

---

SAARLAND UNIVERSITY

Faculty of Mathematics and Computer Science  
Department of Computer Science  
Bachelor Thesis

---



# Evaluating the Impact of Pre-Processing Methods on Deep Learning models in Diabetic Retinopathy Classification

submitted by  
Robert Andreas Leist  
Saarbrücken  
August 2022

---

**Advisor:**

Alexander Prange  
German Research Center for Artificial Intelligence (DFKI)  
Saarland Informatics Campus  
Saarbrücken, Germany

**Reviewers:**

Prof. Dr. Daniel Sonntag  
German Research Center for Artificial Intelligence (DFKI)  
Saarland Informatics Campus  
Saarbrücken, Germany

Prof. Dr. Antonio Krüger  
German Research Center for Artificial Intelligence (DFKI)  
Saarland Informatics Campus  
Saarbrücken, Germany

**Submitted on:**

15th August 2022

Saarland University  
Faculty MI – Mathematics and Computer Science  
Department of Computer Science  
Campus - Building E1.1  
66123 Saarbrücken  
Germany

## **Declarations**

### **Statement in Lieu of an Oath:**

I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis.

Saarbrücken, 15th August, 2022

### **Declaration of Consent:**

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Saarbrücken, 15th August, 2022

## Acknowledgements

I would like to thank everybody that supported me during the course of my studies and especially during the creation of this thesis even if they are not mentioned by name.

First, I want to sincerely thank my advisor, Alexander Prange, who helped me in the creation of the idea for this thesis and supported me in every stage following.

Moreover, I want to thank Prof. Dr. Sonntag for giving me the chance to write the thesis at the research department for Interactive Machine Learning at the German Research Center for Artificial Intelligence in Saarbrücken. Additionally, I want to thank Prof. Dr. Krüger for reviewing this thesis.

Special thanks goes to my family and friends, who accompanied and encouraged me during the development of this thesis. I want to especially thank the proofreaders amongst them.



## Abstract

Diabetic Retinopathy (DR) is a complication of diabetes, that leads to blindness if left untreated. Since DR is detectable on easily accessible imaging techniques, many Machine Learning models, especially Deep Learning (DL) models, have been developed in order to classify DR or segment eye scans. Most of them, do not make use of advanced pre-processing, although literature suggests performance improvements using pre-processing. Furthermore, pre-processing might improve generalisability, which can be especially beneficial for human-in-the-loop models. In this work, I evaluated the impact of the following pre-processing methods on three DL models: Contrast Limited Adaptive Histogram Equalization (CLAHE), Non-local Means Denoising (NLMeans) and Real-ESRGAN. Additionally, I evaluated a combination of CLAHE with NLMeans and Real-ESRGAN, respectively. I evaluated three specific DL models of which two were classification models and one was a segmentation model. The classification models were a general architecture, ResNet-50, and an architecture specifically designed for DR classification, OpticNet-71. The segmentation model was U-Net++. The results of my work show, that CLAHE improved the performance of all three DL models in comparison to the training on raw data. It especially improved sensitivity for under represented classes, which is very important for a medical setting, where data of unhealthy patients will be rare. For the more general architecture ResNet-50, all other pre-processing methods improved the performance significantly. However, for U-Net++ and OpticNet-71 those techniques performed similar to the raw data.

It is left to show how CLAHE helps generalisability of these models, although its performance boost already makes it a desirable pre-processing step in the classification and segmentation of OCT data.

---

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.1.1	Machine Learning in Medical Diagnosis . . . . .	1
1.1.2	Diabetic Retinopathy . . . . .	1
1.2	Goal . . . . .	2
<b>2</b>	<b>Related Work</b>	<b>6</b>
2.1	Problems in OCT Classification . . . . .	6
2.2	Pre-processing medical images . . . . .	7
2.2.1	Contrast Limited Adaptive Histogram Equalization . . . . .	7
2.2.2	Non-local Means Denoising . . . . .	8
2.2.3	Generative Adversarial Networks . . . . .	9
<b>3</b>	<b>Implementation</b>	<b>10</b>
3.1	Tech-Stack . . . . .	10
3.2	Pre-Processing . . . . .	11
3.2.1	Contrast Limited Adaptive Histogram Equalization . . . . .	12
3.2.2	Non-local Means Denoising . . . . .	13
3.2.3	Real-ESRGAN . . . . .	16
3.2.4	Combining Denoising and Contrast Enhancement . . . . .	19
3.3	Deep Learning Models . . . . .	19
3.3.1	ResNet-50 . . . . .	23
3.3.2	OpticNet-71 . . . . .	23
3.3.3	UNet++ . . . . .	26
3.4	Human-In-The-Loop Model . . . . .	27
<b>4</b>	<b>Evaluation</b>	<b>29</b>
4.1	Datasets . . . . .	29
4.1.1	Kermany2018 . . . . .	29
4.1.2	AROI . . . . .	30
4.2	Training . . . . .	31
4.3	Evaluation . . . . .	33

4.3.1	Classification Metrics . . . . .	33
4.3.2	Segmentation Metrics . . . . .	34
<b>5</b>	<b>Results</b>	<b>37</b>
5.1	ResNet-50 . . . . .	37
5.2	OpticNet-71 . . . . .	38
5.3	U-Net++ . . . . .	39
5.4	Applying U-Net++ to the Kermany2018 dataset . . . . .	41
<b>6</b>	<b>Discussion</b>	<b>44</b>
<b>7</b>	<b>Conclusion</b>	<b>46</b>
7.1	Overview . . . . .	46
7.2	Outlook . . . . .	46
	<b>Bibliography</b>	<b>48</b>

# List of Figures

1.1	Symptoms of NPDR [2]. . . . .	3
1.2	Symptoms of PDR [2]. . . . .	4
3.1	Illustration of the redistribution of pixel intensities that exceed the clip limit in CLAHE [3]. . . . .	13
3.2	OCT image (top left), HE processed image (middle left) and CLAHE processed image (bottom left) and their corresponding intensity histograms (right). One can clearly see how HE leads to extreme brightness in the tissue area of the OCT. . . . .	14
3.3	CLAHE with different clip limits applied to OCTs from Kermany2018 (upper half) and AROI (lower half). . . . .	15
3.4	Original OCT image (top) and NLMeans filtered image with $h = 8$ (middle) and $h = 16$ (bottom). One can clearly see how a stronger filter removes more noise. . . . .	17
3.5	NLMeans with different filter strength $h$ applied to OCTs from Kermany2018 (upper half) and AROI (lower half). . . . .	18
3.6	Real-ESRGAN with downsizing afterwards applied to OCTs from Kermany2018 (upper half) and AROI (lower half). . . . .	20
3.7	Combination of NLMeans and CLAHE applied to OCTs from Kermany2018 (upper half) and AROI (lower half). . . . .	21
3.8	Combination of Real-ESRGAN and CLAHE applied to OCTs from Kermany2018 (upper half) and AROI (lower half). . . . .	22
3.9	Overview of the ResNet-50 architecture. a) Stem block b) Block 1 in each stage, also called convolution block. This block changes the dimensions of its input. c) Block 2 in each stage, also called identity block. This block does not change its input's dimensions. d) Fully Connected (FC) Layer. Image taken from Wang et al., 2021 [52]. . . . .	24
3.10	Overview of the OpticNet-71 architecture by Shariff et al. [25]. a) shows the proposed residual unit with the parallel atrous and atrous separable convolution layers. b) shows the proposed building block, which replaces the original identity layers and contains two branches, one for convolution via the residual units and one for signal exhaustion. Part c) shows the overall architecture of the network, which corresponds to the ResNet architecture. . . . .	25
3.11	High level overview of the UNet++ architecture. UNet++ is an encoder decoder network which is connected through a series nested skip connections. These skip connections are dense convolutional blocks (shown in green and blue). Moreover, these connections only appear in UNet++ but not in UNet. Red indicates Deep Supervision. . . . .	26
3.12	Human-in-the-loop model for classifying/ segmenting OCT images. . . .	28

4.1	Example of an OCT from the AROI dataset [30]. Shown are an OCT with annotations for retinal layers (left) and retinal fluids (mid) and the resulting mask for this OCT (right). . . . .	30
4.2	Overview of the training and evaluation of the different architectures and pre-processing pipelines. . . . .	32
4.3	Visual representation of the dice coefficient. . . . .	35
4.4	Visual representation of the IoU score. . . . .	36
5.1	Performance metrics for the best ResNet-50 models trained on differently preprocessed Kermany2018 data. . . . .	38
5.2	Performance metrics for the best OpticNet-71 models trained on differently preprocessed Kermany2018 data. . . . .	40
5.3	Average Dice Scores of the best U-Net++ models trained on differently pre-processed datasets. . . . .	41
5.4	Average Intersection-over-Union Scores of the best U-Net++ models trained on differently pre-processed datasets. . . . .	42
5.5	Pre-processed images (left) and predicted masks of their correspondent UNet++ model (right). The original image was taken at random from the Kermany2018 dataset. . . . .	43

---

## List of Tables

1.1	Levels of DR severity according to ICDRSS [13]. IRMA = intra-retinal microvascular abnormalities . . . . .	5
2.1	Results of evaluating DenseNet trained on raw and CLAHE pre-processed lung x-rays [35] . . . . .	8
2.2	Results of evaluating a ResNet ensemble trained on raw and CLAHE pre-processed CT scans [44] . . . . .	8
3.1	Specifications of the machine used for training. . . . .	11
4.1	Distribution of labels in the Kermany2018 dataset. . . . .	29
4.2	Distribution of AROI labels. . . . .	30
4.3	Training parameters of all models. k-CV = k-fold Cross-Validation . . . . .	31
4.4	Overview of the evaluated pre-processing methods, what they are supposed to improve and their parameters. . . . .	31
5.1	Class dependent accuracy of the best ResNet-50 models trained on differently preprocessed Kermany2018 data. C = CLAHE, NLM = NLMeans, RG = Real-ESRGAN . . . . .	39
5.2	Class dependent F1-Scores of the best ResNet-50 models trained on differently preprocessed Kermany2018 data. C = CLAHE, NLM = NLMeans, RG = Real-ESRGAN . . . . .	39
5.3	Class dependent accuracy of the best OpticNet-71 models trained on differently preprocessed Kermany2018 data. C = CLAHE, NLM = NLMeans, RG = Real-ESRGAN . . . . .	39
5.4	Class dependent F1-Scores of the best OpticNet-71 models trained on differently preprocessed Kermany2018 data. C = CLAHE, NLM = NLMeans, RG = Real-ESRGAN . . . . .	40

---

# Chapter 1

## Introduction

### 1.1 Motivation

#### 1.1.1 Machine Learning in Medical Diagnosis

There has been a trend, where Machine Learning (ML) experts are building automatic end-to-end systems for the diagnosis of diseases from medical scans. With the ongoing Covid-19 pandemic the need for these ML systems has gotten more attention. However, ML in the medical field often suffers from noisy data, which might be due to poor equipment or bad data acquisition by medical staff. Moreover, medical staff does not necessarily need perfect data for their diagnosis, because they can still make sense of even noisy data. However, ML systems are not inherently capable of that and, therefore, pre-processing is needed to remove noise as much as possible. Pre-processing is the preparation, modification or selection of data before training a model or analyzing the data. In general, pre-processing can enhance the quality or performance of models that are being trained on the pre-processed data [22, 6, 50]. Specifically in medical imaging and biological computing, literature suggests that pre-processing can improve the performance of Machine Learning models [8, 31, 15]. This shows that pre-processing is very important for automatic diagnosis of diseases using ML systems.

One of the most common diseases nowadays is diabetes. As of 2019, 463 million people are suffering from diabetes, which makes up almost 9% of the world population [1]. Diabetes affects patients in many ways, one of which is Diabetic Retinopathy (DR). High blood sugar levels lead to damages of vessels in the retina of patients. These damages can include fibrosis or neovascularization, which unavoidably lead to blindness if left untreated.

#### 1.1.2 Diabetic Retinopathy

DR imposes a great risk to the vision of many patients. If it is left untreated, DR will evolve into an irreversible stage which will lead to blindness. However, DR in its early

stage is treatable and early detection of DR is crucial in order to save a patients vision. This early stage of DR is called non-proliferative DR (NPDR). It is the first of two stages and is caused by diabetic microvascular abnormalities leading to focal ischemia (Impaired blood flow in the retina) [2]. NPDR has several symptoms including Microaneurysms (MA), Dot blot hemorrhages (DBH), Hard Exudates (HE), Cotton Wool Spots (CWS) and Venous Beading (VB). Figure 1.1 shows the mentioned symptoms on fundus images and a description on how they evolve. Although these symptoms can impact vision negatively, they do not lead to blindness and can be treated.

However, in the second stage of DR, pro-liferative DR (PDR), which evolves from NPDR, if the ischemia is severe and longstanding enough, patients can lose their vision. This is due to neovascularization, the growth of new blood vessels. Neovascularization damages and occludes the retina and, hence, leads to irreversible blindness [2]. Figure 1.2 shows the symptoms of PDR, which include Neovascularization of the disc, iris or elsewhere, and preretinal and vitreous hemorrhage as well as fibrosis. Additionally, diabetic macular edema (DME) can occur at any point during DR [2]. DME describes the build up of intraretinal fluids in the macula, the center of the eye, and leads to impaired vision [4]. However, just as NPDR symptoms DME can be treated.

As we have seen, DR comes with many symptoms and to assess the severity of DR one can use the International Clinical Diabetic Retinopathy Severity Scale (ICDRSS). The ICDRSS is defined as a five stage severity scale, whereas the first stage is "no disease", the following three stages are phases of NPDR ranging from mild to severe and the last stage is PDR [13]. One can see these stages and their corresponding symptoms in Table 1.1. This scale is important for the classification of retinal scans and, hence, a desirable label for datasets in machine learning. Unfortunately, it only applies to fundus images (a type of retinal scan that will be explained in later sections) and, hence, is not fit for all datasets. However, a classification into NPDR and PDR is always possible given the symptoms mentioned before.

In general, DR is not curable [2] but can be treated to avoid PDR and enable normal vision for patients. According to Tapp et al. [48], at least 90% of new cases of severe DR could be reduced with early diagnosis and early treatment. Therefore, many automatic diagnosis systems have been developed to identify and classify DR. However, classification is often not enough, as ophthalmologists need to know the location of symptoms such as microvascular abnormalities. Therefore, there is also a need for segmentation models. Furthermore, segmentation models allow for more interpretable results in classification, as doctors can see the segmented layers and intraretinal fluids.

## 1.2 Goal

As mentioned before, pre-processing is expected to have a major impact on DL models especially in a medical setting. However, DL models in DR classification from OCT scans often do not use major pre-processing. Therefore, the goal of this work is to evaluate the impact of some of the most commonly used pre-processing methods that proved useful in related fields. The models trained in this thesis can then be used for an human-in-the-loop model.



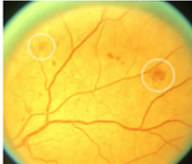
	<p><b>Microaneurysms (MAs)</b></p> <p>These are the first sign of diabetic retinopathy. Weakening of capillary walls results in small vessel aneurysms with distinct margins (no larger than largest retinal vessel). They may be hard to find clinically but they really light up on FA.</p>
	<p><b>Dot blot hemorrhages (DBHs)</b></p> <p>When MA's rupture, they can create medium sized round hemorrhages that are dot, blot, or flame shaped. These are larger than MAs with indistinct margins. These are blocking on FA (appear as dark blotches). A clear sign of moderate DR.</p>
	<p><b>Hard exudates (HEs)</b></p> <p>Active or resolved macular edema (intraretinal fluid) can leave behind distinct white/yellow cholesterol deposits, as absorption of the edema (fluid) occurs faster than absorption of the high molecular weight substances like cholesterol.</p>
	<p><b>Cotton Wool Spots (CWS)</b></p> <p>Also known as soft exudates, these are white and feathery, and are infarcts within the retinal nerve fiber layer.</p>
	<p><b>Venous beading (VB) – Left circle on image</b></p> <p>This is a late stage finding in nonproliferative diabetic retinopathy and represents weakened walls of major retinal vessels. This is one of the strongest predictors for progression to proliferative diabetic retinopathy (PDR).</p> <p><b>Intraretinal microvascular abnormality (IRMA) – Right circle on image</b></p> <p>These are small abnormally shaped blood vessels that shunt blood from arterioles to venules. They characteristically <b>do not</b> leak on FA, unlike neovascularization elsewhere (NVE).</p>

Figure 1.1: Symptoms of NPDR [2].

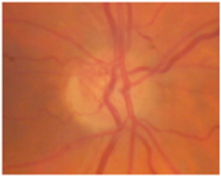

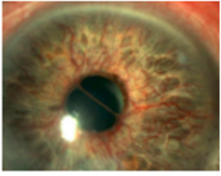
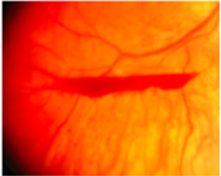
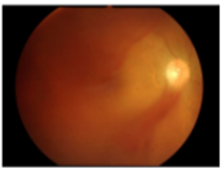
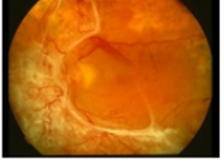
	1. Neovascularization of the disc (NVD): Refers to neovascularization at the optic disc
	2. Neovascularization elsewhere (NVE): Refers to neovascularization in other areas of the retina, including the macula or in the periphery
	3. Neovascularization of the iris (NVI): Refers to neovascularization occurring in the iris
	4. Preretinal hemorrhage: Blood trapped in a potential space between the posterior hyaloid and the internal limiting membrane, appears as a "boat-shaped hemorrhage".
	5. Vitreous hemorrhage: Appears as a red haze of hemorrhage directly into the vitreous. You may see a faint glow of the white optic nerve. Vitreous hemorrhage ranges from mild inferior vitreous blood to dense blood with no view. Blood becomes dehemoglobinized over time and will eventually turn white.
	6. Fibrosis over the retina happens most commonly over the optic nerve and the vascular arcades.

Figure 1.2: Symptoms of PDR [2].

Severity Level	Observable Findings
No apparent DR	No abnormalities
Mild NPDR	Only Microaneurysms
Moderate NPDR	More than mild but less than severe NPDR
Severe NPDR	Any of the following: <ul style="list-style-type: none"><li>- More than 20 intraretinal hemorrhages in 4 quadrants</li><li>- Definite venous beading in 2+ quadrants</li><li>- Prominent IRMA in 1+ quadrant</li></ul> And no sign of PDR
PDR	One or more of the following: <ul style="list-style-type: none"><li>- Neovascularization</li><li>- Vitreous/preretinal hemorrhage</li></ul>

Table 1.1: Levels of DR severity according to ICDRSS [13]. IRMA = intra-retinal microvascular abnormalities

---

## Chapter 2

### Related Work

#### 2.1 Problems in OCT Classification

Optical Coherence Tomography (OCT) is a cross-section image of the retinal layers acquired using the reflection properties of tissues on near-infrared light [10]. These images are prone to noise which is caused by secondary reflections or other light sources. This type of noise is called speckle noise and is present in almost all medical ultrasound or coherence tomography images [45]. Speckle noise often results in coarse edges, which are disadvantageous for any edge detection or segmentation algorithm. This shows that there is a need for pre-processing methods that get rid of this type of noise.

Moreover, there is another problem present in the classification, detection and segmentation of OCT images. In 2018, the Deep Mind Team around author De Fauw published a paper in nature about a two-step-classification network, in which OCTs will first be segmented and, second, be classified by two different network [17]. This approach reportedly outperformed four retina specialists and four optometrists and, hence, shows how powerful DL can be in this field. The authors claim that by using two separate steps, they can also leverage generalisability problems by only re-training the segmentation network on new data. However, in their work they also demonstrate, how vulnerable DL architectures are towards data inconsistencies by testing their architecture on a new data set acquired from a new device type. Their original segmentation network had a 46.6% total error rate on the new data set, which is unusable for a real world application, especially in the field of medical diagnosis. Only by re-training the segmentation network they could achieve similar performance as before. This problem is called domain shift problem. The domain shift problem describes the setting, where a network performs well on one domain, in this case OCTs from one device, but performs poorly on a different but related domain, e.g. OCTs from a different device. Nguyen et al. have found the same problem, when classifying DR severity from fundus images [33]. They used a self-supervised learning approach in order to learn domain adaptation to a new device. Both of these approaches follow the domain adaptation approach. However, one could also follow a domain generalization approach. Domain generalization is the setting, where a model is trained on several, different but related domains and then tested on a

new unseen domain [51]. In this case, we can see this as training on OCTs from multiple devices and then being able to generalize to a new device. Domain generalization can be achieved by three different ways: Data manipulation, representation learning and a proper learning strategy. The latter two focus on the way the networks learn by for example using ensemble learning to generalize over multiple predictions. In fact, the self-supervised learning network by Nguyen et al. also follows the learning strategy approach for domain generalization. However, the data manipulation approach is only concerned about the data, which is used for training and, hence, is model independent. Zhou et al. claim that by manipulating the data before training through image transformations (pre-processing) one can achieve a better generalizing model [55]. However, this approach is widely unused in DL. At most, data augmentation methods are being used to artificially increase the amount of training data. Therefore, I propose that using pre-processing techniques in order to get more representative data with more apparent image features and less noise will help in developing more performant and more robust DL models.

## 2.2 Pre-processing medical images

### 2.2.1 Contrast Limited Adaptive Histogram Equalization

Contrast Limited Adaptive Histogram Equalization (CLAHE) is a pre-processing method, which enhances the contrast of an image by redistributing the intensities of the image to cover the whole range of intensity [41]. CLAHE is an advanced version of Histogram Equalization (HE), which redistributes the intensity of pixels globally. However, CLAHE looks at local patches of the image and redistributes their intensities locally. Additionally, CLAHE clips any intensities that surpass a certain threshold and redistributes them to lower intensities.

CLAHE in combination with median filtering has proven to be effective in boosting the performance of models using fundus images [16, 42, 38, 28, 37], but to the best of my knowledge has not been evaluated on OCT images yet. However, there has been a study proving CLAHE to be one of the best contrast enhancement methods for OCTs [44].

A recent study of the research group around Paniagua et al. from 2021 [35], examined the influence that CLAHE has on DL models for detecting Covid-19 from Lung X-Rays. Their Pre-Processing pipeline consists of resizing the images for better computation times, and then applying CLAHE or leaving the images as they are. They then trained a DenseNet-121 [21] on both datasets and compared their performances. Additionally, they trained a new architecture consisting of two separate models on raw input. The first model being a U-Net [39], which was supposed to identify Regions of Interest (ROIs) for the classification. The second model was also a DenseNet-121, which classified based on the ROIs three different classes: Healthy, unhealthy with bacterial pneumonia and unhealthy with viral pneumonia.

They evaluated their model on many different performance metrics and always emphasized on the importance of generalizability and robustness of their models. The DenseNet trained on CLAHE images converged after 7 epochs, while their own architecture with ROI segmentation trained on raw input converged after 9 epochs, speaking for a computational performance improvement. A DenseNet trained on raw input only, without segmentation converged much earlier, hence bringing doubts about the ROI-Segmentation used. The differences between DenseNet trained on raw versus CLAHE input are marginal as one can see in Table 2.1. Although the models perform exactly the

Metric	Raw Input	CLAHE
Sensitivity	0.83	0.84
Specificity	0.91	0.90
Accuracy	0.88	0.88
F1 Score	0.87	0.87

Table 2.1: Results of evaluating DenseNet trained on raw and CLAHE pre-processed lung x-rays [35]

Metric	Raw Input	CLAHE
Precision	0.86	0.91
Recall	0.73	0.81
Accuracy	0.73	0.87
F1 Score	0.67	0.84

Table 2.2: Results of evaluating a ResNet ensemble trained on raw and CLAHE pre-processed CT scans [44]

same for most metrics, the sensitivity in the CLAHE model is better, which is a desirable result in medical diagnosis, as false negatives are more of a danger towards patients than false positives. Although the results of this work are not promising, they show one interesting aspect, which is that using CLAHE might not improve performance but make the models better for discovering a potential disease.

A different study from Sanagavarapu et al. from 2021 [43] also researched the impact of CLAHE on DL models. However, they used CT scans instead of X Rays and also used a ResNet-20 ensemble [20] for training. They trained their models for a binary classification between "healthy" and "unhealthy". Looking at Table 2.2 we can see that the ResNet ensemble performs significantly better on CLAHE pre-processed data for every metric. Additionally, they examined traditional machine learning methods such as Random Forests on the impact of CLAHE and also found that CLAHE improves their accuracy. Therefore, CLAHE seems to be a promising contrast enhancement method for medical imaging and might improve DL models in OCT training as well.

### 2.2.2 Non-local Means Denoising

Non-local Means Denoising (NLMeans) is a method that removes noise from an image by replacing each pixels color (or intensity) with the average color (or intensity) of the most similar pixels. NLMeans does not use locality to compute those averages, hence the name "non-local" [9]. According to Yang et al., NLMeans filtering helps in removing speckle noise found in OCTs because of its properties of removing noise while preserving edges [54].

As a noise reduction method NLMeans has already found use in classification of DR as shown by Alban and Gilligan [7], which used NLMeans as a pre-processing step for several different DL architectures.

In a recent work from 2019, Firdausy et al. [18] examined the impact different filter strengths of NLMeans have on a machine learning model. More specifically, they applied NLMeans with filter strengths 2 to 18 applied in steps of 2 to 45 fundus images and evaluated their impact on a 2-dimensional Gaussian surface fitting approach for vessel

extraction in terms of the determination coefficient  $R^2$ . Their results show that a filter strength of 4 to 16 yield the best results as  $R^2$  reaches values of about 0.98. The best filter strength was 12 as it yields a  $R^2$  of 0.9862.

Nazaré et al. investigated the effect of different noise types, such as salt and pepper noise or gaussian noise, applied to images and using NLMeans to restore those images [32]. Moreover, they trained CNNs on both types of images and their results suggest that introducing noise at training time improved the generalizability of those networks. However, contrary to the results of Firdausy et al. restoring the images using NLMeans lead to a decreased performance of the CNNs. The authors claim that this might be caused by choosing the wrong parameters for NLMeans and, hence, over-smoothing the images.

NLMeans seems to be a promising denoising technique for OCTs as it removes speckle noise. However, the results of Nazaré et al. show that one needs to be careful in choosing the right parameter for the filter strength in order to not remove important image features.

### 2.2.3 Generative Adversarial Networks

Recently, there has been a trend of using generative networks to produce synthetic new data. This trend originated with the introduction of generative adversarial networks (GANs) by Goodfellow et al. in 2014 [19]. By using the adversarial play between a generator model and a discriminator model, Goodfellow et al. were able to produce realistic new synthetic data. Nowadays, many more types of GANs have been developed with such realistic results, that a whole new field of cybersecurity has arisen, which deals with the detection of so called deep fakes [14]. One purpose of GANs is the restoration and super-resolution of images. In this field, Real-ESRGAN (Enhanced Super Resolution GAN) has been developed by Wang et al. [53]. This network removes noise of all types, restores and refines edges and upsamples images to a higher resolution. Therefore, it might be useful for the pre-processing of OCT images. However, it is usually not applied to medical images and, hence, must be evaluated for this specific purpose.

---

# Chapter 3

## Implementation

### 3.1 Tech-Stack

The implementation of the whole project was done in Python, as it is the standard in Data Science and Machine Learning.

In this work, many libraries or modules have been used but the following three were able to provide functionality for the majority of the implementation.

- PyTorch, Version 1.10.2
- TensorFlow, GPU Version 2.8.0
- OpenCV, Version 4.5.3.56

PyTorch and TensorFlow are the standard libraries when it comes to DL and have also been used for such purposes. OpenCV provides a wide range of image manipulation algorithms and hence has been used for the majority of the pre-processing methods.

The following DL architectures and pre-processing method come from publicly accessible GitHub repositories:

- Real-ESRGAN
- OpticNet-71
- U-Net++

The training was done on a remote machine located at DFKI Saarbrücken with the specifications shown in Table 3.1.



PC Part	Specification
CPU	Intel Core i9-10900K 10x3.7GHz
RAM	128GB DDR4-2666
GPU	NVIDIA GeForce RTX 3090 24GB
Memory	2TB SSD + 10TB HDD
OS	Windows 10 + Linux

Table 3.1: Specifications of the machine used for training.

## 3.2 Pre-Processing

In order to apply many different pre-processing methods, I decided to create a base class called "Method". The code for this class looks as the following:

```

1 class Method:
2     """Base class for Pre-Processing methods."""
3     def __init__(self, name, title):
4         self.name = name
5         self.title = title
6         pass
7
8     def process(self, img):
9         """
10        Applies the class intern pre-processing to an image
11        :param img: array or ndarray. Usually only works with grayscale images.
12        :return: pre-processed image.
13        """
14        return img
15
16    def get_description(self):
17        """ Returns a textual description of the method as a string.
18        Usually contains name and parameters of the method. """
19        return self.title

```

Listing 3.1: Implementation of Base Pre-Processing class

As one can see in the code above, every instance of "Method" is initialized with a name, which is a string containing a abbreviation for the method and is used for creating file paths belonging to this class, and also a title, which contains a short human readable description of the method. Each class inheriting from Method has to implement "process", which represents the main functionality of each class. "process" takes an image array of shape width times height, meaning it only takes one color channel or grayscale images as input. The class also implements a method called "get\_description", which is used to get a textual description of the method including parameter values. For example, one can see the class for no pre-processing in the following:

```

1 class Raw(Method):
2     def __init__(self):
3         super(Raw, self).__init__("raw", "Raw Image")
4
5     def process(self, img):
6         return img
7
8     def get_description(self):
9         return "No preprocessing"

```

Listing 3.2: Implementation of No Pre-Processing class

For some training it was necessary to implement additional functionality as the network for example required resized images of a different size than what the implementations of "Method" work with. Therefore I decided to introduce a "Helper" class, which inherits from "Method" but takes an object of type "Method" as initialization argument. The following code shows the "Helper" class for training OpticNet-71:

```

1 class OpticNetHelper(Method):
2     def __init__(self, method: Method):
3         super(OpticNetHelper, self).__init__(method.name, method.title)
4         self.method = method
5
6     def process(self, img):
7         new_img = img.astype(np.uint8)
8         new_img = self.method.process(new_img)
9         if len(new_img.shape) <= 2:
10             new_img = np.expand_dims(new_img, 2)
11         return new_img

```

Listing 3.3: Implementation of the OpticNet-71 Helper class

In its "process" function there is some additional functionality which resizes the image from shape (256, 256) to shape (256, 256, 1), because the implementation of OpticNet-71 expects this shape for its inputs.

### 3.2.1 Contrast Limited Adaptive Histogram Equalization

Contrast Limited Adaptive Histogram Equalization (CLAHE) is a method which enhances contrast in an image. It is a more advanced version of Histogram Equalization (HE). HE redistributes all pixel intensities such that they cover the whole range of intensities [5]. This improves contrast in an image, when all pixel intensities are clustered together. To apply HE, we have to define a probability function for a discrete grayscale image  $X$ :

$$p_X(i) = \frac{\text{number of pixels with intensity } i}{\text{total number of pixels}} \quad (3.1)$$

We can now define the cumulative distribution function:

$$\text{cdf}_X(i) = \sum_{j=0}^i p_X(j) \quad (3.2)$$

We now have to define a transformation of the pixel intensities, such that  $\text{cdf}_X$  grows linearly. For any cdf such a function exists because of the cdf's properties and can be easily computed. More specifically, if  $L$  denotes the maximum intensity, then by taking the inverse of formula 3.3, we can get the desired transformation [5].

$$\text{cdf}_Y(i) = (i + 1)K \text{ for } 0 \leq i < L \quad (3.3)$$

In Figure 3.2 one can see how HE changes an OCT. While the intensities are redistributed in a way to cover the whole range of intensities, the retinal layers are too bright. This is because these layers were already bright before HE was applied and afterwards will be pushed even more to the maximum intensity. However, CLAHE looks at local patches of the image and applies HE locally only [41]. This avoids that already bright parts of the image get too bright, hence local contrast will be more prominent. Additionally, CLAHE clips and redistributes intensity counts that surpass a certain threshold, the clip limit, as it is illustrated in Figure 3.1 [41]. This avoids amplifying noise and leads to a more

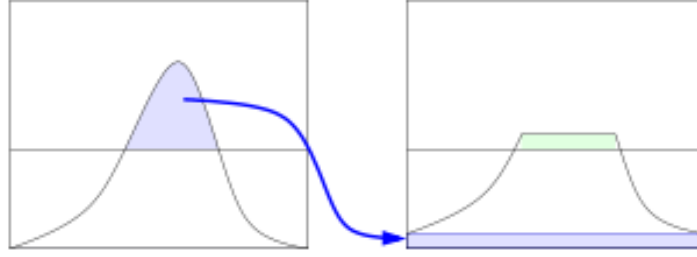


Figure 3.1: Illustration of the redistribution of pixel intensities that exceed the clip limit in CLAHE [3].

evenly distributed histogram, as one can observe in Figure 3.2. Hence, CLAHE takes two parameters, the clip limit and the size of the local patches or also called tile grid size. Figure 3.3 shows the impact of changing the clip limit parameter. We have seen that Sanagavarapu et al. used CLAHE for Lung CTs and achieved significant improvements, hence i will be using the parameters they used, more specifically I will use a clip limit of 5 [43] and the standard tile grid size of  $8 \times 8$ .

For the implementation of CLAHE, I used the OpenCV framework. More specifically, the class for CLAHE is shown in Listing 3.4.

```

1 class CLAHE(Method) :
2     ...
3     def process(self, img):
4         clahe_obj = cv.createCLAHE(self.cliplimit, self.tilegridsize)
5         return clahe_obj.apply(img)
6     ...
7 
```

Listing 3.4: Implementation of CLAHE class

### 3.2.2 Non-local Means Denoising

Non-local Means Denoising (NLMeans) is another denoising method. However, NLMeans does not look at the neighbourhood of a pixel in order to compute its new value. Instead NLMeans replaces the target pixel's intensity with an average of all other pixel weighted by the similarity between their neighbourhood and the target pixel's neighbourhood [12]. More specifically, if  $\Omega$  is the area of an image and we have two points  $p$  and  $q$  in the image, then according to Buades [9] formula 3.4 describes the filtered value at point  $p$  for a discrete image.

$$u(p) = \frac{1}{C(p)} \sum_{q \in \Omega} v(q) f(p, q) \quad (3.4)$$

In this formula  $C(p)$  is a normalizing factor given by formula 3.5, while  $v(q)$  is the original value and  $f(p, q)$  is the new value.

$$C(p) = \sum_{q \in \Omega} f(p, q) \quad (3.5)$$

For discrete images, formula 3.6, the gaussian weighting function, describes a possible function for  $f$ , where  $h$  describes the filter strenght and  $B(p)$  is the local mean of pixels

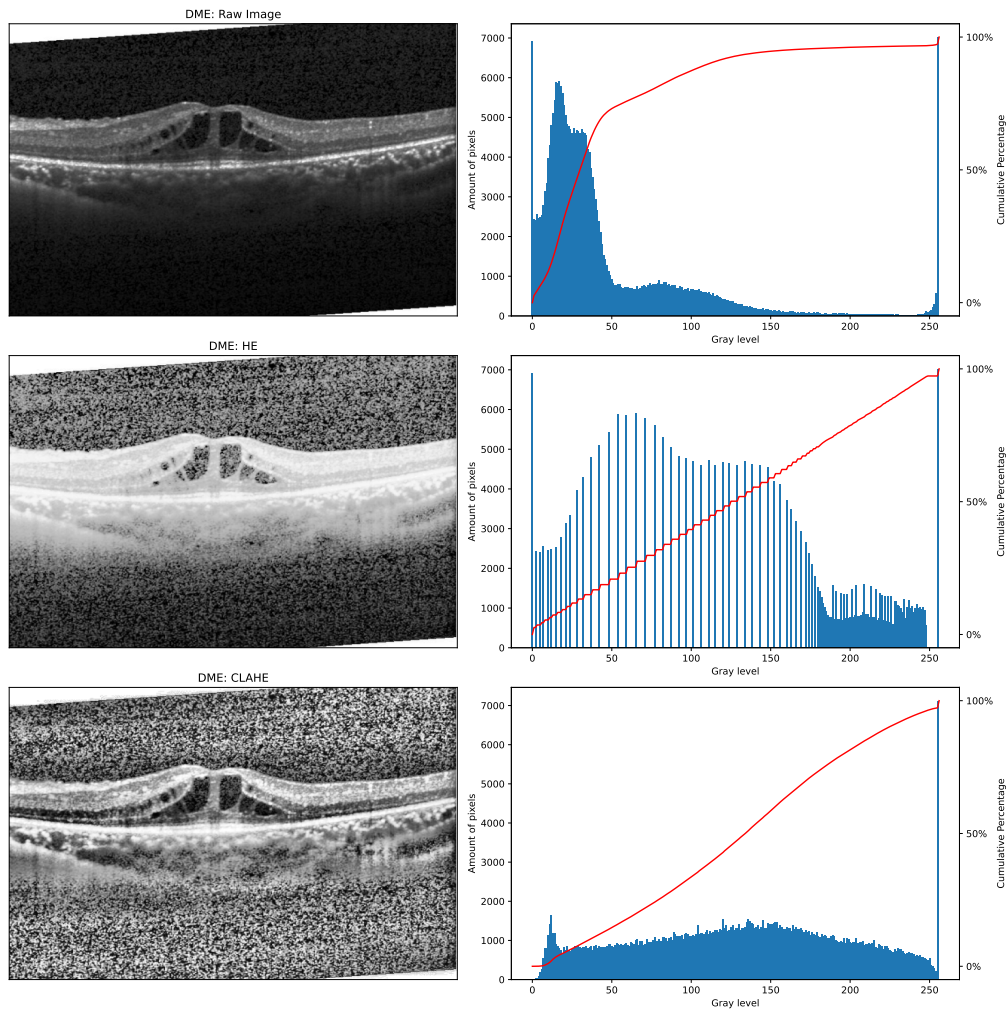


Figure 3.2: OCT image (top left), HE processed image (middle left) and CLAHE processed image (bottom left) and their corresponding intensity histograms (right). One can clearly see how HE leads to extreme brightness in the tissue area of the OCT.

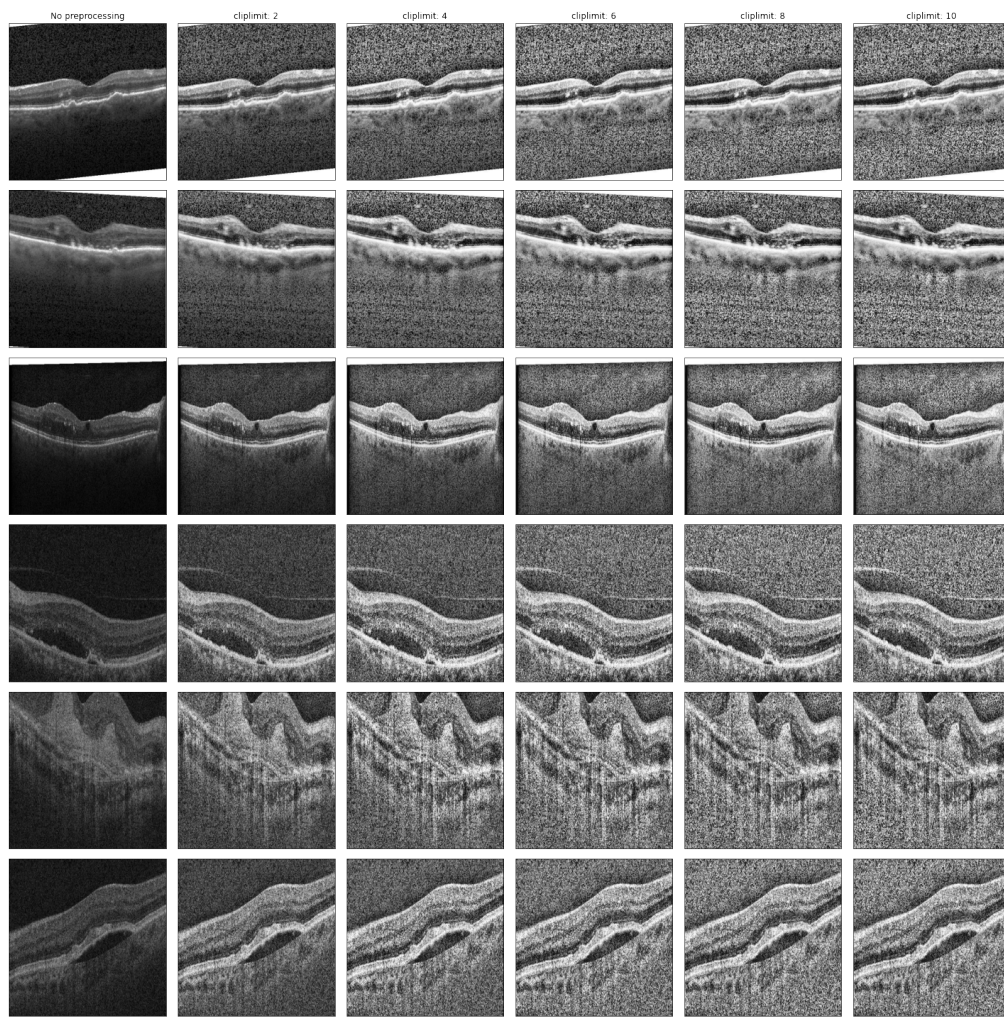


Figure 3.3: CLAHE with different clip limits applied to OCTs from Kermany2018 (upper half) and AROI (lower half).

surrounding  $p$ .

$$f(p, q) = e^{-\frac{|B(q) - B(p)|^2}{h^2}} \quad (3.6)$$

Therefore NLMeans has two parameters, filter strength  $h$  and the size of the neighbourhood for  $B$ . NLMeans is supposed to remove noise while preserving edges because of its non-locality [12]. In Figure 3.4 we can see NLMeans applied to an OCT with different filter strengths.

For the implementation I used OpenCV again. The OpenCV implementation differentiates between the size of the neighbourhood for the target pixel (template window size) and the size of the neighbourhood that will be compared (search window size), hence leaving us with three parameters. The class for NLMeans is shown in Listing 3.5. In Figure 3.4 one can observe the impact the parameter  $h$  has on OCTs. The higher the values for  $h$  the stronger the filter and the more blurred the image gets. However, edges will be preserved in the image and mainly information inside the tissue will be lost. Figure 3.5 shows different filter strengths applied to a multitude of images from both datasets.

```

1 class NLMEANS(Method) :
2     ...
3     def process(self, img):
4         return cv.fastNlMeansDenoising(img,
5                                         self.h,
6                                         self.templatewindowsize,
7                                         self.searchwindowsize)
8     ...
9 
```

Listing 3.5: Implementation of NLMeans class

### 3.2.3 Real-ESRGAN

Real-ESRGAN is a Generative Adversarial Network (GAN) based on the Enhanced Super Resolution GAN (ESRGAN) architecture, which is trained with pure synthetic data [53]. This architecture is used to restore images and upscale them to a higher resolution. It uses a U-Net architecture as a discriminator trained with spectral normalization which increases discriminator capability and training stability [53]. Real-ESRGAN removes noise and additionally provides sharper edges while upscaling images. Therefore this method seems interesting as a Pre-Processing method for DR classification since Fundus and especially OCT images are often noisy. Figure 3.6 shows that Real-ESRGAN in fact seems to remove noise quite well. Additionally the image looks clearer as it is four times higher resolution. However Real-ESRGAN upsamples parts of the image and then stitches them back together. It can appear that one can see the edges of upsampled parts. Additionally, I added methods that downsize the image before or after the upsampling. This was needed as otherwise the segmentation masks would not fit anymore. Furthermore, OCT and Fundus images often already come in high resolution and upsampling them leads to ridiculously large images which are not fit for DL. Hence, downsampling before can increase performance. Downsampling happens via bilinear interpolation, which was implemented using OpenCV's `resize` function. Since Real-ESRGAN already does noise reduction and image upsampling, it might be fit to be a complete pre-processing pipeline on its own and will therefore be evaluated as one. The code for Real-ESRGAN comes from a publicly available repository owned by the inventors of Real-ESRGAN. The Real-ESRGAN class is shown in Listing 3.6.

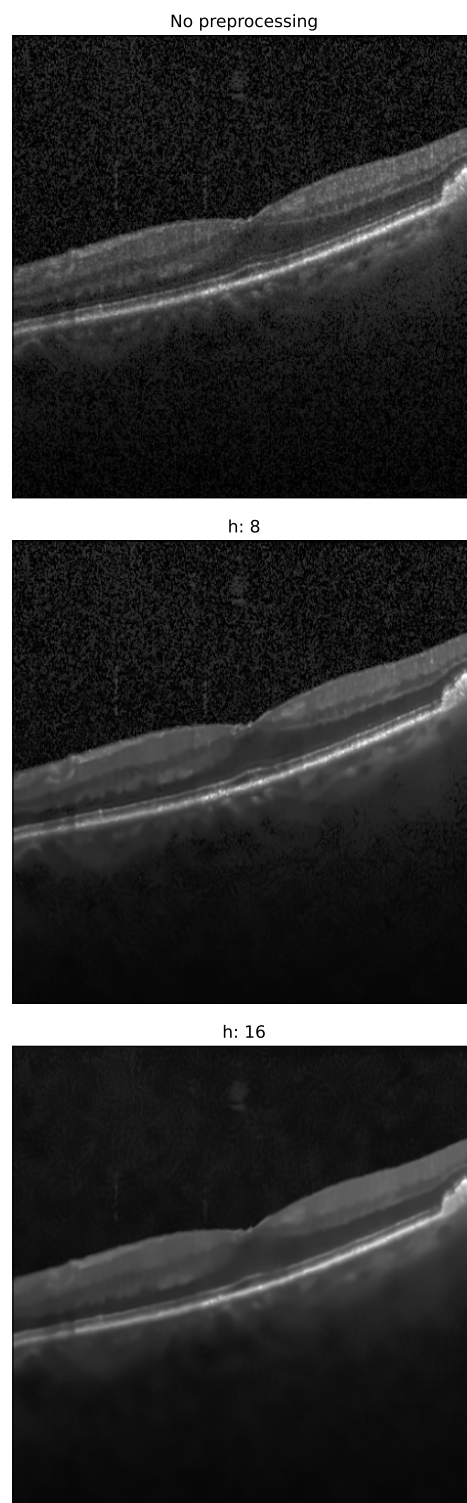


Figure 3.4: Original OCT image (top) and NLMeans filtered image with  $h = 8$  (middle) and  $h = 16$  (bottom). One can clearly see how a stronger filter removes more noise.



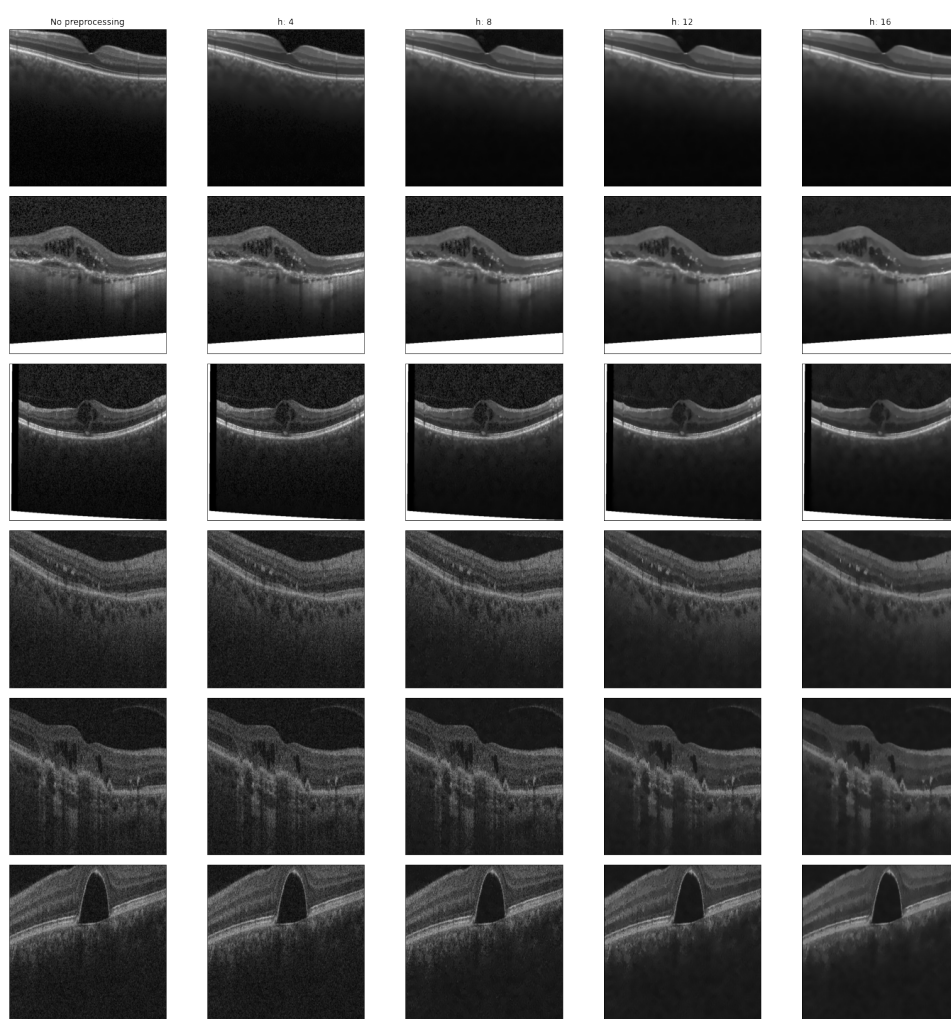


Figure 3.5: NLMeans with different filter strength  $h$  applied to OCTs from Kermany2018 (upper half) and AROI (lower half).



```

1 class RealESRGAN(Method):
2
3     def __init__(self, downsize_after=False, downsize_before=False, ...):
4         ...
5         model = RRDBNet(...)
6         ...
7         self.upsampler = RealESRGANer(...)
8         self.downsize_after = downsize_after
9         self.downsize_before = downsize_before
10
11     def process(self, img):
12         if self.downsize_after:
13             shape = (img.shape[1], img.shape[0])
14             if self.downsize_before:
15                 img = cv.resize(img, (...))
16             img, _ = self.upsampler.enhance(img, outscale=4)
17             return cv.resize(img, shape)
18         else:
19             if self.downsize_before:
20                 img = cv.resize(img, (...))
21             img, _ = self.upsampler.enhance(img, outscale=4)
22             return img
23         ...
24

```

Listing 3.6: Implementation of Real-ESRGAN class

### 3.2.4 Combining Denoising and Contrast Enhancement

All methods shown so far, can only either enhance contrast or denoise the image. However, both are desirable. Therefore, a combination of denoising first and contrast enhancement second might be interesting to look at as well. This is why, additionally to the already mentioned methods, I am also going to evaluate a combination of NLMeans and CLAHE as shown in Figure 3.7, as well as a combination of Real-ESRGAN and CLAHE as shown in Figure 3.8. However, the figures show that the combination does not necessarily look better. For the NLMeans + CLAHE combination we can see that we get a pattern in the background and especially for the already noisy OCTs of the AROI dataset, we get seemingly noisier images after processing. Real-ESRGAN has a similar effect. It produces blur spots in the background of the images which remind of condensation spots on glass surfaces. However, it does not lead to the same noise amplification in the AROI dataset as NLMeans + CLAHE seems to do. Although these methods produces some background noise, the important part of the image, the retinal layers are more clear and more contrastful.

## 3.3 Deep Learning Models

Many DL networks have been developed for image classification. In this work, I will look at one general DL model and one DL model specifically developed for DR classification on OCTs.

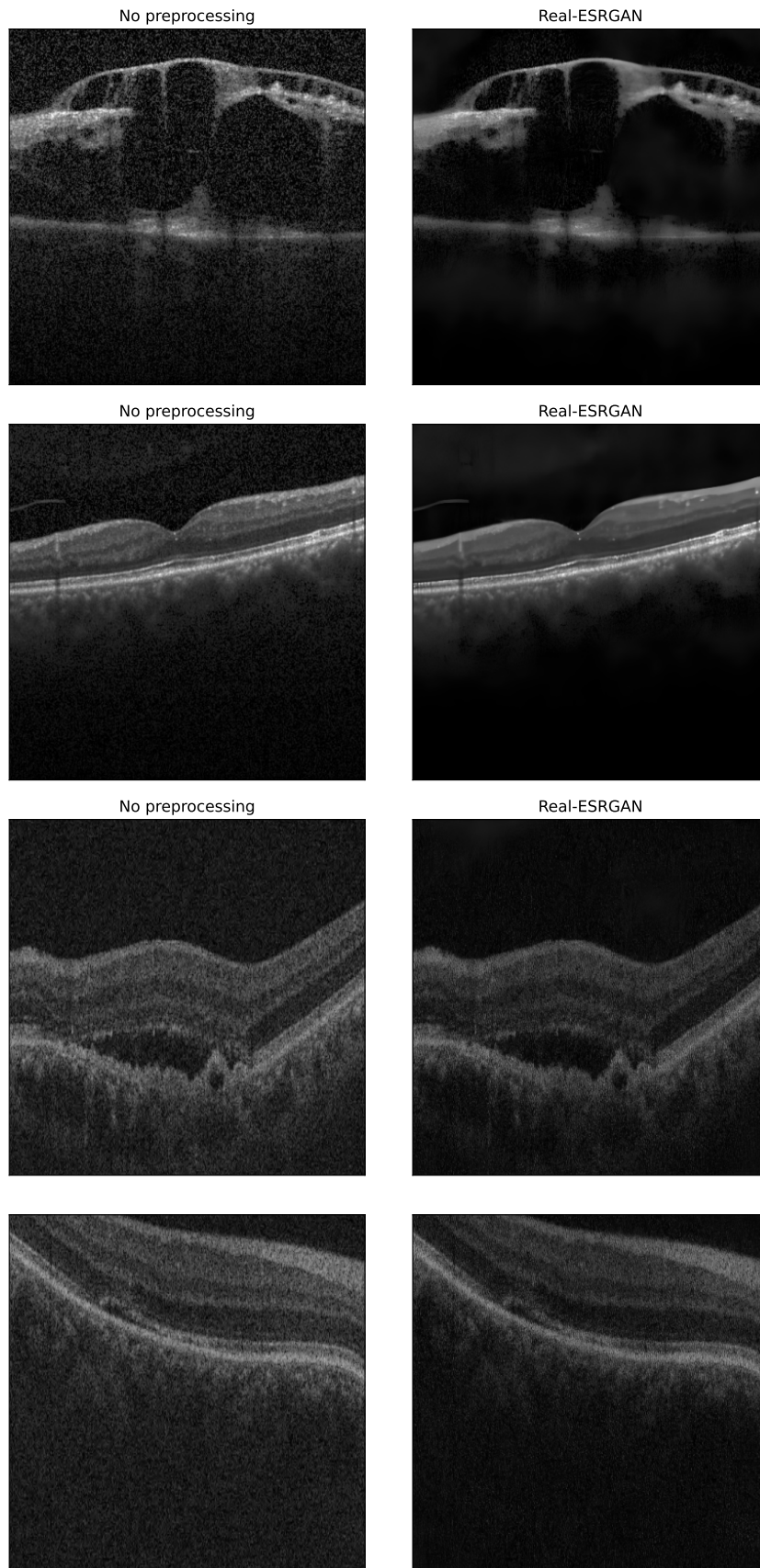


Figure 3.6: Real-ESRGAN with downsizing afterwards applied to OCTs from Ker-  
many2018 (upper half) and AROI (lower half).

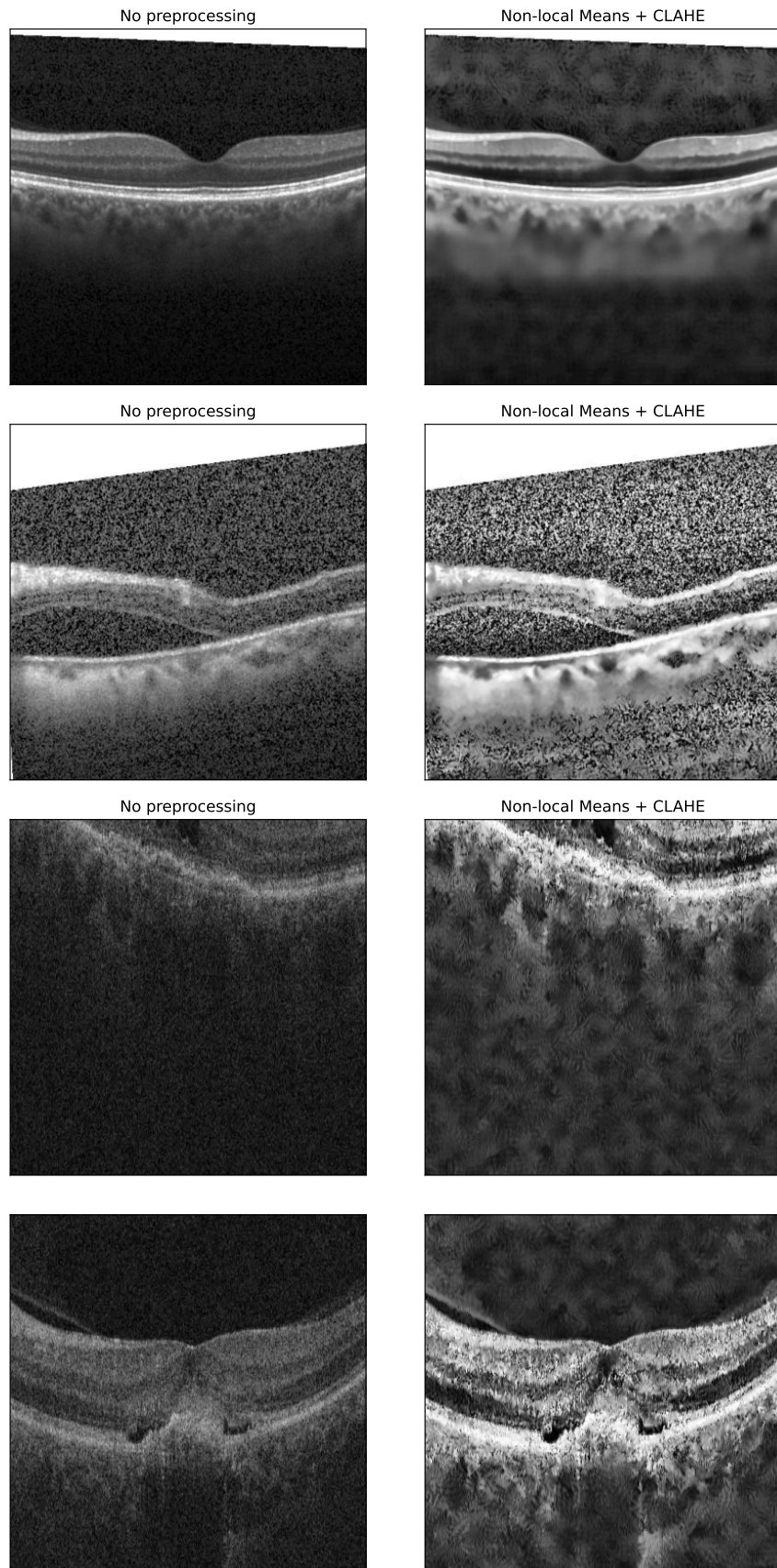


Figure 3.7: Combination of NLMeans and CLAHE applied to OCTs from Kermany2018 (upper half) and AROI (lower half).

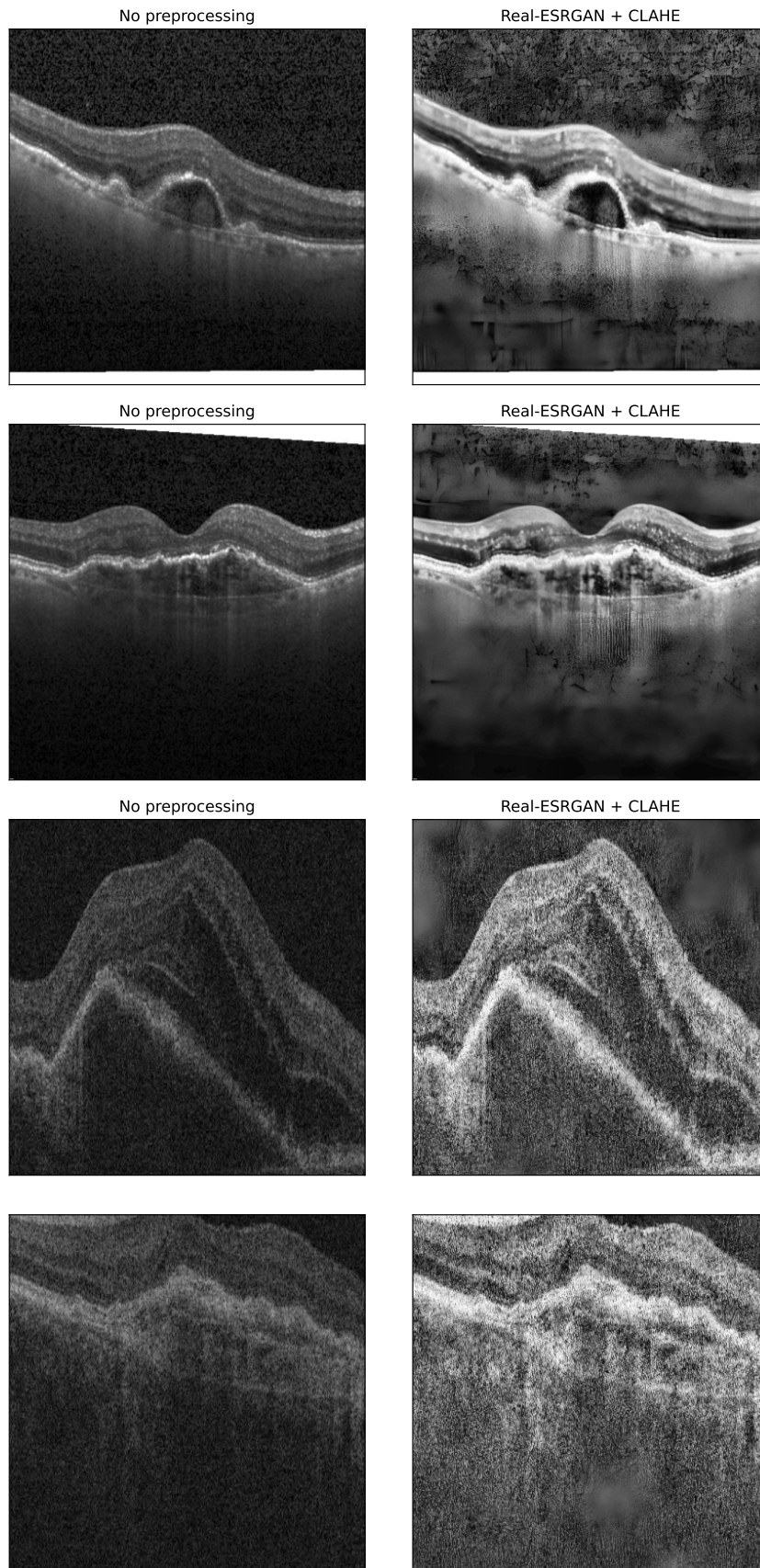


Figure 3.8: Combination of Real-ESRGAN and CLAHE applied to OCTs from Ker- many2018 (upper half) and AROI (lower half).

### 3.3.1 ResNet-50

The introduction of Residual Networks (ResNets) in 2015 revolutionized the way neural networks were built. ResNets introduced residual units or skip connections which allowed the gradients to flow to shallower parts of the network and, hence, allowed for deeper networks without heavy performance sacrifices [20]. The architecture of this network is widely known, which is why I am not going to explain it in more detail here. Figure 3.9 shows an overview of the ResNet-50 architecture and its building blocks. We can see the residual skip connections in b) and c) of this figure. It is obvious to see how these connections help to make gradient computations easier by allowing the gradients to pass by a lot of convolutional layers. The number 50 in the name ResNet-50 refers to the amount of layers in this architecture.

### 3.3.2 OpticNet-71

OpticNet-71 has specifically been designed for classifying DR from OCT images and does so extremely efficiently reaching near perfect accuracy of 99.8% for the Kermany2018 dataset [25, 26]. The architecture builds on a Deep Residual Convolutional Neural Network (ResNets) and uses specific building blocks and residual units. Figure 3.10 part a) shows these residual units and their structure. One can easily see the similarity to the ResNet, where we have a skip connection from the input to just before the output of the residual unit.

However, instead of having three convolutional layers, the middle layer has been replaced by an atrous and an atrous separable convolutional layer, which run in parallel and take only half of the channels of the input signal each. Atrous or also called dilated convolutions space out the pixels each convolution filter is looking at, such that the looked at pixels are not neighbours of each other [36]. Separable convolutions separate the convolution kernel into two smaller kernels of one dimension. This allows to learn spatial features and depth-wise features separately and also decreases the computational effort [24]. The authors claim that by using atrous and atrous separable convolutions, the network produces more robust features, because it learns the spatial and depth-wise feature maps entangled and separated [25].

These residual units are used inside a new proposed building block as seen in part b) of Figure 3.10. The building blocks replace the identity blocks of the original ResNet [25]. One can see, that in those building blocks the input is passed into two branches. The left branch contains a linear stack of the proposed residual units, while the right branch contains a so called signal exhaustion followed by a sigmoid activation. The signal exhaustion downsamples the input through a Max-pooling layer and then upsamples it again using Bi-linear interpolation. This process is supposed to intrinsically exhaust the signal and serves as another residual skip unit. We can also observe that both branches will be multiplied before they are added. This is supposed to balance out the increase in signal propagation before the addition [25]. The basic architecture shown in Figure 3.10 c) follows the ResNet architecture.

As already mentioned, this model achieved near perfect accuracies for OCT datasets without any major pre-processing. However, it is unclear, if the model generalizes well for new data or if it could be even better with pre-processing. Moreover, testing pre-processing on this model shows, whether or not it is necessary to actually pre-process data, or, if having a highly adapted network architecture is sufficient.

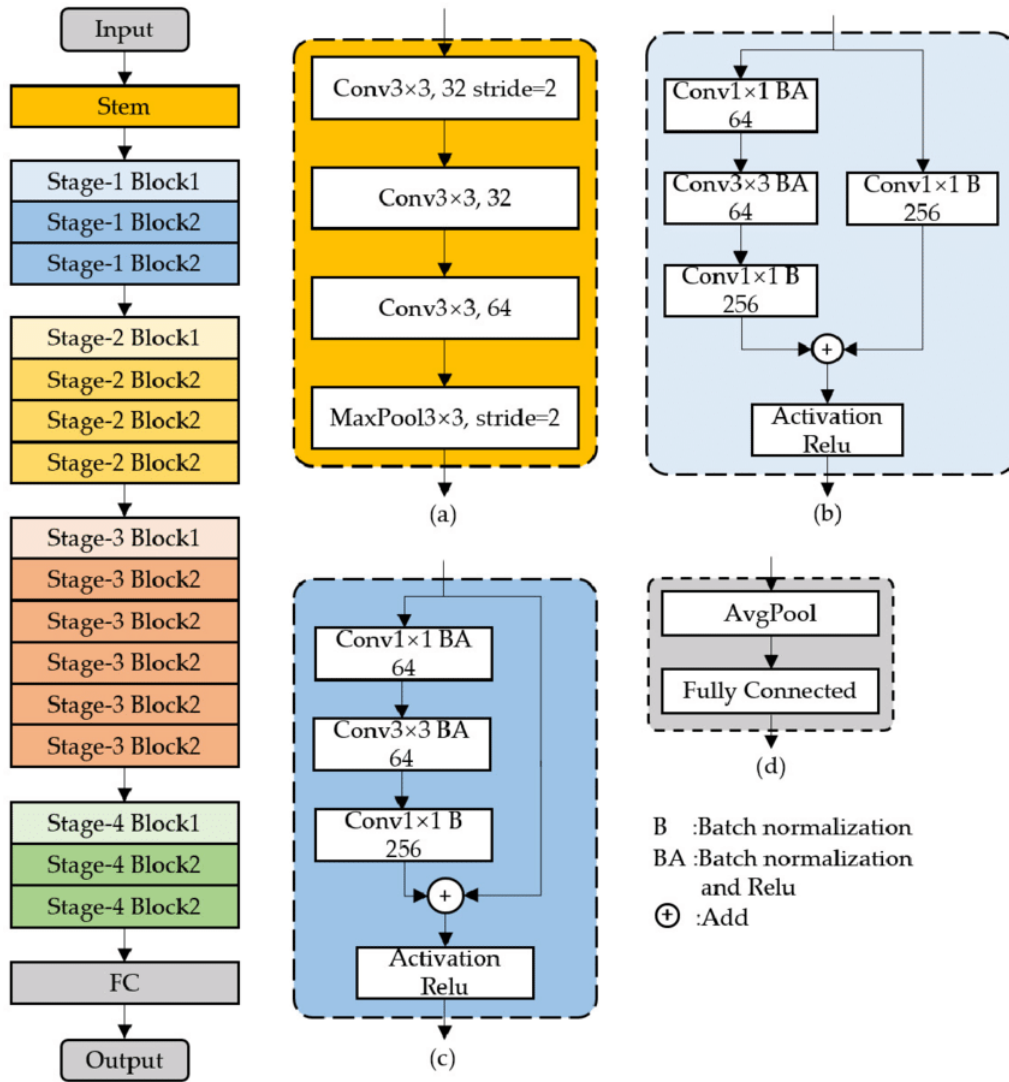


Figure 3.9: Overview of the ResNet-50 architecture. a) Stem block b) Block 1 in each stage, also called convolution block. This block changes the dimensions of its input. c) Block 2 in each stage, also called identity block. This block does not change its input's dimensions. d) Fully Connected (FC) Layer. Image taken from Wang et al., 2021 [52].

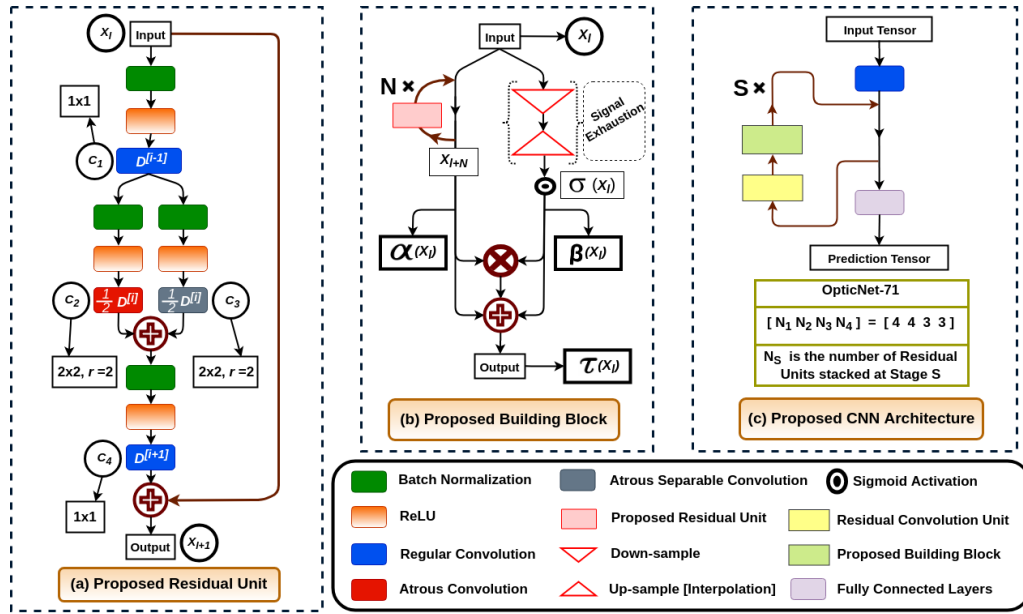


Figure 3.10: Overview of the OpticNet-71 architecture by Shariff et al. [25]. a) shows the proposed residual unit with the parallel atrous and atrous separable convolution layers. b) shows the proposed building block, which replaces the original identity layers and contains two branches, one for convolution via the residual units and one for signal exhaustion. Part c) shows the overall architecture of the network, which corresponds to the ResNet architecture.



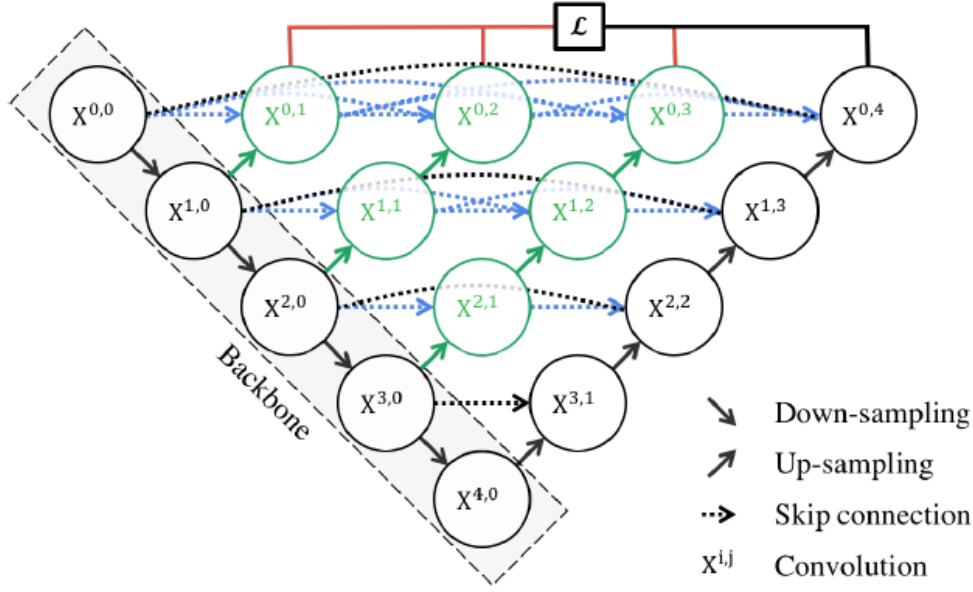


Figure 3.11: High level overview of the UNet++ architecture. UNet++ is an encoder decoder network which is connected through a series nested skip connections. These skip connections are dense convolutional blocks (shown in green and blue). Moreover, these connections only appear in UNet++ but not in UNet. Red indicates Deep Supervision.

### 3.3.3 UNet++

The UNet architecture has been introduced by Ronneberger et al. in 2015 and ever since has been a very popular choice for training semantic segmentation and medical segmentation models. In 2018 Zhou et al. introduced UNet++, which is an advanced version of UNet. Therefore, it inherits the basic UNet architecture, which consists of a contracting, encoding path of convolutions and an expanding, decoding path of up-convolutions, which are connected via skip connections. This basic architecture gives it its "U" shape and, hence, its name[40]. Figure 3.11 shows an high level overview of UNet++. In black is shown the basic UNet architecture. One can see that UNet++ does not just use skip connections from the contracting path (or "backbone" in the figure) to the expanding path, but it also introduces subnetworks inbetween both paths. Therefore, UNet++ is a deeply supervised encoder-decoder network where encoder and decoder sub-networks are connected via nested, dense skip pathways [56]. These skip pathways are supposed to close the gap between the semantic feature maps of decoder and encoder. The authors claim that similar feature maps make the learning process easier for the optimizer of the network [56]. The network is also capable of using deep supervision. Deep supervision refers to the model using "companion" losses at each hidden layer. These losses will later be used to compute the final loss of the model and combat issues like robustness and the vanishing gradients problem [29]. However, deep supervision was not used in this thesis. Figure 3.11 gives a high level overview of U-Net++ and also showcases the differences to the standard U-Net. UNet++ has been evaluated on several medical image segmentations but to the best of my knowledge not yet on OCT images. There exists an off-the-shelf implementation of UNet++, which can be adapted to OCT data [58].



### 3.4 Human-In-The-Loop Model

Pre-processing can also help in a different way as well. In our research group, Ophthalmology-AI, at the research department for Interactive Machine Learning at DFKI, we are currently working on an annotation tool for OCTs. In general, annotation is very time intensive and, hence, not many datasets will have large amounts of data with fine grained annotations. In our annotation tool we want to include ML models, which help human experts annotate data. This creates a loop of ML algorithm and human expert whereas both profit off of each other. The algorithm profits off of more data, while the human profits off of a smaller workload as well as a more accurate algorithm. It becomes obvious, that a well generalising model will be of great value in such an annotation tool as it saves the human expert time. Therefore, improving generalisability by pre-processing data does not only improve the performance of the model but also allows for finer annotation in the annotation tool. Figure 3.12 shows this human-in-the-loop model. We can see that new unlabeled data coming from hospitals or doctors gets pre-processed before being fed to the ML model. The ML model then predicts the annotation for this data, which will be sent to the annotation tool. The annotation can either be a label or a segmentation. In the annotation tool, the human experts see the new data as well as the given annotation and, hence, only have to accept or refine the annotation. The annotated data is then added to the training data of our model and the model can be retrained on the expanded dataset. This concept basically implements incremental deep learning, where the model is continuously trained as new data is being made available. Moreover, this approach can be seen as an Active Learning approach, where the model presents a prediction and asks for feedback from a user. This feedback is then incorporated into the model making the model actively learning from the user. Furthermore, this approach might help in mitigating bias towards automatic diagnosis systems in medicine as medical staff regularly validate the models performance. This process would help in establishing trust into these automatic diagnosis systems.

Moreover, pre-processing could help in making DL models easier to train on less data by removing image artifacts and noise while making meaningful image features more apparent. In this case, pre-processing can be used to enable One-Shot-Learning models, which are trained on very few data. This would be desirable, as one could efficiently train one model per image acquisition apparatus (eg. Optical Coherence Tomograph) and, hence, eliminate data variations that are due to different acquisition methods. These models might not generalize well over datasets from lots of different sources. However, they could be deployed for usage in hospitals and be trained from scratch there. Therefore, this model will then be better adapted to the conditions and might also be more trusted as it has been freshly trained by local staff.

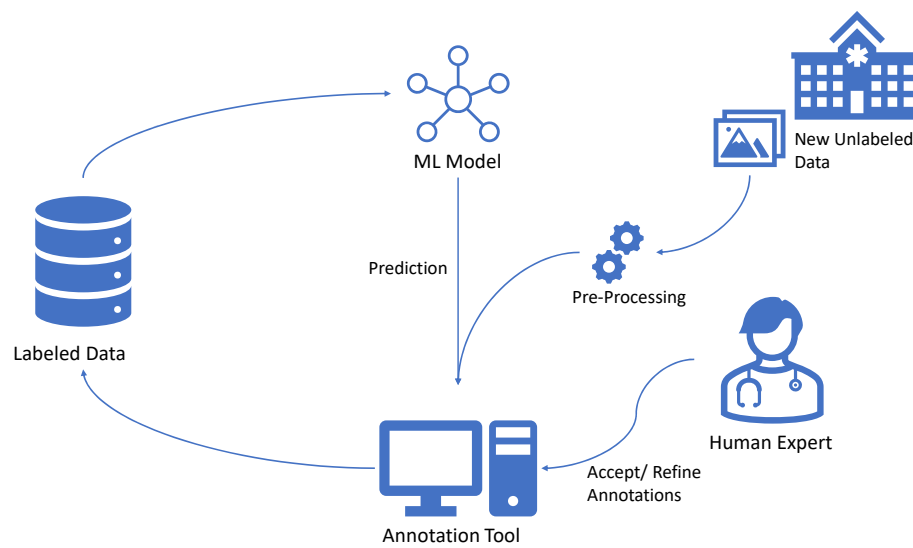


Figure 3.12: Human-in-the-loop model for classifying/ segmenting OCT images.

Label	Counts
NORMAL	51390
DRUSEN	8866
CNV	37455
DME	11598

Table 4.1: Distribution of labels in the Kermany2018 dataset.

---

## Chapter 4

### Evaluation

#### 4.1 Datasets

In order to train the aforementioned models, one needs data. There exists a variety of image datasets of eye scans from patients with different severity of Diabetic Retinopathy. Those datasets fall into one of two categories. They either contain Fundus images or Optical Coherence Tomography (OCT) Scans, whereas their name refers to how the images are acquired. Fundus images are images of the rear of the eye, which is called fundus hence the name. These images are taken using fundus cameras which are low-power microscopes [46]. For OCT images however one uses light waves to take cross-section pictures of the retina. This allows to see the distinct retinal layers and possibly enclosed fluids or symptoms of NPDR or PDR as mentioned before [34].

There exists a variety of image datasets of OCT scans such as Srinivasan2014 [47], Kermany2018 [26], AROI [30] and many more. In contrast to fundus image datasets, most of these are publicly available.

##### 4.1.1 Kermany2018

This dataset contains 109309 OCT images. The dataset is labelled by normal, drusen, cnv and dme and their distribution can be seen in Table 4.1. Since the dataset only contains labels for symptoms of DR, it can only be used for classification. The dataset is also not balanced.

Layer	Label	Counts
Background	/	1136
Internal Limiting Membrane	ILM	1136
Area between Inner Plexiform and Inner Nuclear Layer	IPL/INL	1136
Retinal Pigment Epithelium	RPE	1136
Bruch Membrane	BM	1136
Intraretinal Fluid	IRF	228
Subretinal Fluid	SRF	648
Pigment Epithelial Detachment	PED	1014

Table 4.2: Distribution of AROI labels.

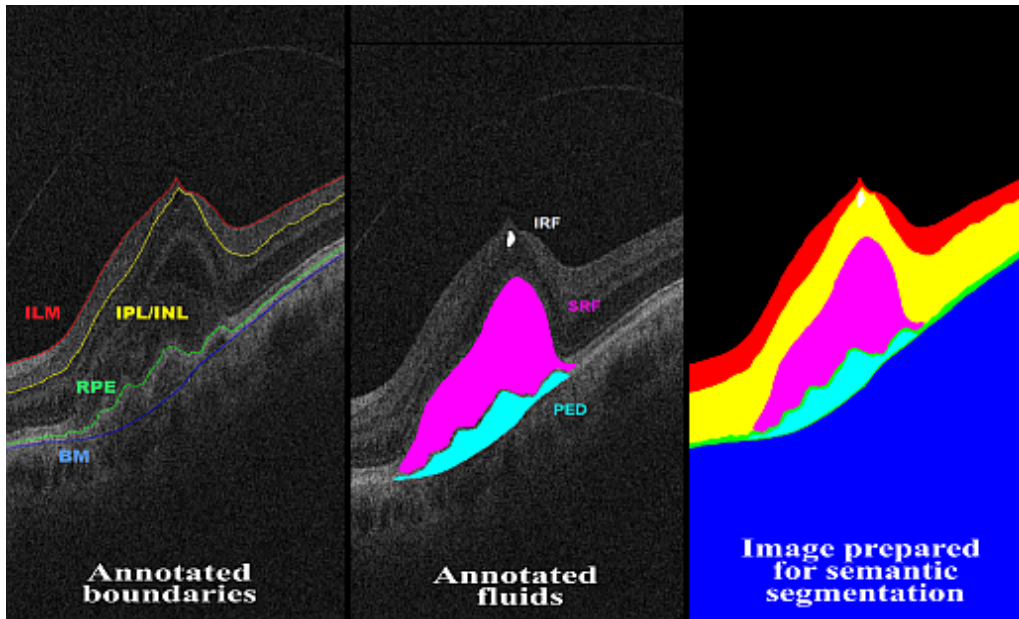


Figure 4.1: Example of an OCT from the AROI dataset [30]. Shown are an OCT with annotations for retinal layers (left) and retinal fluids (mid) and the resulting mask for this OCT (right).

#### 4.1.2 AROI

This dataset contains in total 1136 OCT images of 24 different patients. Each image has a corresponding mask which assigns each pixel a retinal layer or disease abnormality such as drusen. The dataset originally also investigates inter and intra observer error and contains data for that purpose [30]. Figure 4.1 shows an annotated OCT from this dataset.

The OCTs are segmented into up to 8 classes, of which only the first 5 appear on every OCT. The labels for these classes are shown in Table 4.2 alongside their number of appearances in the dataset. The last 3 classes do not appear on every OCT and therefore lead to an imbalance of labels.

Model	Loss	Optimizer	Learning rate	Number of epochs	k-CV
ResNet-50	CCE	Adam	0.001	15	None
OpticNet-71	CCE	Adam	0.0001	20	None
U-Net++	Lovász	Adam	0.0001	25	5

Table 4.3: Training parameters of all models. k-CV = k-fold Cross-Validation

Method	Expected improvement	Parameters
CLAHE	Contrast	Cliplimit = 5, Tile Grid Size = 8
NLMeans	Noise	h = 12
Real-ESRGAN	Noise	None
NLMeans + CLAHE	Contrast + Noise	h = 12 Cliplimit = 5, Tile Grid Size = 8
Real-ESRGAN + CLAHE	Contrast + Noise	Cliplimit = 5, Tile Grid Size = 8

Table 4.4: Overview of the evaluated pre-processing methods, what they are supposed to improve and their parameters.

## 4.2 Training

Figure 4.2 shows the basic pipeline used for training and evaluating different DL architectures on differently pre-processed datasets. First, the data is split into training and test data. This split is the same for each pre-processing and model to ensure comparability. Training data is then fed to one of the pre-processing pipelines. Table 4.4 shows all pre-processing methods as well as their expected impact on the image and their parameters, that were used for the experiments. Optionally these pipelines might be wrapped by a "Helper" class in order to ensure compatibility with the network, that will be used for training. The data is then pre-processed, which usually happens in real-time right before training to avoid using too much disk space. But since Real-ESRGAN is not computationally efficient, pipelines using this method will have their data pre-processed and saved to disk before training in order to speed up training times. After pre-processing is done, a network is trained using the parameters shown in Table 4.3. For ResNet-50 and OpticNet-71 no cross validation will be done for time reasons. Therefore, we get one model per pre-processing for these networks. However, for the UNet++ architecture i used 5-fold cross validation, which means i used five different data splits to train five different models. In the end, i computed an average of all five models in order to compare them. This gives five models per pre-processing pipeline for the UNet++ architecture. The loss used for OpticNet-71 and ResNet-50 was categorical cross entropy (CCE) and for U-Net++ it was Lovász Softmax loss [11]. The optimizer for all three was Adam [27].

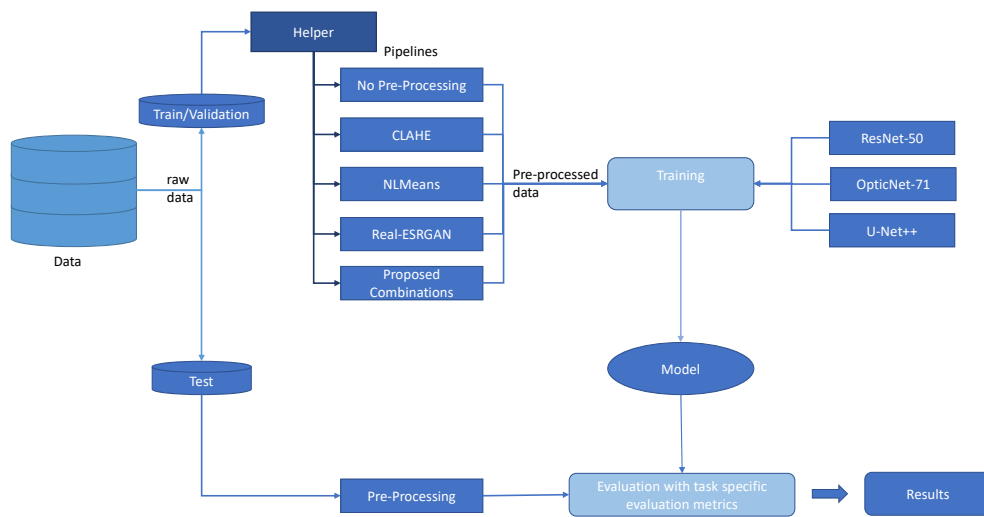


Figure 4.2: Overview of the training and evaluation of the different architectures and pre-processing pipelines.

## 4.3 Evaluation

After training is done, the best models will be evaluated using the test data and task-specific evaluation metrics. To find the best models, I look at the validation loss and take the model with the lowest validation loss. This ensures that the model is not overfitting on the training data. The metrics used for evaluation will be explained in the following. However, many other metrics might be interesting, such as generalizability, time or disk space metrics, but could not be evaluated in the scope of this thesis.

### 4.3.1 Classification Metrics

#### Accuracy

Classification accuracy is the standard in evaluating classification models. Accuracy is the amount of true positives (TP) and true negatives (TN) divided by the amount of all samples, so true and false positives (FP) and true and false negatives (FN). While high accuracy is always desirable, it does not always represent the best metric in the context of medical diagnosis. Medical datasets are often imbalanced in a sense that healthy samples are overrepresented. A model predicting "healthy" in the majority of cases will therefore always perform well in terms of accuracy. However, it does not perform well in finding unhealthy patients; hence, making it less useful for medical diagnosis. Formula 4.1 describes how accuracy is computed.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (4.1)$$

#### Sensitivity

Sensitivity refers to the probability of a positive prediction under the condition that the sample is truly positive. In the context of medical diagnosis, a high sensitivity is desirable as this means that more diseases are being detected among unhealthy patients and, therefore, can be treated. Formula 4.2 describes how sensitivity is computed.

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (4.2)$$

#### Specificity

Specificity refers to the probability of a negative prediction under the condition that the sample is truly negative. While sensitivity is key for medical diagnosis, specificity plays a less important role as disease predictions will still be checked by a professional. However, one wants both, sensitivity and specificity, to be near 1 for the best performance. Formula 4.3 describes how to compute specificity.

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \quad (4.3)$$

#### Precision

Precision describes the probability of a true positive prediction under the condition that the prediction is positive. In the context of medical diagnosis, precision plays an

important role as it describes how trustworthy a positive prediction is. For example, a model with high sensitivity and low precision will simply predict positive in the majority of cases and therefore catch almost all positive cases. However, its precision is low as it also misses a lot of true negatives. It is desired to be near 1, but it is not as important as high sensitivity. Formula 4.4 describes how to compute specificity.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (4.4)$$

### F1-Score

The F1-Score gives a direct measure between sensitivity and precision by taking the harmonic mean of both. It is therefore a direct measure of the performance of a model and of similar importance as sensitivity. Formula 4.5 describes how to compute the F1-Score.

$$\text{F1-Score} = \frac{2 \cdot \text{Sensitivity} \cdot \text{Precision}}{\text{Sensitivity} + \text{Precision}} \quad (4.5)$$

## 4.3.2 Segmentation Metrics

### Dice Coefficient

The Dice coefficient independently proposed by Thorvald Sørensen and Lee Raymond Dice was first introduced as a segmentation metric by Zijdenbos et al. in 1994 [57]. The Formula 4.6 shows the computation of the Dice score:

$$\text{Dice} = \frac{2 \times \text{Area of Overlap}}{\text{Predicted Area} + \text{True Area}} \quad (4.6)$$

Therefore the Dice coefficient can take up values from 0 to 1, whereas 1 resembles a perfect prediction and 0 a completely wrong prediction. Figure 4.3 shows a visual representation of the dice score.

The implementation of the Dice coefficient is shown in Listing 4.1. In order to avoid a "DivisionByZero" exception, a smoothing factor is added to the denominator and numerator.

```

1 def dice_coefficient(pred, true, smooth=1e-15):
2     intersection = 2 * (torch.sum((torch.logical_and(true, pred)))) .item()
3     union = torch.sum(true).item() + torch.sum(pred).item()
4     dice = (intersection + smooth) / (union + smooth)
5     return dice

```

Listing 4.1: Implementation of Dice coefficient

### Intersection over Union

The Intersection over Union score (IoU) or also called Jaccard Index was invented by Paul Jaccard [23]. It works similar to the Dice coefficient and in fact both scores are positively correlated [49]. Formula 4.7 shows how to compute the Intersection over Union score. Looking at the formula we can see that again the values for this score reach from 0 to 1 and similar to the Dice coefficient high scores yield a good segmentation while low



scores yield a bad segmentation. Figure 4.4 shows a visual representation of the IoU score.

$$\text{IoU} = \frac{\text{Intersection of Predicted and True Area}}{\text{Union of Predicted and True Area}} \quad (4.7)$$

The implementation of the IoU score is shown in Listing 4.2. Again a smoothening factor was used in order to avoid a "DivisionByZero" exception.

```
1 def intersection_over_union(pred, true, smooth=1e-15):
2     intersection = (torch.sum((torch.logical_and(true, pred)))).item()
3     union = (torch.sum((torch.logical_or(true, pred)))).item()
4     return (intersection + smooth) / (union + smooth)
```

Listing 4.2: Implementation of IoU

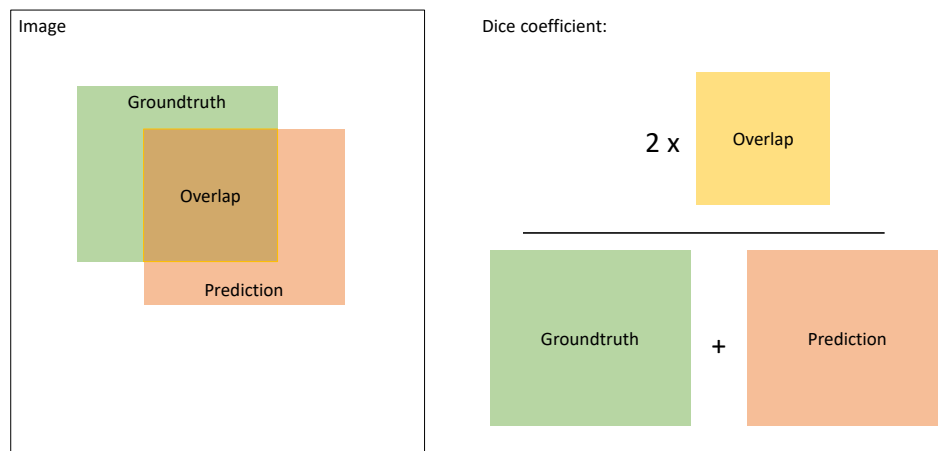


Figure 4.3: Visual representation of the dice coefficient.

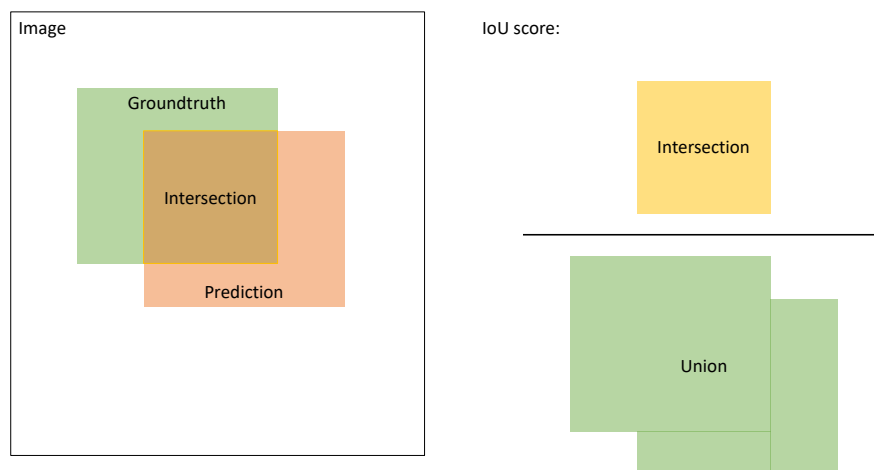


Figure 4.4: Visual representation of the IoU score.

---

# Chapter 5

## Results

### 5.1 ResNet-50

As shown in Figure 5.1, the ResNet-50 architecture performs worst on the not preprocessed, raw data with an average accuracy of 84%, while all other models except the one trained on Real-ESRGAN preprocessed data perform significantly better with an 97% average accuracy. Additionally, F1-Score increases for these models by at least 20% from 74% in raw data to 94% in the NLMeans and NLMeans + CLAHE pre-processed data and 95% in the CLAHE and Real-ESRGAN + CLAHE pre-processed data.

CLAHE and Real-ESRGAN + CLAHE lead to the best performing models considering the average metrics. However, Real-ESRGAN alone performs only slightly better than no preprocessing with 87% average accuracy and 77% average F1-Score. Looking at Table 5.1, we can see the class dependent accuracies of the models. The table shows that for all classes except DRUSEN no preprocessing performs worst, while CLAHE performs best. However, all pre-processing methods except Real-ESRGAN alone only differ slightly in accuracy for each class. For class CNV, all models have an accuracy of 97%, while Real-ESRGAN and no pre-processing only reach 85% and 73%, respectively.

Looking at the individual F1-Scores, shown in Table 5.2, we can see that the same holds for this metric. No pre-processing performs worst with 64% F1-Score. However, CLAHE, NLMeans, NLMeans + CLAHE and Real-ESRGAN + CLAHE do not perform completely similar, because Real-ESRGAN + CLAHE performs 1% better than the rest with 95% F1-Score. Real-ESRGAN alone achieves an F1-Score of 76%. For class DME, NLMeans + CLAHE achieves 97% accuracy and, hence, performs 1% better than CLAHE, NLMeans and Real-ESRGAN + CLAHE regarding this class, while Real-ESRGAN and no pre-processing only achieve 83% and 90% accuracy. Similarly, NLMeans + CLAHE achieves 1% better F1-Score of 93% than CLAHE, NLMeans and Real-ESRGAN + CLAHE. No pre-processing and Real-ESRGAN achieve F1-Scores of 50% and 78% respectively. For class DRUSEN, CLAHE, NLMeans + CLAHE and Real-ESRGAN + CLAHE reached an accuracy of 98%, while NLMeans performed 1% worse. No pre-processing and Real-ESRGAN, again, performed worse than the remaining models with 85% and 84% accuracy respectively. This also shows in the F1-Scores for this class. However, CLAHE

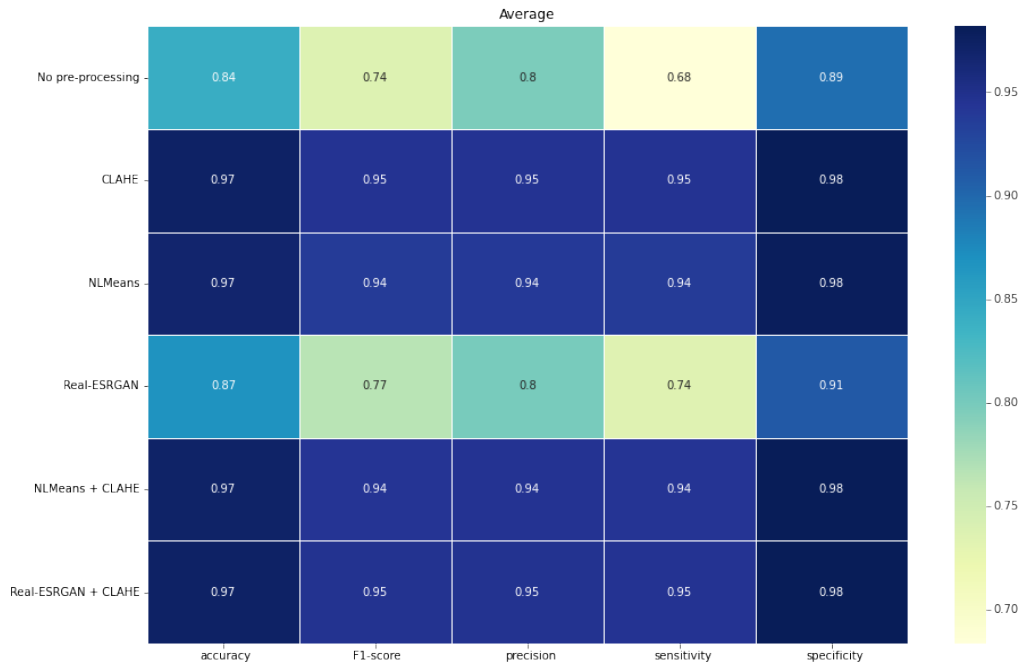


Figure 5.1: Performance metrics for the best ResNet-50 models trained on differently preprocessed Kermany2018 data.

performs 2% better than NLMeans and 1% better than NLMeans + CLAHE and Real-ESRGAN + CLAHE with 97% F1-Score. No pre-processing for this class performs better than Real-ESRGAN with 64% and 52%, respectively. For class NORMAL, however, no pre-processing only performed slightly worse than the other methods with 95% accuracy. CLAHE performed best with 98% accuracy, while NLMeans, NLMeans + CLAHE and Real-ESRGAN + CLAHE achieved 97% accuracy. Again, Real-ESRGAN alone performed worse with 88% accuracy. Moreover, the F1-Score confirm these findings, whereas no pre-processing reaches an F1-Score of 91%, while CLAHE, NLMeans, NLMeans + CLAHE and Real-ESRGAN + CLAHE achieve 96%, 94%, 95% and 95% F1-Score, respectively. Real-ESRGAN, however, performs worse than the aforementioned model with only 80% in F1-Score.

## 5.2 OpticNet-71

As seen in Figure 5.2, one can see that for OpticNet-71 the model trained on CLAHE performs best, although all models perform similar with only one percent difference for average accuracy. Moreover, CLAHE has the lead regarding all other metrics as well, making it the best performing model for all classes. This finding can be confirmed looking at the class dependent accuracies shown in Table 5.3. All models achieve the

Class	No pre-processing	C	NLM	RG	NLM + C	RG + C
CNV	73%	97%	97%	85%	97%	97%
DME	83%	96%	96%	90%	97%	96%
DRUSEN	85%	98%	97%	84%	98%	98%
NORMAL	95%	98%	97%	88%	97%	97%

Table 5.1: Class dependent accuracy of the best ResNet-50 models trained on differently preprocessed Kermany2018 data. C = CLAHE, NLM = NLMeans, RG = Real-ESRGAN

Class	No pre-processing	C	NLM	RG	NLM + C	RG + C
CNV	64%	94%	94%	76%	94%	95%
DME	50%	92%	92%	78%	93%	92%
DRUSEN	64%	97%	95%	52%	96%	96%
NORMAL	91%	96%	94%	80%	95%	95%

Table 5.2: Class dependent F1-Scores of the best ResNet-50 models trained on differently preprocessed Kermany2018 data. C = CLAHE, NLM = NLMeans, RG = Real-ESRGAN

same accuracies for each individual class, while CLAHE is always 1% better. The same observation can be made looking at the individual F1-Scores from Table 5.4, although here the values vary a bit more between the other models. For class CNV, CLAHE achieves an accuracy of 97% and an F1-Score of 94%. As already said, all other models have 1% less in accuracy, but also for this class 2% less F1-Score. For class DME, CLAHE reached an accuracy of 95% and an F1-Score of 90%. Here again, CLAHE is 1% better both in accuracy as well as F1-Score in comparison to the other models. For DRUSEN, however, Real-ESRGAN performs 1% worse while no pre-processing performs 1% better than NLMeans, NLMeans + CLAHE and Real-ESRGAN + CLAHE with 93% and 95% in F1-Score. CLAHE achieves an F1-Score of 96% and an accuracy of 98% for this label, which, again, is 1% better than the accuracy of the other models. The same holds for class NORMAL, where CLAHE achieves an accuracy of 97%. Regarding the F1-Score, however, Real-ESRGAN + CLAHE performs worst for this class with 92%, while CLAHE achieves 94% and the remaining models 93%.

### 5.3 U-Net++

As shown in Figure 5.3 and 5.4, one can see that U-Net++ trained on the CLAHE pre-processed dataset performed best in terms of average Dice and average IoU score with scores of 0.81 and 0.75 respectively. However, it is only slightly better than no pre-processing with a difference of 2% for both metrics. Second best is Real-ESRGAN +

Class	No pre-processing	C	NLM	RG	NLM + C	RG + C
CNV	96%	97%	96%	96%	96%	96%
DME	95%	95%	95%	95%	95%	95%
DRUSEN	97%	98%	97%	97%	97%	97%
NORMAL	96%	97%	96%	96%	96%	96%

Table 5.3: Class dependent accuracy of the best OpticNet-71 models trained on differently preprocessed Kermany2018 data. C = CLAHE, NLM = NLMeans, RG = Real-ESRGAN

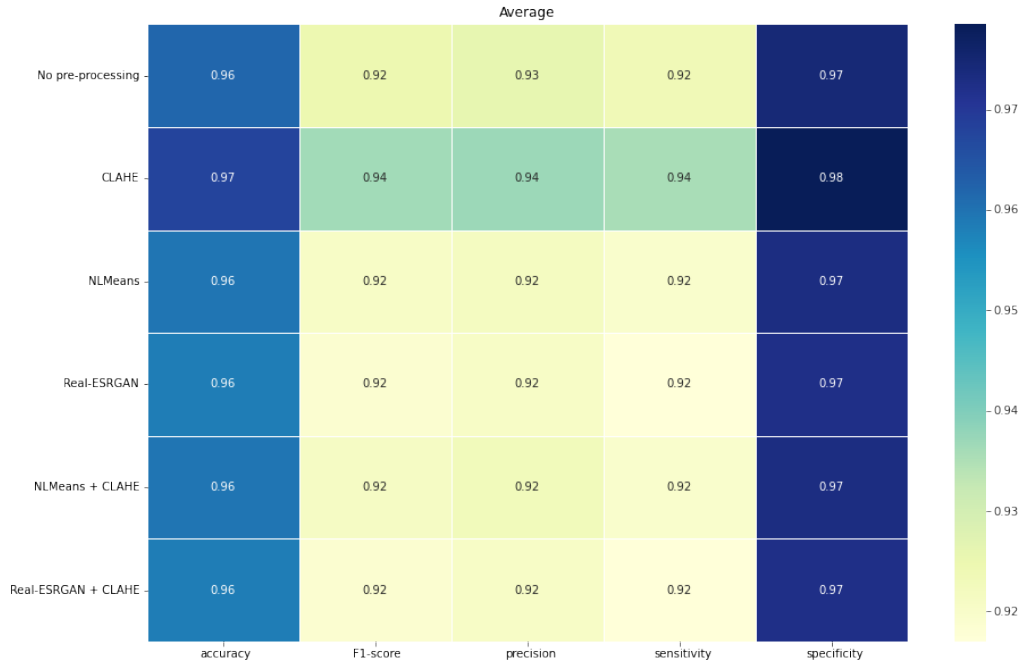


Figure 5.2: Performance metrics for the best OpticNet-71 models trained on differently preprocessed Kermany2018 data.

Class	No pre-processing	C	NLM	RG	NLM + C	RG + C
CNV	92%	94%	92%	92%	92%	92%
DME	89%	90%	89%	89%	89%	89%
DRUSEN	95%	96%	94%	93%	94%	94%
NORMAL	93%	94%	93%	93%	93%	92%

Table 5.4: Class dependent F1-Scores of the best OpticNet-71 models trained on differently preprocessed Kermany2018 data. C = CLAHE, NLM = NLMeans, RG = Real-ESRGAN

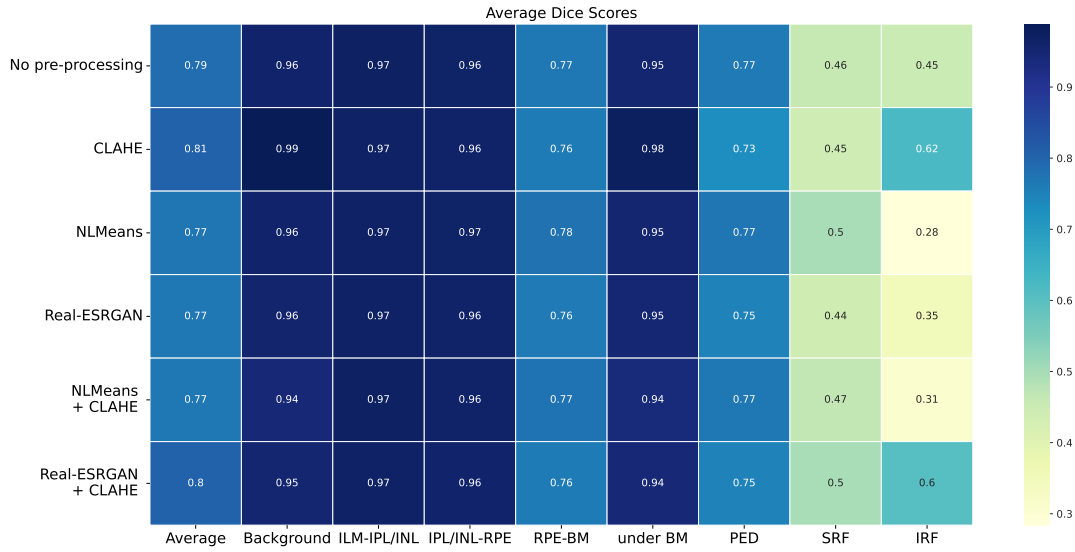


Figure 5.3: Average Dice Scores of the best U-Net++ models trained on differently pre-processed datasets.

CLAHE with 0.8 Dice score. On the third place is no pre-processing with 0.79 Dice Score. All other pre-processed datasets achieved an average Dice score of 0.77. However, looking at the score for the individual layers we can see that the CLAHE pre-processed dataset lead to the best performances for layers Background, under BM and IRF, the first two being background layers. For other layers other methods perform better. For ILM-IPL/INL all models performed the same. For IPL/INL-RPE the NLMeans pre-processed dataset performed best with 0.01 difference to the other models as well as for RPE-BM layer. For PED NLMeans, NLMeans + CLAHE and no pre-processing had the best performance of 0.77. Furthermore, for fluid SRF NLMeans and Real-ESRGAN + CLAHE performed best with 0.5 Dice score. Even tho NLMeans does perform best for some layers, its worse average Dice score comes from its inability to segment IRF correctly, where it only achieves a 0.28 Dice score. This also transfers to NLMeans + CLAHE which achieves a slightly better score of 0.31. The same observations can be made looking at the IoU Scores.

## 5.4 Applying U-Net++ to the Kermany2018 dataset

Additionally to the previous work, I also applied the trained UNet++ models to the Kermany2018 dataset. In general, no pre-processing, NLMeans and Real-ESRGAN produce better masks on this new dataset than CLAHE. However, since there are no segmentation masks for this dataset, this evaluation is subjective to my personal experience and opinion. Figure 5.5 shows one example image of the mask produced by these networks. We can clearly see that CLAHE does produce a very unrealistic mask. NLMeans and Real-ESRGAN seem to improve the models capability of generalizing to a completely

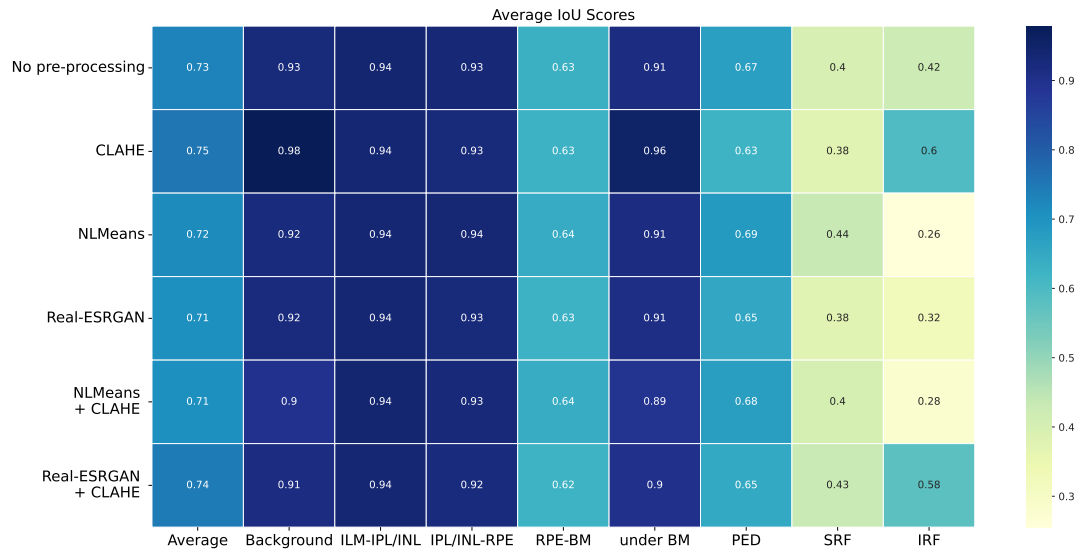


Figure 5.4: Average Intersection-over-Union Scores of the best U-Net++ models trained on differently pre-processed datasets.

new data set, as there are less impossible predictions such as the pink segmentation above all retinal layers as seen in the predicted mask of the model trained on raw data. The Real-ESRGAN trained network produces the most probable mask in this specific example, as there are very few of these predictions that are obviously wrong. However, all segmentation masks seem to be flawed to some extent.



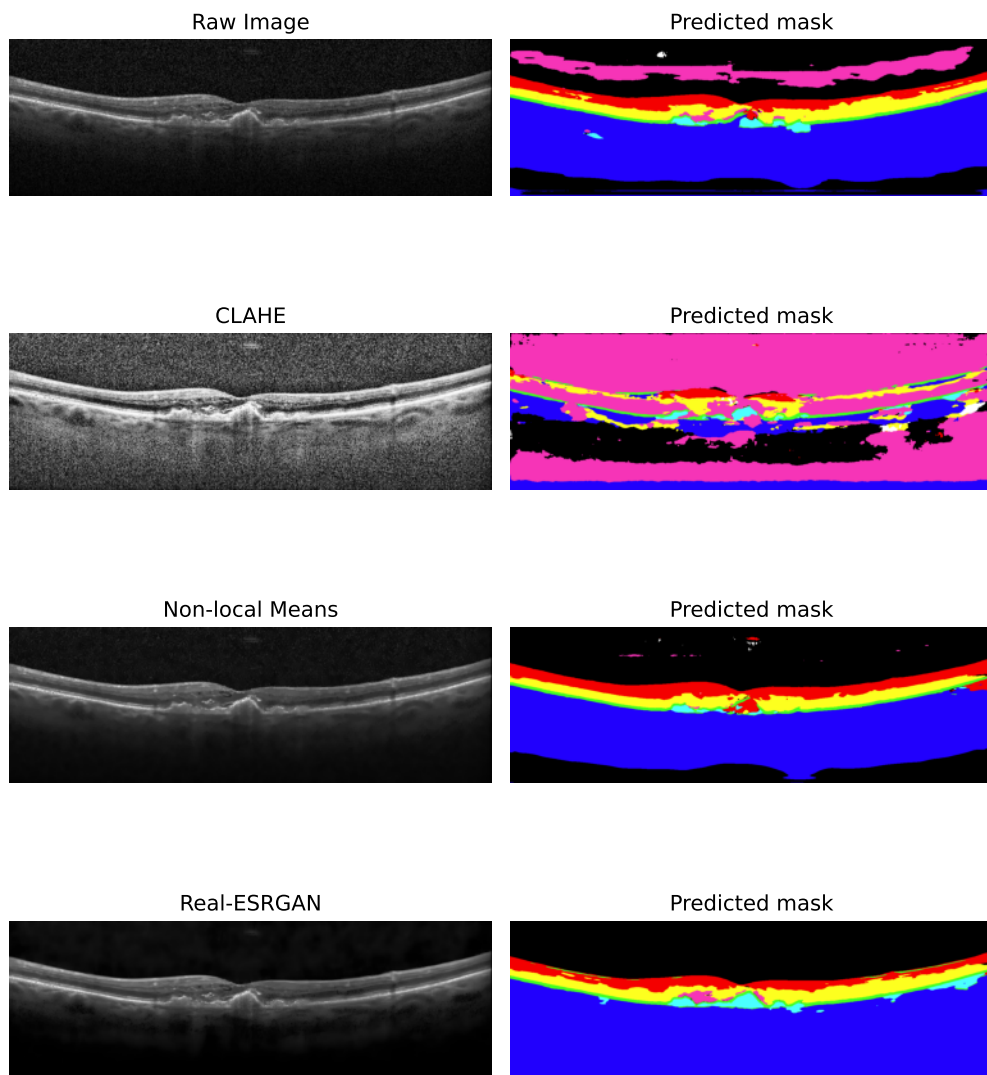


Figure 5.5: Pre-processed images (left) and predicted masks of their correspondent UNet++ model (right). The original image was taken at random from the Kermany2018 dataset.

---

## Chapter 6

### Discussion

For the ResNet-50 architecture, pre-processing majorly improved the models performance in all cases leading to an accuracy increase of 13% for all methods except Real-ESRGAN. Real-ESRGAN also improved the accuracy, but only by a smaller margin of 3%. However, it also improved the average sensitivity and F1-Score of the model making it more useful in a medical diagnosis system. Its lesser impact on the model might be caused by the downscaling, that is done in order to keep the input dimensions the same, as Real-ESRGAN will upsample the images by a four times factor. This downscaling might introduce artifacts, which could explain, why Real-ESRGAN is not performing as well as other methods. However, it still outperforms the not pre-processed model and in combination with CLAHE can lead to very good results. CLAHE itself seems to be the best pre-processing technique for ResNet-50. It lead to the best performance for almost all classes except DME. Especially its accuracy of 98% on the under represented class DRUSEN is remarkable.

For the OpticNet-71 architecture CLAHE merely increased accuracy by 1% in comparison to no pre-processing. However, it also increased sensitivity as well as F1-Score by 2%, which is a desirable result regarding medical diagnosis systems especially for the under represented class DRUSEN. Moreover, all other methods performed exactly the same as no pre-processing making the effort of pre-processing unnecessary for this architecture. However, for the less specialised ResNet-50 architecture all pre-processing methods actually improved their respective models. Therefore, these pre-processing methods might be more useful on general architectures that have not been developed for the specific purpose of classifying OCT scans.

Looking at the segmentation results, we can make similar observations. CLAHE did improve the model compared to no pre-processing regarding Intersection-over-Union and Dice coefficient, although it did not perform best for all layers. More specifically, it segments almost perfectly the background layers, "Background" and "under BM", with 0.99 and 0.98 Dice score respectively. However, it performed the same or even worse for the remaining layers in comparison to the other models. Additionally, it does not segment enclosed fluids such as PED and SRF as good as NLMeans does. This shows that the pre-processing methods applied have an influence on, which layers will be more easily recognised by the network. However, CLAHE segments IRF the best, which is the least represented class in the dataset showing the value these pre-processing methods

can have for medical diagnosis.

The results show that CLAHE improved all models by a fair margin. The other pre-processing methods, in general, could not compete with the performance boost CLAHE offered. However, they lead to better results for specific classes and layers. Over all, pre-processing had a positive impact on the models. Only for the UNet++ architecture, some pre-processing methods lead to slightly worse results than no pre-processing. However, CLAHE seemed to decrease generalizability of the UNet++ architecture. Looking at the results of applying these models to a different dataset, we can clearly observe that CLAHE produces highly unlikely masks. This is probably due to the noise amplification introduced by this method. The two denoising methods, NLMeans and Real-ESRGAN, however, produced much more likely masks, which shows that denoising is still advantageous, when it comes to generalizability of models. Although these models performed better, they are still flawed to an extent, where they are probably useless for a real-world application. This is due to the fact that the AROI dataset is very homogeneous, because all images have the same dimensions and come from the same device. However, the Kermany2018 dataset is very heterogeneous coming from multiple different devices and having different image dimensions.

---

# Chapter 7

## Conclusion

### 7.1 Overview

In this work I have evaluated a selection of pre-processing methods in the context of medical diagnosis regarding the classification and segmentation of OCT scans. The results of my work show that CLAHE is an appropriate pre-processing technique for this purpose, because it improved the results of all models tested regarding certain metrics. Additionally, CLAHE improved the models' performances regarding under represented classes for both datasets, which is especially good for the context of medical diagnosis as disease labels will most likely be less present than healthy labels in any medical dataset. However, CLAHE did not seem to have a positive effect on generalizability, which is probably due to its noise amplification. Moreover, other pre-processing methods for denoising have shown an improving impact on base architectures like ResNet-50, which make them desirable when applying these architectures to a novel medical dataset. Furthermore, a generative model (Real-ESRGAN) was used to improve the images and it performed better than or at least on par with the original images depending on the model that it was used for. Combining denoising and contrast enhancement methods did not outperform the individual pre-processing methods themselves. However, it did perform similar to its individual methods, which goes somewhat against the intuition one got from seeing the pre-processed images. The findings of this work suggest, that denoising methods might improve generalizability.

### 7.2 Outlook

To confirm the findings of my work, further models need to be trained on different data splits. Additionally, models need further hyperparameter tuning to exclude that the different performances of the models are not caused by better fitted hyperparameters. Another interesting point to look into is the generalisability of the models. Pre-processing

could improve the models in this point. Therefore, one needs an independent dataset with the same labels to evaluate the generalisability, which was not possible in the scope of this thesis. However, the Kermany2018 dataset already has a lot of variance with different OCT scans taken by different devices and, hence, the classification models trained on this dataset should generalise well. The AROI dataset was very homogeneous and the models should not generalize very well. However, one could evaluate generalizability with another segmentation dataset. The findings of this work suggest, that denoising methods might help in this case. Moreover, the models and pre-processing techniques need to be evaluated for the human-in-the-loop model mentioned before. However, a proper pipeline needs to be developed first. For this, we need to implement an annotation tool with our DL models, which suggest labelling or segmentations to human experts. These experts can now adjust these predictions and the models can be retrained on the new labelled data. This process makes it easier for medical staff to produce highly detailed labels and segmentations for a large number of OCTs without the need to manually label every pixel of the image. This will in turn produce better models, which should in combination with the shown pre-processing generalize well on new, unseen data from all kinds of devices.

Moreover, it would be interesting to further develop the idea of a generative model as a pre-processing step. The results of this work show that this process can improve a network trained this way. Therefore, it would be interesting to fine-tune a generative network to specifically improve OCT images. Furthermore, one could follow the idea of combining multiple image enhancement methods such as NLMeans + CLAHE. The results of this thesis show that these combinations in general perform at least as good as their individual counterparts and, hence, it might even perform better, if one finetunes these methods.

---

# Bibliography

- [1] 2019. IDF DIABETES ATLAS Ninth Edition 2019. [www.diabetesatlas.org](http://www.diabetesatlas.org). (2019). Accessed: 2022-03-01.
- [2] 2020. How to diagnose and manage diabetic retinopathy. <https://eyeguru.org/essentials/diabetic-retinopathy/>. (2020).
- [3] 2021. Adaptive Histogram Equalization. [https://en.wikipedia.org/wiki/Adaptive\\_histogram\\_equalization](https://en.wikipedia.org/wiki/Adaptive_histogram_equalization). (2021). Accessed on: 2022-03-31.
- [4] 2022. Diabetic Retinopathy. <https://www.diabetes.co.uk/diabetes-complications/diabetic-retinopathy.html>. (2022).
- [5] 2022. Histogram Equalization. [https://en.wikipedia.org/wiki/Histogram\\_equalization](https://en.wikipedia.org/wiki/Histogram_equalization). (2022). Accessed on: 2022-03-31.
- [6] Saqib Alam and Nianmin Yao. 2019. The impact of preprocessing steps on the accuracy of machine learning algorithms in sentiment analysis. *Computational and Mathematical Organization Theory* 25, 3 (2019), 319–335.
- [7] Marco Alban and Tanner Gilligan. 2016. Automated detection of diabetic retinopathy using fluorescein angiography photographs. *Report of standford education* (2016).
- [8] Sarab Almuhaideb and Mohamed El Bachir Menai. 2016. Impact of preprocessing on medical data classification. *Frontiers of Computer Science* 10, 6 (2016), 1082–1102.
- [9] Jean-Michel More Antoni Buades<sup>1</sup>, Bartomeu Coll<sup>2</sup>. 2011. Non-Local Means Denoising. *Image Processing On Line* (2011).
- [10] Silke Aumann, Sabine Donner, Jörg Fischer, and Frank Müller. 2019. *Optical Coherence Tomography (OCT): Principle and Technical Realization*. Springer International Publishing, Cham, 59–85. DOI:[http://dx.doi.org/10.1007/978-3-030-16638-0\\_3](http://dx.doi.org/10.1007/978-3-030-16638-0_3)
- [11] Maxim Berman and Matthew B. Blaschko. 2017. Optimization of the Jaccard index for image segmentation with the Lovász hinge. *CoRR* abs/1705.08790 (2017). <http://arxiv.org/abs/1705.08790>
- [12] A. Buades, B. Coll, and J.-M. Morel. 2005. A non-local algorithm for image denoising. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2. 60–65 vol. 2. DOI:<http://dx.doi.org/10.1109/CVPR.2005.38>
- [13] Ronald E Klein Paul P Lee Carl David Agardh Matthew Davis Diana Dills Anselm Kampik R Pararajasegaram Juan T Verdaguer Global Diabetic Retinopathy Project Group C P Wilkinson 1, Frederick L Ferris 3rd. 2003. *Ophthalmology* (2003). DOI:[http://dx.doi.org/10.1016/S0161-6420\(03\)00475-5](http://dx.doi.org/10.1016/S0161-6420(03)00475-5)

- [14] Bobby Chesney and Danielle Citron. 2019. Deep fakes: A looming challenge for privacy, democracy, and national security. *Calif. L. Rev.* 107 (2019), 1753.
- [15] Chico D. 2017. Ten quick tips for machine learning in computational biology. *BioData Mining* 10 (2017), 35. DOI:<http://dx.doi.org/10.1186/s13040-017-0155-3>
- [16] Niladri Sekhar Datta, Himadri Sekhar Dutta, Mallika De, and Saurajeet Mondal. 2013. An Effective Approach: Image Quality Enhancement for Microaneurysms Detection of Non-dilated Retinal Fundus Image. *Procedia Technology* 10 (2013), 731–737. DOI:<http://dx.doi.org/https://doi.org/10.1016/j.protcy.2013.12.416> First International Conference on Computational Intelligence: Modeling Techniques and Applications (CIMTA) 2013.
- [17] J. De Fauw, J.R. Ledsam, and B. et al Romera-Paredes. 2018. Clinically applicable deep learning for diagnosis and referral in retinal disease. *Nature Medicine* 24 (2018), 1342–1350. DOI:<http://dx.doi.org/10.1038/s41591-018-0107-6>
- [18] Kartika Firdausy, Oyas Wahyunggoro, Hanung Adi Nugroho, Muhammad Bayu Sasongko, and Risanuri Hidayat. 2019. Impact of Different Degree of Smoothing on Non-Local Means based Filter for Retinal Vessel Modeling. In *2019 5th International Conference on Science in Information Technology (ICSITech)*. 118–122. DOI:<http://dx.doi.org/10.1109/ICSITech46713.2019.8987555>
- [19] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger (Eds.), Vol. 27. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf>
- [20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. (2015). DOI:<http://dx.doi.org/10.48550/ARXIV.1512.03385>
- [21] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2016. Densely Connected Convolutional Networks. (2016). DOI:<http://dx.doi.org/10.48550/ARXIV.1608.06993>
- [22] Jianglin Huang, Yan-Fu Li, and Min Xie. 2015. An empirical analysis of data preprocessing for machine learning-based software cost estimation. *Information and Software Technology* 67 (2015), 108–127. DOI:<http://dx.doi.org/https://doi.org/10.1016/j.infsof.2015.07.004>
- [23] Paul Jaccard. 1912. THE DISTRIBUTION OF THE FLORA IN THE ALPINE ZONE. (1912). DOI:<http://dx.doi.org/10.1111/j.1469-8137.1912.tb05611.x>
- [24] Lukasz Kaiser, Aidan N. Gomez, and Francois Chollet. 2017. Depthwise Separable Convolutions for Neural Machine Translation. *arXiv* (2017). <https://arxiv.org/pdf/1706.03059.pdf>
- [25] Sharif Amit Kamran, Sourajit Saha, Ali Shihab Sabbir, and Alireza Tavakkoli. 2019. Optic-Net: A Novel Convolutional Neural Network for Diagnosis of Retinal Diseases from Optical Tomography Images. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. 964–971.

- [26] Goldbaum Michael Kermany Daniel, Zhang Kang. 2018. Large Dataset of Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images. *Mendeley Data* (2018). DOI:<http://dx.doi.org/10.17632/rscbjbr9sj.3>
- [27] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. (2014). DOI:<http://dx.doi.org/10.48550/ARXIV.1412.6980>
- [28] Shailesh Kumar and Basant Kumar. 2018. Diabetic Retinopathy Detection by Extracting Area and Number of Microaneurysm from Colour Fundus Image. In *2018 5th International Conference on Signal Processing and Integrated Networks (SPIN)*. 359–364. DOI:<http://dx.doi.org/10.1109/SPIN.2018.8474264>
- [29] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. 2015. Deeply-supervised nets. In *Artificial intelligence and statistics*. PMLR, 562–570.
- [30] Z. Vatauvuk M. Melinščak, M. Radmilović and S. Lončarić. 2021. Annotated retinal optical coherence tomography images (AROI) database for joint retinal layer and fluid segmentation. *Automatika* 62, 3 (2021), 375–385.
- [31] Miguel Monteiro, Konstantinos Kamnitsas, Enzo Ferrante, Francois Mathieu, Steven McDonagh, Sam Cook, Susan Stevenson, Tilak Das, Aneesh Khetani, Tom Newman, and others. 2019. TBI lesion segmentation in head CT: Impact of preprocessing and data augmentation. In *International MICCAI Brainlesion Workshop*. Springer, 13–22.
- [32] Tiago S. Nazaré, Gabriel B. Paranhos da Costa, Welinton A. Contato, and Moacir Ponti. 2018. Deep Convolutional Neural Networks and Noisy Images. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, Marcelo Mendoza and Sergio Velastín (Eds.). Springer International Publishing, Cham, 416–424.
- [33] Duy M. H. Nguyen, Truong T. N. Mai, Ngoc T. T. Than, Alexander Prange, and Daniel Sonntag. 2021. Self-Supervised Domain Adaptation for Diabetic Retinopathy Grading using Vessel Image Reconstruction. (2021). DOI:<http://dx.doi.org/10.48550/ARXIV.2107.09372>
- [34] American Academy of Ophthalmology. 1980. What Is Optical Coherence Tomography? <https://www.aaao.org/eye-health/treatments/what-is-optical-coherence-tomography>. (1980). Accessed: 2022-03-01.
- [35] Juan Guillermo Paniagua, David Restrepo Rivera, Leandro Ariza-Jiménez, Jose Julian Garces Echeverri, Christian Andrés Diaz León, Diana Lucia Serna-Higuaita, Wiston Arrazola, Sebastian Arango, Ramiro Velez-Koeppel, Miguel Angel Mejia, and others. 2021. Influence of preprocessing and segmentation on the complexity of the learning machines in medical imaging. *Intelligence* 19, 1 (2021), 157–177.
- [36] Paul-Louis Pröve. 2017. An Introduction to different Types of Convolutions in Deep Learning. <https://towardsdatascience.com/types-of-convolutions-in-deep-learning-717013397f4d>. (2017). Accessed: 2022-03-01.
- [37] Husna Ab Rahim, Ahmad Syahir Ibrahim, W Mimi Diyana W Zaki, and Aini Hussain. 2014. Methods to enhance digital fundus image for diabetic retinopathy detection. In *2014 IEEE 10th International Colloquium on Signal Processing and its Applications*. 221–224. DOI:<http://dx.doi.org/10.1109/CSPA.2014.6805752>



- [38] Girish Singh Ramlugun, Vivek Krishna Nagarajan, and Chandan Chakraborty. 2012. Small retinal vessels extraction towards proliferative diabetic retinopathy screening. *Expert Systems with Applications* 39, 1 (2012), 1141–1146. DOI:<http://dx.doi.org/https://doi.org/10.1016/j.eswa.2011.07.115>
- [39] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015a. U-Net: Convolutional Networks for Biomedical Image Segmentation. (2015). DOI:<http://dx.doi.org/10.48550/ARXIV.1505.04597>
- [40] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015b. U-Net: Convolutional Networks for Biomedical Image Segmentation. (2015). DOI:<http://dx.doi.org/10.48550/ARXIV.1505.04597>
- [41] J. D. Austin et al. S. M. Pizer, E. P. Amburn. 1987. Adaptive Histogram Equalization and Its Variations. *Computer Vision, Graphics, and Image Processing* 39 (1987), 355–368.
- [42] Siva Ram Krishna Vadali Saikat Kumar Shome. 2011. Enhancement of Diabetic Retinopathy Imagery Using Contrast Limited Adaptive Histogram Equalization. *International Journal of Computer Science and Information Technologies* 2 (2011), 2694–2699.
- [43] Sowmya Sanagavarapu, Sashank Sridhar, and T.V. Gopal. 2021. COVID-19 Identification in CLAHE Enhanced CT Scans with Class Imbalance using Ensembled ResNets. In *2021 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*. 1–7. DOI:<http://dx.doi.org/10.1109/IEMTRONICS52119.2021.9422556>
- [44] Santhi Sayanandini. 2019. Improved OCT Image Enhancement using CLAHE. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)* 8 (2019).
- [45] Joseph M Schmitt, SH Xiang, and Kin Man Yung. 1999. Speckle in optical coherence tomography. *Journal of biomedical optics* 4, 1 (1999), 95–105.
- [46] Ophthalmic Photographers’ Society. 1978. Fundus Photography Overview. <https://www.opsweb.org/page/fundusphotography>. (1978). Accessed: 2022-03-01.
- [47] Pratul P. Srinivasan, Leo A. Kim, Priyatham S. Mettu, Scott W. Cousins, Grant M. Comer, Joseph A. Izatt, and Sina Farsiu. 2014. Fully automated detection of diabetic macular edema and dry age-related macular degeneration from optical coherence tomography images. *Biomed. Opt. Express* 5, 10 (Oct 2014), 3568–3577. DOI:<http://dx.doi.org/10.1364/BOE.5.003568>
- [48] Robyn J. Tapp, Jonathan E. Shaw, C. Alex Harper, Maximilian P. de Courten, Beverley Balkau, Daniel J. McCarty, Hugh R. Taylor, Timothy A. Welborn, Paul Z. Zimmet, and on behalf of the AusDiab Study Group. 2003. The Prevalence of and Factors Associated With Diabetic Retinopathy in the Australian Population. *Diabetes Care* 26, 6 (06 2003), 1731–1737. DOI:<http://dx.doi.org/10.2337/diacare.26.6.1731>
- [49] Ekin Tiu. 2019. Metrics to Evaluate your Semantic Segmentation Model. *Towards Data Science* (2019).
- [50] Alper Kursat Uysal and Serkan Gunal. 2014. The impact of preprocessing on text classification. *Information processing & management* 50, 1 (2014), 104–112.

- [51] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip Yu. 2022. Generalizing to Unseen Domains: A Survey on Domain Generalization. *IEEE Transactions on Knowledge and Data Engineering* (2022), 1–1. DOI:<http://dx.doi.org/10.1109/TKDE.2022.3178128>
- [52] Shuai Wang, Xiaojun Xia, Lanqing Ye, and Binbin Yang. 2021a. Automatic Detection and Classification of Steel Surface Defect Using Deep Convolutional Neural Networks. *Metals* 11 (02 2021), 388. DOI:<http://dx.doi.org/10.3390/met11030388>
- [53] Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan. 2021b. Real-ESRGAN: Training Real-World Blind Super-Resolution with Pure Synthetic Data. (2021).
- [54] Jian Yang, Jingfan Fan, Danni Ai, Xuehu Wang, Yongchang Zheng, Songyuan Tang, and Yongtian Wang. 2016. Local statistics and non-local mean filter for speckle noise reduction in medical ultrasound image. *Neurocomputing* 195 (2016), 88–95. DOI:<http://dx.doi.org/https://doi.org/10.1016/j.neucom.2015.05.140> Learning for Medical Imaging.
- [55] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. 2021. Domain Generalization: A Survey. (2021). DOI:<http://dx.doi.org/10.48550/ARXIV.2103.02503>
- [56] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. 2018. UNet++: A Nested U-Net Architecture for Medical Image Segmentation. (2018).
- [57] A.P. Zijdenbos, B.M. Dawant, R.A. Margolin, and A.C. Palmer. 1994. Morphometric analysis of white matter lesions in MR images: method and validation. *IEEE Transactions on Medical Imaging* 13, 4 (1994), 716–724. DOI:<http://dx.doi.org/10.1109/42.363096>
- [58] zonasw. 2020. Unet and Unet++: multiple classification using Pytorch. <https://github.com/zonasw/unet-nested-multiple-classification>. (2020). Accessed: 2022-03-01.