**Blindness Detection on Diabetic Retinopathy Images**

Team_4: Xiaoyu Wan, Yonghao Duan, Yu Sun

Diabetic retinopathy (DR) is a leading cause of blindness and affects up to 80 percent of those who have had diabetes for 20 years or more [1]. Epidemiology study has shown that about 4.1 million US adults (>=40 years old) have DR, and 1 out of 12 patients has advanced, vision-threatening retinopathy [2]. Thus, early diagnosis of DR enables early treatment, which alleviates retinopathy progressing. Nowadays, DR can be diagnosed by non-invasive eye retinal photography, or also called fundus photography. Taking advantage of convolutional neural networks (CNNs), predicting the severity of DR from thousands of images automatically becomes possible.

## Dataset Description

Our data were generated by Aravind Eye Hospital in India. To detect and prevent the diabetic retinopathy among people living in rural areas where medical screening is difficult to conduct [3]. The image data were collected from multiple clinics in Indian rural areas using fundus photography over an extended period of DR. We acquired two public datasets on Kaggle, one published this year, and a previous one published in 2015 [4]. In our project, we refer them 2019 dataset and 2015 dataset respectively.

For the 2019 dataset, the public dataset size is about 10G, consisting 3,662 images. The public test datasets are another local 1,928 images. The private test set on Kaggle consisting of 20 GB of data across 13,000 images. For the 2015 dataset contains 35,108 images, with the total size about 8.5 GB. Those data have been pre-processed to reduce their size, since the original images were large and hard to manipulate, the processed train datasets are 7GB.Both train data sets are labeled by clinicians with with 0-4 scales, showing the severity of the disease symptoms from 0 as normal, 1- mild, 2-Moderate, 3-Severe, and 4 as proliferate.

## Exploratory Data Analysis (EDA)

We performed exploratory analysis to preview the distribution of labels, and the overview of images in the training sets of 2015 and 2019 dataset. The target is distributed as Figure 1 shows. The largest class is class 0, the population with no DR which takes 49.29% in our dataset. The smallest class is class is class 3, which takes 5% in the dataset. Class 1, 2, and 4 are with percentage 27%, 10% and 8% accordingly. Given the skewed ratio of majority to minority samples of the train dataset, our data is imbalanced and needs to be balanced before modeling.

Before preprocessing, sampling the images from train and test datasets are useful to determine the data transformation methods for data augmentation. We are expecting to see the symptoms as shown in Figure 2, including hemorrhages, aneurysm, hard exudates, etc. By sampling the images from the train label, as Figure 3 shows, we could generate several insights from the data samples: 1) There exists a large variation of image size across the dataset. 2) Some images have the black edges so the area of center is relatively small. 3) The brightness and contrast of images differ in a large scale as two images in label 4 showed. By checking different batches of samples, we found the hard exodus and “cotton wool” spots are easier to identify in the train set, with other symptoms hard to detect directly through the eyes. This informed us to adopt data transformation methods such as resizing, shifting the focus and center of images, adjusting the contrast and brightness of images, to improve the classification results.

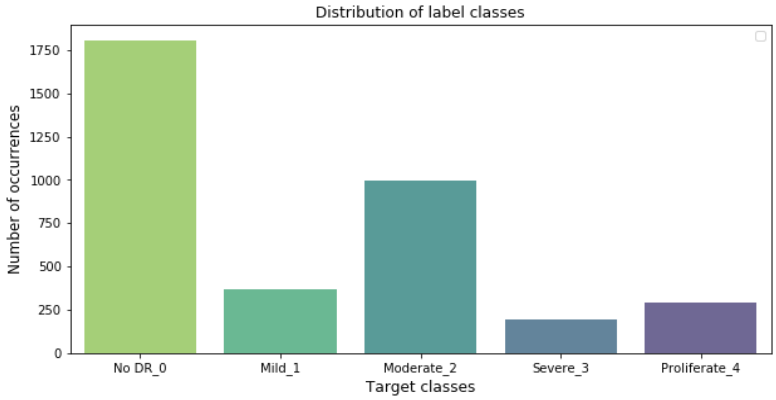



*Figure 1*: Train Set target label distribution (Label 0-4).

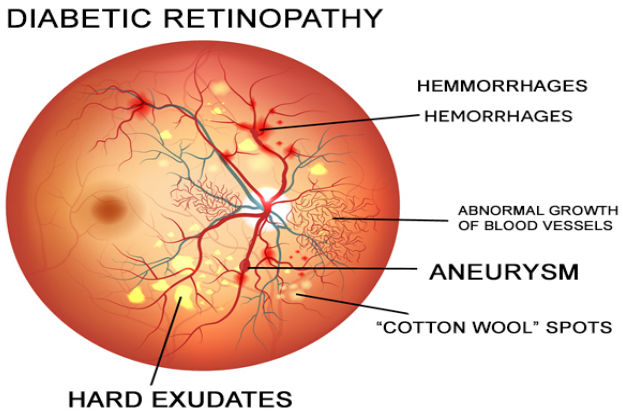



*Figure 2:* The diabetic retinopathy symptoms.

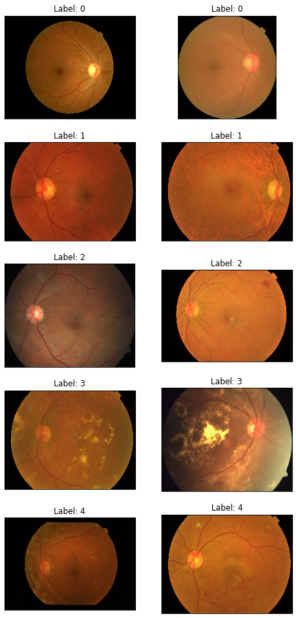



*Figure 3*: Sample images from train dataset (label 0 - 4).

## Data Preprocessing

After the exploratory analysis, we identified two critical problems to be addressed are: 1) Data imbalance 2) Data transformation.

1) Data Imbalance

We decided to add the dataset in the 2015 Kaggle competition with the similar problem to augment our original dataset. We first checked the distribution of the 2015 datasets. As Figure 4 below shows the distribution of the label classes. The label classes are in the similar distribution as shown in 2015 class, with class_0 as the largest class, and class_4 as the smallest class. After combined the class from the train dataset from 2015, we had the whole train set label as described below in Table 1. We then randomly sampled the class_0 the same number as shown in 2019, keep the class_4 the same size as in the combined dataset and randomly sample the same number of samples as class_4 for the rest of class. The sampled results are shown in the following table 2. The ratio of the largest class and the smallest classes are now as 3:2, as in Figure 5.

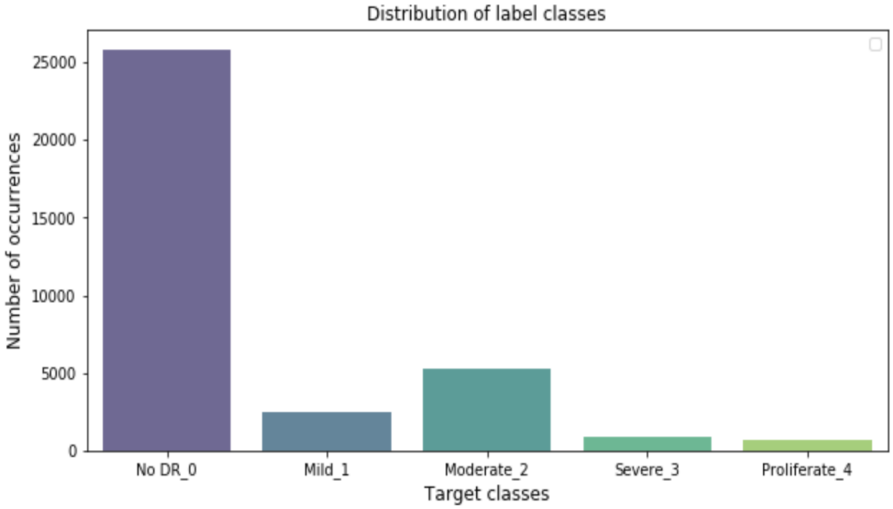



*Figure 4*: 2015 Train Set target label distribution in 2015 data (Label 0-4).

| class | diagnosis | percentage |
|---|---|---|
| 0 | 27607 | 71.21 |
| 2 | 6287 | 16.22 |
| 1 | 2808 | 7.24 |
| 3 | 1065 | 2.75 |
| 4 | 1003 | 2.59 |

*Table 1:* The distribution of label class in the dataset 2015.

| class | diagnosis | percentage |
|---|---|---|
| 0 | 1805 | 31.030 |
| 2 | 1003 | 17.242 |
| 1 | 1003 | 17.242 |
| 3 | 1003 | 17.242 |
| 4 | 1003 | 17.242 |

*Table 2:* The distribution of resampled label class in the combined dataset.

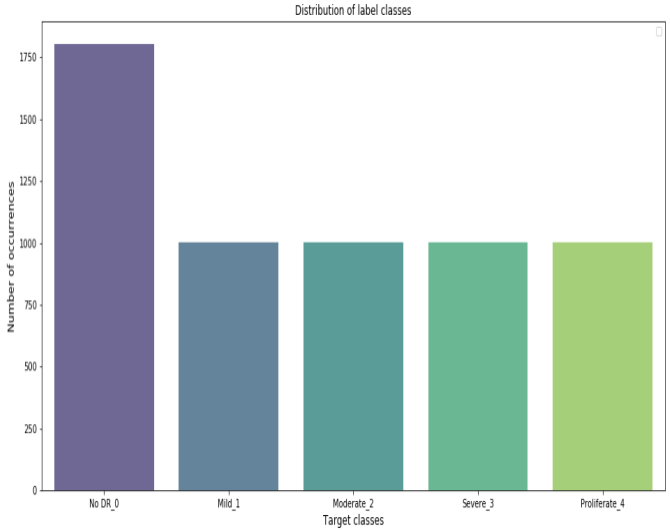



*Figure 5*: Combined label distribution in 2015 data (Label 0-4).

2) Data Augmentation

We conducted research to identify the methods that are relevant to solve the problems of color brightness, size and focus from the website and literature resources [6][7]. A variety of methods from the previous winners and literature on spatial transformation such as random rotation, random affine, random horizontal flip, random vertical flip, and color jitter for brightness & contrast adjustment. Here we used the transformed images to identify the effect after data transformation shown in Figure 6. In our project, we selected several methods from the previous winner and literature and experiment the result.

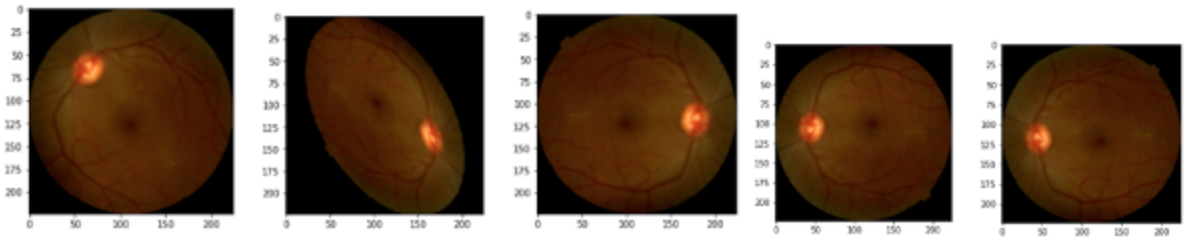


*Figure 6:* Data transformation methods. (From Left to Right: Random rotation, Random affine, Random horizontal flip, Random vertical flip, color jitter).

## Modeling

### Evaluation Metric

In this project, as the Kaggle competition declared to quadratic weighted kappa as the evaluation metric for classification result, and thus we could submit our result in the kernel to gain the test result at last. Quadratic weighted kappa is not a common metric used for classification; however, it is used to measure the amount of agreement between the prediction and true labels. The kappa score ranges between 0 to 1, with 0 as the no agreement between the predicted and true label and the 1 equals the complete agreement between the predicted and true label [8]. The kappa coefficient aims to adjust for the probability of the algorithm and the true labels assigning items to the same category "by chance" based on the assumption that the algorithm and the true labels each have a predetermined quota for the proportion of objects to assign to each category[9].The kappa coefficient used in our project deal with the ordinal categories through "weighting". The mismatched classification results were penalized by assigning partial credit. The calculation process for the quadratic weighted kappa is as below: 1) Matrix O including the number of images received a rating i by A and j by B. And N by N matrix of weights is calculated based on the difference between rater's scores; 2) Matrix of Expected ratings E is calculated as the outer product between each rater's histogram vector of ratings and normalized [10]. The suggested cohen's kapa criteria is shown as Table 3. A good kappa score is beyond 0.6 and if the kappa is above 0.9, that is almost perfect.

| Value of Kappa | Level of Agreement | % of Data that are Reliable |
|---|---|---|
| 0-.20 | None | 0-4% |
| .21-.39 | Minimal | 4-15% |
| .40-.59 | Weak | 15-35% |
| .60-.79 | Moderate | 35-63% |
| .80-.90 | Strong | 64-81% |
| Above .90 | Almost Perfect | 82-100% |

*Table 3*: Suggested Cohen's Kappa Criteria by McHugh & Mary (2012)

**ResNet**

Residual Neural Network is the deep networks that is first introduced in 2015 by He et.al (2015) and won the ILSVRC 2015 image classification task. Resnet was pretrained on ImageNet. It aims to solve the degradation problem that with the network depth increasing, accuracy gets saturated and then degrades rapidly. Resnet features the residual block design, and include three types of skip/shortcut connections to address the problem. In this project, we adopted Resnet101, with the total of 101 layers.

We trained Resnet101 model with 2019 dataset without using the transformation method. Figure 7 showed the train and valid loss and the minimal loss was achieved with the epoch 25 and 0.300 on the valid dataset. The highest valid kappa score as 0.614, and the test kappa score as 0.70 on the public set and test kappa score as 0.70 in the public dataset, as Figure 8 indicated.

Using the transformation method on the Resnet 101, the overall performance was slightly worse. The minimal loss was achieved when the epoch was 25 and the valid loss was 0.300, as shown in Figure 9. The highest kappa score achieved was 0.606, and the test kappa score as 0.730 on the public set and test kappa score as 0.881 in the public dataset (See Figure 10). The potential reason for the slightly worse result might be ineffective transformation methods or lack of hyperparameter tuning.

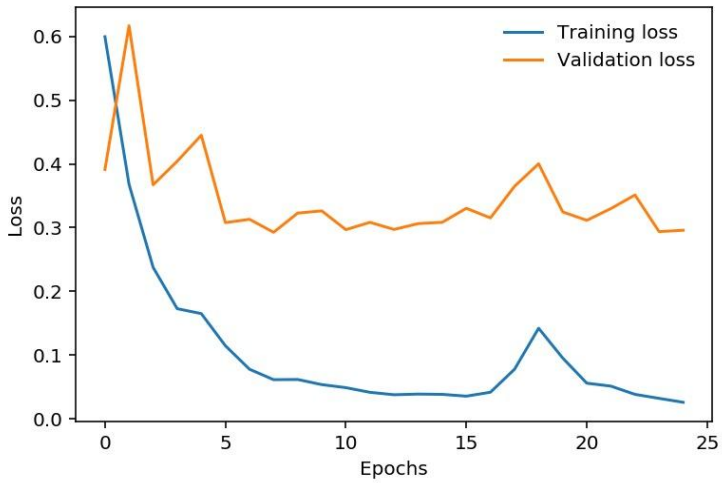



*Figure 7:* Train and Validation Loss for Resnet 101.

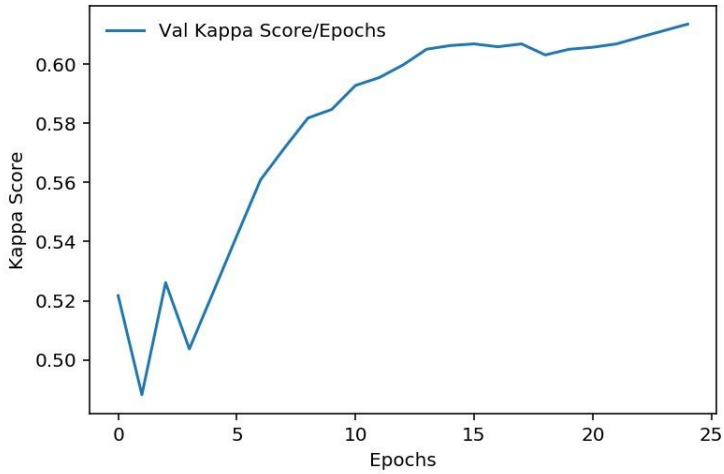



*Figure 8*: Validation kappa scores versus the number of epochs.

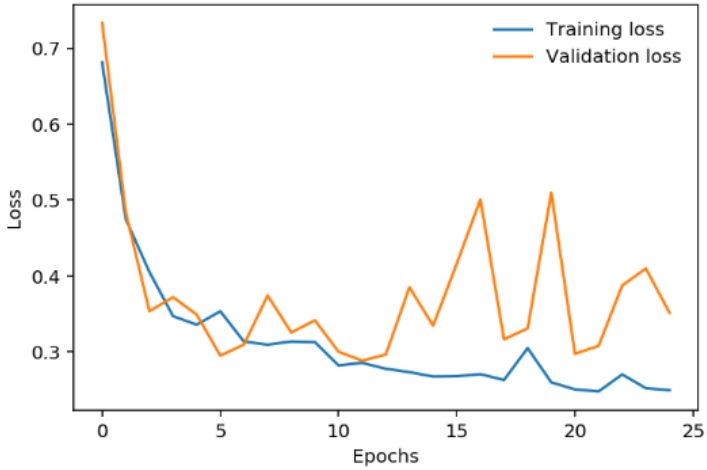



*Figure 9*: Train and Validation Loss for Resnet 101 with transformation methods.

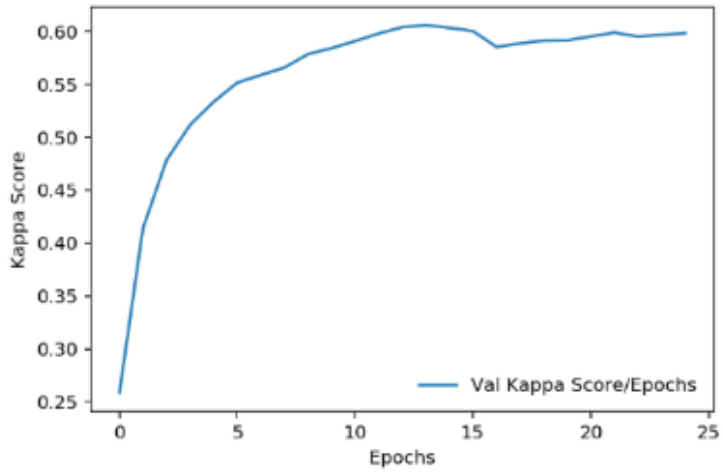
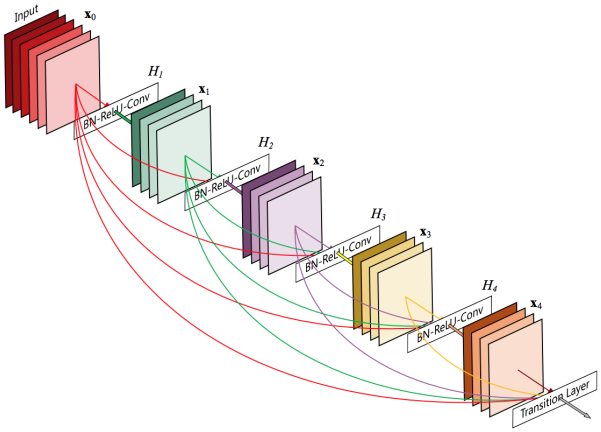



*Figure 10*: Validation kappa scores versus the number of epochs-Resnet101 with transformation methods.

**DenseNet**

Published in 2016, Dense Convolutional Network (DenseNet) paper won CVPR 2017 best paper award and has been cited over 6,000 times so far [11]. Different from ResNet, DenseNet uses distinct architecture by connecting each layer to every other layer in a feed-forward fashion. So each layer receives input from all previous layers, and its output can be used in all subsequent layers (See Figure 11). DenseNet has several advantages including efficient feature reuse and reduction of number of parameters (See Figure 12). It also alleviates vanishing-gradient problem.

We trained DenseNet using our 2019 dataset without further transformation. The result showed the model with 30 epochs and the performance has been presented in Figure 13 by plotting training loss, validation loss and QWK. Our model reaches minimal validation loss at epoch 18, with value 0.282, and reaches highest QWK at epoch 30 with value 0.642. We also submitted our model to Kaggle, and we got public score 0.684 and private score 0.870.



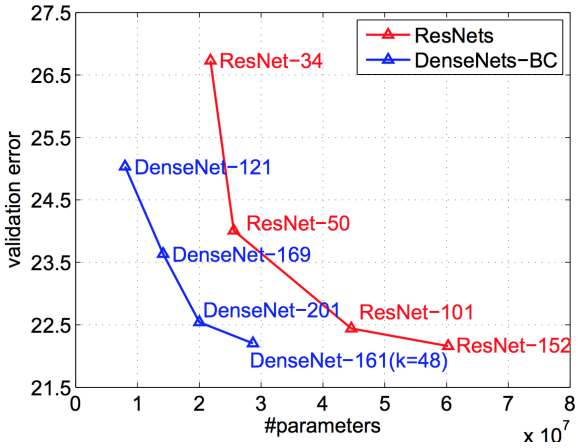




*Figure 11*: DenseNet architecture (Left)
*Figure 12:* Comparison of validation error between DenseNet and ResNet (Right).

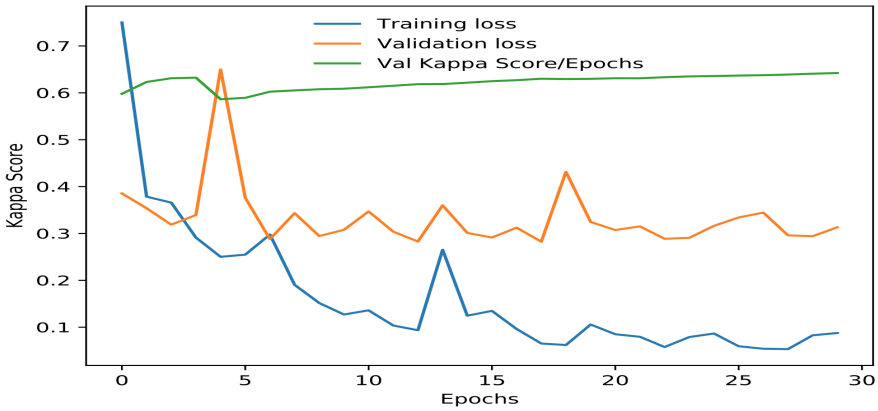



*Figure 13: DenseNet model training loss, validation loss, and QWK score*

**EfficientNet-B4**

We continued our experiment with using the efficientNet as suggested by previous winners for the competition. EfficientNet-B4 is an image classification model and achieve state-of-the-art accuracy as published by Tan & Le (2019) [12]. The model was built based on AutoML and Compound Scaling. The Network is 8.x smaller, 6.1x faster and better score in ImageNet. The result is shown on Figure below. It introduces mobile inverted bottleneck and connects a much fewer parameters squeeze-and-excitation(SE) optimization. In our project, we adopted efficient B4 network.

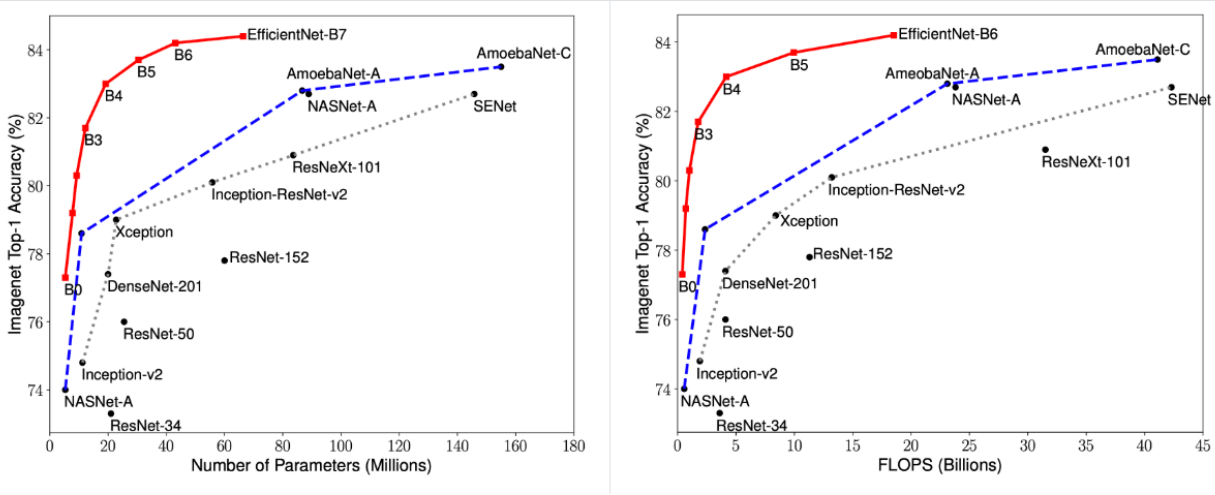



*Figure 14*: Model size vs ImageNet accuracy comparison (Left).
*Figure 15:* Scaling network width and accuracy comparison (Right).

We used two different datasets to train EfficientNet-B4. 1) The 2019 dataset with the train and validation splitting as 8:2. 2) The combined dataset with the train and validation splitting as 8:2. The result is shown below (See Figure 16). The minimal training loss, minimal validation loss and the highest kappa score for validation set is 0.0858, 0.258 and 0.888 accordingly. We achieved the private test QWK score and public QWK score 0.902 and 0.756 accordingly using our best model.

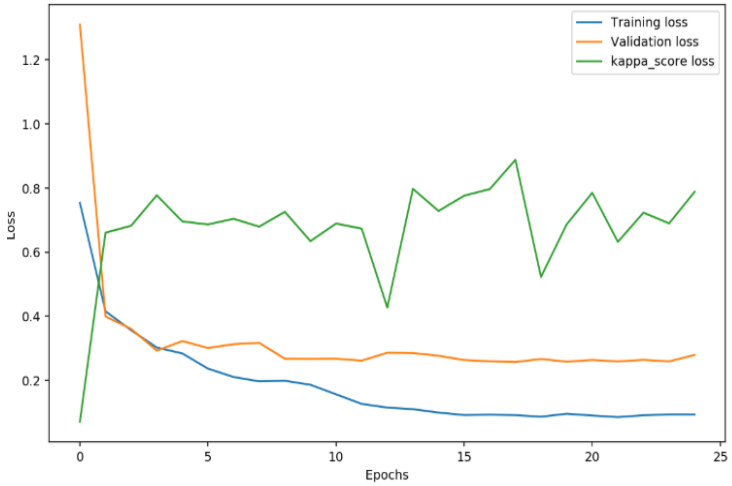



*Figure 16: EfficientNet-B4 model training loss, validation loss, and QWK score using 2019 dataset.*

Using the combined dataset, the results achieved are shown below (See Figure 17). The minimal training loss, minimal validation loss and the highest valid kappa score is 0.199, 0.311 and 0.823 accordingly. We achieved the private test QWK score and public QWK score 0.828 and 0.710 accordingly using our best model.

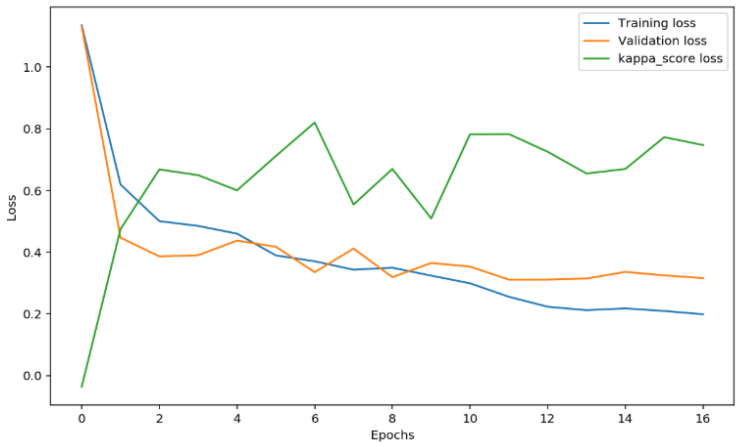



*Figure 17: EfficientNet-B4 model training loss, validation loss, and QWK score using combined dataset.*

The second model performed worse than the first model. The unexpected result might result from different distribution of 2015 dataset and 2019 dataset, so the combined result did not perform well.

## Conclusion

In conclusion, EfficientNet performs the best as regard to the QWK score compared with the three models using testing with using the data transformation and combined dataset (See Table 4). And the top winner achieved 0.936 QWK, which is slightly higher than our best model. However, we still did not find the solid reason with the appropriate transformation methods and the reason that drives to the worse result using the combined dataset. This leads us to the further analysis to optimize our result.

| Model | QWK (Valid) | QWK (Public) | QWK (Private) |
|---|---|---|---|
| **ResNet101(Vanilla)** | 0.614 | 0.709 | 0.890 |
| **DenseNet121** | 0.642 | 0.684 | 0.869 |
| **EfficientNet-B4** | 0.888 | 0.759 | 0.902 |

*Table 4*: Model comparison.

## Future Work

One important step further to optimize our result is to conduct systematic hyperparameter training. As the computation cost for east EfficientNet epoch is high, it takes much time to train and tune the model, which limits the performance result for this project. We plan to tune hypermeters such as learning rate or adjust the regularization methods in the future to tune the model. Also, we might test different EfficientNet and experiment the results. In addition, the current data transformation did not perform well, so this drives us to conduct other potential data augmentation methods shared by other top winners, such as Fliplr, Zoom etc, to find out more efficient data augmentation methods.

## Reference


[1] https://www.kaggle.com/tanlikesmath/intro-aptos-diabetic-retinopathy-eda-starter
[2] Kempen, John H., et al. "The prevalence of diabetic retinopathy among adults in the United States." Archives of ophthalmology (Chicago, Ill.: 1960) 122.4 (2004): 552-563.
[3] https://www.kaggle.com/c/aptos2019-blindness-detection/overview
[4] https://www.kaggle.com/c/aptos2019-blindness-detection/data
[5] https://www.kaggle.com/tanlikesmath/diabetic-retinopathy-resized
[6] Shorten, Connor, and Taghi M. Khoshgoftaar. "A survey on image data augmentation for deep learning." Journal of Big Data 6, no. 1 (2019): 60.
[7] https://www.kaggle.com/c/aptos2019-blindness-detection/discussion/108307
[8] Cohen, Jacob. "Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit." Psychological bulletin 70, no. 4 (1968): 213.
[9] https://stats.stackexchange.com/questions/248583/quadratic-weighted-kappa
[10] https://www.kaggle.com/c/aptos2019-blindness-detection/overview/evaluation
[11] Huang, Gao, et al. "Densely connected convolutional networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
[12] Tan, Mingxing, and Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks." arXiv preprint arXiv:1905.11946 (2019).