

Capstone Project Phase B

Diagnosis of Diabetic Retinopathy

25-1-R-11

[Git link](https://github.com/lital1998/Final-Project---DR-Diagnosis.git)

**Supervisor:**

Miri Weiss Cohen

**Submitters:**

Lital Haibetov

Katrin Frimus

**Table of Contents:**

1. Introduction……………………………………………………………………..3
2. Related work……………………………………………………………………3-4
3. Project Review and Process Description……………………………….………5-13

3.1 Architectures research………………………………………………5-7

3.2 Description of the research process…………………………………7-12

3.3 Examples…………………..…………………………………………13

3.4 Discussion and Conclusions………………………………………….13

1. User documentation ……………………………………………………………14-16
2. References……………………………………………………………………….17
3. **Introduction**

Diabetes is a chronic condition that affects the body’s ability to regulate blood glucose levels, either due to insufficient insulin production by the pancreas or the body’s inability to use insulin effectively.

A common and severe complication of diabetes is diabetic retinopathy (DR), which affects the blood vessels in the retina and can lead to vision loss. DR occurs when elevated blood sugar levels damage retinal blood vessels, causing them to swell, leak, or close entirely. These changes can result in irreversible vision loss [1], and due to the absence of early symptoms, DR often progresses to permanent blindness [4]. By 2040, it is estimated that there will be 200 million cases of diabetic retinopathy globally [2].

In this project we predict if a patient has a DR based on retina images. The challenge is to use machine learning techniques – Deep Learning -Convolution Neural Networks to make a prediction with the images as input.

The methodology we used involves employing DenseNet, a powerful neural network architecture well-suited for classification tasks, particularly in the medical field, and testing various hyperparameters to optimize the performance of the trained system.

A collage of images of a human eye

Description automatically generated

**Fig 1:** DR is classified into five stages: normal (a), mild (b), moderate (c), severe (d), and proliferative (e)[

**2. Related work**

InDai et al. article [2], the aim was to develop A deep learning system ,that called DeepDR, for detecting diabetic retinopathy across the disease spectrum proposeddeep learning system that can detect early-to-late stages of diabetic retinopathy.  
The DeepDR system had three sub-networks:

1. image quality assessment sub-network: make binary classification of whether the image was gradable and recognize certain quality issues in terms of artifacts, clarity, and field problems of the retinal images.
2. lesion-aware sub-network: aim to achieve detection and segmentation of microaneurysms, CWS, hard exudates, and hemorrhages.,
3. DR grading sub-network: classify the images into non-DR, mild, moderate, severe, or PDR, and binary classification of whether there was DME.

The DeepDR system achieved the whole-process diagnosis of DR from early to late stages based on the accurate detection of retinal lesions that was especially accurate for microaneurysms.

In Hacisoftaoglu et al. article[3] was described a Deep learning frameworks for diabetic retinopathy detection with smartphone-based retinal imaging systems. articleshowed how CNN-based frameworks like AlexNet, GoogLeNet, and ResNet50 can improve the detection of diabetic retinopathy (DR) in retina images from both smartphone-based and traditional fundus cameras. The study utilized pretrained networks using the transfer learning approach**.** ). The classification layer has two classes since images are separated into two classes: DR and No DR. As a results ResNet50 reached the highest accuracy of 98.6%, the sensitivity of 98.2% and specificity of 99.1%.

In Qureshi et al. article[4] the aim was to develop a new multi-layer architecture of active deep learning algorithms (ADL-CNN) to recognize the severity level of DR through retinography images. The ADL-CNN model is based on the LeNet architecture with 7 layers. The significant advantages of the model include its ADL-based architecture, which efficiently identifies lesions associated with disease severity classification by learning visual features from images to generate masks for predictions and segment regions of interest (ROI). Another advantage is the model’s ability to detect samples, prominent parts, and lesions in images through a single backward-forward pass, learning from the most relevant image areas. Additionally, the preprocessing stage, which includes contrast enhancement and noise removal, enriches the features necessary for DR severity detection. The output is a classification of diabetic retinopathy severity (normal, mild NPDR, moderate NPDR, severe NPDR, or PDR).  
ADL-CNN achieved the highest results comparing to other approaches, particularly in specificity (SP) 95.10%, sensitivity (SE) 92.20%, and accuracy 98%.

In Zhang et al. article the classification of diabetic retinopathy algorithm based on a novel dual‑path multi‑module model article proposes a dual-path multi-module network (DP2M-Net) algorithm designed to classify retinal images into DR and non-DR categories by distinguishing normal and diseased regions. The model first optimizes local information based on the multiplexing structure of different size kernels, and then obtains the global information of lesions, correctly reflects the locations of retinal lesions, and allows accurate DR classification with fewer medical images. The DP2M-Net algorithm achieves classification accuracies of 98.9%, 99.3%, and 98.3% for 3 different datasets

**3. Project Review and Process Description**

We built a system that classifies retinal fundus images to determine whether a person has Diabetic Retinopathy (DR) or not. We used a DenseNet architecture, which is known for its strong performance in medical image classification tasks. tested the system using different hyperparameters to find out what is the optimized way to build the network. The hyperparameters we experimented with include batch size, number of epochs, dropout rate, and learning rate.

**3.1 Architectures research**

**3.1.1 Dense Convolutional Network (DenseNet)**

When convolutional neural networks (CNNs) become deeper, a major problem appears information or gradients can fade as they pass through many layers. This issue makes it harder to train deep networks. Recent work has shown that convolutional networks can be substantially deeper, more accurate, and efficient to train if they contain shorter connections between layers close to the input and those close to the output.

The network has a simple connectivity pattern: to ensure maximum information flow between layers in the network.it connects all layers (with matching feature-map sizes) directly with each other. To preserve the feed-forward nature, each layer obtains additional inputs from all preceding layers and passes on its own feature-maps to all subsequent layers.

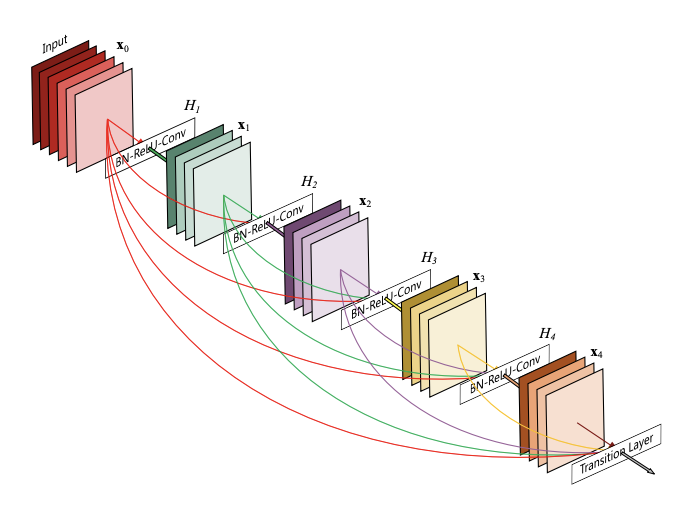
In regular convolutional networks with L layers, there are L connections, bone between each layer and its subsequent layer. In DenseNet design, there are L(L+1)/2 direct connections. the feature-maps of all preceding layers are used as inputs, and its own feature-maps are used as inputs into all subsequent layers.[7]

Figure 11: A 5-layer dense block with a growth rate of k = 4. Each layer takes all preceding feature-maps as input.

DenseNet is an extension of ResNet [8]. ResNet uses skip connection techniques The skip connection connects activations of a layer to further layers by skipping some layers in between [9]. ResNets make this information preservation explicit through additive identity transformations. An advantage of ResNets is that the gradient can flow directly through the identity function from later layers to the earlier layers. However, ResNets still have many parameters because each layer has its own weights.  DenseNet architecture explicitly differentiates between information that is added to the network and information that is preserved. In contrast to ResNets, DenseNet never combines features through summation before they are passed into a layer; instead, it combines features by concatenating them and containing much fewer parameters [7].

Moreover, DenseNet layers are very narrow, adding only a small set of feature-maps to the “collective knowledge” of the network and keeping the remaining feature maps unchanged—and the final classifier makes a decision based on all feature-maps in the network.[7]

Another big advantage of DenseNets is their improved flow of information and gradients throughout the network, which makes them easy to train. Each layer has direct access to the gradients from the loss function and the original input signal, leading to an implicit deep supervision. This helps training of deeper network architectures.[7]

Despite the many advantages of DenseNet, there is one major disadvantage. The feature maps of each layer are connected with the previous layer, and the data is replicated multiple times. As the number of network layers increases, the number of model parameters grows linearly, eventually leading to explosive growth in computation and memory overhead during training.[8]

DenseNet has achieved better results among many deep learning models due to the new architecture of dense connectivity. [8]

**DenseNet structure:**

The basic structure of DenseNet consists of dense block, transition layer, convolutional layer, and fully connected layer. [8]

Denseblock consists of densely connected dense units with nonlinear mapping functions of  Batch Normalization (BN) , rectified linear units (ReLU) ,Pooling  and a 3X3 Convolution (Conv), which are designed with preactivation strategy to make network training easier and generalization performance better.[8]

The process begins with a single input image x0 that is passed through a convolutional network and a Hℓ(·) that represent a composite function of three consecutive operations: BN, ReLU, Conv. The output of the ℓ th layer as xℓ Consequently, the ℓ th layer receives the feature-maps of all preceding layers, x0, . . ., xℓ−1, as input: [7]



Transition layer is the structure between adjacent dense blocks, which consists of convolution layer and pooling layer, compressing dense block input and all extracted feature information, reducing feature map size and dimensionality, which can effectively reduce the number of dense block parameters and prevent network from overfitting.[8]

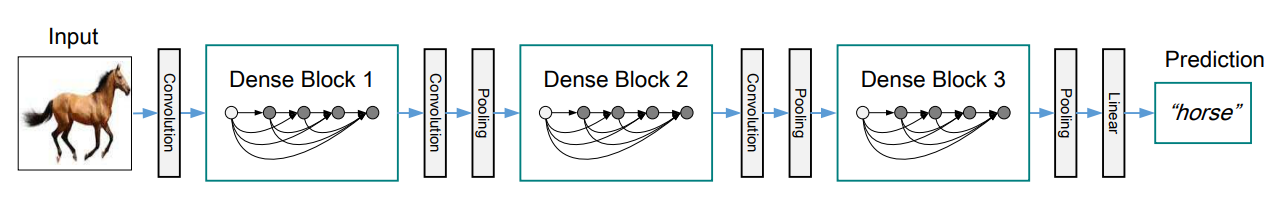


Figure 12: A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature-map sizes via convolution and pooling.

Another parameter of DenseNet is Growth rate each function Hℓ produces k feature maps, it follows that the ℓ th layer has k0 +k ×(ℓ−1) input feature-maps, where k0 is the number of channels in the input layer.[7]

Although each layer only produces k output feature-maps, it typically has many more inputs. To address the increasing dimensions of feature maps, bottleneck units (1×1 convolutions) and compression in transition layers were introduced, as implemented in DenseNet-B, DenseNet-C, and DenseNet-BC, reducing computational effort while preserving diverse features.[7]

In summary, DenseNet, which can learn deeper and more distinctive features from images, has been applied to several research areas of medical image analysis and has made breakthroughs.[8]

**3.2. Description of the research process**

We researched CNN model, DenseNet, and tested it with various learning rates, epochs, dropout rates, and batch sizes on the same dataset. We used platforms such as Kaggle and Google Colab, that provided us services to run projects with big amount of data**.**

Our data set contains a large collection of high-resolution retinal images captured under varying imaging conditions. Each image has been labeled either 0 (Diabetic Retinopathy present) or 1 (No Diabetic Retinopathy). This binary classification task supports early detection efforts and aims to make the analysis easier for healthcare systems by using an automated tool**.**

The Researched hyperparameters evaluated are as follows:

• Learning rate {1e-6 – 1e-7}

• Epochs {100, 150}

• Batch size {32, 64}

• Dropout {0.2-0.5}

We defined the tested parameters this way:

**TP**=True positive, the model predicted that there was an existence of DR, and the retina image was correctly classified.

**TN** = True Negative, the model prediction was that there was no an existence of DR and the retina image was correctly classified.

**FP** = False Positive, the model predicted there was an existence of DR and the retina image was misclassified.

**FN** = False Negative, the model prediction was that there was no an existence of DR and the retina image was misclassified.

**Accuracy** = The relative share of the positive answer in all our parameters. The result will be between zero and one, with one being the best level of accuracy and zero being the worst.  
Accuracy=(TP + TN )/(TP + TN + FP + FN)

**Precision**= retina image correctly classified as DR out of all those classified as DR Precision=TP/(TP+FP)

**Recall**= Out of all those with a DR, the number of retina images correctly classified as a DR.  
Recall=TP/(TP+FN)

**The results we got are:**

In the tables below there are the training (Train\_Loss, Train\_Accuracy), validation (val\_accuracy, val\_loss) and testing (recall, precision, accuracy) results. With those parameters we can see the results of each model in each phase. The tables indicate which hyperparameters impacted the most on the models.

**Train Loss:** 2.68%

**Train Accuracy:** 99.42%

**Validation Loss:** 8.02%

**Validation Accuracy:** 97.12%

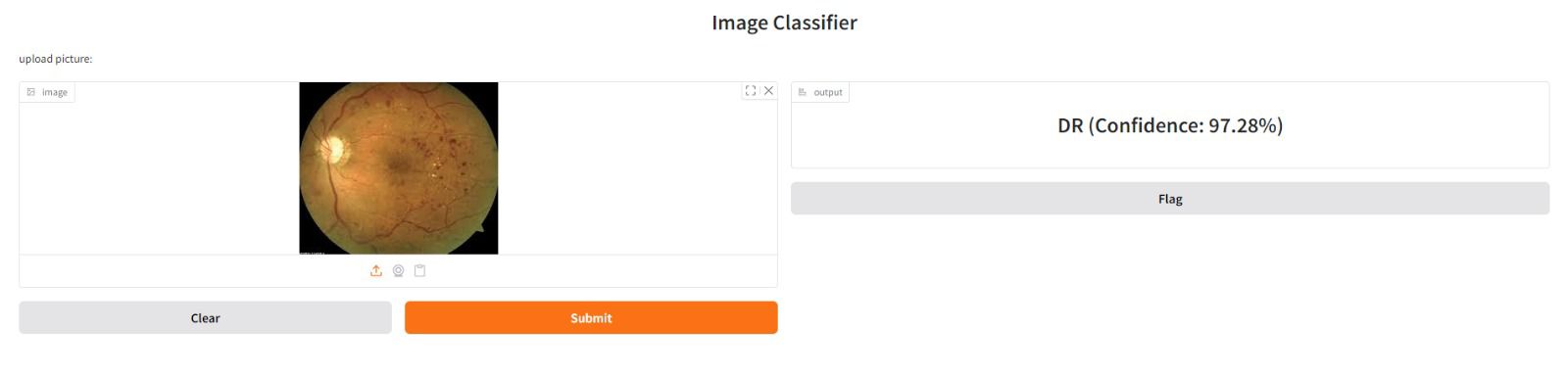
**Test Accuracy:** 96.97%

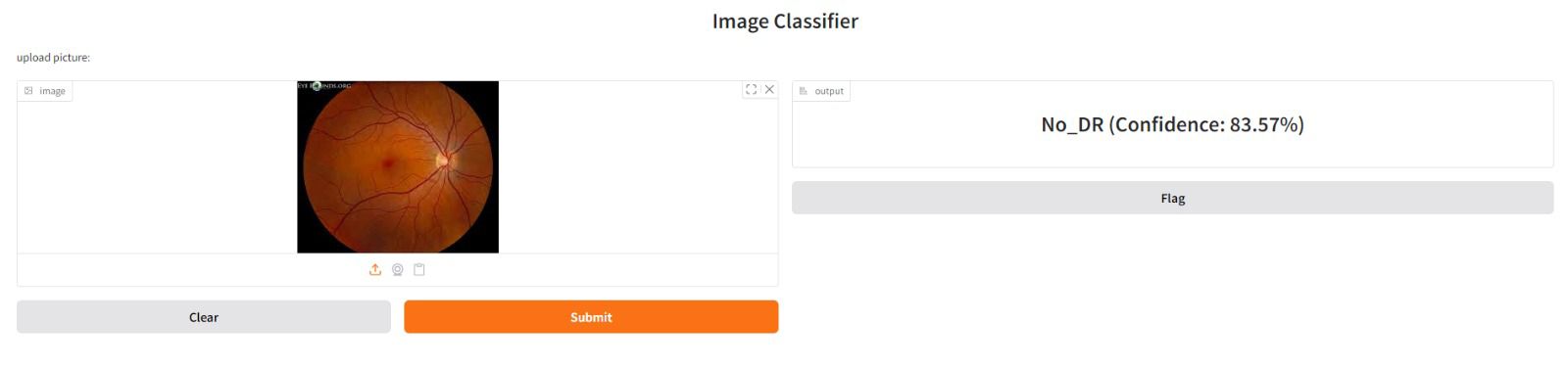
**Precision:** 95.87%

**Recall:** 98.31%



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **b** | **D** | **e** | **Lr** |
|  | A screenshot of a graph  AI-generated content may be incorrect. |  | 32 | 0.3 | 100 | 1.00E-06 |
|  |  |  | 32 | 0.2 | 100 | 1.00E-07 |
|  |  |  | 32 | 0.5 | 100 | 1.00E-06 |
|  |  |  | 32 | 0.3 | 100 | 1.00E-07 |
|  | A graph of different colored lines  AI-generated content may be incorrect. | A graph of different colored lines  AI-generated content may be incorrect. | 32 | 0.5 | 100 | 1.00E-07 |
|  | A graph of a graph  AI-generated content may be incorrect. | A graph of a graph  AI-generated content may be incorrect. | 32 | 0.2 | 100 | 1.00E-06 |
|  | A comparison of graphs with numbers  AI-generated content may be incorrect. | A comparison of graphs with numbers  AI-generated content may be incorrect. | 32 | 0.3 | 150 | 1.00E-06 |
|  |  |  | 32 | 0.5 | 150 | 1.00E-07 |
|  | A comparison of a graph  AI-generated content may be incorrect. | A comparison of a graph  AI-generated content may be incorrect. | 64 | 0.2 | 100 | 1.00E-06 |
|  |  |  | 64 | 0.5 | 100 | 1.00E-06 |
|  |  |  | 64 | 0.3 | 100 | 1.00E-06 |
|  | A comparison of graphs with numbers and symbols  AI-generated content may be incorrect. | A comparison of graphs with numbers and symbols  AI-generated content may be incorrect. | 64 | 0.2 | 100 | 1.00E-07 |
|  | A graph of a graph of a graph  AI-generated content may be incorrect. | A graph of a graph of a graph  AI-generated content may be incorrect. | 64 | 0.2 | 150 | 1.00E-06 |

**3.3 Examples**

****

**3.4. Discussion and Conclusions**

In our research of DenseNet architecture and different hyperparameters on the same data set, we concluded the effects of each hyperparameter on the results. The lowest training loss value and the highest training accuracy was with a learning rate of 0.000001. The lowest validation loss and the highest validation accuracy was with the same learning rate. In this model the precision value is 95%, the accuracy value is 97% and the recall value is 98%. In our research the combination of a learning rate of 1.00E-06, a dropout rate of 0.2, and a batch size of 64 consistently led to the best performance results across all evaluation metrics.

The system delivered impressive results that confirmed the model is suitable for detecting Diabetic Retinopathy within the retina.

The model's speed, combined with it’s high accuracy, can lead to a significant improvement in healthcare systems.

**4.User documentation:**

**4.1Operating instructions - User’s Guide:**

The purpose of our project is to classify if a person has diabetic retinopathy or not, based on retinal images, using a DenseNet model. We experimented with different combinations of hyperparameters, including dropout, learning rates, number of epochs and batch size, to optimize the model’s performance.  
Our system outputs graphs showing the accuracy and loss over the training and validation datasets. In addition, we provide key evaluation metrics such as recall, accuracy, and precision.  
We created a user-friendly graphical interface using Gradio, which allows users to upload an retina image and receive an immediate classification result.

**Maintenance Guide:**

Install Required Packages

Download and Load Dataset

Preprocess the Data

Set the Hyperparameters

Train the Model and Evaluate on Validation Set

Test the Model

Save the Model in Google Drive

**4.2 Description of the organization and structure of the data:**

We used the "Diagnosis of Diabetic Retinopathy" dataset from Kaggle. This dataset consists of a large collection of high-resolution retinal images captured under various imaging conditions labeled according to DR or No\_DR. The dataset is organized into three main subsets: Training, Validation, and Testing, each subset consists of image files stored in separate folders based on their classification labels.

To ensure consistency and improve performance, we applied preprocessing steps including resizing the images to a fixed size (224x224).

**Description of the operating environment of the system:4.3**

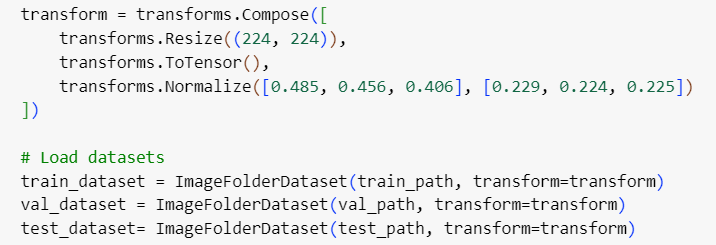
1. Set up Kaggle API key

2. Go to the [link](https://www.kaggle.com/datasets/pkdarabi/diagnosis-of-diabetic-retinopathy) and download the data

A close-up of a computer code

AI-generated content may be incorrect.3. Prepare Dataset Paths

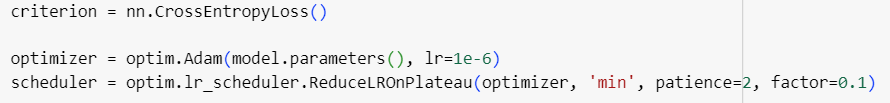
4. Define Data Preprocessing



5. Initialize Model with a specified number of output classes and dropout rate



6. Set Loss Function and Optimizer



7. Train the Model

8. Test the Model



9.open the GUI

A screenshot of a computer

AI-generated content may be incorrect.

10.upload a picture

11.press submit



12.Get result:



**References**

[1] American Academy Of Ophthalmology [link](https://www.aao.org/eye-health/diseases/what-is-diabetic-retinopathy)

[2] Dai, Ling, Liang Wu, Huating Li, Chun Cai, Qiang Wu, Hongyu Kong, Ruhan Liu et al. "A deep learning system for detecting diabetic retinopathy across the disease spectrum." *Nature communications* 12, no. 1 (2021): 3242

[3] Hacisoftaoglu, Recep E., Mahmut Karakaya, and Ahmed B. Sallam. "Deep learning frameworks for diabetic retinopathy detection with smartphone-based retinal imaging systems." *Pattern recognition letters* 135 (2020): 409-417.

[4] Qureshi, Imran, Jun Ma, and Qaisar Abbas. "Diabetic retinopathy detection and stage classification in eye fundus images using active deep learning." *Multimedia Tools and Applications* 80, no. 8 (2021): 11691-11721.

[5] World Health Organization [link](https://www.who.int/news-room/fact-sheets/detail/diabetes)

[6] Zhang, Lirong, Jialin Gang, Jiangbo Liu, Hui Zhou, Yao Xiao, Jiaolin Wang, and Yuyang Guo. "Classification of diabetic retinopathy algorithm based on a novel dual-path multi-module model." *Medical & Biological Engineering & Computing* (2024): 1-17.

6 ->1 3->2 2->3 1->4 4->6