Development of machine learning algorithms operating on small datasets to assist in the diagnosis of eye diseases

Dominika Sułot

Wrocław University of Science and Technology

A thesis submitted for the degree of Doctor of Philosophy at Wroclaw University of Science and Technology in 2022

Department of Biomedical Engineering

Supervisors:

D. Robert Iskander

David Alonso-Caneiro

Above all, don't fear difficult moments. The best comes from them.

Rita Levi-Montalcini

Abstract

This dissertation concerns supporting the diagnosis of eye diseases with the use of increasingly rapidly developing artificial intelligence methods. As this field grows rapidly, it has become more and more data-voracious and needs a large amount of data to achieve high and stable results. In the case of medical data, access to large datasets or obtaining them is sometimes not feasible, consequently, methods of working on such limited datasets are needed to support medicine with machine learning algorithms. Therefore, this dissertation focuses on supporting the diagnosis of eye diseases with machine learning algorithms, with the additional consideration that all experiments should be carried out in the so-called low-data regime. Four different machine learning approaches are evaluated in depth, including, image classification using undervalued Scanning Laser Ophthalmoscopy images, image segmentation, feature selection, and a general approach for the low-data regime in image processing. In each approach, different machine learning algorithms were used and different approaches to dealing with a challenging dataset were compared. The obtained results, in terms of developed machine learning algorithms, show that despite the additional difficulties it is possible to work with small datasets in the context of machine learning algorithms. They also show that, with the dataset used, the best approach is to choose simple and shallow deep learning architectures, ensemble learning, and, for image data, to use data augmentation. Additionally, the classification study confirmed that Scanning Laser Ophthalmoscopy images can be successfully used to support glaucoma diagnostics as the accuracy of the classifier can reach over 96 percent. Given the challenging to access large dataset in certain medical application, the findings of the dissertation may inform future studies in the field to help overcome some of the challenges while dealing with low-data regime problems.

Acknowledgments

I would like to especially thank my main supervisor, D. Robert Iskander, for his continued support. His optimistic attitude made it much easier to go through the entire course of studies and, in fact, made it possible to complete it. Through all the long discussions about the experiments but also their directions, he has guided me and taught me more than I could ever expect. I would also like to thank my second supervisor, David Alonso-Caneiro, for helping me better understand machine learning in medical applications. Also, for the time he devoted to me both during my stay at the Queensland University of Technology, but also during countless online consultations. I could not imagine having better supervisors for my Ph.D. thesis.

In addition, I would like to highlight the contribution of my friends and family who sometimes listened to perhaps less interesting stories about experiments and research. They were always available to me and supported me, which allowed me to develop my scientific interest and never feel alone. I would especially like to thank Paulina Graba, who helped me design the cover for this dissertation.

All appreciation also to my colleagues from the research team and university, who accepted me very friendly to the team and always offered advice and conversation. They shared their experience and knowledge, which accelerated my development. Additionally, I would like to thank Dr. Paweł Ksieniewicz for the opportunity to cooperate at the Faculty of Electronics on experiments with numerical data.

In addition, I would like to point out that to create this dissertation I used the public LaTeX template provided by the University of Queensland.

Publications included in this thesis

1. [1] **Dominika Sułot**, David Alonso-Caneiro, D. Robert Iskander and Michael J. Collins, Deep learning approaches for segmenting Bruch's membrane opening from OCT volumes, *OSA Continuum*, 3, 12, 2020

2. [2] **Dominika Sułot**, David Alonso-Caneiro, Paweł Ksieniewicz, Patrycja Krzyżanowska-Berkowska and D. Robert Iskander, Glaucoma classification based on scanning laser ophthalmoscopic images using a deep learning ensemble method, *Plos one*, 16, 6, 2021

3. [3] **Dominika Sułot**, Paweł Zyblewski and Paweł Ksieniewicz, Analysis of Variance Application in the Construction of Classifier Ensemble Based on Optimal Feature Subset for the Task of Supporting Glaucoma Diagnosis, *International Conference on Computational Science*, 2021

Other publications during candidature

Conference Proceedings

1. [4] **Dominika Sułot**, Selection of Interpretable Decision Tree as a Method for Classification of Early and Developed Glaucoma, IEEE EMBS International Student Conference (pp. 144-150). Springer, Cham, 2020.

Monograph chapter

1. [5] **Dominika Sułot**, Paweł Zyblewski, and Paweł Ksieniewicz, A novel approach to learning and designing neural networks-based ensemble, *Selected model based architectures and algorithms for learning, signal processing and optimization, EXIT*, 2021.

Conference abstracts

1. [6] **Dominika Sułot**, David Alonso-Caneiro, and D. Robert Iskander, Differentiating glaucoma patients from healthy controls and glaucoma suspects using deep learning, *Investigative Ophthalmology & Visual Science*, 61, 7, 2020

Research involving human or animal subjects

The study was approved by the Bioethical Committee of the Wroclaw Medical University (kb–332/2015) and adhered to the tenets of the Declaration of Helsinki. Informed written consent to participate in the study was obtained from all subjects.

Financial support

The doctorate was supported by by InterDok – Interdisciplinary Doctoral Studies Projects at Wroclaw University of Science and Technology, a project co-financed by the European Union under the European Social Fund and PhD scholarship from Wrocław University of Science and Technology. The internship in Queensland University of Technology was funded by InterDok.

Keywords

machine learning, eye diseases, low-data regime, deep learning, glaucoma, feature selection, classification, segmentation, orthogonality, image processing, convolutional neural network

To my family,

for their support in moments of uncertainty

Contents

List of Figures

List of Tables

List of Abbreviations

ACC	Accuracy
Adam	Adaptive moment estimation
AE	Autoencoders
AI	Artificial intelligence
AMD	Age-related macular degeneration
ANN	Artificial neural network
ASM	Angular second moment
BAC	Balanced accuracy
BMO	Bruch's membrane opening
CART	Classification and regression trees
CNN	Convolutional neural networks
CT	Computer tomography
DL	Deep learning
DNN	Deep neural network
DSC	Dice coefficient
DT	Decision trees
FCN	Fully convolutional network
FN	False negative
FP	False positive
GAN	Generative adversarial network
GLCM	Gray-level concurrence matrix
GNB	Gaussian naive Bayes
JSC	Jaccard similarity coefficient
k-NN	k-nearest neighbours
LSTM	Long short-term memory

MAE	Mean absolute error
MDAE	Median absolute error
ME	Mean error
MIT	Massachusetts Institute of Technology
ML	Machine learning
MLP	Multilayer perceptron
MRI	Magnetic resonance imaging
mRMR	minimum redundancy maximum relevance
MV	Majority voting
NN	Neural network
OCNN	Orthogonal convolutional neural network
OCT	Optical coherence tomography
ONH	Optic nerve head
PCA	Principal components analysis
POAG	Primary open-angle glaucoma
ReLu	Rectified linear unit
RMSE	Root mean square error
RNFL	Retinal nerve fiber layer
RNN	Recurrent neural network
SA	Support accumulation
SE	Squeeze-excitation
SLO	Scanning laser ophthalmoscopy
SVM	Support vector machine
TN	True negative
TNR	True negative rate
TP	True positive
TPR	True positive rate
UBPE	Unsigned border positioning error
VAE	Variational autoencoder

Preface

The motivation behind the work presented in this dissertation comes from a combination of scientific interests. It concerns two different fields of research: computer science and biomedical engineering, of which computer science focuses mostly on machine learning algorithms, in particular for image analysis applications. From this union, a topic on which the thesis is focused was created, and it is as follows: *Development of machine learning algorithms operating on small datasets to assist in the diagnosis of eye diseases*. Throughout this dissertation, the reader will be guided to this specific topic and some of the studies that have been performed during the last four years. For the image analysis, some of the studies include the topic of eye diseases, particularly glaucoma detection. Patients suffering from glaucoma are often diagnosed with the visual field procedure, which is very time-consuming. While other imaging modalities such as Optical Coherence Tomography or Scanning Laser Ophthalmoscopy, which are faster and also usually collected as part of the screening process tend to be overlooked. Therefore, it is possible to support diagnostics by improving the process of analyzing the available images. Nowadays, machine learning algorithms are the most popular solution for data analysis tasks such as classification, and segmentation, for which manual, expert-based handling is laborious and prone to mistakes, for example, due to fatigue. Regarding image analysis itself, deep learning is getting the most attention. Furthermore, it enables automatic feature selection without the need to manually prepare image features for further classification or segmentation. The difficulty in using such algorithms in medical problems is the so-called low-data regime that frequently occurs in those cases. This relates to having a ‘low’ amount of images to train the model. The risk of using small sets of data is overfitting or a complete inability to learn the task (model returns random results). Therefore, within this thesis, this limitation will be discussed and some solutions will be presented to obtain the highest possible (in terms of accuracy) and most stable results under these conditions.

The thesis is organized into four chapters. The first is the *Introduction* that summarizes the current state of development of the machine learning techniques supporting the diagnosis of eye diseases, and in addition, it contains basic information on the field of machine learning methods. The second chapter, *Research summary*, contains a description of all studies that have been performed, together with the main motivation for their implementation and the experimental protocol. In the third chapter, the results obtained from research performed during the course of doctoral studies are presented. The fourth and final chapter connects the whole dissertation and provides the concluding remarks from all of the findings. Additionally, some of the future directions are outlined.

Hypotheses

The four main hypotheses for the thesis are presented below.

1. Low-resolution Scanning Laser Ophthalmoscopy images contain substantial information that can support glaucoma diagnosis, especially when combined with the use of deep learning algorithms.
2. The size of the data fed into a neural network can have a significant impact on performance, particularly in the case of Bruch's membrane opening segmentation, the shape of input data can affect the final results of the trained model despite using the same dataset across different models.
3. Machine learning algorithms can be successfully applied in the low-data regime applications, provided they are equipped with some additional techniques to deal with the small dataset.
4. The use of orthogonalization with convolutional networks can have a significant positive impact on performance, particularly when the model is trained on small datasets.

Chapter 1

Introduction – Artificial Intelligence

Although the topics related to *artificial intelligence* (AI) were already theoretically considered by philosophers several hundred years ago, its real development began only in the twentieth century with the invention of the computer. Determining the specific date of the beginning of AI as a field of science is an issue, but the most frequently chosen year is 1956 when the summer workshop at Dartmouth College took place. It is worth noting that the first works on AI and related computer programs were created earlier. Even a well-known Turing test, originally called an imitation game, was proposed in 1950 [8]. However, during that summer workshop, held under the name of Dartmouth Summer Research Project on Artificial Intelligence, the name Artificial Intelligence was proposed for the first time. Although criticized many times during the workshop, this terminology has remained as such until today. The main organizer of that workshop was John McCarthy, professor of mathematics at Dartmouth, with the help of Marvin Minsky, Nathaniel Rochester, and Claude Shannon. This several-week meeting, which included scientists from various scientific fields, was to establish common definitions and general development paths in a newly created field [9]. Since then, the growth of AI has accelerated and expanded to other fields and applications. A few years later, from 1964 to 1966 at the Massachusetts

Institute of Technology (MIT), Eliza – the first algorithm for natural language processing was created. It was the prototype of the chatbots in use today [10]. Even during the so-called 'AI winter', which took place in total from 1970 to 1990 (the first and second AI winter combined) and consisted in reducing funding for research related to AI, there were many studies on the subject, which later transformed into the basics of today's Machine Learning. During that time, for example, Kohonen networks were created, which gave rise to Competitive Learning [11]. Also in the field of medicine, efforts were made to use those algorithms. An example in the field of eye diseases from those years is CASNET – a program for glaucoma consultation [12]. Additionally, the first conference on neural networks (the First International Conference on Neural Networks) took place in 1987 [13]. Since then, AI has been developing rapidly, especially the subdivision of AI called *machine learning* (ML). ML allows machines to learn and infer rules based on the data provided to them. Nowadays, AI is applied to almost every area of life, from medical applications, such as supporting diagnostics, to recommendation systems, for example when selecting films on streaming platforms.

1.1 Machine learning

Machine learning (ML), which is considered to be a part of AI, is a wide field of science and the range of its applications is even wider. Nowadays, it is not surprising to see that ML algorithms are used almost in every scientific and consumer application. However, due to the fact that this field is so broad, it is difficult to talk about it in its entirety without giving an appropriate division into its different components. Many possible kinds of division could be considered, nevertheless, only one of them will be discussed in this description because it is the most important for further understanding of the considered research. However, before this division is presented, it is worth noting that in this work, an ML *algorithm* is understood as a set of procedures that are to perform a specific task, whereas the ML *model* corresponds to the learned algorithm with all the parame-

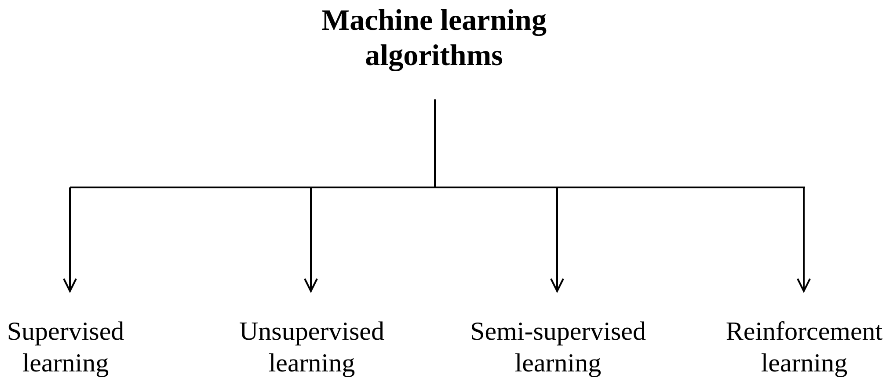


Figure 1.1: Learning-based types of machine learning algorithms.

ters that have been adjusted in the learning process, which can perform the specified task based on the new data provided. There are four different types of ML algorithms divided according to the different learning processes [14]. These are presented in Fig. 1.1. They differ based on the data they are learned from. In *supervised learning*, the algorithm is fed with the data and the corresponding labels or annotations. So it can be said that it learns from the example of pairs: example data – that is, input and their labels – output. According to the type of expected output (classes or continuous values), we can further divide this type into classification, regression, or segmentation tasks.[1] In this case of *unsupervised learning*, the algorithm does not have labels for the training data and its task is to uncover similarities and patterns in the data and based on that information, in most cases, group the data. Unsupervised learning is commonly used for clustering, segmentation, or data compression. The next learning-based type is created from the combination of the two previously mentioned categories and it is called *semi-supervised learning*. The algorithm in this case operates both on data with and without labels. This is especially useful in cases where data labeling is expensive and there is a substantial amount of unlabeled data. It is used in both classification and clustering tasks. The last type of ML algorithm, divided based on the learning process, is the *reinforcement learning*, whose principle of operation is different than those of the previous ones. Instead

[1]In supervised learning the most popular is semantic segmentation, which is a classification of each pixel in the image.

of providing training data, only information about the direction of action is used here to inform the algorithm, this action can be extracted from the interaction of the algorithm with the environment. There are two types of feedback: positive (reward) and negative (penalty). The goal of the algorithm is to maximize the reward and minimize the penalty. It is used, for example, in autonomous cars. Summarizing this division, the ML algorithm should be selected based on the analysis of available data and the final purpose of the algorithm. As mentioned previously, there is more than one division for ML algorithms. Others include, for example, division on the basis of whether it learns on all available data at one time or whether the data is provided to the algorithm, e.g., incrementally. This type of division is not going to be described in more detail here because, in all the studies discussed in this work, the algorithms have access to all possible data at once.

1.1.1 Standard machine learning algorithms

Considering the above-mentioned types of ML categories, special attention is paid to supervised learning due to the subject matter of the work. Many different algorithms are dedicated to particular problems, datasets, etc. In the case of the numerical datasets, the five most popular standard ML algorithms, which are commonly used in the benchmark process against others, will be discussed in the following paragraphs.

Multilayer perceptron

The first algorithm is the *multilayer perceptron* (MLP) proposed in 1985 [15]. This is an algorithm from the category of neural networks. They were inspired by the human neural network in the brain. The basic unit of an artificial neural network is a neuron, which is to aim to replicate the real human neuron. Fig. 1.2 shows a schematic view of both the real and the artificial neuron. On the left side, there are inputs. The main difference in this input is that in the artificial input is some numerical value and in the real input is an electrical and chemical impulse through dendrites. After that, the signals are combined and processed by a certain activation function and the re-

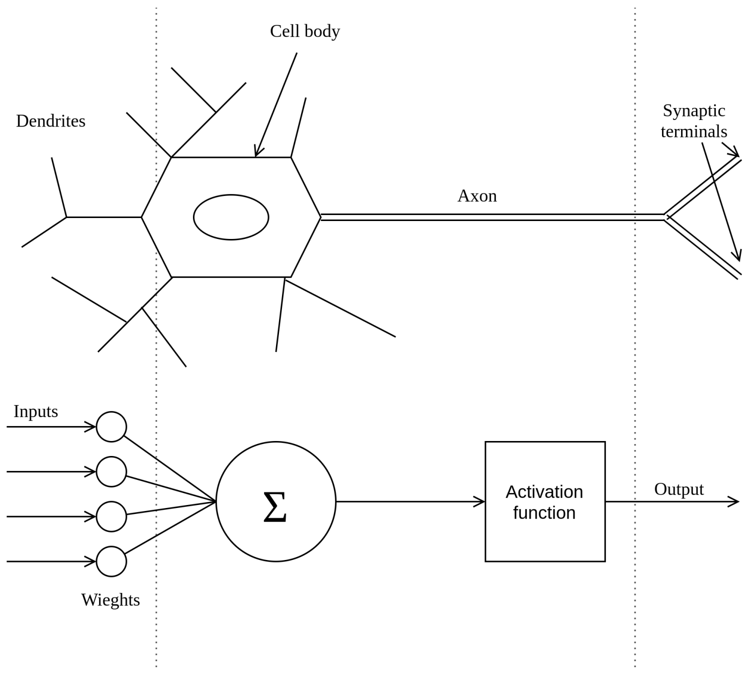


Figure 1.2: Schematic view of human (top) and artificial neuron (bottom). On the left side, there are impulses (top) and numeric values (bottom) inputs. In the middle, there is signal combination and activation. On the right are the impulse (top) and the numeric values (bottom) output.

sult in both neurons is an output value or impulse. Thus, as such, artificial neurons are just a simplified version of the biological model of the neuron. The simplest neuronal network, which may even consist of just one neuron, is the perceptron – proposed in 1958 by Frank Rosenblatt [16]. It is a binary classifier, the major disadvantage of which is that it works only on linearly separable problems. The solution to this limitation is a more advanced version of the neural network, called the MLP. It is a feedforward network, and it consists of more layers of neurons. If there are more than three layers, it is considered a *deep learning* (DL) algorithm. Since MLP has more layers, it is able to solve nonlinear problems as well. Because of this and its simplicity, it is still a commonly used algorithm today.

Gaussian Naive Bayes

The second algorithm, having a different operating principle, is the *Gaussian Naive Bayes* (GNB) [17]. It is an algorithm from the group of probabilistic classifiers, which means that during classification it calculates the probability of the feature vector belonging to each possible class. It operates based on Bayes' theorem (1.1) and it assumes that given features are independent. It is based on the conditional probability of events (A and B). Despite its simplicity and the assumption of independence, it is commonly used. In many real-world applications in which this assumption of independence is not fulfilled, the GNB algorithm can still perform well. In addition, it works efficiently for multidimensional data and is also increasingly used in natural language processing [18].

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1.1}$$

Decision trees

The next algorithm that can be used for both regression and classification problems is the *Decision Tree* (DT). They belong to a group of algorithms with a rule-based approach. In a simplified way, it can be said that decision trees are algorithms that focus on subsequent conditional statements executed until they reach the leaves, i.e., in the case of classification – the target class. An example of such a decision tree is presented in Fig. 1.3. The initial set of features is divided according to a specific rule into two nodes that can be further divided if needed. There are two basic types of rules – univariate and multivariate, depending on the number of features that it used to divide the tree. As a result, the optimal (in the sense of interpretability) model should have a small number of nodes and small depth, because in this way the user can assess what was the reason for such a decision. Some methods try to prune trees to reduce their size, but this topic is outside the scope of this work.

K-nearest neighbors algorithm

The *k-nearest neighbors algorithm* (k-NN) is another type of algorithm used for both regression and classification problems [19]. It is a non-parametric method. It operates by comparing an object with the k-closest objects in the training set. The closest ones are understood as the objects with

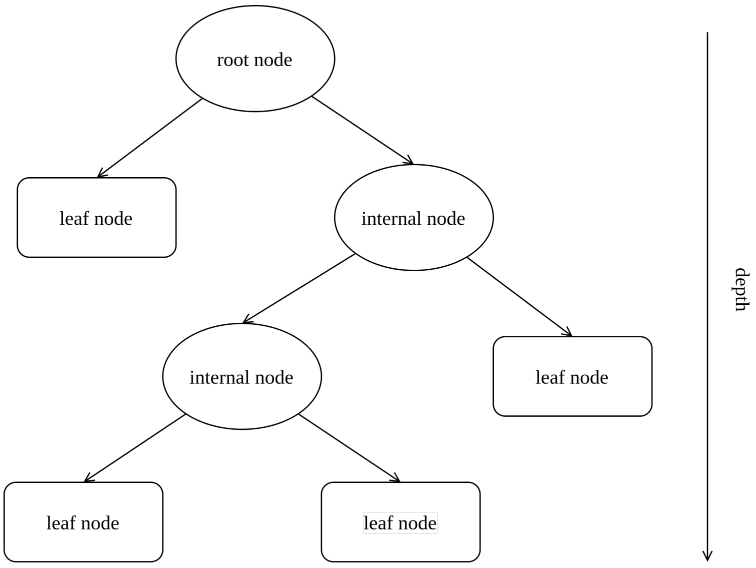


Figure 1.3: A sample decision tree diagram. At the top is the root node where the algorithm starts. Then the nodes are divided into internal or leaf nodes. Internal – if a further division is still to be performed, leaf – this is the last node which, for example, is a target class in the classification task.

the lowest value of the set distance.[2] Then the class is selected as the most common class among these k-objects in the classification task and it is an average of them in the case of regression. This algorithm belongs to a group of *instance base learning*, which means that all the processing is performed during classification, not like in most other algorithms, which occurs during training. Unfortunately, due to the need to calculate distances for each object in the training set, for large datasets and high-dimensional data, k-NN has low computational efficiency. However, there are several search algorithms available to help deal with this problem by speeding up the entire algorithm [20].

Support Vector Machine

The last ML algorithm for supervised learning discussed here is the *Support Vector Machine* (SVM) [21]. The principle of operation of this algorithm is that it iteratively searches the best (in some sense) hyperplane to separate

[2]The distance can be defined in many ways, depending on the data type.

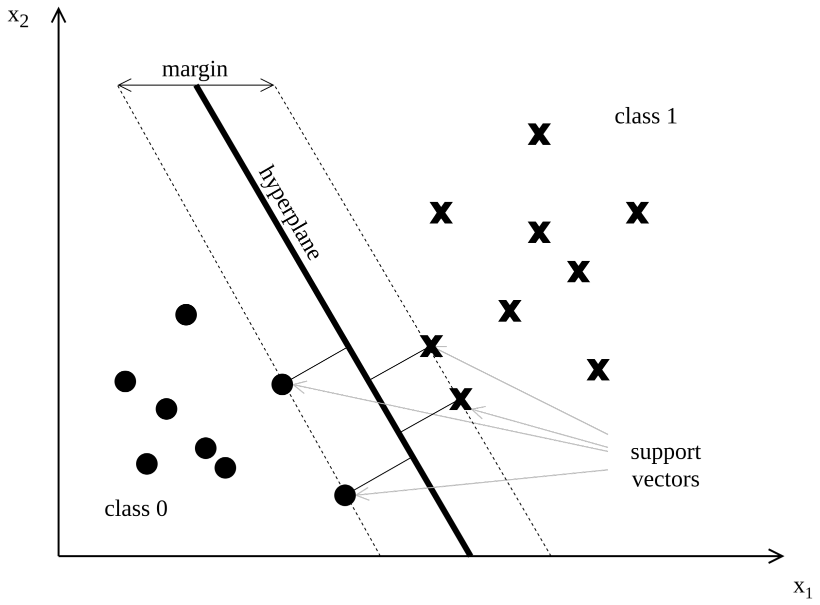


Figure 1.4: Schematic view of Support Vector Machine principle of operation. The axes contain the values of two features (x_1 and x_2) based on which the classification is made. The bold black line marks the designated hyperplane based on support vectors, which are objects closest to the dividing line. Margin determines the distance between the support vectors for two classes.

two classes[3] in n-dimensional space. This hyperplane should fulfill the condition of maximizing the margin between these two classes. The *support vector* refers to the data points that are closest to the hyperplane and based on them the margin is calculated. The schematic view of this principle is presented in Fig. 1.4 with hyperplane, margin, and support vectors shown. SVM has proved to be useful even with high-dimensional datasets and, with the choice of appropriate kernel function [22], it can provide good generalization power and most often outperforms neural networks. It is capable of handling both continuous and categorical numeric data. Depending on the available dataset, there is a need to choose the right kernel function that will convert the data to an appropriate form. Additionally, the algorithm works well with image data [23]. Although it was proposed in 1963 it is still commonly used.

[3]In the case of multiclass classification, the problem is divided into multiple two-class problems.

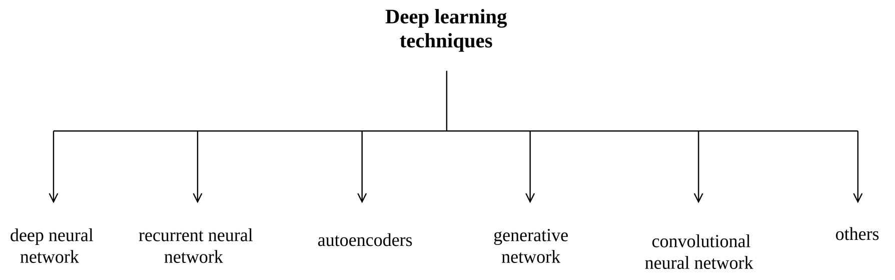


Figure 1.5: Examples of deep learning techniques that are considered in this work.

1.1.2 Deep learning

ML, as outlined above, has many advantages and it is commonly used for a wide variety of tasks and applications. However, its performance is closely related to the given data representation (i.e., features extracted), which may impact the performance of the final trained model. It is worth restating that ML algorithms need extracted features as an input, which is typically manually done, where a feature extraction method needs to be selected and the quality of these features influences the final model performance. To overcome the issue with the features extraction, DL method provides a solution to bypass this need. Instead of a time-consuming and sometimes manual feature extraction in the preprocessing stage, DL algorithms as part of their learning process are able to extract the necessary features and then use this information to complete the task. It is not uncommon for models that extract features automatically, to achieve much better results than those learned using extracted features from manual pre-processing. A good example would be image classification, where features may differ due to many aspects of the image itself, such as lighting or the angle at which the image was taken. In this case, DL tends to describe the complicated image representation more easily, for example using edges [24], than manual human extraction. The simplest example of DL is MLP which, assuming it has more than three layers, maps input through different layers to different representations in the output. It is worth mentioning that DL derives from the subfield of ML, which are the neural networks hand above all, compared to their shallow versions, they can represent nonlinear functions [25].

The rapid development of DL, and in particular its branch of computer vision, began when Alex Krizhevsky proposed a network called AlexNet in 2012 [26]. This structure is based on the use of convolution to extract features from an image and then, on their basis, to classify these features into particular groups with the help of fully-connected layers. This type of network is known as a *convolutional neural network* (CNN) and was already proposed in 1988 [27]. The CNN will be discussed in detail later in this subsection. Despite the earlier introduction, due to the lack of appropriate hardware and optimization methods, CNN's true potential was noticed only with the success of AlexNet. The AlexNet model achieved, at that time, the state-of-the-art accuracy in the ImageNet Large Scale Visual Recognition Challenge [25], thus, beating all the algorithms based on traditional ML and achieving over 10% lower error than the other models participating in this image-analysis competition. AlexNet was the first convolutional network model to win this challenge and sparked a great deal of interest and development in these kinds of algorithms. Although DL is not just about convolution-based networks, this event started an intense development of DL.

There are several different types of DL techniques. Some of the considered here are presented in Fig. 1.5. It is worth noting that DL can be used in all of the ML types, meaning supervised, unsupervised, semi-supervised, and also reinforcement learning. The first one presented is a *deep neural network* (DNN). The scheme of this network is presented in Fig. 1.6. The hidden layer vector, h, is calculated using Eq. 1.2 (example for hidden layer 1 from the scheme). w_i represents weights vector for all neurons in this layer and b_i is a bias.

$$h_1 = \sum(w_i x) + b_i \tag{1.2}$$

As it can be seen in Figures 1.2 and 1.6, the main difference between DNN and artificial neural networks (like MLP) is the number of hidden layers, which in the case of DNN is always greater than one. With an increase in the number of layers, the ACC of the model usually increases, whereas

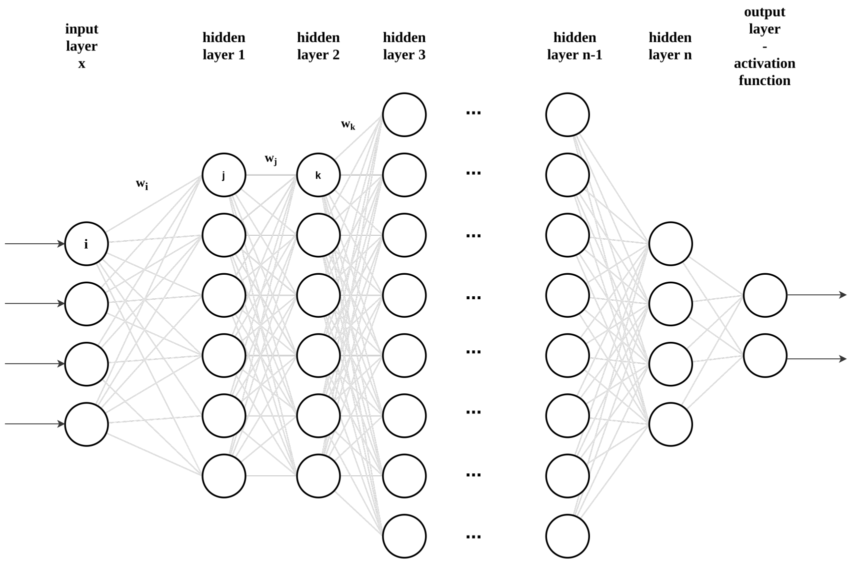


Figure 1.6: An illustrative example of the DNN algorithm. The input layer represents the feature vector provided to the model. The number of hidden layers and neurons in specific layers should be chosen during development. The output layer, most often, is an activation function that ultimately provides an output over a certain interval. For example, a sigmoid – in the range 0–1. w_i represents the weight of the i-th neuron.

its interpretability decreases along with it, so there is a problem with explaining why the model made such a decision. This issue is commonly raised in the discussion of DL.

The second considered DL technique (see Fig. 1.5) is a *recurrent neural network* (RNN). The major difference between DNN and RNN is that does not consider the input objects as an independent. In the case of an RNN, information about the previous inputs is also used during the analysis of subsequent objects. It can be said that they remember this previous information. This process is most often achieved with the use of feedback loops. Their main task is to work with sequential data such as text, speech, and time series, where consecutive data are dependent on each other, and information about neighboring objects (mostly previous) is valuable in the learning process. The *long short-term memory* (LSTM) [28] network is the most frequently presented example of RNN. It is a kind of RNN that solves the problem of long-term dependencies, which is an issue with its standard version. The scheme of the LSTM block, representing a single iteration,

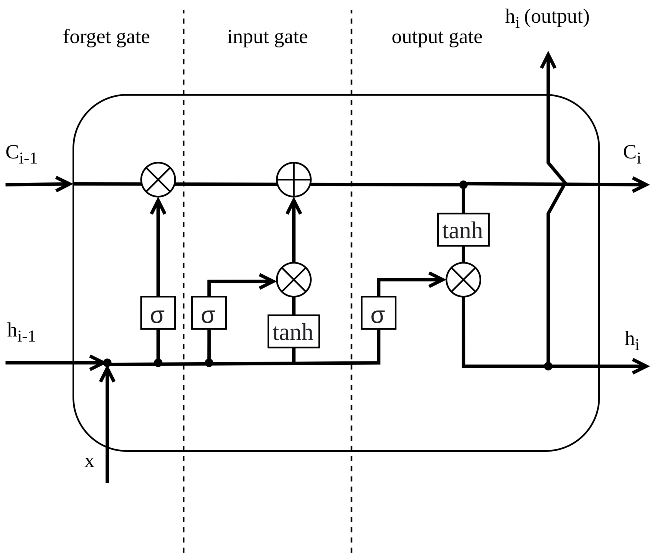


Figure 1.7: The schematic view of LSTM block. *x* represents the layer input. σ denotes a sigmoid function and *tanh* – the hyperbolic tangent activation function. *C* is a cell state while *h* is an output. The *h* at the top represents the layer output. Each function block (σ and *tanh*) represents a separate function with different parameters. The figure is based on Fig. 6 in [29].

is presented in Fig. 1.7. The first part (left side) of this block is called *forget gate layer*. By using a sigmoid function on combined information from the previous layer and the given input *x*, the output from zero to one is generated. Next, this value will be multiplied by the cell state (C). The value of zero means that all information from C is forgotten and the value of one means that all information is kept. The next step is the *input gate layer* which decides what information passes to the next step and is added to the current C value. The last part is to decide what will be passed as an output. The filtered, processed multiplication of C and the input will be the results of that. Additionally, in both the input and output gate the hyperbolic tangent function (tanh) is utilized that the values inside the block are in the range from -1 to 1. All this makes this architecture deal with long-term dependencies.

The third DL technique is *autoencoder* (AE) [15]. It is an example of the use of the neural network for unsupervised learning, and it represents the field of representation learning. The main purpose of this network

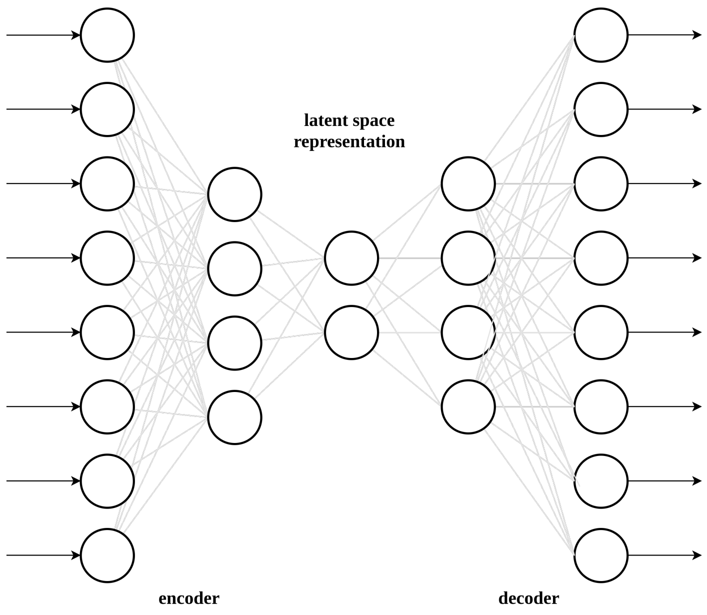
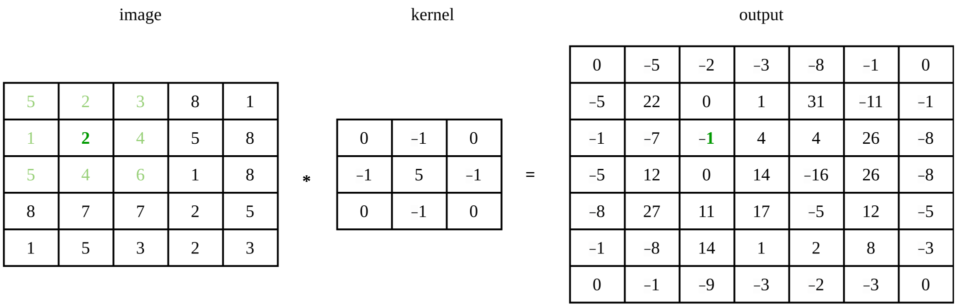


Figure 1.8: Schematic view of autoencoder architecture. On the left side, there are inputs, which are later processed by smaller and smaller layers of neurons and create a latent space representation in the smallest of them. Then this representation is applied to larger and larger layers of neurons until the initial number of neurons is obtained and thus the same output size.

is to calculate the function that will try to copy the input as an output through all layers. The schematic view of this network is shown in Fig. 1.8. It consists of two parts – encoder and decoder. Initially, the input data is encoded into a latent space representation (this step often compresses data into a smaller representation), then, the decoder is supposed to convert this representation into an output that would be as similar to the input as possible. Early studies used AE mainly as a tool for compressing data, reducing its dimensionality, or for denoising. However, their advantages gave rise to another type – *Variational Autoencoder* (VAE) [30], which belongs to the group of generative models.

The next technique is a *generative adversarial network* (GAN), which was proposed by Goodfellow and others in 2014 [31]. This technique belongs to generative models. In general, these kinds of models try to learn the representation of the given data, and then, use this to generate new samples. GANs are unsupervised techniques. They consist of two com-



$$\begin{array}{|c|c|c|c|c|}\hline 5 & 2 & 3 & 8 & 1 \\\hline 1 & \mathbf{2} & 4 & 5 & 8 \\\hline 5 & 4 & 6 & 1 & 8 \\\hline 8 & 7 & 7 & 2 & 5 \\\hline 1 & 5 & 3 & 2 & 3 \\\hline \end{array} * \begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\\hline -1 & 5 & -1 \\\hline 0 & -1 & 0 \\\hline \end{array} = \begin{array}{|c|c|c|c|c|c|c|}\hline 0 & -5 & -2 & -3 & -8 & -1 & 0 \\\hline -5 & 22 & 0 & 1 & 31 & -11 & -1 \\\hline -1 & -7 & \mathbf{-1} & 4 & 4 & 26 & -8 \\\hline -5 & 12 & 0 & 14 & -16 & 26 & -8 \\\hline -8 & 27 & 11 & 17 & -5 & 12 & -5 \\\hline -1 & -8 & 14 & 1 & 2 & 8 & -3 \\\hline 0 & -1 & -9 & -3 & -2 & -3 & 0 \\\hline \end{array}$$

Figure 1.9: Example of 2D convolution operation with 3×3 kernel. On the left is a two-dimensional matrix that is meant to represent the input image in this example. The kernel is in the middle. On the right, the result of the convolution is shown. The values in green on the left show the kernel's field of view for a single iteration and on the right is the result of it.

ponents: generator and discriminator which are trained at the same time. The first component is responsible for learning the data representation and, based on that knowledge, generates a new sample that would imitate the real one. The second component, the discriminator, is responsible for, kind of, classification – whether a given sample has been generated or comes from a training set. To be exact, it calculates the probability of whether a given object belongs to the given data representation. Both of these parts are adversarial to each other, meaning gain to each of them is a loss for the other. GANs are mostly utilized to generate images, and they are trained until the discriminator makes a lot of mistakes, which means that it will not distinguish the generated images from those coming from the training set. This technique has become popular in recent years and proved to be useful in many applications, for example, from generating additional training images to generating new human faces or even aging the original ones [32, 33].

Convolutional neural network

The last DL technique presented here is a *convolutional neural network* (CNN) [34], sometimes referred to as a convolutional network. This technique is presented here as the last one not accidentally. Given most of the research presented in this dissertation is based on this type of DL technique, it will be described in more detail. The premise of this method is that the input data has a grid-like topology [24]. Although it can be used for other non-image data, in practice, CNN is most often applied to images. It must consist of at least a convolutional part, and in most cases, it has also a part that is fully connected. This convolutional part distinguishes CNN from the standard NN. The basis of this technique is the convolution operation, and as it is the most important part of it, it is worth taking a look at. Definition of convolution is an integral operation on two given functions. In mathematical terms, it is presented in Eq. 1.3 and Eq. 1.4 for the one- and two-dimensional (e.g., image) case, respectively.

$$(f*g)(t) := \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau \tag{1.3}$$

$$(f*g)(s,t) := \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\tau_1,\tau_2)g(s-\tau_1,t-\tau_2)d\tau_1 d\tau_2 \tag{1.4}$$

In simple terms, in order to convolve one function with another, we need to get an integral of multiplication of this first function with the reversed and shifted version of the other in each possible point. It is worth mentioning that the result from $(f*g)$ is the same as $(g*f)$ because the convolution operation is commutative. Equation 1.4 corresponds to a well-known relationship between the input (image) and the impulse response (kernel) of a linear system. Fig. 1.9 shows the result of convolving sample 2D matrix (which aims to represent a region of an image) with a 3×3 kernel. However, in ML field, two subtypes of discrete convolution can be used: *valid* and *same*. An example of a valid convolution operation on an image data (2D matrix) is shown in Fig. 1.10. In terms of image processing, most often a given image is convolved by a kernel which is a small matrix used to enhance certain image characteristics (i.e., extract a feature), for example, edge detection or sharpening. In the example above, the commonly used 3×3 represents a sharpening kernel. As in the example, the valid convolution operation on the image results in a smaller image in the output, because the kernel is only shifted when it fits completely in the image. Sometimes however, zero padding is added to keep the original size of the image (see Fig. 1.11) and this is the *same* subtype of discrete convolution. Note

$$\overset{\text{image}}{\begin{array}{|c|c|c|c|c|}\hline 5 & 2 & 3 & 8 & 1 \\ \hline 1 & 2 & 4 & 5 & 8 \\ \hline 5 & 4 & 6 & 1 & 8 \\ \hline 8 & 7 & 7 & 2 & 5 \\ \hline 1 & 5 & 3 & 2 & 3 \\ \hline \end{array}} * \overset{\text{kernel}}{\begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 5 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}} = \overset{\text{output}}{\begin{array}{|c|c|c|}\hline -1 & 4 & 4 \\ \hline 0 & 14 & -16 \\ \hline 11 & 17 & -5 \\ \hline \end{array}}$$

Figure 1.10: Example of 2D *valid* convolution operation with 3×3 kernel.

$$\overset{\text{image + zero padding}}{\begin{array}{|c|c|c|c|c|c|c|}\hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 5 & 2 & 3 & 8 & 1 & 0 \\ \hline 0 & 1 & 2 & 4 & 5 & 8 & 0 \\ \hline 0 & 5 & 4 & 6 & 1 & 8 & 0 \\ \hline 0 & 8 & 7 & 7 & 2 & 5 & 0 \\ \hline 0 & 1 & 5 & 3 & 2 & 3 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline \end{array}} * \overset{\text{kernel}}{\begin{array}{|c|c|c|}\hline 0 & -1 & 0 \\ \hline -1 & 5 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array}} = \overset{\text{output}}{\begin{array}{|c|c|c|c|c|}\hline 22 & 0 & 1 & 31 & -11 \\ \hline -7 & -1 & 4 & 4 & 26 \\ \hline 12 & 0 & 14 & -16 & 26 \\ \hline 27 & 11 & 17 & -5 & 12 \\ \hline -8 & 14 & 1 & 2 & 8 \\ \hline \end{array}}$$

Figure 1.11: The same example that was presented in Fig 1.10 with additional zero padding that allows to keep the original spatial dimensionality of the image.

that even this *same* subtype is producing larger output it is still a truncated version of the convolution. The number of zeros added to the original image depends on the size of the kernel.

The presented operation of convolution on images is the primary one. However, there are a few other types of this operation, such as dilated, transposed, and a separable convolution [35, 36]. In the dilated one, the kernel's field of view is shifted by a value given as a parameter of this convolution. As it is presented in Fig. 1.10 (green numbers in the image) in the standard version of convolution, the kernel's field of view is equal to its size, while in the case of a separable convolution the operations are performed every k (where k is just a given integer parameter). A transposed convolution is often mistaken with deconvolution operation, but its principle of operation is different. The main point of transposed convolution is to reconstruct the spatial resolution of the original image – upsample the image. It is a method that, in some applications, can replace traditional interpolation methods. It does not revert the convolution operation, but in fact, it does a convolution with

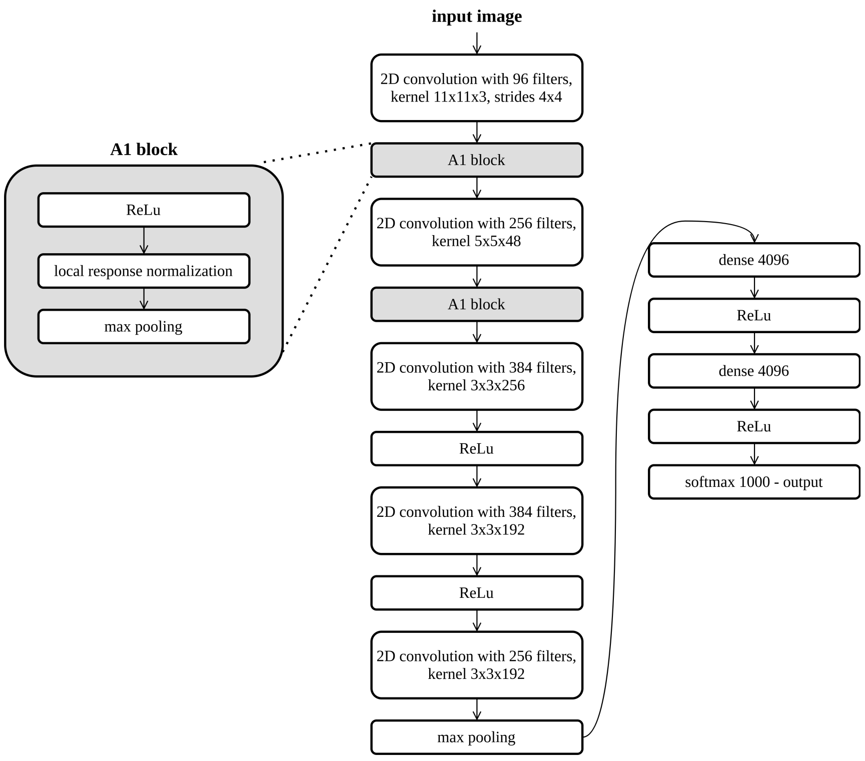

input image
2D convolution with 96 filters, kernel 11x11x3, strides 4x4
A1 block
2D convolution with 256 filters, kernel 5x5x48
A1 block
2D convolution with 384 filters, kernel 3x3x256
ReLu
2D convolution with 384 filters, kernel 3x3x192
ReLu
2D convolution with 256 filters, kernel 3x3x192
max pooling
dense 4096
ReLu
dense 4096
ReLu
softmax 1000 - output
A1 block
ReLu
local response normalization
max pooling

Figure 1.12: The schematic view of the AlexNet architecture. It consists of different layers: convolution, pooling, activation, normalization, and dense. ReLu – rectified linear unit, softmax – normalized exponential function, dense - fully connected neural layer.

additional zero padding in an input. Transposed convolution is often used in the Encoder-Decoder architectures.[4] The last type is the separable convolution which in simple words splits the convolution operation into parts, for example, 2D convolution with two 1D convolutions with possible different kernels. In DL, it is used in depthwise separable convolution. The main advantage of this method is that it sometimes allows reducing the number of parameters while maintaining the same performance [37].

While explaining the concept of convolution, it is worth noting that convolution layers consist of several kernels, and their values are selected automatically in the optimization process. As was mentioned at the beginning of this subsection, the development of DL techniques began with

[4]It is a transposed convolution that has been applied in the fully convolutional networks in this dissertation (see chapter 2)

the AlexNet architecture (Fig 1.12), and based on that the concept of CNN will be described. The architecture contains a several different layers. The most important one is a convolutional layer, where the convolution operation is performed. It could be formed into a convolutional block meaning more than one layer following each other without pooling operation, which will be described shortly (the third, fourth, and fifth convolutional layer in Fig. 1.12). Additionally, each convolutional or fully connected layer is accompanied by an activation layer, most often by a rectified linear unit (ReLu), which introduces the non-linearity into a model. The output of this sequence after the first and second convolutional layer is followed by local response normalization. After this block (or single layer with activation), there is a pooling layer. The purpose of which is to reduce the spatial dimensionality of the convolutional output – future maps. The most popular types are maximum and average pooling. The sequence of convolutional and pooling layers could be repeated as needed. After this, feature maps are flattened into a single vector and transferred into a fully connected block (dense layer) of the CNN architecture. It works as a standard neural network, where each value in the input vector is connected to each neuron in the fully connected layer. The number of neurons and layers in this part could be chosen depending on the problem. In AlexNet there are two fully connected layers with 4096 neurons each. At the end of the network, an activation function is used, such as softmax [5] or sigmoid functions, to get the expected output. In the case of this specific architecture, there are 1000 units as an output, and this is equal to the number of classes that the input images can be classified into. Nowadays, to avoid overfitting, a dropout layer is often added between fully connected layers, which in the training process randomly omits a given percentage of units in the network. Additionally, in 2015 another improvement for CNN was made with a batch normalization [38]. This layer is used for stabilizing the training process and to reduce the required number of epochs by normalizing its input before transferring it to another layer. It is usually added after the activation layer and follows the convolutional one. It is worth adding here that due to the presence

[5]The softmax function is also known as the normalized exponential function. As the output of this function, a vector that sums to one is received.

of many different layers, the number of parameters for training this type of network is usually very high. In the case of AlexNet, it is about 60 million of parameters. Because of that, there is a need to have a relatively large training set to obtain non-random results or to avoid overfitting. This issue of a large dataset will be discussed in the low-data regime subsection below. Described above layers are the most basic CNN ones, but over time, some additional techniques have been presented to improve performance, for example, residual learning in ResNet types of architectures [39].

The presented type of network is not the only one in the group of convolutional networks. Another example of commonly used one that is based on convolution operation is semantic segmentation with a *fully convolutional network* (FCN) [40]. Image segmentation is the process of grouping parts of an image based on their similarities. The semantic kind of segmentation is achieved by classifying each image pixel into a specific group/class. This method will be described based on the U-net architecture because this type of FCN is commonly used in biomedical applications [41]. The principle of this architecture is similar to that of autoencoders with additional skip connections. It consists of encoder and decoder paths except they are built from convolutional layers, and are named: contracting and expansive paths, respectively. The U-Net architecture is presented in Fig. 1.13. The left part of it is a contracting path, which is similar to the convolutional part of the previously described CNN. It is a sequence of convolutional blocks with ReLu activation and pooling operations. The right part, the expansive path, strives to increase the size of feature maps with the sequence of convolutional layers with ReLu and also transposed convolution layers. The latters are utilized for upsampling of the feature maps. After that, its output is concatenated with the corresponding feature map from the encoder part (cropped one). In the end, there is a segmentation map, but its size is smaller than that of the original image, and this is because of this architecture use convolutions without padding. U-Net-like architecture has proved very useful in the biomedical field. The results obtained using it significantly exceeded the results obtained by other proposed methods in the ISBI cell tracking challenge 2015 [41]. Currently, various modifications of this

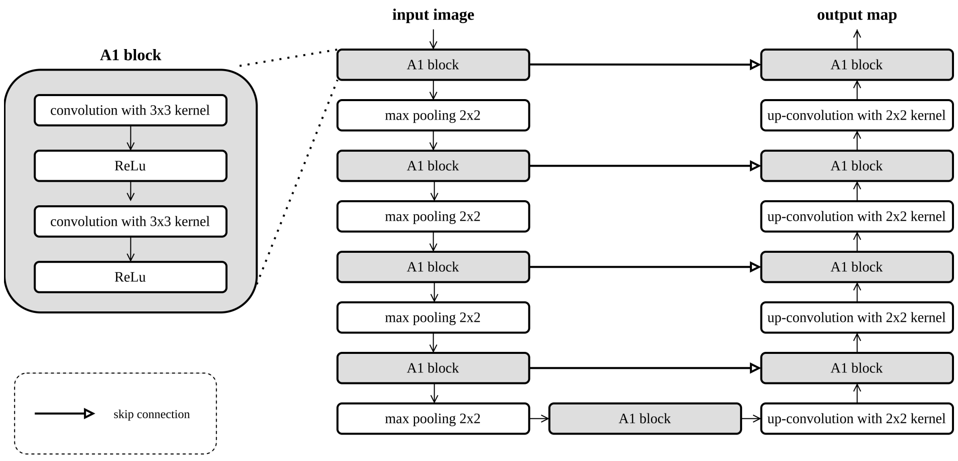

input image
output map
A1 block
convolution with 3x3 kernel
ReLu
convolution with 3x3 kernel
ReLu
A1 block
max pooling 2x2
up-convolution with 2x2 kernel
skip connection

Figure 1.13: Schematic view of the U-Net architecture, which is a fully convolutional network. On the left side, there is its encoder part (also called as contracting path), in which the size of the given image is reduced by various convolution and pooling operations. On the right is the decoder part (expansive path) where the original image size is restored. Additionally, there are skip connections between these parts. The figure is based on Fig. 1 in [41].

method are available, and they are commonly used for image segmentation [42, 43]. A particular advantage of the U-Net architecture is that it does not require a large dataset to obtain satisfactory results.

Additionally, all of those DL and ML techniques can be combined with each other. For example, for U-Net like architecture, a RNN block could be added.

1.1.3 Quality assessment metrics used in machine learning

An important topic in ML is the metrics that are used to assess the quality of models. These metrics differ depending on the selected problem, and they are typically different for classification and segmentation tasks. Consequently, their choice should be closely related to the problem under consideration. Furthermore, the quality assessment of ML models should always be performed on a dataset that the model has not been exposed to during the learning process (i.e., training phase), which could be both a validation set, in the case of cross-validation technique, as well as a test

predicted true	positive	negative
positive	TP	FN
negative	FP	TN

Figure 1.14: An example of a confusion matrix for a binary classification task. TP – true positive, TN – true negative, FN – false negative, FP – false positive.

set. Given this dissertation's main focus is placed on segmentation and classification tasks, metrics that are commonly utilized for them in a supervised learning context will be explained. One issue to start with, as many other metrics rely on this, is the *confusion matrix* (sometimes referred to as the error matrix) [44]. It is known as a table, based on which the performance of the model can be assessed. The confusion matrix can be calculated only when the true labels are known, therefore it is only suitable in supervised learning. Confusion matrix is shown in Fig. 1.14. One can assume, that there is a binary classification task to be performed. For example, the model is designed to detect glaucoma. The dataset on which the model is taught is irrelevant in this matter. The model's response is set to 1 (positive) if it has detected glaucoma in a given sample, and 0 (negative) if it presumes that the given sample belongs to the control set (no glaucoma). The confusion matrix will assign the response of the model to one of four categories depending on the relationship of the response with respect to the known true value. The categories are *true positive* (TP), *true negative* (TN), *false positive* (FP), and *false negative* (FN). The response will be assigned as TP when both the model and the true label indicate glaucoma in this given example (in general – positive class), to FP when a label will indicate control group (negative class) and the model – glaucoma (positive group in general). It is similar in the case of the negative class, TN will be if both the model and a true label will indicate the negative class, whereas FN will be if the model will indicate the negative class, and the positive class will be true. It is obvious then that a higher quality model will strive to minimize both FP and FN. Ideally, they will both be zero. However, when it comes to practical cases, depending on the application, it is sometimes preferable to minimize mainly one of these values. For example, in medical screening, it is better to get a false-positive result and do more tests on a patient (however, it may be more expensive) than to get a false-negative result and leave the disease undiagnosed (this can greatly affect the patient's health). Not only raw values are important, but also their ratios on the basis of which other metrics are calculated. Here, a few of these will be presented, including *true positive rate* (TPR), *true negative rate* (TNR), *accuracy* (ACC), and *balanced accuracy* (BAC). Equations for all of them are listen below (Eq. 1.5, 1.6, 1.7, 1.8).

$$TPR = \frac{TP}{TP + FN} \tag{1.5}$$

$$TNR = \frac{TN}{TN + FP} \tag{1.6}$$

$$ACC = \frac{TP + TN}{TP + FP + TN + FN} \tag{1.7}$$

$$BAC = \frac{TPR + TNR}{2} \tag{1.8}$$

TPR, also called sensitivity, is a metric that indicates the probability that if the model determines the positive class, then the object it classifies will belong to this positive class. The same is the case with the TNR, only here the probability of the negative class is being checked. TNR is often referred to as specificity. However, the most commonly used metric in this work is accuracy. By maximizing its value, the learning process of many models is being optimized. Accuracy is calculated as the ratio of correctly qualified examples (both from the positive and negative class) to all examples from the set. However, this method of calculating accuracy is only effective if one has a balanced dataset.[6] In a case where one class has many more examples

[6]A balanced dataset is a dataset that contains an equal or almost equal number of samples in each considered class.

in the set, the accuracy metric might be deceitful. Therefore, in such cases, BAC expressed as an arithmetic mean between sensitivity and specificity, is used 1.8.

In the case of a segmentation task, it is also possible to use the above-mentioned metrics. For example, the segmentation accuracy is referred to as pixel accuracy and treats each pixel in the image as a separate sample for classification, based on the ratio of correctly classified pixels to ground truth. However, due to the spatial nature of the result of this task, it may be better to look at more appropriate metrics like *Jaccard similarity coefficient* (JSC) or *Dice similarity coefficient* (DSC) (see Eq. 1.9, 1.10). Both these metrics, although slightly different, are used to measure the similarity of two sets, i.e., in the case of image segmentation - the similarity between the result of segmentation by the model and the ground truth.

$$JSC = \frac{TP}{TP + FP + FN} \tag{1.9}$$

$$DSC = \frac{2TP}{2TP + FP + FN} \tag{1.10}$$

Sometimes, the specific point or points at the image need to be segmented and in this case standard error metrics like mean absolute error, root mean square error or mean error could be utilized. Additionally, there are other methods to evaluate the quality of models in ML, such as segmentation errors or area errors, but those are beyond the scope of this work.

1.2 Low-data regime

In the DL problem, it is generally accepted that there is a positive correlation between the amount of data and the performance of the network. This could be a major challenge, particularly in the medical image analysis field since big datasets are usually unavailable or hard to capture and annotate.[7] The me-

[7]In general, big data are datasets that are too large, too complex, or too demanding that they could not be handled by traditionally image analysis methods [45]. Here, a big dataset is a set whose size is sufficient to train the model without overfitting, and this set represents well a given problem.

dian number of medical images used in various studies in the years 2011-2018 was assessed in [46]. These images were acquired using two different imaging modalities: *Computed Tomography* (CT) and *Magnetic Resonance Imaging* (MRI). The conclusion of this study shows that the size of datasets increases year by year, but its median value still does not exceed 200 images, which in the case of DL is considered a small dataset. Therefore, different techniques to deal with the low-data regime were proposed, and they will be briefly described here.

The most common problem to deal with, when there is a lack of data or when the chosen model is too complex in comparison to the data, is overfitting. It refers to a situation when the model corresponds too closely to training data and does not generalize well, i.e., to the information existing there, as well as to noise (which is random, so the model should not describe it). In other words, the overfitted model has good training accuracy, but the validation or testing one is much worse, or sometimes even random. This overfitting negatively affects the overall generalization of the model and thus reduces the model's ability to recognize new patterns. And this ability is the essence of using ML models in real life. On the other side, there is a problem called underfitting. It occurs when a model is too simple (has too low complexity) for a given problem, and thus the model is not able to capture the appropriate relationship between the inputs and their corresponding labels. Therefore, problems can be noticed both on the results of the training set and the validation/test set.

1.2.1 The curse of dimensionality

There is another issue related to the shape of the dataset.[8] As explained before, it is better to have a larger dataset than a smaller one. Additionally, the dataset should represent the distribution of the given problem. However, sometimes for each training set, there is a large number of variables explaining it. Let's say that n is the size of the dataset and p is the number of features for each training object. If n is relatively small and there is a lot

[8]The shape of a dataset is understood as both the number of learning objects and the number of features that define each learning object, for example, its size.

of different features in p a phenomenon called a *curse of dimensionality* arises [47]. One way to explain it is to think about these features, each one as a single dimension. With an increase in the number of them, the number of dimensions increases as well. If one takes a similarity as a distance (Euclidean, for example), then this distance will also increase and the object will be less similar than with a smaller feature number. Because of that, distinguishing objects belonging to different classes is more challenging [48]. Several methods have been developed to help deal with this issue. They can be divided into two main strategies as presented in Fig. 1.15, which include: (i) to reduce the dimensionality of the feature space or (ii) to use ensemble learning to help deal with high-dimensional data. The first one focuses mostly on methods for feature selection, for example, selecting the subset of features based on some statistical method that will rank them. Another example of this first strategy could be the use of Principal Component Analysis [49]. It is a statistical method that, through the appropriate transformation of coordinates space, aims to maximize the variance of the first components so that it is possible to reduce the dimensionality with the least possible loss of information contained in the original data. The first strategy is effective in most cases but sometimes, with the rejection of features, the information that is hidden in these features is lost. And because of that, the second strategy was developed, which includes the usage of ensemble techniques to train different classifiers on a different set of features. This leads, in simple words, to the fact that each base classifier receives data of lower dimensionality avoiding a curse of dimensionality. At the same time, looking at the whole ensemble, all or most of the features are being used so no information is lost. The most popular method in this strategy is Random Subspace and this is also called feature bagging [50].

1.2.2 Methods to deal with low-data regime

Besides the methods for dealing with a curse of dimensionality, several methods are useful when there is a need to train DL algorithms in a low-data regime. The first of them is a dropout layer with one important parameter – rate. This layer is responsible for randomly omitting units or connections

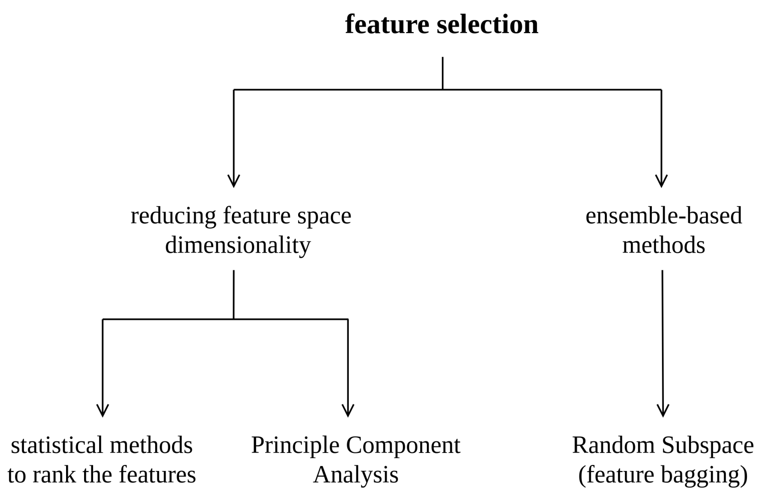

feature selection
reducing feature space dimensionality
ensemble-based methods
statistical methods to rank the features
Principle Component Analysis
Random Subspace (feature bagging)

Figure 1.15: Overview of the basic feature selection methods. Shows the division with examples of algorithms.

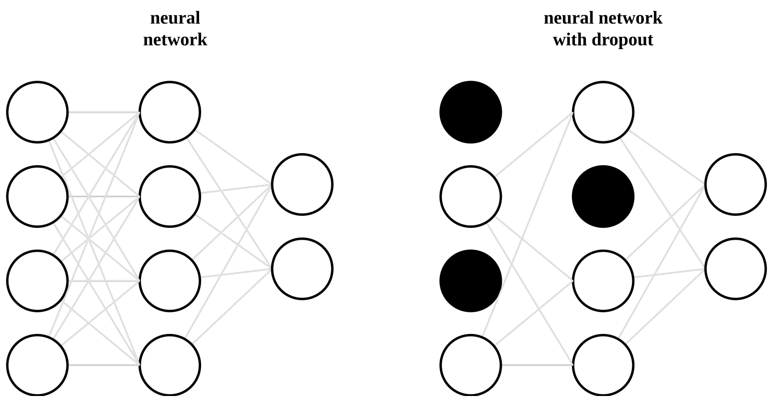


Figure 1.16: Example of a dropout operation. On the left side, there is a neural network and on the right side is a neural network with the use of dropout operation.

during training. The number of units that will be dropped out is determined by the rate parameter given as a percentage and could be different for each model layer. Such action prevents the network from being overly matched to the training set and thus is an effective tool to reduce overfitting. Fig. 1.16 shows how a dropout with skipped units is performed. In the first layer, 50% of units are omitted, and in the second – only 25%.

Another technique for dealing with a low-data regime is called data augmentation. Taking into account this dissertation which focuses on medical images, its subtype called image augmentation is presented here. Image augmentation is utilized when training convolutional networks and it is possible due to the main assumption of these networks, which is the independence from spatial shifts. This technique is used to increase the size of training data by adding more samples. These additional objects are based on modifications of existing ones. Data augmentation can be performed for both single and multidimensional data and the transformations performed should be carefully selected based on the dataset analysis, because it is important to remember that the training set still has to represent the problem. An illustrative example of image augmentation is a dataset consisting of images of a cup and the task is to classify whether the image contains the cup or not. To increase the available dataset, a series of subtle transformations can be made to change the position of the cup within the image, such as random rotation or translation. Still, the cup will be present in this new image, and the answer of the network should be the same. Thus, it is possible to increase the dataset using modified copies of existing images without changing their labels so that the network has some additional data to train on. In the context of images, Fig. 1.17 collects transformations that are commonly applied to images [51]. Sample results of geometric augmentation transformations (that can be combined) including rotations, zooming, translations and reflections are shown in Fig. 1.18. The example is based on the image of a coffee cup and shows that using data augmentation the core information from the image is not changed. Additionally, there is another subgroup of data augmentation techniques called data generating. In contrast to those presented above, it does not consist in modifying the data, but in generating synthetic data on the basis of existing ones. This type could be realized by mixing existing images [52] or, what is mostly used, by using GANs to generate additional training data.

The third method to deal with a low-data regime is *transfer learning*, which should be considered to be strictly related to it – *fine-tuning*. In transfer learning, the model is trained on another, larger dataset that is somehow related to the problem, and this knowledge is then used on the considered dataset [53]. It is similar to the human learning process. Often when explaining a more advanced topic people try to understand it based on its similarity

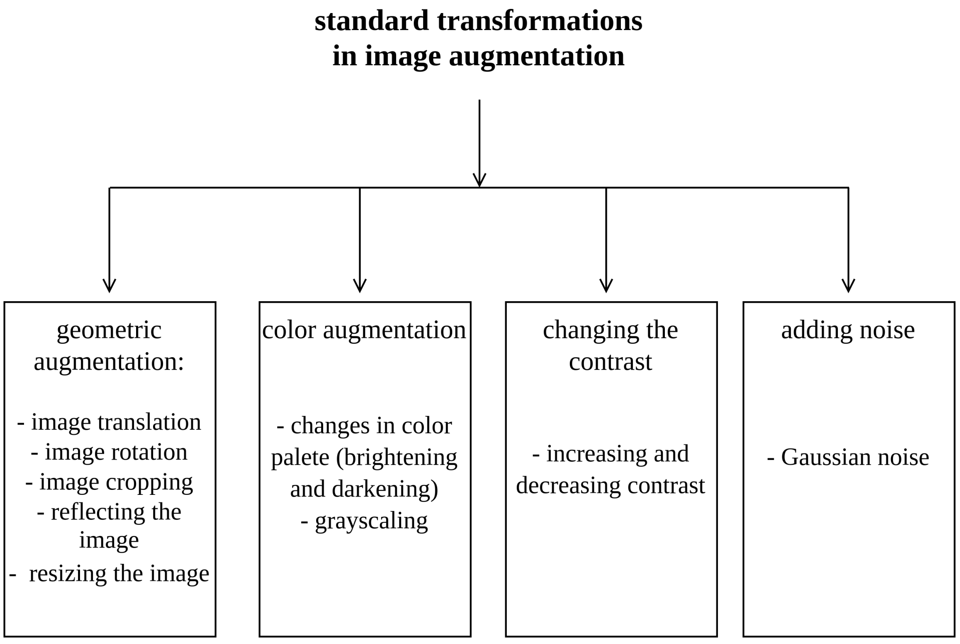


Figure 1.17: Different types of image augmentation. It presents the division and specific examples.

to another, known subject. For example, when someone wants to learn a new programming language it is easier when he/she has some previous programming experience and they can transfer that knowledge to a new language. The process is similar in ML. In the case of the convolutional network at the beginning of its first part, which is the convolutional part, the network is able to recognize corners, edges, etc. This could be used in almost every image recognition task, therefore, this knowledge could be easily transferred from one task to another. Subsequent layers can detect more advanced objects and this knowledge would be only helpful if these objects are relevant to the considered problem as well. In case when a sufficiently large enough dataset is difficult to obtain or it is very expensive, it is possible to use transfer learning using a dataset from a similar domain. Usually, the application of transfer learning tens to improve or at least will not worsen the network performance. Occasionally, however, a negative transfer may occur, this is a process by which knowledge gained in one domain negatively impacts the results if it tries to transfer it to another.

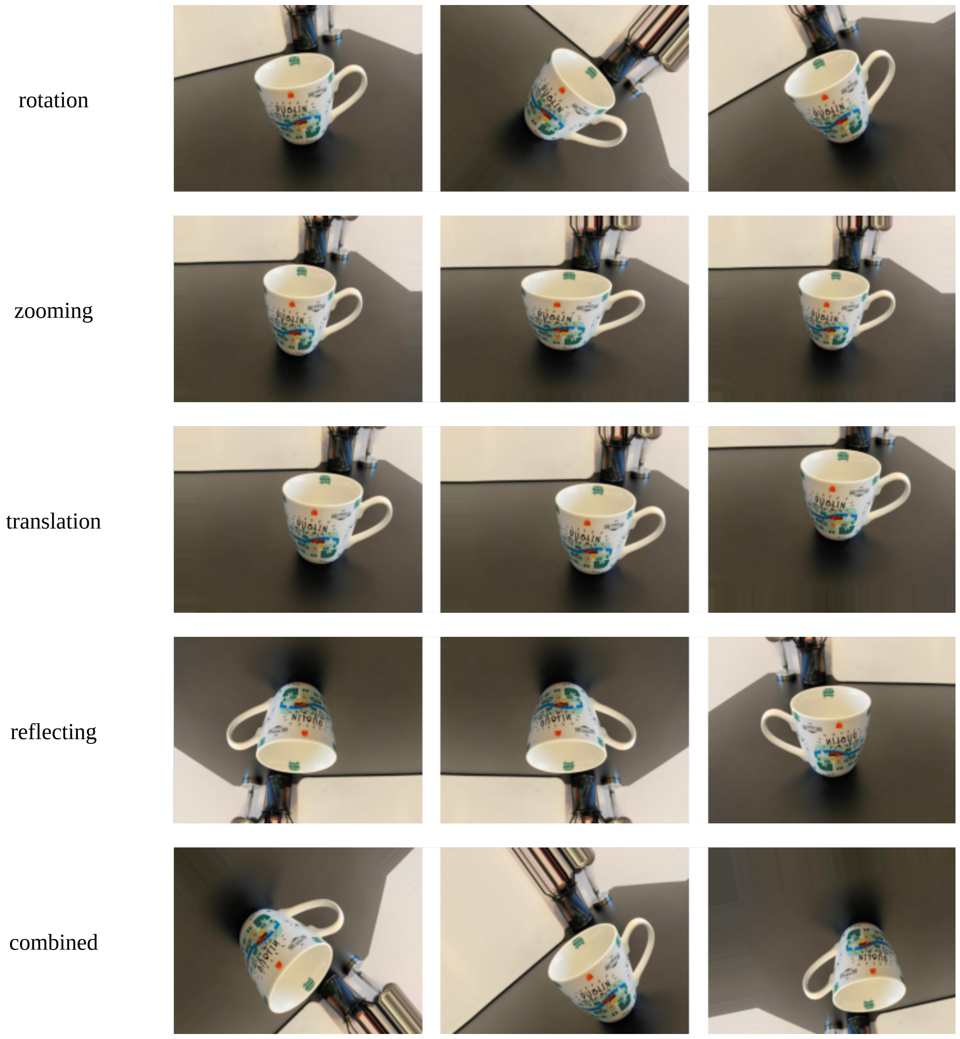


Figure 1.18: Sample results of using geometric transformation on the image of a cup.

As mentioned at the beginning of this paragraph, transfer learning is strictly related to fine-tuning. The model that is trained on one task can be fine-tuned to another, which means that all, or some layers could be retrained on this smaller dataset. This process will allow the model to adjust to a new task by adjusting previously established, on another task, weights.

A reduction in the complexity of the model is another technique that could be considered in order to improve the accuracy of the model trained on a small dataset. More complex models need more training data, so reducing the model complexity could be sufficient to obtain higher accuracy even

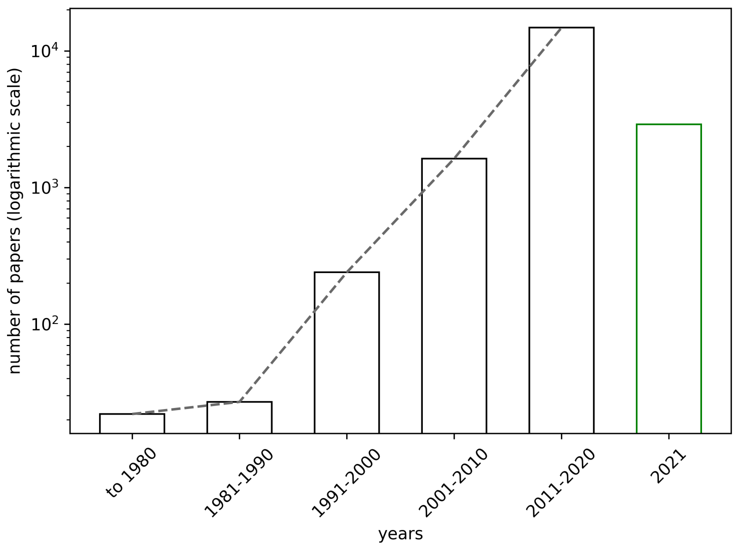


Figure 1.19: A graph showing the number of published research papers containing keywords (in the title or text) ML and ophthalmology in time intervals based on a Google Scholar search.

on the test set, since it prevents overfitting. Additionally, ensemble learning as mentioned in the curse of dimensionality section could be a good approach to dealing with small datasets. Also, a properly chosen normalization and regularization can decide the quality of the training model in a low-data regime. It needs to be emphasized that there is no universal method that can be applied to all low-data regime cases. Therefore, it is worth remembering that when creating a model in a situation where the amount of data is small, it is important to test several solutions and check which one provides the optimal performance for a given problem. Sometimes even the rejection of outliers can bring a significant improvement.

1.3 ML application in ophthalmology

In this subsection, the current state of application of ML in the field of ophthalmology will be presented. It will be divided according to the type of task into three sections - classification, segmentation, and feature selection. However, before the current state of knowledge for individual studies in ophthalmology is presented, it is worth taking a look at the total number of published scientific articles on this field at different time intervals. This is shown in Fig. 1.19, which presents results from a *google scholar* search engine.[9] Two phrases were searched: *ML* and *ophthalmology*. Due to an indexing error for JAMA journals, they were excluded from the search, since the search for this journal would result in a larger number of studies that were not relevant to the topic. Concluding from the graph, the number of studies, and therefore the interest in the use of ML in ophthalmology, is still growing.[10] Additionally, in 2021 alone (until December 28), a total of 2900 studies on this field have been already published. This interest comes both from the need to support the diagnosis of various eye diseases with ML and also from the success of these algorithms in the field so far. Several commercial devices for imaging eye structures already use these algorithms, which only further encourages the development of this field.

1.3.1 Classification

The task of classification in ophthalmology employing ML methods has been present in the scientific literature for many years. With the help of these algorithms, the authors try to support the automatic diagnosis of eye diseases, such as diabetic retinopathy, glaucoma, age-related macular degeneration, and many others. For glaucoma, as mentioned earlier, since the beginning of ML as a field, efforts were made to apply these techniques to aid diagnostics. This could be due to the fact that glaucoma progresses asymptomatically for many years, and it is usually detected too late. Therefore, the screening methods would make it possible to detect this disease at an earlier stage, and thus could have a great benefit for patients. When ML is used to diagnose glaucoma or other eye diseases, the subjects whose data are then processed are assigned (labeled) to the appropriate group by experienced ophthalmologists, and the algorithms are trained based on these data. Initially, numerical data was used for classification. For example, the output of the visual field test from standard automated perimetry was used with mul-

[9]https://scholar.google.com

[10]This trend probably does not differ from other fields of science in which ML is used.

tiple ML algorithms and compared with standard statistical metrics showing promising results [54]. Additionally, parameters from a standard Heidelberg Retina Tomograph of about 200 people in total were used to train bagged classification trees and showed a decrease in classification error compared to other classical methods [55]. Instead of numerical data alone, images from various imaging modalities could be used to classify eye diseases, such as fundus cameras, *Scanning Laser Ophthalmoscopy* (SLO), or *Optical Coherence Tomography* (OCT). Initially, before the extensive use of DL, standard ML algorithms were used to classify images, to which the features previously extracted in the preprocessing stage were fed. However, such a preprocessing step added computational overhead and its execution and feature selection was critical in the final quality of the classifier obtained. Attempts were made to use, e.g., texture parameters [56–58] or independent component analysis [59]. However, the development of DL has dominated the task of classifying eye diseases, while other techniques have received little attention. In recent years, it is the work based on these DL algorithms that have been the common standard. The use of convolutional networks allowed avoiding manual feature extraction in the preprocessing stage [60,61]. Both fundus camera and OCT images are used to detect glaucoma. When it comes to fundus images, the studies very often compared the use of traditional ML algorithms with different DL network architectures [62–64]. The results confirm the advantage of using DL over manually extracted features. In the case of OCT, there are many different methods of handling this data (in glaucoma detection images from the posterior segment of the eye are utilized). Due to the fact that OCT scans create a multidimensional structure, some algorithms only use 2D cross-sectional images of the retina, and some use all the information – 3D volumetric data, thus increasing the computation time. Although there are many applications of OCT imaging to support glaucoma diagnostic, most of them have to use some low-data regime techniques to deal with the small data size, for example, data augmentation and transfer learning [65–67]. Table 1.1 presents selected publications along with the size of the dataset that was used in the particular study. It is apparent that the amount of data used to detect glaucoma is increasing (espe-

Table 1.1: A list of selected articles on the classification of glaucoma, detailing the size of the data used. FI - fundus images; SLM - scanning lases microscope, HF - handcrafted features.

	data	algorithm	dataset size
Samanta et al. [56]	FI	ANN	455
Yadav et al. [57]	FI	ANN	20
Dua et al. [58]	FI	ML algorithms	60
Fink et al. [59]	SLM	KNN	120
Al-Bander et al. [60]	FI	CNN + SVM	455
Chen et al. [61]	FI	5 CNN	650, 1676
Ahn et al. [62]	FI	CNN	1542
Asaoka et al. [63]	FI	CNN	ca. 3000
An et al. [65]	FI	CNN	ca. 3000
An et al. [67]	OCT scans	hierarchical CNN	954
Mehta et al. [68]	OCT scans + FI	two CNN	2476
Benzenouchi et al. [69]	FI + HF	SVM + CNN	650, 1676

cially for fundus camera images) but still not enough to easily train complex convolutional networks that have many parameters. Hence, methods to deal with the low-data regime are needed.

Increasingly, instead of using data from one imaging modality, algorithms operating on multi-modal data are also used [68–70]. In addition, in many of the above studies, instead of focusing only on the separation between the control group and the group of subjects suffering from glaucoma, a third group is also considered - subjects suspected of having glaucoma either because of the glaucomatous optic disc appearance or ocular hypertension. This is more challenging, but it could help detect the disease before clinical symptoms, such as visual field narrowing, occur. Thus, it is very valuable from a clinical perspective.

1.3.2 Segmentation

The segmentation task is also widely used in ophthalmic applications. As with classification, it is used to understand the health of the eye and can aid in the diagnosis of eye diseases, such as glaucoma, diabetic retinopathy, and Age-related macular degeneration (AMD). Images from fundus cameras and OCT are most often used to perform this task. At the beginning of the

2000s, solutions based on mathematical operations and standard image processing techniques were utilized. Unfortunately, they were associated with the need to perform, sometimes, even multi-stage pre-processing to prepare the images for further processing. Fundus images were used, for example, for optic disc segmentation based mostly on the detection of edges and Hough transformation [71], microaneurysms segmentation for the diagnosis of diabetic retinopathy using generalized eigenvectors of affinity matrix [72], drusen segmentation for the detection of the first symptoms of AMD using morphological operations [73], or vessel segmentation also for the diagnosis of diabetic retinopathy using wavelets, or various morphological techniques, filtering, and thresholding [74, 75]. In [75], which was published in 2004, a slightly different approach was introduced. In addition to comparing more classical methods, it also performs pixel classification for this task using automatically assigned attributes for each pixel and classification using standard ML algorithms. It can be seen that already in the early 2000s there was an interest in using ML to perform image segmentation. As for OCT images, they began to be used especially for the segmentation of intra-retinal layers for glaucoma diagnosis. For this purpose, methods based on, for example, adaptive thresholding or active contours were used [76, 77]. It is worth noting here that often to use such methods, speckles were removed from the images. In later years, ML methods started to increasingly replace classic segmentation techniques. In [78], authors presented a ML application for retinal layer segmentation of OCT 3D scans. They proposed a method that consists of two main steps: (i) processing to compute features for each pixel based on its neighborhood, including Haar-like features, and based on them (ii) classifying using an ML SVM algorithm. An interesting solution used is also the use of information from the entire A-scan to count the features for a single pixel, taking into account information from neighboring pixels. Not only retinal layers are segmented from OCT volumes. Also, the Bruch's membrane opening, which can be used for the diagnosis of glaucoma [79, 80] is an important task that is sometimes solved using ML. In [81], the combination of graph-theoretic approach with ML random forest algorithm is utilized to perform this task. For fundus images, ML algorithms were utilized for example to predict AMD progression using biomarkers and ML algorithms as classifiers [82]. The biomarkers used were derived from the segmentation of the retinal structures. As can be seen, most of the similar works use ML algorithms to classify previously learned features [83, 84] to specific segments. As with the classification task, DL is used to automate the process. For the segmentation task with the use of DL, networks based on the convolution operation are most often used. In [85] the application of CNN to segment various types of fluids from OCT scans is presented. In the field of medical image segmentation, approaches based on FCN have become increasingly popular, especially architecture like U-Net [86–88]. Currently, research is also conducted on the use of image patches in comparison to the use of the entire image [89, 90], which could be also considered as a strategy for low-data regime problems, since the original sample is subdivided into many samples. There are also solutions combining the use of U-Net and CNN architecture [91].

1.3.3 Feature selection

Although most work on ophthalmic applications today focuses on image analysis, there are also some applications that rely on numerical values. These numerical values may be the format in which the instrument collects the data (visual fields) or they could represent features that are extracted from images and classification is carried out on their basis, as has already been discussed above in the classification section. At the same time, it should be remembered that if the number of features is too large and the number of learning objects (i.e., the size of the dataset) is small, it may adversely affect the final result of the ML algorithm. For this purpose, feature selection methods are used to select the optimal subspace of features. For example, in [92] to perform the task of classifying subjects with open-angle glaucoma into four groups of different optic disc morphology, the *Minimum Redundancy Maximum Relevance* (mRMR) [93] with SVM and Naïve-Bayes algorithms were used for feature selection. The same method, mRMR, has been used to improve the glaucoma detection alone [94]. Genetic algorithms have also found application, not as the main method but as a single step in the classification of glaucoma [95]. Another solution is proposed for the assessment of the angle-closure glaucoma mechanisms [96]. There, for feature selection, various algorithms are used to rank the features, and then, a feature selection is done using the Rank Combination method. Results indicate that the combination of ranks from different algorithms allows for improving the final quality of the classifier (in this case AdaBoost) compared to the results of single feature selection algorithms. Additionally, some works combine feature selection with DL. In [97], for the detection of diabetic retinopathy, a combination of two algorithms was used. Initially, features are extracted from the image based on preprocessing. Then a new algorithm, named Modified Gear and Steering-based Rider Optimization Algorithm, is proposed and used to select the optimal features. At the end of this proposed method, a *deep belief network* [98] is used for detection. The results show that the combination of the two methods significantly improves the accuracy of the detection. Although the works using feature selection in ophthalmology are not very popular, the ones that have been presented show that this is a very important issue when it comes to classification and the use of these algorithms allows one to improve the quality of models without the need to provide new data, that in many cases are limited.

Chapter 2

Research summary

In this chapter, the research that was carried out during the Ph.D. will be presented. Although these studies concern various problems in ophthalmology, such as the classification of glaucoma or segmentation of retinal structures, they have one thing in common: low-data regime. The small set of available data shaped the proposed final algorithms used to solve the image analysis task. Different methods for dealing with the low-data regime are tested and the effectiveness of the methods on such datasets is demonstrated. According to the convection used in the Introduction section, this chapter is divided according to the main task of the algorithm, which included: classification, segmentation, feature selection, and a general method of dealing with the low-data regime. The main purpose of these studies was, above all, to demonstrate that even on a small local dataset it is possible to build a high-quality model to support the diagnosis of eye diseases.

2.1 Glaucoma classification

During the early stage of the doctorate, it was decided to start the research with the classification of *primary open-angle glaucoma* (POAG) based on image data. Before commencing the study, mainly fundus camera images or OCT scans were used for this purpose (considering the available literature). However, when scanning with certain OCT models, in addition to the cross-sectional OCT scans, SLO images are also captured. These images show a ‘front-on’ image of the back of the eye, the retina, and are centered around the optic nerve. SLOs are mainly used for image positioning and to facilitate follow-up examinations. Fig. 2.1 shows an example of OCT output. The images considered here are acquired by a spectral domain OCT instrument from Spectralis (Heidelberg Engineering, Heidelberg, Germany). On the left side of the figure, the SLO image with a green arrow pointing to where the OCT scan is captured on the eye (right side). In a situation where the task is to support the diagnosis of some diseases, the final quality of the ML model and its reliability are of great importance. Therefore, it is important to use all the available data that contains relevant information about the particular disease. For that reason, the study focused on ascertaining whether these SLO images contain sufficient information to support the diagnosis of POAG. A substantial part of this research was published in [2]. It is worth noting, that at the time of writing this dissertation, there were already applications of SLO images in supporting the detection of eye diseases, for example in multimodal solutions [70]. However, the research carried out in this dissertation uses only SLO images to directly indicate their usefulness in supporting diagnostics.

2.1.1 Dataset and preprocessing

A retrospective local dataset, which was collected during another study [99], was utilized. Each person participating in the study was asked for written consent. The division into disease groups was performed by an experienced ophthalmologist. The available database contains the SLO images for the control group and the POAG group, as well as OCT scans and values of various biomarkers that were collected for accurate diagnosis such as *retinal nerve fiber layer* (RNFL) thickness. As for the SLO part, the dataset contains 227 images, each from a different subject (122 control group, 105 POAG), which means that the available dataset, even though it is small, is rather balanced, which allows avoiding additional problems while training the model.

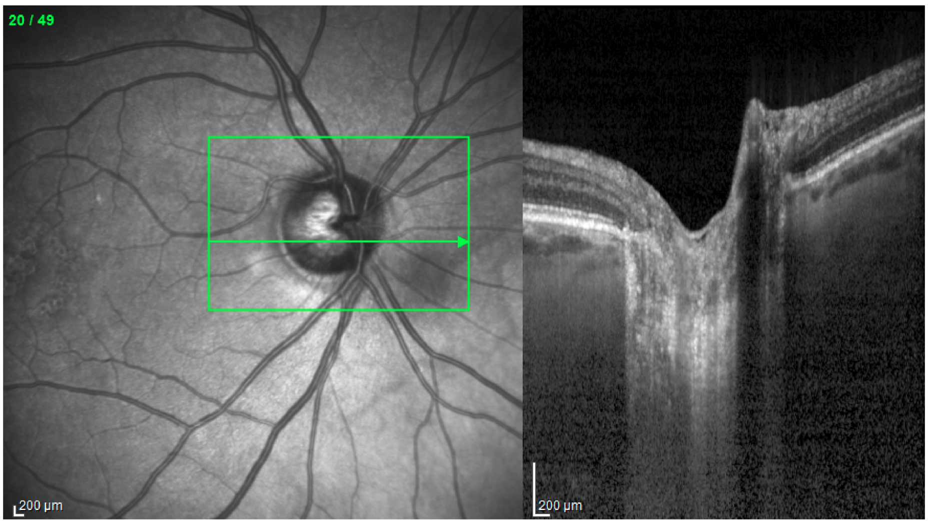


Figure 2.1: Example of the OCT output. On the left side, there is an SLO image with a green box centered on the optic nerve and a green arrow that indicates a place where an actual OCT scan is taken. The OCT scan is shown on the right side.

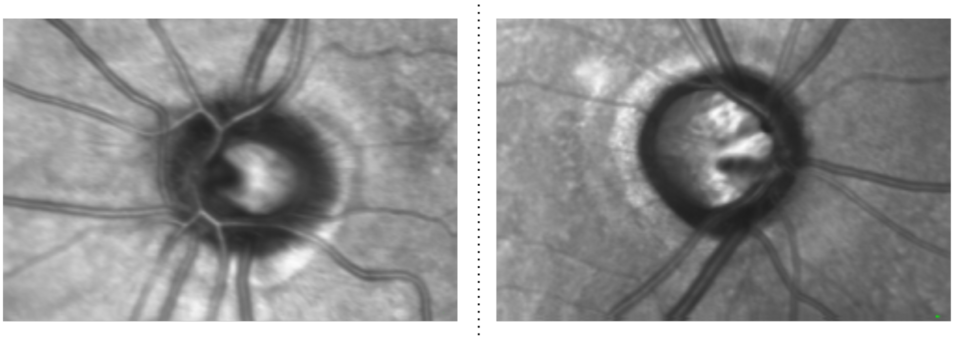


Figure 2.2: Examples of SLO images of a subject from the control group (left), and from the POAG group (right).

Fig. 2.2 presents sample images that were created by cropping the original SLO images from the OCT output to the place where the green rectangle (see Fig. 2.1) that is cantered on the *optic nerve head* (ONH). On the left side, there is an image from a subject from the control group and on the right side is an image from a subject from the POAG group. The final size of the single image is 156 $\times$ 238 pixels. Additionally, in the preprocessing step, the pixel values in the images have been normalized by dividing the grayscale intensity value by 255, so that they fall within the range of $0-1$. Mean

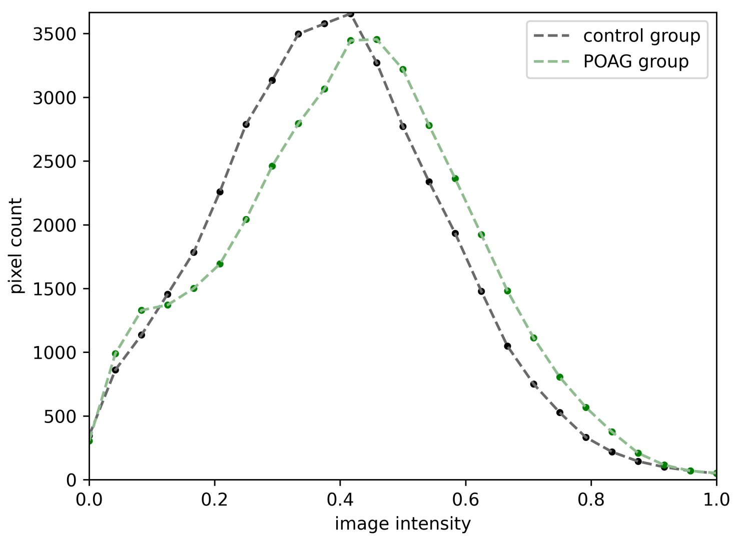


Figure 2.3: An average pixel count in a particular image intensity interval for both considered groups (control and POAG).

value of the normalized image intensity is 0.430 ± 0.083 and 0.462 ± 0.082 for control and POAG groups, respectively. Median values present similar characteristics, and they are 0.434 ± 0.096 and 0.470 ± 0.094, respectively. The distribution of these pixel values for both groups under consideration is shown in Fig. 2.3; it is an average pixel count in a particular image intensity interval. The average is calculated from all images in the group. The intensity interval was divided into 25 bins. A slight, statistically significant (Mann–Whitney U test, $p = 0.005$) shift of the POAG group from the control group can be observed, but in most cases, these values overlap, which makes it infeasible to distinguish between groups based on their intensity values alone.

Additionally, texture properties calculated based on the *gray-level co-occurrence matrix* (GLCM) could be utilized to distinguish two groups of images [100]. In this work, six such measures are considered, including, *contrast*, *dissimilarity*, *homogeneity*, *angular second moment* (ASM), *energy*, and *correlation*. The relationships between all possible combinations of these parameters are shown in Fig. 2.4. As one can see, the distributions of points for both groups overlap to some extent for all of these relationships.

Therefore, the classification based on these texture parameters may not yield good performance. Because the classic extraction of features from images does not allow for high-quality classifications (the results will be presented in more detail in the next section), it was decided to use DL.

Furthermore, to avoid dividing the dataset into three groups (i.e., training set, validation set, and test set), the *k*-fold cross-validation technique was used, which additionally allows to better define the final parameterized quality of the model together with its standard deviation (which is not possible when testing on a single test set). This technique was used in all considered methods for *k* equals 5 and 10, and the results include the mean and standard deviation.

2.1.2 Methods

In this part of the research project, it was decided to use one of the DL methods, which is particularly often applied to image data tasks. The selection process will be discussed later in this section. However, due to the desire to test also various other standard ML methods, it was decided to construct and then compare their performance on SLO data. First, a classifier trained using the RNFL thickness parameter was assessed. The RNFL thickness is commonly used in the clinical diagnosis of glaucoma because it has been shown to change with the progression of the disease [101–103]. In this work, seven different mean values for this parameter could be used: six measured in separate sectors of the retina and their average value [79]. However, during a preliminary test, this average value was proven to be redundant as it did not improve the classification results. Hence it was omitted in further analysis. All the standard ML algorithms presented in the Introduction (MLP, GNB, DT, k-NN, and SVM) were trained based on sector values of RNFL thickness.

Then, the classical approach, presented earlier in the application section, was tested, which is the extraction of features from the image and the subsequent classification of these features using a standard ML algorithm. Various features were considered and compared in this assessment. Initially, all image pixels were used, each as a separate feature. Then a vector consisting

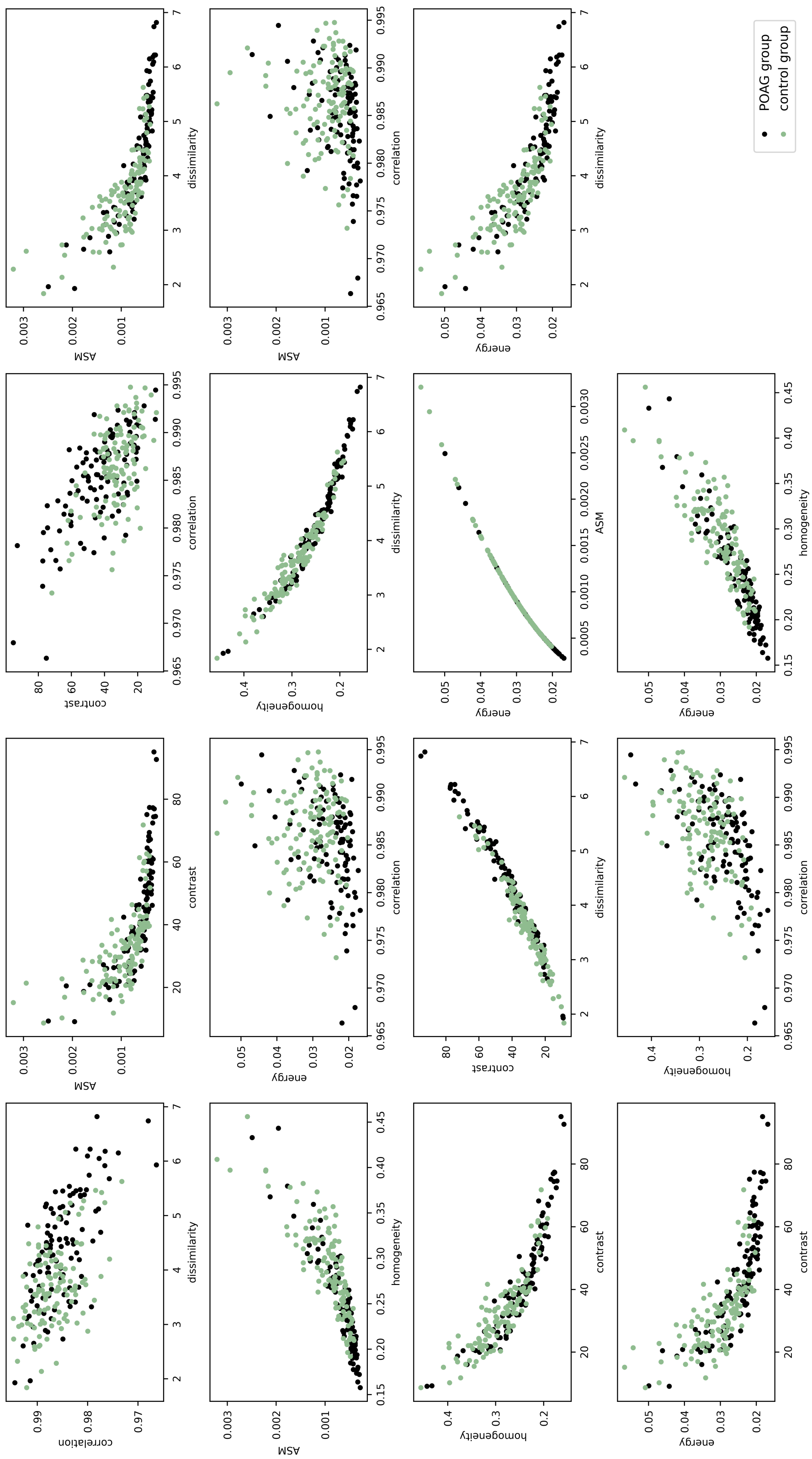


Figure 2.4: The relationships of all possible combinations of the six texture parameters that were calculated based on GLCM for two considered groups.

of the averaged image, both over the columns and the rows, was used as the feature vector. The more commonly used parameters, such as texture ones extracted using the GLCM, as well as the components from the PCA, were then processed and used. After performing the preliminary experiment, only dissimilarity and contrast were selected from the GLCM parameters to be used in this study as the best results were obtained with their use. To apply PCA, the input image was flattened into a vector, and then a 99% of the explained variance was used. In this part, based on the initial tests and knowledge of ML algorithms, it was decided to use only the SVM algorithm, which shows good results while working with image data. Additionally, an SVM classifier trained on combined components from PCA and parameters from GLCM was also considered.

Another solution considered in this analysis was convolutional networks. The final method used was created incrementally based on the results obtained during preliminary testing. It began with the use of popular convolutional network architectures based on their results on the ImageNet [104] dataset. Due to the value of top-1 accuracy[1] and a relatively small number of parameters, it was decided to start with Inception-v3 architecture [105]. This model was chosen because of the relatively small number of parameters while maintaining a good performance (on the ImageNet dataset). Preliminary research showed a BAC score of 0.689 ± 0.020 with the use of this architecture on the glaucoma dataset with $k = 5$ in the cross-validation technique. This unsatisfactory performance led to a search for a different solution that may improve the results. It was decided to use one of the techniques of dealing with the low-data regime – image augmentation. The principle of this method was presented earlier in the 1.2 section. Initially, an analysis of the available image set was performed, and three geometric transformations, which were logical for the dataset, were selected, including translation, reflection, and rotation. The next step focused on the experimental selection of parameters for chosen transformations. The reflection was performed

[1]Top-1 accuracy is a measure counted from the one highest response of the model, sometimes it is simply referred to as accuracy. When comparing the results on models trained on the ImageNet dataset, both top-1 and top-5 accuracy are used, therefore it is distinguished.

both vertically and horizontally. For translation, it was decided to shift a maximum of 15% in relation to the entire size (height or width). And in rotation, a maximum value of up to 50 degrees was used. All these operations and their specific values in a given epoch were selected randomly. Then it was decided to use this image augmentation to train the Inception-v3 network. Significantly improved results were obtained, with a BAC score value of 0.822 ± 0.162. However, this value was still not satisfactory, and the standard deviation was too high, so it was decided to continue looking for a better solution. As mentioned earlier in 1.2 section, when it is not possible to enlarge the available dataset, it is worth considering reducing the complexity of the model. Therefore, a task-specific architecture was designed (see Fig. 2.5). It consists of five convolutional blocks with an increased number of filters from 8 to 64. After each convolutional operation batch normalization and ReLu activation function is performed. The fully connected part consists of two neural layers (dense) with 128 and 2 neurons, respectively. Additionally, the dropout layer between these two fully connected ones is present with a 0.6 rate to further avoid overfitting. The main assumption of the proposed architecture was to have as few parameters as possible so that it could be trained with the relatively small dataset that was available.

The first tests on the architecture, which is presented above (see Fig. 2.5), performed with the same image augmentation as before, showed promising results. They were similar for k equals 5 and 10 and are as follows: 0.905 ± 0.023, and 0.893 ± 0.076, respectively. The benefits of using the new less complex architecture were noticeable, but it was decided to try to further improve the results. At this point, the use of ensemble learning was considered, because it can improve both the final classifier score and the stability of the results. Due to the small amount of data and the already used cross-validation technique, a special type of ensemble learning was selected, which is *cross-validation ensemble* [106]. In cross-validation, the dataset is divided into k equal parts. Then one part of this divided dataset is used for testing and all other parts are used to train the classifier. After that, another part is taken as the test set and this procedure is repeated k times. Contrarily,

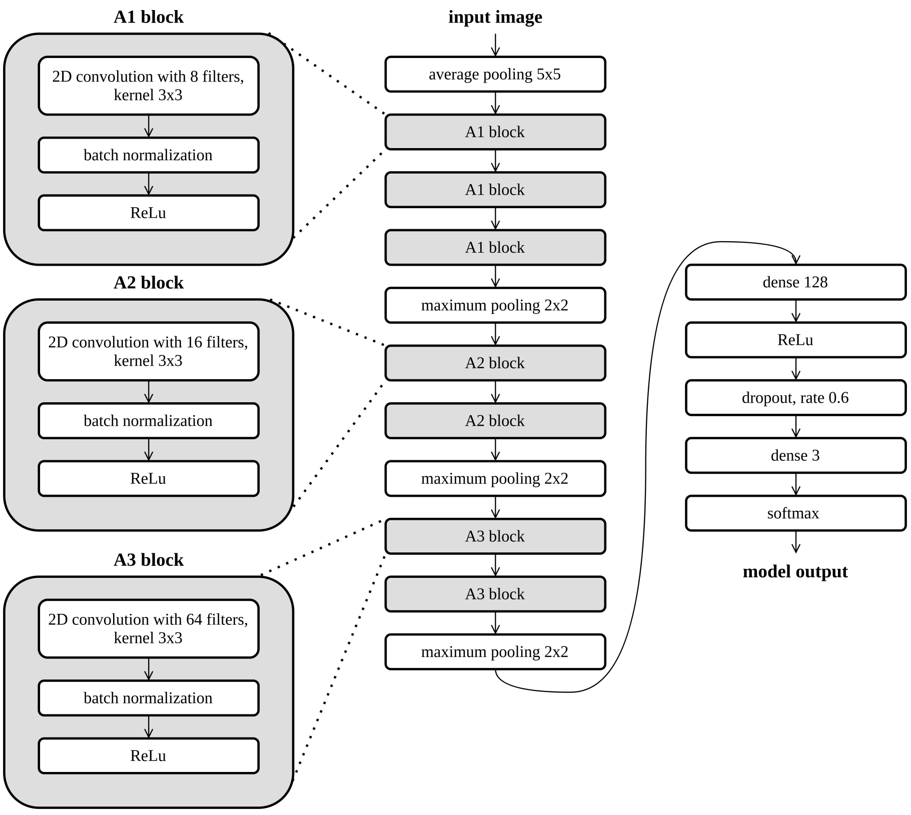


Figure 2.5: The schematic view of the proposed CNN architecture. It consists of different layers: convolution, pooling, activation, normalization, and dense. ReLu — rectified linear unit, softmax — normalized exponential function, dense – fully connected neural layer.

in the cross-validation ensemble, non-test parts of the dataset are not fully combined. For each classifier, different non-test parts are used for validation. In total $k-1$ classifiers are trained on slightly different datasets. An outline drawing of the cross-validation ensemble operation is shown in Fig. 2.6. The classifiers, at each validation step, combine their answer to get the final model output. There are a few different types of classifier combination techniques that could be considered. In this work, two different types were tested and compared: *majority voting* (MV) and *support accumulation* (SA). Additionally, two versions of these approaches were tested: with and without weights. In the version with weights, each base classifier has been assigned a weight according to a validation score obtained during training.

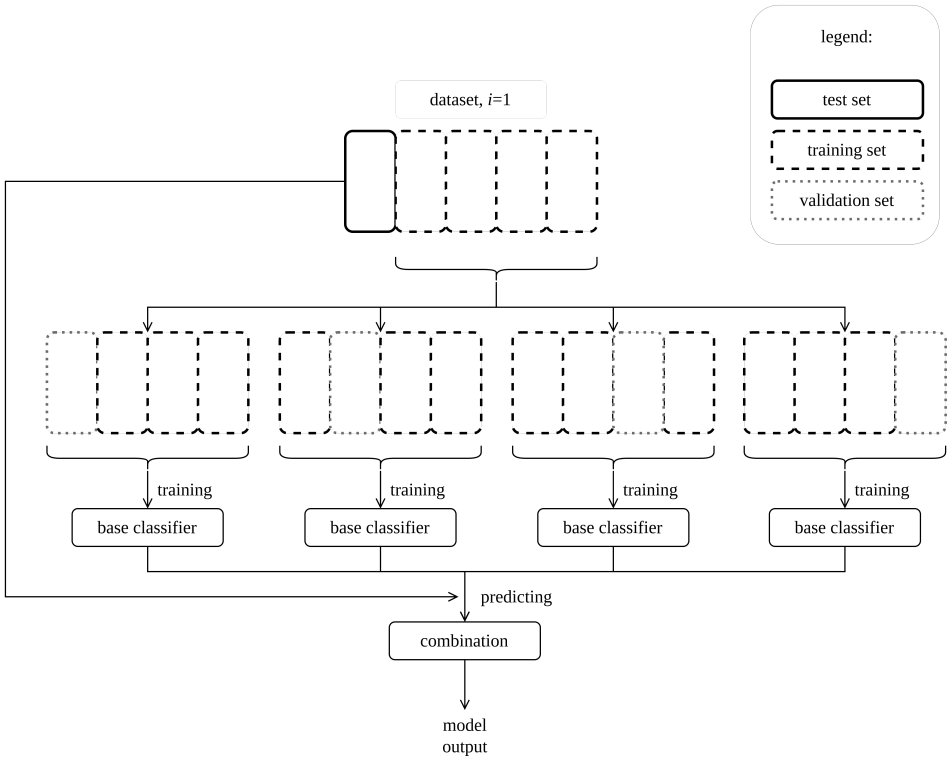


Figure 2.6: Cross-validation flowchart for $k = 5$. This operation will be iteratively repeated for i in range 1 to 5.

The last approach considered in this study was to test frequently used pre-trained convolutional network architectures (pre-trained using ImageNet dataset). This approach is also known as transfer learning. The previously selected Inception-v3 architecture was utilized. A modification to this architecture was added, in which the fully connected part of the model was removed, which corresponds to two layers with 2048 and 1000 neurons each, and it was replaced with two fully connected layers with 128 and 2 neurons. Finally, this modified architecture was fine-tuned on the available data. In this case, a cross-validation ensemble was also used.

All performed experiments were implemented using Python language (version 3.7.3) with the use of the following software libraries: Keras 2.2.5 with Tensorflow backend and Scikit-learn 0.20.3. For the experiments with deep learning algorithms, the Adaptive Moment Estimation (Adam) [107] optimizer was used together with the cross-entropy function as a loss function. For initialization, the Glorot uniform initializer [108] was used both for kernel and weights with initial bias equal to zero. The learning rate was set to 0.001. The algorithm training procedure consisted in training models for up to 250 epochs. During the training process, for each epoch the current state of the model, which achieved the best results on the validation set, was saved. This best final model was recreated and used later in tests. In most cases, the best model occurs well before the 250 epochs limit.

2.2 Segmentation of Bruch's membrane opening

Another issue that was investigated in this research was the task of biomedical image segmentation. The focus here was put on the segmentation of OCT images. As already mentioned, these scans could be used directly in the classification tasks, where the algorithm assigns a particular scan to a given disease group. They could also be used indirectly, to extract the values of key image biomarkers that can be used by ophthalmologists to make an informed decision on the diagnosis and monitoring of glaucoma. One group of such biomarkers that can be determined using OCT scans are anatomical values extracted from the optic nerve head, particularly the position of the *Bruch's membrane opening* (BMO). This membrane is located at the back of the eye and extends over it. It adjusts its shape based on changes in the intraocular pressure [109]. The opening of this Bruch's membrane defines the place where the optic nerve passes through the retina. In the B-scan, this opening appears as two points that mark the termination of Bruch's membrane. One of the parameters that can be extracted from these two termination points is the sectorial Bruch's membrane opening, which is defined as the distance between the two points. Other distance parameters can also be determined, between the termination of BMO and other retinal membranes. For example, the horizontal rim width, which is calculated as the distance between the termination of BMO and the internal limiting membrane in the horizontal line, or the minimum rim width that is calculated as the shortest distance between this termination and an internal limiting membrane. These biomarkers are becoming more popular to support the diagnosis of glaucoma [79, 110, 111], and in addition, they may also be useful in the early detection of changes caused by this disease [112]. However, a manual delineation of these parameters is time-consuming, and standard segmentation algorithms (without the use of ML) may not always be effective when changing, for example, image exposure or due to a high level of noise. Thus, the motivation for the work is to overcome some of these issues. This work has been described in detail in [1]. There are two main goals in this task, firstly to perform the segmentation task with the lowest possible error values, and second, to check the different OCT scan segmentation methods currently in use and compare which of these methods achieves the best performance. Particular attention is paid to understanding the effect that input data size has on the performance of the deep learning algorithm. Thus, exploring whether using the entire image at once or dividing it into smaller parts can be used as a strategy to obtain a better result.

2.2.1 Dataset and preprocessing

This study used data from the same dataset as previously in the glaucoma classification task (data from a retrospective, comparative study [99]). In this case, the dataset used was not limited to only two groups (the control group and the POAG group), but the entire dataset was used, even that coming from subjects suspected of having glaucoma. This data selection, motivated by the fact that the disease group should not affect the quality of BMO determination itself, and even the diversity of data may have a positive effect on the final result of the trained model. In total, data from 325 subjects were used. For each of them, one eye was randomly selected for analysis. For the segmentation study, SLO images were not considered, but the focus was solely on OCT scans. Each OCT examination for each subject, consists of a volumetric scan, comprise of a total of 49 horizontal scans (B-scans). The position, where the scans were taken, is shown in Fig. 2.7 based on the SLO image. A single cross-sectional scan is grayscale and measures 436×384 pixels (height $\times$ width). On such single scans, the BMO that needs to be segmented can be observed. However, not all of the OCT images may show this BMO structure, because these 49 scans may cover beyond

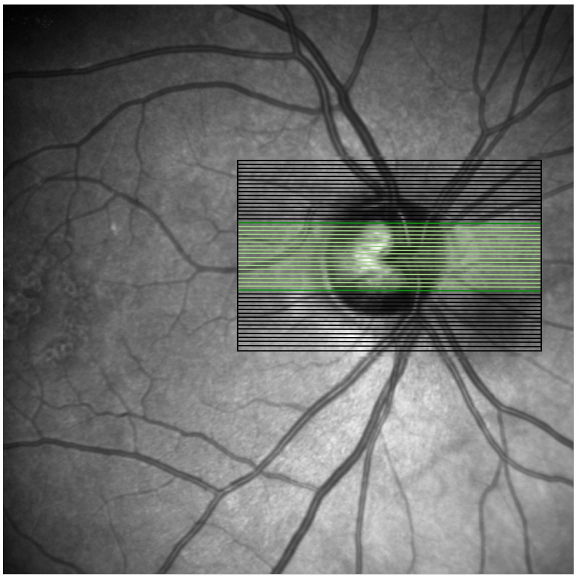


Figure 2.7: An example of an SLO image with annotated the places where OCT scans were taken. Light green lines indicate sixteen middle scans that were used in all analyses. Dark green ones indicate two additional scans that are used only in some experiments.

the ONH. Therefore, not all of these 49 scans are used. It was decided to use mainly the 16 middle scans (in some cases also two additional scans, but this will be described later). These scans are shown in Fig. 2.7 as light green lines, where it can be appreciated that the BMO is always visible on all of them.

A total of 5200 images (16 scans $\times$ 325 subjects) were used for the study (still without counting the two additional scans in some cases). Manually marked BMO points (two points for each scan) were annotated by an experienced ophthalmologist for each scan. Due to the type of data (image + annotations), it was decided to use supervised methods. The method of preparing data for further processing is shown in Fig 2.8. At first, a line is drawn from the two annotated points through the entire height of the OCT B-scan (left side). Then, based on the image and these two lines, a mask is created with two classes, which indicate – outside and inside the BMO (right side). All pixels that were located inside two BMO termination points with them included were marked as class 1, whereas all pixels outside – class 0. These masks will be used as ground truth at a later stage.

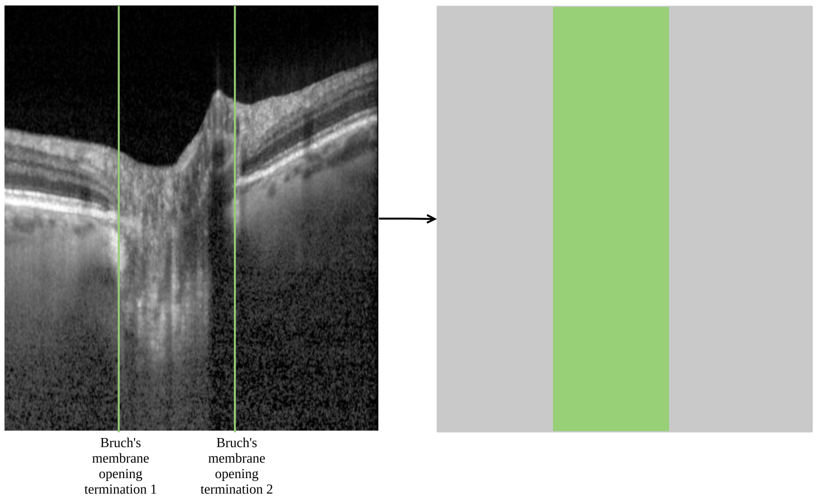

Bruch's membrane opening termination 1
Bruch's membrane opening termination 2

Figure 2.8: Scheme for creating masks from the available data that are later used for the segmentation task. On the left side, there is an OCT scan with the places where BMO was annotated are marked with green lines. On the right side, these annotations are a ready-made mask for further processing with two classes – inside and outside of BMO.

Ultimately, the dataset consisted of images, and each image is accompanied by a mask indicating the location of the BMO. The next step was to divide the dataset into the different sets for training, validation, and testing. It was ensured that scans from one subject are in the same group to avoid data bias. Initially, the entire set was divided into two subsets: training, which made 60 percent of the entire set and test, 40 percent (2080 images). Then the training set was further divided to obtain a set for training and a second one for validation. Here, the relationship was established at 80 (2495 images) and 20 percent (625 images), respectively. The validation set was not involved in the training and was only used for training supervision and model selection. The same type of data division was used when training all the segmentation methods to avoid bias in the results.

As previously noted, special attention has been paid to understanding the effect that input data size has on network performance. This means whether it is better to use the whole image and the appropriate algorithm for it, or it is better to further divide the image into smaller parts. Three types of entry into the network were considered. All of them are presented in Fig. 2.9.

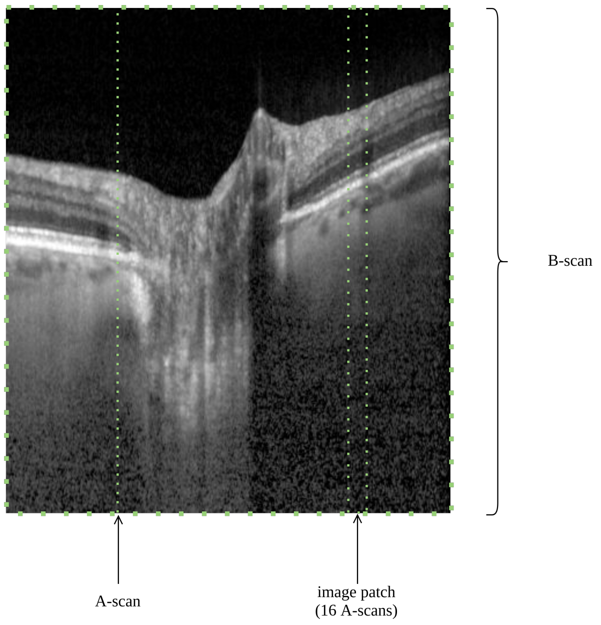


Figure 2.9: It presents three types of input used in further experiments: single A-scans, image patches (16 columns from an image) or the whole image.

1. First, it was decided to use a one-dimensional column (A-scan) that is extracted from the original image (see Fig. 2.9). A label – class 1 or class 0 – was assigned to such a single column, which now becomes a vector and no longer a 2D image. As it was already mentioned, in some cases more than 16 middle B-scans were used in order to complete the segmentation task. In this case, the approach, which will be called the 3-D approach, is also considered. The 3-D approach consisted in the fact that instead of using a single B-scan, a 3-D cube containing this scan and its two adjacent scans was used. So, for the two extreme scans of these 16, two additional scans were used – 18 in total to complete the information. In the case of A-scan as an input, individual columns were not extracted only from one scan but also from neighboring scans. At the end of this data preparation process, instead of one A-scan, the algorithm had 3 single A-scans as input (these A-scans were taken in the same place on the image).

2. Another type of input that was tested was 'parts from the image' – patches. The way of creating such patches with images depends

on the data subset. For the training and validation set, they were pulled out every 4 A-scans, and their width was 16 columns (or A-scans), which means that they were overlapping. In this case, as in the A-scan data, a class was assigned to each such patch. This was done based on the majority class in a given fragment of the image – that is, the class to which the most columns belonged. For the test set, the patches were extracted in every column (or A-scans), which was justified by the desire to achieve more precise results. For the 3-D version of this type of input, patches were not taken from a single scan but also from two additional adjacent scans at the same locations. It is worth noting here that the final class that was assigned was calculated based on the middle scan (in the neighboring scans, BMO points could have been slightly shifted).

3. The last type of input that was considered was the use of the whole B-scan as an input to the ML algorithm. The whole ground truth mask was used as a reference here. In the case of the 3-D approach, as before, 3 scans were taken instead of a single image, but the mask was still used from the middle scan. Because this scan represented the one to be segmented.

For all of the considered entry types, data augmentation was applied. When analyzing the available images, it can be noticed that the vertical position of the BMO points varies across different scans. Therefore, it was decided to use vertical data augmentation, which consisted of shifting all previously presented data (A-scans, image patches, and entire images) randomly up to 90 pixels up or down (about 20 percent of the entire image height) during training. The value of transformations was set empirically.

2.2.2 Methods

Since each of the types of input is different – vector, image patch, entire image – it was decided to use a different, appropriate algorithm for each of them, the performence of which was also assessed at the same time. As DL is gaining more and more popularity as a tool for segmenta-

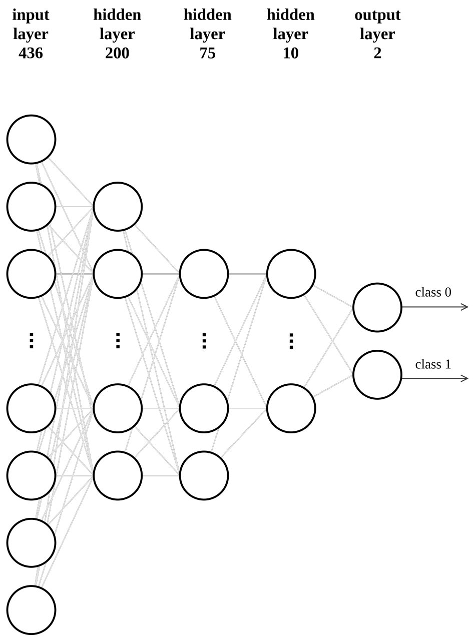


Figure 2.10: The DNN architecture overview with the numbers of neuron in each layer.

tion (see Introduction), it is this type of ML that has been decided to focus on. Three algorithms, which accounted for the different data sizes, were implemented and compared. It is important, however, that each of these algorithms, although it will receive a different type of data, trains on the same dataset, so it is possible to compare their performance.

1. At the very beginning, a popular algorithm was used for data composed of many vectors of values from A-scans, which, depending on the number of hidden layers, can be considered both as an ML algorithm – with one hidden layer, and a DL algorithm – when there are more layers. In the case when it has only one hidden layer, it is often called *Artificial Neural Network* (ANN), and if more – DNN. The idea to use this type of algorithm to image data came from a literature review. The model architecture was adopted from a previous study that used A-scan for classification purposes [113] and it is presented in Fig. 2.10. For the 3-D approach of this method, each A-scan is processed by a separate DNN model, and outputs of these models are combined with the average layer, which provides the final

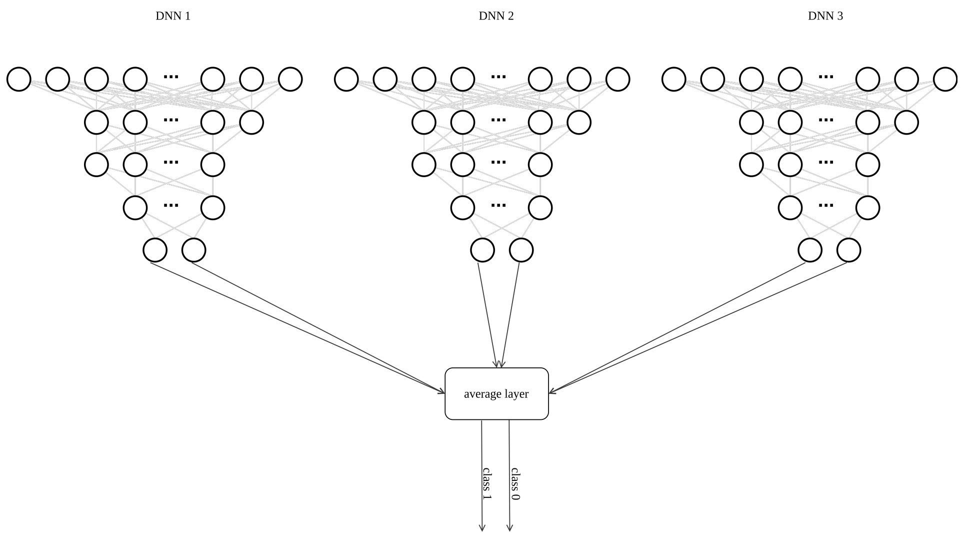


Figure 2.11: The DNN architecture overview for the 3-D approach of considered method. The number of first layer corresponds to the number of pixel in each column, thus to the size of data vector.

response. This cumulative architecture is shown in Fig. 2.11 and it is a single model that consists of a total of three connected models of the previously depicted architecture.

2. The patches extracted from B-scans represent a different data type and, thus a different DL technique should be considered. Since the data type is an image, the CNN algorithm is utilized here. As with the previous method, implemented architecture here was adapted from the work that also considered the task of segmenting OCT scans for ophthalmic applications [7]. This architecture is shown in Fig. 2.12. Additionally, this architecture has also been tested for the 3-D version. In the case of this particular architecture, this adaptation consisted only in changing the size of the received image at the input – from one channel to three channels.

3. The last DL technique that was implemented, for the entire image, was an FCN. This type of network is frequently used in the semantic segmentation task, especially in medical applications. In semantic segmentation, each pixel in an image is classified into a specific class,

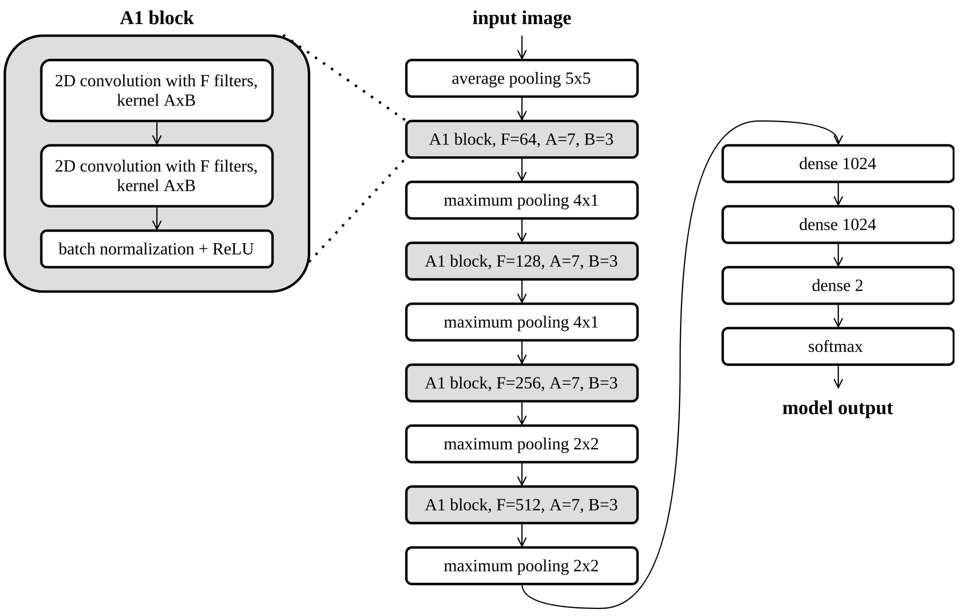


Figure 2.12: The CNN model architecture for image patches extracted from a B-scan. The architecture was adapted from [7].

based on a ground truth provided. One of the most commonly used types of FCN, the U-Net architecture, was implemented [41]. Details of the used architecture are shown in Fig. 2.13. To make five pooling operations possible, appropriate padding was added to the original image (zero padding 38×0 – one dimension only). This padding was removed at the end of the network, using a cropping operation, so the size of the output mask matches the size of the original image. This operation made a comparison between the ground truth mask and the output of the network possible. The 3-D version of this method, similar to the CNN network, did not require any changes in the network architecture. The only change is the number of channels that were entered as the input image. Each such channel represents an adjacent B-scan. The output of this network, both in the standard version and in the 3-D version, were segmented masks. However, in the case of the two previously used methods, all pixels that belonged to the same column were assigned to the same class because they only operated on entire columns. With this method, the output prediction map could contain columns where pixels were assigned to different classes. Therefore, it was decided to apply an additional post-processing step. It consisted in computing the mean value in a given column (keeping in mind that only classes 0 – outside of BMO and 1 – inside of BMO, are available). Then it was checked whether this mean was greater than 0.5. If so, class 1 was assigned to all pixels that were in such a column, otherwise class 0.

As there are also works showing modified versions of the U-Net architecture with promising results, it was decided to test some of its modifications as well. Three such modifications were considered. The first one was a standard architecture with *residual learning* incorporated into the network [39], which will be later referred to as Residual. The second one was a modification of a standard version by using additional *squeeze-excitation* blocks [114], this will be called SE in short. And the last one considered here, had a bottleneck modification with a RNN layer [115]. For this method, instead of using RNN layers, CNN-LSTM layers were utilized [116].

For all of the experiments, the Adam optimizer was used together with the cross-entropy as a loss function. The learning rate for training was empirically set to 0.001. The algorithm training procedure consisted in training models up to 30 epochs. This value was determined empirically during experiments based on a loss function trajectory. The process of training was similar to the one from a classification study and consisted in iteratively saving the model that achieved the best results on the validation set during training. The last saved model was then used in the evaluation process. The batch size used during experiments was different for different methods. For the segmentation using DNN and CNN the batch size was set to 100 A-scans/image patches respectively, while for the FCN it was set to 5 images. To obtain the most reliable results, it was decided to train each method three times and the means and standard deviations of each of them were compared. It is worth noting here that with each training, the models were initialized from the beginning (i.e., from the scratch), without information from previous training sessions. All performed experiments were imple-

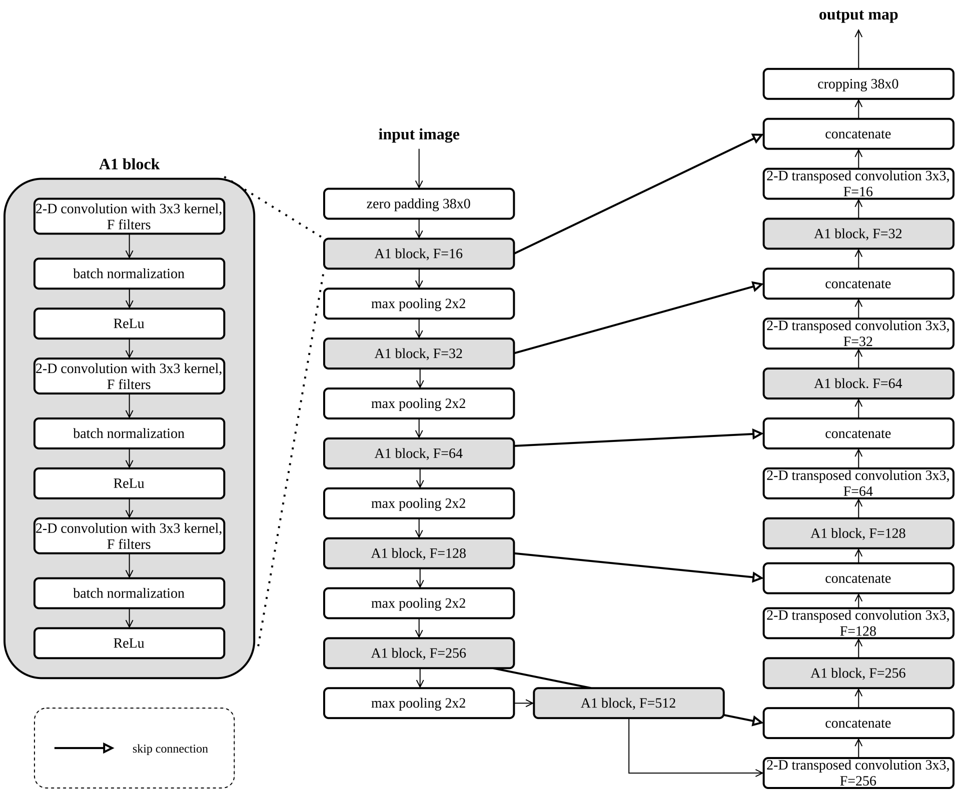


Figure 2.13: Schematic view of the U-Net like architecture used in this study.

mented using Python language (version 3.6.4) with the use of the following software libraries: Keras 2.2.4 with Tensorflow backend and Scikit-learn 0.22.2, Scipy 1.1.0, Tensorflow 1.14.0, and Numpy 1.10.0.

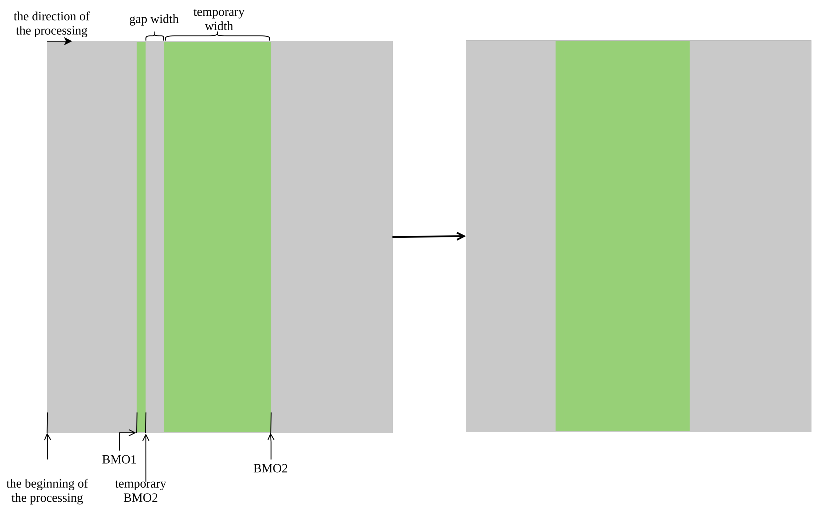

the direction of the processing
gap width
temporary width
BMO1
temporary BMO2
BMO2
the beginning of the processing

Figure 2.14: Overview drawing for post-processing step. On the left, there is the unprocessed mask with marked places used for processing. On the right side, there is the output mask after post-processing.

Post-processing

The result of all the methods presented before provide output probability masks where each column has a class assigned, inside the BMO and outside the BMO. For the final result of each method, it was assumed that an entire group of columns that represent the BMO have the same value. Hence, it was necessary to add a post-processing step that would allow the correct calculation of the metrics for comparison. A universal procedure was developed, since sometimes individual columns that belong to class 0 were inside the BMO and vice versa, while sometimes individual columns assigned to class 1 were outside the BMO. This led to more than one block with class 1 (i.e., BMO) on one probability map, which caused problems in determining the true position of termination points of the Bruch's membrane (which will be defined as BMO1 and BMO2). So the following post-processing step was added to determine the exact positions of the opening (see Fig. 2.14). The pseudocode of this process is shown in algorithm 1. After searching for BMO1 and BMO2 items, a new mask is created which assigns all columns on the left side of BMO1 and the right side of BMO2 to class 0, and all those that were between BMO1 and BMO2 – class 1 (see right side of Fig. 2.14).

Algorithm 1: Pseudocode for the post-processing algorithm used.

```
Input: X as an 2-D array which represents probability masks
Output: The location of termination points of Bruch's membrane –
        BMO1 and BMO2
l := number of columns in X;
BMO1 := 0;
BMO2 := 0;
gap_width := 0;
temporary_gap_width := 0;
for i := 1 to l do
    label := the label of ith column;
    if label = 1 and BMO1 = 0 and i $\geq$ 50 then
        BMO1 := i;
    end
end
for i := BMO1 to l do
    label := the label of ith column;
    if BMO1 $\neq$ 0 & label = 0 then
        gap := 0;
        if BMO2 $\neq$ 0 then
            BMO2 := i;
        end
        while label = 0 do
            gap := gap + 1;
            label := the label of ith column;
        end
    end
    else if label = 1;
    & BMO1 $\neq$ 0 & BMO2 $\neq$ 0 then
        temporary_gap_width := 0;
        while label1 = 1 do
            temporary_gap_width := temporary_gap_width + 1;
            i := i + 1;
            label := the label of ith column;
        end
    end
    else if gap $\neq$ 0 & temporary_gap_width $\neq$ 0 then
        if temporary_gap_width $\geq$ gap then
            BMO1 := i;
        end
    end
    end
```

2.3 Dealing with the curse of dimensionality – feature selection

In a clinical context, the classification of a patient into a particular group of eye diseases is not typically made using only medical images. Before the extensive use of image data for the ML classification task in ophthalmology, the most popular method was classification by numerical data (still very popular). In the case of diseases, these are usually key biomarkers that change as the disease progresses and are used to inform clinical decisions. These values are collected during patient diagnosis by ophthalmologists. In clinical practice, the physician can make a diagnosis based on these biomarkers and the subjective assessment of the images. Therefore, it was decided to use these biomarkers (excluding images) to perform classification using standard classifiers as described in the Introduction. Unfortunately, in the case of glaucoma, the potential number of available biomarkers is substantial. For example, in the dataset used in this study, there are as many as 48 different biomarkers for each patient. Therefore, to achieve good results in classifying these parameters, the best solution is to have a large set of training data. In the medical field, this is not always feasible, for example, when the disease is not that common, it is not always possible to get a large number of subjects in this group. In these kinds of situations, the a curse of dimensionality occurs, which was described in the first chapter. Therefore, the main challenge here was to propose a new method for dealing with a curse of dimensionality without the information lost, so without features rejection in the process. The considered task is a classification of three groups of glaucoma patients, which include, the control group – subjects without glaucoma, POAG group – subjects with primary open-angle glaucoma diagnosed, and, the glaucoma suspects group – subjects that are suspected of having glaucoma in the feature. This last group is the one that is the most difficult to classify and interpret since patients that belong to this group do not have glaucoma yet, but there were assigned to this group based on the glaucomatous optic disk appearance.

The purpose of this part of the study was to propose a method for feature selection without any feature rejection for the small datasets, as well as a method that can accelerate learning because it can operate using only small parts of the dataset for each base classifier.

2.3.1 Dataset

The dataset that was utilized here is a numerical one and contains values from 211 subjects, evenly divided into three groups (69 subjects from the control group, 70 from a POAG one, and 72 glaucoma suspects). For each subject, as mentioned before, there are 48 features, which include, among others, the value of intraocular pressure, parameters that describe the morphology of the optic disc, and biomarkers that are extracted from OCT B-scan – retinal nerve fiber layer thicknesses. All of these biomarkers were collected during standard glaucoma diagnosis. A subject was assigned to the particular group by an experienced ophthalmologist based on biomarker values and subjective analysis of eye images. More details about the utilized dataset can be found in [117]. Additionally, all features were standardized using *StandardScaler* function from the scikit-learn package [118]. This function was fitted to the training set and then all features were transformed.

Taking into account the low-data regime operation, to avoid the additional division of the dataset into three groups and to improve the overall validation process, a 5-fold cross-validation was used for all experiments which was repeated 5 times.

2.3.2 Methods

According to the knowledge presented in the introduction, the methods of selecting features can be divided into two main groups. In the first one, some features that are least correlated with a given problem are rejected in order to improve the quality of a given classification. As in this work, it was decided not to reject any information that is typically used by ophthalmologists in the decision process, the focus was on the second group of feature selection methods. That is, using a classifier ensemble to train each base classifier from a different subset of the entire feature space, which will minimize feature rejection, and possibly completely resolve it. Consequently, the purpose of this study was to propose a method of subsets extraction that would be done at the same time as the classifier ensemble would be created. In contrast to Random Subspace, the extraction, or drawing of subsets from the feature space, is not random. In other words, each feature in the feature space will not be drawn to a specific subspace with the same probability. To increase the probability of drawing more correlated features, the F-value from the *analysis of variance* (ANOVA) test was used here. For each feature, a single F-value is calculated, and then, a set of these values is transformed so that it can be considered as a discrete probability distribution. In order to do that, each F-value was divided by the sum calculated based on the entire set which results in a vector whose sum is equal to 1. After creating this probability vector, a function is called which uses this vector, and draws the given number of features, taking into account the given probability. In this way, it was possible to increase the probability of drawing features with higher F-value into a feature subset, but at the same time, despite the diminished likelihood, without initially discarding any features. Ultimately, the drawing procedure was repeated as many times as there were base classifiers in the ensemble so that each of them had a separate training set. The core of the proposed method is presented in algorithm 2. Additionally, to the process described earlier, in the training process of each base classifier, a weight was calculated based on the value of balanced accuracy in the validation step. After the training, a weighted support accumulation is performed. Weighting, which means, that multiplication of the supports with the calculated weights is done and then a support accumulation is executed based on the results of this multiplication. To summarize, the proposed method is non-randomly drawing a set of subsets from the entire feature space, and then, based on these subsets a classifier ensemble is trained. With this approach, there is no feature rejection at the beginning of the training, and the features that have higher F-value are favored.

Algorithm 2: The fit function for the proposed method. This pseudocode come from [3].

```
Input: X as an array of training examples with y containing
       corresponding labels and n as a subspace size and k as a size
       of ensemble
Output: A list of trained base classifier (ensemble)
        and corresponding weights
n_features := number of features available in X;
p := the list of F-value for each feature in X obtained from ANOVA;
p := p/sum(p);
f := [0, 1, ..., n_features];
classifiers := a list of k base classifiers;
for classifier in classifiers do
    ss_indexes := a list of n drawn numbers from f with
      the probability of p;
    train classifier on all samples from X but only on features with
      ss_indexes;
    append trained classifier to a list of ensemble;
    append weight, calculated as a balance accuracy score for
      classifier determined on those training samples, to a weights
      list ;
end
```

The proposed method is universal in the sense that any base classifier can be selected for it. In this study, four different base classifiers were evaluated, including MLP, k-NN, SVC, and a specific type of decision tree, which is a *Classification and Regression Tree* (CART). The parameters of the classifiers have been selected empirically and are as follows. For the MLP, the number of hidden layers was set to 100, the learning rate to 0.001; for training Adam optimizer and the ReLu activation function were utilized with the number of iterations in single epoch up to 20. In the case of k-NN, the number of neighbors was set to 5 and leaf size to 30; weights were initialized uniformly and the Euclidean measure was used. For the SVC, after preliminary results, a linear kernel was chosen with additionally enabled probability estimates; the number of iterations was set to 1 with the shape of decision function one vs rest. The CART method was trained with the Gini impurity criterion and without setting maximum depth; the minimum number of samples required to split an internal node and to be at a leaf node was set to 2 and 1, respectively.

Additionally, in order to be able to reliably evaluate the obtained results, it was also decided to test other commonly used methods for the feature selection on the same dataset, together with training the models on the entire feature space. One from each group of feature selection algorithms was chosen. From the first, the algorithm consisted in selecting *k* most differentiable features, where *k* is a parameter of this algorithm, was utilized. To find these features, the ANOVA test was used and this algorithm will be later referred to as *k-best*. From the second group, the Random Subspace algorithm was selected.

For all considered solutions that operate using feature selection, experiments were performed using different sizes of subspace, from 1 to 48 (selected values from this range were chosen with linear distances from each other).

2.4 Orthogonal convolutional neural network

The last project of this Ph.D. investigated a general problem that appeared in all previous studies – small dataset challenge, or in other words - low data regime. Despite the many existing solutions that are successfully used, such as data augmentation and transfer learning, among others. The use of a small dataset in machine learning applications still remains a challenge. Additionally, the shift toward deep learning models has made working with small datasets even more difficult. The idea to propose a new advancement to this field came while reading an interesting work on the *orthogonal convolutional neural networks* (OCNN) [119]. Although this study is not the first or a unique attempt to use orthogonalization in CNN, it was an inspiration to test the use of these networks for small training datasets. As CNNs are mostly used in problems based on image-type data, also in supporting medical diagnostics, and they require a sufficiently large set of samples due to the large number of parameters to be trained – very often their users are faced with the problem of insufficient data. However, even if this data problem does not occur, typically already trained models have filters (convolutions layers) that are very similar to one another, which can lead to redundancy of features that will be extracted. Additionally, sometimes there is instability while training as well which may be caused by e.g. exploding or vanishing gradient. These are the problems that OCNN tries to solve with orthogonalization. An additional advantage of this particular orthogonalization method is the lack of need to make changes to the network architecture itself. Obviously, the use of the method involves an additional computational overhead, but it is not significant and the potential benefits in performance are worth the trade-off.

There are two main approaches how to use orthogonalization in CNN, including *kernel orthogonality* [120] and *orthogonal convolution*. In [119] the second approach is used, which does not require reshaping the input or output, but connects them with the doubly block-Toeplitz matrix and only enforces orthogonality to this matrix.

The study shows that classifiers trained using this kind of orthogonalization could outperform the ones with kernel orthogonalization. The details of this orthogonal convolution could be found in the paper, but the main thing to remember is that in the end, additional regularization loss, which is named *orthogonal regularization loss*, is added to the standard loss of the CNN (see Eq. 2.1 from [119]). In the equation, L_{CNN} is the standard loss that is used in the task, for example, softmax, and L_{orth} is added orthogonalization loss. α corresponds to a hyperparameter that could be tuned during the training process.

$$L_{final} = L_{CNN} + \alpha L_{orth} \tag{2.1}$$

From the experimental findings of the study, the idea was born to check whether OCNN would also be of benefit for small datasets. The reason for this was that less similar filters would produce more unique features and perhaps both the size of the model and the size of the training data that would be needed could be reduced. Although the research that is done here adopts a previously proposed method, the results demonstrate a new application of this method that shows much greater benefits of using if (for small datasets).

2.4.1 Dataset

As a dataset for experiments, it was decided to use two kinds of datasets, which include widely used data for the assessment of image classification algorithms, the CIFAR-10 dataset, and a medical dataset containing OCT scans from the posterior part of the eye. CIFAR-10 [121] dataset contains 10 classes with 60000 images in total. Each image has a size of 32×32 pixels. The dataset is further divided into 50000 training images and 10000 test ones. The second, medical dataset contains a total of 7340 OCT images, with a number of the internal retinal layers visible and delineated. The main purpose of the dataset is to enable the training of segmentation algorithms. This dataset is part of an open access dataset, which is described in detail in [122]. The dataset includes retinal images of subjects with and without AMD. Together with the image set, there are also ground truth images with four marked layers. The B-scan size is 512×512 pixels. The dataset is divided into three parts with a ratio of about 60/20/20 for training (4439 images), validation (1481 images), and test (1420 images), respectively.

2.4.2 Experimental protocol

The whole experiment was designed to compare the results obtained with the chosen algorithm without any modification (*baseline*) and then, on the same dataset and method but adding the orthogonalization. Subsequently, the dataset was reduced, and the training process was repeated. It is worth noting that only the size of the training set was changed, while the size of the test dataset was kept the same throughout the experiments, to the comparison would be fair. For the CIFAR-10 dataset, nine different sizes of the training set were selected at 100/50/20/10/5/2/1/0.5/0.2 ratio compared to the original dataset, where 100 corresponds to the whole original dataset.

Table 2.1: Details of U-Net architecture utilized in this study. The last filter size refereed to a number of classes in the problem, which is four due to the number of retinal layers, which means that each channel will present a single class.

PART	DETAILS
encoder part	double convolution followed by four convolution blocks
decoder part	four up-convolution blocks followed by a single convolution operation with kernel size equal to one
convolution block	maximum pooling 2x2 with double convolution
up-convolution block	upsample with double convolution
double convolution	two times repeated sequence of 2-D convolution with kernel 3x3 followed by batch normalization and ReLu
filter sizes	64, 128, 256, 512, 512, 256, 128, 64, 64, 4

Table 2.2: Values of parameters in utilized in this study U-Net architecture.

PARAMETER	VALUE
SGD momentum	0.99
learning rate	0.01
number of epochs	20
batch size	1
random seed	42

For the medical dataset, because of the smaller size of the entire set, only four different sizes were tested, including the whole dataset, 20, 10, 2 percent of the original dataset.

Additionally, it was also decided to check the results for two different machine learning tasks: classification and segmentation. The classification was performed on the CIFAR-10 set with the ResNet-18 as a classifier, and the segmentation was performed on the OCT scan set using the U-Net architecture. The details of utilized U-Net is shown in Table 2.1. The model was trained with the parameters presented in Table 2.2. For the ResNet-18 the same architecture and the same parameters as in [119] were used, without modifications.

Additionally, for this part of the experiment that simulates a binary classification problem (with only two classes), several datasets were created from all possible combinations of two classes in the CIFAR-10 set, and then algorithm performance based on these sets was checked and compared. In this case, six different dataset sizes were evaluated at a 100/50/10/5/1/0.5 ratio compared to the original dataset.

Each algorithm was trained three times on each dataset size and the mean value together with the standard deviation was calculated. All experiments were performed based on the code provided by the authors of the original work [2] using Python 3.8.6 and PyTorch 1.10.2 for classification. For segmentation, the code had to be adapted and a different architecture was implemented to fit the problem.

[2]https://github.com/samaonline/Orthogonal-Convolutional-Neural-Networks

Chapter 3

Results and partial conclussions

The following chapter presents all the results obtained from previously described studies. The results are organized in the same order as the description in chapter 2. This means the order is as follows: glaucoma classification, segmentation of Bruch's membrane opening, feature selection, and OCNN. Additionally, for each of the studies, the results are discussed, and a conclusion of the main findings is provided.

3.1 Glaucoma classification

As described in the previous chapter, the glaucoma classification study consist in the task of classifying two disease groups: control and glaucoma one – POAG. The classification was performed using a small image dataset of SLO images, which are not typically used for this classification task. All experiments were carried out using the *k*-fold cross-validation technique (or in the case of the CNN – cross-validation ensemble) for two different *k* values: 5 and 10. In order to be able to assess whether differences in the obtained results are statistically significant, the Wilcoxon test was used [123].

The first part of the experiments considered standard image classification methods to separate the images into two groups. The results of which are presented in Table 3.1. Initially, all standard classifiers were tested on the RNFL thickness data in which each object has 6 different features (i.e., thickness values). All of these classifiers achieved a BAC score above 0.700, which placed it higher than the Inception-v3 architecture without any data augmentation (see Chapter 3). The best performance obtained from the thickness data as an input was achieved using an SVM classifier (SVC) with a BAC score of 0.888 ± 0.083. Whereas the worst results were obtained with MLP. All algorithms achieved a similar level of standard deviation, with relatively high values. The lower part of Table 3.1 shows the results obtained with the SVM classifier trained using image data. Both the whole image and features obtained directly from this image were tested. The results obtained are lower than those using the RNFL thickness values. The highest BAC value obtained is 0.786 ± 0.043 using a combination of parameters from GLCM and PCA. However, no method has obtained statistically better results than the other.

The second part of the experiment shows results from the CNN approach, in which as an input to the algorithm unprocessed SLO image is fed. As mentioned in the previous chapter, the Inception-v3 architecture trained from scratch achieved only 0.698 ± 0.020 without data augmentation and 0.822 ± 0.162 when using data augmentation. Therefore in this part of the experiment, transfer learning was also used while training the Inception-v3 architecture, the results of which can be seen in the upper part of Table 3.2. This table provides a summary that includes the results for both the single trained model with the cross-validation technique as well as the models trained using the cross-validation ensemble, for k equal to 5 and 10. Considering this architecture, the use of a classifier ensemble positively influenced the results, resulting in an average improvement of 20%. For $k = 5$, this improvement is not statistically significant, however, for $k = 10$ it is. Ultimately, with the help of this architecture and the use of ensemble learning, it was possible to obtain an SLO image classifier with a BAC score of 0.945, which is superior to the performance of the RNFL thickness classifier. In this work, also an additional experiment was performed, which involved the use of a custom-made, task-specific architecture that was characterized by a smaller number of parameters within the network.

Table 3.1: Results obtained from two experiments using approaches that do not involve deep learning techniques. The first part shows mean values and standard deviations of balanced accuracy across the five considered techniques using the SVM classifier based on: (1) whole image as a vector, (2) averaged image over columns and rows, (3) GLCM parameters, (4) PCA results from an image, and (5) the combination of the PCA results and the GLCM parameters. The second part of the table presents mean values and standard deviations of balanced accuracy across the five considered machine learning algorithms trained based on RNFL thickness. The number below the balanced accuracy value shows to which model, the model that this BAC belongs to obtained statistically significantly better performance (Wilcoxon test, $\alpha = 0.05$) (considering only the given data).

utilized data	k					
RNFL thickness values	model	MLP 1	KNN 2	SVC 3	DTC 4	GNB 5
	5	0.721 ± 0.063 —	0.886 ± 0.065 1	0.880 ± 0.073 1	0.861 ± 0.058 1	0.886 ± 0.065 1
	10	0.749 ± 0.104 —	0.881 ± 0.086 1	0.894 ± 0.093 1	0.913 ± 0.075 1	0.888 ± 0.083 1
image	method	WHOLE IMAGE 1	AVERAGED IMAGE 2	GLCM 3	PCA 4	GLCM + PCA 5
	5	0.770 ± 0.047 —	0.734 ± 0.088 —	0.694 ± 0.108 —	0.768 ± 0.047 —	0.786 ± 0.043 —
	10	0.777 ± 0.083 —	0.755 ± 0.098 —	0.695 ± 0.119 —	0.760 ± 0.067 —	0.770 ± 0.074 —

This architecture was trained without the use of transfer learning and the results are presented in the lower part of Table 3.2. The use of ensemble learning improved the results by about 30%. The best score is obtained with the use of a cross-validation ensemble, with k set to 5-fold and regular majority voting, which provide a mean BAC value of 0.962. With a task-specific architecture, only this best model achieved statistically significantly better results than a single model and ensemble classifier with the support accumulation. Comparing these two different architectures, it can be seen

Table 3.2: Results obtained from experiments involving CNN algorithm. The first part (top) shows mean values and standard deviations of balanced accuracy using modified Inception-v3 architecture and different DL approaches. The second part (bottom) presents also mean values with standard deviations of balanced accuracy across different DL approaches but using a task-specific architecture. The number below the balanced accuracy value shows to which model, the model that this BAC belongs to obtained statistically significantly better performance (Wilcoxon test, $\alpha = 0.05$) (considering only the given data).

architecture	k	single model	ensemble methods			
			majority voting		support accumulation	
			regular	weighted	regular	weighted
		1	2	3	4	5
Inception-v3	5	0.909 ± 0.044 —	0.945 ± 0.042 —	0.926 ± 0.055 —	0.930 ± 0.050 —	0.930 ± 0.050 —
	10	0.877 ± 0.058 —	0.920 ± 0.050 1	0.920 ± 0.050 1	0.920 ± 0.050 1	0.920 ± 0.050 1
		1	2	3	4	5
task-specific	5	0.905 ± 0.023 —	0.962 ± 0.016 1,4,5	0.930 ± 0.028 —	0.931 ± 0.015 —	0.931 ± 0.015 —
	10	0.893 ± 0.076 —	0.930 ± 0.070 —	0.930 ± 0.070 —	0.930 ± 0.070 —	0.930 ± 0.070 —

that in most cases (except for one single model) the use of a smaller architecture allowed for better results, and training time was also shorter (due to fewer parameters and no need to pre-trained on the ImageNet dataset).

Additionally, for the task-specific architecture, because of the superior results, it was decided to calculate some additional metrics to analyze the performance in more detail (see Table 3.3). Those metrics include all parameters from the confusion matrix as well as sensitivity and specificity. Similar to the BAC, these results also confirm that single models perform worse than the ones using a classifier ensemble. Additionally, all models obtained high sensitivity which indicates the ability of a model to correctly identify a disease class, which was one of the main motivations of this study.

Table 3.3: Mean values of the confusion matrix parameters along with sensitivity and specificity across different DL approaches based on SLO images using task-specific architecture, where SEN – sensitivity and SPE – specificity.

k	method		TN	FP	FN	TP	SEN	SPE
5	single model		21.6	2.8	1.6	19.4	0.924	0.885
	MV	regular	22.7	1.7	0.0	21.0	1.000	0.930
		weighted	22.6	1.8	1.4	19.6	0.933	0.926
	SA	regular	22.4	2.0	1.2	19.8	0.943	0.918
		weighted	22.4	2.0	1.2	19.8	0.943	0.918
10	single model		11.0	1.2	1.2	9.3	0.886	0.902
	MV	regular	11.2	1.0	0.6	9.9	0.943	0.918
		weighted	11.2	1.0	0.6	9.9	0.943	0.918
	SA	regular	11.2	1.0	0.6	9.9	0.943	0.918
		weighted	11.2	1.0	0.6	9.9	0.943	0.918

The results obtained in this study demonstrate that the SLO images contain sufficient information to support the diagnosis of glaucoma. These SLO images could be used independently or combined with another imaging modalities (e.g. OCT scans) to support the clinical diagnosis of glaucoma. It is interesting to observed that, the performance obtained while classifying these images exceed the results obtained while using RNFL thickness biomarkers. In addition, despite the use of a small dataset, it was possible to achieve over 96 percent accuracy. However, there is a clear need to use techniques to deal with a low-data regime such as transfer learning, and smaller ‘custom’ architectures that are more suitable for a small dataset, thus improving the performance.

3.2 Segmentation of Bruch's membrane opening

In this part of the study, the task of segmenting BMO was performed. The experiments aim to accomplish two goals, the first was to carry out segmentation with the highest possible precision, and the second – to find out which type of input data and corresponding algorithms would be best for this particular task. All the approaches that were tested and compared had the same dataset provided, which include OCT B-scans together with the ground truth marking the BMO region. The only thing that distinguished these approaches was how the data was passed to the algorithm, which included passing separately individual columns from images, a set of columns (image patches), and a whole image. The output for all models is a mask that represents the location of the BMO. The map would typically have a background area (class 0) on the side of the image and a single BMO block (class 1) at the center of the image. From this BMO block, the two BMOs points can be extracted. Based on these masks, similarity metrics such as DSC, JSC, and ACC were calculated (see Introduction). Each model was trained three times; therefore, the presented results are the average with their corresponding standard deviation. Additionally, to compare the obtained results from DL models, inter-grader annotation was performed, and the results based on that are presented.

Table 3.4 presents the results obtained with these segmentation metrics. The order of results is from the smallest input size to the largest. For the method, that involved an A-scan from the image the best results are obtained with the 3-D version of it, with a pixel ACC equal to 0.974 ± 0.008. The data augmentation did not improve the ACC and this could be caused by the lack of convolutional layers in the chosen architecture. The second method, which includes image patches with a CNN architecture, achieved the best performance for both the 2-D and 3-D approaches with the use of data augmentation with ACC equal to 0.989 ± 0.001. This method obtained better results in each experiment and a lower standard deviation while compared to the method using single columns from images. The 3-D approach

Table 3.4: Mean values together with standard deviations of the similarity metrics and results from inter-grader annotations. DA indicates data augmentation.

method	input size (pixels)	DA	ACC	SDC	JSC
inter-grader	$436 \times 384 \times 1$	no	0.975	0.952	0.984
DNN model with A-scan as input	$436 \times 1 \times 1$	no	0.964 ± 0.003	0.949 ± 0.005	0.910 ± 0.006
	$436 \times 1 \times 1$	yes	0.960 ± 0.003	0.945 ± 0.004	0.903 ± 0.006
	$436 \times 1 \times 3$	no	0.973 ± 0.008	0.953 ± 0.027	0.925 ± 0.025
	$436 \times 1 \times 3$	yes	0.963 ± 0.010	0.939 ± 0.025	0.902 ± 0.026
CNN model with image patches as input	$436 \times 16 \times 1$	no	0.984 ± 0.007	0.977 ± 0.002	0.956 ± 0.021
	$436 \times 16 \times 1$	yes	0.989 ± 0.007	0.984 ± 0.002	0.969 ± 0.004
	$436 \times 16 \times 3$	no	0.987 ± 0.001	0.981 ± 0.002	0.964 ± 0.003
	$436 \times 16 \times 3$	yes	0.989 ± 0.000	0.984 ± 0.000	0.968 ± 0.001
FCN model with whole image as input	$436 \times 384 \times 1$	no	0.993 ± 0.000	0.989 ± 0.001	0.979 ± 0.001
	$436 \times 384 \times 1$	yes	0.994 ± 0.000	0.992 ± 0.000	0.983 ± 0.001
	$436 \times 384 \times 3$	no	0.989 ± 0.001	0.983 ± 0.002	0.968 ± 0.004
	$436 \times 384 \times 3$	yes	0.990 ± 0.000	0.985 ± 0.001	0.971 ± 0.001

improved results only when the model was trained without data augmentation, with a small improvement of 3%. When using data augmentation no improvement in the results for the 3-D approach was observed. The third and final method, which used the whole image as an input to the network, achieved better results than the two previous ones, with the best pixel ACC score of 0.994 ± 0.000 from the 2-D approach with data augmentation. The 3-D approach results in lower scores in the case of this method, which proved that these additional scans did not provide supporting information for the U-Net architecture.

To further present how the various DL algorithms dealt with the actual

Table 3.5: Mean values together with standard deviations of the distance metrics and results from inter-grader annotations. DA indicates data augmentation.

method	input size (pixels)	DA	MAE	RMSE	MDAE	ME
inter-grader	$436 \times 384 \times 1$	no	3.12	3.90	2.50	-0.24
DNN model with A-scan as input	$436 \times 1 \times 1$	no	6.84± 0.61	14.33± 2.25	3.33± 0.29	1.07± 0.41
	$436 \times 1 \times 1$	yes	7.59± 0.63	14.78± 1.11	3.33± 0.29	1.61± 0.27
	$436 \times 1 \times 3$	no	5.39± 1.68	9.33± 6.28	3.67± 0.14	2.23± 2.12
	$436 \times 1 \times 3$	yes	7.19± 1.84	13.26± 5.29	3.17± 0.14	0.58± 2.32
CNN model with image patches as input	$436 \times 16 \times 1$	no	4.75± 0.24	6.09± 0.58	4.00± 0.00	−0.35± 0.04
	$436 \times 16 \times 1$	yes	4.67± 0.24	6.09± 0.58	4.00± 0.00	−0.56± 0.06
	$436 \times 16 \times 3$	no	2.55± 0.21	15.10± 0.12	1.83± 0.29	−0.93± 0.27
	$436 \times 16 \times 3$	yes	2.25± 0.06	14.82± 0.11	1.50± 0.00	−0.99± 0.12
FCN model with whole image as input	$436 \times 384 \times 1$	no	1.43± 0.04	2.25± 0.010	1.00± 0.00	−0.54± 0.12
	$436 \times 384 \times 1$	yes	1.15± 0.01	2.33± 0.39	1.00± 0.00	−0.44± 0.13
	$436 \times 384 \times 3$	no	2.02± 0.02	4.34± 1.17	1.00± 0.00	−0.42± 0.17
	$436 \times 384 \times 3$	yes	1.87± 0.10	3.49± 0.32	1.00± 0.00	−0.41± 0.17

estimation of the BMO points, which are the clinically relevant markers, it was decided to extract these points from the output masks and calculate the distance metrics. These metrics were calculated based on the difference between annotations and extracted values and they include *mean absolute error* (MAE), *root mean square error* (RMSE), *median absolute error* (MDAE), and *mean error* (ME). The results obtained with this calculation are presented in Table 3.5 and they support the conclusions obtained from the previous analysis. Additionally, based on the RMSE value, it can be seen that the whole-image method produces the fewest outliers, also showing MDAE values of around one pixel, which is very low.

Table 3.6: Mean values together with standard deviations of the distance metrics and results from inter-grader annotations.

modification	DSC	JSC	ACC	MAE	RMSE	MDAE	ME
none	0.992± 0.000	0.983± 0.001	0.994± 0.000	1.15± 0.01	2.33± 0.39	1.00± 0.00	−0.44± 0.13
residuals	0.986± 0.001	0.971± 0.001	0.990± 0.000	2.12± 0.07	3.56± 1.01	1.66± 0.29	−1.08± 0.17
RNN	0.984± 0.001	0.970± 0.001	0.990± 0.001	2.29± 0.01	3.88± 0.07	1.50± 0.00	−1.00± 0.19
SE	0.989± 0.001	0.973± 0.001	0.990± 0.001	2.17± 0.06	3.73± 2.14	2.00± 0.00	−1.07± 0.08

Additionally, some modifications to the best model obtained in the first two experiments were made. These include additional components in the FCN architecture including Residual, SE, and RNN methods. For more detail see the previous chapter. Results from that are shown in Table 3.6. While comparing the different modifications to the FCN method, the standard non-modified architecture achieved the best results. The other architectures also showed high accuracy with marginally inferior performance. However, overall, all could be successfully used in this task. Although given the size of the dataset we cannot confirm this hypothesis, one might suppose that this small decrease in performance is due to the more complex model having additional layers/parameters and needing a larger amount of training data.

The last part of this study considers one possible clinical application of using such a model. From the set of segmented BMO points, the shape of the ONH can be visualized. This ONH can be used to extract further structural metrics that may help to track changes due to glaucoma progression. Knowing that the semantic segmentation in the 2-D version with data augmentation obtains the best ACC, it was decided to use this model in further analysis. However, the model needed additional training, as previously it had only been taught on scans where BMO was always present. For this additional analysis, this model was fine-tuned with additional scans (10 of them for each training subject) that do not have BMO but only a background class. Fine-tuned model achieved pixel ACC of 0.9894, which is comparable to previous performance. To evaluate the shape of BMO, each scan from the testing set was processed from which two BMO points were extracted (if present), then all these points were collected together to estimate the shape of the ONH. For comparison, the manually annotated points were also used to estimate the ONH shape. Two metrics are used here to evaluate this algorithm: DSC and the *unsigned border positioning error* (UBPE). The second metric was selected because it is commonly used in other studies that estimate ONH shape and it is expressed in μm. Without any post-processing, the model achieved DSC of 0.953 ± 0.030 and UBPE of $11.4 \pm 51.9\mu m$. Additionally, one post-processing step was tested which consisted in modeling of the shape of the BMO with a convex hull algorithm, which results in 0.959 ± 0.032 DSC and $9.8 \pm 31.9\mu m$ UBPE. The approach with post-processing obtained better results which shows the advantage of using it.

Additionally, some details about computational time and the number of trainable parameters in each of these models could be found directly in the publication [1].

Summarizing the obtained results, it was possible to show that BMO segmentation using DL algorithms and a small dataset obtains very good accuracy, with very small error metrics, such as MDAE of 1 pixel which exceeds the performance obtained by the inter-grader. Additionally, segmentation by FCN, which uses the whole image, has proven to be the most accurate for this problem. This method also simplifies the data preparation steps since there is no need to split the data beforehand and the raw data can be used. This superior performance indicates that for this particular problem, having access to the whole image (information) is key for the DL method to provide a more accurate location of the BMO boundaries. In addition, at the time of performing these tests, the proposed model obtained the highest (in the authors' knowledge) results for both DSC and UBPE for determining the shape of ONH, when compared to previously published studies. That said these results cannot be compared directly because the studies use different datasets. The findings also show the benefits (improve performance) when using additional scans or data augmentation, which is relevant for the low-data regime problems assessed in this project.

Table 3.7: Mean and standard deviation BAC values for all of the considered methods based on feature selection for four different base classifiers (MLP, k-NN, DTC, SVC). The results are presented for different sizes of feature subspace. The number below the balanced accuracy value shows to which model, the model that this BAC belongs to obtained statistically significantly superior results (based on a t-test with non-parametric correction).

sub-space size	k-best				Random Subspace				proposed method			
	MLP	k-NN	DTC	SVC	MLP	k-NN	DTC	SVC	MLP	k-NN	DTC	SVC
	1	2	3	4	5	6	7	8	9	10	11	12
9	.781± .062 5,9	.807± .056 5, 6, 9	.794± .052 5, 9	.850± .057 3, 5–7, 9	.675± .048 —	.721± .063 —	.746± .065 —	.831± .059 5–7, 9	.686± .029 —	.849± .064 5–7, 9	.836± .054 5–7, 9	.872± .055 1–3, 5–9
18	.668± .061 —	.742± .065 —	.782± .059 1	.833± .055 1, 2, 5, 6	.736± .048 1	.723± ±.063 —	.814± .062 1, 5, 6	.853± .060 1–3, 5, 6, 9	.786± .046 1, 2	.847± .055 1, 2, 5, 6, 9	.832± .058 1–3, 5, 6	.868± .055 1–3, 5, 6, 9
30	.717± .068 —	.694± .066 —	.766± .058 —	.813± .049 1, 2, 5, 6	.712± .070 —	.723± .073 —	.835± .055 1, 2, 5, 6	.859± .056 1–3, 5, 6	.832± .052 1, 2, 5, 6	.849± .050 1–3, 5, 6	.819± .062 2, 5, 6	.861± .060 1–3, 5, 6
39	.712± .063 —	.694± .056 —	.764± .061 2	.810± .059 1, 2, 6	.752± .059 —	.723± .071 —	.830± .060 1, 2, 5, 6	.862± .058 1–3, 5, 6	.840± .061 1, 2, 5, 6	.847± .056 1, 2, 5, 6	.814± .060 1, 2	.853± .068 1–3, 5, 6
48	.616± .043 —	.634± .061 —	.605± .055 —	.681± .066 1, 3	.719± .042 1–3	.721± .066 1, 3	.833± .054 1–6	.849± .054 1–6	.846± .052 1–6	.852± .052 1–6	.814± .059 1–6	.861± .069 1–6

3.3 Dealing with the curse of dimensionality – feature selection

In the course of doctoral studies, not only image classification tasks were considered. Since ophthalmologists often based their diagnosis on biomarker values (such as thickness or intraocular pressure) obtained during various clinical tests, it was decided to also classify such biomarker values using standard machine learning algorithms. In this study, a total of 48 biomarker values are available for each participant. And these participants belonged to three groups, (i) healthy, (ii) glaucoma, and (iii) glaucoma suspects, this last group represents a real clinical challenge. Because the dataset was relatively small (around 70 participants per group) and there were many

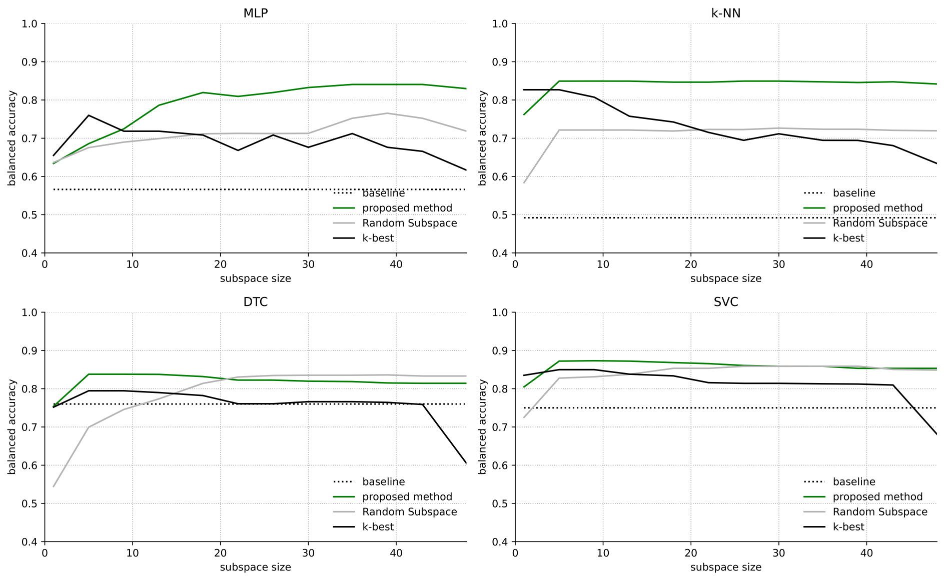


Figure 3.1: The relationship between mean BAC score and the size of feature subspace for all of the considered methods.

features for each participant (48 biomarkers), it was decided to focus not only on the classification problem itself but also on the methods of selecting features that would improve the results without rejecting any information. For comparison, already existing methods were tested, as well as the baseline – which was a single classifier trained on the entire dataset. BAC was utilized as a metric to evaluate the performance of the considered methods. Results of BAC from the baseline experiment for each classifier are as followed: 0.566, 0.493, 0.760, and 0.750 for MLP, k-NN, DTC, and SVC, respectively. To check which method obtained statistically significantly better results, a t-test with non-parametric correction was employed.

Table 3.7 presents mean values with standard deviations BAC calculated across cross-validation folds. The results in the table include all the considered methods, except baseline, because for this baseline method the subspace size was kept constant and equal to 48 (maximum feature count). Comparing the results obtained for the proposed method with the baseline results, it can be seen that for each subspace size, even for only 9 features, the proposed one always presents a superior accuracy. Concluding from the table, it can be noticed that for a subspace size of 18 and above, none

Table 3.8: Mean and standard deviations BAC values calculated across all base classifiers for each considered method. The number that is placed below the balanced accuracy value shows to which model, the model that this BAC belongs to obtained better and statistically significantly different results (based on a t-test with non-parametric correction).

subspace size	k-best	Random Subspace	proposed method
	1	2	3
9	0.808 ± 0.044	0.743 ± 0.048	0.811 ± 0.042
	2	—	2
18	0.756 ± 0.044	0.781 ± 0.043	0.833 ± 0.043
	—	—	1, 2
30	0.748 ± 0.034	0.782 ± 0.046	0.840 ± 0.046
	—	—	1, 2
39	0.745 ± 0.038	0.792 ± 0.050	0.839 ± 0.049
	—	1	1, 2
48	0.634 ± 0.048	0.780 ± 0.041	0.843 ± 0.047
	—	1	1, 2

of the other methods obtained a statistically superior performance to the proposed one. Additionally, for the lower subspace size only the proposed method, with an MLP as a base classifier, presents an inferior accuracy. The rest of the variations of the method show comparable or superior results to the other feature selection methods. Figure 3.1 presents the same results in a graphical form, comparing BAC score versus feature subspace size. It can be observed that BAC remains stable with the proposed method when changing the feature subspace size. This brings an additional advantage to the method as there is no need to perform time-consuming hyperparameter tuning. With a method like *k*-best, optimizing these hyperparameters is especially important to get good results. Similar results to the proposed method are obtained using Random Subspace, but it is worth mentioning here that the proposed method achieves these results already for much smaller sizes of selected subspaces, which has the added advantage of speeding up the learning process.

The mean values of all results from the considered method are presented in Table 3.8. They were calculated across all base classifiers. From the results obtained one can notice the advantage of the proposed method.

Table 3.9: Mean values together with a standard deviation of BAC calculated across all considered methods for each of the base classifiers. The number that is placed below the balanced accuracy value shows to which model, the model that this BAC belongs to obtained better and statistically significantly different results (based on a t-test with non-parametric correction).

subspace size	MLP 1	k-NN 2	DTC 3	SVC 4
9	$0.714 \pm$ 0.033 —	$0.793 \pm$ 0.051 1	$0.793 \pm$ 0.044 1	$0.851 \pm .051$ 1,2,3
18	$0.730 \pm$ 0.035 —	$0.771 \pm$ 0.048 —	$0.809 \pm$ 0.050 1, 2	$0.852 \pm$ 0.050 1, 2
30	$0.754 \pm$ 0.044 —	$0.755 \pm$ 0.050 —	$0.803 \pm$ 0.049 —	$0.841 \pm$ 0.053 1, 2
39	$0.768 \pm$ 0.051 —	$0.755 \pm$ 0.050 —	$0.803 \pm$ 0.049 —	$0.841 \pm$ 0.053 1, 2
48	$0.727 \pm$ 0.034 —	$0.735 \pm$ 0.044 —	$0.751 \pm$ 0.042 —	$0.797 \pm$ 0.048 1, 2, 3

For the feature subspace size greater than 9, it obtained statistically better results than the other ones.

Additionally, to check which base classifier achieves the highest BAC, the mean value was calculated across all the considered methods (see Table 3.9). Results show that SVC is the best choice for this particular problem, providing a superior BAC value.

To summarize this study, the proposed method presents a superior performance in comparison to the available methods of selecting features. Its main advantage is the fact that it does not discard any information (biomarkers) at the beginning – reducing or completely removing the rejection of features. This method only increases the probability of using features that better differentiate the classes. With the proposed method, it was possible to solve the problem under consideration, i.e. the classification of three glaucoma groups with a BAC equal to 0.872, which is a very promising performance, especially considering the data includes a group of glaucoma suspects, that may develop glaucoma in the future. In addition, the fact that the changes in accuracy are small over the size of the feature subspace, training of this method can be accelerated, and the process of optimizing the number of features does not need to be time-consuming.

3.4 Orthogonal convolutional neural network

The study was to assess whether the use of OCNN can improve the performance of the DL method that has been trained on small datasets, and compare the result with larger training sets. This project was divided into two separate experiments: one for classification with the CIFAR-10 dataset, and the other one for segmentation with OCT images. For evaluation purposes, the ACC metric was selected. In both experiments, each model was trained three times, from which mean values are presented here.

3.4.1 Classification

Two approaches to the classification of CIFAR-10 datasets were evaluated. First, using the entire dataset, and second, using all possible combinations of two classes from the dataset. The second approach was considered because it could show a different angle of the problem, reducing it to a binary classification problem. Fig. 3.2 and 3.3 presents different aspects of the performance for the non-binary approach. In Fig. 3.2, Top-1 ACC from the last epoch was calculated for each model, and the relationship between ACC and training set size is shown. The value of ACC for the full dataset is equal to 92.20 and 92.85, for CNN and OCNN approaches, respectively. In each plot point, the ACC values obtained using the OCNN model were higher than the standard CNN. This difference between the method was more obvious with the smaller dataset size, which supports the concept that using orthogonalization is particularly useful with the low-data regime. The largest difference is obtained for the training set size of 10% of the original dataset, with an ACC difference of 9.3%. Although as expected the ACC deteriorates for the reduced dataset, this difference between the method is still evident. In Fig. 3.3, the training curve for each data size and these two

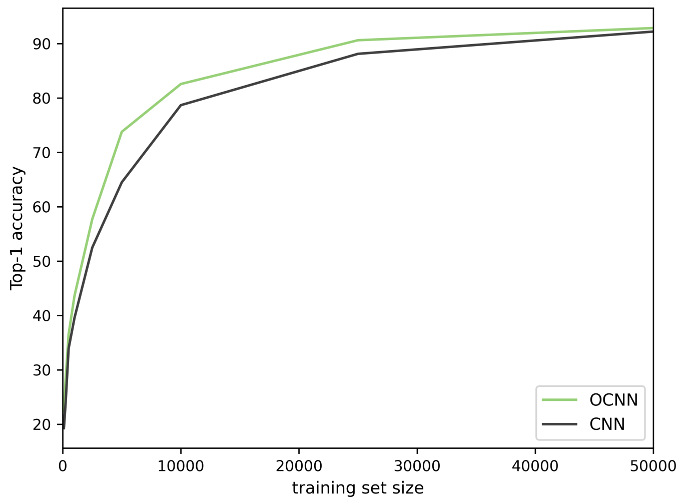


Figure 3.2: The relationship between the Top-1 accuracy and the size of the training set (from CIFAR-10 set). Two architectures are compared: with and without orthogonalization. The green line represents the OCNN and the black line represent a standard CNN architecture. Nine different sizes of the training set were extracted from the original one and evaluated.

approaches is presented, which shows the relationship between validation ACC for the different training epochs. In most cases, from the very beginning of the training process, an improvement when using the orthogonal model can be noticed. While the differences in ACC are not so noticeable with a larger dataset, for reduced ones the OCNN provides a very significant improvement in the performance of the entire classifier.

The second approach considered training several models using a reduced dataset that contained only two classes. A combination of all possible two-classes sets was trained and an average across all combinations is shown. The results are presented in a similar way to that of the non-binary classification. Fig. 3.4 shows the average from all combinations at given dataset sizes. Similar to previous results, the difference between the model with and without orthogonalization is larger for a smaller dataset. More precisely these differences can be seen in Fig. 3.5. The largest difference can be seen with a training set size equal to 500 (5% of the original dataset) with a difference of 22.8%. However, even for a smaller set this difference is substantial

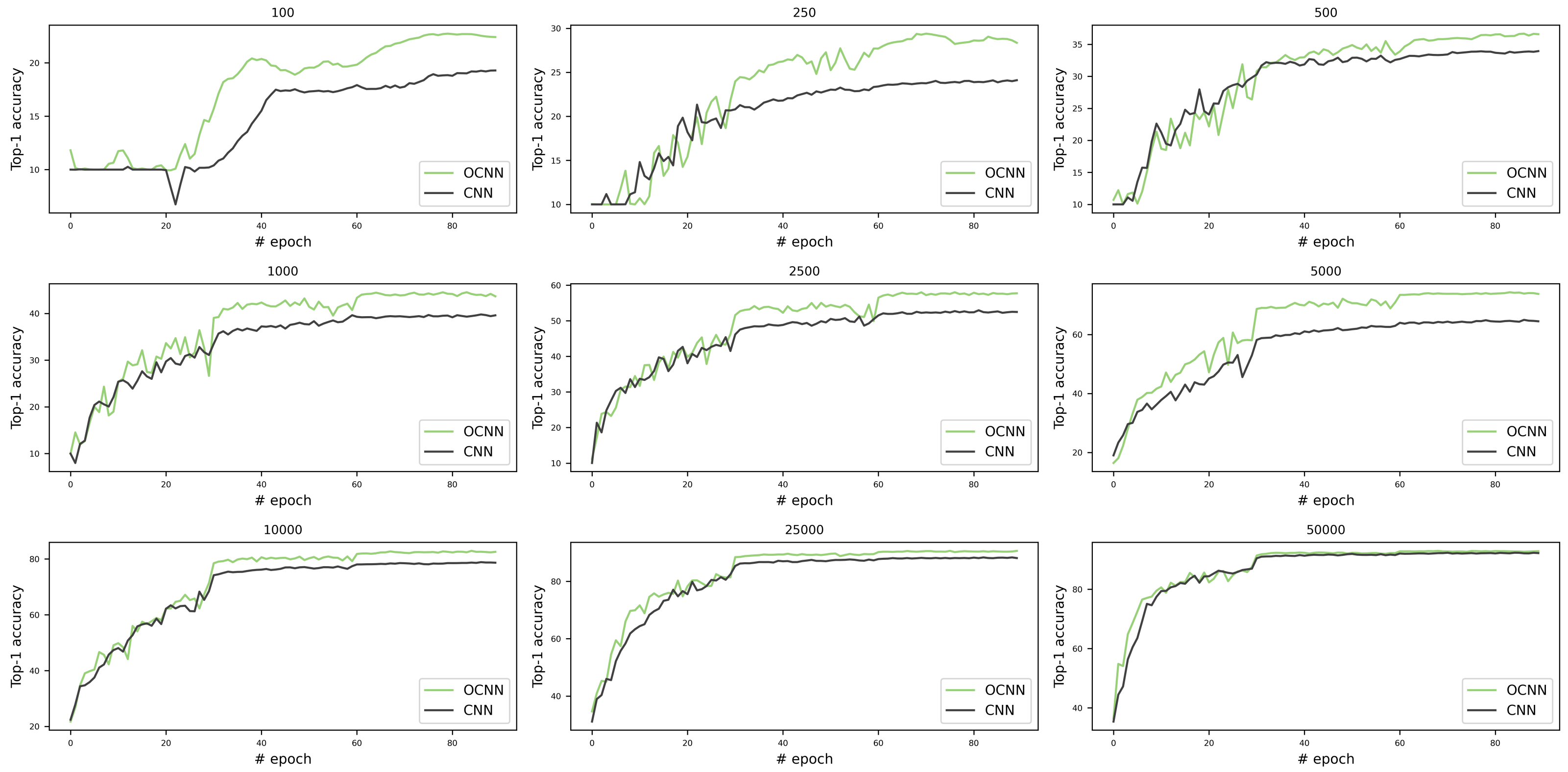


Figure 3.3: The relationship between the Top-1 accuracy and the epoch number. Two architectures are compared: with and without orthogonalization for nine different sizes of a dataset, each in a separate plot. The size of the used dataset is shown as a title of a specific plot. The green line represents the OCNN and the black line represents a standard CNN architecture.

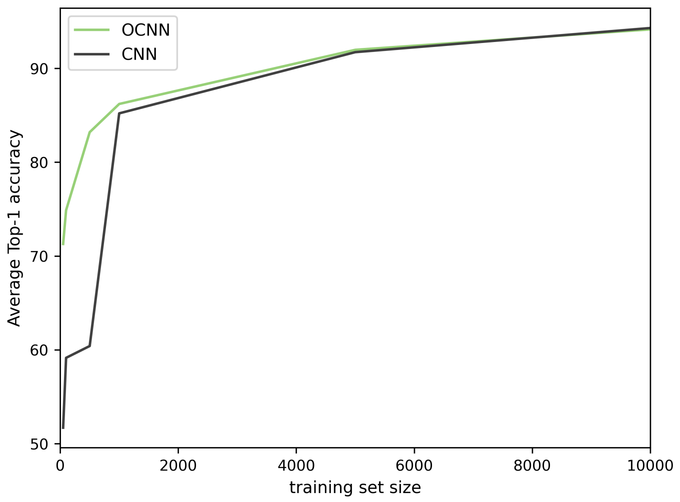


Figure 3.4: The relationship between the averaged Top-1 accuracy over different two-classes combinations from the dataset and the size of the training set. Two architectures are compared: with and without orthogonalization. The green line represents the OCNN and the black line represents a standard CNN architecture. Six different sizes of the training set were extracted from the original one and evaluated.

and is equal to 15.7% and 19.6% for 100 and 50 objects in the training set, respectively. If the smallest tested dataset size (50) is considered, the standard CNN architecture achieved a random classification performance of 51.7%, but with the use of orthogonalization this metric rises to 71.3%, which is no longer a random classification.

3.4.2 Segmentation

Experiments for segmentation were carried out similarly. An original dataset was randomly reduced into four different sizes, ensuring that each model is trained with the same dataset. The sizes of training datasets are as followed: 4439 (entire dataset), 1000, 400, and 100 images. Fig. 3.6 presents pixel accuracy that was achieved after the last epoch of training for all of these dataset sizes. All the results provide a very high ACC with values above 96%, therefore the differences between the individual methods are not that substantial. However, for each dataset size, the method using additional orthogonalization performed better, with the largest difference

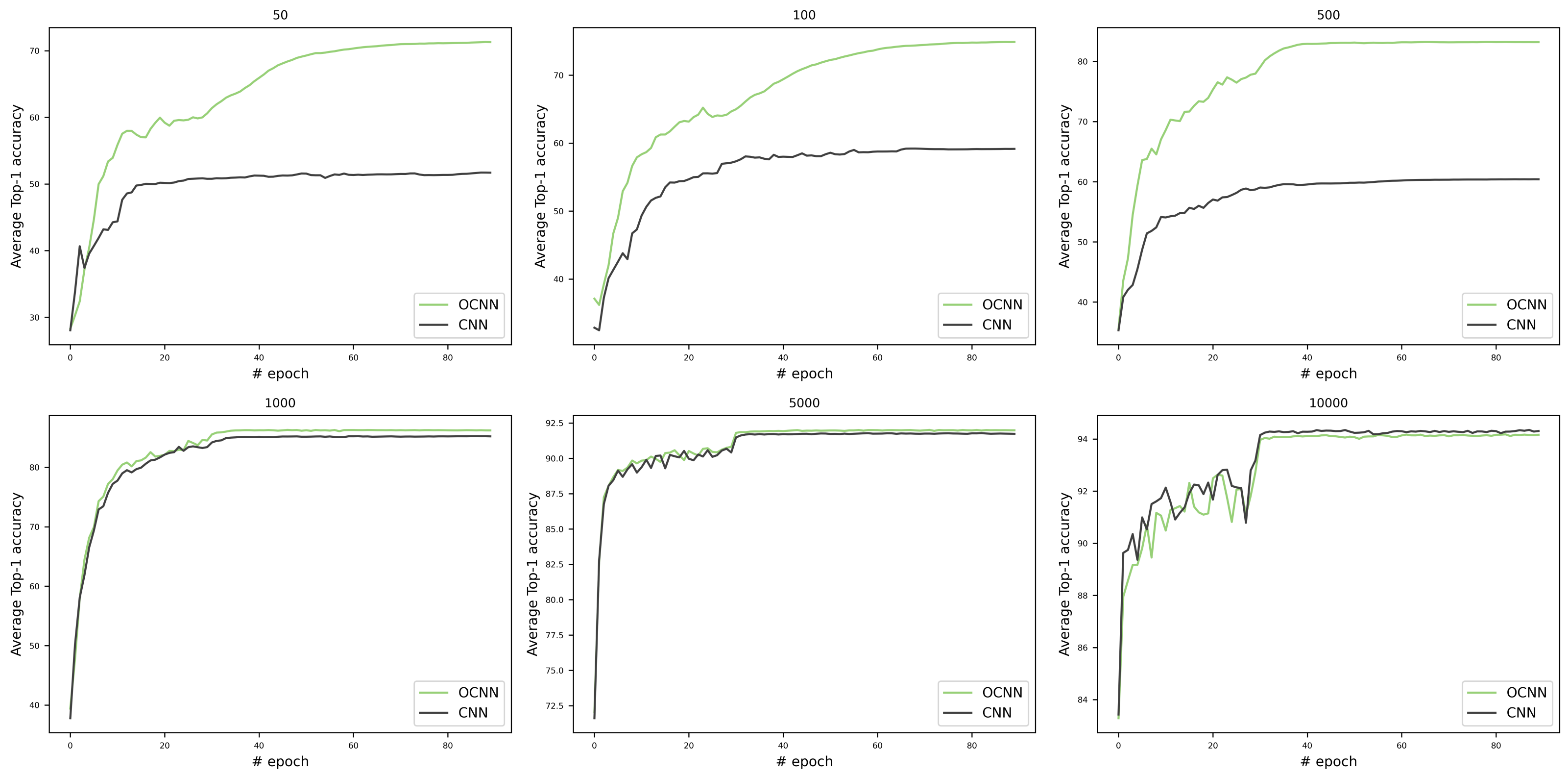


Figure 3.5: The relationship between the averaged Top-1 accuracy over different two-classes combinations from the dataset and the epoch number. Two architectures are compared: with and without orthogonalization for six different sizes of a dataset, each in a separate plot. The size of the used dataset is shown as a title of a specific plot. The green line represents the OCNN and the black line represents a standard CNN architecture.

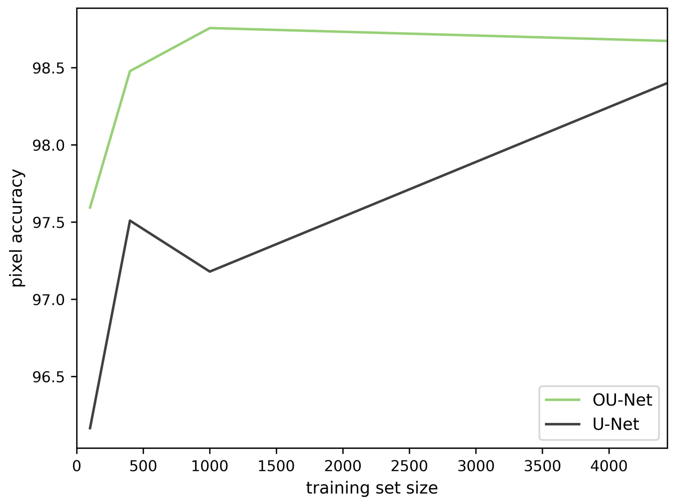


Figure 3.6: The relationship between the pixel accuracy and the size of the training set. Two architectures are compared: with and without orthogonalization. The green line represents the orthogonal U-Net (OU-Net) and the black line represents a standard U-Net architecture. Four different sizes of the training set were extracted from the original one and evaluated.

between models observed while training with the smaller datasets, 1.6 percent of difference for 25% of the original data and 1.4 percent for about 2.5% of the original dataset. Thus, the results for the segmentation task are similar to those for the classification task, indicating that the orthogonal networks achieve a larger improvement in accuracy for the low-data regime. That said, in Fig. 3.7 it can be seen that the training curve behaves not the same as for the classification task. Here, the method with orthogonalization takes more time (epochs) to reach a good result for dataset sizes 100 and 400 than the method without orthogonalization. However, the orthogonalization method achieves a higher ACC in the end. It is important to remember that this is a validation ACC which may not translate directly into testing ACC.

Summarizing these experiments, without focusing on the specific values of the results, it can be seen that the additional orthogonalization added to the convolutional networks is useful and improves the results for both smaller and larger datasets. However, for low-data regime cases, any improvement in the accuracy while training on the small dataset is particularly

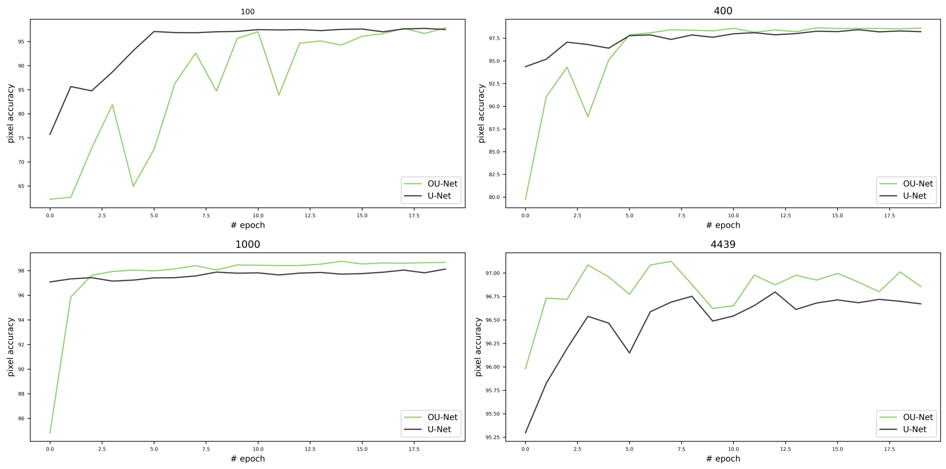


Figure 3.7: The relationship between the pixel accuracy and the epoch number. Two architectures are compared: with and without orthogonalization for four different sizes of a dataset, each in a separate plot. The size of the used dataset is shown as a title of a specific plot. The green line represents the orthogonal U-Net (OU-Net) and the black line represents a standard U-Net architecture.

important. While the orthogonal network shows a performance improvement while using the entire dataset, the results demonstrate these benefits (accuracy improvement) are more significant while training with small datasets. These conclusions were confirmed by the results for both the classification task and the segmentation task for the publicly available datasets.

Chapter 4

Conclusions and future directions

This dissertation presents a range of different studies and their corresponding results concerning machine learning and deep learning methods for the diagnosis of eye diseases. All experiments were carried out in a so-called low-data regime, which indicates the amount of data available to train the model is relatively low. Working with small datasets is particularly challenging, especially in the context of machine learning algorithms that generally requires a relatively large number of samples. It is worth noting that typically the performance of machine learning / deep learning problem is closely related to the number of images used to train the model (i.e., more data tends to provide superior performance). However, in many situations collecting the optimal amount of data is not possible and therefore working with such a limited dataset may be necessary. In many medical applications, including the ophthalmic applications considered in this thesis, the available datasets can be often small, but there is still a need for a better diagnostic tool to be developed and additionally also serve and support the patients and clinicians that could also benefit from such tools. This work focuses on eye diseases, however, the presented methods, findings, and conclusions can be translated into other disciplines and other medical diagnostic tasks. The study was divided into four main experiments, the first two experiments focus on two popular machine learning tasks that included image classification and segmentation in a low-data regime. The other two experiments propose, and test solutions aimed at overcoming the problem of small datasets and making it possible to work on them. They include classification using additional feature selection methods, and the general approach to convolutional networks and a low-data regime. A brief summary of each experiment is provided below.

The first experiment, the classification task, considered the classification of two groups: a group of people diagnosed with glaucoma and a control (healthy) group. For this, scanning laser ophthalmoscopy images were utilized, which are typically not used for diagnostics purposes, but its main utility is to serve as a positioning reference for the OCT imaging. During this experiment, several approaches for handling small datasets were implemented and tested and the conclusions show that the best solution is to test different approaches to deal with the low-data regime because there is no single approach that would work for all cases, as in the case of our data, a simpler and smaller model architecture with ensemble learning allows for better results than the well-known pre-trained architecture with fine-tuning. Additionally, the obtained results show that SLO images contain useful information that can be used for the diagnosis of glaucoma, and they can be successfully used in this context, either individually or in conjunction with other imaging modalities or biomarkers. Thus, adding another possible source of information for diagnosis.

The second experiment of this research projects concerned the task of Bruch's membrane opening segmentation. Two main goals of this experiment were: (i) to perform the segmentation task with the best possible result and (ii) to understand the impact which input data size (and the corresponding architecture) has on the performance of the network. Additional information from adjacent scans (3D strategy) and data augmentation was also tested. The results clearly showed that the U-Net architecture which provides a fully semantic segmentation obtained the best accuracy in this task. This network is fed with the whole image and when combined with data augmentation, obtained a mean absolute error of around 1 pixel for determining the opening position of the Bruch's membrane. This superior performance may be related to the need to have additional contextual information (whole image) for the network to make a good prediction on the Bruch's membrane opening location. Additionally, procedures such as data augmentation can further increase the final accuracy of the model. Dividing the data into smaller parts to feed them into the model during training, as in the case of the deep neural network (with A-scan) and the convolutional neural network (with patches) methods, generates a larger training set. However, in this case, the selection of the appropriate architecture (with the use of fully convolutional network architecture) allowed for better results. In the case of other architectures and input data sizes, the positive impact of information from additional scans on the learning process was noticeable in the method using A-scans (single columns from B-scans).

The third experiment of the studies concerned the approach to a slightly different problem of small datasets, which has been called the curse of dimensionality. In this case, the dataset was numerical and composed of different ocular biomarkers that included values such as retinal thickness and ocular pressure. The subjects were categorized into three classes: a control group, a glaucoma group, and a glaucoma suspect group, this last one included participants suspected to develop glaucoma in the future. Not only is the training dataset small and includes a challenging group (glaucoma suspects), but in addition, the number of features for each participant is relatively large compared to the size of the set (211 participants in total, 48 features each participant). Given that these features are being used by ophthalmologists in the diagnosis process, it is assumed that each of these features is important, and removing some of them during the analysis process may lead to the loss of potentially important information. Therefore, it was decided to focus both on the task of classifying data into appropriate groups and to develop a method for feature selection, which allows the use of most or all the features while favoring the features that differentiate the classes. The results showed that the proposed method is superior to all standard feature selection methods. It was tested with several standard classifiers, which makes this method a universal preprocessing step that could be adopted in other studies.

The fourth and final experiment of the study focused on the use of orthogonalization while working with small datasets. Previous work proved that architecture using this method improves performance with a normal dataset and, in addition, the filters of trained convolutional layers are less similar to each other (less redundancy and more diversity). Considering these benefits of orthogonalization, it was decided to explore the use of this method to work with small datasets for the classification and segmentation task. In both tasks, which include classification of the CIFAR-10 dataset and segmentation of retinal layers, the obtained results proved that the use of this approach confirmed the previous findings of performance improvement. However, these performance improvements were more significant with smaller datasets. Therefore, it may be one of the additional approaches to be used while working with small datasets using convolutional networks.

All the experiments performed in this study proved that it is possible to work with small sets of medical data. Overall, there is no ’one-size-fits-all’ solution that will work for every low-data regime problem, so it is likely that different experiments need to be performed to find the right solution for a given problem. One of the conclusions from the obtained results is that custom smaller task-specific architectures may perform better than standard large ones, even if this standard network has been pre-trained and transfer learning is used. Additionally, the use of orthogonalization, which is not typically considered in the deep learning problem, seems to provide a performance improvement for convolutional networks. So, this regularization method can be also a good strategy for dealing with small datasets.

The findings of this study constitute a good basis for the further development of the research in the future. A natural extension of this research is the work on a multimodal system that would combine different datasets and, based on them and the results obtained so far, combine the information contained in them to more accurately support the diagnosis of eye diseases. In the case of the last experiment, additional work could be performed to understand the influence of the α parameter on the results of both classification and segmentation on small datasets. This α parameter affects how much the orthogonal component is weighted within the standard loss function (i.e., the orthogonal contribution to the loss). Such an experiment would be aimed at checking whether there is a correlation between this parameter and the size of the dataset or whether the α may need to be treated as a hyperparameter and needs to be tuned during the training process.

The study shows a range of experiments that were performed to prove the four main hypotheses postulated at the beginning of the dissertation. For the first hypothesis, which covers the use of Scanning Laser Ophthalmoscopy images combined with deep learning as a tool for the diagnosis of glaucoma, the classifier trained using these images achieved a balanced accuracy metric of over 96 percent, which is strong evidence that this imaging modality can support the diagnosis of glaucoma. The second hypothesis, which looks at the effect of data size on performance, shows that the size of the input data fed to machine learning models has an impact on the final training result. In the case of segmentation of the opening of the Bruch's membrane, the best result was obtained using the entire spatial information (i.e., the entire image) without dividing the original image into smaller components. The findings from all experiments support the third hypothesis, which is related to the application of machine learning methods in low data regime scenarios. Overall, it was possible to obtain a good performance for the models. Nevertheless, additional techniques or different processing approaches may be required to obtain the high performance of utilized models, which may be application dependent. It should be noted that two new methods to work in conditions of small sets of training data in machine learning application have been proposed. These methods have been shown to achieve superior results compared to standard methods. The first method deals with the curse of dimensionality, i.e., with a high-dimensional and small set of data, using a classifier ensemble and focusing on operating on all available information. The second method, which is related to the fourth and final hypothesis, covers the use of orthogonalization with convolutional networks. The obtained conclusions demonstrate that the use of orthogonalization with these networks provides an improvement in performance, which is particularly significant in the case of a small training dataset. This is a premise that orthogonalization can be effectively used when training machine learning algorithms under low-data regime conditions. Overall, the findings supported all four main hypotheses.

Bibliography

[1] D. Sułot, D. Alonso-Caneiro, D. R. Iskander, M. J. Collins, Deep learning approaches for segmenting bruch's membrane opening from oct volumes, OSA Continuum 3 (12) (2020) 3351–3364.

[2] D. Sułot, D. Alonso-Caneiro, P. Ksieniewicz, P. Krzyzanowska-Berkowska, D. R. Iskander, Glaucoma classification based on scanning laser ophthalmoscopic images using a deep learning ensemble method, PloS One 16 (6) (2021) e0252339.

[3] D. Sułot, P. Zyblewski, P. Ksieniewicz, Analysis of variance application in the construction of classifier ensemble based on optimal feature subset for the task of supporting glaucoma diagnosis, in: International Conference on Computational Science, Springer, 2021, pp. 109–117.

[4] D. Sułot, Selection of interpretable decision tree as a method for classification of early and developed glaucoma, in: IEEE EMBS International Student Conference, Springer, 2020, pp. 144–150.

[5] D. Sułot, P. Zyblewski, P. Ksieniewicz, A novel approach to learning and designing neural networks-based ensemble, in: P. Ksieniewicz, M. Uchroński (Eds.), Selected model based architectures and algorithms for learning, signal processing and optimization, Akademicka Oficyna Wydawnicza EXIT, 2021, pp. 103–112.

[6] D. Sulot, D. Alonso-Caneiro, P. Krzyzanowska-Berkowska, D. R. Iskander, Differentiating glaucoma patients from healthy controls and

glaucoma suspects using deep learning, Investigative Ophthalmology & Visual Science 61 (7) (2020) 1648–1648.

[7] J. Loo, L. Fang, D. Cunefare, G. J. Jaffe, S. Farsiu, Deep longitudinal transfer learning-based automatic segmentation of photoreceptor ellipsoid zone defects on optical coherence tomography images of macular telangiectasia type 2, Biomedical Optics Express 9 (6) (2018) 2681–2698.

[8] M. Flasiński, Introduction to artificial intelligence, Springer, 2016.

[9] P. McCorduck, C. Cfe, Machines who think: A personal inquiry into the history and prospects of artificial intelligence, CRC Press, 2004.

[10] J. Weizenbaum, Eliza—a computer program for the study of natural language communication between man and machine, Communications of the ACM 9 (1) (1966) 36–45.

[11] T. Kohonen, Self-organized formation of topologically correct feature maps, Biological Cybernetics 43 (1) (1982) 59–69.

[12] V. Kaul, S. Enslin, S. A. Gross, History of artificial intelligence in medicine, Gastrointestinal Endoscopy 92 (4) (2020) 807–812.

[13] C. Butler, IEEE First International Conference on Neural Networks, Sheraton Harbor Island East, San Diego, California, June 21-24, 1987, Vol. 2, SOS Print., 1987.

[14] I. H. Sarker, Machine learning: Algorithms, real-world applications and research directions, SN Computer Science 2 (3) (2021) 1–21.

[15] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning internal representations by error propagation, Tech. rep., California Univ San Diego La Jolla Inst for Cognitive Science (1985).

[16] F. Rosenblatt, The perceptron: a probabilistic model for information storage and organization in the brain., Psychological Review 65 (6) (1958) 386.

[17] H. Zhang, The optimality of naive bayes, AA 1 (2) (2004) 3.

[18] A. Kelly, M. A. Johnson, Investigating the statistical assumptions of naïve bayes classifiers, in: 2021 55th Annual Conference on Information Sciences and Systems (CISS), IEEE, 2021, pp. 1–6.

[19] S. A. Dudani, The distance-weighted k-nearest-neighbor rule, IEEE Transactions on Systems, Man, and Cybernetics (4) (1976) 325–327.

[20] Y.-C. Liaw, M.-L. Leou, C.-M. Wu, Fast exact k nearest neighbors search using an orthogonal search tree, Pattern Recognition 43 (6) (2010) 2351–2358.

[21] C. Cortes, V. Vapnik, Support-vector networks, Machine learning 20 (3) (1995) 273–297.

[22] B. Ghaddar, J. Naoum-Sawaya, High dimensional data classification and feature selection using support vector machines, European Journal of Operational Research 265 (3) (2018) 993–1004.

[23] Y. Ma, G. Guo, Support vector machines applications, Vol. 649, Springer, 2014.

[24] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, MIT Press, 2016.

[25] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, B. C. Van Esesn, A. A. S. Awwal, V. K. Asari, The history began from alexnet: A comprehensive survey on deep learning approaches, arXiv preprint arXiv:1803.01164 (2018).

[26] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, Advances in Neural Information Processing Dystems 25 (2012) 1097–1105.

[27] K. Fukushima, Neocognitron: A hierarchical neural network capable of visual pattern recognition, Neural Networks 1 (2) (1988) 119–130.

[28] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (8) (1997) 1735–1780.

[29] S. Varsamopoulos, K. Bertels, C. Almudever, Designing neural network based decoders for surface codes accelerated bwa-mem view project hartes view project designing neural network based decoders for surface codes, no. November (2018) 1–12.

[30] D. P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114 (2013).

[31] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, Advances in Neural Information Processing Systems 27 (2014).

[32] N. Diamant, D. Zadok, C. Baskin, E. Schwartz, A. M. Bronstein, Beholder-gan: Generation and beautification of facial images with conditioning on their beauty level, in: 2019 IEEE International Conference on Image Processing (ICIP), IEEE, 2019, pp. 739–743.

[33] S. Vandenhende, B. De Brabandere, D. Neven, L. Van Gool, A three-player gan: generating hard samples to improve classification networks, in: 2019 16th International Conference on Machine Vision Applications (MVA), IEEE, 2019, pp. 1–6.

[34] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural computation 1 (4) (1989) 541–551.

[35] F. Yu, V. Koltun, Multi-scale context aggregation by dilated convolutions, arXiv preprint arXiv:1511.07122 (2015).

[36] V. Dumoulin, F. Visin, A guide to convolution arithmetic for deep learning, arXiv preprint arXiv:1603.07285 (2016).

[37] L. Bai, Y. Zhao, X. Huang, A cnn accelerator on fpga using depthwise separable convolution, IEEE Transactions on Circuits and Systems II: Express Briefs 65 (10) (2018) 1415–1419.

[38] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: International Conference on Machine Learning, PMLR, 2015, pp. 448–456.

[39] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[40] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.

[41] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 234–241.

[42] Y. Weng, T. Zhou, Y. Li, X. Qiu, Nas-unet: Neural architecture search for medical image segmentation, IEEE Access 7 (2019) 44247–44257.

[43] B. Baheti, S. Innani, S. Gajre, S. Talbar, Eff-unet: A novel architecture for semantic segmentation in unstructured environment, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 358–359.

[44] F. Provost, R. Kohavi, Glossary of terms, Journal of Machine Learning 30 (2-3) (1998) 271–274.

[45] S. Madden, From databases to big data, IEEE Internet Computing 16 (3) (2012) 4–6.

[46] Y. Landau, N. Kiryati, Dataset growth in medical image analysis research, arXiv preprint arXiv:1908.07765 (2019).

[47] M. Köppen, The curse of dimensionality, in: 5th Online World Conference on Soft Computing in Industrial Applications (WSC5), Vol. 1, 2000, pp. 4–8.

[48] N. Altman, M. Krzywinski, The curse (s) of dimensionality, Nat Methods 15 (6) (2018) 399–400.

[49] K. Pearson, Liii. on lines and planes of closest fit to systems of points in space, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2 (11) (1901) 559–572.

[50] T. K. Ho, The random subspace method for constructing decision forests, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (8) (1998) 832–844.

[51] L. Perez, J. Wang, The effectiveness of data augmentation in image classification using deep learning, arXiv preprint arXiv:1712.04621 (2017).

[52] S. Huang, X. Wang, D. Tao, Snapmix: Semantically proportional mixing for augmenting fine-grained data, arXiv preprint arXiv:2012.04846 (2020).

[53] K. Weiss, T. M. Khoshgoftaar, D. Wang, A survey of transfer learning, Journal of Big data 3 (1) (2016) 1–40.

[54] K. Chan, T.-W. Lee, P. A. Sample, M. H. Goldbaum, R. N. Weinreb, T. J. Sejnowski, Comparison of machine learning and traditional classifiers in glaucoma diagnosis, IEEE Transactions on Biomedical Engineering 49 (9) (2002) 963–974.

[55] T. MacGillivray, E. Trucco, J. Cameron, B. Dhillon, J. Houston, E. Van Beek, Retinal imaging as a source of biomarkers for diagnosis, characterization and prognosis of chronic illness or long-term conditions, The British Journal of Radiology 87 (1040) (2014) 20130832.

[56] S. Samanta, S. S. Ahmed, M. A.-M. M. Salem, S. S. Nath, N. Dey, S. S. Chowdhury, Haralick features based automated glaucoma classification using back propagation neural network, in: Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014, Springer, 2015, pp. 351–358.

[57] D. Yadav, M. P. Sarathi, M. K. Dutta, Classification of glaucoma based on texture features using neural networks, in: 2014 Seventh International Conference on Contemporary Computing (IC3), IEEE, 2014, pp. 109–112.

[58] S. Dua, U. R. Acharya, P. Chowriappa, S. V. Sree, Wavelet-based energy features for glaucomatous image classification, IEEE Transactions on Information Technology in Biomedicine 16 (1) (2011) 80–87.

[59] F. Fink, K. Worle, P. Gruber, A. Tome, J. Gorriz-Saez, C. Puntonet, E. Lang, Ica analysis of retina images for glaucoma classification, in: 2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, IEEE, 2008, pp. 4664–4667.

[60] B. Al-Bander, W. Al-Nuaimy, M. A. Al-Taee, Y. Zheng, Automated glaucoma diagnosis using deep learning approach, in: 2017 14th International Multi-Conference on Systems, Signals & Devices (SSD), IEEE, 2017, pp. 207–210.

[61] X. Chen, Y. Xu, S. Yan, D. W. K. Wong, T. Y. Wong, J. Liu, Automatic feature learning for glaucoma detection based on deep learning, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 669–677.

[62] J. M. Ahn, S. Kim, K.-S. Ahn, S.-H. Cho, K. B. Lee, U. S. Kim, A deep learning model for the detection of both advanced and early glaucoma using fundus photography, PloS One 13 (11) (2018) e0207982.

[63] R. Asaoka, M. Tanito, N. Shibata, K. Mitsuhashi, K. Nakahara, Y. Fujino, M. Matsuura, H. Murata, K. Tokumo, Y. Kiuchi, Validation of a deep learning model to screen for glaucoma using images from different fundus cameras and data augmentation, Ophthalmology Glaucoma 2 (4) (2019) 224–231.

[64] N. Shibata, M. Tanito, K. Mitsuhashi, Y. Fujino, M. Matsuura, H. Murata, R. Asaoka, Development of a deep residual learning algorithm

to screen for glaucoma from fundus photography, Scientific Reports 8 (1) (2018) 1–9.

[65] G. An, K. Omodaka, K. Hashimoto, S. Tsuda, Y. Shiga, N. Takada, T. Kikawa, H. Yokota, M. Akiba, T. Nakazawa, Glaucoma diagnosis with machine learning based on optical coherence tomography and color fundus images, Journal of Healthcare Engineering 2019 (2019).

[66] K. A. Thakoor, X. Li, E. Tsamis, Z. Z. Zemborain, C. G. De Moraes, P. Sajda, D. C. Hood, Strategies to improve convolutional neural network generalizability and reference standards for glaucoma detection from oct scans, Translational Vision Science & Technology 10 (4) (2021) 16–16.

[67] G. An, M. Akiba, K. Omodaka, T. Nakazawa, H. Yokota, Hierarchical deep learning models using transfer learning for disease detection and classification based on small number of medical images, Scientific Reports 11 (1) (2021) 1–9.

[68] P. Mehta, C. A. Petersen, J. C. Wen, M. R. Banitt, P. P. Chen, K. D. Bojikian, C. Egan, S.-I. Lee, M. Balazinska, A. Y. Lee, et al., Automated detection of glaucoma with interpretable machine learning using clinical data and multimodal retinal images, American Journal of Ophthalmology 231 (2021) 154–169.

[69] N. E. Benzebouchi, N. Azizi, A. S. Ashour, N. Dey, R. S. Sherratt, Multi-modal classifier fusion with feature cooperation for glaucoma diagnosis, Journal of Experimental & Theoretical Artificial Intelligence 31 (6) (2019) 841–874.

[70] Y. Li, Y. Han, X. Zhao, Z. Li, Z. Guo, Multinet: Multimodal neural networks for glaucoma based on transfer learning (2021).

[71] R. Chrástek, M. Wolf, K. Donath, G. Michelson, H. Niemann, Optic disc segmentation in retinal images, in: Bildverarbeitung für die Medizin 2002, Springer, 2002, pp. 263–266.

[72] P. Pallawala, W. Hsu, M. L. Lee, S. S. Goh, Automated microaneurysm segmentation and detection using generalized eigenvectors, in: 2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05)-Volume 1, Vol. 1, IEEE, 2005, pp. 322–327.

[73] Z. B. Sbeh, L. D. Cohen, G. Mimoun, G. Coscas, A new approach of geodesic reconstruction for drusen segmentation in eye fundus images, IEEE Transactions on Medical Imaging 20 (12) (2001) 1321–1333.

[74] R. M. Cesar Jr, H. F. Jelinek, Segmentation of retinal fundus vasculature in nonmydriatic camera images using wavelets, Angiography and Plaque Imaging (2003) 193–224.

[75] M. Niemeijer, J. Staal, B. van Ginneken, M. Loog, M. D. Abramoff, Comparative study of retinal vessel segmentation methods on a new publicly available database, in: Medical Imaging 2004: Image Processing, Vol. 5370, International Society for Optics and Photonics, 2004, pp. 648–656.

[76] H. Ishikawa, D. M. Stein, G. Wollstein, S. Beaton, J. G. Fujimoto, J. S. Schuman, Macular segmentation with optical coherence tomography, Investigative Ophthalmology & Visual Science 46 (6) (2005) 2012–2017.

[77] A. Yazdanpanah, G. Hamarneh, B. R. Smith, M. V. Sarunic, Segmentation of intra-retinal layers from optical coherence tomography images using an active contour approach, IEEE Transactions on Medical Imaging 30 (2) (2010) 484–496.

[78] K. Vermeer, J. Van der Schoot, H. Lemij, J. De Boer, Automated segmentation by pixel classification of retinal layers in ophthalmic oct images, Biomedical Optics Express 2 (6) (2011) 1743–1756.

[79] J. M. Gmeiner, W. A. Schrems, C. Y. Mardin, R. Laemmer, F. E. Kruse, L. M. Schrems-Hoesl, Comparison of bruch's membrane

opening minimum rim width and peripapillary retinal nerve fiber layer thickness in early glaucoma assessment, Investigative Ophthalmology & Visual Science 57 (9) (2016) OCT575–OCT584.

[80] P. Enders, W. Adler, D. Kiessling, V. Weber, F. Schaub, M. M. Hermann, T. Dietlein, C. Cursiefen, L. M. Heindl, Evaluation of two-dimensional bruch's membrane opening minimum rim area for glaucoma diagnostics in a large patient cohort, Acta Ophthalmologica 97 (1) (2019) 60–67.

[81] M. S. Miri, M. D. Abràmoff, Y. H. Kwon, M. Sonka, M. K. Garvin, A machine-learning graph-based approach for 3d segmentation of bruch's membrane opening from glaucomatous sd-oct volumes, Medical Image Analysis 39 (2017) 206–217.

[82] U. Schmidt-Erfurth, H. Bogunovic, S. Klimscha, X. Hu, T. Schlegl, A. Sadeghipour, B. S. Gerendas, A. Osborne, S. M. Waldstein, Machine learning to predict the individual progression of amd from imaging biomarkers, Investigative Ophthalmology & Visual Science 58 (8) (2017) 3398–3398.

[83] R. Geetharamani, L. Balasubramanian, Automatic segmentation of blood vessels from retinal fundus images through image processing and data mining techniques, Sadhana 40 (6) (2015) 1715–1736.

[84] M. K. SV, R. Gunasundari, Computer-aided diagnosis of anterior segment eye abnormalities using visible wavelength image analysis based machine learning, Journal of Medical Systems 42 (7) (2018) 1–12.

[85] A. Rashno, D. D. Koozekanani, K. K. Parhi, Detection and segmentation of various types of fluids with graph shortest path and deep learning approaches, in: Proc. MICCAI Retinal OCT Fluid Challenge (RETOUCH), 2017, pp. 54–62.

[86] S. Xiao, F. Bucher, Y. Wu, A. Rokem, C. S. Lee, K. V. Marra, R. Fallon, S. Diaz-Aguilar, E. Aguilar, M. Friedlander, et al., Fully auto-

mated, deep learning segmentation of oxygen-induced retinopathy images, JCI insight 2 (24) (2017).

[87] S. Sengupta, A. Singh, H. A. Leopold, T. Gulati, V. Lakshminarayanan, Application of deep learning in fundus image processing for ophthalmic diagnosis–a review, arXiv preprint arXiv:1812.07101 (2018).

[88] M. Wilson, R. Chopra, M. Z. Wilson, C. Cooper, P. MacWilliams, Y. Liu, E. Wulczyn, D. Florea, C. O. Hughes, A. Karthikesalingam, et al., Validation and clinical applicability of whole-volume automated segmentation of optical coherence tomography in retinal disease using deep learning, JAMA Ophthalmology 139 (9) (2021) 964–973.

[89] S. Sengupta, A. Singh, H. A. Leopold, V. Lakshminarayanan, Ophthalmic diagnosis and deep learning–a survey, arXiv preprint arXiv:1812.07101 (2018).

[90] J. Hamwood, D. Alonso-Caneiro, S. A. Read, S. J. Vincent, M. J. Collins, Effect of patch size and network architecture on a convolutional neural network approach for automatic segmentation of oct retinal layers, Biomedical Optics Express 9 (7) (2018) 3049–3066.

[91] D. Chen, Y. Yu, Y. Zhou, B. Peng, Y. Wang, S. Hu, M. Tian, S. Wan, Y. Gao, Y. Wang, et al., A deep learning model for screening multiple abnormal findings in ophthalmic ultrasonography (with video), Translational Vision Science & Technology 10 (4) (2021) 22–22.

[92] M. Akiba, G. An, H. Yokota, K. Omodaka, S. Tsuda, T. Kikawa, H. Takahashi, T. Nakazawa, et al., Most contributing quantified ocular parameters for classification of glaucomatous optic disc shape, Investigative Ophthalmology & Visual Science 59 (9) (2018) 1718–1718.

[93] C. Ding, H. Peng, Minimum redundancy feature selection from microarray gene expression data, Journal of Bioinformatics and Computational Biology 3 (02) (2005) 185–205.

[94] Z. Zhang, C. Khow, J. Liu, Y. Cheung, T. Aung, et al., Automatic glaucoma diagnosis with mrmr-based feature selection, J Biomet Biostat S 7 (2012) 2.

[95] K. Stapor, Support vector clustering algorithm for identification of glaucoma in ophthalmology, Bulletin of the Polish Academy of Sciences: Technical Sciences (2006) 139–141.

[96] S. I. Niwas, W. Lin, X. Bai, C. K. Kwoh, C. C. Sng, M. C. Aquino, P. T. Chew, Reliable feature selection for automated angle closure glaucoma mechanism detection, Journal of Medical Systems 39 (3) (2015) 1–10.

[97] A. S. Jadhav, P. B. Patil, S. Biradar, Optimal feature selection-based diabetic retinopathy detection using improved rider optimization algorithm enabled with deep learning, Evolutionary Intelligence 14 (4) (2021) 1431–1448.

[98] G. E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Computation 18 (7) (2006) 1527–1554.

[99] P. Krzyżanowska-Berkowska, K. Czajor, P. Syga, D. R. Iskander, Lamina cribrosa depth and shape in glaucoma suspects. comparison to glaucoma patients and healthy controls, Current Eye Research 44 (9) (2019) 1026–1033.

[100] R. M. Haralick, K. Shanmugam, I. H. Dinstein, Textural features for image classification, IEEE Transactions on Systems, Man, and Cybernetics (6) (1973) 610–621.

[101] F. A. Medeiros, L. M. Zangwill, C. Bowd, R. M. Vessani, R. Susanna Jr, R. N. Weinreb, Evaluation of retinal nerve fiber layer, optic

nerve head, and macular thickness measurements for glaucoma detection using optical coherence tomography, American Journal of Ophthalmology 139 (1) (2005) 44–55.

[102] G. Wollstein, J. S. Schuman, L. L. Price, A. Aydin, P. C. Stark, E. Hertzmark, E. Lai, H. Ishikawa, C. Mattox, J. G. Fujimoto, et al., Optical coherence tomography longitudinal evaluation of retinal nerve fiber layer thickness in glaucoma, Archives of Ophthalmology 123 (4) (2005) 464–470.

[103] T. Ojima, T. Tanabe, M. Hangai, S. Yu, S. Morishita, N. Yoshimura, Measurement of retinal nerve fiber layer thickness and macular volume for glaucoma detection using optical coherence tomography, Japanese Journal of Ophthalmology 51 (3) (2007) 197–203.

[104] A. Canziani, A. Paszke, E. Culurciello, An analysis of deep neural network models for practical applications, arXiv preprint arXiv:1605.07678 (2016).

[105] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818–2826.

[106] A. Krogh, J. Vedelsby, Neural network ensembles, cross validation, and active learning, Advances in Neural Information Processing Systems 7 (1994).

[107] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

[108] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the XIII International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[109] C. A. Curcio, M. Johnson, et al., Structure, function, and pathology of bruch’s membrane, Retina 1 (Part 2) (2013) 466–481.

[110] D. Li, M. Wang, T. Elze, E. I. Paschalis, E. Taniguchi, D. Li, A. V. Turalba, L. R. Pasquale, L. Q. Shen, Relationship between minimum-rim width at bruch's membrane opening and paracentral visual field loss in glaucoma, Investigative Ophthalmology & Visual Science 58 (8) (2017) 4003–4003.

[111] P. Enders, W. Adler, F. Schaub, M. M. Hermann, T. Dietlein, C. Cursiefen, L. M. Heindl, Novel bruch's membrane opening minimum rim area equalizes disc size dependency and offers high diagnostic power for glaucoma, Investigative Ophthalmology & Visual Science 57 (15) (2016) 6596–6603.

[112] E. V. Taniguchi, E. I. Paschalis, D. Li, K. Nouri-Mahdavi, S. C. Brauner, S. H. Greenstein, A. V. Turalba, J. L. Wiggs, L. R. Pasquale, L. Q. Shen, Thin minimal rim width at bruch's membrane opening is associated with glaucomatous paracentral visual field loss, Clinical Ophthalmology (Auckland, NZ) 11 (2017) 2157.

[113] K. K. Dansingani, K. K. Vupparaboina, S. T. Devarkonda, S. Jana, J. Chhablani, K. B. Freund, Amplitude-scan classification using artificial neural networks, Scientific Reports 8 (1) (2018) 1–7.

[114] A. G. Roy, N. Navab, C. Wachinger, Concurrent spatial and channel 'squeeze & excitation'in fully convolutional networks, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2018, pp. 421–429.

[115] F. Visin, M. Ciccone, A. Romero, K. Kastner, K. Cho, Y. Bengio, M. Matteucci, A. Courville, Reseg: A recurrent neural network-based model for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2016, pp. 41–48.

[116] J. Wang, L.-C. Yu, K. R. Lai, X. Zhang, Dimensional sentiment analysis using a regional cnn-lstm model, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (volume 2: Short papers), 2016, pp. 225–230.

[117] P. Krzyżanowska-Berkowska, K. Czajor, D. R. Iskander, Associating the biomarkers of ocular blood flow with lamina cribrosa parameters in normotensive glaucoma suspects. comparison to glaucoma patients and healthy controls, PloS One 16 (3) (2021) e0248851.

[118] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[119] J. Wang, Y. Chen, R. Chakraborty, S. X. Yu, Orthogonal convolutional neural networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 11505–11515.

[120] R. Balestriero, R. Baraniuk, Mad max: Affine spline insights into deep learning, arXiv preprint arXiv:1805.06576 (2018).

[121] A. Krizhevsky, G. Hinton, et al., Learning multiple layers of features from tiny images (2009).

[122] S. Farsiu, S. J. Chiu, R. V. O'Connell, F. A. Folgar, E. Yuan, J. A. Izatt, C. A. Toth, A.-R. E. D. S. . A. S. D. O. C. T. S. Group, et al., Quantitative classification of eyes with and without intermediate age-related macular degeneration using optical coherence tomography, Ophthalmology 121 (1) (2014) 162–172.

[123] E. Alpaydin, Introduction to machine learning, MIT Press, 2020.