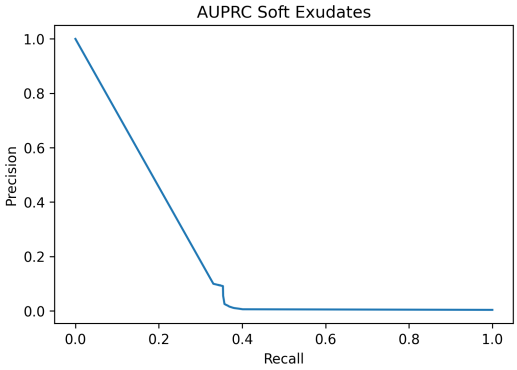


Figure 23: AUPRC SE 50 layers UNet

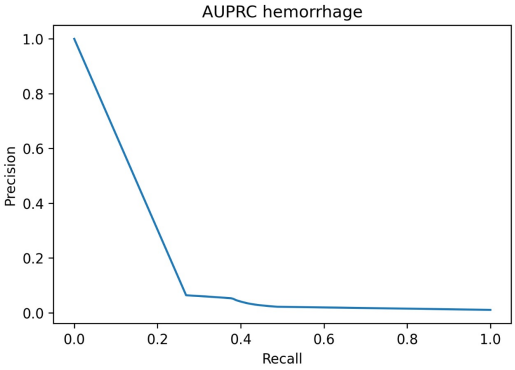


Figure 24: AUPRC HEM 50 layers UNet

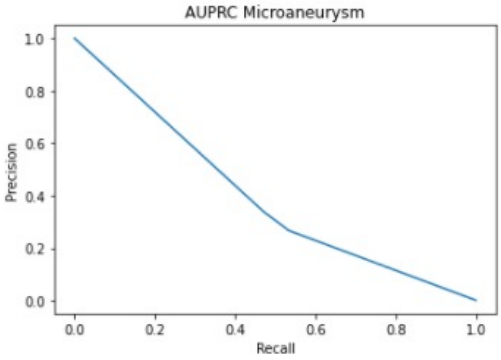


Figure 25: AUPRC MA 50 layers UNet

5.5 Predictions

The final output of the Deep Learning component of our project comprises of the predictions of the network. To have better terms of comparison, the upcoming figures will contain the predicted lesion and the actual lesion.

Figure 26: HEx Predicted

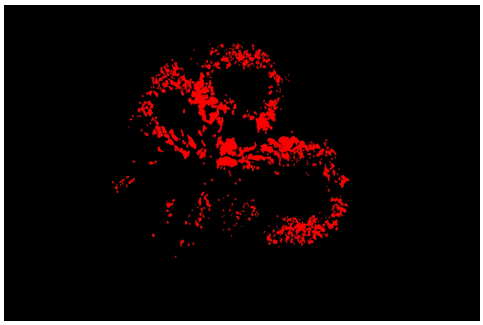


Figure 27: Actual HEx

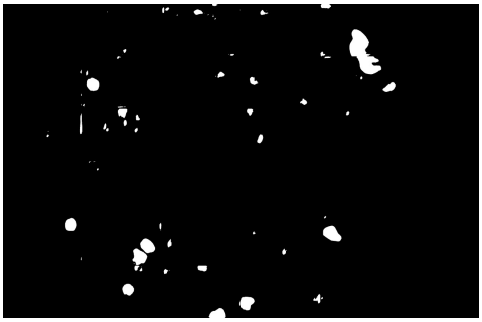


Figure 28: SE Predicted

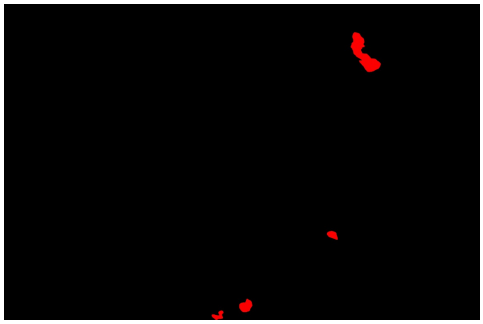


Figure 29: Actual SE

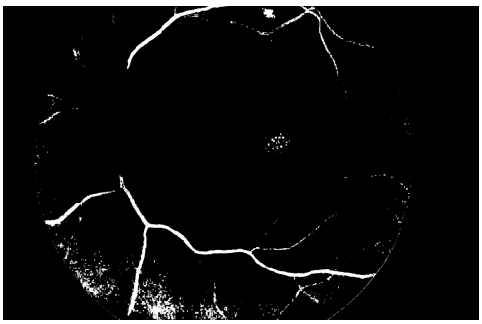


Figure 30: HEM Predicted

Figure 31: Actual HEM

Figure 32: MA Predicted

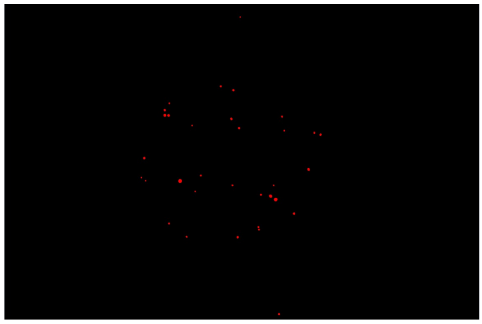


Figure 33: Actual MA

6 Conclusions

In this report, we have presented the details of our multi-disciplinary project including information about the data, elaboration on the algorithms and integrated pipelines, evaluation metrics over different pipelines and final results for all sub-tasks, i.e., lesion candidates, feature extraction, hierarchical multi-class classification (Machine Learning and Image Processing), segmentation of retinal lesions using various deep neural networks (Deep Learning) and the performance evaluation. Relentless efforts have been made in devising relevant pipelines, open for future developments and capable of advancing collective efforts to uplift the architectures to the standards.

The report presents 2-Stage Multi-Class Classification (AIA+ML) and Modified UNet with introduction to ResNet Transfer Learning approach (DL) to explore the algorithms for the improved medical image segmentation results on the IDRiD Dataset. In the future, we plan to extend these pipelines with requisite aforementioned measures, and reduce the deteriorations of learning settings to obtain better results.

References

[1] K Abinaya et al. "DETECTION AND CLASSIFICATION OF DIABETIC RETINOPATHY USING MACHINE LEARNING -A SURVEY". In: *The International journal of analytical and experimental modal analysis* 12 (July 2020), pp. 2128–2134.

[2] M. Usman Akram, Shehzad Khalid, and Shoab A. Khan. "Identification and classification of microaneurysms for early detection of diabetic retinopathy". In: *Pattern Recognition* 46.1 (2013), pp. 107–116. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2012.07.002. URL: https://www.sciencedirect.com/science/article/pii/S003132031200297X.

[3] Saeid Asgari Taghanaki et al. "Combo Loss: Handling Input and Output Imbalance in Multi-Organ Segmentation". In: (2018).

[4] S.L. Aleena and C.A. Prajith. "Retinal Lesions Detection for Screening of Diabetic Retinopathy". In: *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. 2020, pp. 1–6. DOI: 10.1109/ICCCNT49239.2020.9225617.

[5] Ahmad Zoebad Foeady et al. "Automated Diagnosis System of Diabetic Retinopathy Using GLCM Method and SVM Classifier". In: *2018 5th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*. 2018, pp. 154–160. DOI: 10.1109/EECSI.2018.8752726.

[6] Sarika Patil and B P Patil. "Detection of Small Red Lesions in Retinal Fundus Images Using AC-CLAHE, Gabor Filter and One Class SVM". In: *Test Engineering and Management* 83 (Mar. 2020), pp. 2177–2187.

[7] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[8] N.A. Rashed, Shaker Ali, and A.J. Dawood. "Diagnosis retinopathy disease using GLCM and ANN". In: *Journal of Theoretical and Applied Information Technology* 96 (Sept. 2018), pp. 6028–6040.

[9] Nadeem Salamat, Malik M. Saad Missen, and Aqsa Rashid. "Diabetic retinopathy techniques in retinal images: A review". In: *Artificial Intelligence in Medicine* 97 (2019), pp. 168–188. ISSN: 0933-3657. DOI: https://doi.org/10.1016/j.artmed.2018.10.009. URL: https://www.sciencedirect.com/science/article/pii/S0933365718300630.

[10] Ali Gholipour Seyed Sadegh Mohseni Salehi Deniz Erdogmus. "Tversky loss function for image segmentation using 3D fully convolutional deep networks". In: (2017).

[11] Vijayan T et al. "Gabor filter and machine learning based diabetic retinopathy analysis and detection". In: *Microprocessors and Microsystems* (2020), p. 103353. ISSN: 0141-9331. DOI: https://doi.org/10.1016/j.micpro.2020.103353. URL: https://www.sciencedirect.com/science/article/pii/S0141933120305123.