Diabetic Retinopathy Detection: Deep Learning Applications

Alisia Sara Baielli
Irene Maria Colombo
Andreea Doloc
Arianna Feliziani
Emanuele Marino Librandi

June 24, 2024

Abstract

Diabetic Retinopathy (DR) is a severe complication of diabetes that affects the retina and can lead to vision loss or blindness if not detected early. Characterized by retinal lesions, DR is irreversible, and treatments focus on preserving remaining vision rather than restoring lost sight. Traditional diagnostic methods rely on manual examination of retina fundus images, a process that is time-consuming, costly, and prone to errors. In contrast, computer-aided diagnosis systems, particularly those utilizing deep learning techniques, have demonstrated greater accuracy and efficiency. Among these, Convolutional Neural Networks (CNNs) are widely used in medical image analysis due to their effectiveness. This project aims to optimize CNNs to enhance the accuracy and efficiency of DR detection in retina images.

Contents

1 Introduction

1.1 Diabetes and Diabetic Retinopathy

Diabetes is a chronic medical condition characterized by high levels of glucose in the blood. Over time, this can lead to serious complications affecting various organs in the body, including the eyes. One of the most severe ocular complications of diabetes is Diabetic Retinopathy (DR). DR is a condition that affects the retina, the light-sensitive tissue at the back of the eye, and can lead to vision impairment and blindness if not detected and treated promptly. As the number of individuals diagnosed with diabetes continues to rise, the incidence of DR is expected to increase correspondingly. It is projected that by 2030, the number of people with DR will reach 191 million, with 56 million suffering from vision-threatening forms of the disease. These alarming statistics put emphasis on the need for effective screening and early detection methods.

1.2 Current Methods of Detection

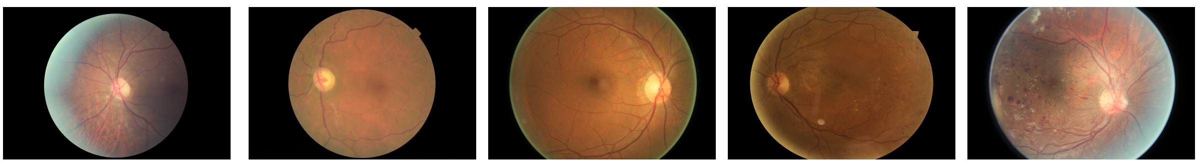
Traditionally, Diabetic Retinopathy is diagnosed through manual examination of retina fundus images by trained ophthalmologists. This process involves inspecting the images for characteristic lesions such as microaneurysms, hemorrhages, and exudates. While effective, manual examination has significant drawbacks. It is a time-consuming and costly process, requiring specialized expertise and equipment. Moreover, the accuracy of manual diagnosis can be compromised by human error and subjectivity, leading to inconsistent results.

1.3 Convolutional Neural Networks as an Alternative

In recent years, advancements in computer-aided diagnosis systems have shown promises in addressing the limitations of traditional methods. Convolutional Neural Networks (CNNs), a class of Deep Learning algorithms, have proven particularly effective in medical image analysis. CNNs are capable of automatically learn and extract features from images, making them an extremely efficient tool for detecting patterns associated with Diabetic Retinopathy. Our project thus aims to explore how CNNs can be optimized to improve the accuracy and efficiency of Diabetic Retinopathy detection in retina images.

2 Dataset description

The dataset used for this study was sourced from the Kaggle Diabetic Retinopathy Detection Competition.[1] It's comprised of 88,702 high-quality retina images with varying resolutions, ranging from 433 $\times$ 289 pixels to 5184 $\times$ 3456 pixels and it's divided into train and test sets. The images are classified into five categories based on the severity of diabetic retinopathy (these labels are assigned by clinicians): (0) - No DR, (1) - Mild, (2) - Moderate, (3) - Severe, and (4) - Proliferative DR.



(a) Healthy retina (b) Mild DR (c) Moderate DR (d) Severe DR (e) Prolif. DR

Figure 1: Examples of various stages of the condition, ranging from absence to proliferative

[1] https://www.kaggle.com/c/diabetic-retinopathy-detection/data

3 Methodology

3.1 Data Preprocessing

Initially, we imported the training and test labels from CSV files into separate pandas dataframes. For each entry in these dataframes, we constructed the full file path to the corresponding image. After loading the data, we combined the training and test sets into a single dataframe to simplify subsequent processing steps. This combined dataframe contained all 88,702 images along with their respective labels and paths.
To standardize the images and improve model performance, we preprocessed them using the OpenCV package. The steps included:

- Rescaling the images to have the same radius (300 pixels).
- Subtracting the local average color to map the images to 50% gray.
- Clipping the images to 90% of their size to remove boundary effects, particularly the black background around the eye.

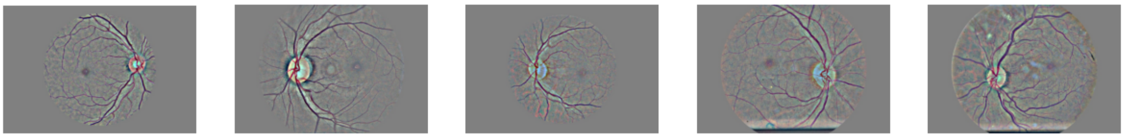


Figure 2: Sample of preprocessed images

Additionally, we experimented with padding (enlarging the images up to the size of the largest image in the dataset). However, we decided not to retain this approach due to the high computational cost associated with processing images of such large dimensions.
For the same reason, we chose to resize the images to 224 $\times$ 224 for our models and perform normalization of pixel values.

However, this dataset, composed of high-resolution images, was still very computationally expensive to process. Given this constraint, we decided to reduce its size by creating a dataset of 10,000 images. To do so we randomly sampled images from each class ensuring that the sampling reflected the original class distribution, as shown in Figure 4 in Appendix A. Throughout our analysis we will reference to this dataset with the name "Unbalanced Dataset".

Upon running our models on this dataset, we observed suboptimal performance due to class imbalance. Therefore, we created a new dataset of around 12,000 images, balanced with respect to having or not having diabetic retinopathy (binary labeling). This dataset aimed for an equal number of images with and without the disease.
After assessing the results and observing improvements (especially for the binary classification), we created a final dataset of 3,540 images, evenly distributed across five classes. To enhance classification performance across all severity levels, we ensured each class had an equal number of samples, matching the count of the least represented class.
These will be referenced respectively as "Balanced for Binary" and "Balanced for Quinary" datasets.

3.2 EDA

We have performed Exploratory Data Analysis on the unbalanced dataset. We noticed that there is a high correlation between the right and the left eye of the patient's DR level, as can be see in Figure 8a in Appendix A. For the purposes of our analysis, however, we can use the

sample data as independent images, since describing the correlation between the eyes' levels of advancement of the illness is out the scope of the research. In fact, our purpose is to correctly determine the presence and severity of DR, rather than relying on the numerical correlation between measurements from the two eyes.
As a further analysis, we studied the presence of right and left images in our subset and the distribution of images per patient. In particular, we notice that we mostly have one image for each patient, and that the percentage of left and right eye's images is balanced. This information would be highly relevant if we were not considering each image independently, since having only one image per patient ensures that the model does not inadvertently learn patient-specific features, thereby improving the generalizability and robustness of the model.
However, as stated before, we are not considering the patient as a metric of our analysis, so we can ignore the instances of multiple images for the same patient and we can consider them to be independent.

4 Models

We selected the pre-trained models VGG, AlexNet, GoogleNet and ResNet, because they offer proven high performance and efficient feature extraction, significantly reducing training time and resource requirements. We implemented transfer learning with these models based on evidence from the literature indicating their effectiveness for diabetic retinopathy (DR) detection on the Kaggle EyePACS dataset. Transfer learning allows us to leverage the knowledge gained from large-scale image classification tasks, thereby improving our model's overall robustness for DR detection.

VGG16 is renowned for its simplicity and depth, consisting of 16 layers with learnable parameters. The architecture employs small 3×3 convolutional filters, enabling the network to capture features in images. For our purposes, we leveraged a pre-trained VGG16 model (trained on ImageNet) and fine-tuned it for DR detection and classification, using transfer learning techniques to benefit from extensive prior training on general image features.

AlexNet is a pioneering deep learning convolutional neural network architecture that achieved significant breakthroughs in image classification tasks. AlexNet consists of five convolutional layers, followed by three fully connected layers, with ReLU activations applied to introduce non-linearity. It employs dropout for regularization and max-pooling layers to reduce spatial dimensions, enhancing computational efficiency and mitigating overfitting.

GoogleLeNet, also known as *Inception v1*, features 22 layers with learnable parameters and is characterized by its innovative Inception modules. Each Inception module applies multiple convolutional filters of different sizes (1×1, 3×3, and 5×5) and a parallel max-pooling operation, concatenating their outputs to capture features at various scales efficiently. GoogleLeNet's design includes nine Inception modules, significantly reducing computational cost while maintaining high accuracy.

ResNet, short for Residual Networks, introduces the concept of identity shortcut connections, which skip one or more layers. This allows the model to learn residual functions with reference to the layer inputs, marking a significant improvement in the field of computer vision by addressing the problem of vanishing gradients when training very deep neural networks. ResNet50 is a 50-layer deep convolutional neural network, comprising an initial convolutional layer followed by multiple residual blocks, each containing convolutional layers and identity connections. These blocks are grouped into stages, each reducing the spatial dimensions while increasing the depth of feature maps. The network ends with a global average pooling layer and a fully connected (dense) layer with a softmax activation for classification.

Ensembling is a technique used to improve the accuracy and robustness of models by combining predictions from multiple of them. In this case, we decided to use three different architectures: VGG16, AlexNet, and GoogleLeNet. These were the best-performing models. The individual predictions from these models are averaged to create a final ensemble prediction. This approach leverages the strengths of each model, reducing the likelihood of errors and increasing overall performance.

KAN (Kolmogorov-Arnold Networks) architecture was also used to build a simple network. A detailed discussion of this approach, which was not effective in our application, can be found in Appendix B.

4.1 Binary classification

Here, our goal was to build a binary classifier to distinguish between healthy retina images and those affected by diabetic retinopathy.

VGG16
The final fully connected layer of the VGG16 model was replaced with a custom classifier head to output a single probability score, followed by a Sigmoid activation function to determine the class label. The model was trained using Binary Cross Entropy Loss, with optimization handled by Stochastic Gradient Descent. A learning rate scheduler and early stopping were implemented to enhance training efficiency and prevent overfitting.
The best performance was found training the model on the balanced dataset. This approach achieved a training accuracy of 70% and a test accuracy of 67%. The close proximity of these accuracies suggests that the model generalizes adequately from the training data to the test data.

AlexNet
Similarly to the previous model, the final layer of the AlexNet model was customized to output a single probability score, followed by a Sigmoid activation function for binary classification of healthy and DR-affected images. The feature extraction layers of AlexNet were frozen, and the classifier was modified to include additional fully connected layers, ReLU activation, and dropout for regularization. The model was trained using Binary Cross Entropy Loss and optimized with Stochastic Gradient Descent with momentum and weight decay. A learning rate scheduler and early stopping were employed to enhance training efficiency and prevent overfitting. Early stopping was triggered to prevent overfitting based on validation loss.
A training accuracy of 66.3% and a validation accuracy 65.9% were reached in the balanced dataset for binary classification.

GoogleLeNet
The GoogleLeNet model's final fully connected layer was replaced with a custom classifier head for binary output. The model was trained using Binary Cross Entropy with Logits Loss and optimized with Stochastic Gradient Descent. To enhance training efficiency and prevent overfitting, a learning rate scheduler and early stopping were implemented.
The model demonstrated effective learning without significant signs of overfitting in the balanced dataset with a training accuracy of 67.2% and a validation accuracy of 67%.

ResNet
To perform binary classification, we adapted the pre-trained ResNet50 model, by retaining its convolutional base of the pre-trained, and replacing the final fully connected layer with a classifier head comprising a batch normalization layer, a ReLU activation layer, a global average pooling

layer, a dropout layer for regularization, and a dense layer with a softmax activation function to output class probabilities for the two classes. Early Stopping was used to prevent overfitting; Reduce Learning Rate on Plateau was applied to reduce the learning rate by a factor of 0.1 if the validation loss did not improve for 5 consecutive epochs, with a minimum learning rate of 1×10^{-5}. Finally, the CSV Logger logged the training process for later analysis.
On the dataset comprised of 12,000 images we got a training accuracy of 50% and the validation accuracy reached 54%. When evaluated on the test dataset, the model yielded a test accuracy of 51.14%. These results suggest that its performance on both the validation and test sets indicates limited predictive power, with accuracy hovering around chance levels, as well as a slight overfitting.

4.2 Multi-class Classification

The multi-class classifier purpose is to classify images into one of five DR severity levels.

VGG16
Similar to the binary classifier, the VGG16's fully connected layers were modified to output probabilities for five classes, followed by a Softmax activation function. The model was trained using Cross Entropy Loss and optimized with Stochastic Gradient Descent. The training regimen included a learning rate scheduler and early stopping to optimize performance. The classification performance improved with the balanced dataset; however, neither the balanced nor the unbalanced datasets yielded satisfactory results. This outcome could be attributed to the complexity of multi-class classification, the model's suitability (VGG16 may be more effective for binary classification or tasks with fewer classes), and an insufficient number of images.

AlexNet
For multi-class classification, we modified the final fully connected layer of AlexNet to output probabilities for five classes. The feature extraction layers of AlexNet were frozen, and the classifier was adapted to include additional fully connected layers, ReLU activation, and dropout for regularization. The model was trained using Cross Entropy Loss and optimized with Adam. A learning rate scheduler and early stopping were employed to enhance training efficiency and prevent overfitting.
Despite these measures, satisfactory results were not achieved on the unbalanced dataset, while the dataset balanced for quinary achieved an accuracy of almost 40%. Moreover, we noticed a high risk of overfitting in this last model.

GoogleLeNet
To make the desired classification on five classes we adapted GoogleLeNet Model. The model's final fully connected layer was replaced with a custom classifier head, comprising two linear layers separated by ReLU activation and dropout for regularization, followed by a softmax activation to output class probabilities. The training process involved iterating over the dataset for 12 epochs, adjusting model parameters based on backpropagation. We monitored training and validation loss and accuracy, implementing early stopping to prevent overfitting. However, despite these efforts, the model's performance did not yield satisfying results during training and testing in both balanced and unbalanced dataset.

ResNet
The second model adapts the process for a quinary classification task by reverting to the original 'level' column. The model was trained for 20 epochs on the dataset containing 12,000 images. By the end of epoch 20, the training loss was 1.0309, and the training accuracy was 59.85%. The validation loss concluded at 1.7139, with a validation accuracy of 50.29%. The evaluation results indicated a test loss of 1.6321 and a test accuracy of 53.57%. Although marginally better

than the results achieved on the binary classification, the model's performance on both the validation and test is still only slightly better than random chance. On the balanced datset, ResNet reaches its lowest validation accuracy, 24.76%, and misclassifies most of the images as having proliferative DR.

5 Discussion

Following is the final table with all the results we obtained using the different models applied to the unbalanced and balanced datasets. In this section we'll comment the achieved results.

Dataset	Model	Binary	Multiclass
Unbalanced Dataset	VGG16	74.8%	73.3%
	AlexNet	75.1%	73.7%
	GoogleLeNet	74.8%	73.2%
	Random	50.1%	20.4%
Balanced for Binary	VGG16	67.5%	49.8%
	AlexNet	65.9%	53.6%
	GoogleLeNet	67%	49.8%
	ResNet	51.1%	53.6%
	Random	48.9%	20.1%
	Ensemble	67.9%	-
Balanced for Quinary	VGG16	-	34.2%
	AlexNet	-	38.7%
	GoogleLeNet	-	27.9%
	ResNet	-	22.5%
	Random	-	19.4%
	Ensemble	-	40.2%

Table 1: Accuracy Results

The first point worth addressing in our discussion is the structure of the dataset, which posed a significant challenge throughout the entire process of devising, training and testing our models. The main advantage of our dataset, which also constitutes its disadvantage, is its accurate reflection of real-life conditions, where instances of healthy retinas are far more prevalent than those affected by DR. For context, the distribution of the dataset is as follows: 74% healthy retinas, 15% with mild symptoms, 7% with moderate symptoms, and 2% each with severe symptoms and proliferative diabetic retinopathy. As already mentioned, when creating subsets to work with, we first sampled in a way that was reflective of this very skewed distribution, as we wanted a robust model that could be applicable to real-life situations. Upon further investigation, we also saw fit to create additional datasets with different balancing strategies. One of them was balancing between healthy and affected retinas, while the other provided an equal representation of healthy retinas and each level of DR. The following results highlight different problematic aspects of our sampling choices.

For the unbalanced dataset, while AlexNet (75.1%), VGG16 and GoogleLeNet (74.8%) report high accuracies for the binary classification, the confusion matrices reveal significant misclassification issues. This discrepancy between accuracy and true instances highlights the fundamental issue of using an unbalanced dataset.
In the multiclass classification task, AlexNet leads with test accuracy 73.7%, yet the confusion matrices reveal a similar trend of pervasive misclassification.

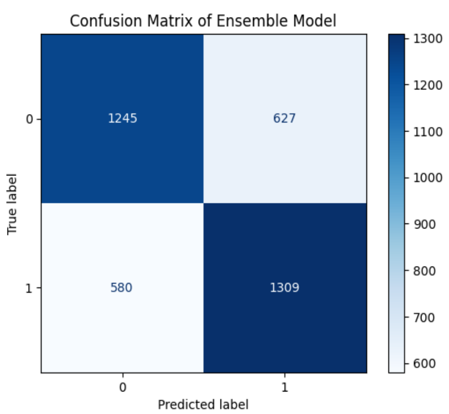
This demonstrates the importance of considering not only accuracy as a metric, but also the detailed distribution of true positives, false positives, and false negatives when evaluating a model. Moreover, the high accuracy is quite likely due to the large proportion of zeros being correctly classified, thus giving a misleading impression of good model performance.

We obtain improvements when training our models on the balanced dataset for binary classification. Although the test accuracies decrease, with VGG16 at 67.5%, AlexNet at 65.9% and GoogleLeNet at 67% , the confusion matrices show much better classification results. However, as expected, the quinary classification on this dataset didn't yield good results, due to the disproportionately low number of samples in each class other than 0.

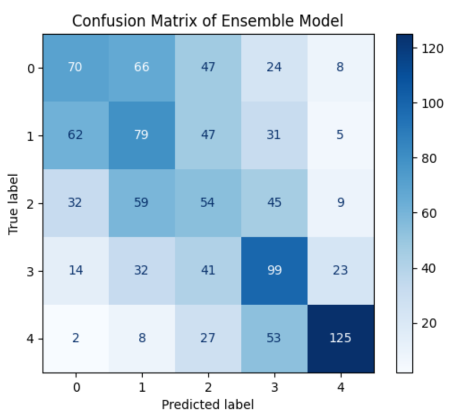
In the last dataset, balanced on the five classes, the test accuracies are significantly lower. VGG16's accuracy drops to 34.2%, and AlexNet, though better at 38.7%, still indicates poor performance, as supported by the confusion matrices. GoogleLeNet and ResNet perform even worse, with accuracies of 27.9% and 22.5%, respectively. However, the confusion matrices suggest some improvement in classification for these models, as they no longer classify all test images as zeros. The classification pattern shifts, with the best classified image now being 4. This can be seen especially from the confusion matrix of ResNet, where almost all instances are classified as 4. The models perform worse on the balanced dataset likely because they were previously overfitting to the majority class (healthy retinas) in the unbalanced dataset. Balancing the dataset removes this bias, forcing the models to learn more features across all classes. Additionally, the balanced dataset does not contain a high number of images, which is crucial for effective training.

Finally, by employing the technique of *ensembling* on the balanced dataset for binary classification, we achieved higher accuracy compared to our best individual model, VGG16, which had an accuracy of 67.5%. The ensemble method not only improved accuracy with a final result of 67.9% but it also produced a more satisfying confusion matrix, indicating better overall model performance.

For quinary classification on the balanced dataset, the ensemble approach also demonstrated slight improvements. The best individual model accuracy was 38.7%, achieved by AlexNet. The ensemble method surpassed all individual models, achieving an accuracy of 40.2%, indicating improved effectiveness in handling multi-class classification tasks as well.



(a) Confusion matrix of Ensemble Model - binary classification



(b) Confusion matrix of Ensemble Model - quinary classification

Figure 3: Confusion matrices for binary and quinary classification ensemble models

Notice, from the confusion matrices plotted above, that the following pattern emerges: although the model does not perfectly distinguish between the 5 classes, misclassifications often occur within neighboring categories. For instance, images belonging to class 1 are frequently predicted as class 0 or 2.
This is interesting because even if the image is not classified correctly, it is often assigned to a class that is close in severity to the true class. This implies that the classifier captures some of the ordinal nature of the problem, where the severity of diabetic retinopathy progresses gradually.

We finally compared all of these models to a random classifier. The comparative analysis shows that the pre-trained models for binary and multi-class classification, despite their flaws, perform significantly better than this random model. The trained models demonstrate better learning, higher accuracy, and less randomness.[2]

6 Conclusion and limitations

Throughout the analysis of our results, the primary challenges we encountered were the class imbalance of the dataset and the significant computational power required to run the models effectively. [3]

To manage the computational constraints, we had to considerably reduce the image resolution, which adversely impacted the accuracy of our models. Although we addressed the class imbalance by creating subsets of our data, a more robust solution would involve training on the 88,000 images or data augmentation - techniques we were unable to implement due to limited computational resources.

Despite these limitations, we are satisfied with our results and confident that with additional time and computing capabilities, we could significantly improve our outcomes.

Future efforts could involve: leveraging the full dataset, employing data augmentation and addressing class imbalance, optimizing image resolution and color balance to enhance model performance, and exploring hybrid models that combine different CNN architectures for better feature extraction and classification accuracy. We also envision the potential development of an application capable of detecting diabetic retinopathy from retinal scans, which could prioritize patient visits based on DR classification scores.
These enhancements, driven by advanced deep learning techniques, have the potential to significantly improve early detection and diagnosis of diabetic retinopathy, ultimately leading to better patient outcomes.

[2] You can refer to Appendix B for the plots produced
[3] For technical difficulties in the implementation of ResNet with Keras please refer to Appendix C

Appendix

A Images

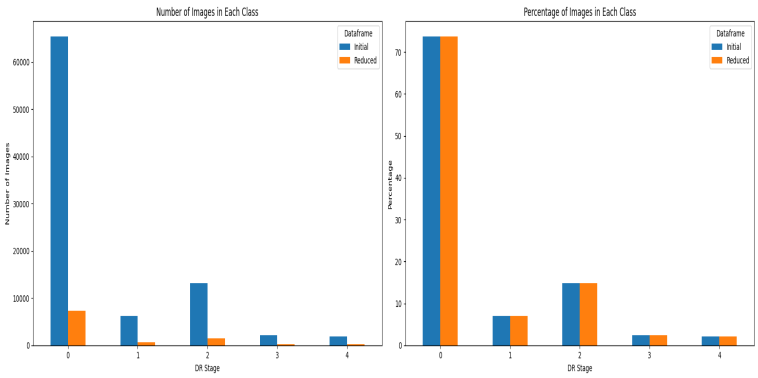


Figure 4: Comparison of Number versus Percentage for the unbalanced dataset

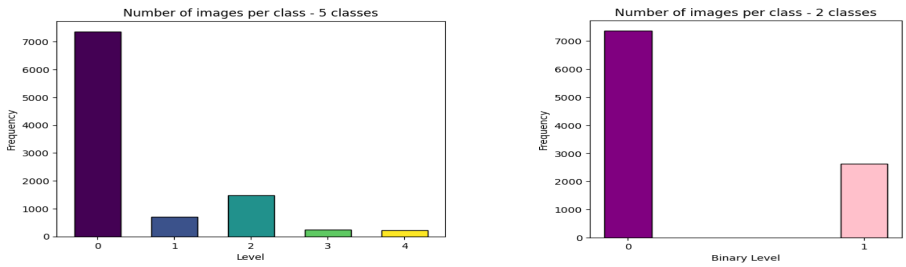


Figure 5: Proportions of images' classes in the unbalanced dataset

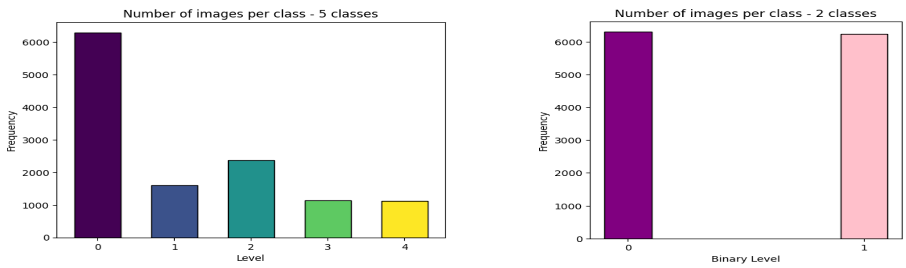


Figure 6: Proportions of images' classes in the binary balanced dataset

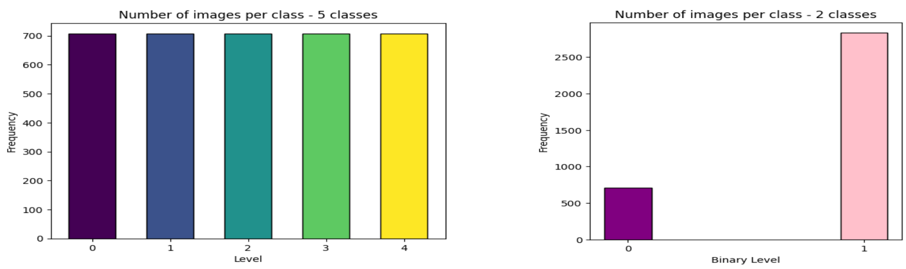
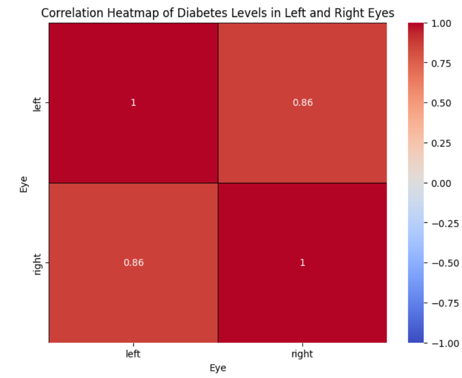
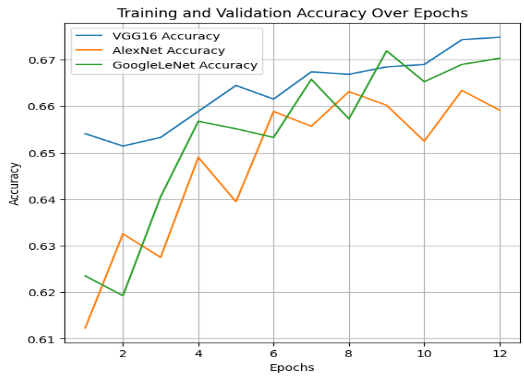


Figure 7: Proportions of images' classes in the 5 classes balanced dataset

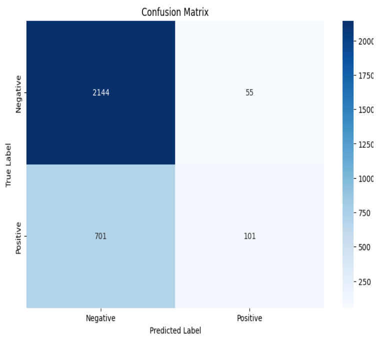


(a) Confusion matrix: Correlation of DR in left and right eyes

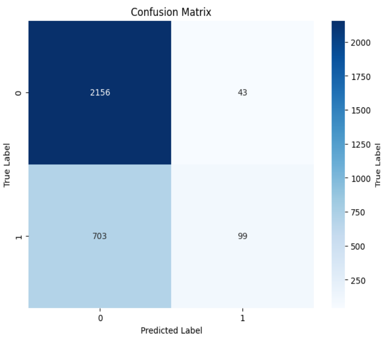


(b) Validation accuracy over epochs - Binary

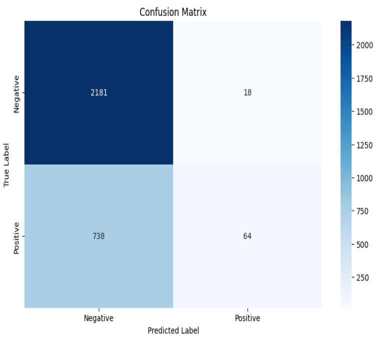
Figure 8



(a) VGG16

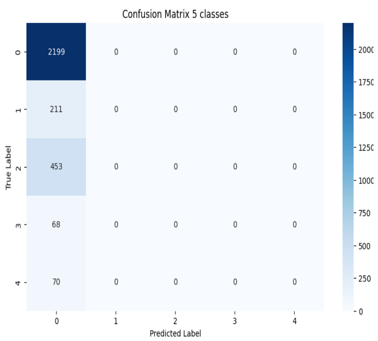


(b) AlexNet

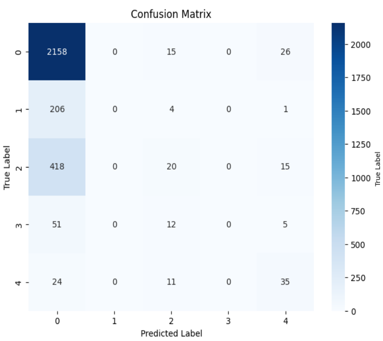


(c) GoogleLeNet

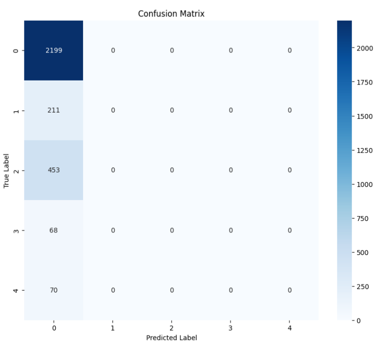
Figure 9: Confusion Matrices of unbalanced dataset - Binary



(a) VGG16

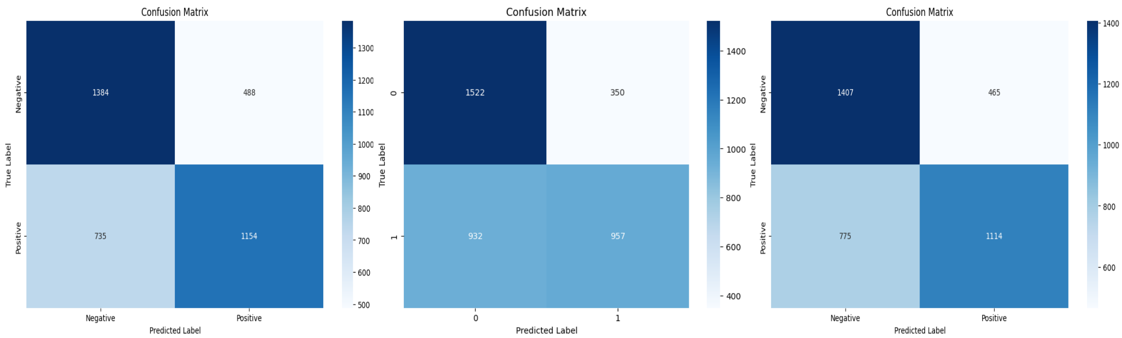


(b) AlexNet



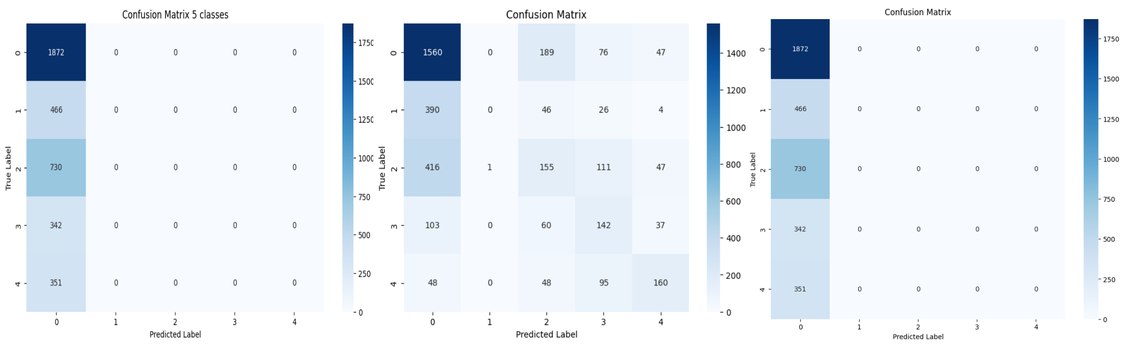
(c) GoogleLeNet

Figure 10: Confusion Matrices of unbalanced dataset - Quinary



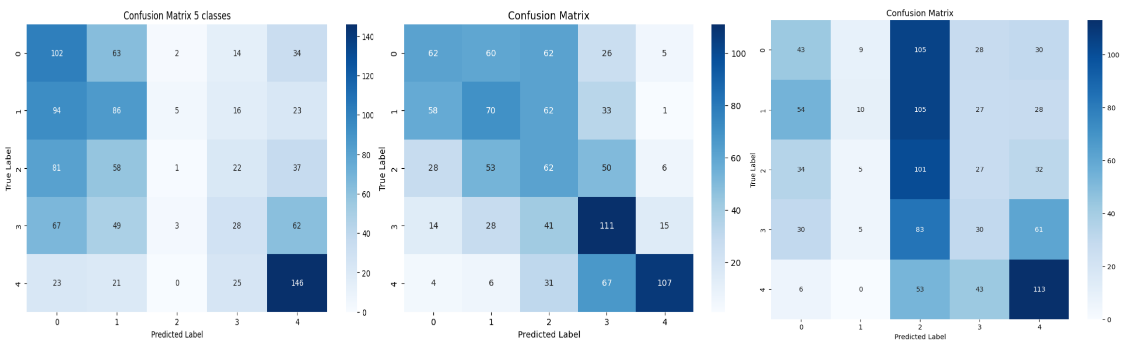
(a) VGG16 (b) AlexNet (c) GoogleLeNet

Figure 11: Confusion Matrices of first balanced dataset - Binary



(a) VGG16 (b) AlexNet (c) GoogleLeNet

Figure 12: Confusion Matrices of first balanced dataset - Quinary



(a) VGG16 (b) AlexNet (c) GoogleLeNet

Figure 13: Confusion Matrices of second balanced dataset - Quinary

B Kolmogorov-Arnolds Network (KAN)

Very recently, a promising alternative to Multi-Layer Perceptrons (MLPs) has been proposed[4], called **Kolmogorov-Arnold Networks (KANs)** and based on the **Kolmogorov-Arnold representation theorem**. Vladimir Arnold and Andrey Kolmogorov established that if f is a multivariate continuous function on a bounded domain, then f can be written as a finite composition of continuous functions of a single variable and the binary operation of addition. More specifically, for a smooth $f : [0,1]^n \to \mathbb{R}$,

$$f(\mathbf{x}) = f(x_1, x_2, ..., x_n) = \sum_{q=1}^{2n+1} \Phi_q\big(\sum_{p=1}^{n} \phi_{q,p}(x_p)\big)$$

where $\phi_{q,p} : [0,1] \to \mathbb{R}$ and $\Phi_q : \mathbb{R} \to \mathbb{R}$.
Moreover, f can be rewritten in matrix form as

$$f(\mathbf{x}) = \Phi_{out} \circ \Phi_{in} \circ \mathbf{x}$$

where

$$\Phi_{in} = \begin{pmatrix} \phi_{1,1}(\cdot) & ... & \phi_{1,n(\cdot)} \\ ... & & ... \\ \phi_{2n+1,1}(\cdot) & ... & \phi_{2n+1,n}(\cdot) \end{pmatrix}, \quad \Phi_{out} = \big(\Phi_1(\cdot), ..., \Phi_{2n+1}(\cdot)\big)$$

Both Φ_{in} and Φ_{out} are nothing more than special cases of the same matrix:

$$\Phi = \begin{pmatrix} \phi_{1,1}(\cdot) & ... & \phi_{1,n_{in}}(\cdot) \\ ... & & ... \\ \phi_{n_{out},1}(\cdot) & ... & \phi_{n_{out},n_{in}}(\cdot) \end{pmatrix}$$

Therefore, a **KAN layer** with n_{in}-dimensional inputs and n_{out}-dimensional outputs can be defined as a matrix of 1D functions

$$\Phi = \phi_{q,p}, \quad p = 1, 2, ..., n_{in}, q = 1, 2, ..., n_{out}$$

where the functions $\phi_{q,p}$ have trainable parameters, being thus learnable. After having defined a layer, we can proceed by defining a network with L layers, with the l-th layer Φ_l having shape (n_{l+1}, n_l). The whole network is:

$$\text{KAN}(x) = \Phi_{L-1} \circ ... \circ \Phi_1 \circ \Phi_0 \circ x.$$

KAN is different from MLPs in the sense that it doesn't use static activation functions, but univariate functions that act both as weights and as activation functions. This allows KAN to be more flexible and efficient.
We tried to use this new and promising architecture in our project by relying on implementations that extended the idea of Kolmogorov-Arnold Networks to Convolutional Layers, changing the classic linear transformation of the convolution to learnable non-linear activations.
KAN Convolutions (**CKAN**) are very similar to convolutions, but instead of applying the dot product between the kernel and the corresponding pixels in the image, a Learnable Non Linear activation function is applied to each element, that are then added up. The kernel of the CKAN is equivalent to a KAN Linear Layer of 4 inputs and 1 output neuron. For each input i, a learnable function ϕ_i is applied, and the resulting pixel of that convolution step is $\sum_i \phi_i(x_i)$. The implementation of CKAN has a lot of potential, even though it is still in its early stages of development: we worked on one of the few existing implementation[5]. We trained a KAN model with two KAN convolutional layers and one KAN linear layer on the 12,000 dataset for the quinary classification task, obtaining about 50% test accuracy with a 1.37 loss. However,

[4] "*KAN: Kolmogorov-Arnold Networks*": `https://doi.org/10.48550/arXiv.2404.19756`
[5] `https://github.com/AntonioTepsich/Convolutional-KANs/blob/master/README.md`

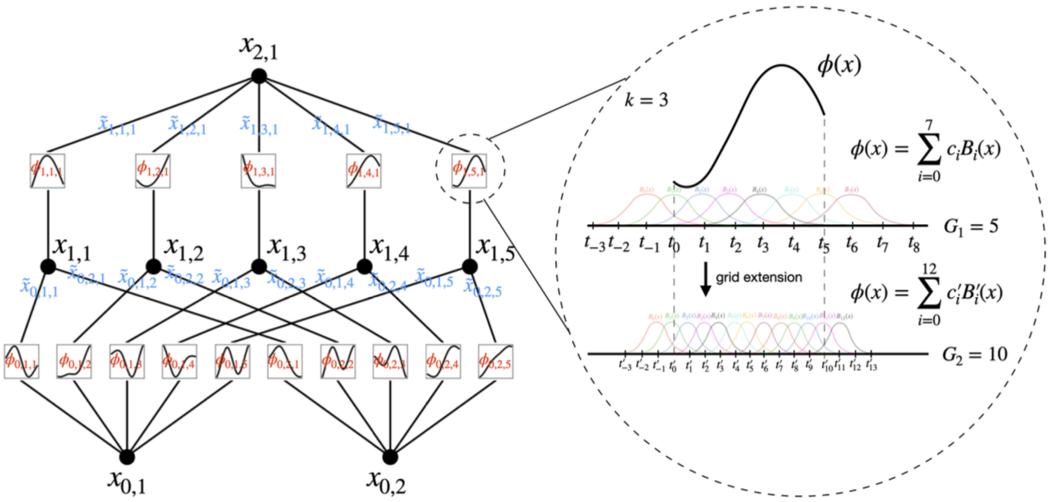


Figure 14: **Left**: Notations of activations that flow through the network. **Right**: an activation function is parameterized as a B-spline, which allows switching between coarse-grained and fine-grained grids.

the model performed poorly as all test images were classified as class 0. We tried increasing the resolution of images and the number of epochs, but we had to interrupt the process, as training was very slow and required a lot of computational power, much beyond our possibilities. Indeed, as pointed out by the authors of the original paper, KAN's efficiency has not been optimized yet:

Currently, the biggest bottleneck of KANs lies in its slow training. KANs are usually 10x slower than MLPs, given the same number of parameters.

To conclude, we weren't able to properly test KAN's performance for our classification task as we couldn't build an extremely complicated network due to the lack of computational power, that was also a constraint on the number of images that we used to train our models.

C Technical difficulties with Keras

We initially attempted to implement all our models using PyTorch but encountered significant challenges in creating and storing the large number of tensors required. To address this, we implemented the ResNet model using Keras, anticipating it would be a less computationally expensive alternative. Unfortunately, this approach proved ineffective; the Keras-based ResNet was exceedingly slow, prone to crashing, and achieved the lowest accuracies in almost all cases (the only instance where it exceeded GoogleLeNet's performance was on the multiclass balanced for binary classification dataset). Consequently, we excluded it from our ensemble modeling, which consisted of VGG, AlexNet, and GoogleLeNet implemented with PyTorch. Nevertheless, we decided to include the Keras-based ResNet in our analysis to illustrate different approaches to implementing CNNs.

References

[1] Wejdan L. Alyoubi, Wafaa M. Shalash, and Maysoon F. Abulkhair. Diabetic retinopathy detection through deep learning techniques: A review. *Informatics in Medicine Unlocked*, 20:100377, 2020.

[2] Wu L. Li H. et al. Dai, L. A deep learning system for detecting diabetic retinopathy across the disease spectrum. *Nat Commun 12*, 2021.

[3] Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y. Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks. *arXiv preprint*, 2024.

[4] Theodoropoulos D. Manikis G. Ktistakis E. Boutsora O. Berto A. Scarpa F. Scarpa A. Fotiadis D. I. Marias K. Tsiknakis, N. Deep learning for diabetic retinopathy detection and classification based on fundus images: A review. *Computers in biology and medicine*, page 104599, 2021.