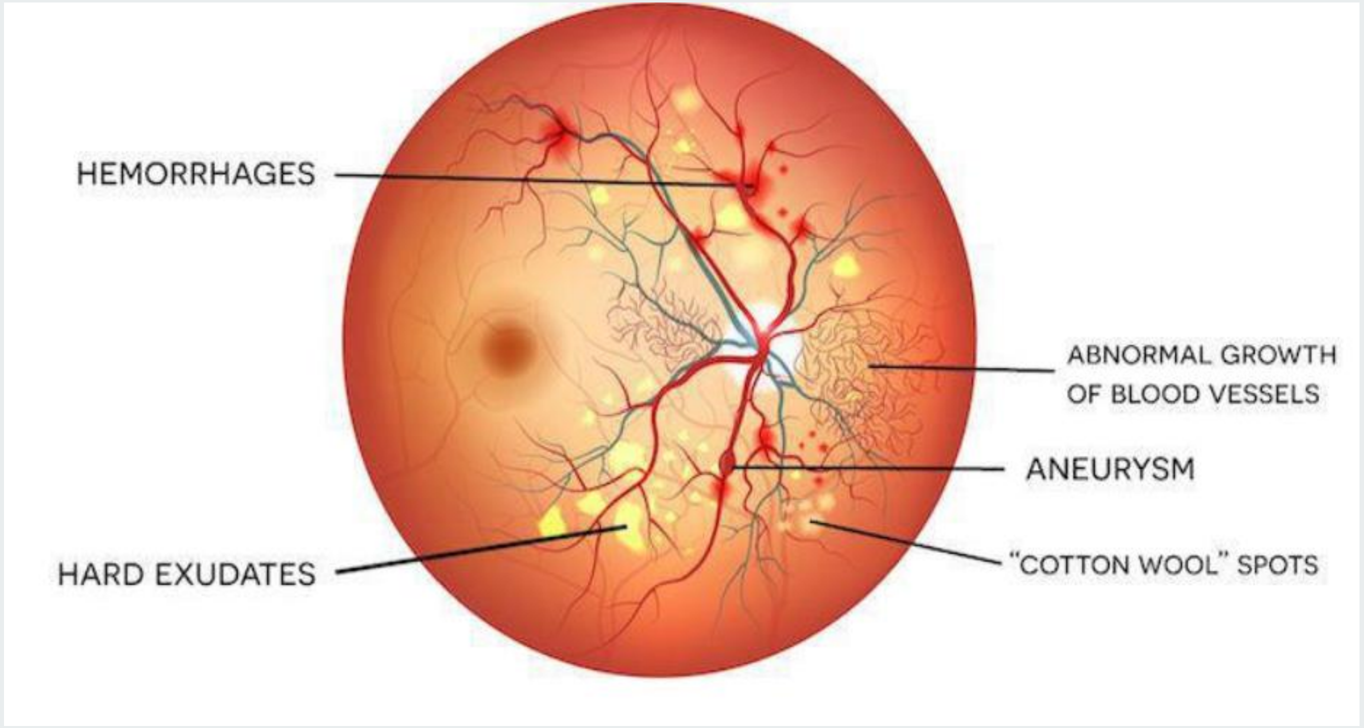
Diabetic Retinopathy Detection Using Deep Learning and Image Processing

Group Members:
Khwaja Wisal Maqsood(16BCS078).
Manoj (16BCS026).
Minor Project Presentation.

Diabetic Retinopathy Detection and Grading Using Deep Learning and Image Processing



What is Diabetic Retinopathy? How does it affect the population and diagnosis.

- Diabetic Retinopathy is a direct consequence of inconsistent and uncontrolled level of blood sugar which results in damage to the blood vessels of the light-sensitive tissue at the back of the eye (retina) causing blindness usually if detected early treatment is possible otherwise laser surgeries are used although the damaged caused to retina in case of severe retinopathy is not completely reversible but in 95% cases it is treatable[1][2].
- The longer one takes for treatment the more damage is caused and in some cases the blindness is permanent this is due to the fact that At first, diabetic retinopathy may cause no symptoms or only mild vision problems. Eventually, it can cause blindness[1][2].
- The condition generally develops amongst people suffering from type-1 or either type-2 diabetes. The longer one has diabetes and the less controlled your blood sugar is, the more likely you are to develop this eye complication[1][2].

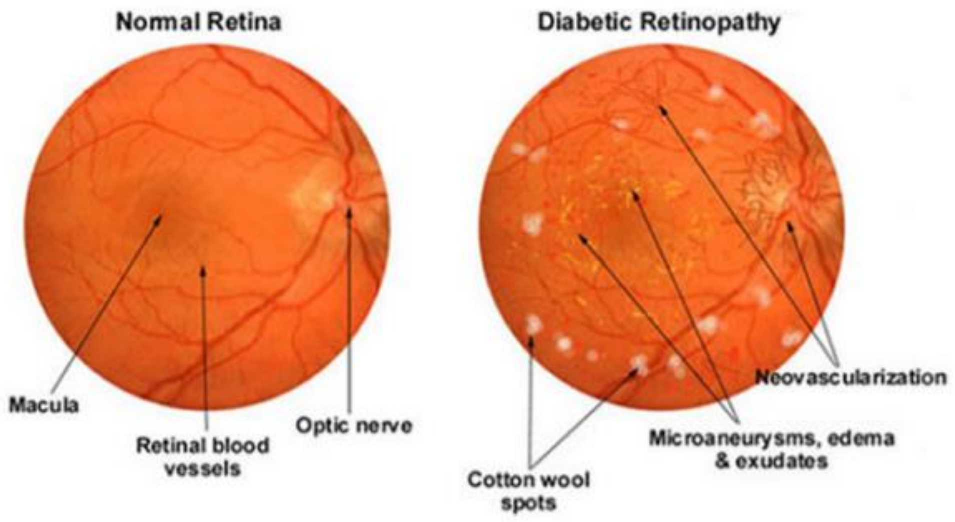


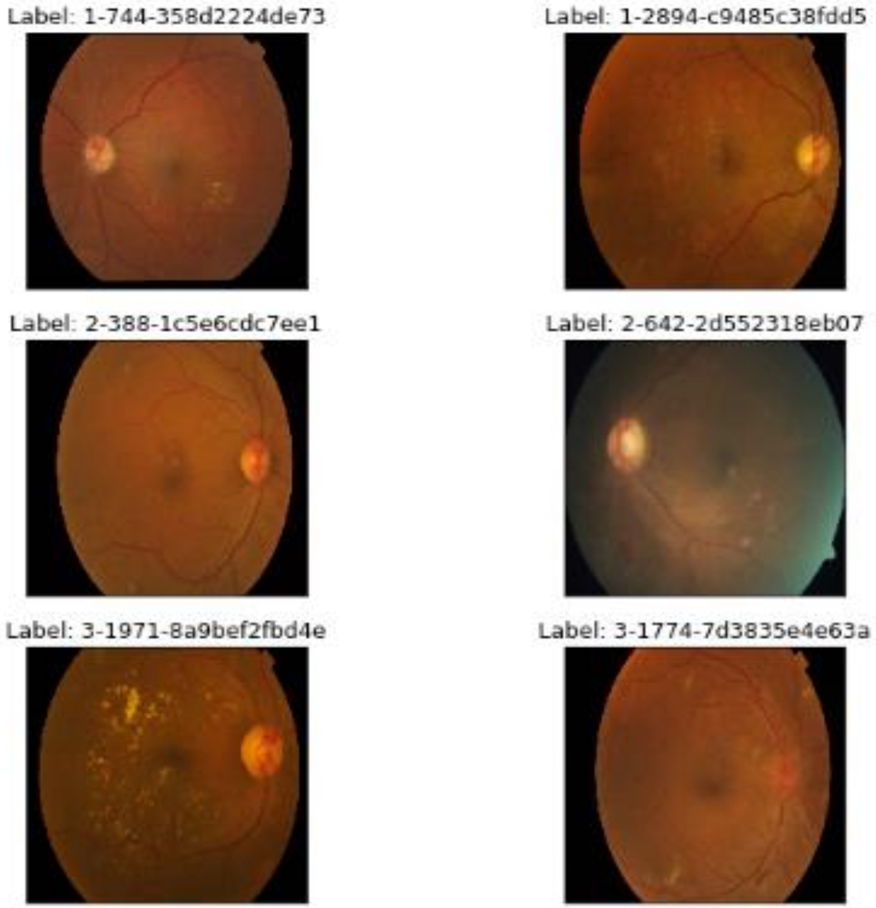
Figure 4: Microaneurysms, Exudates and Cotton-wool Spots
source by: http://www.coatswortheyeclinic.co.uk/photography/3102576

Diabetic Retinopathy Detection and Diagnosis.

Diabetic retinopathy is usually diagnosed by an ophthalmologist and graded into 5 categories as shown:

- 0-No DR.
- 1-Mild DR.
- 2-Moderate DR.
- 3-Severe DR.
- 4-Proliferative DR.
- This grading scheme was designed to tackle the earlier complex procedure for scaling DR diagnosis which was also known as ETDRS(Early Treatment of Diabetic retinopathy Study).

Due to the complexity of ETDRS a new scale was adopted which was easy to understand and implement and also improved diagnosis, this came to be known as the INTERNATIONAL CLINICAL DISEASE SEVERITY SCALE FOR DR [1][2][6].


Label: 1-744-358d2224de73
Label: 1-2894-c9485c38fdd5
Label: 2-388-1c5e6cdc7ee1
Label: 2-642-2d552318eb07
Label: 3-1971-8a9bef2fbd4e
Label: 3-1774-7d3835e4e63a

Existing Work and Detection Systems based on Deep Learning

S.NO	Authors	Year	Technique	Quadratic weighted kappa	AUC-ROC Metric	Dataset Used
1	Varun Gulshan, PhD; Lily Peng, MD, PhD; Marc Coram, PhD; Martin C. Stumpe, PhD; Derek Wu, BS; A	2016	Inception V-3.	0.8851(5)	0.991(3 vs All), 0.993(Avg-AUC across all datasets)	Eyepacs-1(15,000, Messidor-2(2000), Google's own DB-1,20,000, APTOS(13000)
2	Daniel Shu Wei Ting, MD, PhD[1,2]; Carol Yim-Lui Cheung, PhD[1,3]; Gilbert Lim, PhD[4]; et al	2017	Inception V-3 and V-4 + CLAHE	0.856(5)	0.971(4 vs all), Avg AUC-0.951	Eyepacs-2(76000),Messidor-2(200)
3	Bellemo, Valentina, et al.	2019	VGG-net	0.831(3)	0.973(2 vs all) and	Eyepacs-2,and Zambian eye scociety(18000)
4	Wei Zhang, Jie Zhong, Shijun Yang, Zhentao Gao	2019	Two Part ensemble. 1-binary classification then, 2- grading Sytem. 1-Xception,Resnet-50,Inception resnet-v2.	0.8771(4)	0.981(Classification), 0.994(Avg-AUC)	Sichuan Medical Center(15000)

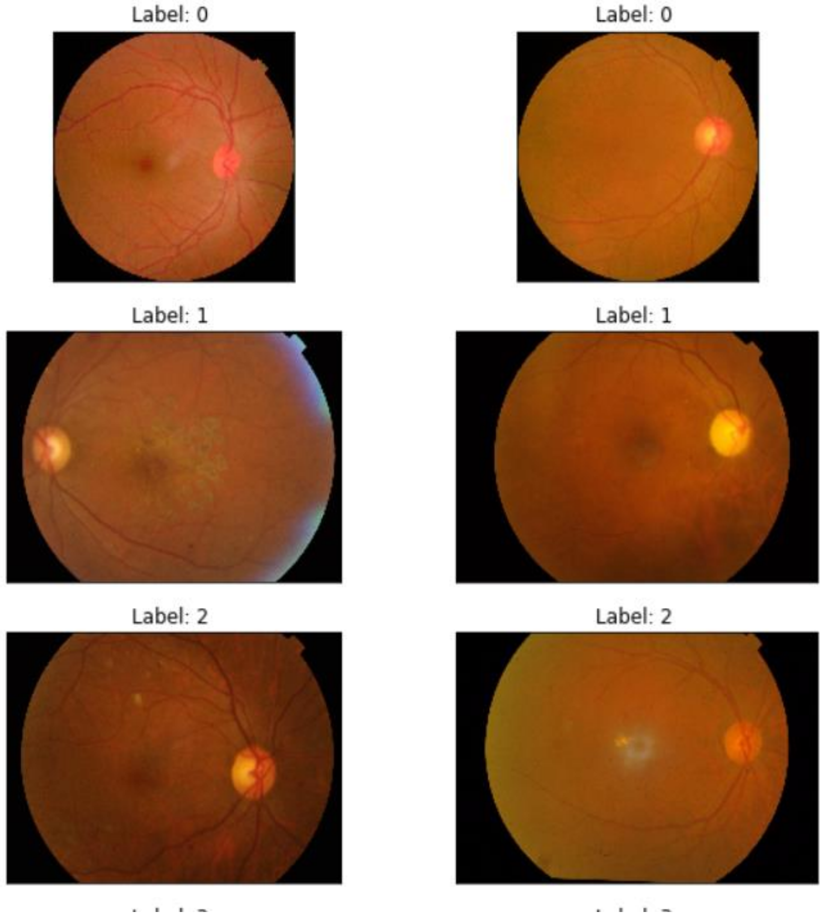
Datasets Used

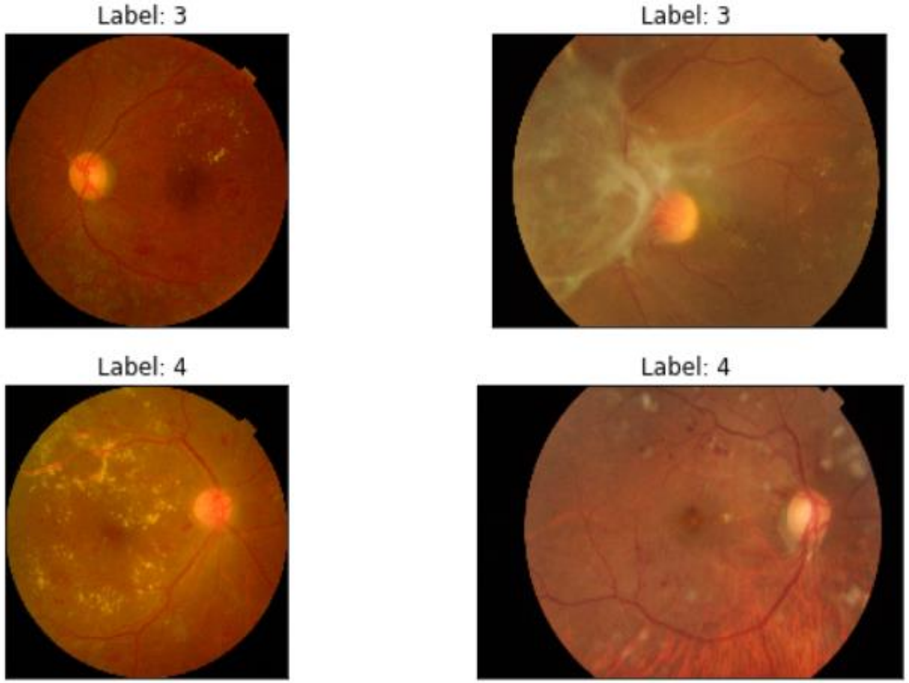
For this Project we selected Open- Sourced datasets that have been made publicly available:

- The First dataset we took from Eyepacs the Eyepacs - 2 dataset containing 84,000 images of 41,254 patients.
- Eyepacs - 1 dataset from Kaggle as part of DR detection competition 5 years ago. It Contains 1,20,000 images.
- APTOS DR detection Competition, a kaggle dataset now made publicly available by Aravind Eye care, It consists of 8,000 images.
- Messidor-2 dataset, It consists of 2000 images.
- All the images used are Retino-fundal images with a fixed resolution of 1024 x 1024 for all the images i.e in total we have 2,14,000 retino-fundal images of the retina which have already been graded by opthamologists into 5 categories of DR which are as follows:
    - 0-No DR.
    - 1-Mild DR.
    - 2-Moderate DR.
    - 3-Severe DR.
    - 4-Proliferative DR.

We train our models on 1,50,000 images then further fine tune the models using 40000 images. We validate our models on the remaining 15000 images. We finally test our models on the remaining 13,000 images using Test Time Augmentation.

Dataset Description and Anomalies





Anomalies in the dataset

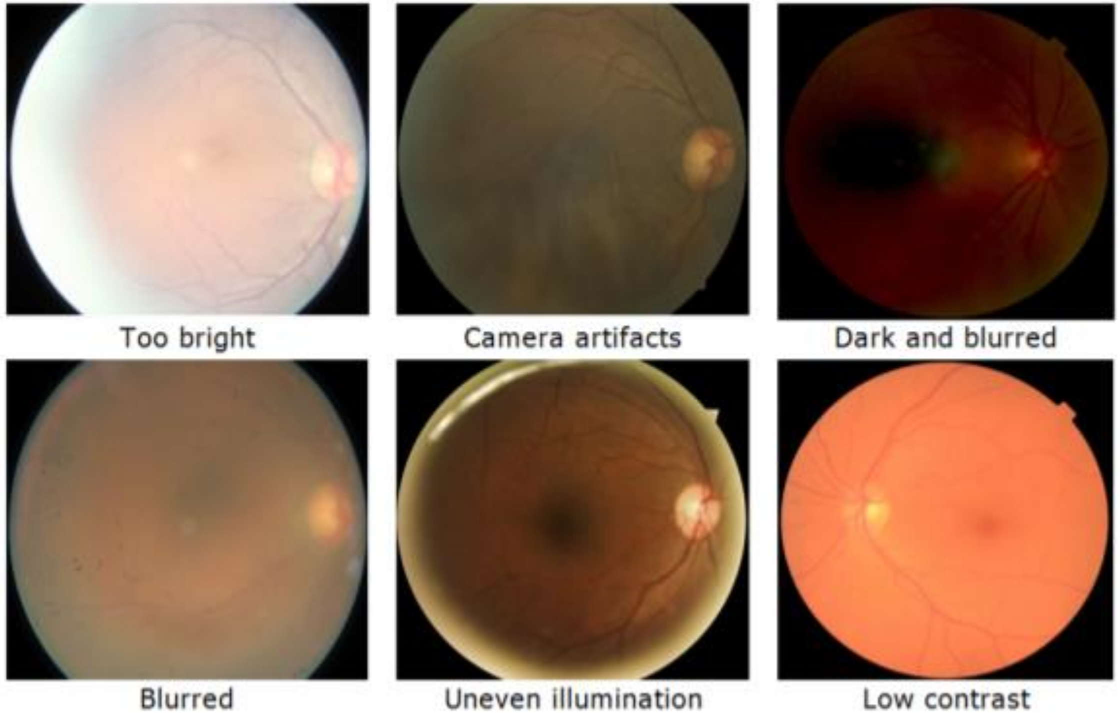


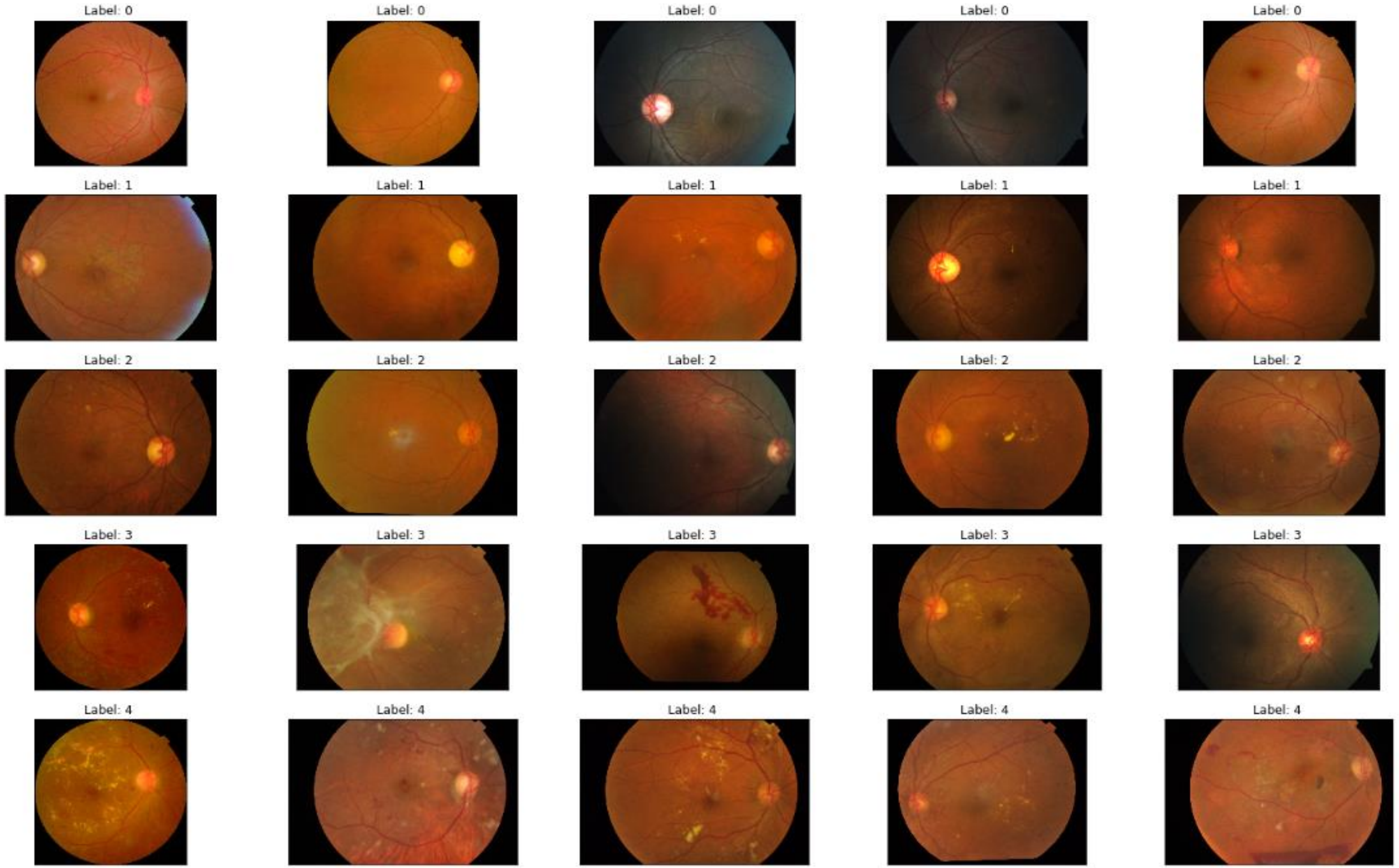
Figure 2: Retinal images of deficient quality

Image Preprocessing and Techniques

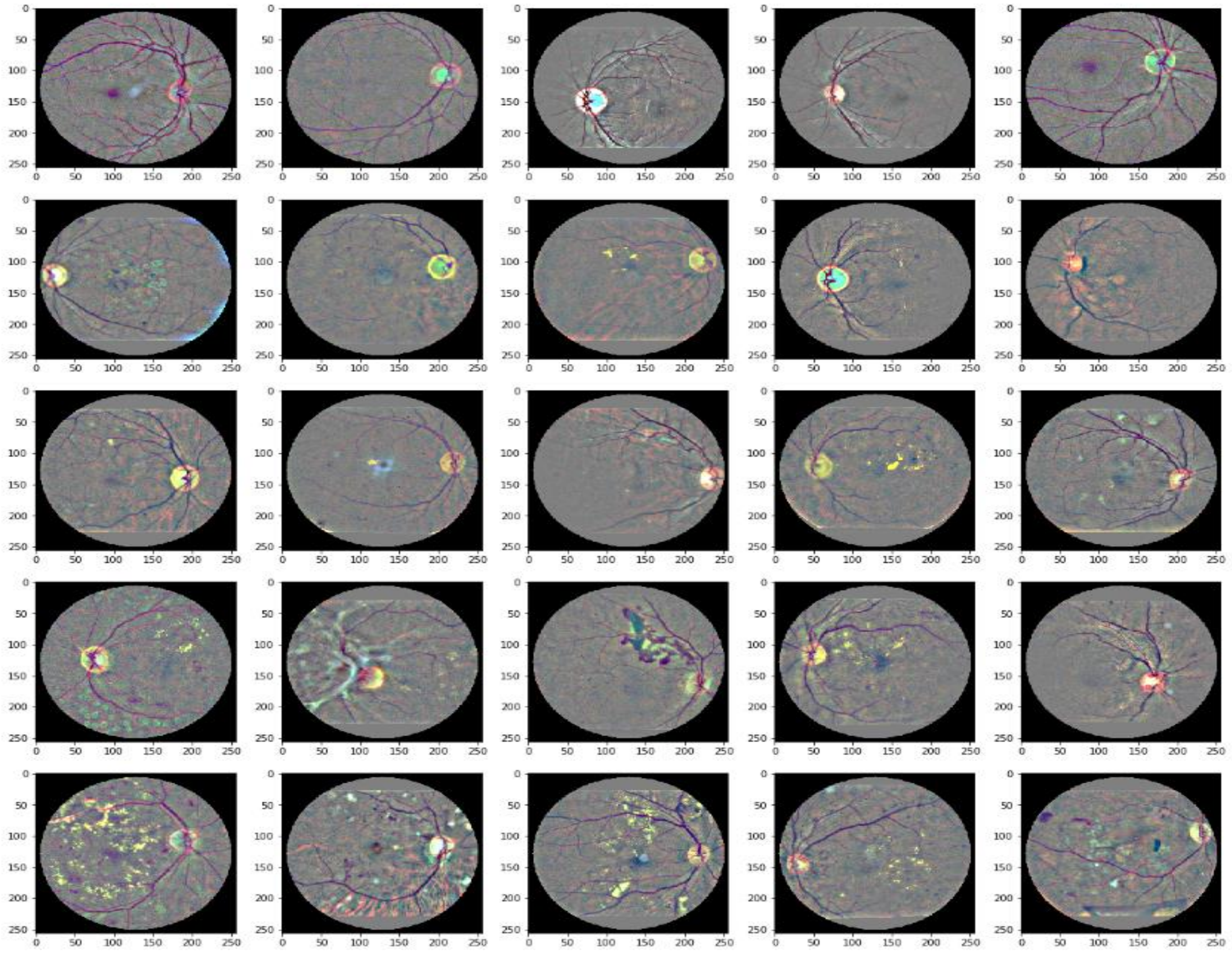
- We used a variety of Image processing techniques to improve model performance which are as follows:
- Median Subtraction.
- Gamma Correction.
- Adaptive histogram Equalization.
- Ben Grahams’s Image Processing applied to black and white as well as Colour images.

Median Subtraction:

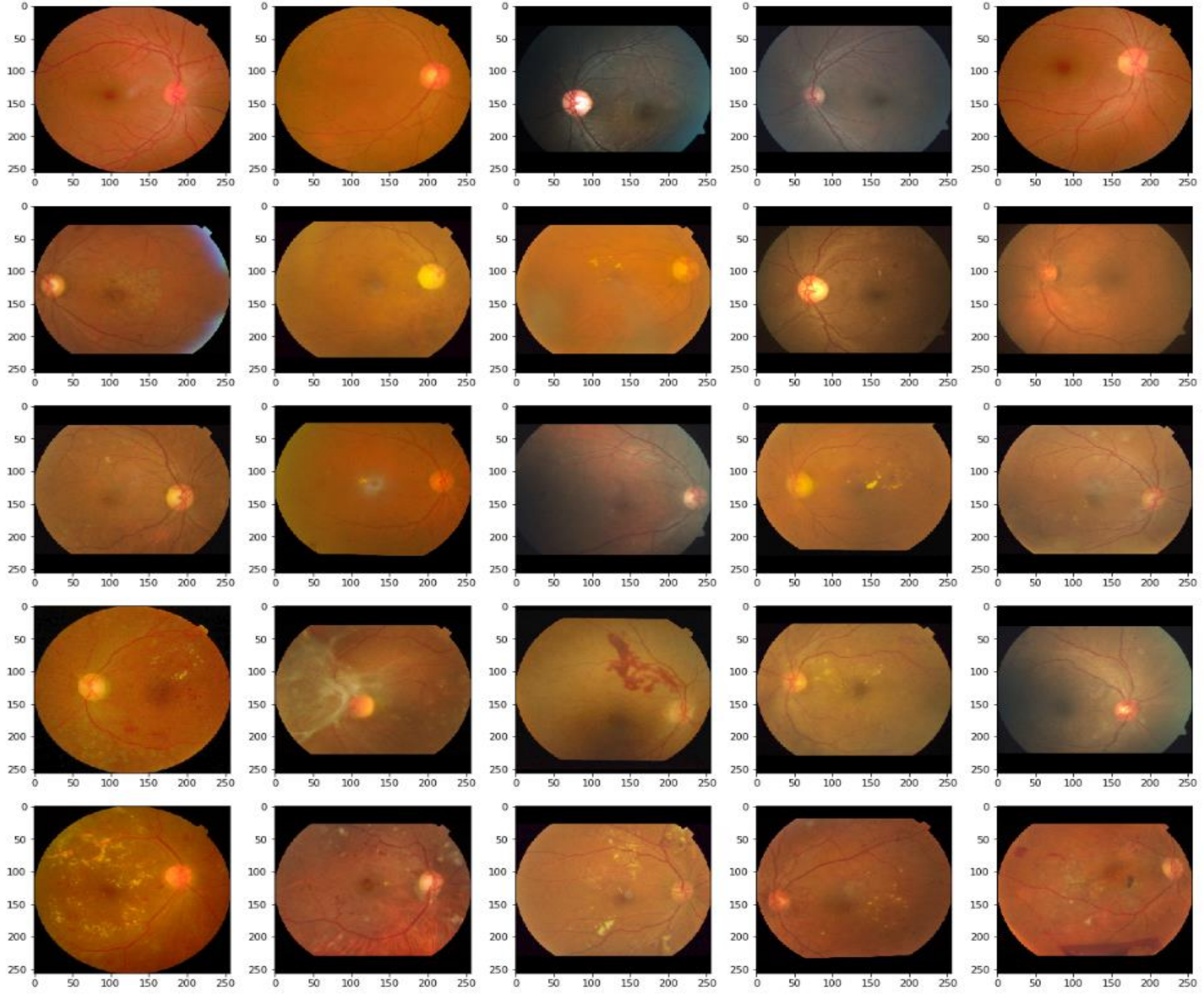
Original Image:



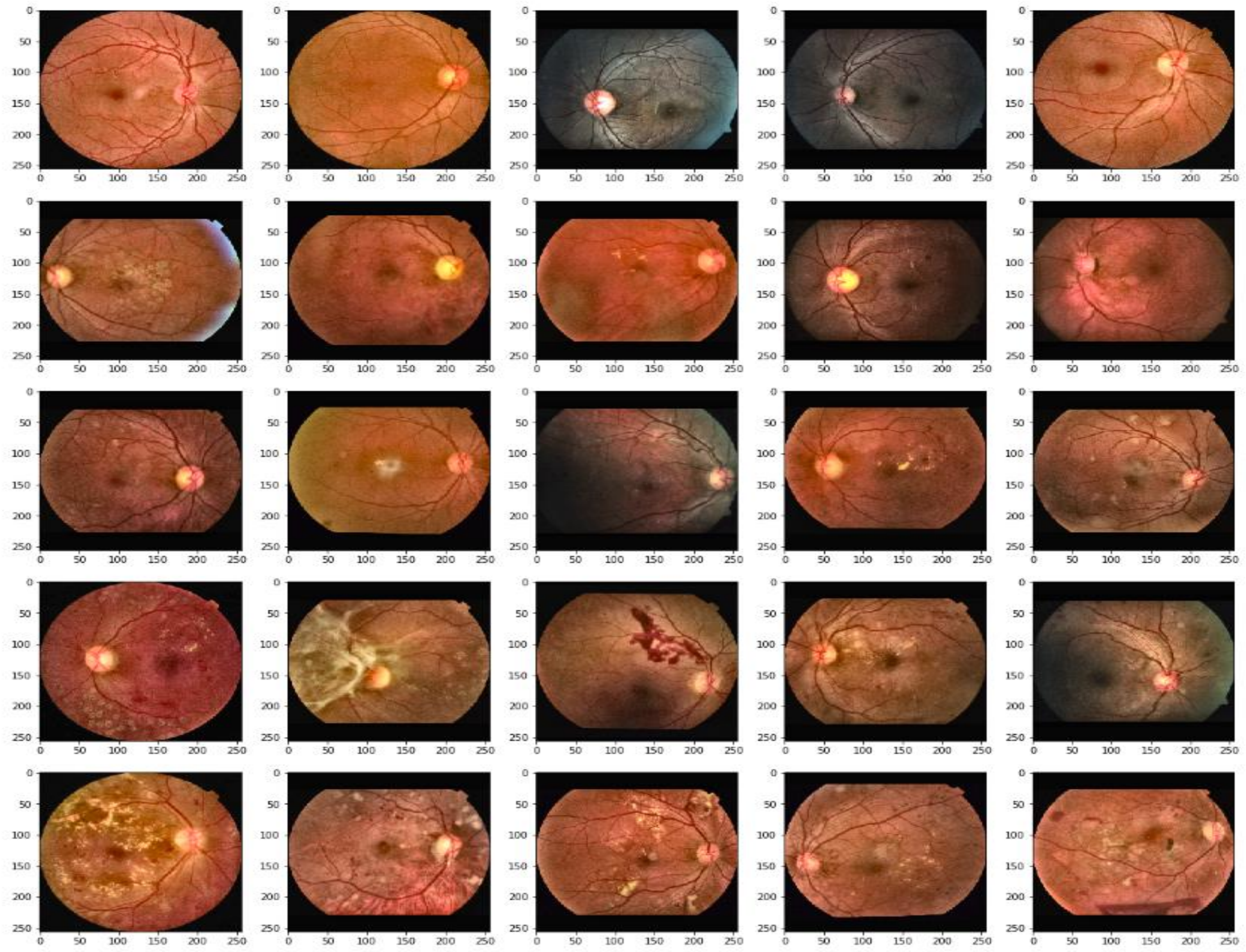
Preprocessed images after applying median Subtraction:



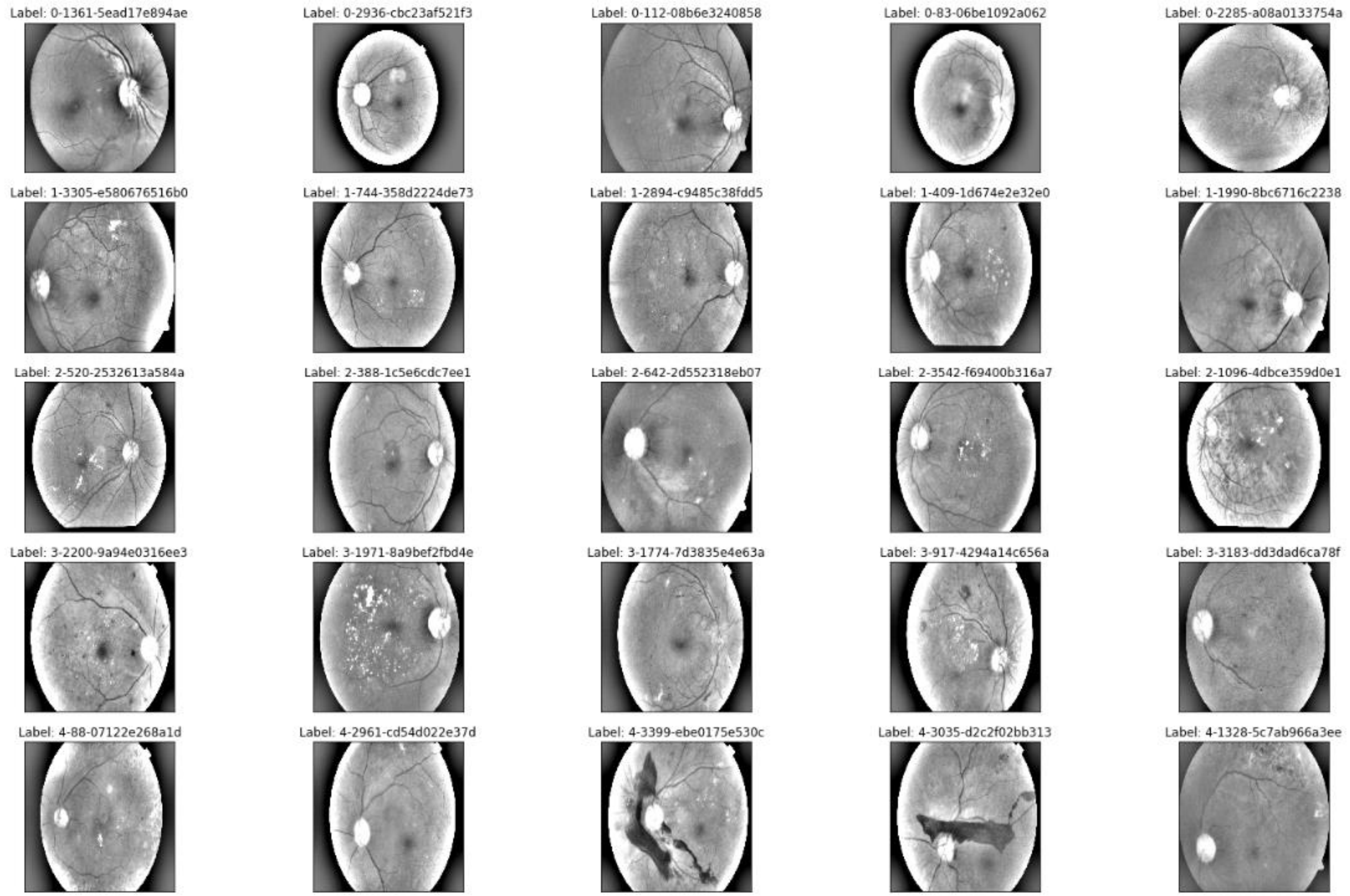
Preprocessed images after applying Gamma Correction:



Preprocessed Image after Applying Adaptive Histogram Equalization:



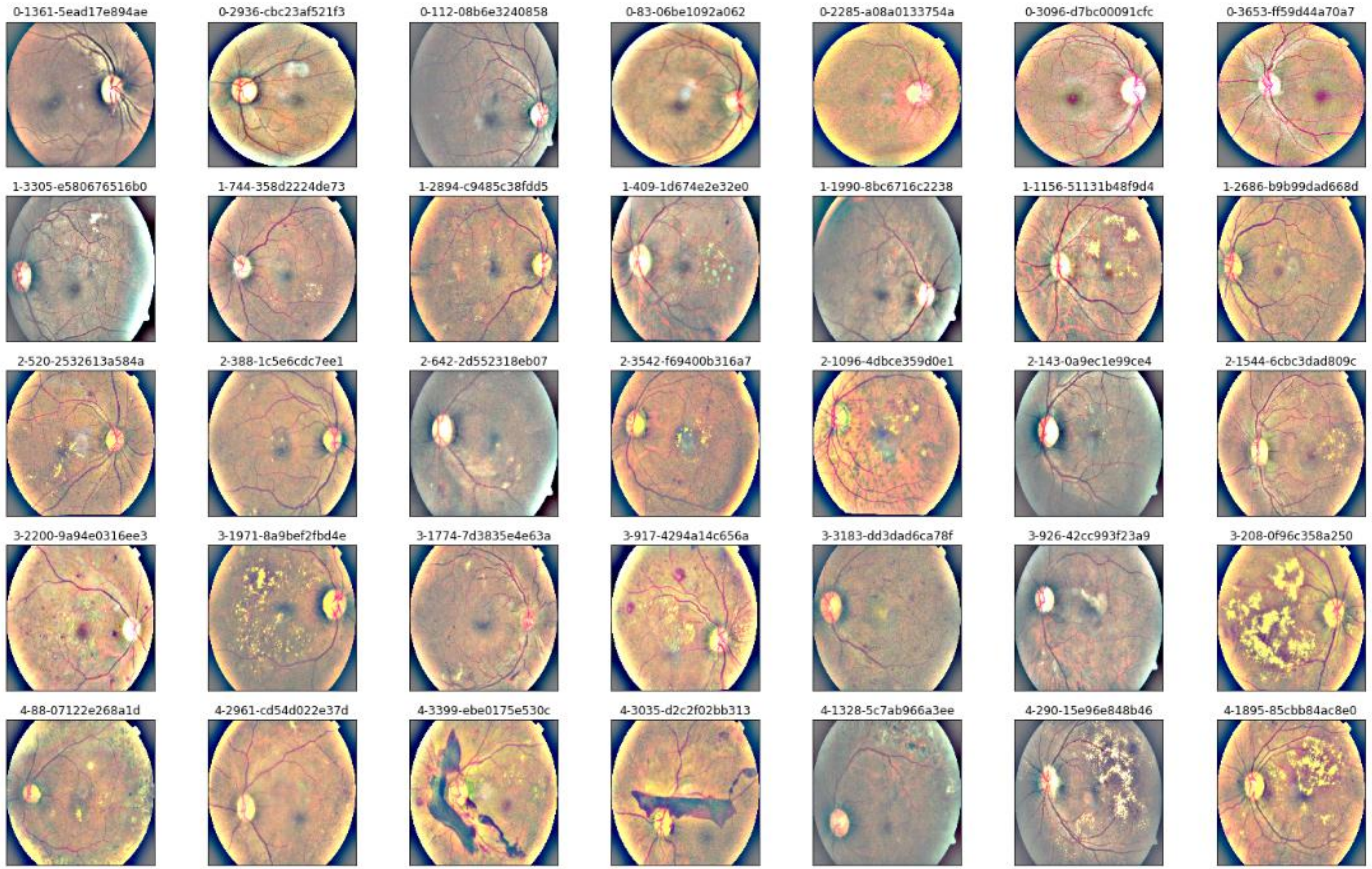
The Magic: Ben Graham's Image Preprocessing, It is a preprocessing method developed by Ben Graham in 2015 as part of then Eyepacs DR Detection Competition, Since then this has been adopted as the world-wide standard for image preprocessing of Retino-Fundal Images.



Label: 0-1361-5ead17e894ae
Label: 0-2936-cbc23af521f3
Label: 0-112-08b6e3240858
Label: 0-83-06be1092a062
Label: 0-2285-a08a0133754a
Label: 1-3305-e580676516b0
Label: 1-744-358d2224de73
Label: 1-2894-c9485c38fdd5
Label: 1-409-1d674e2e32e0
Label: 1-1990-8bc6716c2238
Label: 2-520-2532613a584a
Label: 2-388-1c5e6cdc7ee1
Label: 2-642-2d552318eb07
Label: 2-3542-f69400b316a7
Label: 2-1096-4dbce359d0e1
Label: 3-2200-9a94e0316ee3
Label: 3-1971-8a9bef2fbd4e
Label: 3-1774-7d3835e4e63a
Label: 3-917-4294a14c656a
Label: 3-3183-dd3dad6ca78f
Label: 4-88-07122e268a1d
Label: 4-2961-cd54d022e37d
Label: 4-3399-ebe0175e530c
Label: 4-3035-d2c2f02bb313
Label: 4-1328-5c7ab966a3ee

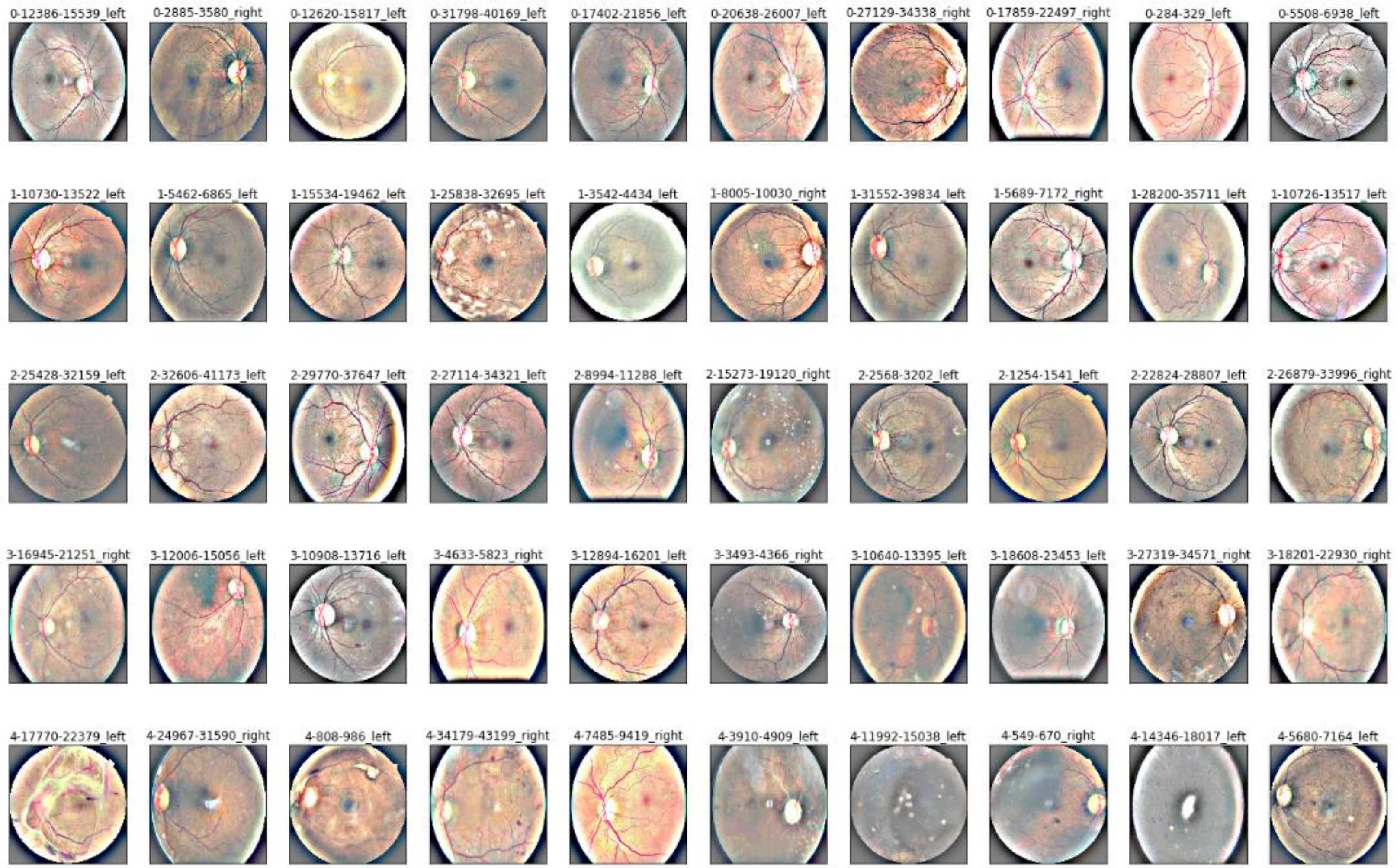
Preprocessed Colour images using Ben Graham’s Technique,

Eyepacs 1



Preprocessed Colour images using Ben Graham's Technique,

Eyepacs 2



Transfer Learning

- The aim of this study is to produce a Robust model for DR detection over the 5 classes to produce a model which is robust as well as is able to generalize well when faced with different kinds of images taken in different conditions with different kinds of devices.
- All the works stated earlier are based on deep learning models and they generally use Precision,Recall,Specificity and sensitivity as metrics to measure and compare the performance of the produced model.In this study, our main focus will be to optimize AUC,and more specifically Quadratic Weighted Kappa or QWK.Quadratic Weighted Kappa or QWK has been one metric ignored by quite a few researchers because Quadratic Weighted Kappa or QWK obtained in DR detection usually is quite less around 0.83-0.88 and as this is a multi-class classification problem QWK is a metric that gives very accurate measure becasue simply of the fact how it calculates the score e.g if model predicts the class as 4 but the actual class is 3 penalty will be less in that proportion but if actual class is 0 and model predicts 4 we are way off and penalty on the QWK is proportional to this.
- Thus QWK has now been made a standard by eyepacs and many organisations to measure the Robustness of the Models produced for the task of DR detection and has also been used as a metric to evaluate various other models giving a uniform scale to measure other model performance as well as its robustness and generalization capabilities.
- For this task we have used the following models:
- As it is already known transfer learning does give really good results when done sensibly, so our starting approach was to try out pre-trained models which included:
- **Resnet 50,EfficientNet B5,B6,B4,B7,Res-Next 32x4d.**
- Finally we produce an ensemble of all the above models.

Quadratic Weighted Kappa

Kappa is *Chance adjusted index* for reliability of categorical measurements. This means that, it accounts for amount of agreement between raters that can be expected to have occurred due to chance (i.e., random guessing). This is unlike accuracy where you can get relatively fair score by intelligently random guessing. Here 3 matrices are involved:

- N×N(N is number of categories) histogram matrix **O**, where each element e_{ij} of O corresponds to the number of observations that received a category i by A and a category j by B. In our case, N=5 so O is 5×5 matrix. Each element e_{ij} will represent count of images that recieved category i by A(say human) and category j by B(our models). So greater the number in diagonal, greater good.
- N×N weights matrix **W**, where each element is calculated using distance between ratings. More on this later.
- N×N histogram matrix of expected ratings **E**, which is calculated as the outer product between each rater's histogram *vector* of ratings. **E** is normalized so that **E** and **O** have the same sum. Now, each cell in **O** is multiplied by corresponding cell in **W** and sum the results across all the cells. Call this P_o. Same is done for **E**. Call this P_e. Then kappa is calculated as below:

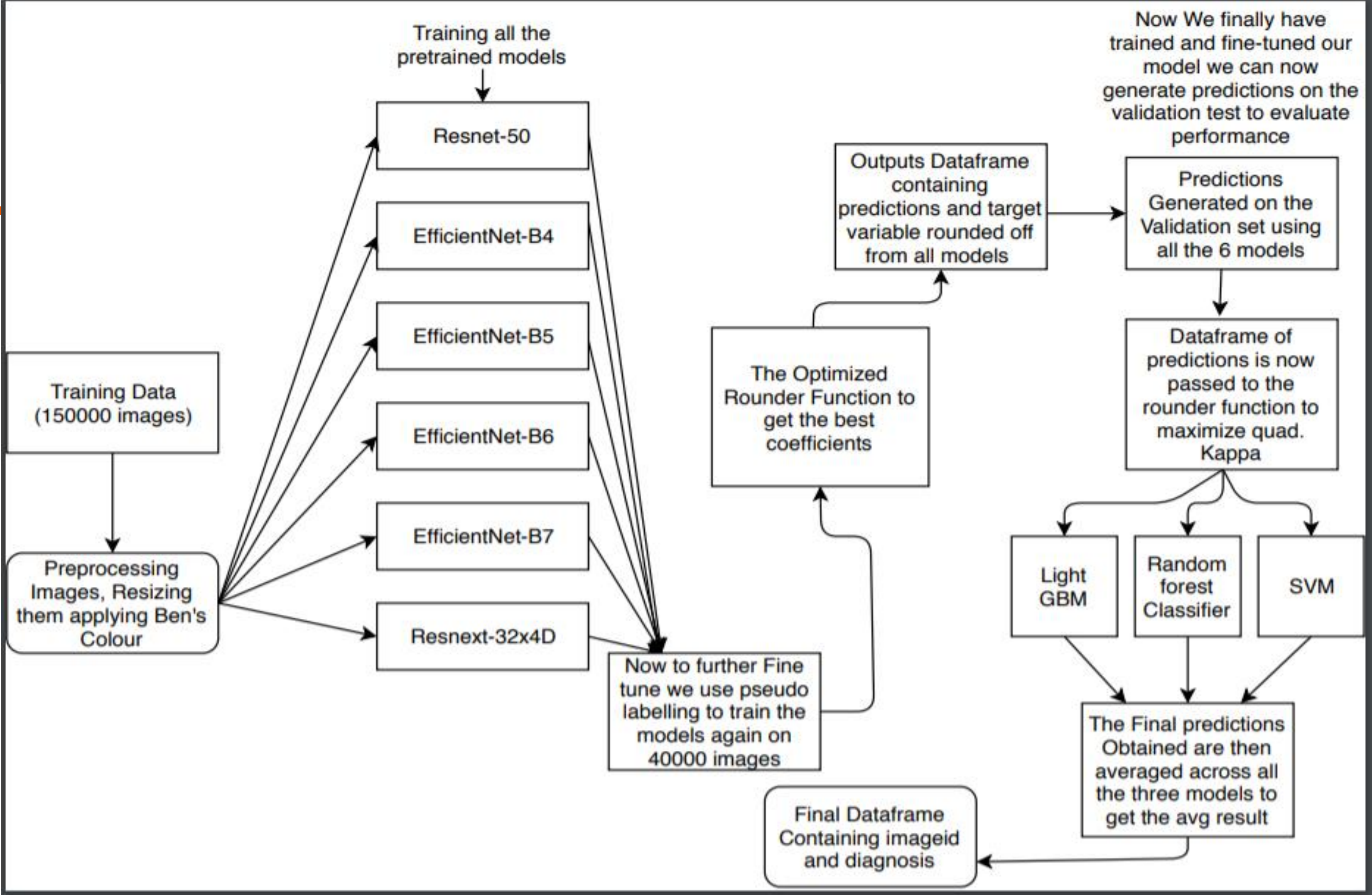
$$\text{kappa}_{LW} = \frac{P_{\text{observed}} - P_{\text{expected}}}{1 - P_{\text{expected}}}$$

$$\text{weight} = 1 - \frac{(\text{distance})^2}{(\text{maximum possible distance})^2}$$

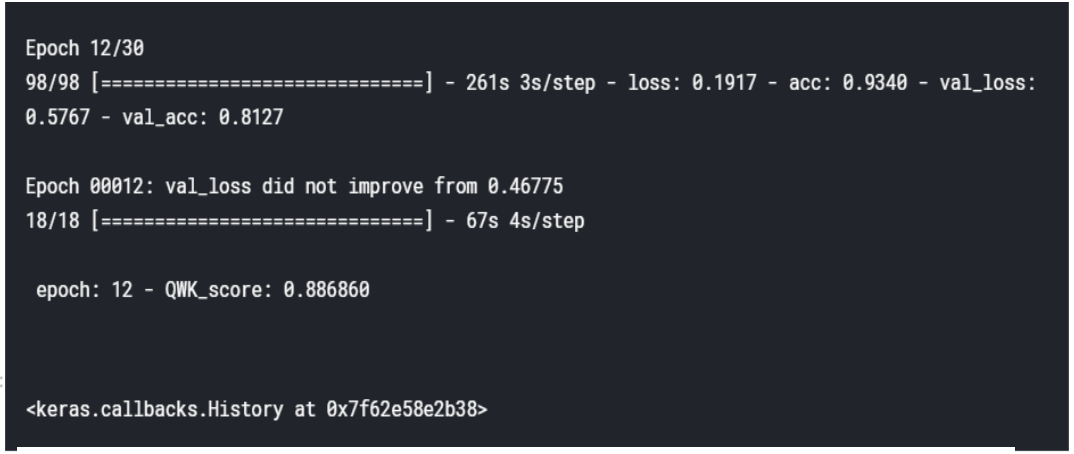
Training, Validation and Testing:

- We Train all the Pretrained models again on 150000 images by unfreezing the weights in the backbone layers of the Network.
- **We Trained the following models:**
- Efficientnet B7,B6,B5,B4.
- Resnet 50.
- Resnext 32x4d.
- **For all the models Following Data Augmentations were used:**
- do_flip
- flip_vert
- Max_zoom.
- Rotate-60,90,120.
- Differential Learning rates were used using FAST.AI API and its Learner function.
- We used RAdam as the Optimizer.
- Batch Size for Efficient Nets was kept at -84 and for Resnets-120 and for resnext-124.
- All the Images were Resized to 3 image categories as part of the Technique called Progressive resizing as follows:
- 240,360,440
- All the models were each trained on images with 240 the 360 and 440 thus in total 3 models for each model hence in total 18 models . 1 model is trained across all the three image sizes to complete its training.

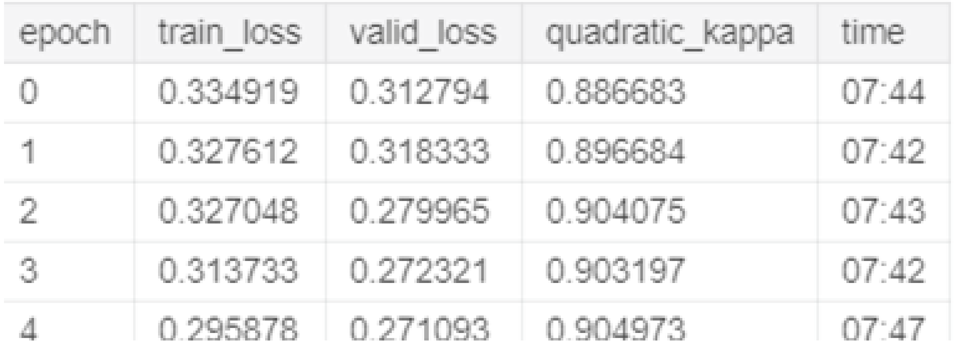
The proposed Model Framework


Training all the pretrained models
Training Data (150000 images)
Preprocessing Images, Resizing them applying Ben's Colour
Resnet-50
EfficientNet-B4
EfficientNet-B5
EfficientNet-B6
EfficientNet-B7
Resnext-32x4D
Now to further Fine tune we use pseudo labelling to train the models again on 40000 images
The Optimized Rounder Function to get the best coefficients
Outputs Dataframe containing predictions and target variable rounded off from all models
Now We finally have trained and fine-tuned our model we can now generate predictions on the validation test to evaluate performance
Predictions Generated on the Validation set using all the 6 models
Dataframe of predictions is now passed to the rounder function to maximize quad. Kappa
Light GBM
Random forest Classifier
SVM
The Final predictions Obtained are then averaged across all the three models to get the avg result
Final Dataframe Containing imageid and diagnosis

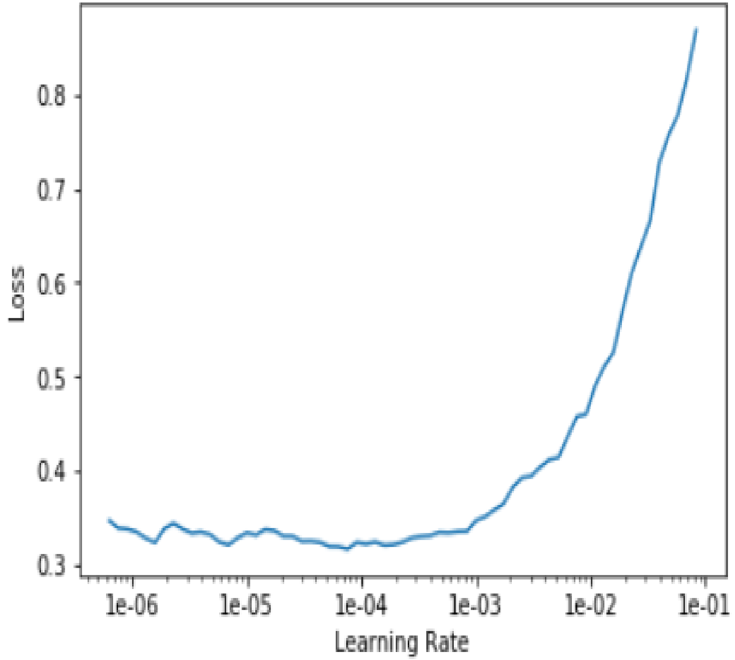
Resnets and Results



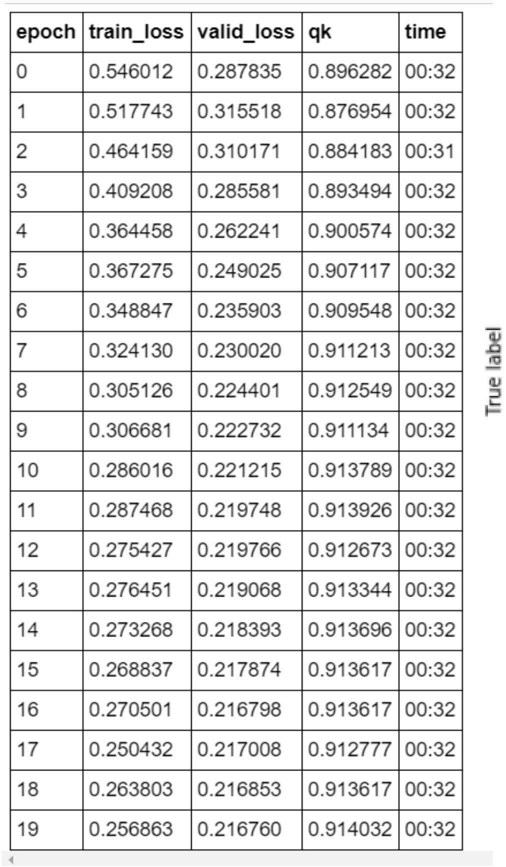
```
Epoch 12/30
98/98 [==============================] - 261s 3s/step - loss: 0.1917 - acc: 0.9340 - val_loss:
0.5767 - val_acc: 0.8127

Epoch 00012: val_loss did not improve from 0.46775
18/18 [==============================] - 67s 4s/step

 epoch: 12 - QWK_score: 0.886860



<keras.callbacks.History at 0x7f62e58e2b38>
```



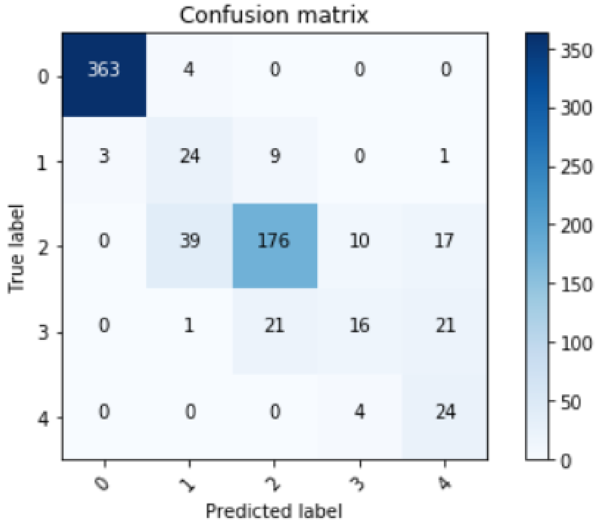
epoch	train_loss	valid_loss	quadratic_kappa	time
0	0.334919	0.312794	0.886683	07:44
1	0.327612	0.318333	0.896684	07:42
2	0.327048	0.279965	0.904075	07:43
3	0.313733	0.272321	0.903197	07:42
4	0.295878	0.271093	0.904973	07:47



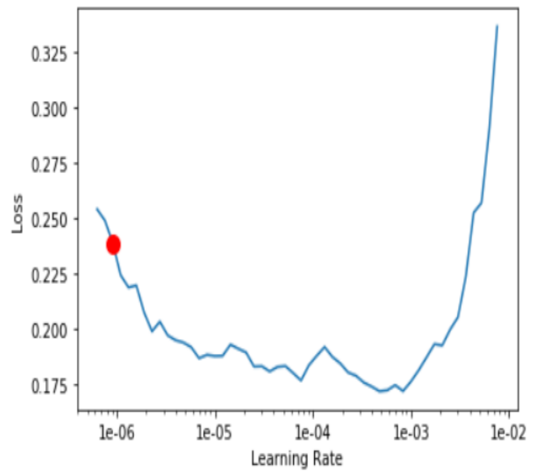
Efficientnet-B4 Results



epoch	train_loss	valid_loss	qk	time
0	0.546012	0.287835	0.896282	00:32
1	0.517743	0.315518	0.876954	00:32
2	0.464159	0.310171	0.884183	00:31
3	0.409208	0.285581	0.893494	00:32
4	0.364458	0.262241	0.900574	00:32
5	0.367275	0.249025	0.907117	00:32
6	0.348847	0.235903	0.909548	00:32
7	0.324130	0.230020	0.911213	00:32
8	0.305126	0.224401	0.912549	00:32
9	0.306681	0.222732	0.911134	00:32
10	0.286016	0.221215	0.913789	00:32
11	0.287468	0.219748	0.913926	00:32
12	0.275427	0.219766	0.912673	00:32
13	0.276451	0.219068	0.913344	00:32
14	0.273268	0.218393	0.913696	00:32
15	0.268837	0.217874	0.913617	00:32
16	0.270501	0.216798	0.913617	00:32
17	0.250432	0.217008	0.912777	00:32
18	0.263803	0.216853	0.913617	00:32
19	0.256863	0.216760	0.914032	00:32



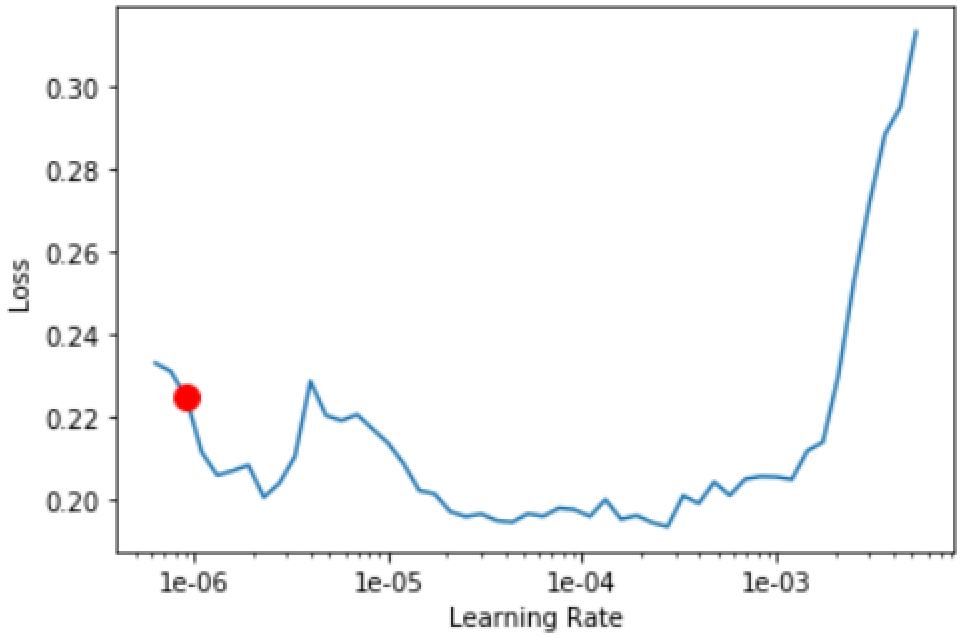
```
LR Finder is complete, type {learner_name}.recorder.plot() to see the graph.
Min numerical gradient: 9.12E-07
Min loss divided by 10: 4.79E-05
```



EfficientNet-B5 Results

epoch	train_loss	valid_loss	qk
0	0.174309	0.215754	0.914902
1	0.194766	0.217427	0.910209
2	0.189688	0.218218	0.911894
3	0.201021	0.218246	0.910874

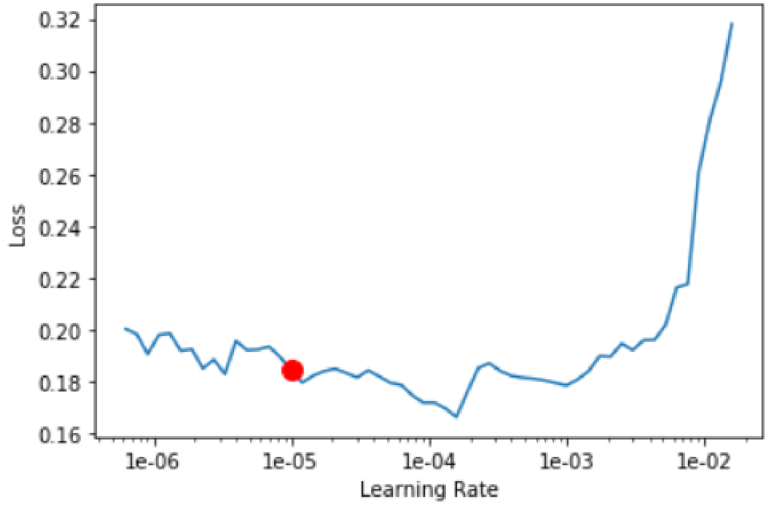
```
Min numerical gradient: 9.12E-07
Min loss divided by 10: 2.75E-05
```

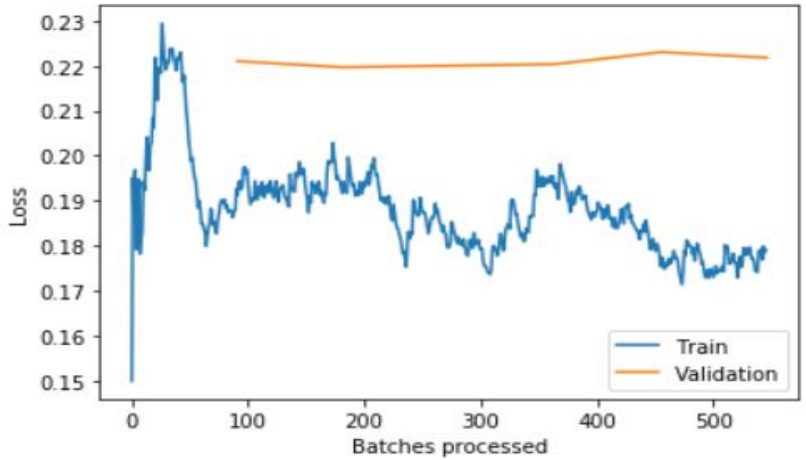


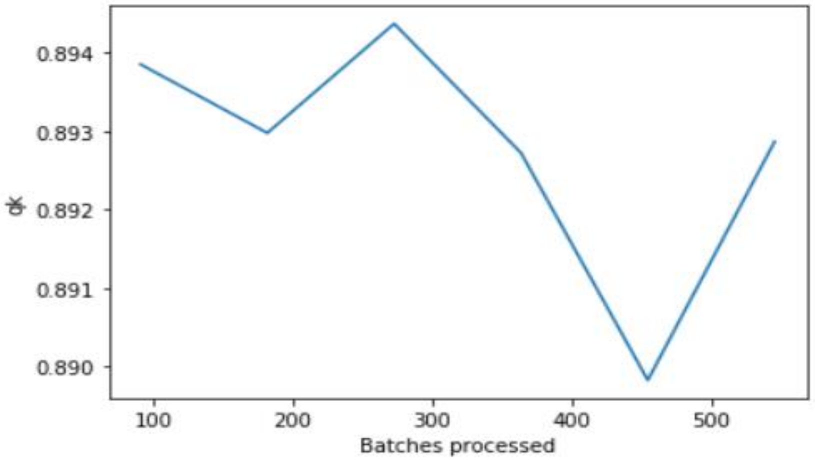
EfficientNet-B6 Results

epoch	train_loss	valid_loss	qk
0	0.181005	0.200695	0.901204
1	0.179247	0.201257	0.901269
2	0.176989	0.201248	0.900760
3	0.176912	0.200754	0.901269

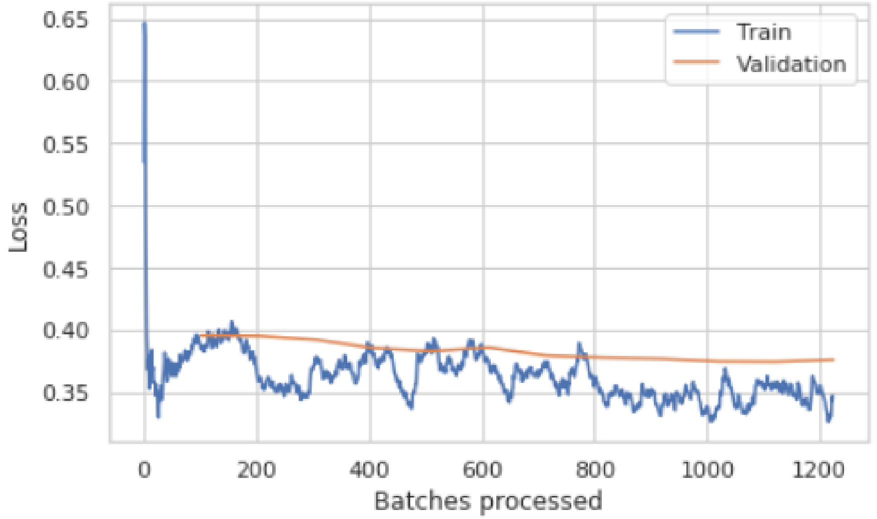
```
Min numerical gradient: 1.00E-05
Min loss divided by 10: 1.58E-05
```

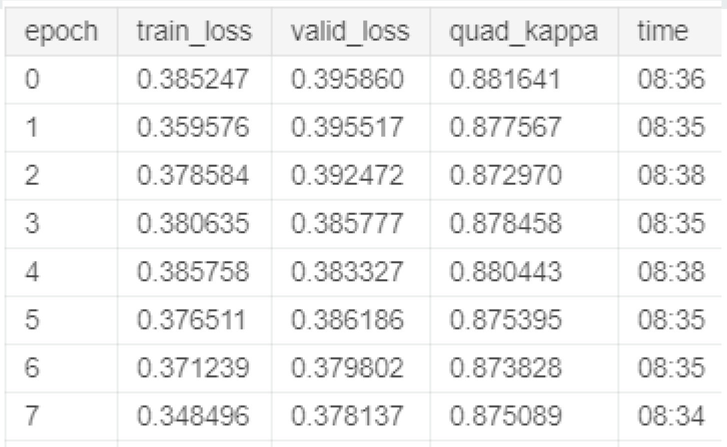




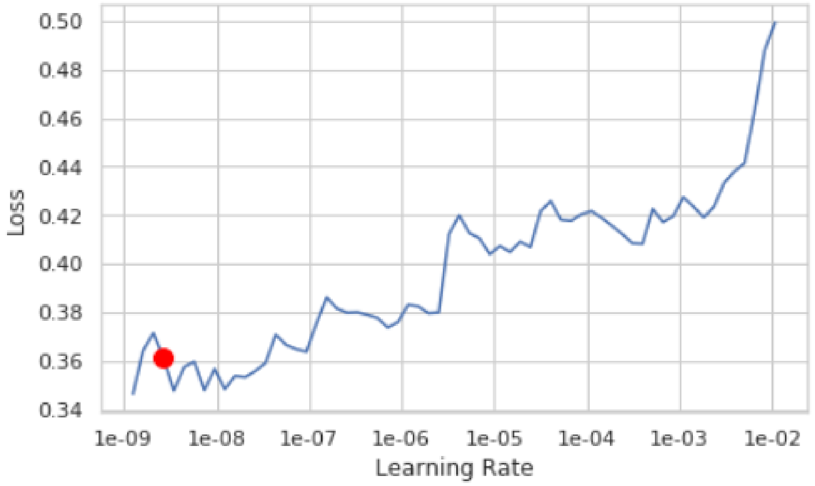


Res-Next 32x4D Results:



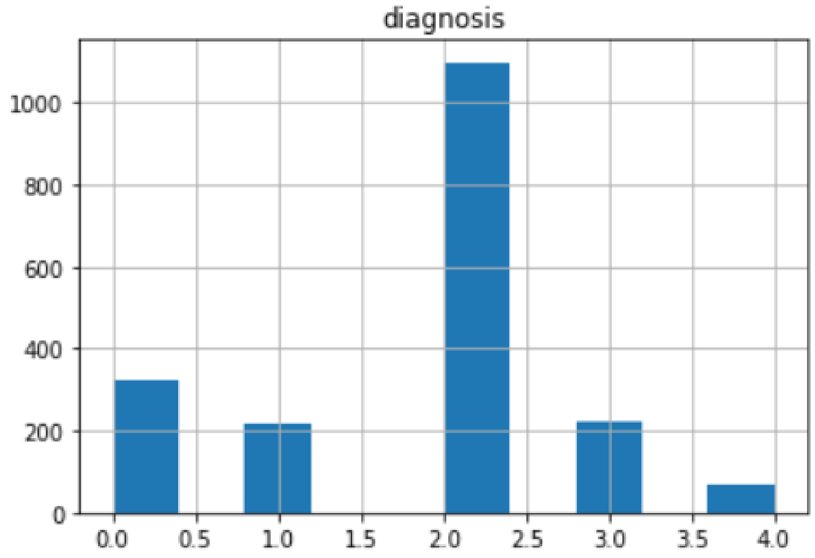


epoch	train_loss	valid_loss	quad_kappa	time
0	0.385247	0.395860	0.881641	08:36
1	0.359576	0.395517	0.877567	08:35
2	0.378584	0.392472	0.872970	08:38
3	0.380635	0.385777	0.878458	08:35
4	0.385758	0.383327	0.880443	08:38
5	0.376511	0.386186	0.875395	08:35
6	0.371239	0.379802	0.873828	08:35
7	0.348496	0.378137	0.875089	08:34



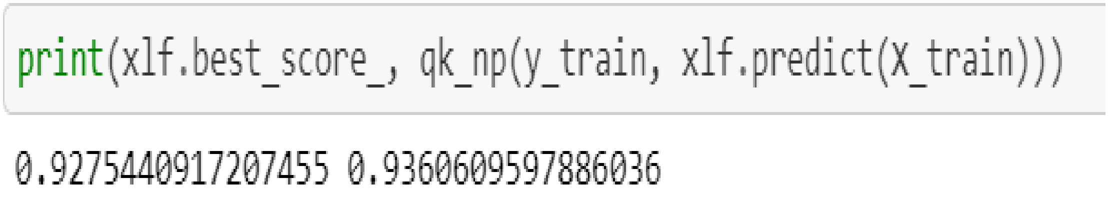
Final Predictions and Results

Final predictions on the Test set and Quadratic kappa value.



Quadratic kappa : 0.9471472160819255

Quadratic kappa and best score by stacking efficientnet B5,B6,B7,B4.
1st value indicates avg quad kappa just using LGbm and the2nd value is avg quad kappa using LGbm,SVM .

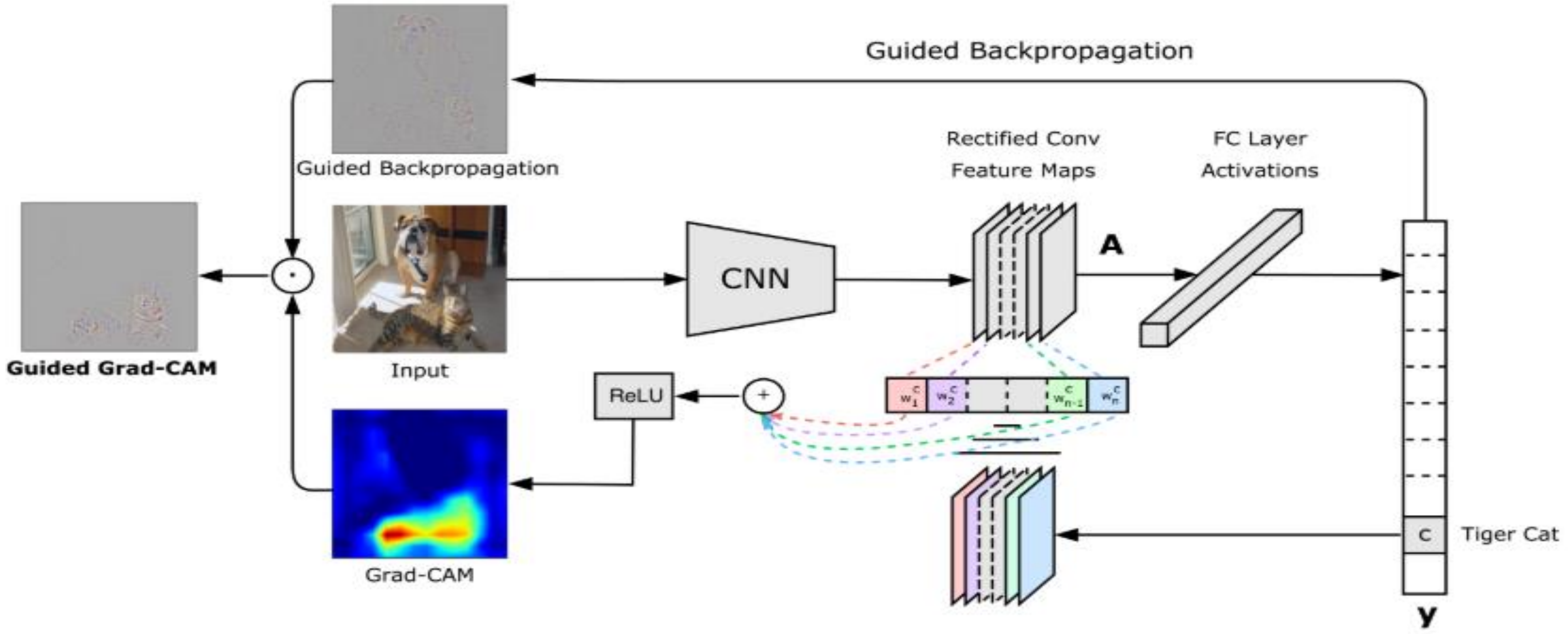


```
print(xlf.best_score_, qk_np(y_train, xlf.predict(X_train)))
```

```
0.9275440917207455 0.9360609597886036
```

Model Interpretation and Analysis

Results are great but what is our model actually seeing. Interpreting and analysing the model is one of the most important aspects of creating a robust model that generalizes well because we need to understand where our model is failing and why. So for this we use a technique called GRAD-CAM which was engineered by researchers to understand the activation values of feature maps in the last Convolutional layer just before Pooling is applied.



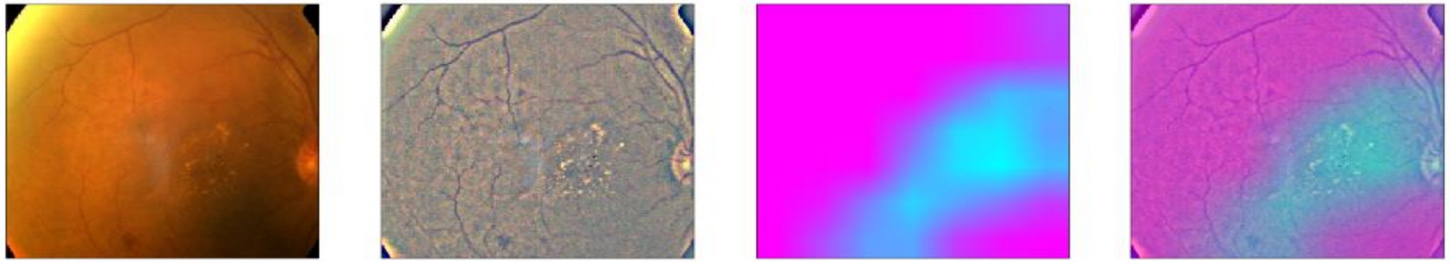
GRAD-CAM(Gradient-Weighted class actvation mapping).

Basic idea behind GRAD-CAM:

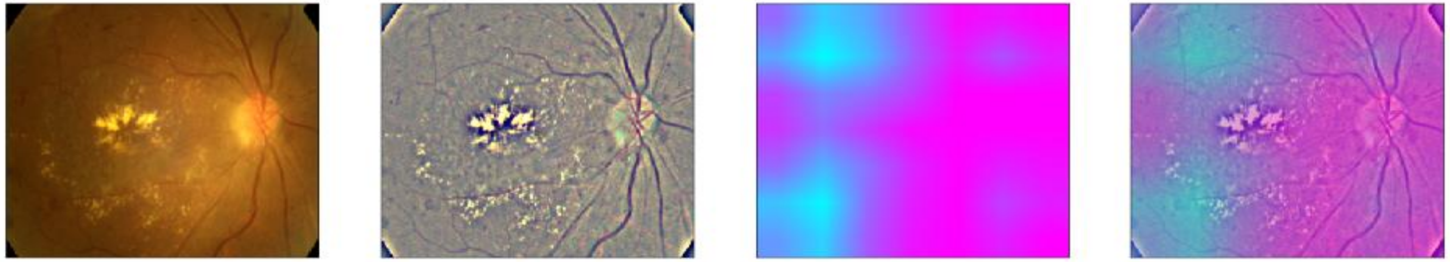
- We believe that most important spatial information come from the 3D-tensor of the *last convolutional layer* (just before GlobalPooling layer), which is the nearest spatial information flowing to the last FC layer.
- For each channel of this 3D-tensor, each activated pixel region represent important features (e.g. blood vessel / scab / cotton wool) of the input image. Note that some features are important to determine class 0 (perfectly fine blood vessel), some features are important to determine class 4 (big cotton wools). Normally, we expect each channel to capture different set of features
- To emphasize features which finally affected the final prediction, we calculate the **gradient of the final predicted class with respect to each feature.** If that feature is important to this class, it should have high gradient (i.e. increase the value of this feature, the prediction confidence increases)
- Therefore, we multiply the activated values of this 3D-tensor and gradients together, to obtain the visualized heatmap for each channel. Note that we have multi-channels, and each channel usually have multi-features.
- Finally, we combine heatmaps of all channels using simple average, and remove negative value (the ReLu step in the above picture) to obtain the final heatmap.

GRAD-CAM analysing Resnets

```
test pic no.1
raw output from model :
1.000 0.966 0.520 0.268 0.109
```

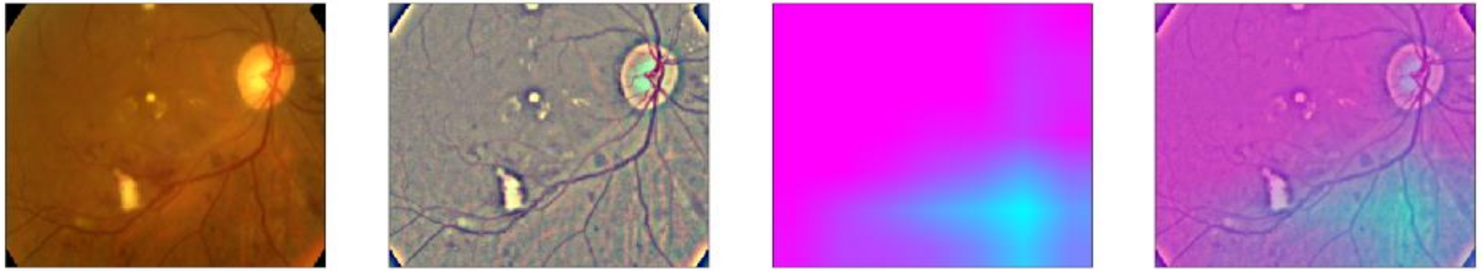


```
test pic no.2
raw output from model :
0.998 0.999 0.999 0.931 0.153
```

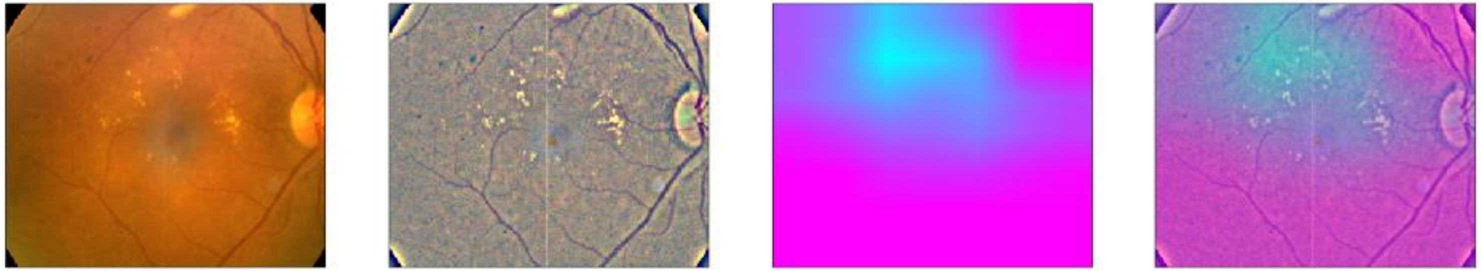


GRAD-CAM analysing Resnets

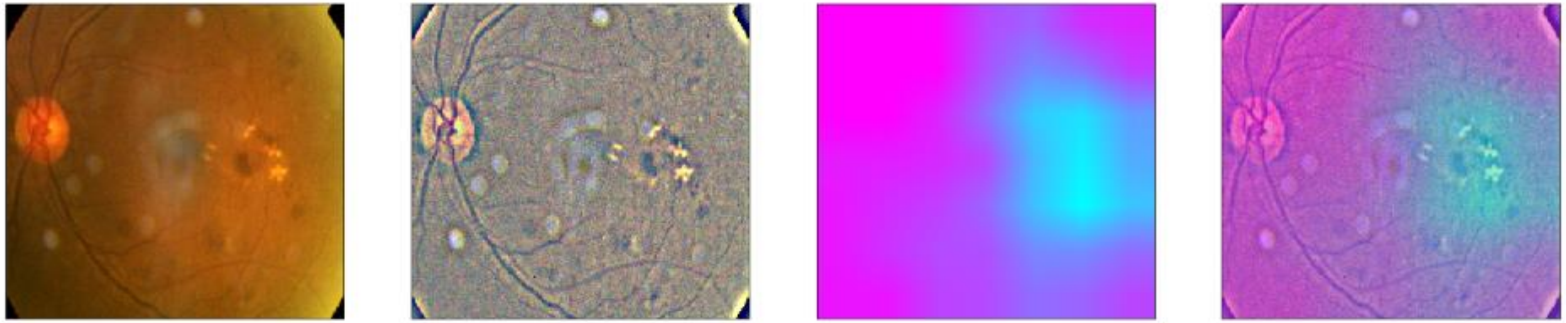
```
test pic no.3
raw output from model :
0.999 0.998 0.991 0.812 0.076
```



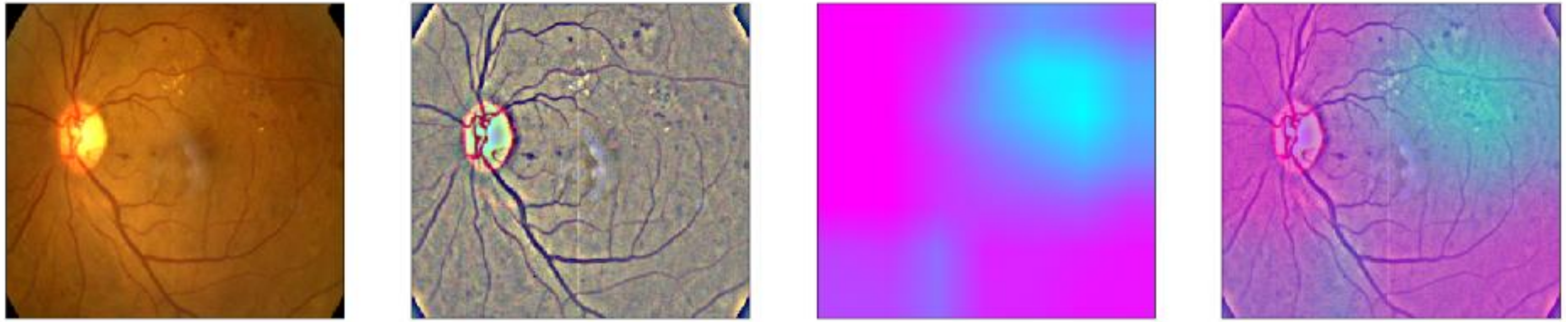
```
test pic no.4
raw output from model :
0.998 0.998 0.758 0.031 0.017
```



GRAD-CAM analysing Efficient-Net and its efficiency

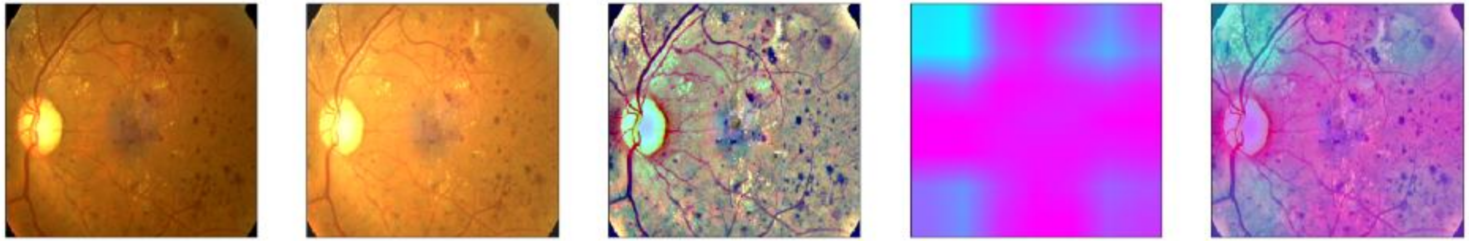


```
test pic no.6
raw output from model :
0.994 0.995 0.959 0.172 0.042
```

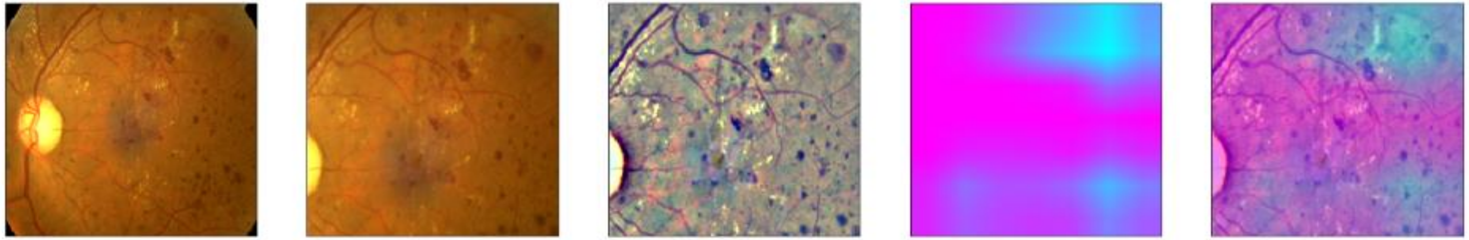


Analysing efficientnet with augmentations

```
test pic no.2 -- augmentation: brightness or contrast
raw output from model :
0.998 1.000 1.000 0.946 0.085
```

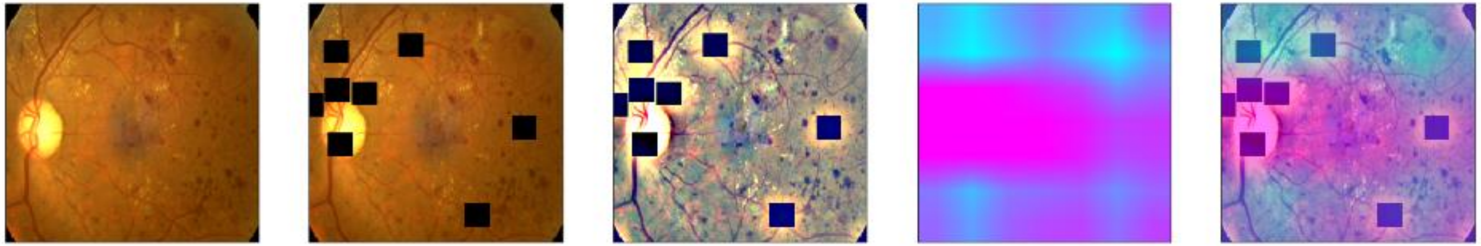


```
test pic no.3 -- augmentation: crop and resized
raw output from model :
0.997 1.000 0.999 0.897 0.063
```

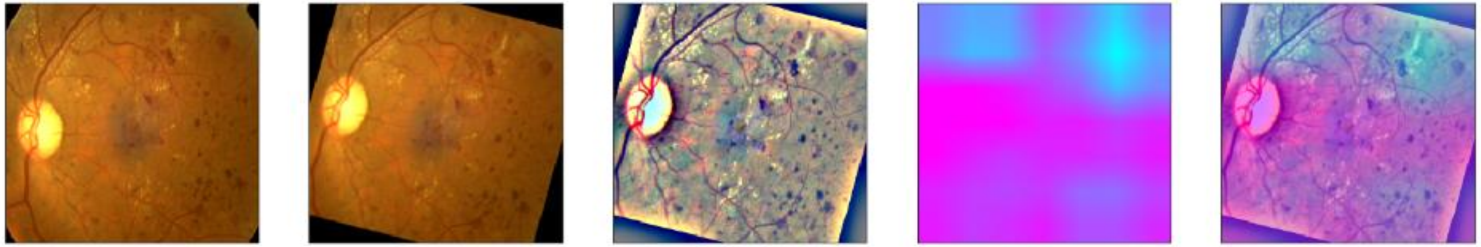


Analysing efficientnet with augmentations

```
test pic no.4 -- augmentation: CutOut
raw output from model :
0.996 1.000 1.000 0.974 0.094
```

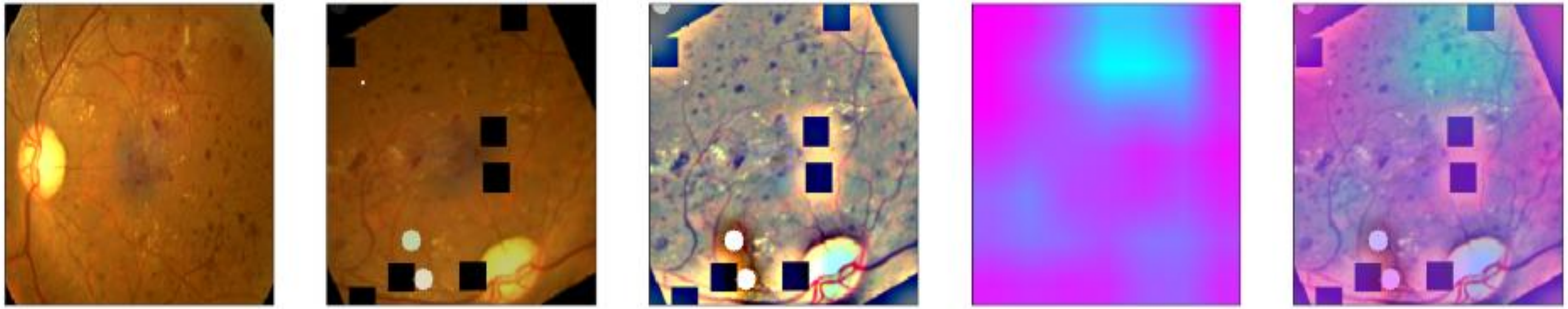


```
test pic no.5 -- augmentation: rotate or flip
raw output from model :
0.998 1.000 1.000 0.979 0.095
```



All augmentations and Analysis

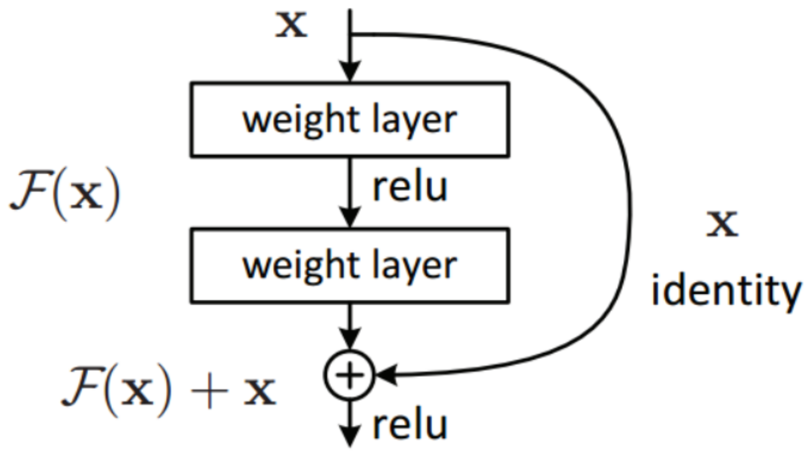
```
test pic no.6 -- augmentation: Everything Combined
raw output from model :
0.996 1.000 1.000 0.990 0.143
```



Final Results:

S.NO	Authors	Year	Technique	QWK	AUC-ROC Metric	Dataset Used
1	Varun Gulshan, PhD; Lily Peng, MD, PhD; Marc Coram, PhD; Martin C. Stumpe, PhD; Derek Wu, BS; A	2016	Inception V-3.+image processing.	0.8851(5)	0.991(3 vs All), 0.993(Avg-AUC across all datasets)	Eyepacs-1(15,000, Messidor-2(2000), Google's own DB-1,20,000, APTOS(13000)
2	Daniel Shu Wei Ting, MD, PhD[1,2]; Carol Yim-Lui Cheung, PhD[1,3]; Gilbert Lim, PhD[4]; et al	2017	Inception V-3 and V-4 + CLAHE	0.856(5)	0.971(4 vs all), Avg AUC-0.951	Eyepacs-2(76000),Messidor-2(200)
3	Bellemo, Valentina, et al.	2019	VGG-net	0.831(3)	0.973(2 vs all) and	Eyepacs-2(71000),and Zambian eye society(18000)
4	Wei Zhang, Jie Zhong, Shijun Yang, Zhentao Gao	2019	Two Part ensemble. 1-binary classification then, 2- grading Sytem. 1-Xception,Resnet-50,Inception resnet-v2.	0.8771(4)	0.981(Classification), 0.994(Avg-AUC)	Sichuan Medical Center(15000)
5	Khwaja Wisal, Manoj	2019	EfficientNet+Resnet+Resnext+SVM+LGBM+RF+Image Processing+Pseudo labelling.	0.9417(5)	0.9971(4 vs all) 0.9959(avg AUC)	Eyepacs 1(120000) Eyepacs 2(84000) Messidor-2(2000) APTOS(8000)

Understanding Resnets.



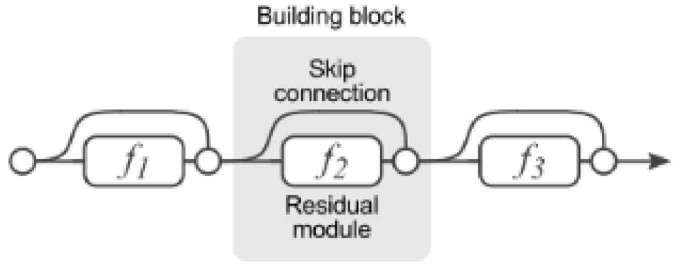
Single Residual Block

$$R(x) = \text{Output} - \text{Input} = H(x) - x$$

Rearranging it, we get,

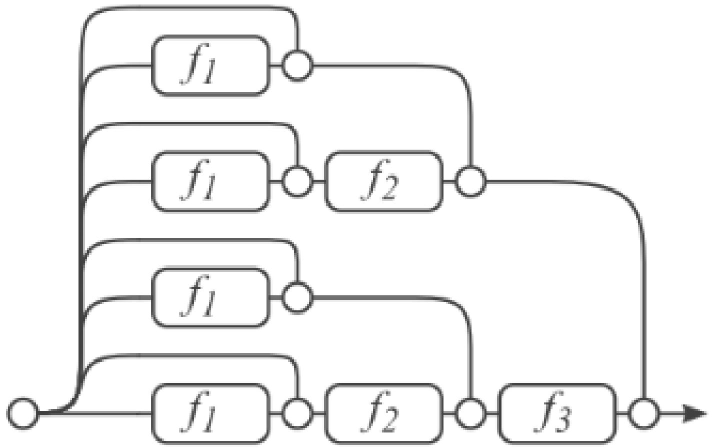
$$H(x) = R(x) + x$$

Resnets are residues forming Ensembles!!!!!!



(a) Conventional 3-block residual network

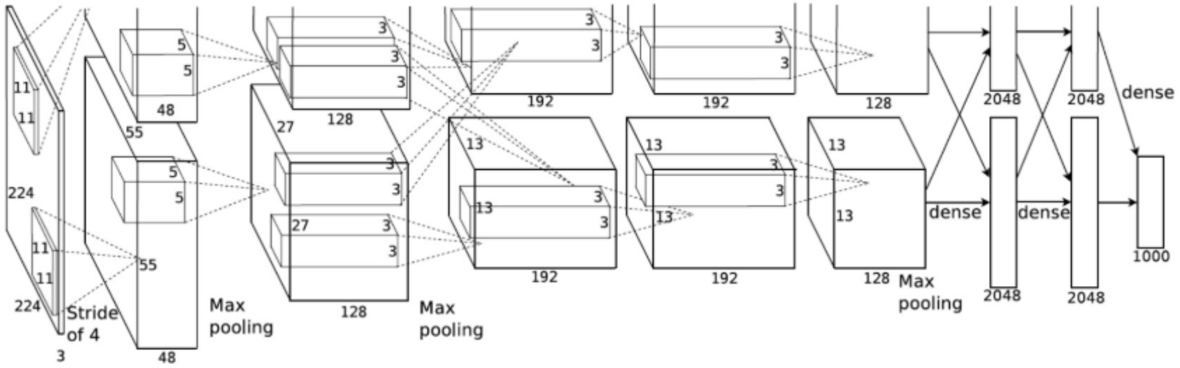
$=$



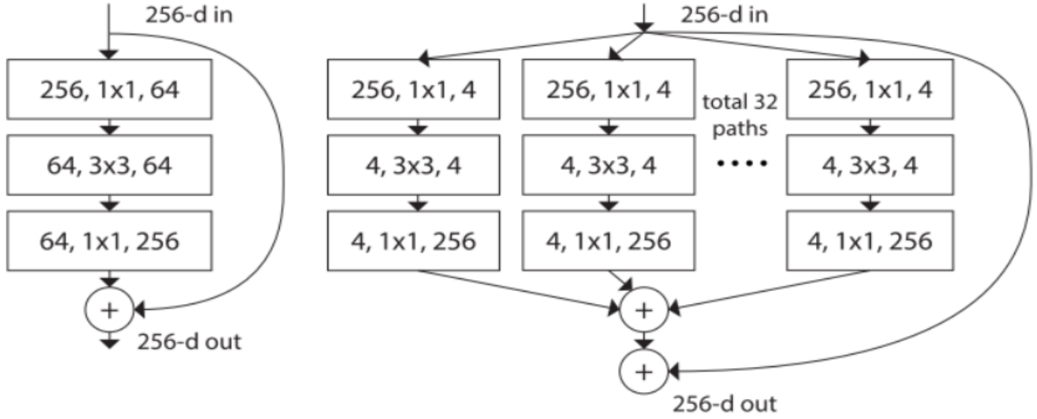
(b) Unraveled view of (a)

For a N-block Residual Network there are 2 to the power N paths for the gradients to flow.

Understanding Resnext:

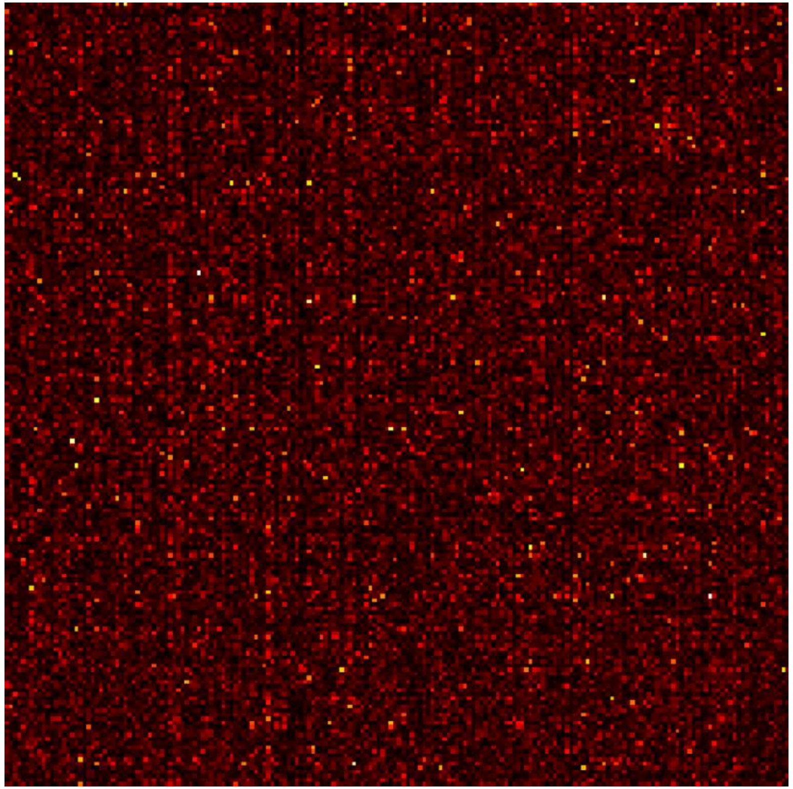


This image is adopted from the AlexNet paper.

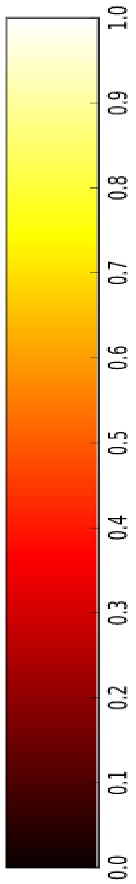


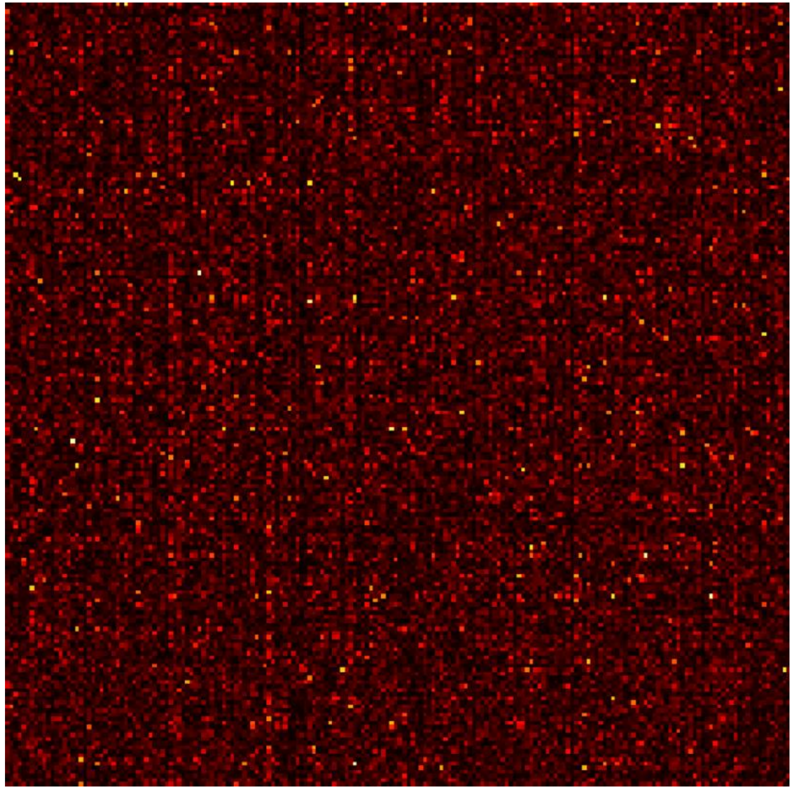
left: a building block of [2], right: a building block of ResNeXt with cardinality = 32

Why Grouping?



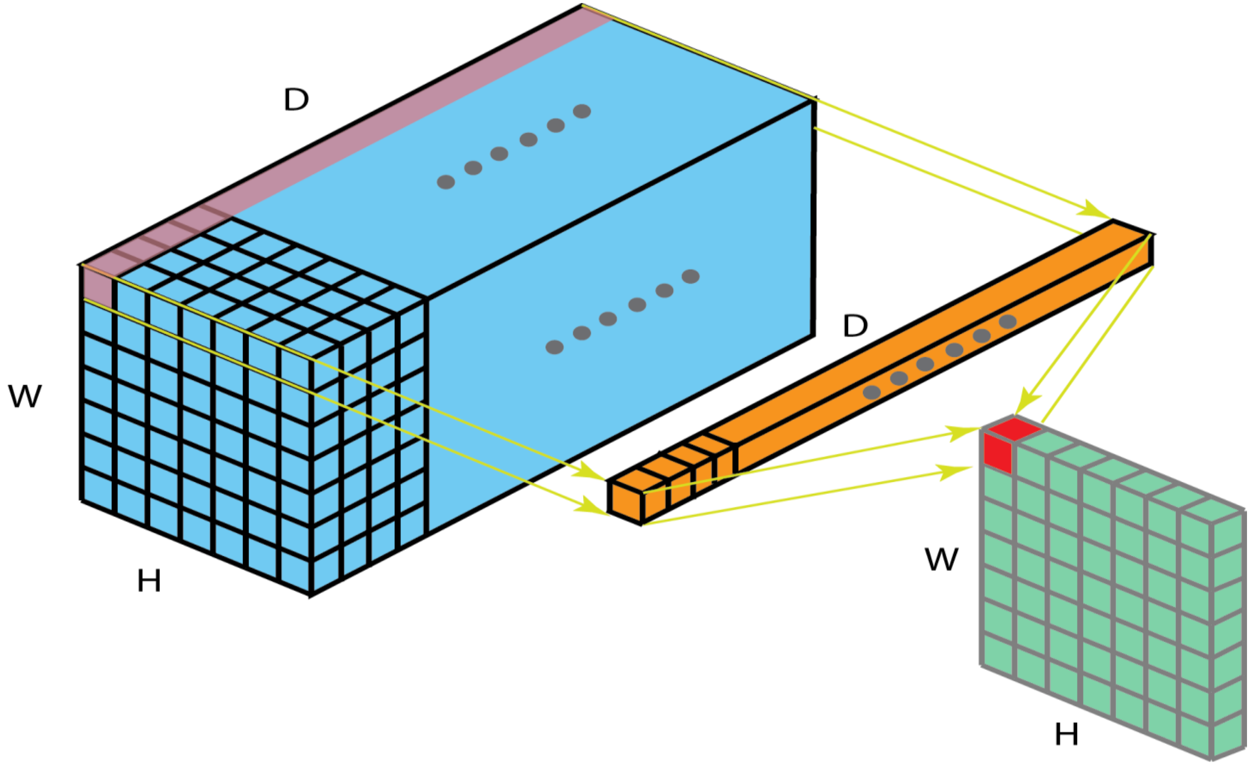
Correlations Between Learned Filters of a Resnext network with groups of 1,2,4,16 Filter Groups





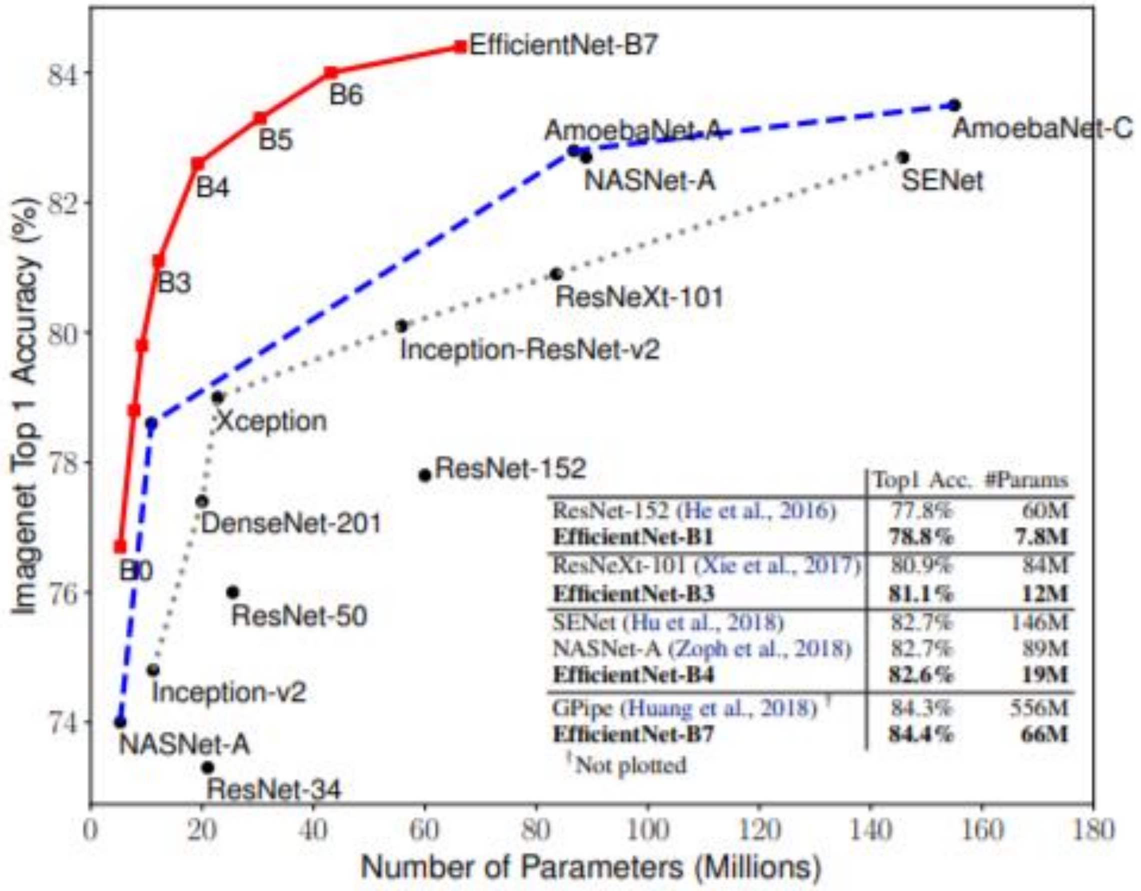
Correlations Between Learned Filters of a Resnet network with no Grouped Convolutions.

The Point-Wise Convolutions. The Workhorse of Modern day CNN's.



1 x 1 convolution, where the filter size is 1 x 1 x D

EfficientsNets are they really Efficient?



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**78.8%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.1%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.6%**	**19M**
GPipe (Huang et al., 2018)[†]	84.3%	556M
EfficientNet-B7	**84.4%**	**66M**

[†]Not plotted

$$\begin{aligned}
\text{depth: } & d = \alpha^{\phi} \\
\text{width: } & w = \beta^{\phi} \\
\text{resolution: } & r = \gamma^{\phi} \\
\text{s.t. } & \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \\
& \alpha \geq 1, \beta \geq 1, \gamma \geq 1
\end{aligned}$$

Understanding the Efficiency:

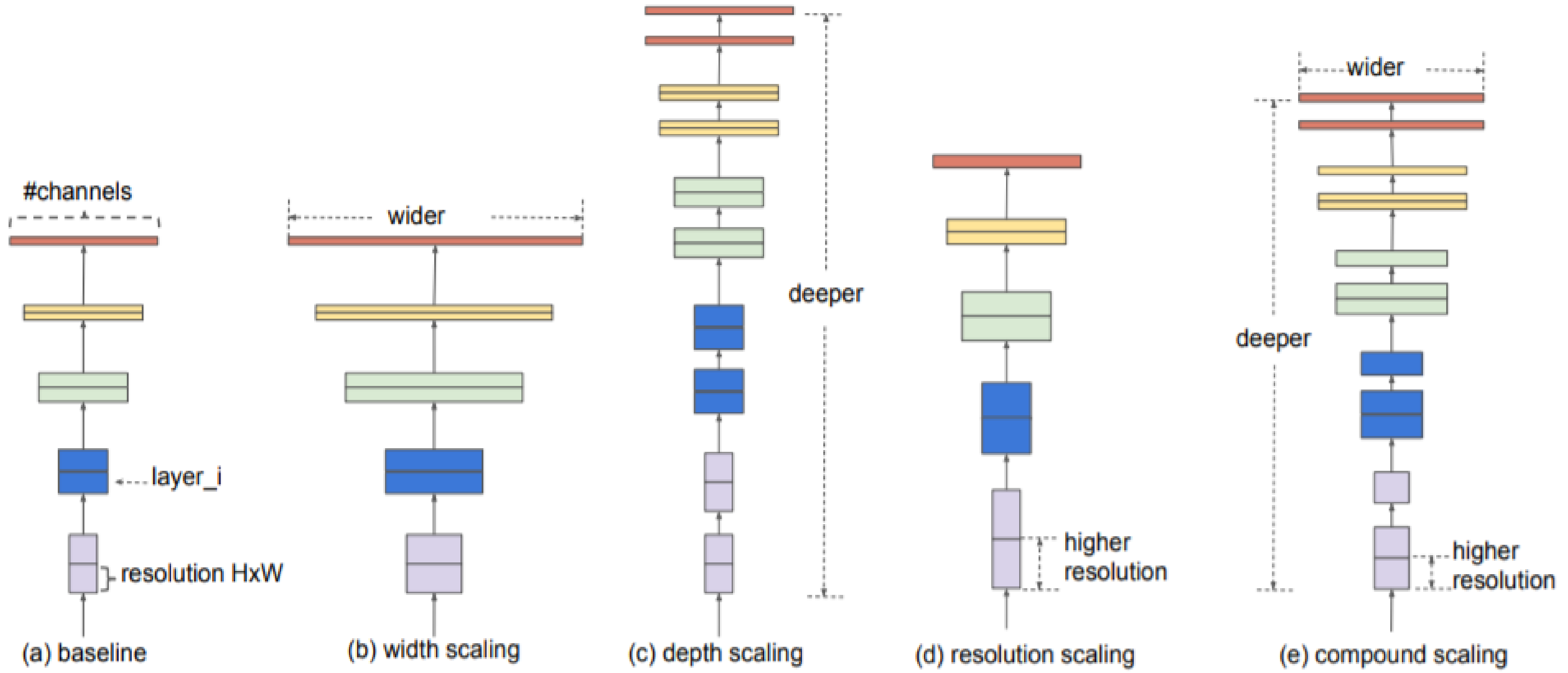


Figure 2. **Model Scaling.** (a) is a baseline network example; (b)-(d) are conventional scaling that only increases one dimension of network width, depth, or resolution. (e) is our proposed compound scaling method that uniformly scales all three dimensions with a fixed ratio.