DIABETIC RETINOPATHY DETECTION

A PREPRINT

David Unger
XXXXXXX
M. Sc. Autonome Systeme
University of Stuttgart
stXXXXXX@stud.uni-stuttgart.de

Nick Wagner
3524444
MSc. Autonome Systeme
University of Stuttgart
st175644@stud.uni-stuttgart.de

February 11, 2022

ABSTRACT

Diabetic retinopathy is an eye disease that can affect people suffering diabetes. It causes damage to the blood vessels of the eyes, deteriorates the eyesight and can lead in the worst case to blindness of the patient. It is important to detect the disease in an early stage to mitigate it as good as possible with an early treatment. Analyzing images of eyes and classify the severity of diabetic retinopathy is a challenging task that requires expert knowledge. To assist doctors and medical personnel, a classification model shall be trained to classify the severity automatically.

1 Introduction

Diabetic retinopathy is a complication of diabetes, which can cause damage the retina of the eye. If not detected early, this damage may cause cause vision impairment or even blindness. To treat this condition successfully, it has to be detected at an early stage, which is difficult due to minimal to no early warning signs. Furthermore, the different grades can only be distinguished by a trained professional due to its subtle symptoms. Examples are leaking blood vessels, fatty deposits or retinal swelling. Since this task is difficult even for trained professionals, an algorithm for automatic detection of the diabetic retinopathy grade is necessary. This is the goal of this paper.

The used dataset is the Indian Diabetic Retinopathy Image Dataset (IDRID), which is publicly available. It contains five class labels, which refer to the different eye disease grades (0-4).

2 Object Classification

2.1 Problem analysis

To tackle the problem of diabetic retinopathy detection, several methods are possible. Because the dataset consists of ordinally scaled data of 5 classes, regression could be used to estimate the serverity of a case. In addition, a the problem can be handled as a classification problem after one-hot-encoding the labels. As a third option, one can define a threshold to define problematic diabetic retinopathy and non-problematic diabetic retinopathy and can handle the problem as a binary classification. Further, only binary and multiclass classification are anaylzed.

A binary classification has the advantage of higher accuracy, but lacks details, because the network only outputs 0 or 1 and no information about the exact serverity of the disease. Metrics are also easy to implement, because precision, recall and f1-score are standard implementations and nicely interpretable.

A multiclass classification has typically a lower accuracy, because the network needs to pick the right class among several classes. It provides the benefit or receiving richer information, i.e. the exact serverity of the disease. Evaluating a multiclass classification problem becomes harder, because missclassifications can vary in their error. Classifiying a class 1 as class 2 is for example less problematic than classifying class 1 as class 5.

2.2 Architecture

2.2.1 Custom architecture

» draw 3D architecture in powerpoint VGG, Resnet, Weight freeeze / unfreeze, GAP, Flatten, Dense Layers

2.3 Weight initialization

Weight initialization refers to the initial values of parameters that are used in specific neural network layers. Changing the initialization of the layers changes the starting point for the optimization process and potentially also the performance. Keras initializes the weights of dense layers with the Glorot uniform initializer, which draws samples from a truncated uniform distribution. Generally the goal is to avoid vanishing and exploding gradients, even better is if the variance of layer outputs is approximately one. [paper KUMAR] Specifically for the ReLU activation function, this is achieved with the He initialization, which draws from a normal distribution with the following parameters. [paper HE]

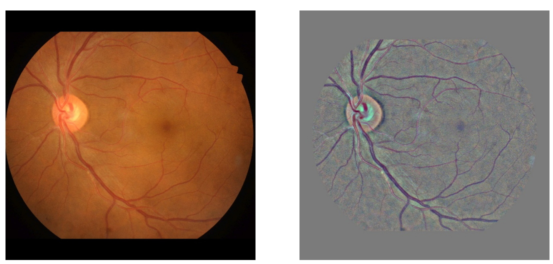
$$\mu = 0 \quad (1) \qquad \sigma^2 = 2/N \quad (2)$$

This led to a faster training and performance increase of X percent.

2.4 Augmentation

Within the input pipeline three main types of augmentation are applied. The goal is to make feature extraction easier for the network and increase the amount of input images which reduces overfitting.

- **Graham preprocessing: [report kaggle]**
  1. rescale the eye radius to 300 pixels
  2. subtract the local average color such that the local average gets mapped to gray
  3. clip the image to remove boundary effects]



(a) Original image (b) Processed image

Figure 1: Graham pre-processing

- **Color jittering:** Slight random changes to brightness, hue, saturation or contrast.
- **Random cropping:** Crop a window that is slightly smaller than the image. Then resize to original size again.

2.5 Dataset Balancing

Taking a closer look at the sample distribution within the dataset 1, it is obviously imbalanced. This will inevitably lead to a trained model that is fitted better to the overrepresented classes.

Label:	0	1	2	3	4
# train	134	20	136	74	49
# test	34	5	32	19	13

Table 1: Dataset sample distribution

To avoid this, a method called oversampling is applied. Oversampling allows drawing underrepresented classes from the training set more often, such that all classes are equally represented when training the model.

2.6 Training

2.6.1 Optimizer

2.6.2 Loss Function

2.6.3 Learning Rate

Momentum / Decay

2.7 Metrics

Traning deep neural networks requires some performance metrics indicating the success or failure of the model fitting. Accuracy is easy to understand and to implement, but not suited for imbalanced data, because it doesn't take into account how well each single class is predicted. To overcome this, precision (how accurate was the model with its prediction) and recall (how thorough is a model with its prediction) can be used. Especially a combination of both, the f1-score is a metrics that indicates how well a classifier handles different classes.

For the diabetic retinopathy dataset, the f1-score comes with a drawback. The dataset is ordinally scaled, which means predicting a wrong class is not equally bad. As an example is predicting class 4 for the true class 0 much worse that predicting a 1. The f1-score would count both missclassifications equally, while the Quadratic Weighted Kappa (QWC) metric takes the distance of the classification error into account.

3 Experiments

3.1 Procedure

The training of the deep neural network classifier requires the selection of suitable hyperparameters that differ from problem to problem. A useful strategy to find a good set of hyperparameters are parameter sweeps. Weights&Biases is a python library that enables the easy implementation of sweeps.

Hyperparameter optimization requires besides training and test dataset a third, the validation dataset, to evaluate the model after hyperparameter tuning and to avoid overfitting on the hyperparameters. Because the given dataset only contains training and test data, the original training dataset was split into 80% training data and 20% validation data.

In total, x sweeps consisting of x runs and x epochs were performed during the project.

3.2 Hyperparameter selection

The following parameters show a big effect on the performance of the neural network on the validation data, why they are selected for the final classifier.

Balancing ...

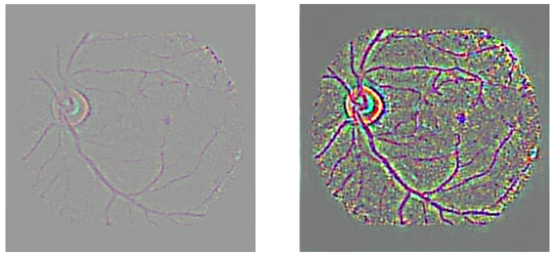
3.3 Deep Visualization

3.3.1 Guided Backpropagation

Guided Backpropagation belongs to the family of pixel-space gradient visualizations. The goal is to exploit the idea that neurons act like detectors of image features by using backpropagation. What makes this backpropagation guided is that negative gradient are set to zero. This way, only pixels that are positively important to the output get highlighted. This method is not class-discriminative.

3.3.2 CAM

Class Activation Map (CAM) is class-discriminative and highlights relevant image regions, but in a less fine-grained manner To achieve this, global average pooling has to be performed on the last feature maps of the last convolutional layer, followed by a dense layer as output. The weights of this dense layer are then projected back to the convolutional feature maps which results in the class activation mappings. As a consequence, it is hard to generalize this approach due to the architecture restrictions.



(a) Original result (b) Enhanced result

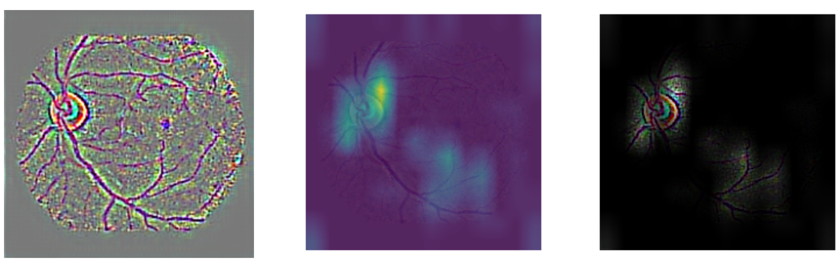
Figure 2: Guided Backpropagation

3.3.3 GradCAM

GradCAM is class-discriminative as well, but can be used by any CNN-based network without architectural adaptions. Similar to CAM, it describes the activation as a linear combination of weighted feature maps. But in this case, the weights are calculated by deriving the logit per class by the feature maps of the chosen convolutional layer. These gradients then get global average pooled, followed by a ReLU. The ReLU leaves only feature with a positive impact behind, similar to how it is applied in Guided Backpropagation.

3.3.4 Guided GradCAM

Guided GradCAM combines the ideas of Guided Backpropagation with Class Activation Maps. This makes GradCAM class-discriminative due to the localization of relevant image regions and high-resolution due pixel-space gradient visualization.



(a) Guided Backprop. (b) gradCAM (c) Guided gradCAM

Figure 3: Guided Backpropagation

Note: The chosen model architecture leads to graph issues within the gradCAM algorithm. The workaround affects the result negatively. A working example with another model and image can be found in the code within the "helper" folder.

4 Results

best binary + multiclass performance; color coded confusion matrix

5 Headings: first level

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. See Section 5.

5.1 Headings: second level

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

$$\xi_{ij}(t) = P(x_t = i, x_{t+1} = j | y, v, w; \theta) = \frac{\alpha_i(t) a_{ij}^{w_t} \beta_j(t+1) b_j^{v_{t+1}}(y_{t+1})}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i(t) a_{ij}^{w_t} \beta_j(t+1) b_j^{v_{t+1}}(y_{t+1})} \tag{3}$$

5.1.1 Headings: third level

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Paragraph Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

6 Examples of citations, figures, tables, references

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus. Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui. [**?**, **?**] and see [**?**].

The documentation for `natbib` may be found at

`http://mirrors.ctan.org/macros/latex/contrib/natbib/natnotes.pdf`

Of note is the command `\citet`, which produces citations appropriate for use in inline text. For example,

```
\citet{hasselmo} investigated\dots
```

produces

> Hasselmo, et al. (1995) investigated…

`https://www.ctan.org/pkg/booktabs`

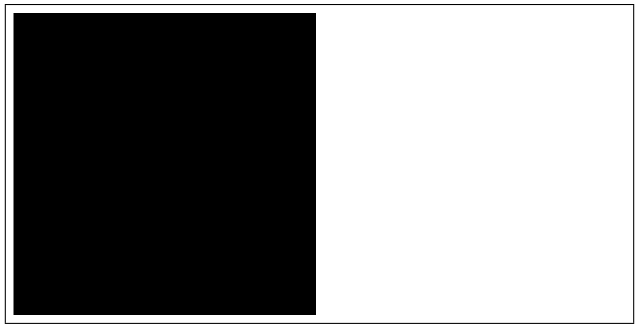


Figure 4: Sample figure caption.

Table 2: Sample table title

Part		
Name	Description	Size (μm)
Dendrite	Input terminal	$\sim$100
Axon	Output terminal	$\sim$10
Soma	Cell body	up to 10^6

6.1 Figures

Suspendisse vitae elit. Aliquam arcu neque, ornare in, ullamcorper quis, commodo eu, libero. Fusce sagittis erat at erat tristique mollis. Maecenas sapien libero, molestie et, lobortis in, sodales eget, dui. Morbi ultrices rutrum lorem. Nam elementum ullamcorper leo. Morbi dui. Aliquam sagittis. Nunc placerat. Pellentesque tristique sodales est. Maecenas imperdiet lacinia velit. Cras non urna. Morbi eros pede, suscipit ac, varius vel, egestas non, eros. Praesent malesuada, diam id pretium elementum, eros sem dictum tortor, vel consectetuer odio sem sed wisi. See Figure 4. Here is how you add footnotes. [1] Sed feugiat. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Ut pellentesque augue sed urna. Vestibulum diam eros, fringilla et, consectetuer eu, nonummy id, sapien. Nullam at lectus. In sagittis ultrices mauris. Curabitur malesuada erat sit amet massa. Fusce blandit. Aliquam erat volutpat. Aliquam euismod. Aenean vel lectus. Nunc imperdiet justo nec dolor.

6.2 Tables

Etiam euismod. Fusce facilisis lacinia dui. Suspendisse potenti. In mi erat, cursus id, nonummy sed, ullamcorper eget, sapien. Praesent pretium, magna in eleifend egestas, pede pede pretium lorem, quis consectetuer tortor sapien facilisis magna. Mauris quis magna varius nulla scelerisque imperdiet. Aliquam non quam. Aliquam porttitor quam a lacus. Praesent vel arcu ut tortor cursus volutpat. In vitae pede quis diam bibendum placerat. Fusce elementum convallis neque. Sed dolor orci, scelerisque ac, dapibus nec, ultricies ut, mi. Duis nec dui quis leo sagittis commodo. See awesome Table 2.

6.3 Lists

- Lorem ipsum dolor sit amet
- consectetur adipiscing elit.
- Aliquam dignissim blandit est, in dictum tortor gravida eget. In ac rutrum magna.

References

[1] Siddharth Krishna Kumar. On weight initialization in deep neural networks. *arXiv:1704.08863*, 2017.

[2] name. title. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 417–422. IEEE, 2014.

[1] Sample of the first footnote.

Guided Backpropagation: https://arxiv.org/abs/1412.6806