EYENURI

By Azadeh & Buse

Content

1. The Road So Far
    a. Understanding Diabetic Retinopathy
    b. Method: Fundus Photography and Deep Learning
    c. Convolutional Neural Network
    d. Neural Network Architectures
2. Platforms and Languages Used
3. Used Libraries in Python
4. Data Selection
5. Data Preprocessing
6. Evaluation of the Model
7. Explanation of the Model
8. Prototype: EYENUR The App
9. Challenges
10. Next steps

EYENUR

Understanding Diabetic Retinopathy

Problem

Diabetic retinopathy (DR)

- Microvascular disorder that causes vision loss
- Long term effect of diabetes mellitus
- Very common for Type I diabetic patients
- Has 5 severity levels

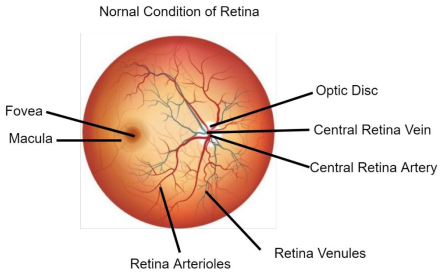
Painpoints

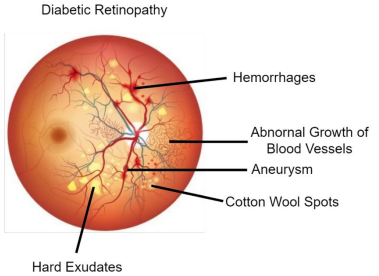
- Regular check-ups are required
- OCT is practical but appointments with a specialist is necessary
- Treatments have many side effects and expensive

Solution

An app that uses an image of your fundus to detect if your eye is healthy or not

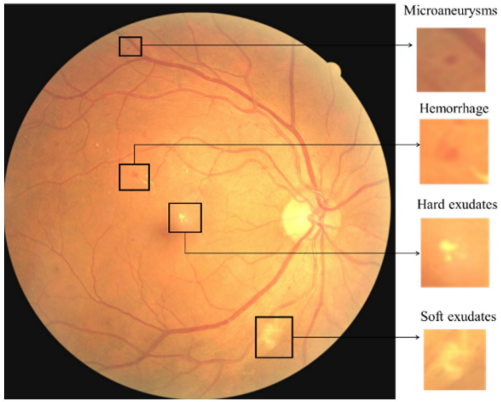
- Time and cost efficient
- Patient friendly = Self-diagnosis
- Allows early diagnosis
- Extra source for the ophthalmologists

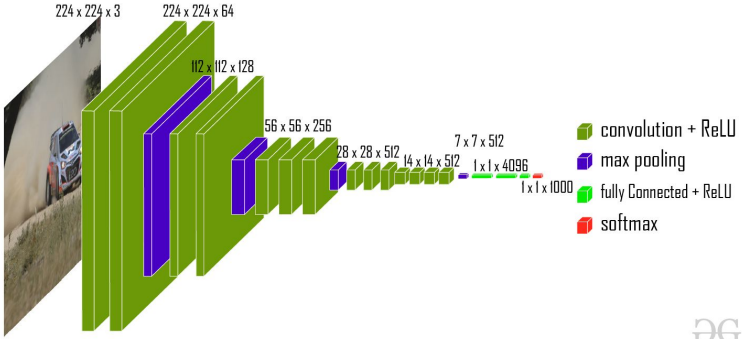




EYENUR

Method: Fundus Photography and Deep Learning

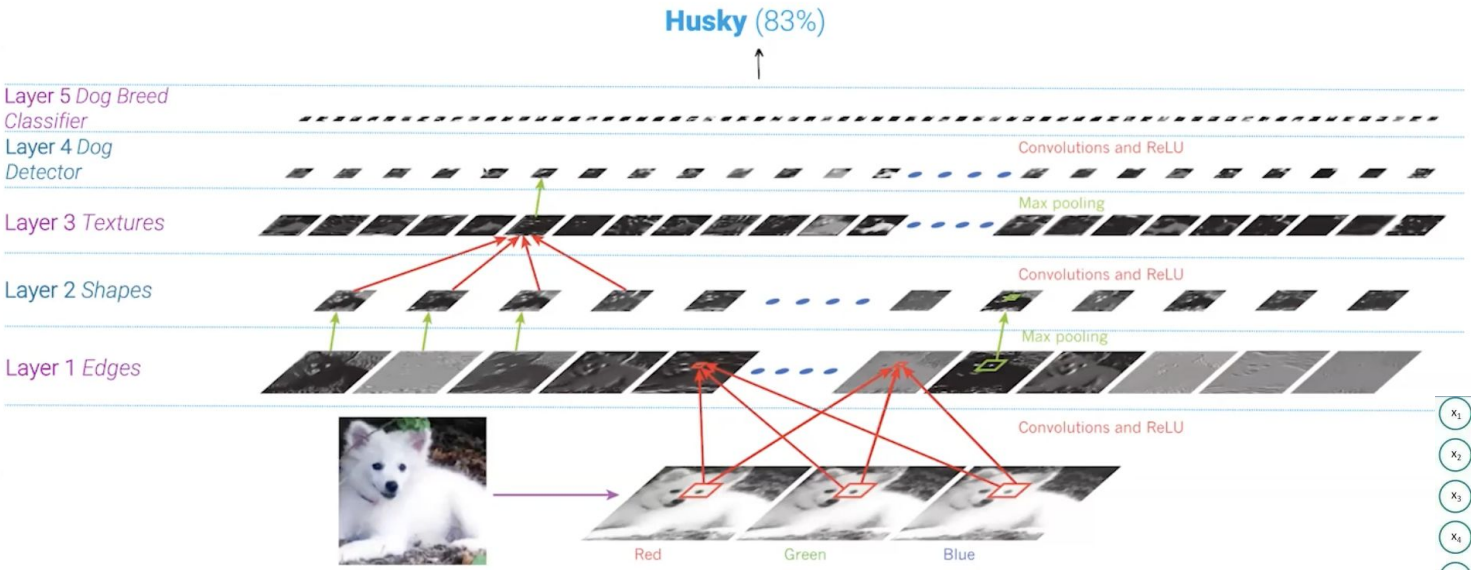




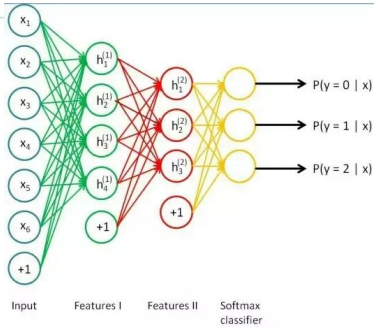
1. **Fundus image** preprocessing, where the data is obtained from **Kaggle**
2. Training and testing 4 different pre-trained **CNN models** and determine the best approach
   a. ResNet50
   b. Efficient Net
   c. Inception V3
   d. VGG16
3. TensorFlow model → Core ML model
4. Integrate the model to the **IOS app**

EYENUR

Convolutional Neural Network (CNN)



LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature (2015)*

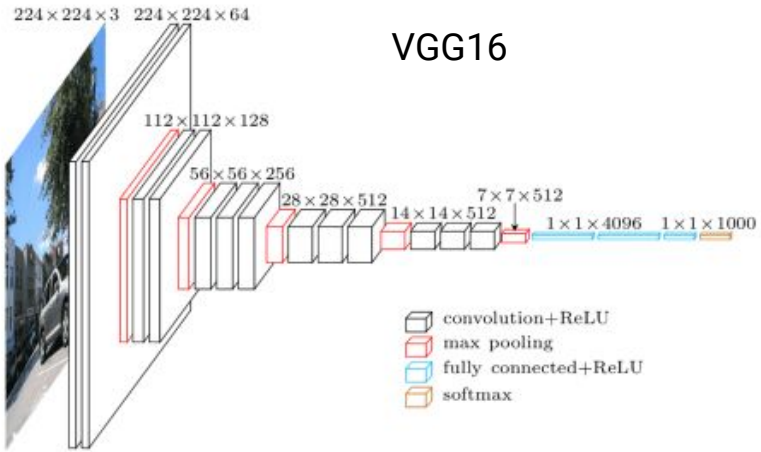


Stacked Autoencoders for the P300 Component Detection, Front. Neurosci., 2017

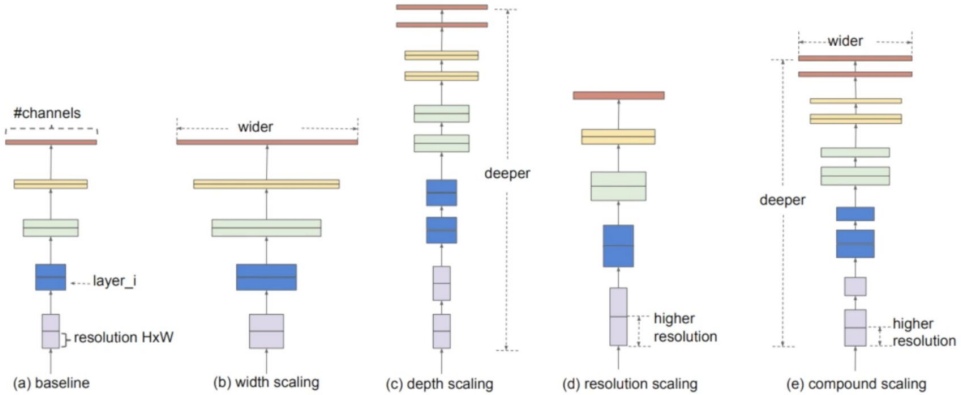
EYENUR

CNN Architectures

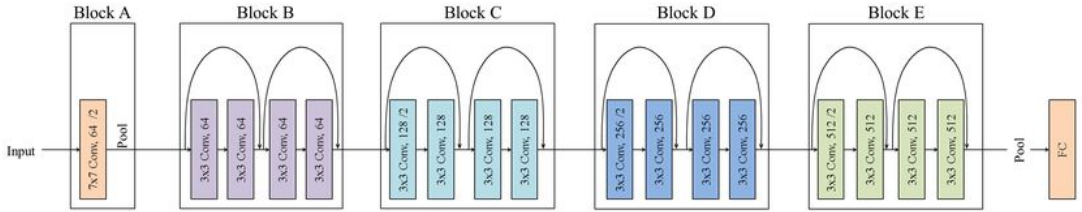
VGG16



EfficientNet



ResNet



EYENUR

Platforms and Languages Used

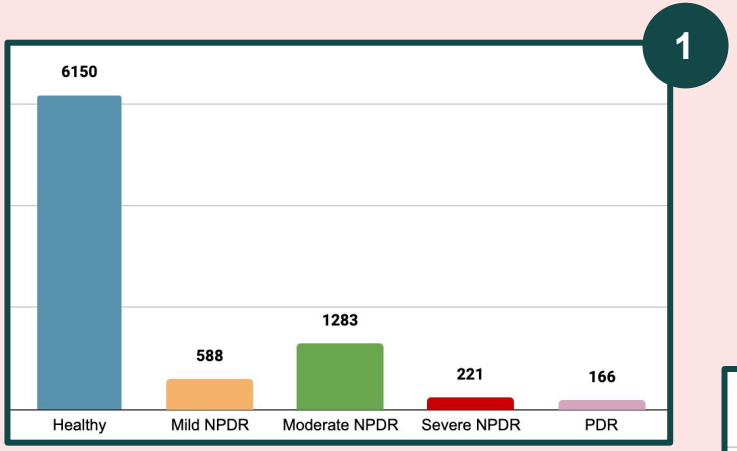
- Kaggle: For data and computing (GPU advantage)
- Visual Studio Code: IDE for Python
- XCode: IDE for Swift

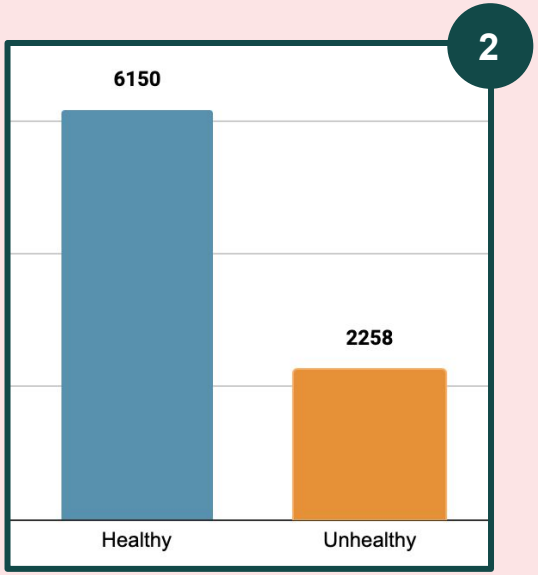
Used Libraries in Python

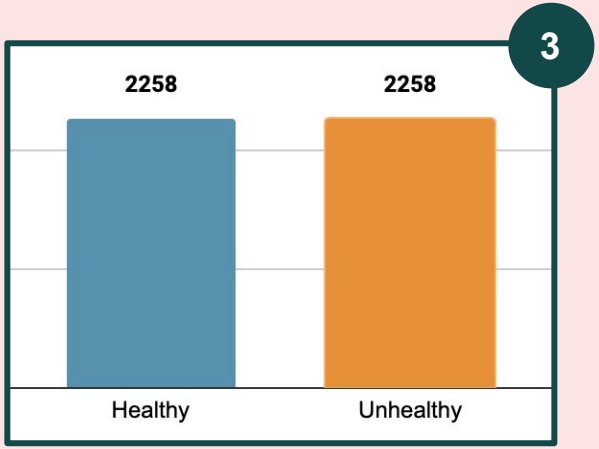
```
import [library name]
from [library name] import [function name]
```

- **tensorflow:** An end-to-end platform used for ML, e.g. data preprocessing, transforming, loading, validating
- **keras:** A part of TensorFlow. Most common deep learning framework. Improves the process of user actions
- **shutil:** High-level handling of files and directory, such as file copying or removing
- **pandas:** High-level data analysis, easy and flexible, can be used on labeled data sets
- **numpy:** Mainly for array computing, mathematical functions
- **matplotlib.pyplot:** Visualisations, and pyplot functions are for altering figures
- **seaborn:** Based on matplotlib, and used for creating statistical graphics
- **cv2 (OpenCV-Python):** Reading an image from a specific file
- **imutils:** Image processing, e.g. rotation, resizing
- **coremltools:** Converts trained ML models to Core ML format to integrate the model to the app

Data Selection







Data Preprocessing

Sorting images to right files

Selecting and processing a fraction of the Kaggle data

Checking the distribution of healthy and 4 different stages of unhealthy images

Categorization

Classification of data based on the distribution:

- **Class 0 = Healthy**
- **Class 1= Unhealthy**

Balancing no. of samples in each class for a better model

Separating no. of samples for training, validating and testing stages.

- **Training: 3162**
- **Validating: 678**
- **Testing: 676**

Image preparation

- Cropping
- Threshold
- Erode
- Dilate
- Contour
- Resize
- Specific preprocessing for model

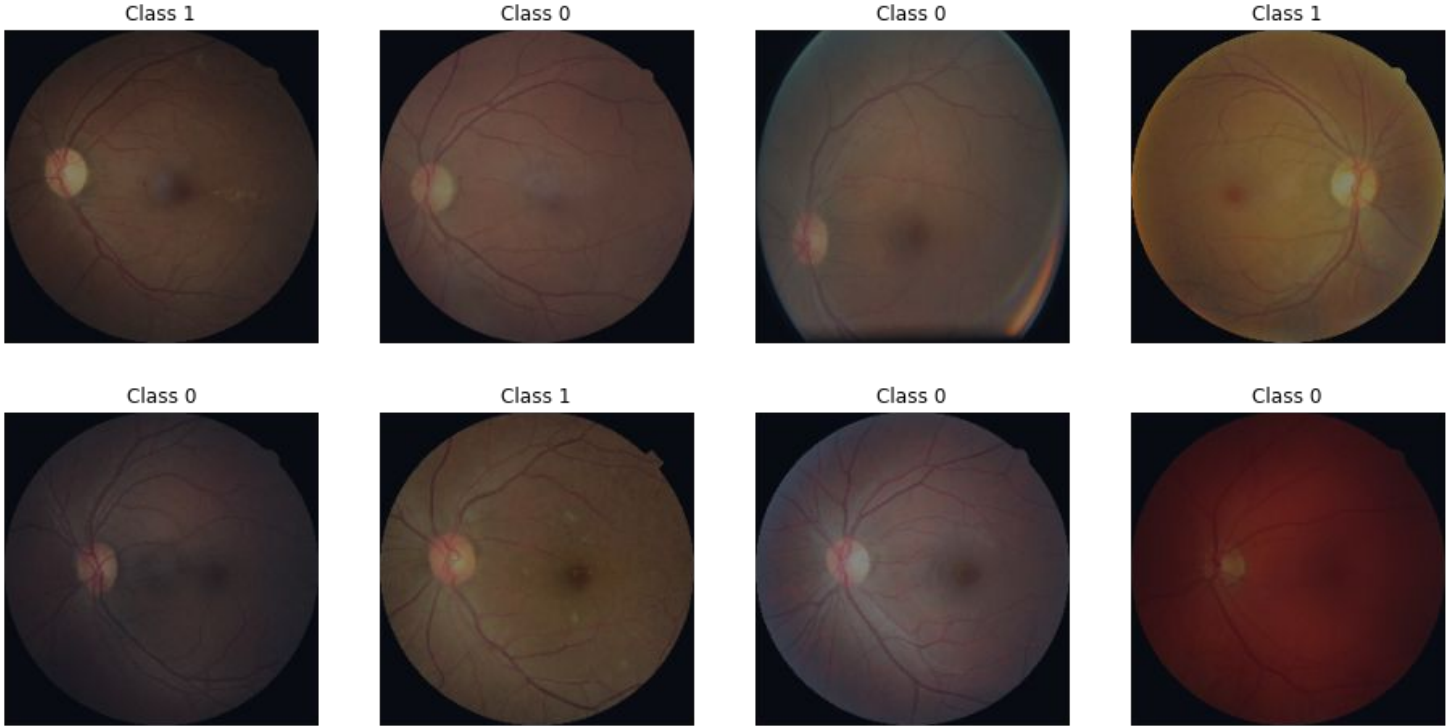
Image conversion for the model

Every model has its own image processing method.

So all the images must be converted to a specific format for each individual model.

EYENUR

Data Preprocessing

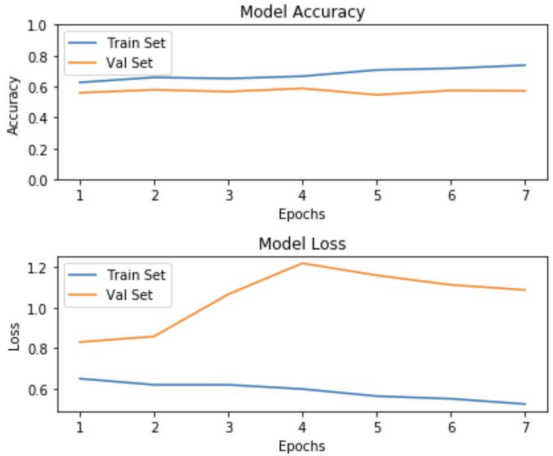


EYENUR

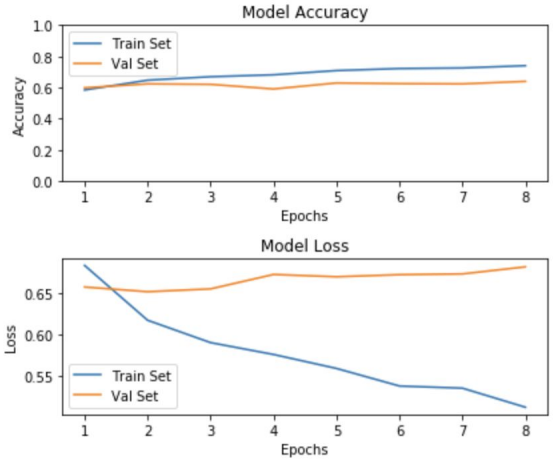
Evaluation of the Model

Training and Validation

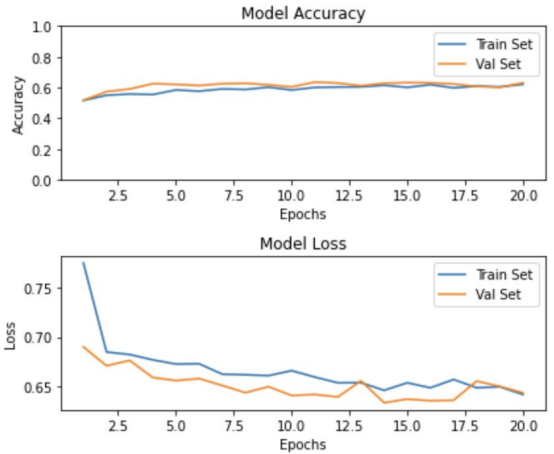
ResNet50



VGG16



EfficientNet

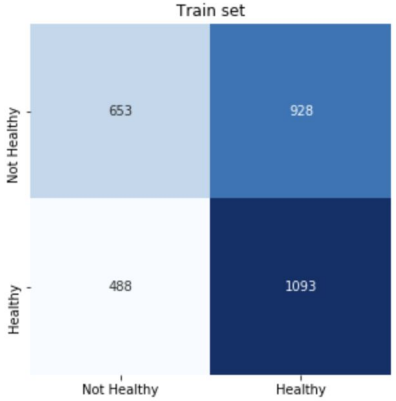


EYENUR

Evaluation of the Model

Train Set and Validation Set

ResNet50



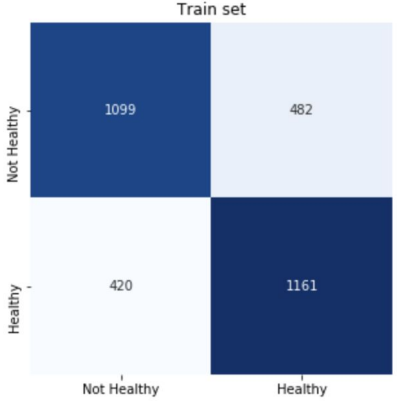
Accuracy: 0.55%

	precision	recall	f1-score	support
0.0	0.57	0.41	0.48	1581
1.0	0.54	0.69	0.61	1581
avg / total	0.56	0.55	0.54	3162

Accuracy: 0.57%

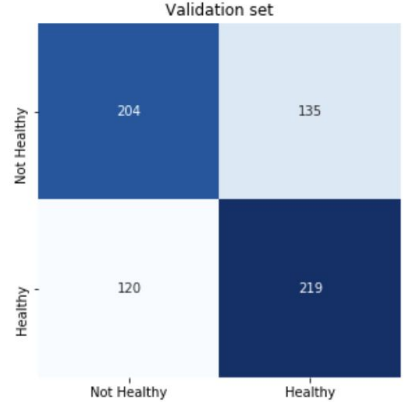
	precision	recall	f1-score	support
0.0	0.60	0.44	0.51	339
1.0	0.56	0.71	0.62	339
avg / total	0.58	0.57	0.56	678

VGG16



Accuracy: 0.71%

	precision	recall	f1-score	support
0.0	0.72	0.70	0.71	1581
1.0	0.71	0.73	0.72	1581
avg / total	0.72	0.71	0.71	3162



Accuracy: 0.62%

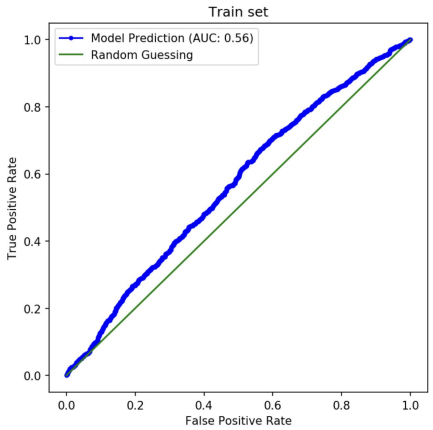
	precision	recall	f1-score	support
0.0	0.63	0.60	0.62	339
1.0	0.62	0.65	0.63	339
avg / total	0.62	0.62	0.62	678

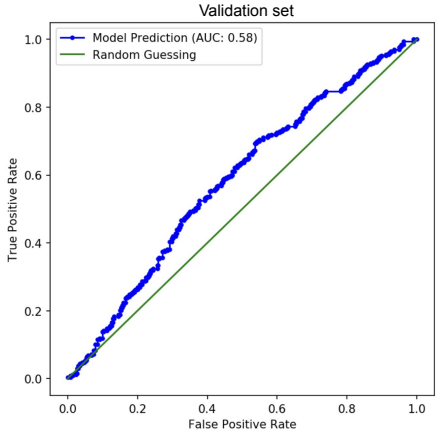
EYENUR

Evaluation of the Model

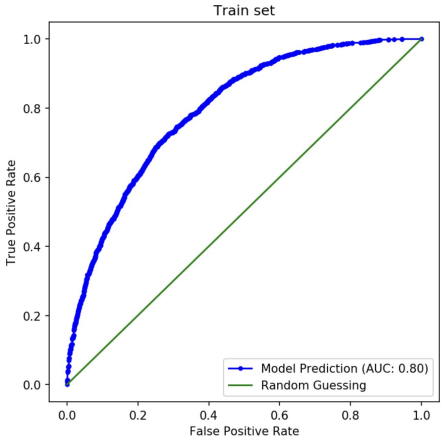
Train Set and Validation Set

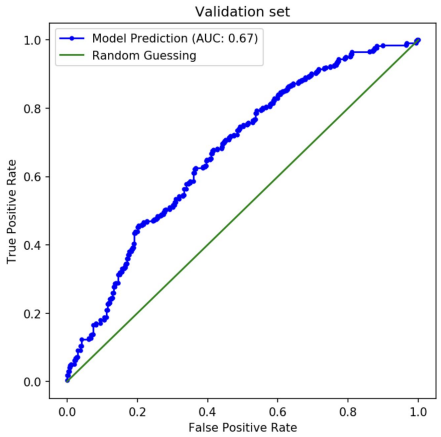
ResNet50





VGG16



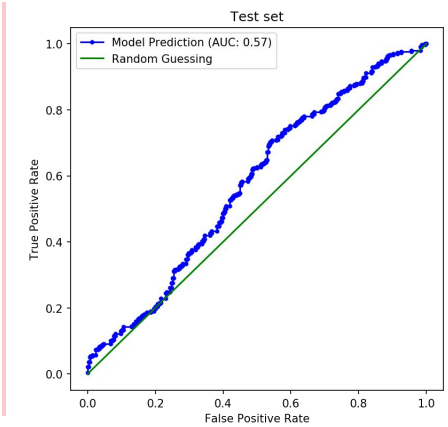


EYENUR

Evaluation of the Model

Test Set

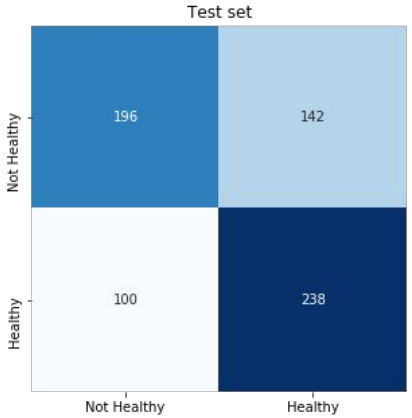
ResNet50

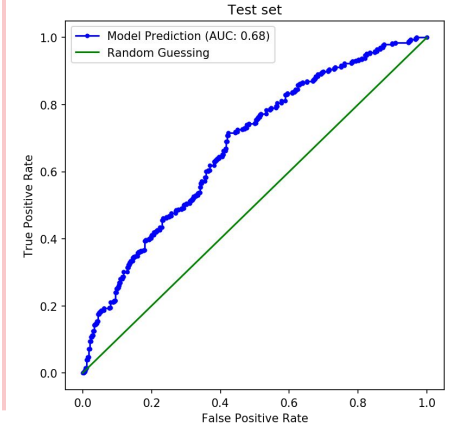


Accuracy on Test Data: 0.51%

	precision	recall	f1-score	support
0.0	0.51	0.75	0.61	338
1.0	0.52	0.28	0.36	338
avg / total	0.52	0.51	0.48	676

VGG16



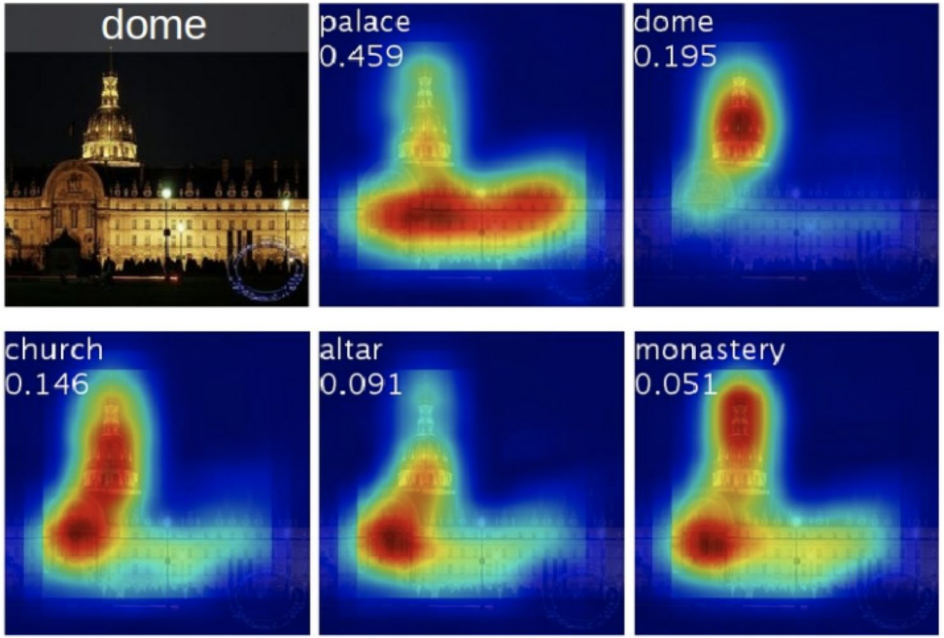


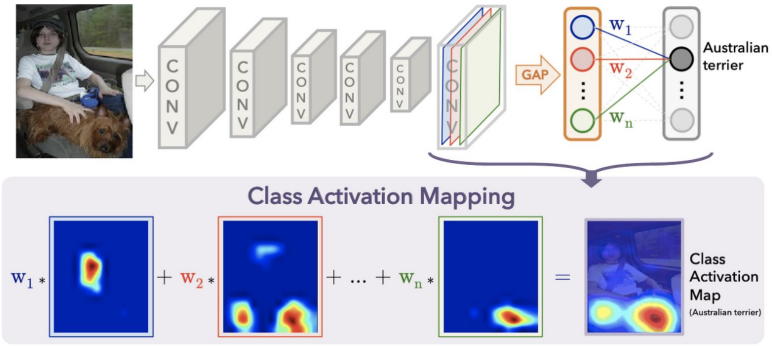
Accuracy on Test Data: 0.64%

	precision	recall	f1-score	support
0.0	0.66	0.58	0.62	338
1.0	0.63	0.70	0.66	338
avg / total	0.64	0.64	0.64	676

Explanation of the Model

Attention Maps: A way of improving the performance of the model by extracting the relevant feature of the input



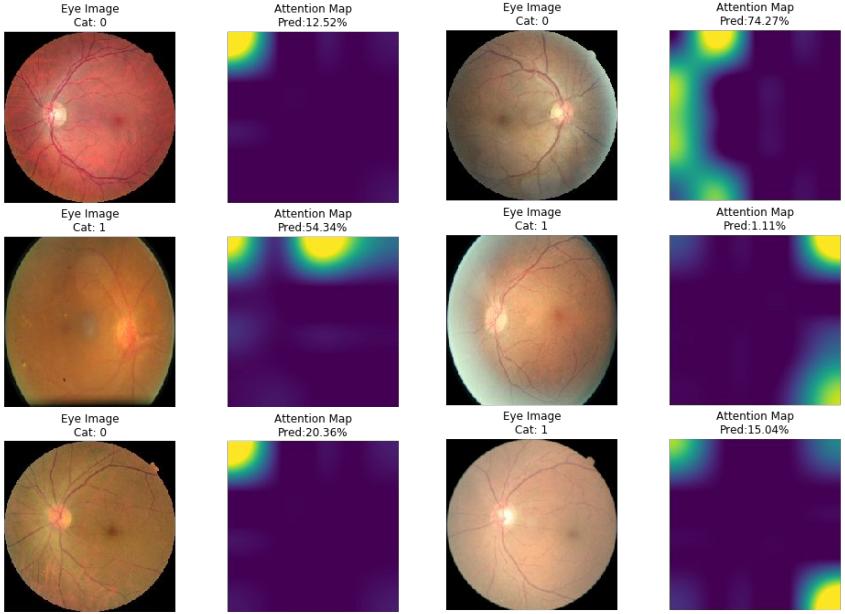


EYENUR

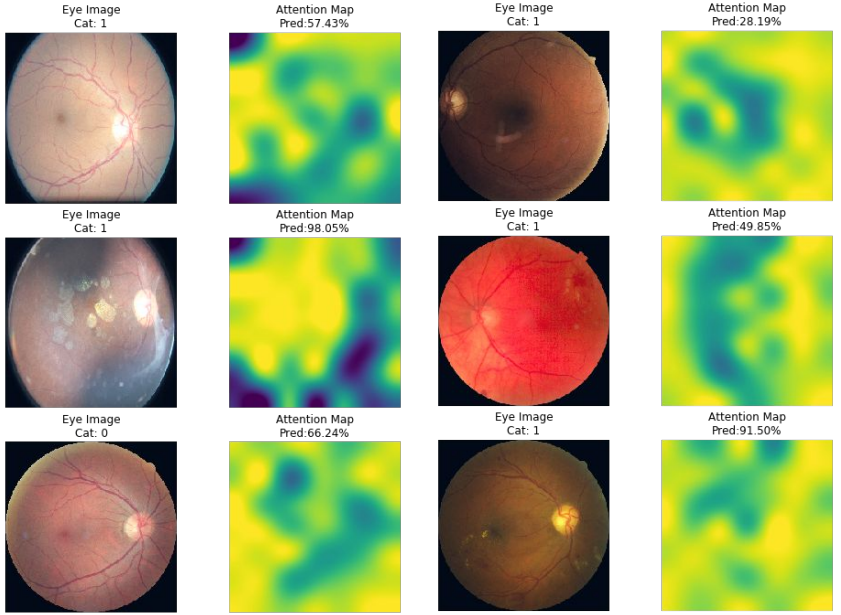
Explanation of the Model

Attention Maps

ResNet50

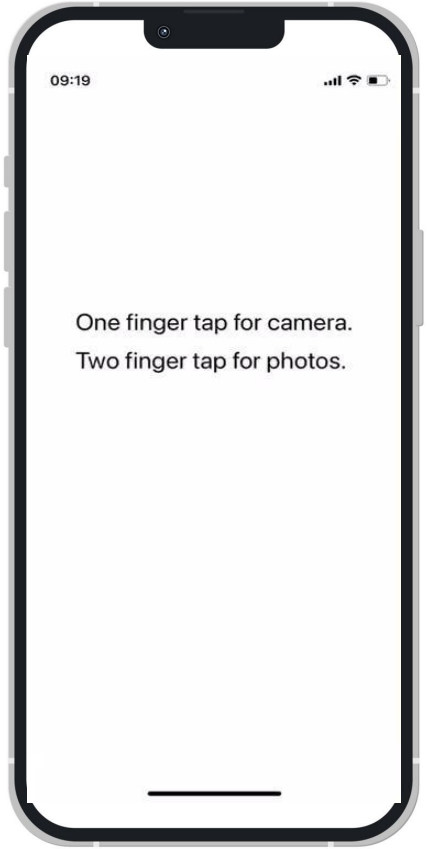


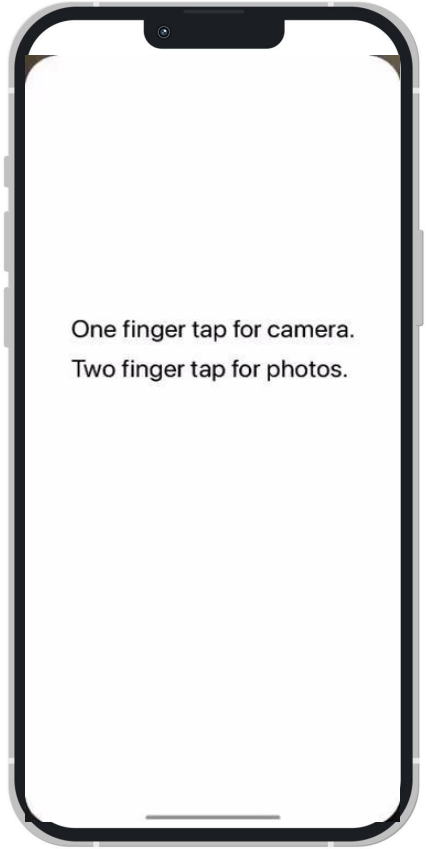
VGG16



EYENUR

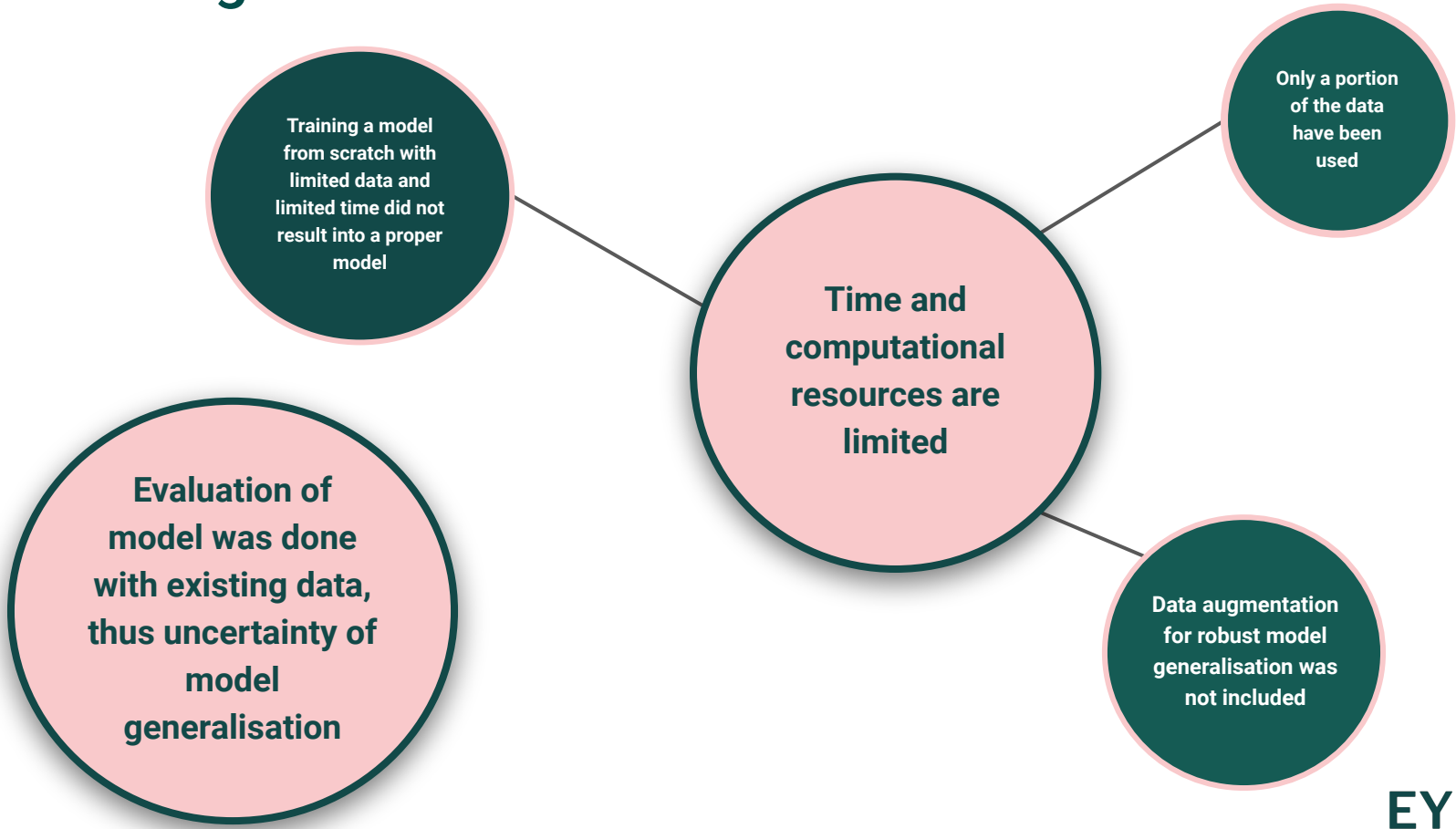
Prototype: EYENUR The App





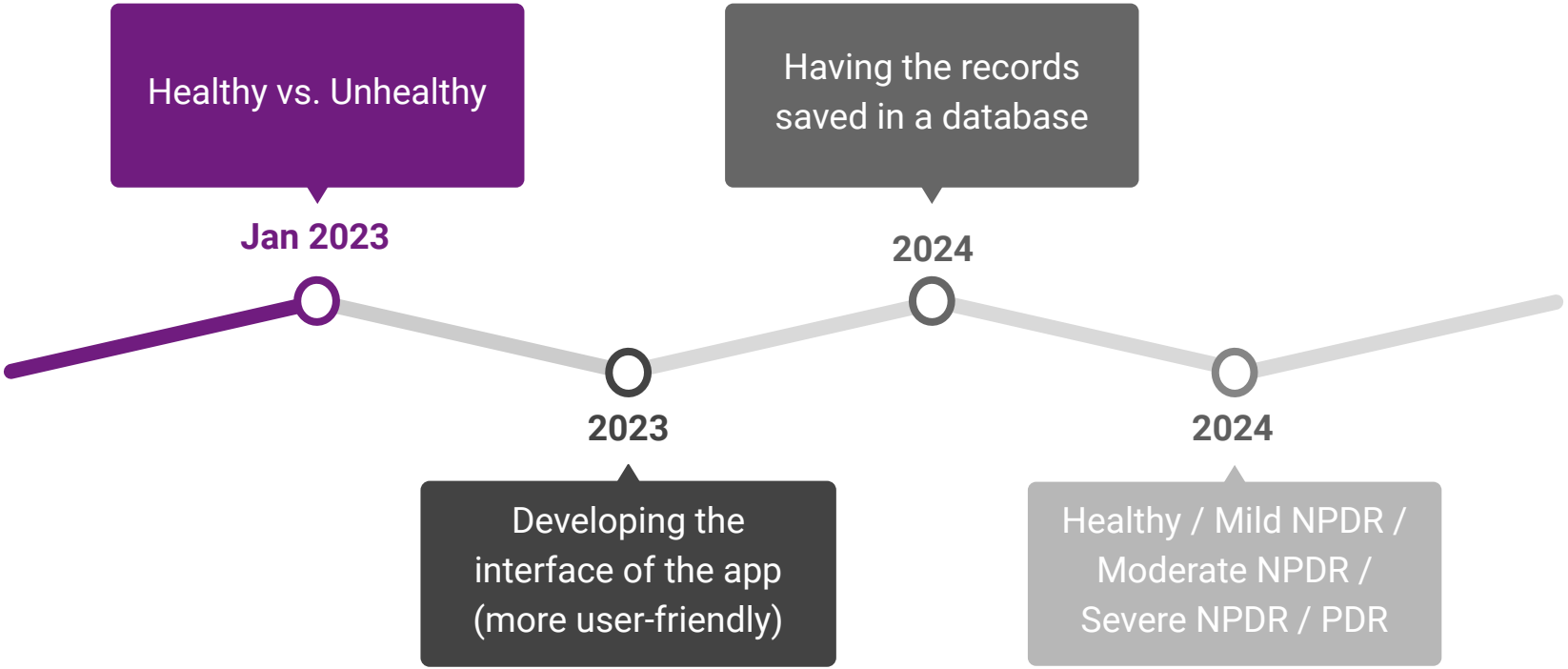
EYENUR

Challenges



EYENUR

Next Steps



EYENUR

Thank You!