Explainable Artificial Intelligence

for Image Segmentation and for Estimation of Optical Aberrations

vorgelegt von
BSc.
Kira Vinogradova
geb. in Moskau, Russland am 6. Juni 1997

an der Fakultät Informatik
der Technischen Universität Dresden
zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften
-Dr.-Ing.-

Dissertation

Promotionsausschuss:
Vorsitzender: Prof. Dr. Jeronimo Castrillon
Hauptbetreuer und Gutachter: Prof. Dr. Ivo F. Sbalzarini
Betreuer: Prof. Dr. Eugene W. Myers
Gutachterin: Dr. Anna Kreshuk
Fachreferent: Prof. Dr. Björn Andres
Tag der wissenschaftlichen Aussprache: 19. Juni 2023

Dresden 2023

Abstract

State-of-the-art machine learning methods such as convolutional neural networks (CNNs) are frequently employed in computer vision. Despite their high performance on unseen data, CNNs are often criticized for lacking transparency — that is, providing very limited if any information about the internal decision-making process. In some applications, especially in healthcare, such transparency of algorithms is crucial for end users, as trust in diagnosis and prognosis is important not only for the satisfaction and potential adherence of patients, but also for their health. Explainable artificial intelligence (XAI) aims to open up this "black box," often perceived as a cryptic and inconceivable algorithm, to increase understanding of the machines' reasoning.

XAI is an emerging field, and techniques for making machine learning explainable are becoming increasingly available. XAI for computer vision mainly focuses on image classification, whereas interpretability in other tasks remains challenging. Here, I examine explainability in computer vision beyond image classification, namely in semantic segmentation and 3D multitarget image regression.

This thesis consists of five chapters.

In Chapter 1 (Introduction), the background of artificial intelligence (AI), XAI, computer vision, and optics is presented, and the definitions of the terminology for XAI are proposed.

Chapter 2 is focused on explaining the predictions of U-Net, a CNN commonly used for semantic image segmentation, and variations of this architecture. To this end, I propose the gradient-weighted class activation mapping for segmentation (Seg-Grad-CAM) method based on the well-known Grad-CAM method for explainable image classification.

In Chapter 3, I present the application of deep learning to estimation of optical aberrations in microscopy biodata by identifying the present Zernike aberration modes and their amplitudes. A CNN-based approach PhaseNet can accurately estimate monochromatic aberrations in images of point light sources. I extend this method to objects of complex shapes.

In Chapter 4, an approach for explainable 3D multitarget image regression is reported. First, I visualize how the model differentiates the aberration modes using the local interpretable model-agnostic explanations (LIME) method adapted for 3D image classification. Then I "explain," using LIME modified for multitarget 3D image regression (Image-Reg-LIME), the outputs of the regression model for estimation of the amplitudes.

In Chapter 5, the results are discussed in a broader context.

The contribution of this thesis is the development of explainability methods for semantic segmentation and 3D multitarget image regression of optical aberrations. The research opens the door for further enhancement of AI's transparency.

Acknowledgements

I am grateful to:

- Eugene Wimberly (Gene) Myers — for his generous support of my initiatives and for giving me a unique opportunity to start my PhD with just a Bachelor's degree.
- The Myers lab — for being supportive colleagues.
- The Sbalzarini lab — for being my foster lab and for the friendly group spirit in 2022/2023.
- The Jug lab — for being my online academic support group during the second lockdown of 2020/2021.
- Alexandr Dibrov — for fruitful discussions about computer vision and microscopy and for helping me like an older brother.
- Coleman Walker Broaddus, Tom Burke, and Fabio Cunial — for giving me very helpful feedback on my second publication and for their support as colleagues in general.
- Michele Marass — for giving very valuable advice on scientific writing in general and for feedback on the manuscript from an editor's perspective.
- Gene Myers and Debayan Saha — for the permission to use the images of fluorescent beads and of a live *Drosophila melanogaster* embryo acquired on the in-house custom-built microscope.
- Debayan Saha, Nicola Maghelli, and Sergei Klykov — for building the custom adaptive optics microscope and developing software for its operation.
- Bruno Cossermelli Vellutini — for providing Debayan Saha (Myers lab) with a stock of genetically modified *Drosophila melanogaster*, whose offspring were used in the experiments conducted by Debayan and me.
- Debayan Saha — for teaching me how to operate the microscope and how to prepare samples.
- Qinrong Zhang and Na Ji (University of California, Berkeley) — for acquiring on an adaptive optics microscope and kindly sharing with Debayan Saha and me (Myers lab) the images of the fixed mouse brain slices.
- Gregoire Montavon, Klaus-Robert Mueller, Wojciech Samek, Jacob Reinhard Kauffmann, and Christopher Johannes Anders (TU Berlin) — for helping me with the questions regarding the LRP method and for giving me a great opportunity to learn from them.
- Robert Haase — for advising me on image analysis in microscopy at the beginning of my PhD.
- Reni Schimmel and Carolyn Fritzsche (international office) — for their amazing help with my "international" issues.

• Reni Schimmel and Susann Gierth — for being the best lab assistants ever.

• My Thesis Advisory Committee — Ivo Fabian Sbalzarini, Gene Myers, Agnes Toth-Petroczy, and Christian Dahmann — for being friendly and supportive.

• My parents Igor and Natalya and grandparents Zoya, Vasiliy, and Evgeniy — for pushing me to do a PhD and, of course, for loving me.

• Researchers from all over the world who used Seg-Grad-CAM in biomedical applications, giving me the contentment that my work was meaningful and impactful.

Table of Contents

List of Figures

List of Tables

1

Introduction

Humans have been fascinated by the idea of creating a machine intelligent like a person since the invention of mechanical devices. One of the earliest attempts to build a machine capable of accomplishing a task that only an intelligent nobleman could perform was taken by Swiss watchmakers Pierre Jaquet-Droz, his son Henri-Louis Jaquet-Droz, and Jean-Frédéric Leschot. Between 1768 and 1774, they designed the Writer, the Draughtsman, and the Musician automatons [1].

The mechanical writer with the "spine" of "programmable disks" was a technological sensation for the 18th century. However, the humanoid was programmable for only a single task and was not truly intelligent in the sense of making decisions, or *thinking* on its own.

What does it mean for a machine to be intelligent? Can machines think?

This question occupied the brightest minds since the late 1940s, when artificial intelligence was born as a research discipline [2]. In 1950, computer science pioneer Alan M. Turing proposed the imitation game [3], which later became known as the "Turing test," as a way to determine whether a machine can be intelligent.

> "I propose to consider the question, 'Can machines think?' This should begin with definitions of the meaning of the terms 'machine' and 'think.'"
> – Alan Turing, 1950.

Seven decades later, in the 2020s, researchers are asking themselves, "What does it mean for artificial intelligence to be able to explain its decisions? Can we explain its decisions and the underlying algorithms in detail?"

I propose to consider the question, "Can artificial intelligence be explainable?" and to begin with defining the meaning of the terms "artificial intelligence" and "explainable."

1.1 Essential Definitions

1.1.1 Artificial intelligence

Artificial intelligence (AI) is a field of computer science that expands the ability of machines (computer systems) to solve problems that require human intelligence. A system that is able to solve such problems is called an AI system (machine, algorithm, etc.) or simply AI.

Alan Turing defined an "intelligent machine" as a machine that is able to mimic human behavior and trick a user by pretending to be a real person [3]. According to this definition, a smart machine with superhuman abilities solving any task with the speed of light is not needed to achieve "intelligence"; working as good as a human is sufficient to be considered "intelligent".

At present, such a definition of AI would be seen rather as that of *artificial general intelligence*, which is also referred to as *Strong AI* or *Full AI*.

Artificial general intelligence is such AI that is capable of solving any problem that a human can solve.

Opposite to that, common AI, called *weak AI* or *artificial narrow intelligence*, is a machine specifically designed to perform a single task [4].

In other words, general artificial intelligence is a machine that has the ability to think, learn, and solve problems in a way that is similar to a human being. This type of AI should be able to perform a wide range of tasks, from playing various games [5] to chatting like a friend [6], and to adapt to new tasks and learn further. It is believed that strong AI has not been achieved yet.

Even stronger AI that outperforms humans in any imaginable task may come into life one day. This hypothetical superhuman AI is called *superintelligence* or *artificial super intelligence*. In the book "Superintelligence: Paths, Dangers, Strategies", Nick Bostrom describes a potential superintelligence as [7]

> "... any intellect that greatly exceeds the cognitive performance of humans in virtually all domains of interest."

It remains an open ethical question whether humanity would benefit or suffer from it.

In this thesis, **the term AI is used for weak AI** and all mentioned methods belong to the category of weak AI.

Machine learning (ML) is a subdiscipline of AI research concerning the algorithms that learn meaningful relations in data and produce the desired answers without being explicitly programmed to follow predefined rules.

The concept of machine learning was first described by Arthur L. Samuel in 1959 [8] and gained popularity afterward.

To construct a machine learning algorithm, the following is needed [9]:

- dataset,
- cost function,
- optimization procedure,
- model.

A machine, commonly called a *model*, learns the patterns in a collection of examples (dataset) and the desired outcome. *Training a model* means making a model learn from the data how to produce the desired answers so that the cost function is minimized. To minimize the cost function during training, an optimization procedure is needed.

There are different ways to train models, known as the types of learning:

- *Supervised learning*

  The answers for the input data are given (data is labeled). The model learns from the error — the difference between the model's predictions and the correct answers.

  Example: classification of images from ImageNet [10].

- *Unsupervised learning*

  The answers are not given (data is unlabeled). The model learns the patterns and relations within the input data.

  Example: clustering of explosives based on spectroscopy measurements [11].

- *Semisupervised learning*

  A portion of the data is labeled but the rest is not. Supervised training is conducted on the labeled data; additionally, the unlabeled data is used for unsupervised training to improve model's accuracy by showing it more training examples.

  Example: classification of protein–protein interactions with limited labeled data available [12].

- *Reinforcement learning*

  A dynamic environment is given. The model learns by taking actions, observing changes in the environment, and receiving feedback in the form of rewards and punishments.

  Example: virtual agents learning to play games [5].

  Nearly all modern reinforcement learning algorithms rely on deep neural networks, hence they are *deep* reinforcement learning models.

Deep learning (DL) is a subcategory of ML and the respective research subfield in which the trained models are artificial neural networks (ANNs) with deep (hidden) layers, also referred to as *deep neural networks (DNNs)*.

Artificial neural network (ANN) is a system of computational units, artificial neurons with activation functions, organized into connected layers.

1.1.2 Explainable

Explainable or interpretable? Which term is applicable to AI?

The community has not reached consensus in definitions of explainability versus interpretability. Many works report the lack of clarity in terminology [13, 14, 15, 16]. The overall tendency is to use the terms "explainability" and "interpretability" interchangeably [16, 17], depending on the context and research field [18], based on popularity [19], or to introduce other terms [14, 20, 21]. In the German-speaking community, the terms "explainable AI", or "erklärbare künstliche Intelligenz," seem to be standard [22, 23, 24]. Nevertheless, some authors insist on disambiguation [15, 25].

Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller [25] defined *interpretation* as

> "the mapping of an abstract concept (e.g., a predicted class) into a domain that the human can make sense of."

They called such domains *interpretable* and gave examples: images and texts. The same authors defined *explanation* as

> "the collection of features of the interpretable domain that have contributed for a given example to produce a decision (e.g., classification or regression)."

In the case of image classification, the interpretable domain is images, the features are parts of the input images. The features in the collection can be assigned scores of relevance that reflect the contribution of the feature to the class' prediction. A *heat map* (often spelled as *heatmap*) formed by the collection of the image parts with their relevance is a classical example of an *explanation*.

According to these definitions, *interpretation* can be seen as a process of retrieving an *explanation*, and *interpretable* means that "a human can make sense of" it. In literature [25, 26], the terms *interpretable* and *explainable* and the verbs *to interpret* and *to explain* have approximately the same meaning. This leads to the conclusion that it is negligible whether we name a method designed to provide an explanation *interpretable* or *explainable*, both are correct.

Acknowledging other existing definitions and differences in the popularity of the terms among the scientific community and Internet users [15, 19], I prefer to use the word "explainable" because it is easier to understand intuitively for nonspecialists, thereby broadening the audience of potential users and contributors. This intuition is supported by my research on the terms' popularity, which shows that "explainable AI" is the most searched for among the general audience (Figure 1.1).

Explainable artificial intelligence (XAI) and *interpretable machine learning (IML)* can be well differentiated on the basis of the algorithms covered by the terms "AI" and "ML."

To the community of XAI, XML (explainable ML), and IML, I would suggest striving unitedly for developing methods to achieve better explainability, interpretability, understandability, transparency, intelligibility, and comprehensibility instead of introducing

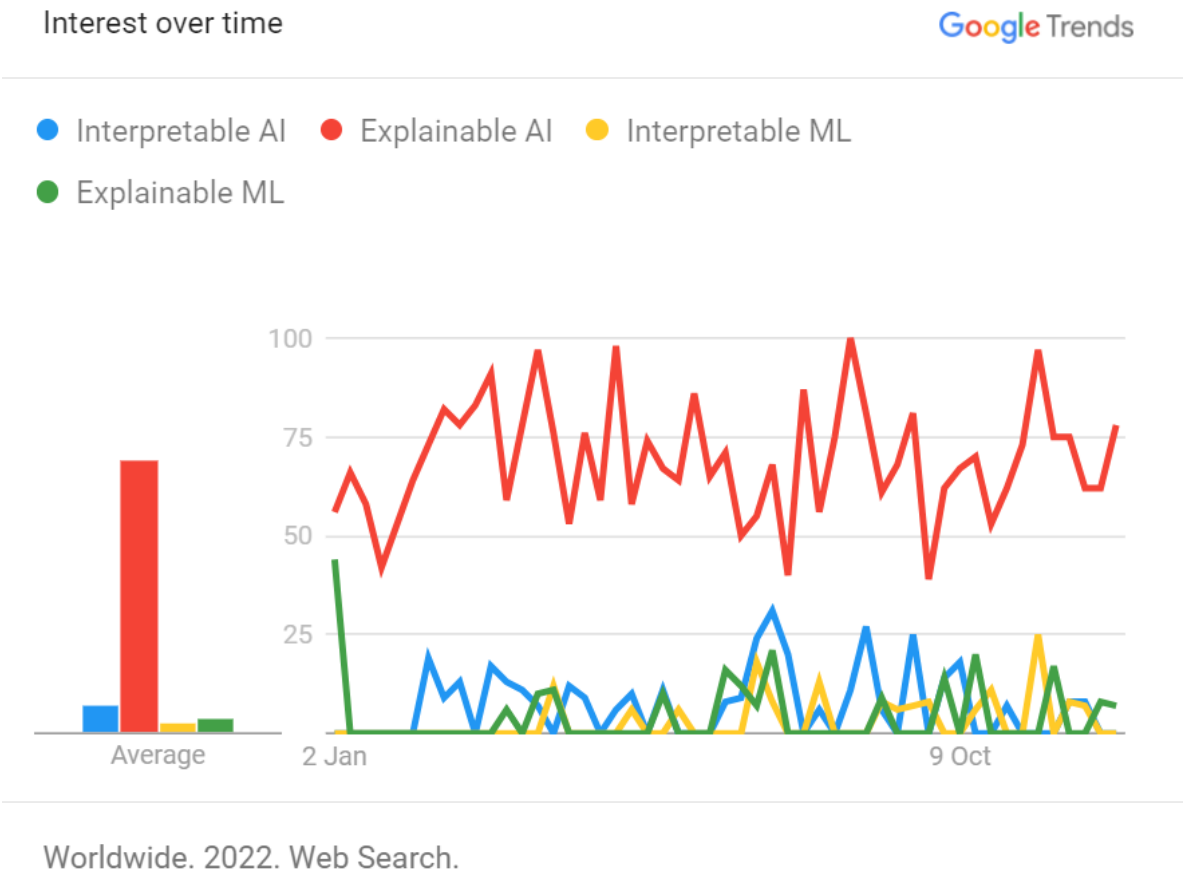

Figure 1.1: Comparison of the popularity of the terms "interpretable AI," "interpretable ML," "explainable AI," and "explainable ML." Data source: Google Trends (https://www.google.com/trends).

new terms and arguing on the old ones.

Explainable artificial intelligence (XAI) is the field of AI research that aims to develop techniques for explaining the decision-making processes of AI algorithms to humans.

In this thesis, "explanation" suits to define the output of the explainability methods according to ref [25]. The terms "explain," "explainable," and "explainability" are preferred, although occasionally "interpret" and related ones may be used depending on the terminology mentioned in the corresponding cited literature and to minimize tautology.

1.1.3 Proposed definitions

Several terms used in this dissertation have the following meanings:

Explanation
– an output (result) of an XAI method.

Interpretation
– a process of retrieval of an explanation.

Explain
– to provide an explanation.
Interpret
– to perform intepretation.

Explainable / Interpretable
– can be explained/interpreted.

Explainability / Interpretability
– ability to be explainable/interpretable.

A slight difference can be made between the terms "explainable" and "interpretable." "Explainable" is applicable to AI algorithms (e.g., ML models) and their output (predictions). AI algorithms and output of XAI methods (explanations) can be called "interpretable." More precisely, explanations can be called human-interpretable to emphasize that these explanations are meaningful to a human.

1.2 Explainable Artificial Intelligence

The definition of artificial intelligence and description of what it means for AI to be explainable are given in the previous section. Here, use cases for XAI and available methods are outlined.

1.2.1 Aims and applications

Making AI algorithms explainable is important in the cases when their decisions may have a strong impact on humans' health and safety or other critical applications.

Potential cases include:

- Healthcare: XAI could be used to explain the decisions made by diagnostic AI tools [27] or by systems that suggest medical treatment. This is especially helpful when making the wrong choice may have serious consequences, for example when an AI system decides whether a patient with COVID needs hospitalization [27]. Checking with XAI whether a model uses the right criteria to make its decisions can help to avoid applying wrong models to high-stakes scenarios in real life.

- Medical research: Insights from XAI may facilitate drug discovery by recommending molecular modifications that reduce toxicity [28]. Explaining the decisions of a model that classifies structural elements by their toxicity would help to streamline research and obtain new medicines faster.

- Safety: XAI should be used to explain the actions of autonomous vehicles to ensure their safe and reliable operation [29]. Understanding the reasons of past crashes can help to improve self-driving systems and avoid accidents in the future.

- Legal: AI-produced risk scorings of future crime may be inaccurate because of unethical practices, for example making decisions on the basis of the skin color of the defendant [30].

Such biased predictions originate from unethical decisions made by humans in the past and must be detected and eliminated for fair justice in the present.

In most of the examples above, **the main goal of XAI** is to deal with AI's unwanted behavior: understand its sources and minimize it by utilizing the knowledge gained from the explanations.

Other purposes of applying XAI techniques include:

- Explaining a particular decision: Why AI predicted y for the data point x. In the USA, for example, the reasons for credit denial must be explained to the customer under §1002.9 of the Equal Credit Opportunity Act of the Code of Federal Regulations (CFR) [31].
- Building users' trust in the decisions of AI systems. Convince stakeholders to use AI.
- Sanity check and causality: Ensuring that algorithms perform as expected, make correct decisions based on right criteria, and consider only causal relationships between the input and output.
- Fairness and ethics: Ensuring nondiscriminative predictions.
- Insights: Learning about the properties of the data from the explanations to speed up research (e.g., in medicines development [28]). Also, XAI provides insights into the internal workings of neural networks [32].

1.2.2 Methods

Transparency of AI models

To begin with, not all models require additional XAI techniques to retrieve explanations from them. Deep neural networks are *black boxes* providing little or no insights about their decision-making process. Some other models are explainable by design because of their natural simplicity; they are often called *white boxes* or *glass boxes*. Examples of such models are linear and logistic regression and decision trees. A decision tree can be decomposed to explain a single prediction by tracing the decision flow through the tree, and the importance of the input features can be retrieved [33].

The models lying between these extremes on the transparency scale (Figure 1.2) and whose decisions can be partially explained with the tools built into the model are sometimes called *gray (or grey) boxes* [34]. Some researchers distinguish only white and black boxes [24]. A random forest and a gradient boosting from any machine learning library have a method called "feature importance" that ranks the impact of the individual input features and works in the same way as a similar method for decision trees. Such algorithms can be called gray boxes, considering that their decision flows cannot be fully traced.

Complex models can be transformed into gray boxes that are explainable by design to a certain degree. Incorporating task-specific information into a model increases its transparency and makes it a gray box [35]. Alternatively, a combination of two models, a black box and a

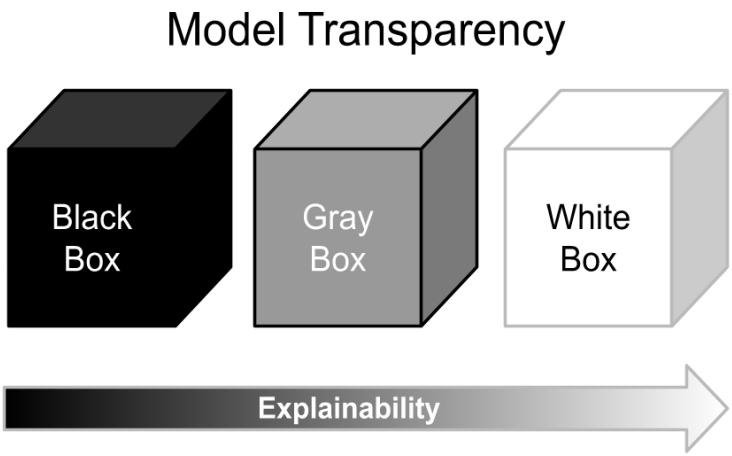

Figure 1.2: Transparency scale of the AI model types: from black boxes that provide little or no explanation about their decision-making to white boxes that are interpretable by design.

white box, is called a gray box [36].

Taxonomy of XAI methods

XAI methods can be categorized using different criteria [16, 24, 33, 34]. The first criterion is closely related to model's transparency, which was discussed above.

Intrinsic (ante hoc) or post hoc

Intrinsic (ante hoc, explainable by design)

Ante hoc is Latin for "before this," which means that interpretation is done before an algorithm produces a prediction. This is achieved when explainability is a part of the algorithm's design. Typically, simple models or models with restricted complexity (e.g., linear regression and decision trees) are considered intrinsically interpretable [33], as intrinsic is a synonym for essential and belonging to a subject naturally. More complex models, such as random forests, may also be regarded as intrinsically interpretable [24]. "Explainable by design" means either that a model is explainable by its simple nature or that a complex model was designed to be more transparent. The layers of deep neural networks can be constructed in such a way that the model becomes explainable [37].

Post hoc

In the context of XAI methods, post hoc (Latin for "after this") means that they are applied to already trained models. As opposed to ante hoc methods, which are always model-specific, post hoc methods are divided into model-specific and model-agnostic categories.

Model-specific or model-agnostic

Model-specific

As the name suggests, model-specific methods are designed only for a certain model or class of models. For example, Gradient-Weighted Class Activation Mapping for Segmentation (Seg-Grad-CAM) [38] is restricted to encoder-decoder architectures.

Model-agnostic

Contrary to model-specific methods, model-agnostic ones are applicable to any AI algorithm, as only an input and output are needed for interpretation, whereas a model is treated as a black box. The Local Interpretable Model-Agnostic Explanations (LIME) [39] method explains the impact of input features on the basis of the changes in the output.

Local or global

Local

Explaining locally means explaining one particular output by, for example, highlighting the parts of the input that were important for producing the output. Seg-Grad-CAM and LIME methods are local.

Global

Explaining globally means explaining the network's decision-making process as a whole. Global methods aim to explain contributions of individual neurons and individual connections between them. Each neuron reacts to certain features, and these features can be visualized either by finding examples in the training dataset that activate the neuron the most or by generating a prototype that maximizes activation. In a recent work of OpenAI Microscope [32], focused on generating prototypes and studying connections between layers, researchers showed how features from a lower layer assemble the next layer of detectors.

Techniques

To name a few:

Removal-based. One of the first attempts to explain image classification was made by Zeiler and Fergus in 2014 in the occlusion experiment [40]. The idea was to remove input features or occlude a portion of the input and then compare the outputs to measure the impact of the occluded part on the decision.

Perturbation-based. The selected input features (e.g., pixels of an area of the input image) are perturbed [41], otherwise these approaches are similar to the removal-based ones.

Architecture modification of neural networks to add explainability to the model. In Class Activation Mapping (CAM), an additional layer was introduced into the model [42].

Surrogate model, relatively simple and intrinsically explainable, is trained to approximate the behavior of the original algorithm (e.g., LIME [39]).

Gradient-based. Methods such as Gradient-Weighted Class Activation Mapping (Grad-CAM) [43] exploit the gradient flow in a neural network.

Propagation-based. In these methods, the relevance of the features is propagated from the output to the input using the selected propagation rule to determine which image parts were relevant for the decision and how high their relevance was. Layer-Wise Relevance Propagation (LRP) technique [44] belongs to this category.

XAI methods can also be classified by the **type of input** they accept (images, text, tabular data, etc.) and the **type of explanation** they produce (visualization of prototypes, selection of important parts of the input, feature importance scores, and others). The types of explanations that can be given depend on the type of input and the task for which the model was trained.

As shown in this section, the question "Can AI be explainable?" has a positive answer. Moreover, AI and its decisions can be explainable in various ways.

In this thesis, I study applications of XAI for computer vision tasks. An introduction to computer vision is presented in the next section.

1.3 Computer Vision

Computer vision (CV) is the branch of artificial intelligence and computer science that studies how to enable computers to comprehend visual input from the environment such as photographs and videos. Its subject is creation of individual algorithms and systems of algorithms capable of autonomously analyzing and comprehending digital videos and images and extracting usable information from them.

1.3.1 Applications

CV algorithms are used for a broad range of tasks, including image and video processing, pattern recognition, object detection, classification, tracking and reidentification, and more. To perform these duties, visual data are acquired using sensors, with digital cameras often being the tool of choice. CV plays an essential role in various applications, such as:

- Biomedical imaging: by analyzing medical images (such as X-rays, computer tomography and magnetic resonance imaging scans, etc.) and identifying patterns typical for certain diseases to help in automatic diagnosis [16]. Also, CV is used to speed up the analysis of research bioimage data such as accurate detection of cells in large images [45].
- Image enhancement, for example, denoising of microscopic images [46] and colorization of old black-and-white photos, as well as in mobile applications adding makeup and other effects to images.

- Autonomous vehicles and robotics: allowing self-driving cars and other types of autonomous vehicles (e.g., delivery bots and carriage platforms at semi-automated factories) to perceive their surroundings, which can be done by detecting and tracking other traffic participants and reading road or local signs [47].

- Security: by detecting anomalous activity on video surveillance [48]. Secure screen lock on mobile devices is powered by biometry (e.g., by face, fingerprint, and iris recognition [49]), which helps to reidentify the user by analyzing the input from the camera or sensor and comparing it with the encoded biometric features of the device owner.

Most of modern state-of-the-art CV solutions are based on deep learning models known as convolutional neural networks (CNNs) because these models can solve more complex problems than traditional programmable algorithms. However, a traditional algorithm can be designed to recognize a shape with distinctive patterns. For example, it is possible to recognize circles using the Hough transform. In the case of 2D grayscale images with uninterrupted circles, this solution may be preferred because of its simplicity and relatively small computation time. More complex cases generally demand other solutions, where CNNs are a key component.

Below, I present more details about the three CV tasks studied in this thesis.

1.3.2 Image classification

A fundamental computer vision task called *image classification* requires an algorithm able to categorize objects or situations (scenes) in pictures into predefined classes. The trained algorithm should output a class (label) for the input image. For example, an image classification system may be taught to classify photos of dogs into breeds (*multiclass classification*) or to divide scenes into "indoor" and "outdoor" (*binary classification*). If multiple classes are allowed to be assigned to a single image, the task is called a *multilabel classification*. An example could be an animal classifier that would output "cat" and "dog" labels if both pets were photographed next to each other.

The desired output for image classification is a class, usually represented by a single integer (technically, often implemented with one-hot encoding).

In binary classification, the target has two possible values (e.g., 0 and 1). In multiclass classification, there is one target that can consist of N potential values, where N is the number of training classes. The target of multilabel classification is a set in which each element has N possible values.

Among many computer vision approaches that have been developed for image classification, the current state-of-the-art ones generally rely on training deep CNNs on large datasets.

ImageNet [10], an extra large collection of about 14 million labeled images in 1000 categories, is one of the most well-known large datasets. Publicly open datasets that can be used to compare models with a predefined metric are called *benchmarks*.

According to the independent leaderboard of ImageNet classification benchmark [50], the top state-of-the-art model is Contrastive Captioner (CoCa) [51], which is a combination of multiple architectures, including transformers. The authors report 91% accuracy after dataset-specific fine-tuning, which means there is still room for improvement for future networks.

Pretraining (fine-tuning excluded) of CoCa lasts approximately five days on Tensor Processing Units. Lighter architectures such as ResNet-50 [52] with 76% top-1 accuracy, Inception-v3 [53] with 78.8% accuracy, or EfficientNet-B7 [54] with 85% accuracy have 30–100 times fewer parameters than CoCa, meaning less time- and energy-consuming training. Depending on how hard the classification task is, these three lighter architectures or even smaller CNNs may be better for identifying fewer classes on smaller datasets.

1.3.3 Image regression

Regression is a task of predicting a finite rational number (usually a fraction on a continuous scale) from input data. An example of regression is a housing price forecast where each prediction is a single non-negative finite decimal number in a local currency. A task of predicting an integer can also be treated as regression rather than classification if:

(1) The classes are ordered: the difference between the labels a and $a + 1$ is smaller than that of a and $a + 2$, therefore when $a + 2$ is predicted for the target label a, the error is larger than in the case of $a + 1$ prediction;

(2) The range of possible target values of the test set might exceed the range of the training set, and generalizability of the model is needed. For example, a model predicting housing prices should be able to extrapolate into the future if the input values suggest prices higher than ever seen in the training set.

If only the first condition applies but the range of targets is known, the problem can be treated as classification and a model can be trained with a loss function (e.g., weighted kappa loss [55]) that takes into account the class order described in condition (1).

In *multitarget (multi-output) regression*, the desired prediction consists of multiple values: it is either a labeled set (data type "dictionary") or a vector.

Image regression is a regression task learned from the image data. For example, estimation of the human age from photographs [56] can be done by solving a regression problem by considering the age as a continuous value and evaluating the result with the mean absolute error (MAE). Counting of tumor cells [57] is a regression problem in which the maximal target value cannot be foreseen during training; this makes the usage of classification approaches irrational. Estimation of the human head pose [58] is an example of *multitarget image regression.*

One of the current state-of-the-art approaches is moving window regression [59] which assigns a rank within the dataset to an input image and refines the rank iteratively within the window between the reference data points. SynergyNet method [60] effectively solves the problem of estimating the head posture, regressing the angles of the face rotation. The approach got its name from the synergy of combined 3D facial landmarks with the special parameters — 3D morphable models — that improve the network's ability to understand 3D facial geometry.

1.3.4 Image segmentation

Image segmentation is the task of splitting an image into meaningful sets of pixels. These sets are meaningful in the sense that they can be used to identify and classify the parts of the image

or the objects in it. Such objects can be traffic participants (pedestrians, cyclists, and cars) and background features (roads, buildings, trees, etc.) in a dataset called Cityscapes [61]. The goal of segmentation is to make a mask with labels for each pixel that say what class or category the pixel belongs to. The mask can be of the size of the original image, where the values of each pixel of the mask correspond to the predicted class. It can also be one-hot-encoded: one dimension is added to the output to represent the binary masks for each class.

Semantic segmentation is a type of image segmentation in which each pixel in an image is labeled with a class without distinguishing between different instances of the same class. E.g., when all cars are labeled with one class, all road signs — with another one.

With regards to distinguishing between different instances, *instance segmentation* has the opposite goal: each individual object of interest gets its own class label. Each car is given its own label in instanced segmentation.

Panoptic segmentation is a hybrid of instance and semantic segmentations: objects of interest (e.g., cars and pedestrians) are subjected to instance segmentation whereas background objects (e.g., roads and buildings) are only semantically differentiated.

Segmentation masks can be *dense* or *sparse*. In dense masks, each pixel in the image is labeled with a class label, and there are no blank areas. Sparse masks label only a subset of pixels, which is commonly desired as an output of instance segmentation.

Partitioning of images into only foreground (objects of interest) and background is known as *binary segmentation*. However simple it may sound, this technique can help to solve important problems such as dermoscopic skin lesion segmentation [62]. It is done by generating an output mask where the pixels with a lesion are labeled with "ones" whereas the background is labeled with "zeros."

Although some segmentation tasks can be done using classical computer vision techniques, solutions utilizing fully convolutional neural networks are preferable. U-Net, a fully convolutional ANN having an encoder-decoder architecture [63], has been a gold standard for years in biomedical semantic image segmentation. The current trend for semantic segmentation is to combine visual transformers with advanced pooling techniques into an encoder-decoder architecture, such as the Lawin Transformer [64].

1.4 Optics

Computer vision has been applied in various domains, including optics and microscopy image analysis. To preface a study of optical aberrations presented in Chapters 3 and 4, a brief introduction to the field is provided below.

Optics is the branch of physics that examines the behavior and properties of light, its interactions with matter, and construction of instruments that detect or utilize light. Its two major subfields are geometrical optics and physical optics. Geometrical optics deals with the propagation of light and the formation of images with optical systems, which may include, for example, lenses and mirrors. In other words, it studies light rays. The area of interest of physical, or wave, optics is light waves: it studies the wavelike properties of light and its nature.

This thesis considers the phenomenon of geometrical optics called optical aberrations.

1.4.1 Aberrations

An *aberration* can be defined as a deviation of a light beam from the trajectory proposed by geometrical optics. Optical aberrations lead to decreased quality of the optical system's output. For example, they cause distorted or blurred images in microscopes and telescopes.

In an ideal optical system, the light rays coming from one point on the object plane focus on one point on the image plane. In practice, however, the rays do not always focus on the image plane as expected. For example, due to imperfections in real-world optical systems, images coming from a microscope always have some aberrations. In microscopy, aberrations can be caused by various factors, such as the curvature of the imaged surface, the compounds, and the materials used in the lenses and the medium between the lens and the sample, or the way the system is designed and assembled [65]. Nevertheless, aberrations are not always the result of manufacturing flaws. Instead, they are always present because real lenses are not infinitely thin like those considered in the theory of geometrical optics [66].

There are different types of optical aberrations, which can be divided into two categories: chromatic (color aberrations) and monochromatic.

Chromatic aberrations

Chromatic aberration is a color distortion of an observed image caused by light rays of different wavelengths (and thus different colors) failing to focus at one point because the refractive index of the lens' material varies with the wavelength. This aberration is often observed in single lenses [66]. Depending on the axis along which the rays pass through the lens, chromatic aberrations are divided into longitudinal (axial) and lateral (transverse, oblique).

Longitudinal (axial) chromatic aberrations

Axial chromatic aberrations occur when the focal length of the lens changes across the wavelengths. This causes the light of different colors to focus at different distances from the lens along the optical axis, creating the color blur around the image's boundary. It happens because beams of light of different wavelengths interact with the lens material and bend differently. The blur caused by axial chromatic aberrations has a lower impact on the image quality than the distortions caused by transverse chromatic aberrations [67].

Lateral (transverse) chromatic aberrations

Transverse chromatic aberrations are expressed when light rays of different colors fall at a lens at an oblique angle and focus at different points on the image plane. The distance from the image center to the focusing point depends on the wavelength, creating color fringes around the image's boundary.

Both types of chromatic aberrations can be corrected using special lens designs [67] such as achromatic doublets [68] consisting of multiple lenses made of different types of glass with various refractive indexes.

Monochromatic aberrations

Chromatic aberrations occur only when several wavelengths are present in the light (i.e., the light is polychromatic, which includes white light). Monochromatic aberrations are observed with monochromatic light (i.e., having one wavelength [69]) or even with quasi-monochromatic light [68]. Monochromatic aberrations result in image deterioration, with the shape of the distortion depending on the type and order of the aberration.

Aberrations of lower orders influence image quality more than those of higher orders. The term order comes from the description of paraxial rays [68], which are the rays close to the optical axis and having a small angle of incidence φ with a refracting surface [69].

Using the following expansion, $\sin\varphi$ can be estimated as

$$\sin\varphi = \varphi - \varphi^3/3! + \varphi^5/5! - \varphi^7/7! + \dots \tag{1.1}$$

The *first-order theory* [68] considers φ to be very small, so that $\sin\varphi \approx \varphi$ (paraxial assumption). This approximation becomes unrealistic if we consider the rays passing through the lens' periphery (nonparaxial rays), which have larger angles of incidence.

The *third-order theory*, which considers $\sin\varphi \approx \varphi - \varphi^3/3!$ in eq 1.1, improves the estimation of the image formation. The deviations from the first-order theory in the image formation were described by Ludwig von Seidel in the 1850s [68] and are called *Seidel aberrations* or *classic aberrations* [69].

Seidel described five types of monochromatic aberrations: astigmatism, coma aberrations, spherical aberrations, distortion, and field curvature. The first three of these — astigmatism, coma aberrations, and spherical aberrations — are considered in this thesis.

Seidel (classic) aberrations

Spherical aberrations

Spherical aberrations occur when light passing through the periphery of a lens focuses at a different point than light passing through the center of the lens, causing blur on the periphery. In the case of positive (undercorrected) spherical aberrations, typical for spherical lenses, nonparaxial rays focus closer to the lens than paraxial rays from the center. Negative (overcorrected) spherical aberrations result in nonparaxial rays focusing farther from the lens than paraxial.

Correction of spherical aberrations is often done by changing the geometry of a lens by, for example, making it more curved in the center than on the periphery. Plano-convex lenses have minimal spherical aberrations when their convex side faces the light beam.

Spherical aberrations are sometimes called longitudinal spherical aberrations [69]. Indeed, their formation is similar to that of longitudinal (axial) chromatic aberrations. The difference is that in spherical aberrations the rays having one wavelength do not meet in the focus, whereas in chromatic aberrations, the rays of different colors are defocused.

Coma

Coma is another type of monochromatic aberration that has a formation principle similar to that of its chromatic counterpart — lateral (transverse) chromatic aberration. Again, the monochromatic rays are defocused along the focus plane instead of the chromatic ones. The name "coma" comes from the comet-shaped blur in an image of a point source. Coma occurs when the incoming light is oblique to the optical axis, which happens when the light source is off-axis or if the components of the optical system are tilted with respect to one another. The direction of coma is positive when the tip of the "comet" faces the optical axis, otherwise, the direction is negative. Coma can be corrected by changing the shape and curvature of a lens at necessary locations.

Astigmatism

Astigmatism is observed when the light propagating in perpendicular planes from a point does not focus at a single point after passing through a lens. The result of astigmatism is elongation of the edges of objects, with the direction of the distortion depending on the angle of astigmatism. Correction of astigmatism can be done by using special combinations of lenses called anastigmatic lenses, or anastigmats [68]. Combined lenses can also be designed to correct other aberrations besides astigmatism.

Field curvature (Petzval field curvature) aberration appears when the distance between a lens and the image plane is not the same at each point (i.e., when the image plane becomes a nonplanar surface.)

Distortion is caused by variations in the magnification at different parts of a lens, which results in straight lines of an object looking bent in the image. Barrel distortion "bends" the image to the outside and makes objects appear more round; pincushion distortion "bends" the image to the inside.

These classic aberrations are particularly important for the design of ophthalmic lenses that are prescribed to vision-impaired patients, as well as in other branches of optics such as photography [69]. However, a model based on the classic aberrations would be insufficient to describe other, less commonly encountered aberrations.

Point spread function

The types of aberrations can be analyzed using the *point spread function (PSF)*. The PSF is defined as the response of an imaging system to a point light source [66]. A computer scientist can think of the PSF as a kernel in the convolutional operation. For example, a microscopic image of an object appears partially blurred or deformed because the convolutional kernel, the PSF, is applied to the object inside the imaging system (Figure 1.3).

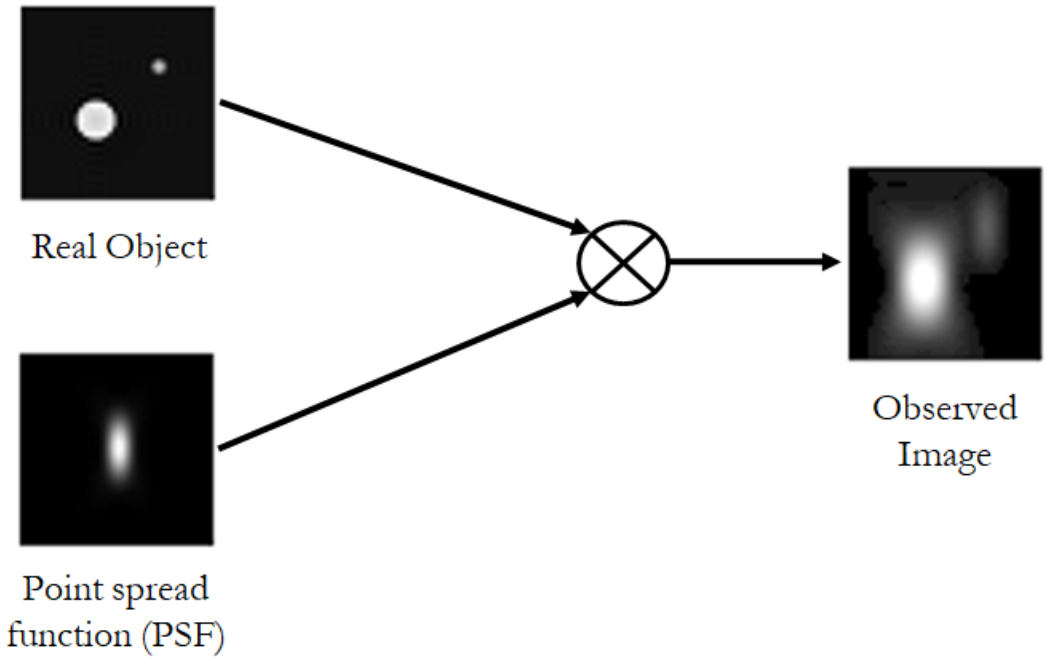

Figure 1.3: Point spread function (PSF). Modified from [70] (public domain image).

1.4.2 Zernike polynomials

In 1934, Fritz Zernike published a more comprehensive classification of optical aberrations [72]. Currently, the Zernike polynomials (Zernike terms) are more commonly used in qualitative and quantitative analysis than Seidel (classic) aberrations [69]. Figure 1.4 shows 15 Zernike polynomials rendered in 3D.

The Zernike polynomials describe the wavefront of an optical system in general. A *wavefront* is a surface that represents a set of points that are in one phase of the wave. The wavefront of a light beam passing through a lens is used to compute the lens's aberrations. The Zernike polynomials are particularly useful for estimation of aberrations because each polynomial corresponds to a specific type of aberration and is orthogonal to the others.

The Zernike polynomials are denoted as

$$Z_n^m \quad \text{or} \quad Z_n^{\pm m} \tag{1.2}$$

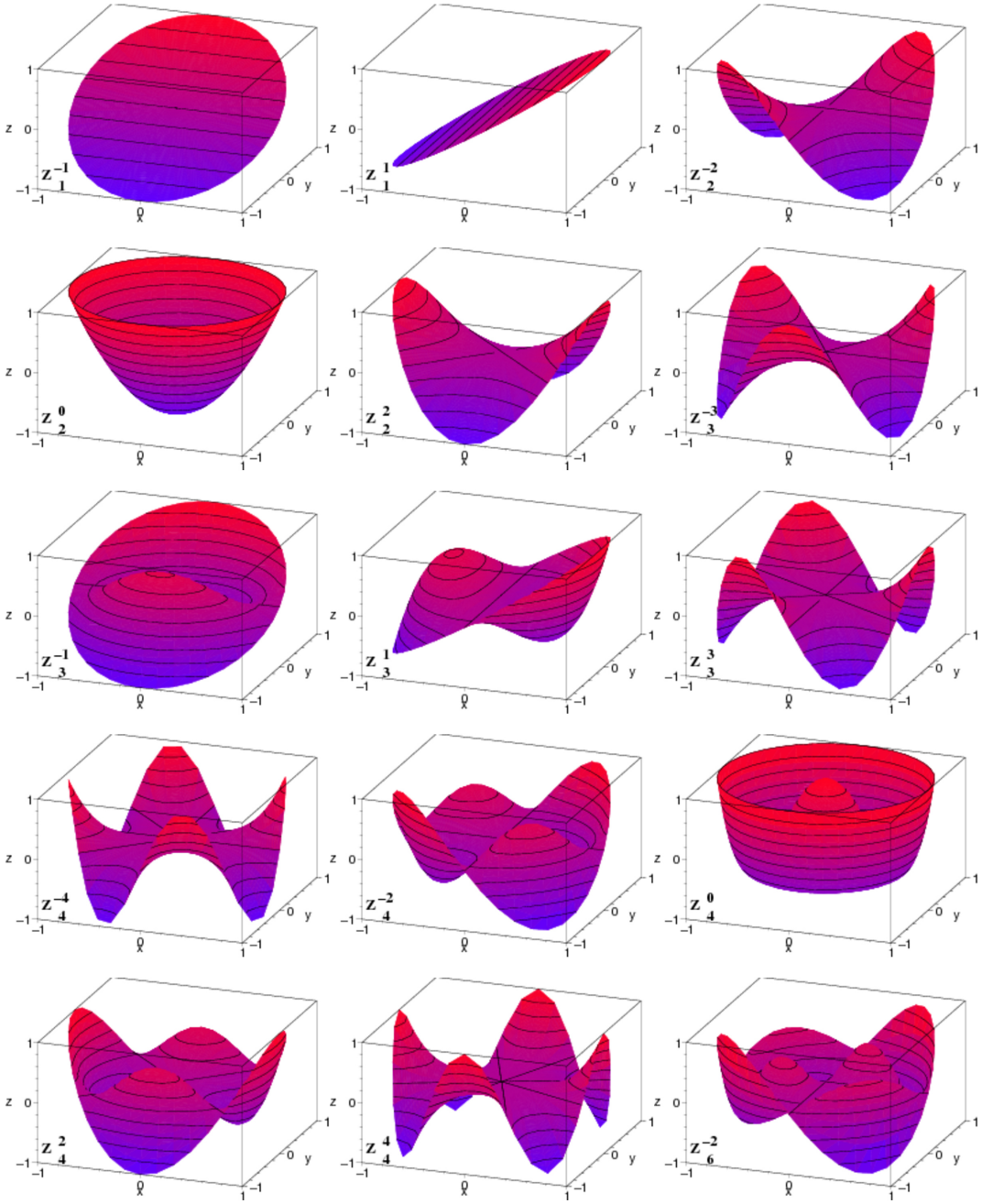

Figure 1.4: 3D visualization of the Zernike polynomials (public domain image) [71]. The first 14 polynomials (piston excluded) are ordered according to ANSI nomenclature and Table 1.1, from Y-tilt to vertical quadrafoil. The image at the bottom right corresponds to sixth-order oblique astigmatism.

where n is the power (radial order) of the radial distance polynomial ρ ($0 \leq \rho \leq 1$) and m is the angular coefficient (azimuthal degree) of the azimuthal angle θ ($0 \leq \theta \leq 2\pi$). The Zernike polynomials can be denoted as a function of ρ and θ in polar coordinates:

$$Z_n^{\pm m}(\rho, \theta) \tag{1.3}$$

The Zernike polynomials are divided into *even*, with $m \geq 0$, and *odd*, with $m < 0$. The even and odd Zernike polynomials are defined as [73]

$$Z_n^m(\rho, \theta) = R_n^m(\rho) \cos(m\theta) \quad \text{for even} \tag{1.4}$$

$$Z_n^{-m}(\rho, \theta) = R_n^m(\rho) \sin(m\theta) \quad \text{for odd} \tag{1.5}$$

where $(n - m) \geq 0$, θ is the azimuthal angle, ρ is the radial distance, and $R_n^m(\rho)$ are the radial polynomials:

$$R_n^m(\rho) = \sum_{k=0}^{\frac{n-m}{2}} \frac{(-1)^k (n-k)!}{k! \left(\frac{n+m}{2} - k\right)! \left(\frac{n-m}{2} - k\right)!} \rho^{n-2k} \tag{1.6}$$

For $(n - m) \leq 0$ or $(n - m)$ being odd, $R_n^m(\rho) = 0$ and a polynomial does not exist.

Then, the wavefront ϕ over a circle of radius R is a sum of the Zernike polynomials with the corresponding amplitudes a_j with a single index j [74]:

$$\phi(\rho, \theta) = \sum_j a_j Z_j(\rho, \theta) \tag{1.7}$$

In the *Noll nomenclature* (indexing system), the indices m and n are converted into j according to Table I in ref [74].

In the *OSA / ANSI nomenclature* [75], another commonly used indexing system, the conversion is done as

$$j = \frac{n(n+2) + m}{2} \tag{1.8}$$

The relations between the indices n, m, both ANSI and Noll nomenclatures, formulas for $Z_n^{\pm m}(\rho, \theta)$ [75], and the names of the first 15 aberrations [76] are presented in Table 1.1. The indices n and m are both either even or odd because otherwise $Z_j = 0$.

The aberrations of the second, third, and fourth order (below oblique astigmatism in Table 1.1) are used in this thesis.

1.5 Thesis Overview

1.5.1 Motivation

Explainability is an important topic of AI research, as modern complex AI systems are difficult for humans to understand. This can be a problem when advanced AI systems are employed to make important decisions, such as in healthcare. XAI methods aim at making AI systems more transparent and understandable, more reliable, safe, and fair. Enhancing XAI techniques can make a positive impact on human society.

Transparent and safe AI solutions can be brought into new domains, such as medicine or justice, where high risks are involved. XAI helps to better understand how AI makes decisions and to check if they are made as expected from the human point of view. This is especially important when it comes to incorrect decisions and can be used to eliminate harmful biased

behavior and to improve AI systems in general.

Computer vision solutions are often employed in biomedicine, handling tasks of classification and segmentation of X-ray scans, computed tomography (CT) scans, magnetic resonance images (MRI), ultrasound images, and laboratory microscopic images. Considering the importance for human well-being, development of explainability techniques for computer vision models in this domain should be one of the top priorities of AI research in the coming years.

In biological research, CV is used in analysis and restoration of microscopic images, which are affected by optical aberrations that decrease image quality, especially when imaging is done deep inside a sample or when the sample's surface is curvy. In the latter case, the sample itself introduces additional aberrations besides those that are due to the microscope's design. To improve the bioimaging quality and observe the real sample more clearly, aberrations should be removed or at least minimized, for which they need to be qualitatively and quantitatively estimated. Estimation of aberrations is a multitarget image regression problem that convolutional neural networks can help to solve.

Challenges

At the time when the research for this thesis was started in 2018, XAI methods for computer vision models were limited to the task of image classification and applicable only to certain problems and domains. The problem of explaining decisions of image segmentation models to humans was not solved and applicability of XAI to image regression was unclear.

Accurate estimation of optical aberrations using computational methods was limited to trivial objects, and the underlying deep learning model for multitarget image regression was not explainable.

Table 1.1: Indices and names of the first 15 Zernike polynomials $Z_j(\rho, \theta)$

Z_n^m	j, ANSI	j, Noll	n	m	$Z_j(\rho,\theta) = Z_n^m(\rho,\theta)$	Name
Z_0^0	0	1	0	0	1	Piston
Z_1^{-1}	1	3	1	-1	$2\rho\sin\theta$	Tilt (Y-Tilt)
Z_1^1	2	2	1	1	$2\rho\cos\theta$	Tip (X-Tilt)
Z_2^{-2}	3	5	2	-2	$\sqrt{6}\rho^2\sin 2\theta$	Oblique astigmatism
Z_2^0	4	4	2	0	$\sqrt{3}(2\rho^2-1)$	Defocus
Z_2^2	5	6	2	2	$\sqrt{6}\rho^2\cos 2\theta$	Vertical astigmatism
Z_3^{-3}	6	9	3	-3	$\sqrt{8}\rho^3\sin 3\theta$	Vertical trefoil
Z_3^{-1}	7	7	3	-1	$\sqrt{8}(3\rho^3-2\rho)\sin\theta$	Vertical coma
Z_3^1	8	8	3	1	$\sqrt{8}(3\rho^3-2\rho)\cos\theta$	Horizontal coma
Z_3^3	9	10	3	3	$\sqrt{8}\rho^3\cos 3\theta$	Oblique trefoil
Z_4^{-4}	10	15	4	-4	$\sqrt{10}\rho^4\sin 4\theta$	Oblique quadrafoil
Z_4^{-2}	11	13	4	-2	$\sqrt{10}(4\rho^4-3\rho^2)\sin 2\theta$	Oblique secondary astigmatism
Z_4^0	12	11	4	0	$\sqrt{5}(6\rho^4-6\rho^2+1)$	Primary spherical
Z_4^2	13	12	4	2	$\sqrt{10}(4\rho^4-3\rho^2)\cos 2\theta$	Vertical secondary astigmatism
Z_4^4	14	14	4	4	$\sqrt{10}\rho^4\cos 4\theta$	Vertical quadrafoil

1.5.2 Dissertation outline

This thesis consists of five chapters.

- Chapter 1 (Introduction) contains definitions of AI and XAI, aims and methods of XAI; presents computer vision techniques for image classification, regression, and segmentation; and describes optical aberrations and their estimation with the Zernike polynomials.
- Chapter 2 presents a method for explainable semantic image segmentation that finds strategies to explain the reasoning of encoder-decoder CNNs that perform image segmentation.
- In Chapter 3, application of computer vision to estimation of monochromatic aberrations in microscopy with the Zernike polynomials in 3D images of research samples is described and image restoration with predicted aberrations is shown.
- Chapter 4 presents a method for explainable multitarget 3D image regression in the context of the taskfrom Chapter 3 and proves the method's reasonability by explaining the related multiclass 3D image classification of aberrations in addition to the regression task.
- Chapter 5 presents conclusions and an outlook into the future of XAI.

Figure 1.5 illustrates the connection between the chapters.

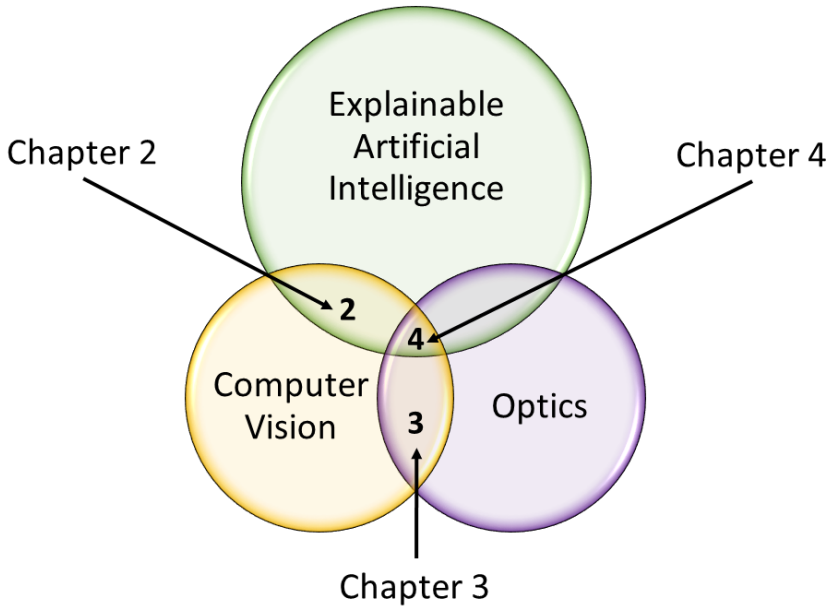

Figure 1.5: Thesis outline: the connection between the topics of Chapter 1 (Introduction) and Chapters 2, 3, and 4 presenting the results.

2

Explainable Image Segmentation

2.1 Abstract

Convolutional neural networks (CNNs) find increasing use in many areas, and it is becoming essential to make them explainable. This is especially true for CNNs employed in critical domains such as healthcare and safety of autonomous vehicles, which often involve the task of image segmentation. For example, to decide whether hospitalization is required for a COVID patient, it may be necessary to evaluate the damage to the lungs. In road safety, the surroundings and traffic participants are identified and located for safe operation of self-driving vehicles. The road identified in the pictures from the front camera must be segmented to ensure that the self-driving car drives only on the allowed parts of the road and does not enter the pedestrian path. In such scenarios, explaining the predictions of the segmentation CNNs is crucial, especially in cases when the performance of the model is unsatisfactory and thereby unsafe.

When this project started in March 2019, no method for explainable image segmentation existed. With multiple approaches for explaining image classification already available, there were four potential strategies to consider to overcome the lack of explainability for segmentation: (1) transforming the segmentation task into a pixel-wise classification problem, (2) constraining the choice of AI models in critical applications to solutions interpretable by design, (3) developing an explainable artificial intelligence (XAI) method from scratch explicitly for segmentation, or (4) adapting to segmentation an existing XAI method for classification. The first option may have very limited applicability due to high computational costs and low performance on complex tasks. In the second strategy, interpretable by design models are either too simple to perform complex segmentation tasks or are developed for a single network architecture prior to its training and are not transferable to already trained state-of-the-art CNNs. In the third strategy, the desired outcome of visualization was less predictable compared to the last one (adapting an existing method), which was therefore preferred as the starting point for exploring the undiscovered research field.

Following an approach of adapting to segmentation an existing XAI method for classification, I proposed with my colleagues a novel technique for explainable semantic image segmentation, presented at the 34th AAAI Conference on Artificial Intelligence in February 2020 [38]. This technique is based on an XAI method for classification called gradient-weighted class activation mapping (Grad-CAM) and is named gradient-weighted class activation mapping for segmentation (Seg-Grad-CAM).

2.2 Related Work

Explainability of the tasks beyond classification is a developing research area. The literature review below represents the state of research as of November 2022. For semantic image segmentation, XAI methods were proposed very recently (since 2019). They are mostly focused on explaining decisions of U-Net neural network [63] and its variations because this architecture and its variants have become standards for semantic image segmentation, especially in the medical domain [77].

The earliest attempt to shed light on concepts learned by a segmentation network is dated June 2019, when Janik et al. proposed an interactive visualization tool for latent space representations by finding corresponding samples in the dataset [78]. They demonstrated their tool on U-Net trained for detecting buildings in satellite images.

The first method toward more transparent semantic segmentation, grid saliency [79], is aimed at detecting biases in the data that may affect the decisions of segmentation CNNs. It is based on perturbation analysis and assumes that co-occurrences of classes and the context are important for segmentation. For the perturbation analysis, a region of interest of one class is selected and the context is perturbed by changing the pixels outside the region of interest. Then, the class prediction change is evaluated and the minimal context, which is necessary to assign the region to the correct class, is identified. This context is shown as a saliency map plotted on top of the input image. If no context is important for the segmentation of the object, the saliency map is blank. This means that the method is explicitly designed to find what was important for the decision only in the context, without taking into account the pixels inside the object of interest. The purpose of this method is to identify data biases (co-occurrences of the classes influencing the decisions). Finding biases in the data is in general essential to ensure that the model makes decisions for the right reasons.

The authors proposed a toy dataset with an intentional bias in one half of the image, which is described in Subsection 2.4.2, and a metric for assessing the quality of explanations quantitatively. According to this metric, the best explanation is the one that assigns relevance only to the pixels in the context. However, the method's ability to highlight important pixels only outside the object may not best serve the purpose of evaluating the explanations. Besides, it is unlikely that the bias located in the background in one half of the image is sufficient for successful segmentation of the object present in the whole image. Moreover, biases may occur in a part of the objects and "assist" the network in learning. The grid saliency method was specifically designed to perturb only the context and find the pixels influencing the prediction inside the context, thus excluding the object from the saliency map. This makes “unfair” a

comparison of this method with those not having such a restriction and designed for the general explanation purposes.

The semantic bottlenecks method [80] is based on the idea of inserting special layers in an already trained network and retraining it to learn special concepts. The authors showed two approaches: supervised semantic bottlenecks and unsupervised semantic bottlenecks. The former requires additional work of selecting concepts manually from another dataset, inserting this bottleneck to retrain the network as a classifier, then removing this bottleneck and measuring the quality of segmentation by intersection over union (IoU). According to IoU, the best position for the insertion is selected and activation maps are retrieved from this bottleneck. In the unsupervised version, where only the number of concepts is selected rather than the concepts themselves, the next steps are fairly similar but the training is unsupervised. The rules for choosing the number of concepts and the position where the bottleneck should be placed are refined empirically for each architecture. Unfortunately, this requires a lot of work for each new network or dataset.

U-Noise method [81], as the name suggests, is based on U-Net. It is similar to the occlusion experiment [40], but in U-Noise the image is occluded with the learned noise. First, U-Net is trained on an unaltered input to do the segmentation. Further modifications are analogous to training of a generative adversarial network consisting of a discriminator and a generator. U-Net is duplicated: one copy is used as a so-called utility model acting like a discriminator in a generative adversarial network, the other copy is called an interpretability model and acts as a noise generator. The weights of the utility model are frozen to prevent retraining. The interpretability model learns to occlude the image with a tolerable amount of noise that still allows the utility model to correctly predict the segmentation mask. The output of U-noise is a map that shows the tolerance of the pixels to noise. Despite requiring additional training, this method is practical because it is not constrained to a single CNN architecture and any segmentation network can be retrained in such a manner.

Generative adversarial networks themselves can be used for semantic segmentation, and it has been shown that layer-wise relevance propagation [44] can be applied to the discriminator to retrieve heat maps showing the most relevant pixels for segmentation with the generative adversarial network [82].

Modifications to Seg-Grad-CAM have been proposed in several works. The technique has been adapted for 3D segmentation [83] and 2.5D segmentation [84]. In the master's thesis of J. Lei [85], a comparison has been made between Seg-Grad-CAM and other gradient-based XAI methods (VanillaGrad [86], SmoothGrad [87], and Grad-CAM++ [88]) adapted for segmentation using the concept of Seg-Grad-CAM, named Seg-VanillaGrad, Seg-SmoothGrad, and Seg-GradCAM++, respectively. For the lack of conventional metric for comparing explanations of segmentation predictions, the methods have been compared visually. Seg-GradCAM++ has been reported to provide the sharpest boundaries of the regions on the explanation heat maps, whereas Seg-Grad-CAM has been suggested as the most time-efficient and analytically reliable.

Besides explaining the results of segmentation, Grad-CAM [43] has been adapted to a convolutional long short-term memory network inspired by the idea of changing the layers

selected for gradient propagation [89]. In a recent work by Humer et al. [90], the researchers have combined the idea of Seg-Grad-CAM with the global average pooling (GAP) layer from a method for object localization — the class activation mapping (CAM) [42], and stated that any gradient-weighted explanation technique can be placed under the hood instead of Grad-CAM. The work of Melching et al. [91] reported a U-Net-based CNN with a parallel path with GAP and retrieval of gradient-weighted explanations for crack tip detection.

CAM, Grad-CAM, and Seg-Grad-CAM mentioned above are described in detail in Methods section.

2.3 Methods

2.3.1 CAM

The class activation mapping (CAM) method has been initially proposed for object localization using a classification network without access to localization labels [42]. The authors achieved 37.1% top-5 error in the ILSVRC 2014 challenge [92] without using bounding box annotations.

Later, this technique was used for visualizing the locations in an input image that were most relevant for a classification decision of a CNN: for example, if the class "brushing teeth" was predicted, CAM highlights the open mouth of a person and the toothbrush in the image [42]. A schematic of the method's operation is shown in Figure 2.1.

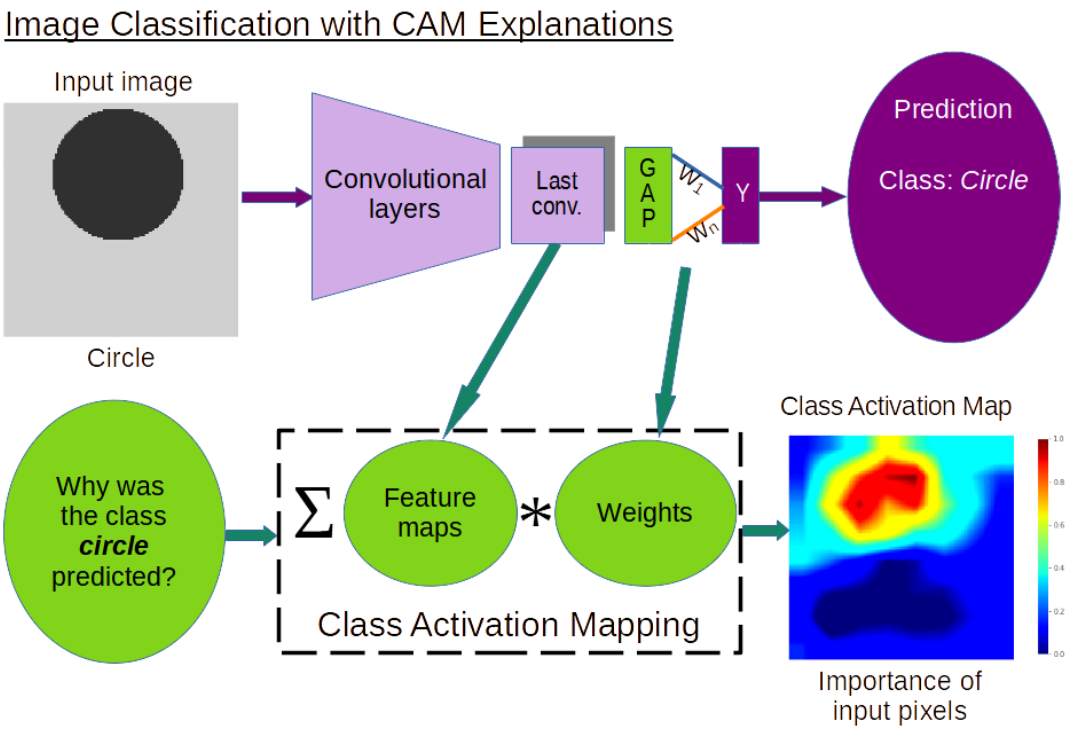

Figure 2.1: Schematic of the class activation mapping (CAM) applied to an image classifier. The global average pooling (GAP) technique was applied to the feature maps from the last convolutional layer to weigh their impact on finding the object of the predicted class.

Concept

CAM generates a class activation map (heat map) that highlights those important regions in the image that have the largest weights for the predicted class label. The CNN must include a global average pooling (GAP) layer between the last convolutional layer and the final dense layer, meaning that only one dense layer can exist. The authors modified three state-of-the-art architectures of that time by removing all extra dense layers, except for the last one, and placing GAP in between instead. These CNNs were then trained for supervised image classification, and the learned weights between the GAP and dense layers together with the feature maps from the last convolutional layer were used for interpretation.

Formal calculation

Let us assume that the network in Figure 2.1 is trained to divide images into three classes: "circle" or "triangle" if it finds a respective shape in the input, and "neither circle nor triangle" if other shape or no geometric shape is found. The CNN consists of a block of convolutional layers, a GAP layer, and a dense layer for classification with softmax as an activation function, which outputs a vector of probability for each class. At the prediction step, an image of a circle is passed into the CNN and receives the prediction, "it is a circle." To understand what the network has learned from the data, the user of the classifier may wonder, "why is it a circle?" To answer this quesion, a class activation map can be generated by extracting the feature maps from the last convolutional layer and the weights between the GAP layer and the dense layer. Each feature map A^k is multiplied by its corresponding weight w_k^c between the result of GAP for this feature map and the neuron responsible for the prediction of class c, "circle" (k is the index of a single feature map in the layer of interest with K feature maps). By this, the feature maps are weighted by their impact on the prediction. The weighted sum of the activation maps is the class activation map, or in other words, the localization map L^c_{CAM} for class c:

$$L^c_{\text{CAM}} = \sum_k w_k^c A^k \tag{2.1}$$

In general, CAM can be used to better understand the decision-making process of deep learning models and can be a useful tool for visualizing the features learned by a model and for finding the spatial location of the objects. The obvious drawback of this method is that it is ante hoc: it enforces the placement of GAP and requires training of the model after that.

2.3.2 Grad-CAM

The gradient-weighted class activation mapping (Grad-CAM) [43], having the same objectives as CAM, lifts the GAP insertion requirement, thereby achieving post hoc interpretability. Designed for CNNs consisting of convolutional layers and fully connected layers at the end, Grad-CAM can be applied to any CNN trained for classification. Besides localization of the objects responsible for image classification, the authors have shown applicability of their approach to networks designed for image captioning and visual question answering. The operation of Grad-CAM for explaining an image classifier (localization of the object) is shown

in Figure 2.2.

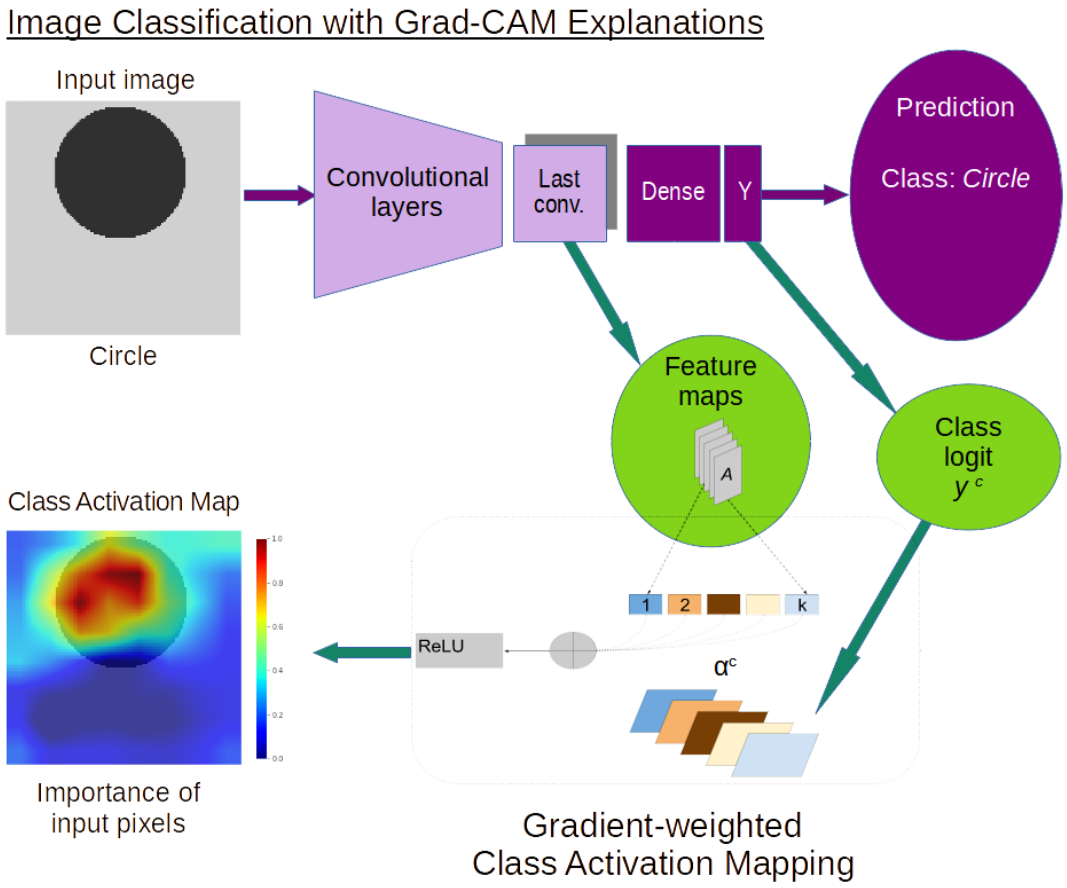

Figure 2.2: Schematic of the gradient-weighted class activation mapping (Grad-CAM) applied to an image classifier.

Concept

Grad-CAM is a class activation mapping technique that produces a heat map for an input picture, indicating the image areas that contribute the most to the model's output. Its central idea is to construct a class activation map using the gradients of the output class with respect to the feature maps of the last convolutional layer. These gradients are used to compute the importance weights of each feature map, which are then multiplied by the corresponding feature maps, producing the activation maps for the class. To obtain the final heat map, the class activation maps are averaged across all the feature maps and then thresholded to cut off any negative relevance speaking against the prediction. The resulting heat map is overlaid onto the input picture to localize the important regions.

The usage of feature maps of the convolutional layer prior to fully-connected layers replaces GAP in CAM: Grad-CAM uses backpropagation of gradients instead of an extra pooling layer. This was done to generalize the approach to a broad variety of network architectures common for image classification.

Formal calculation

Coming back to the toy task of classification of geometric shapes, let us assume that we use an already trained state-of-the-art CNN with a block of convolutional layers, one intermediate dense layer, and one dense layer with the softmax activation function for classification. To get a single class label as the output, we would find the argument with the highest probability in the prediction vector. To explain why the input picture was classified as a circle, we first select the logit y^c, which is the probability of class c, "circle," in the output prediction vector y. Then, the feature maps A should be extracted from the last convolutional layer. Next, the gradients $\frac{\partial y^c}{\partial A^k}$ are propagated from the logit y^c to the feature maps A (the gradients of y^c are taken with respect to A). The importance weights α_k^c, analogous to the GAP weights w_k^c in eq 2.1, are the spatial average of these gradients across N pixels (indexed by u, v) in each feature map A^k:

$$\alpha_k^c = \frac{1}{N} \sum_{u,v} \frac{\partial y^c}{\partial A_{uv}^k} \tag{2.2}$$

Analogous to CAM, the weighted sum is defined as

$$\sum_k \alpha_k^c A^k \tag{2.3}$$

To produce only the positive importance, the weighted sum is rectified (the negative values are set to zero) using the rectified linear unit function (ReLU) defined as

$$\mathrm{ReLU}(x) = \max(0, x) \tag{2.4}$$

By combining eqs 2.2 and 2.3 and applying ReLU, the localization map $L^c_{\text{Grad-CAM}}$ for the class c can be expressed as

$$L^c_{\text{Grad-CAM}} = \mathrm{ReLU}\Big(\sum_k \alpha_k^c A^k\Big) = \mathrm{ReLU}\Big(\sum_k A^k \frac{1}{N} \sum_{u,v} \frac{\partial y^c}{\partial A_{uv}^k}\Big) \tag{2.5}$$

Thereby, Grad-CAM is a generalization of CAM, with the advantage that it does not require any manipulations with the architecture and only needs an access to the layers of the CNN and to the algorithm for backpropagation of the gradients.

2.3.3 U-Net

The deep fully convolutional neural network U-Net won image segmentation challenges in 2015 [63] and became a revolution in bioimage segmentation after beating the previous state-of-the-art architecture, the sliding window approach [93]. Originally designed for biomedical tasks, U-Net has since been used for a wide variety of image segmentation applications.

A deep fully convolutional network is a type of CNN where all trainable layers are of a convolutional type. The input and output can be of any size, provided that the shapes of the intermediate inputs to the convolutional layers do not shrink to zero size after a series of pooling layers, which are usually present in the architecture. The input size is normally equivalent to that of the output. This makes fully convolutional networks a good choice for

tasks such as image segmentation, where the input can be images of arbitrary sizes whereas the output should be as large as the input.

U-Net is such a deep fully convolutional network with pooling layers, consisting of an encoder (contracting path) and a symmetric decoder (expanding path). The encoder path of U-Net is constructed of several layers of 2D convolutional and max pooling operations that downsample the input image and extract feature representations. It is similar to a convolutional block (before dense layers) of a CNN for classification, which gets narrower in the last layers. The layers at the beginning of the encoder learn low-level features, such as edges and simple geometry. The learned concepts become more and more sophisticated as the end of the encoder approaches. The decoder path uses transposed convolutional layers to upsample the feature maps to their original resolution. The bottleneck is the series of convolutional layers (usually two or three layers) between the encoder and decoder.

Convolutional layers in general learn spatially invariant features that can be used to make predictions at any location in the input image, an ability valuable for image classification. However, this is not intended for image segmentation. For this specific reason, shortcut (skip) connections between the encoder and decoder paths are added to localize the features and capture both low-level and high-level features at multiple scales. These shortcut connections were key to success in segmentation.

Since 2015, multiple variations of U-Net were introduced [77]. In this dissertation, two variants of U-Net were used to study the applicability of Seg-Grad-CAM: “vanilla” U-Net with a depth of four (i.e., having four blocks with shortcut connections and the bottleneck path as the fifth connection between the encoder and decoder) and U-Net with VGG16 encoder (also referred to as backbone) and the mirroring decoder with transposed convolutions.

The architecture of “vanilla” U-Net from the Python library "CSBDeep"[94] is summarized in Tables 2.1 and 2.2, that of U-Net with VGG16 backbone from the Python package “Segmentation Models” [95] — in Table 2.3. A schematic overview of the structure of U-Net is shown in Figure 2.3.

2.3.4 Seg-Grad-CAM

The gradient-weighted class activation mapping for segmentation (Seg-Grad-CAM), one of the first published approaches for explainable semantic image segmentation, has been proposed by us (Vinogradova et al. [38]). It is a modification of Grad-CAM designed for explaining the decisions of fully convolutional CNNs for semantic image segmentation. The method has the potential to be transferable to other tasks performed by fully convolutional networks as it does not impose any additional requirements on the network structure. However, to identify biases in the data, an encoder–decoder architecture is highly recommended. Seg-Grad-CAM can explain the following individual decisions of a CNN: prediction of a class label for a single pixel in the segmentation mask, prediction for a user-defined region of interest, and prediction for a selected class in the output mask. The output of Seg-Grad-CAM is a heat map similar to those of CAM and Grad-CAM. The method’s workflow of explaining a selected region of interest is shown schematically in Figure 2.3.

Table 2.1: Model summary for vanilla U-Net

Model: Vanilla U-Net			
Layer (type)	Feature maps	Parameters	Connected to
input 1 (InputLayer)	1	0	
down level 0 no 0 (Conv2D)	32	320	input 1
batch normalization 1	32	128	down level 0 no 0
activation 1	32	0	batch normalization 1
down level 0 no 1 (Conv2D)	32	9248	activation 1
batch normalization 2	32	128	down level 0 no 1
activation 2	32	0	batch normalization 2
max 0 (MaxPooling2D)	32	0	activation 2
down level 1 no 0 (Conv2D)	64	18,496	max 0
batch normalization 3	64	256	down level 1 no 0
activation 3	64	0	batch normalization 3
down level 1 no 1 (Conv2D)	64	36,928	activation 3
batch normalization 4	64	256	down level 1 no 1
activation 4	64	0	batch normalization 4
max 1 (MaxPooling2D)	64	0	activation 4
down level 2 no 0 (Conv2D)	128	73,856	max 1
batch normalization 5	128	512	down level 2 no 0
activation 5	128	0	batch normalization 5
down level 2 no 1 (Conv2D)	128	147,584	activation 5
batch normalization 6	128	512	down level 2 no 1
activation 6	128	0	batch normalization 6
max 2 (MaxPooling2D)	128	0	activation 6
down level 3 no 0 (Conv2D)	256	295,168	max 2
batch normalization 7	256	1024	down level 3 no 0
activation 7	256	0	batch normalization 7
down level 3 no 1 (Conv2D)	256	590,080	activation 7
batch normalization 8	256	1024	down level 3 no 1
activation 8	256	0	batch normalization 8
max 3 (MaxPooling2D)	256	0	activation 8
middle 1 (Conv2D)	512	1,180,160	max 3
batch normalization 9	512	2048	middle 1
activation 9	512	0	batch normalization 9
middle 2 (Conv2D)	256	1,179,904	activation 9
batch normalization 10	256	1024	middle 2
activation 10	256	0	batch normalization 10

Table 2.2: Model summary for vanilla U-Net (continued), decoder (upsampling) path

Model: Vanilla U-Net, decoder path			
Layer (type)	Feature maps	Parameters	Connected to
up sampling2d 1 (UpSampling2D)	256	0	activation 10
concatenate 1 (Concatenate)	512	0	up sampling2d 1
			activation 8
up level 3 no 0 (Conv2D)	256	1,179,904	concatenate 1
batch normalization 11	256	1024	up level 3 no 0
activation 11	256	0	batch normalization 11
up level 3 no 2 (Conv2D)	128	295,040	activation 11
batch normalization 12	128	512	up level 3 no 2
activation 12	128	0	batch normalization 12
up sampling2d 2 (UpSampling2D)	128	0	activation 12
concatenate 2 (Concatenate)	256	0	up sampling2d 2
			activation 6
up level 2 no 0 (Conv2D)	128	295,040	concatenate 2
batch normalization 13	128	512	up level 2 no 0
activation 13	128	0	batch normalization 13
up level 2 no 2 (Conv2D)	64	73,792	activation 13
batch normalization 14	64	256	up level 2 no 2
activation 14	64	0	batch normalization 14
up sampling2d 3 (UpSampling2D)	64	0	activation 14
concatenate 3 (Concatenate)	128	0	up sampling2d 3
			activation 4
up level 1 no 0 (Conv2D)	64	73,792	concatenate 3
batch normalization 15	64	256	up level 1 no 0
activation 15	64	0	batch normalization 15
up level 1 no 2 (Conv2D)	32	18,464	activation 15
batch normalization 16	32	128	up level 1 no 2
activation 16	32	0	batch normalization 16
up sampling2d 4 (UpSampling2D)	32	0	activation 16
concatenate 4 (Concatenate)	64	0	up sampling2d 4
			activation 2
up level 0 no 0 (Conv2D)	32	18,464	concatenate 4
batch normalization 17	32	128	up level 0 no 0
activation 17	32	0	batch normalization 17
up level 0 no 2 (Conv2D)	32	9248	activation 17
batch normalization 18	32	128	up level 0 no 2
activation 18	32	0	batch normalization 18
conv2d 1 (Conv2D)	Classes (11)	363	activation 18
softmax (Activation)	Classes (11)	0	conv2d 1
Total parameters: 5,505,707 Trainable parameters: 5,500,779 Nontrainable parameters: 4,928			

Table 2.3: Model summary for U-Net with VGG16 backbone. Repeating blocks in the encoder and decoder paths are omitted for space.

Model: U-Net-VGG16			
Layer (type)	Feature maps	Parameters	Connected to
input 1 (InputLayer)	3	0	
block1 conv1 (Conv2D)	64	1792	input 1
block1 conv2 (Conv2D)	64	36,928	block1 conv1
block1 pool (MaxPooling2D)	64	0	block1 conv2
block2 conv1 (Conv2D)	128	73,856	block1 pool
... (Encoder blocks)			
block5 pool (MaxPooling2D)	512	0	block5 conv3
center block1 conv (Conv2D)	512	2,359,296	block5 pool
center block1 bn (BatchNorm.)	512	2048	center block1 conv
center block1 relu (Activation)	512	0	center block1 bn
center block2 conv (Conv2D)	512	2,359,296	center block1 relu
center block2 bn (BatchNorm.)	512	2048	center block2 conv
center block2 relu (Activation)	512	0	center block2 bn
decoder stage0 upsampling	512	0	center block2 relu
decoder stage0 concat (Concatenate)	1024	0	decoder stage0 upsam.
			block5 conv3
decoder stage0a conv (Conv2D)	256	2,359,296	decoder stage0 concat
decoder stage0a bn (BatchNorm.)	256	1024	decoder stage0a conv
decoder stage0a relu (Activation)	256	0	decoder stage0a bn
decoder stage0b conv (Conv2D)	256	589,824	decoder stage0a relu
decoder stage0b bn (BatchNorm.)	256	1024	decoder stage0b conv
decoder stage0b relu (Activation)	256	0	decoder stage0b bn
decoder stage1 upsampling	256	0	decoder stage0b relu
... (Decoder blocks)			
decoder stage4b relu (Activation)	16	0	decoder stage4b bn
final conv (Conv2D)	Classes (8)	1160	decoder stage4b relu
softmax (Activation)	Classes (8)	0	final conv
Total parameters: 23,753,288 Trainable parameters: 23,749,256 Nontrainable parameters: 4,032			

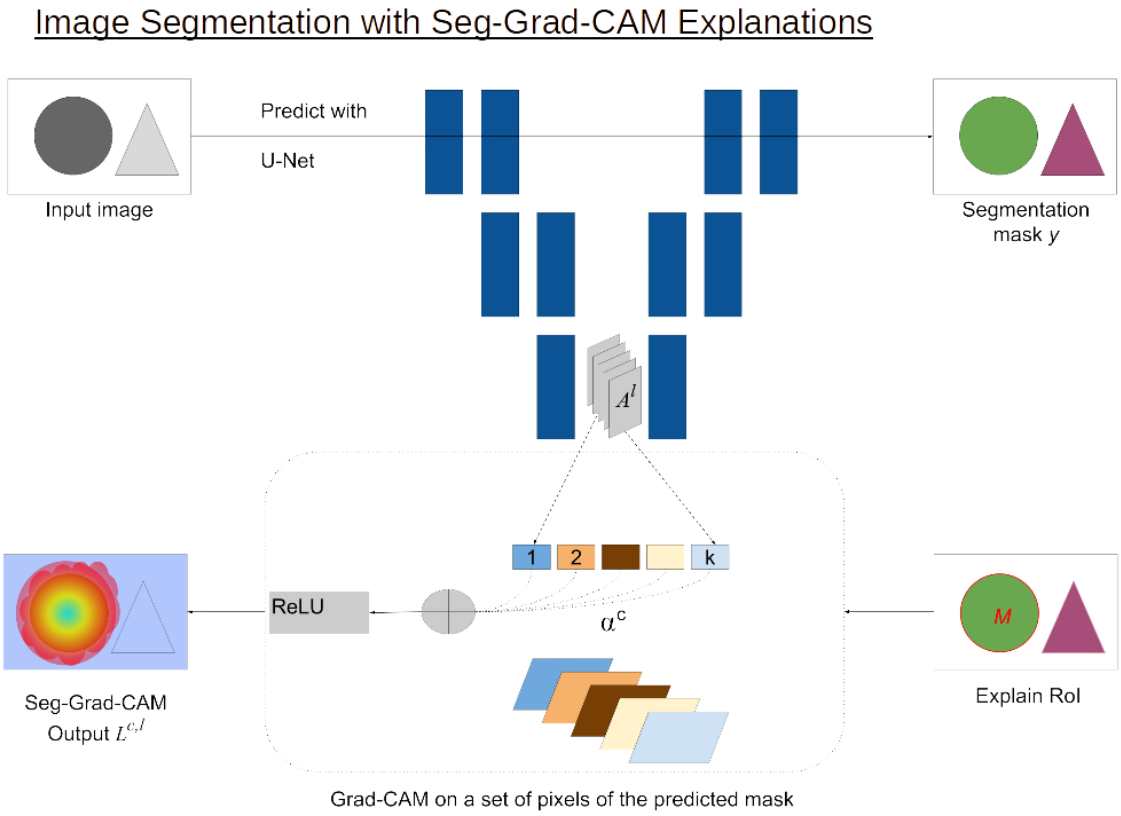

Figure 2.3: Schematic of Seg-Grad-CAM.

Concept

Seg-Grad-CAM is a class activation mapping technique like CAM and Grad-CAM, initially designed to understand the decisions of U-Net. It produces a heat map for an input picture, indicating the image areas that contribute the most to the part of the model's output selected by the user. Its central idea is to construct a class activation map using the gradients of the output pixels within the user-defined region of interest belonging to a single class (e.g., the region M drawn around the green circle in Figure 2.3) with respect to the feature maps from any intermediate convolutional layer (e.g., the feature maps A from the bottleneck in Figure 2.3).

In the technical implementation, the activation function is often placed as a separate layer after the convolutional layer to allow more control over the retrieval of outputs from the intermediate layers. This is also the case in the implementation of the CNNs described in this chapter. For brevity, the last convolutional layer of the decoder passed through the activation function (i.e., the penultimate activation layer in the technical summary of the CNN's layers: "activation 18" in Table 2.2 and "decoder stage4b relu" in Table 2.3) is referred to as "the last decoder layer" in the text.

Also, "retrieval of feature maps A_l from a convolutional layer number l" is a short equivalent of "retrieval of feature maps A_l from an activation layer named 'activation l' placed after the

convolutional layer number l." For a better understanding of the particular implementation, the name of the activation layer (e.g., "activation 18") is provided in the description.

The choice of the layer depends on the goal of the interpretation:

- To detect biased structures or frequent co-occurrence of classes, the gradients should be propagated to one of the bottleneck layers, where U-Net collects the high-level feature representations.
- To visualize the importance of input features within the region of interest, the feature maps from the last decoder layer should be used.
- To learn about the functioning of U-Net, iterative visualizations can be made for Seg-Grad-CAM for each convolutional layer and shown later in Results section.

As in Grad-CAM, the gradients of the selected prediction with respect to the feature maps are used to compute the importance weights of each feature map, which are then multiplied by the corresponding feature maps. In the last step, the weighted maps are averaged and thresholded using ReLU. The latter can also be turned off to preserve the negatively relevant features. The importance heat map is then overlaid and interpolated on the input picture. Depending on the study objectives, Seg-Grad-CAM can produce a variety of importance heat maps. It inherits the advantages of Grad-CAM.

Formal calculation

In the classification task from two previous examples (Subsections 2.3.1 and 2.3.2), the network predicted a vector of probabilities for three classes and the single class prediction was explained. Here, the task is to segment circles, triangles, and the background (Figure 2.3).

For an input image x with a dark circle and a bright triangle, U-Net (schematically shown as a structure of blue rectangles) produces the semantic segmentation mask y with logits y_{ij}^c for every pixel x_{ij} and class c.

We proposed Seg-Grad-CAM by replacing y^c in eq 2.2 with Y^c:

$$Y_{\mathcal{M}}^c = \sum_{(i,j)\in\mathcal{M}} y_{ij}^c \tag{2.6}$$

where $\mathcal{M}$ is the mask of the region of interest (a set of pixel indices of interest in the output mask).

Then, the weights α_k^c of the feature maps A of any intermediate layer l (except the input and output layers) are calculated as

$$\alpha_k^c = \frac{1}{N}\sum_{u,v}\frac{\partial Y_{\mathcal{M}}^c}{\partial A_{uv}^k} = \frac{1}{N}\sum_{u,v}\frac{\partial \sum_{(i,j)\in\mathcal{M}} y_{ij}^c}{\partial A_{uv}^k} \tag{2.7}$$

where u, v are the pixel indices in each feature map A^k, and N is the total number of pixels in A^k.

After rectification of the weighted sum, the importance heat map produced by Seg-Grad-CAM with the feature map A_l retrieved from the layer l is defined as

$$L^{c,l}_{\text{Seg-Grad-CAM}} = \text{ReLU}\Big(\sum_k \alpha^c_k A^k_l\Big) \tag{2.8}$$

By combining eqs 2.6–2.8, the importance heat map formula can be expressed as

$$L^{c,l}_{\text{Seg-Grad-CAM}} = \text{ReLU}\Big(\sum_k A^k_l \frac{1}{N} \sum_{u,v} \frac{\partial \sum_{(i,j)\in\mathcal{M}} y^c_{ij}}{\partial A^k_{uv}}\Big) \tag{2.9}$$

This allows adapting Grad-CAM to a semantic segmentation network in a flexible way because

1. A can be a feature map of any intermediate convolutional layer of interest (not only the last one as used in the standard Grad-CAM),
2. $\mathcal{M}$ can denote just a single pixel, or pixels of a region of interest (e.g., a single instance), or all pixels of the predicted mask belonging to one class.

2.4 Data

The methods used for obtaining the data and constructing the datasets are described in this section. In this project, several educational datasets were designed. Presented here are two educational datasets and one annotated public dataset with real road scenes that was used to obtain the final results.

2.4.1 Circles

The first educational dataset was designed to test the explanations of CAM and Grad-CAM for segmentation via the sliding window pixel-wise classification inspired by ref [93], as well as the explanations of Seg-Grad-CAM for assignment of a class label to a single pixel in the segmentation mask produced by U-Net[63].

The dataset consisted of 1000 generated in Python grayscale images with a size of 128×128 pixels, in which a circle having a radius of 30 pixels and a constant random gray pixel value was randomly placed upon the background having another random gray pixel value. The annotation masks were generated automatically, with the pixels inside the circle assigned to the foreground and the remaining ones — to the background (Figure 2.4).

2.4.2 TextureMNIST

This dataset has been introduced in the publication about the grid saliency method [79], which was specifically designed to find biases in the training data for segmentation tasks. The authors advertised this dataset as a benchmark for testing the ability of saliency methods to detect biases. The code for generating the data together with a set of graphical textures is available in ref [96].

Two possible scenarios for introducing the bias have been demonstrated in the study [79]: a strongly biased dataset, in which the selected bias always appears only in the data for the

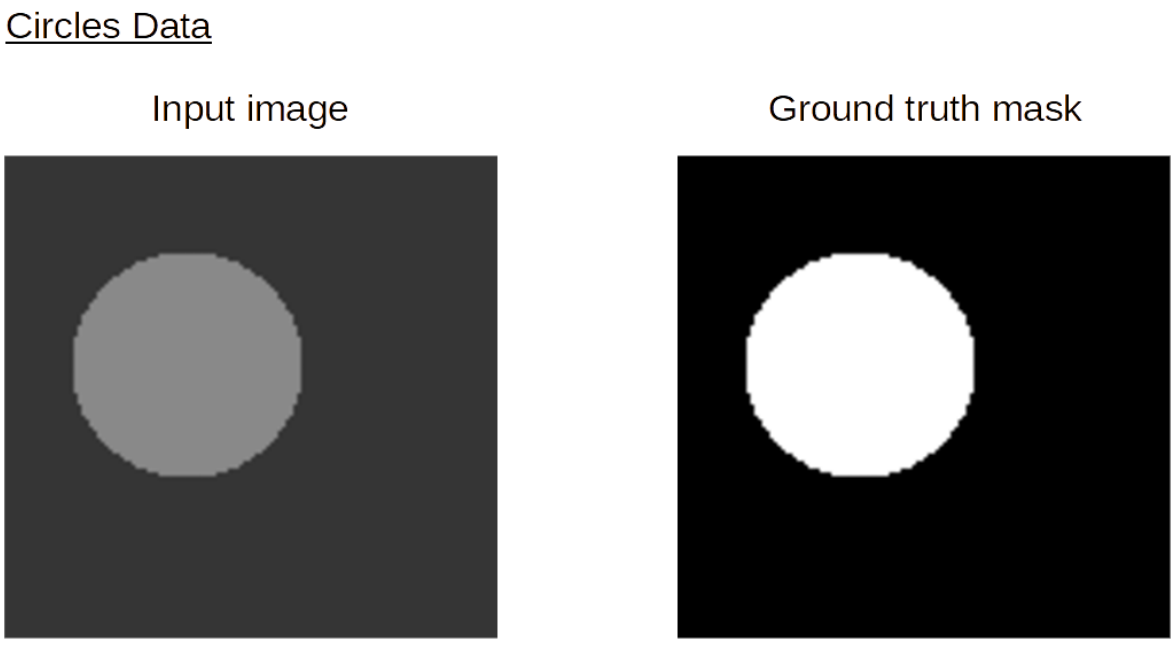

Figure 2.4: Example of the circles data

biased class, and a weakly biased dataset, where the bias always appears in the data for the biased class and may also appear together with other classes. To the best of my knowledge, as of late November 2022, there was no mention in the literature of using neither the suggested benchmark nor the grid saliency method by other researchers to compare explainability methods or saliency methods. In this thesis, this data generation technique is used as a sanity check.

In Figure 2.5, two examples from the generated set of 64×64-pixel grayscale images are shown. The handwritten digits originate from the Modified National Institute of Standards and Technology (MNIST) database [97]. The white background was replaced with two randomly selected textures from ref [96] — one positioned in the upper half of the image, the other in the lower half. The black color of the digits was replaced with another randomly chosen texture. A strongly biased dataset was used: a single texture (bottom, outlined with magenta in Figure 2.5B) always appeared either on top or in the bottom over the background in the images with digit 2, but did not appear in other images in the background, as seen in the example for digit 1 (Figure 2.5D). The dataset was constructed for the segmentation task with 11 classes: each digit from zero to nine was assigned a class from zero to nine, respectively, and the background was marked as class 10 regardless of the textures present there (Figure 2.5C,E).

2.4.3 Cityscapes

Cityscapes is a high-resolution dataset of urban road scenes photographed with a car-mounted camera. It is freely available for research purposes upon registration. The dataset was created to help advance the development of algorithms for urban scene understanding and has become one of the most widely used benchmarks in computer vision for comparing segmentation models, and for research on self-driving cars and related applications. The scenes were photographed

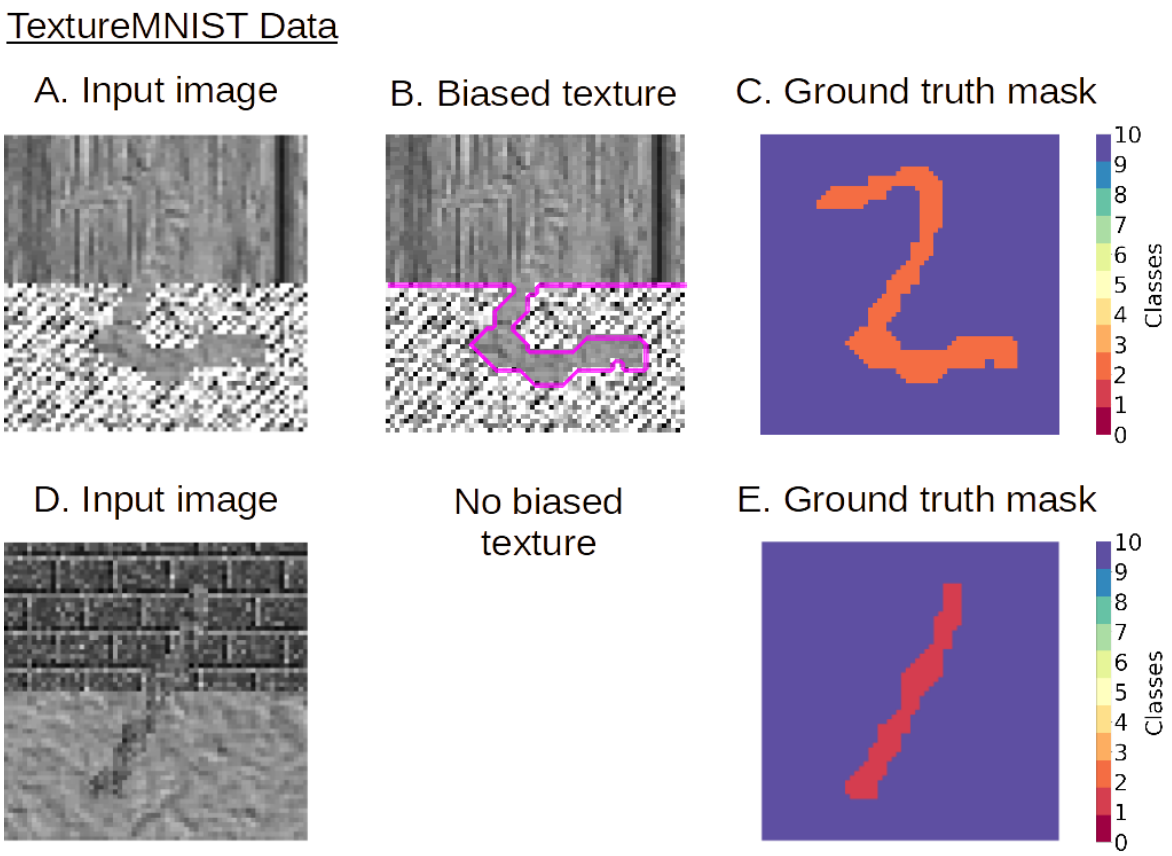

Figure 2.5: Examples of the TextureMNIST data. (A–C) The class with an intentional bias (the bottom texture separated by a magenta line in panel B), which always occurs with the class 2 in one or the other half of the background. (D, E) An example of another class, where no bias is present.

in 48 cities in Germany, in Strasbourg, and in Zurich. There are currently four versions of annotations designed for different purposes: pixel-level semantic segmentation, instance segmentation, panoptic segmentation, and 3D object detection of vehicles.

The dataset of 5000 RGB photos (2048×1024×3 pixels each) with fine-grained pixel-level annotations for 30 classes was selected for this thesis. Because the purpose of the research — the development of an explainability method of segmentation networks such as U-Net — does not include the improvement of segmentation techniques, the complexity of the training task was reduced by lowering the resolution of the images to 512×256×3 pixels using the spline interpolation of the first order (with the function "scipy.ndimage.zoom" [98]) and selecting eight categories as training labels instead of 30 classes. The categorization of the classes is summarized in Table 2.4, a downscaled photo annotated with the category labels is shown in Figure 2.6.

Table 2.4: List of classes in Cityscapes dataset grouped by category

Category	Classes
flat	road, sidewalk, parking, rail track
human	person, rider
vehicle	car, truck, bus, on rails, motorcycle, bicycle, caravan, trailer
construction	building, wall, fence, guard rail, bridge, tunnel
object	pole, pole group, traffic sign, traffic light
nature	vegetation, terrain
sky	sky
void	ground, dynamic, static

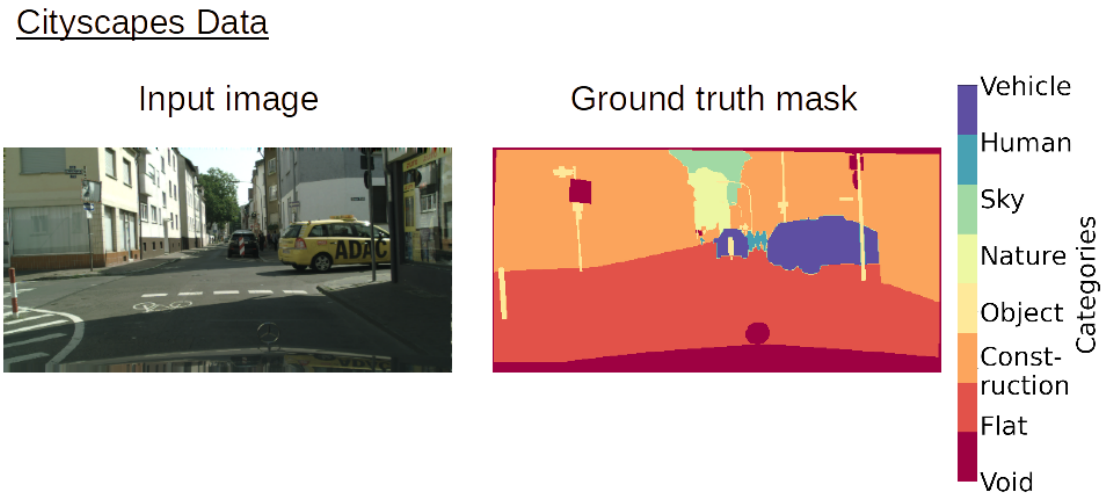

Figure 2.6: Example of the Cityscapes data annotated according to the categories (see Table 2.4).

2.5 Results

The work on this project began with studying the operation of CAM, Grad-CAM, and Seg-Grad-CAM on toy tasks of classification and segmentation of circles. These results are only briefly described in this thesis to demonstrate where the idea of taking intermediate feature maps in U-Net originated from.

2.5.1 Circles

Transforming segmentation into classification

The methods CAM (Subsection 2.3.1) and Grad-CAM (Subsection 2.3.2) were designed for image classification. However, semantic image segmentation can be seen as classification: assignment of a class label to each individual pixel of the input image. After U-Net was developed in 2015 [63], the traditional approach to semantic image segmentation became training of a fully convolutional neural network to produce a segmentation mask with each output pixel representing a class label and the output shape being equal to the input shape (alternatively, the output shape may contain one extra dimension if the class labels are one-hot encoded).

Another approach, less efficient but directly solvable as classification, is supervised training of a convolutional network with dense layers at the end to classify each pixel separately, which can be done using the "sliding window" approach [93]. In this research, this technique was implemented as follows: a small "window" area around each pixel is cropped and passed into the CNN, which learns to output a probability of belonging to the foreground for the central pixel of the received crop. To classify the pixels at the boundary of the input image, it has to be padded. The "window" slides over the input and predicts the class for the central pixel in each position of the "window." The predicted class labels at each position of the input form the output segmentation mask.

A simple CNN with three convolutional layers, global average pooling (GAP), and a dense layer was trained to "segment" circles versus the background on generated grayscale images using the "sliding window" technique for binary classification of each pixel. Because this CNN performs classification, it was possible to construct it with GAP to retrieve class activation maps. Grad-CAM was also applied to this CNN after training, which was possible because the CNN was trained for classification and contained at least one dense layer at the end. As expected, this experiment showed that the edge of the circle was important for the classification of the central pixel if the edge was present in the crop.

Retrieval of feature maps

The same task of segmenting circles versus the background was performed using the traditional approach to semantic segmentation — U-Net (Subsection 2.3.3, Table 2.1). The training of U-Net on the circles dataset described in Subsection 2.4.1 was performed using the Adam optimizer with the learning rate of 0.00001, categorical cross-entropy loss function, and pixel-wise accuracy metric. The training with two classes, categorical cross-entropy loss function, and softmax as the last activation function showed better results than the analogous procedure with one class (the probability of the pixel to belong to the foreground), binary cross-entropy loss, and the sigmoid activation function. Moreover, the gradient-weighted methods are in fact designed for cases with multiple classes. Pixel-wise accuracy is a suitable evaluation metric for simple image segmentation tasks, although it is less robust than the intersection over union (IoU). It measures the proportion of pixels in the predicted segmentation mask that are correctly classified (according to the chosen threshold, typically 0.5) compared to the ground truth mask. This metric calculates the overall accuracy of predicting the class labels for each pixel in the input. The model achieved 99.9% pixel-wise accuracy.

U-Net does not contain dense layers, therefore it was not possible to directly insert GAP to apply CAM or take the gradients in the way described in the Grad-CAM publication [43]. Considering this, Seg-Grad-CAM (Subsection 2.3.4) was applied to U-Net, with the region of interest M equal to a single pixel and the feature maps A retrieved from each of the convolutional layers iteratively. This experiment was designed to prove or refute that the proposed calculation leads to human-understandable explanations in the form of class activation maps.

Figure 2.7 shows examples of class activation maps that were obtained using Seg-Grad-CAM with the feature maps A_4, A_9, and A_{11} retrieved from the activations of a

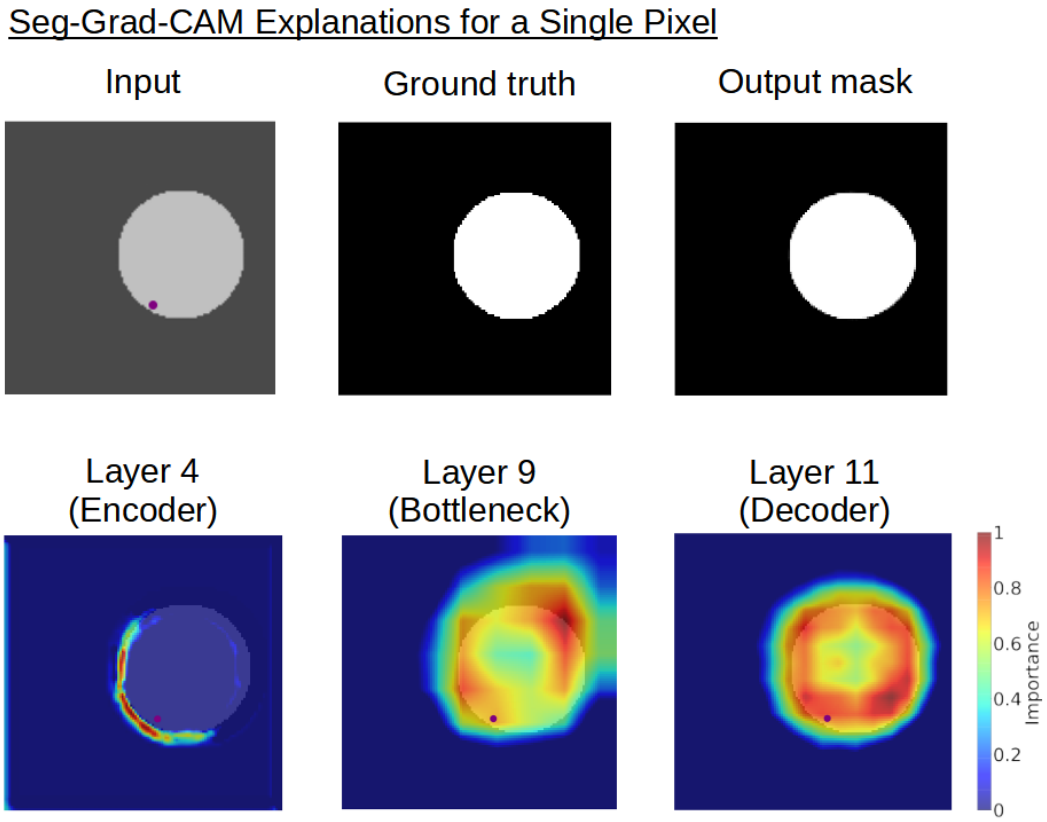

Figure 2.7: Seg-Grad-CAM explanations on the circles data show the results of taking the gradients with respect to three different sets of feature maps retrieved from three layers at different locations of the U-Net, respectively. The pixel being explained is shown in purple.

layer in the encoder path, the first (left) bottleneck layer, and a layer in the decoder path of the U-Net. These heat maps suggest that the edge of the circle "activated" the prediction of the class "circle" for this pixel. The heat maps from the last layers indicated that the whole circle was important. These three heat maps pointed to the edge(s) with very diverse visualization. From which layer the feature maps A should be taken remained a question. To determine this, the next series of experiments with another dataset is introduced.

2.5.2 TextureMNIST

The experiments described here were conducted after the Seg-Grad-CAM method was developed. The publication by Hoyer et al. [79] introducing this toy dataset and the source code for data generation (not for their method) along with the textures (released in late September 2019 [96]) appeared after the first results with Seg-Grad-CAM were obtained on other datasets. These experiments explain the intuition behind the rules for choosing the right layer for gradient propagation.

The U-Net summarized in Table 2.1 was trained to segment handwritten digits from TextureMNIST dataset with a strong bias (Subsection 2.4.2). The model was trained with the

categorical cross-entropy loss function using the Adam optimizer with the learning rate of 0.0003, softmax as the final activation function, and pixel-wise accuracy and intersection over union (IoU) as the quality evaluation metrics. The quality was additionally measured using IoU, a more robust metric commonly used for multiclass segmentation, because it provides a measure of how well the predicted segmentation mask aligns with the ground truth mask by calculating their intersection and dividing it by their union. The model achieved 97.28% pixel-wise accuracy and 94.76% IoU on the generated test dataset with 200 images.

The experiment of explaining the segmentation of the digit of class 2 is shown in Figure 2.8. The heat maps produced using Seg-Grad-CAM are organized in the shape of the U-Net with the depth of four trained in this experiment. They are positioned at the place from where the feature maps were retrieved: the output of Seg-Grad-CAM with the gradients of y_{ij}^c taken with respect to the feature map A_l, where l is the number of the layer. For example, the feature map A_2 was taken from the activation layer after the second convolutional layer with applied batch normalization — "activation 2" layer in Table 2.1) is placed as the second left heat map. Each heat map was computed independently in an iterative manner by taking the feature maps from each activation layer except the final "softmax" activation, which was used as the starting layer of the gradient propagation. The color scale denotes the importance of the pixels normalized to the range of $[0, 1]$. Each heat map was normalized independently to visualize the relative importance within one map. The predicted segmentation mask is shown in the top right corner.

The experiment shows that the explanations with the feature maps retrieved from the layers in the first two blocks of the encoder path highlight simple low-level features such as the intensities of individual pixels and a small portion of edges between the digit and the top half of the background. The heat maps that are focused on the contextual explanations (with the largest areas having high importance concentrated in the biased half of the background) are located in the bottleneck, whereas the explanations within the object (the most semantically relevant pixels within the segmentation mask) are concentrated in the last layers of the decoder path of the U-Net.

According to the experiment setup, the biased texture in the context should assist in the segmentation task and be important to a certain extent. The two bottom heat maps in Figure 2.8 demonstrate the largest areas with high importance located in the context which suggest retrieving the feature maps from the bottleneck layers to identify contextual biases.

The biased texture should be involved in the decision process but, being present in only one half of the background, it is insufficient to successfully segment the digit completely, including its part located on the second half of the background. Therefore, it is expectable that the gradients taken with respect to the feature maps of the layers in the decoder demonstrate the relevance's location inside the texture of the digit. This may suggest the usage of the last block of the decoder to find the most important features that refined the segmentation mask.

Another example from this experiment is shown in Figure 2.9: the input image with a biased texture separated by a magenta line (Figure 2.9A), the ground truth segmentation mask (Figure 2.9B), and the accurately predicted output mask (Figure 2.9C). The predicted classes

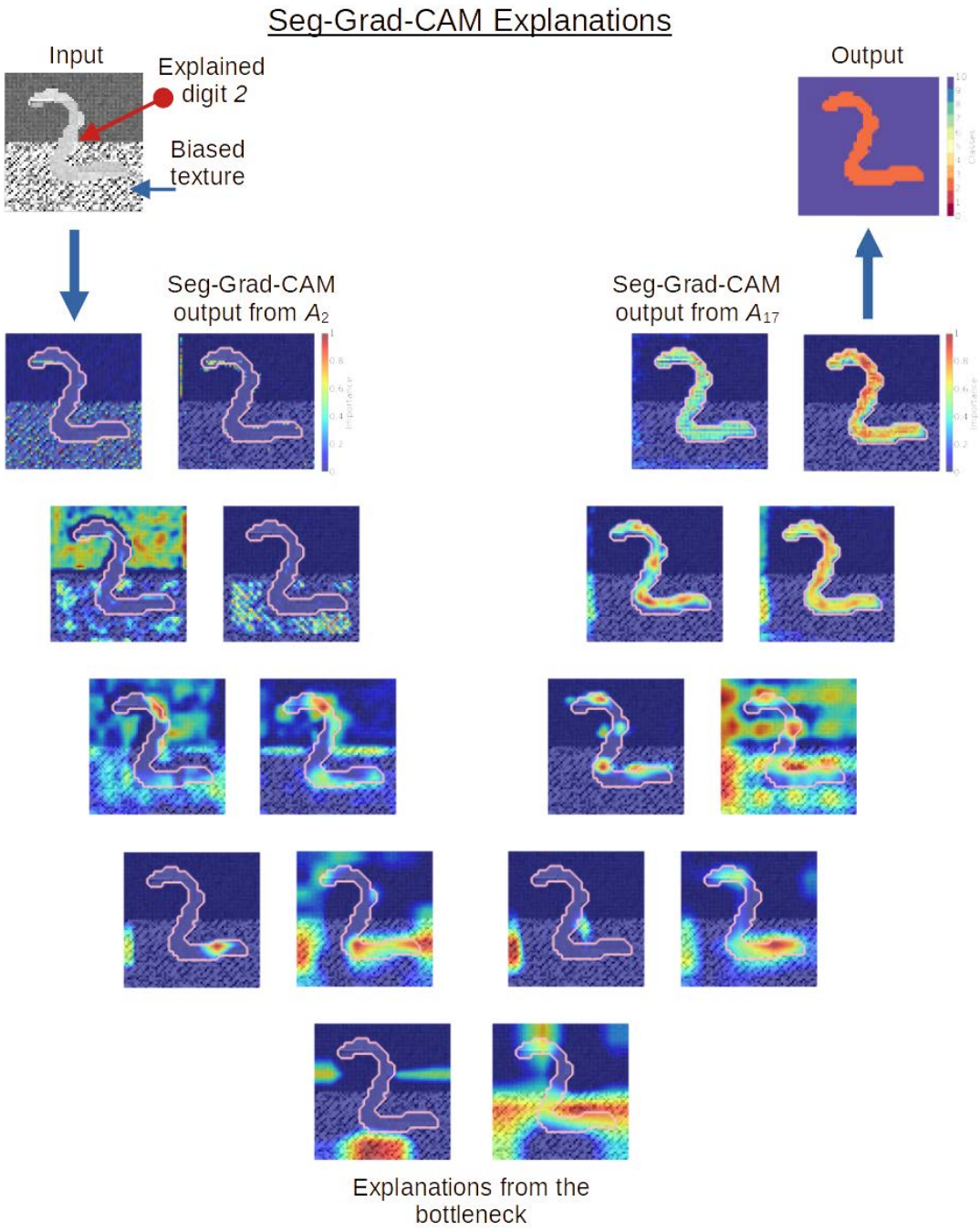

Figure 2.8: Seg-Grad-CAM explanations obtained from every layer of U-Net for the prediction of the segmentation mask of the biased class 2. The input image is at top left, the output mask is at top right with the color scale representing the segmentation classes. The heat maps produced by Seg-Grad-CAM are organized in the shape of the U-Net at the place from where the feature maps were retrieved. The heat map color scale shows the importance of the pixels, from low (blue) to high (red).

are shown according to the color scale at right. In the bottom row are shown the heat maps produced by Seg-Grad-CAM with the feature maps retrieved from the layers "activation 1" (Figure 2.9D), "activation 10" (Figure 2.9E), and "activation 18," the last layer of the decoding path before the final layer with softmax activation (Figure 2.9F), as listed in Tables 2.1 and 2.2. In eq 2.8 this corresponds to $l = 1$ (first layer), $l = 10$ (bottleneck layer), and $l = 18$ (last

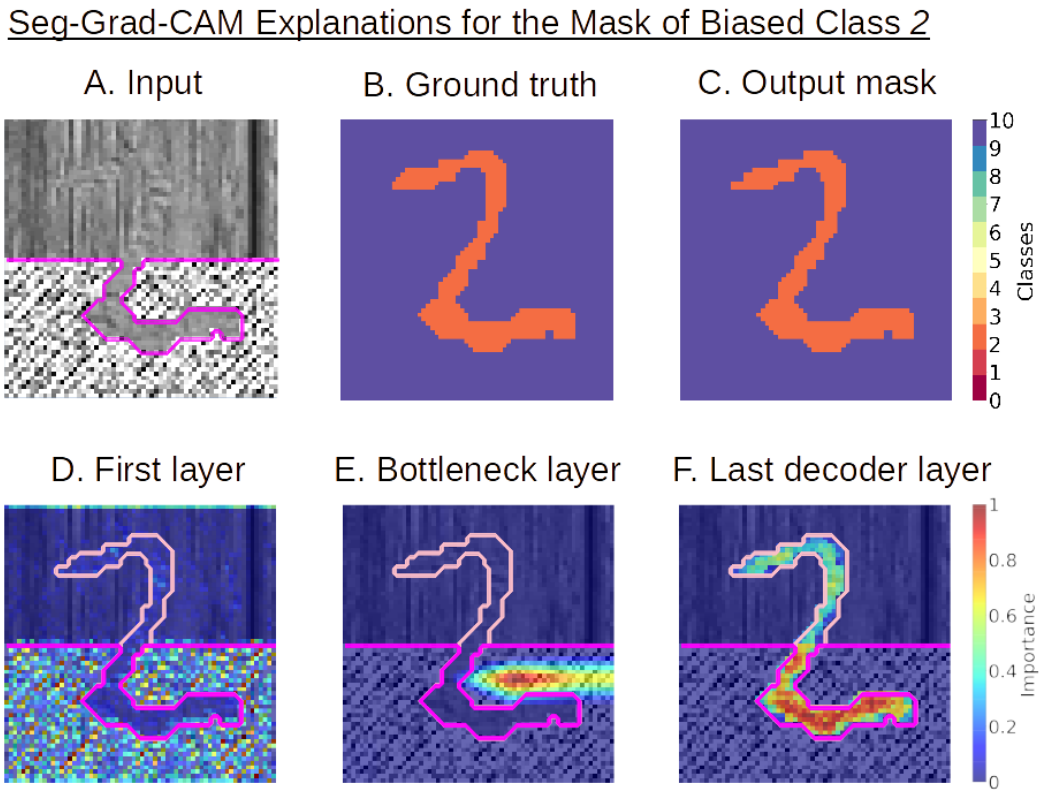

Figure 2.9: Seg-Grad-CAM explanations for the mask of the biased class. The input image (A) contains an intentional bias: the bottom texture separated by a magenta line.

layer in the decoder). To better show the localization of the pixels' importance, the predicted mask is outlined in pink, the biased texture — in magenta. The heat map color scale shows the importance of the pixels, from low (blue) to high (red). Each heat map was normalized by dividing its values by the sum of the values in that heat map because the localization of importance was desired as the output of the explainability method. The absolute values in these heat maps are not directly interpretable per se.

The biased texture in the input image (Figure 2.9A) always co-occurs with digits of class 2 either in the top or bottom half of the background. Both texture tiles of the background should be predicted as class 10 (background), as shown in the ground truth mask (Figure 2.9B). The network should not learn to separate the background parts from each other, it must segment only the digit. The output segmentation mask demonstrates that the network accomplished the task with a top quality of predictions.

The outputs of Seg-Grad-CAM were produced by starting the gradient propagation from the predictions (probabilities output by the final activation function softmax in Table 2.2) for the selected mask $\mathcal{M}$ (the region of interest) — the set of pixel indices in the mask predicted for class 2 (Figure 2.9C), setting c in eq 2.6 to class 2. The gradients propagated to all the activations iteratively. In terms of eq 2.8, the results of propagation to the layers "activation 1," "activation 10," and "activation 18" that are shown in Figure 2.9D–F were obtained by taking the gradients with respect to the feature maps A_1, A_{10}, and A_{18}.

With $A = A_1$ in eq 2.8, the importance is located inside the biased texture, which may, however, be caused by the high contrast of the pixels in this texture. Remarkably, the heat map obtained by taking the gradients with respect to A_{10} contains only a small area of high importance, unlike the heat maps at the bottom of Figure 2.8. This indicates that the explanations may differ significantly from one example to another.

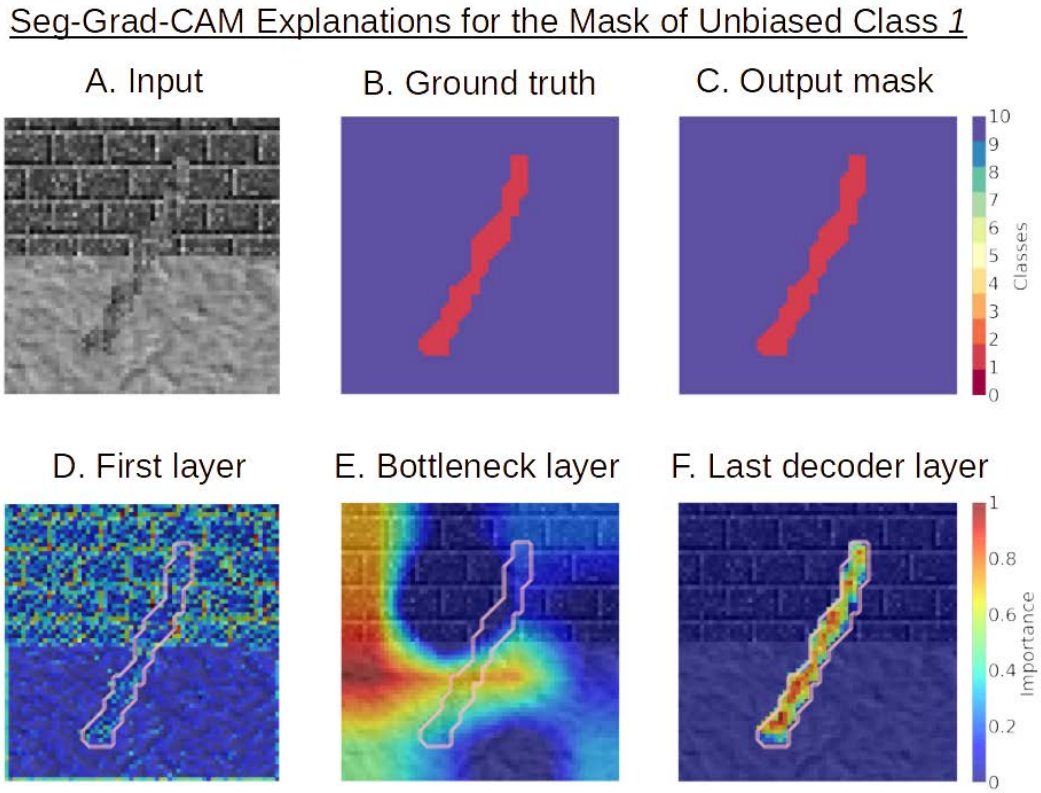

Figure 2.10: Seg-Grad-CAM explanations for the mask of the digit 1 (unbiased class).

To prove that the localization of the importance (retrieved from the bottleneck layers) inside the biased texture in Figures 2.8 and 2.9 was not coincidental, the experiment was repeated on images of other digits, where no intentional bias was introduced.

The result for the segmentation of the digit 1 is shown in Figure 2.10: the input image that did not contain any intentional bias (Figure 2.10A), the ground truth mask (Figure 2.10B), and the output segmentation mask (Figure 2.10C) are shown in the top row. The colors in the masks correspond to the predicted classes according to the color scale on the right. The outputs of Seg-Grad-CAM for $l = 1$ (first layer), 10 (bottleneck layer), and 18 (last layer in the decoder) in eq 2.8 are shown in the bottom row (Figure 2.10D–F), with the predicted mask outlined in pink.

With $A = A_1$ in eq 2.8, the importance in Figure 2.10D is located inside the top half of the background. This is likely to be caused by high-contrast edges in the texture, which are in fact highlighted in the heat map. This result supports the intuition that retrieval of the feature maps from early layers of the U-Net's encoder does not lead to meaningful explanations, but

rather to visualization of low-level image features such as edges and areas of pixels with high contrast.

According to the explanations retrieved from the bottleneck, the images did not contain any particularly important area with common semantics. The results of the repeated experiments with other unbiased data samples led to the conclusion that heat maps with fairly random localizations should indicate the absence of bias in the data.

In Figures 2.8 (top right heat map), 2.9F, and 2.10F, the relevance is located only inside the predicted mask. The importance values in the heat maps were obtained by taking the gradients of the mask prediction with respect to the activation layer after the last decoder layer. The question of whether these importance values are (a) proportional to the prediction confidence and (b) completely independent of the context is investigated on a dataset with a more complex semantics in Subsection 2.5.3. The localization of the relevance inside the object aligns with the intuition that the decoder path of encoder–decoder architectures is responsible for fine-graining of the predicted mask.

The explanations for the class assignment to a single pixel are shown in Figure 2.11. In terms of eq 2.6, an explanation for a single pixel is

$$Y^c_{\mathcal{M}} = \sum_{(i,j)\in\mathcal{M}} y^c_{ij} = y^c_{ij} \tag{2.10}$$

where i, j are the indices of the pixel of class c in the two-dimensional segmentation mask y and $\mathcal{M}$ is the set of pixel indices of interest in the output mask, consisting of only one pixel.

The input image, with the biased texture outlined in magenta and the prediction mask in pink, is shown in Figure 2.11A. The pixel of the class of interest located approximately in the middle of the picture is indicated by a red dot. Figure 2.11B,C shows the explanations of Seg-Grad-CAM for the prediction of class 2 for this pixel (white dot) that were obtained with respect to the activation layers A_9 and A_{10} of each of the two bottleneck layers ("activation 9" and "activation 10" in Table 2.1). Different colors for the dot were chosen to avoid ambiguity in the images onto which the dot was overlaid. The explanations for the pixel of class 1 were retrieved in the same manner. The input image (Figure 2.11D) does not contain any bias.

The heat maps showing the importance for the prediction for the pixel of the unbiased class (Figure 2.11E,F) do not concentrate on features that could be directly understood by a human, whereas both explanations of the prediction for the pixel of the biased class (Figure 2.11B,C) demonstrate that the highest importance was assigned to the areas inside the biased texture.

The results of these experiments show that retrieval of explanations from the bottleneck layers can help to locate biased structures in the input image that co-occur in the context with the class of interest. To identify the most important features for fine-graining of the segmentation mask, the feature maps from the last block of the decoder may be used. However, a suspicion may arise that such explanations visualize only the probability map of class assignments for each pixel. This question is discussed in more detail in the next subsection, in which the results obtained on the real-world dataset with semantically complex concepts are presented.

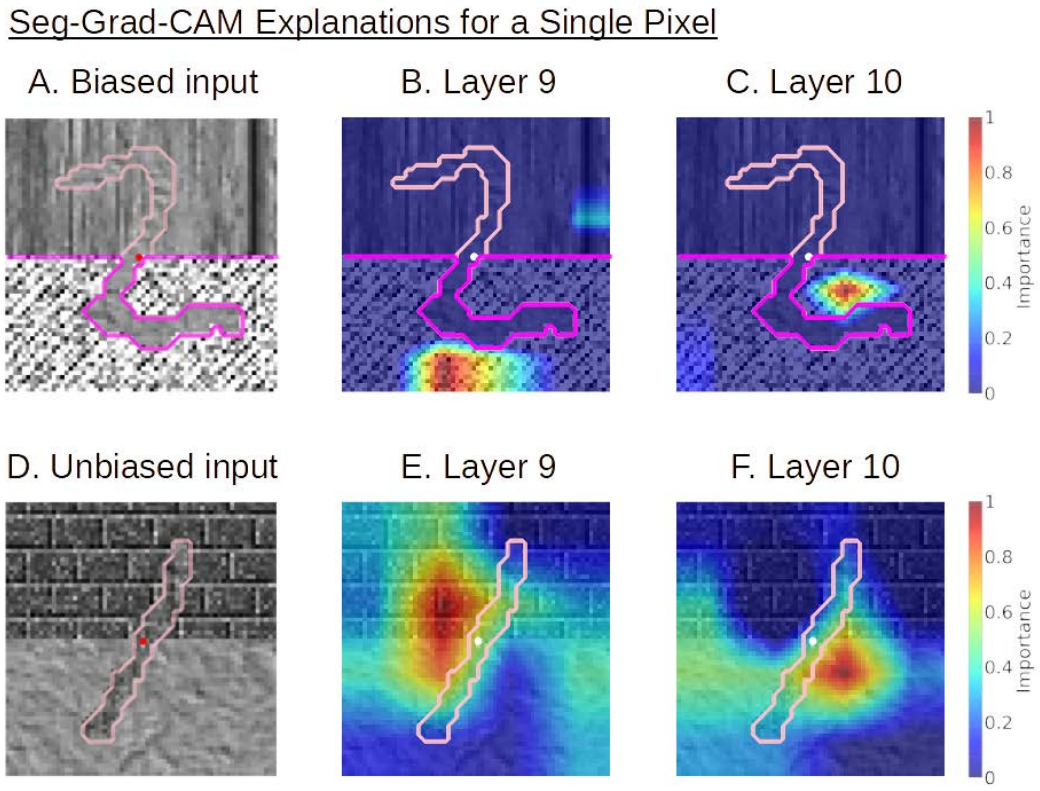

Figure 2.11: Seg-Grad-CAM bottleneck explanations of the U-Net predictions for a single pixel.

2.5.3 Cityscapes

The experiments whose results are reported in this subsection were conducted on Cityscapes, a dataset of real photographs of street scenes with pixel-wise annotations for semantic segmentation (Subsection 2.4.3).

The results of Seg-Grad-CAM explanations are demonstrated on two trained networks: "vanilla" U-Net (Tables 2.1 and 2.2) and U-Net with VGG16 backbone (Table 2.3).

Vanilla U-Net was trained in a similar manner as described in Subsection 2.5.2 and achieved IoU = 62.23% on the validation set, while performing poorly on the test set, however, it was in fact desirable to train a model to utilize biases in the seen data in order to observe these biases in explanations. U-Net with VGG16 backbone (U-Net-VGG16) was trained to overfit on the seen data as well and scored with IoU = 78.75% on validation. U-Net-VGG16 was trained using the Adam optimizer with a learning rate of 0.00045 and the compound loss function equal to the sum of the dice loss and categorical focal loss, as shown in the code examples in ref [95]. The weights of VGG16 were pretrained on the classification task on ImageNet dataset [10]. Although better models exist for this segmentation task, the art of training models for perfect segmentation is beyond the scope of this research. The explanations are shown on the first image of the validation set "frankfurt_000000_000294", on which both networks performed well (73.75% IoU by vanilla U-Net and 89.94% IoU by U-Net-VGG16) to investigate potential biases present in this image that contributed to such high performance.

Figure 2.12A,B allows visual judgment of the segmentation masks predicted by vanilla U-Net and U-Net-VGG16. The predicted categories are shown in the color scale at right (training classes are referred to as categories to avoid confusion with the original classes in the dataset listed in Table 2.4). The original input image and ground truth mask for these categories are shown in Figure 2.6. The visual judgment suggests that vanilla U-Net segmented large objects reasonably well but did not master fine-grained segmentation of small and thin structures such as those of the category "object," shown in yellow. The boundaries between the objects having different labels are not always sharp, although, by and large, the predicted mask captures the scene. The predictions of U-Net-VGG16 are substantially better in defining the boundaries (e.g., of the wheels of the nearest car) and thin structures of the pillars ("object" category).

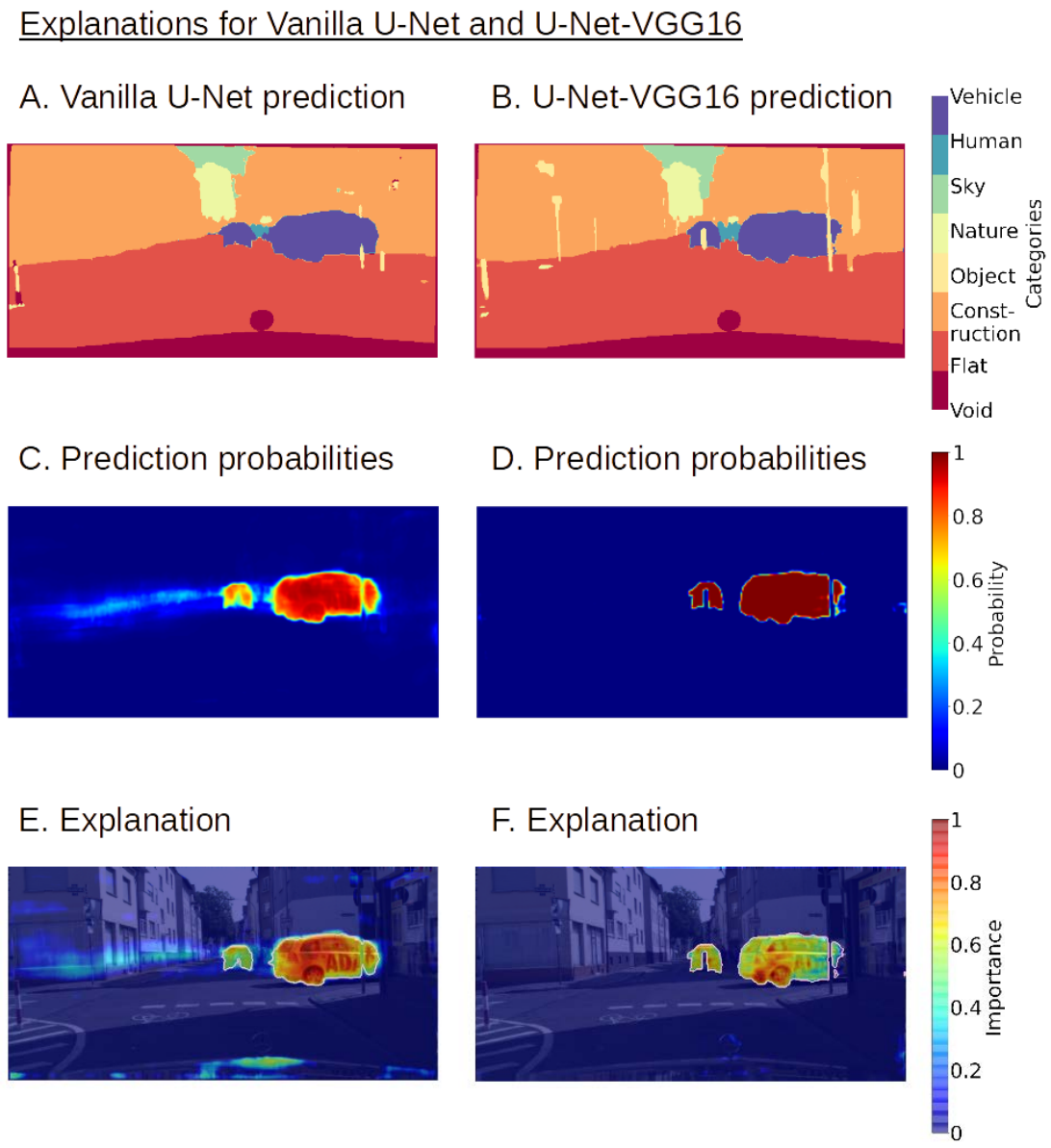

Figure 2.12: Comparison of the prediction probability maps and Seg-Grad-CAM explanations for vanilla U-Net and U-Net-VGG16 trained on Cityscapes. The explanations are retrieved by choosing the feature maps from the last layer of the decoding path of the respective network.

Figures 2.12C,D visualize the probability maps of assignment of the category "vehicle" (shown in violet in Figure 2.12A,B) for each pixel, with the probability scale at right. U-Net-VGG16 predicted the vehicles with a probability close to 1. Vanilla U-Net, on the other hand, assigned a probability of about 0.9 to the pixels of the near car, was "uncertain" about the far car, and predicted a probability lower than 0.5 for the edge between the ground floor of the building and the road.

Figures 2.12E,F show the explanations of Seg-Grad-CAM obtained with respect to the last decoder layers: "activation 18" in vanilla U-Net (Figure 2.12E) and "decoder stage4b relu" in U-Net-VGG16 (Figure 2.12F). The explanations are overlaid onto the input image to increase understanding of the relevance of particular input features. The final prediction masks for the category "vehicle" are outlined by a thin pink contour. The color scale at right denotes the importance of the pixels normalized to the range of $[0, 1]$. The color scheme is identical to that of the prediction probabilities to allow a straightforward comparison of the explanations with the probability maps for the category being explained.

The prediction probabilities and explanation for vanilla U-Net (Figure 2.12C,E) have both visual similarities and significant differences. The hood of the car where the camera was placed is highlighted in the explanation despite not being predicted, as well as parts of the ground floor of the buildings. The highlighted area of the buildings on the left is located slightly higher in the explanation map than in the probability map and focuses more on the buildings rather than the road. Figure 2.12E suggests that the individual features highlighted on the near car, such as the wheel, windows, and letters "AD" may have been learned by the last decoder layer of vanilla U-Net. "ADAC" lettering on the near car points to Allgemeiner Deutscher Automobil-Club which has a large fleet of vehicles assisting in road safety and helping the members of the club in case of vehicle malfunctions. It is likely that such ADAC vehicles appeared in the training set because the dataset was collected in Germany and two nearby cities. The letters "AD" also received the highest probability in the prediction, which is seen in Figure 2.12C, where the heat map is not overlaid on top of the input image so that only the probability map is observed.

Figure 2.12F shows that the explanation heat map lies within the area of the predicted mask and that the explanation is not identical to the probability map (Figure 2.12D). According to the explanation in Figure 2.12F, the wheels of the cars, the corner of the window of the near car, and the rear light are the features that the last decoder layer of U-Net-VGG16 utilizes to output the final segmentation mask.

A comparison of Figure 2.12E with Figure 2.12F suggests that Seg-Grad-CAM explanations of the prediction of the network that made more accurate predictions, U-Net-VGG16, exhibit finer typical class features when the feature maps are retrieved from the last decoder layer. This is likely to indicate that U-Net-VGG16 has learned the features that describe the objects of the class (category) more accurately than vanilla U-Net, and that these learned features helped to achieve a higher quality of predictions according to the measured metric (IoU=89,94% versus IoU=73,75%). These features could have been also encoded in the pretrained on ImageNet VGG16 backbone.

Overall, Figure 2.12 demonstrates that:

1. Explanation heat maps obtained by taking the gradients with respect to the feature maps of the last decoder layer may be used to explain the decisions of a U-Net-like segmentation CNN on a real-world dataset;
2. The highlighted parts of the image may represent the distinctive features of the predicted class;
3. Seg-Grad-CAM explanations provide more insights into the CNN's decision-making process than a simple extraction of the prediction probabilities;
4. The explanation heat maps differ from the prediction probability maps;
5. Seg-Grad-CAM can be used to compare the reasoning of trained networks;
6. The explanations of the decisions of the network that performed the segmentation more accurately highlight more fine-grained features than those of the network with lower performance.

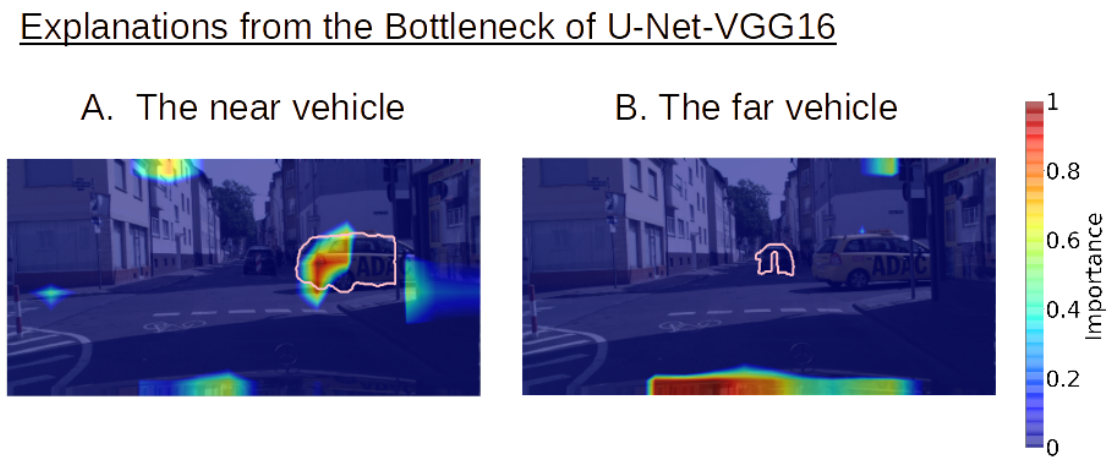

Figure 2.13: Explanations of the predictions of U-Net-VGG16 trained on Cityscapes. The feature maps were retrieved from the bottleneck activation.

Figure 2.13 shows the outputs of Seg-Grad-CAM obtained by selecting the feature maps from the bottleneck activation layer "center block1 relu" of U-Net-VGG16 (Table 2.3). To obtain the heat map shown in Figure 2.13A, the region of interest $\mathcal{M}$ in eq 2.6 was defined as the set of pixels belonging to the prediction mask for the car located closer to the camera. This set of pixels was defined by finding the largest connected component in the prediction mask using the function "skimage.measure.label" from Scikit-image library [99]. The region of interest $\mathcal{M}$ is outlined by a pink contour in the figure.

Similarly, to obtain the heat map shown in Figure 2.13B, the region of interest $\mathcal{M}$ in eq 2.6 was defined as the set of pixels belonging to the car located far from the camera (in the center of the picture) in the prediction mask. The region of the prediction that is being explained is outlined by a pink contour.

Figure 2.13A suggests that no strong contextual bias affected the prediction. Contrary to that, Figure 2.13B shows high importance assigned to the hood of the car that carried the camera. This can point to the bias in the dataset because this hood is observable in all pictures. In addition, because the data collection took place during driving on the streets, other vehicles often appeared in front of the camera in the middle of the image. The network may have exploited the bias (the position of the hood on the bottom of the image) to segment the vehicles in such cases.

This result suggests that Seg-Grad-CAM may be used to uncover biases in the data.

2.6 Applications

It has been demonstrated by scientists from around the globe that Seg-Grad-CAM is applicable to various scientific and practical problems that require explainability in critical domains. Some of the applications are touched upon here.

Suryani et al. integrated Seg-Grad-CAM into Semantic-ResNet101-FPN, a semantic segmentation CNN [100]. They trained the CNN to localize lung tumors in chest X-ray images and used Seg-Grad-CAM visualize the model predictions in a human-interpretable way. Their system of lung tumor detection is reported to operate at Dalin Tzu Chi Hospital in Taiwan, efficiently assisting medical doctors in diagnostic and helping to make treatment decisions.

Wu et al. developed a boundary-aware grid contextual attention net for osteosarcoma segmentation in magnetic resonance images and explained the necessity of each of the three proposed blocks in the architecture using Seg-Grad-CAM visualizations [101]. Seg-Grad-CAM combined with 3D-Grad-CAM and a newly proposed rule for taking the gradients was applied to segmentation with U-Net of multiparametric magnetic resonance images of a prostate tumor [83]. In the glottis bioimage segmentation task, Seg-Grad-CAM assisted in the study of the latent space and in optimization of the encoder–decoder architectures [102]. Jang et al. rigorously investigated how the choice of types of fluorescence microscopy datasets affects what features a U-Net-like network learns to utilize for segmentation of cells [103]. Using Seg-Grad-CAM, the researchers demonstrated what each convolutional block of the encoder learned, additionally proving that training on the multiple-microscopy-type dataset was the most efficient.

As mentioned in Subsection 2.4.3, Cityscapes dataset contains annotations for other types of segmentation tasks. Recently, Seg-Grad-CAM was applied to the task of panoptic video segmentation [104] to explain what the proposed panoptic slots have learned by showing the most contributing region to a specific panoptic object.

For the task of fingerprint segmentation from the background noise and alignment of fingerprint features on images from two different sensors, a recurrent adversarial network consisting of a segmentation network and a discriminator has been proposed [105]. Seg-Grad-CAM was used to provide insights into the learned segmentation and compare the proposed method with three previous state-of-the-art segmentation techniques in terms of which features were important for the predictions of the models. Seg-Grad-CAM was also used to demonstrate the

insights on modeling data uncertainty with a deep Bayesian network applied to the fingerprint segmentation task [106].

Similarly, Zhang et al. proposed a generative adversarial network (GAN) with a segmentor instead of a classifier for the role of the discriminator [107]. This network was designed to generate an image without brain pathology based on the pathological image of the patient. The features learned by the segmentor were explained using Seg-Grad-CAM and compared to the features learned by the classifier in the standard GAN visualized using Grad-CAM. These visual explanations in the form of heat maps supported the motivation to replace the classifier with the segmentor to create a GAN that better understands the difference between the images with and without lesions.

Besides the classical scenario of using Seg-Grad-CAM to explain segmentation predictions, the method can assist in choosing the way of preparation of the training dataset [103] and the neural network architecture [105], [107]. The results in refs [101], [102], [103], and [104] demonstrate that propagation of the gradients to different layers provides insights into what an individual layer learns.

These numerous applications prove the practicality and reliability of the method proposed in this chapter.

2.7 Conclusions

This research proves that Seg-Grad-CAM is a valid approach to explaining predictions of U-Net-based neural networks trained for semantic image segmentation. The method can be used for explaining the predictions for individual pixels or for large objects of interest, and is able to identify the influence of surrounding pixels inside and outside the region of interest. The approach can be applied to any CNN with an encoder–decoder architecture, including U-Net with a pretrained backbone, with only one requirement: the layers of the network must be accessible from the code. The idea of Seg-Grad-CAM was welcomed by the community and used in numerous applications, mostly in healthcare.

Extendability of Grad-CAM to segmentation CNNs was achieved by changing the rules of taking gradients regarding (1) which part of the prediction to take instead of the single class logit of the last dense layer and (2) from which layer to extract the feature maps (i.e., to which layer the gradients should be propagated). The first modification concerns the selection of the region of interest for which the user wishes to explain the CNN's predictions. The class probabilities in the final convolutional layer were used as the starting point of the gradient propagation instead of the class logit in the vector of class probabilities, as it was done in Grad-CAM for classification. The second modification was necessary to adapt Grad-CAM to segmentation because the architecture of U-Net-like networks is different from that of the CNNs for classification.

It was shown that extraction of feature maps from the early layers (the encoder part of U-Net) leads to explanations of low-level image features such as edges, which aligns with the general understanding that the earliest convolutional filters of CNNs act as edge detectors. These explanations were found to be the least informative. Choosing the feature maps from

the bottleneck layers helps to identify contextual biases in the training data (e.g., if another class co-occurs with the class of interest and "assists" in the segmentation procedure, as in the case of a car in the middle of the image being explained, where the highlighted pixels of the hood of the photographer's car indicate that the CNN may have learned the bias of this dataset: this hood appears in every photo). Last but not least, the final layers of the decoder part can be used to locate the areas inside the region of interest that were the most important for the segmentation of the region. Such areas may be interpreted as typical features of the study subject in this particular dataset.

The results suggest that it is worth investigating the heat maps with the feature maps obtained from different layers to get a more complete picture of what the CNN learns; it is especially insightful to select the bottleneck to check whether any crucial contextual bias is present in the data. Finding the precise rules for choosing the layers and combining explanations obtained from different layers is a part of future work.

The visual quality of explanations may depend on the quality of predictions. However, the quality of explanations itself is hard to quantify: there is no metric widely adopted in the community for comparison of XAI methods for semantic segmentation. As with the methods, the quality metrics proposed for XAI techniques for classification may be transferred to the new domain later.

Overall, this research helped to reach a new level of explainability in semantic image segmentation. Currently, as the field of explainable image segmentation is in its early development stage and the available methods are not free of shortcomings, Seg-Grad-CAM has key practical advantages: it is post hoc and allows retrieval of explanations for various purposes.

3

Estimation of Aberrations

3.1 Abstract

In this chapter, I present a study of estimation of optical aberrations by defining present Zernike polynomials and their amplitudes with a deep convolutional neural network trained for multitarget 3D image regression in a supervised manner. This project continues the research by Saha et al. published in 2020 [108] and proposes modifications to their method called PhaseNet to apply it to nontrivial objects. PhaseNet was trained to output a vector of amplitudes based on a 3D input image of a neglectably tiny spherical fluorescent object (bead), often called a point light source. PhaseNet is proven to work fast and efficiently on fluorescent beads but has not been tested on other objects before.

One way to extend PhaseNet to biological samples is to integrate point light sources into a sample to measure aberrations on them, which involves tedious manual work and may be unsuitable for certain study subjects. Another approach is to retrain PhaseNet on the images of fluorescent biosamples to allow it to learn how aberrations affect visualization of fluorescent structures. My work follows the latter computational approach.

Transferring a method based on deep learning from one application domain to another is not always a trivial task. A training setup may give incredibly accurate results in one narrow domain out of the infinity of prospective applications. Adapting applicability to other domains may involve tedious work on fine-tuning, searching for the optimal set of parameters, and preprocessing the data in both machine learning and domain knowledge perspectives.

To address this, I introduce additional methods to be combined with PhaseNet, define the conditions for applying PhaseNet to microscopic biological samples, and demonstrate the usage of PhaseNet's predictions for image restoration. A part of this work was published at the 7th International Conference on Frontiers of Signal Processing (ICFSP) in September 2022[109].

3.2 Related Work

Adaptive optics

Adaptive optics (AO) is a technology that helps to correct optical aberrations, minimizing blur and distortion of images.

As a technique, AO was first established in astronomy. In 1953, in a work devoted to improving the quality of observations of stars on large telescopes, Babcock [110] described the "seeing compensator" with the Eidophor, a special setup with a thin layer of oil on top of a reflecting mirror, invented by Fischer and Thiemann in 1942 [111]. According to Babcock, deterioration of the quality occurs because light rays pass through the turbulent atmosphere of the Earth. It is known that optical systems themselves, including human eye, also introduce aberrations [69].

One of the AO research directions is measurement or estimation of optical aberrations to ensure their precise correction later on. The devices that measure aberrations are called aberrometers or wavefront sensors. One of them is the Hartmann–Shack (or Shack–Hartmann) aberrometer, also known as the Shack–Hartmann wavefront sensor, first developed by Hartmann in 1904 [112]. Aberrometers of this kind are used, for example, in ophthalmology to measure monochromatic aberrations of the human eye [69, 113]. The Shack–Hartmann aberrometer directs a point of light onto the retina, thus creating on it a point source for the eye's optical system. The light rays from the point source pass back through the eye, revealing its monochromatic aberrations. A device called deformable mirror enables correction of the wavefront by adjusting the topography of the mirror's surface. The work of the Shack–Hartmann aberrometer with a deformable mirror is illustrated in Figure 15-12 in ref [69].

In microscopy, the Shack–Hartmann wavefront sensor is also used to quantify the specimen optical path difference of the optical components of a microscope and of microscopic biological samples [114]. In 2022, Imperato et al. proposed an advanced device — an extended-source Shack–Hartmann wavefront sensor for quantification of aberrations in deep fluorescence imaging of fixed mammalian brain slices [115]. This novel approach reduces the loss in signal quality and resolution caused by optical aberrations and light scattering deep in the brain tissue.

Another technique, smart programmable array microscope (S-PAM) [116], can do dynamic aberration correction in any thick fluorescent sample during the acquisition by implementing adaptive optics in a nonconventional confocal microscope. The system has multiple programmable confocal apertures in that light rays, which are not in focus, can be separately detected and used for the optimization of the correction performance. S-PAM, based on programmable array microscope (PAM) [117], introduced two modifications: a high-speed camera to detect out-of-focus fluorescence and a deformable mirror in the pupil plane. PAM is based on another commonly used AO component — the spatial light modulator (SLM), a device placed in the image plane to modulate the spatial distribution of the light waves. An interferometer setup can also be used to measure the wavefront and, combined with a deformable mirror, to correct it [118].

The approaches described above measure aberrations using hardware (sensors) and are called direct wavefront sensing techniques. Other methods, called indirect or sensorless wavefront measuring techniques, do not use physical sensors, relying on software.

Sensorless AO

The main advantage of sensorless methods over the direct wavefront sensing is that the former do not need special hardware or a sophisticated imaging system. This makes sensorless methods more accessible to potential end users (e.g., biologists imaging organisms) and to AO researchers by reducing the cost of experiments and of potential human mistakes. It is faster to run a computational experiment over and over again, and in case of a software mistake researchers can readily return to the previous working version, whereas a fault in optical hardware may lead to a costly replacement, and the physical world does not offer an "undo" command.

Software-based solutions do not completely eliminate the need for hardware; computations still require physical devices such as central processing units (CPUs) and, for some applications, graphical processing units (GPUs) as well. This should not hamper research because, luckily, several large software companies offer cloud platforms that provide an option to rent CPUs and GPUs and scale up the rented hardware as the research grows and more computational power is needed. However, AO methods of "sensing" the wavefront indirectly from images, which involve image processing using deep learning (DL), need a lot of training data to make sure that DL algorithms work well. The necessary amount of data can be obtained via repeated imaging on a microscope equipped with AO sensors (which may not be available to researchers at all), by augmentation of the limited number of existing experimentally aberrated images (again, if they are available), or by simulation of optical aberrations. The latter is possible without the AO devices.

Recent examples of software for wavefront measurement incorporate optimization [119, 120] or DL approaches [108, 121, 122, 123, 124, 125]. For a DL model, quantitative measurement of optical aberrations means prediction of amplitudes for each Zernike aberration mode that it was trained for. Most of those solutions were limited to fluorescent beads having a trivial shape [108, 119, 121, 122, 123]. Fluorescent beads are tiny spherical objects that radiate light waves in the shape of the point spread function (PSF) when lit with a laser in the fluorescent spectrum of the beads.

Hu et al. [124, 125] trained a convolutional neural network (CNN) to measure aberrations on beads and microtubules. In 2022, the researchers announced a novel method for training a CNN on a series of in-focus images and images with experimentally introduced aberrations on an AO microscope with an SLM [124]. They demostrated correction of five Zernike modes on the bead images and correction of nine modes on the images of microtubules. In a preprint in January 2023, these authors in cooperation with a group of fellow scientists from the University of Oxford revealed the details of the method and demonstrated diversity of the collected data [125]. The data consisted of images of microtubules with varying shapes, positions, sizes, background, and noise. The so-called pseudo-PSFs were calculated as an inverse Fourier

transform of the fraction: the Fourier transform of image I_1 with the PSF f_1 divided by the Fourier transform of image I_2 with the PSF f_2:

$$\text{pseudo-PSF} = \mathcal{F}^{-1}\left[\frac{\mathcal{F}(I_1)}{\mathcal{F}(I_2)}\right] \tag{3.1}$$

The network was trained using synthetic data generation to enlarge the dataset and therefore reduce the number of required acquisitions. The precalculated pseudo-PSFs were passed to the neural network with convolutional, max-pooling, and fully connected layers. Variations of this network were trained to predict the Zernike coefficients in a supervised manner by using the amplitudes of synthetic aberrations as the ground truth.

Another DL-based approach compared six neural networks, including five CNNs with varying depths of the architecture [123]. The networks were trained on 2D images of fluorescent beads with simulated aberrations. The results showed that the most accurate model was a relatively short CNN having only seven convolutional layers with five max poolings and three final dense layers. Another work also used only 2D images [122], which may limit application of these methods because the PSF is three-dimentional. All the DL methods mentioned above are based on supervised training of CNNs. An interesting solution using deep reinforcement learning has been recently proposed by Durech et al. [126].

This work is based on a method called PhaseNet developed by Debayan Saha, a fellow student in our lab, and his collaborators [108]. This approach is described below.

3.3 Methods

This section is based on my article [109] prepared with Eugene W. (Gene) Myers, my PhD supervisor at the Max Planck Institute of Molecular Cell Biology and Genetics (MPI-CBG), as the coauthor, and published in 2022 at the 7th International Conference on Frontiers of Signal Processing (ICFSP), IEEE. The schematics are reproduced in a different color scheme, another synthetic image as an inset (output of the generator with phantoms), and with rotated data examples in insets.

3.3.1 PhaseNet

The deep learning method used in this chapter, PhaseNet [108], is a variant of LeNet [127]. This CNN takes a stack of 3D images as an input (for example, one of the models had an input size of 32×32×32 pixels) and outputs the vector Y of amplitudes for 11 Zernike aberration types (or for another subset of aberration modes). The summary of PhaseNet model is presented in Table 3.1. Another architecture in use, called PhaseResNet in this thesis, is based on ResNet [52]. PhaseResNet is summarized in Tables 3.2 and 3.3 (continued). Figure 3.1 outlines the training procedure on synthetically generated data and usage of PhaseNet or PhaseResNet for prediction of the Zernike polynomials and their amplitudes.

PhaseNet was trained in a simulation: 3D spheres with a defined radius were synthetically aberrated. The synthetic generator produced random aberrations with the amplitudes a_j

Table 3.1: Model summary for PhaseNet

Model: PhaseNet		
Layer (type)	Output shape	Parameters
X (InputLayer)	(None, 32, 32, 32, 1)	0
conv1 (Conv3D, size 3×3×3)	(None, 32, 32, 32, 8)	224
conv2 (Conv3D, size 3×3×3)	(None, 32, 32, 32, 8)	1736
maxpool1 (MaxPooling3D, size 1×2×2)	(None, 32, 16, 16, 8)	0
conv3 (Conv3D, size 3×3×3)	(None, 32, 16, 16, 16)	3472
conv4 (Conv3D, size 3×3×3)	(None, 32, 16, 16, 16)	6928
maxpool2 (MaxPooling3D, size 1×2×2)	(None, 32, 8, 8, 16)	0
conv5 (Conv3D, size 3×3×3)	(None, 32, 8, 8, 32)	13,856
conv6 (Conv3D, size 3×3×3)	(None, 32, 8, 8, 32)	27,680
maxpool3 (MaxPooling3D, size 1×2×2)	(None, 32, 4, 4, 32)	0
conv7 (Conv3D, size 3×3×3)	(None, 32, 4, 4, 64)	55,360
conv8 (Conv3D, size 3×3×3)	(None, 32, 4, 4, 64)	110,656
maxpool4 (MaxPooling3D, size 1×2×2)	(None, 32, 2, 2, 64)	0
conv9 (Conv3D, size 3×3×3)	(None, 32, 2, 2, 128)	221,312
conv10 (Conv3D, size 3×3×3)	(None, 32, 2, 2, 128)	442,496
maxpool5 (MaxPooling3D, size 2×2×2)	(None, 16, 1, 1, 128)	0
flat (Flatten)	(None, 2048)	0
dense1 (Dense, size 64)	(None, 64)	131,136
dense2 (Dense, size 64)	(None, 64)	4160
Y (Dense, size number of modes)	(None, 11)	715
Total parameters: 1,019,731 Trainable parameters: 1,019,731 Nontrainable parameters: 0		

within the range from -0.075 to 0.075 μm. Eleven nontrivial aberration modes — oblique astigmatism, vertical astigmatism, oblique trefoil, vertical trefoil, vertical coma, horizontal coma, oblique quadrafoil, vertical quadrafoil, oblique secondary astigmatism, vertical secondary astigmatism, and primary spherical — were selected for the synthetic training.

Using the same generator, the training and validation datasets were created. PhaseNet was evaluated using (1) randomized synthetic data produced by that generator and (2) 3D images of fluorescent beads experimentally collected on widefield and point-scanning AO microscopes.

Following eq 1.7, the model predicted the amplitudes a_j for the Zernike aberration modes Z_j, and the PSF visualization and wave front reconstruction were displayed. The authors made the data and the code in Python — for the model and for synthesizing the data — publicly available on Github. For more information about the data collection and CNN training, please see the original publication [108].

3.3.2 PhaseNet data generator

PhaseNet's synthetic generator produces random PSFs. The user specifies the nomenclature (Noll and ANSI are supported), the size of the PSF to be generated, the list of aberration types, the amplitude range, and the target output size (the size to be cropped out of the PSF). If the PSF is to be cropped, the cropping location can be set using the jitter and max jitter parameters. If jitter is set to false (the default), the crop is taken from the image center; if

Table 3.2: Model summary for PhaseResNet

Model: PhaseResNet			
Layer (type)	Output shape	Param.	Connected to
X (InputLayer)	(None, 50, 50, 50, 1)	0	
conv3d1	(None, 50, 50, 50, 16)	432	X[0][0]
batchnormalization1	(None, 50, 50, 50, 16)	64	conv3d1[0][0]
activation1	(None, 50, 50, 50, 16)	0	batchnormalization1[0][0]
conv3d2	(None, 50, 50, 50, 16)	6912	activation1[0][0]
batchnormalization2	(None, 50, 50, 50, 16)	64	conv3d2[0][0]
add1	(None, 50, 50, 50, 16)	0	X[0][0]
			batchnormalization2[0][0]
activation2	(None, 50, 50, 50, 16)	0	add1[0][0]
conv3d3	(None, 50, 50, 50, 16)	6912	activation2[0][0]
batchnormalization3	(None, 50, 50, 50, 16)	64	conv3d3[0][0]
activation3	(None, 50, 50, 50, 16)	0	batchnormalization3[0][0]
conv3d4	(None, 50, 50, 50, 16)	6912	activation3[0][0]
batchnormalization4	(None, 50, 50, 50, 16)	64	conv3d4[0][0]
add2	(None, 50, 50, 50, 16)	0	activation2[0][0]
			batchnormalization4[0][0]
activation4	(None, 50, 50, 50, 16)	0	add2[0][0]
conv3d5	(None, 25, 25, 25, 32)	13,824	activation4[0][0]
batchnormalization5	(None, 25, 25, 25, 32)	128	conv3d5[0][0]
activation5	(None, 25, 25, 25, 32)	0	batchnormalization5[0][0]
conv3d7	(None, 25, 25, 25, 32)	512	activation4[0][0]
conv3d6	(None, 25, 25, 25, 32)	27,648	activation5[0][0]
batchnormalization7	(None, 25, 25, 25, 32)	128	conv3d7[0][0]
batchnormalization6	(None, 25, 25, 25, 32)	128	conv3d6[0][0]
add3	(None, 25, 25, 25, 32)	0	batchnormalization7[0][0]
			batchnormalization6[0][0]
activation6	(None, 25, 25, 25, 32)	0	add3[0][0]
conv3d8	(None, 25, 25, 25, 32)	27,648	activation6[0][0]
batchnormalization8	(None, 25, 25, 25, 32)	128	conv3d8[0][0]
activation7	(None, 25, 25, 25, 32)	0	batchnormalization8[0][0]
conv3d9	(None, 25, 25, 25, 32)	27,648	activation7[0][0]
batchnormalization9	(None, 25, 25, 25, 32)	128	conv3d9[0][0]
add4	(None, 25, 25, 25, 32)	0	activation6[0][0]
			batchnormalization9[0][0]
activation8	(None, 25, 25, 25, 32)	0	add4[0][0]
conv3d10	(None, 13, 13, 13, 64)	55,296	activation8[0][0]
batchnormalization10	(None, 13, 13, 13, 64)	256	conv3d10[0][0]
activation9	(None, 13, 13, 13, 64)	0	batchnormalization10[0][0]
conv3d12	(None, 13, 13, 13, 64)	2048	activation8[0][0]
conv3d11	(None, 13, 13, 13, 64)	110,592	activation9[0][0]
batchnormalization12	(None, 13, 13, 13, 64)	256	conv3d12[0][0]
batchnormalization11	(None, 13, 13, 13, 64)	256	conv3d11[0][0]
add5	(None, 13, 13, 13, 64)	0	batchnormalization12[0][0]
			batchnormalization11[0][0]
activation10	(None, 13, 13, 13, 64)	0	add5[0][0]
conv3d13	(None, 13, 13, 13, 64)	110,592	activation10[0][0]

Table 3.3: Model summary for PhaseResNet (continued)

Model: PhaseResNet			
Layer (type)	Output Shape	Param.	Connected to
batchnormalization13	(None, 13, 13, 13, 64)	256	conv3d13[0][0]
activation11	(None, 13, 13, 13, 64)	0	batchnormalization13[0][0]
conv3d14	(None, 13, 13, 13, 64)	110,592	activation11[0][0]
batchnormalization14	(None, 13, 13, 13, 64)	256	conv3d14[0][0]
add6	(None, 13, 13, 13, 64)	0	activation10[0][0]
			batchnormalization14[0][0]
activation12	(None, 13, 13, 13, 64)	0	add6[0][0]
globalaveragepooling3d1	(None, 64)	0	activation12[0][0]
Y (Dense)	(None, 6)	390	globalaveragepooling3d1[0][0]
Total parameters: 510,134 Trainable parameters: 509,046 Nontrainable parameters: 1,088			

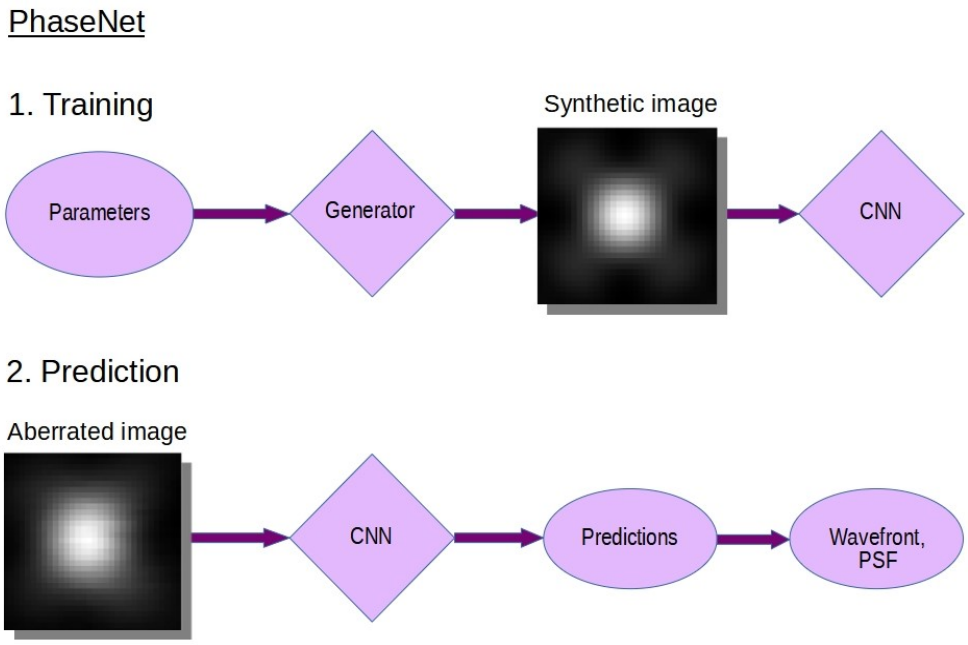

Figure 3.1: Schematic of the PhaseNet operation [108]. CNN is either PhaseNet, based on LeNet [127], or PhaseResNet, based on ResNet [52]. In training, the generator uses the input parameters of the microscope and parameters for data generation to synthesize the ground truth (aberration types with the amplitudes) and a sphere convolved with the 3D PSF. The CNN is trained on the output of the generator (spheres and the ground truth). During prediction, the CNN estimates the vector of aberration types and associated amplitudes (one floating-point number per type) for a 3D picture of a bead. At the last step, the wavefront is rendered.

jitter is set to true, the crop is taken randomly from the image. Max jitter specifies the distance at which the crop can be taken from the center. The user can select the z slices (planes) of the volumetric image as an input to the network for training (for example, only some slices in the middle). The authors of PhaseNet have proved that the best results are obtained when all z planes are utilized.

For accurate calculation of synthetic PSFs, the optical system characteristics (voxel size, light wavelength λ, refractive index, and numerical aperture of the detection objective lens (NA)) must be provided. The generator convolves a sphere of a definite radius (simulating a point source) with a synthetic PSF. After the convolution, Gaussian noise can be applied if its parameters (mean, standard deviation, and signal-to-noise ratio (SNR)) are specified. Each of these parameters should be either a single floating-point number or a range of float numbers. Gaussian blur can also be introduced.

3.3.3 Retrieval of noise parameters

PhaseNet needs data-specific parameters, such as the noise parameters, which should be retrieved from the acquired images. Other parameters like refractive index, light wavelength, numerical aperture (NA) of the detection objective lens, and voxel size are microscope-specific and their values are determined by parts of the assembly.

To determine the noise parameters, the "measure" function in Fiji software [128] was used on selected areas of the image. The high-intensity pixels in the image belonging to the study sample were assigned to the foreground, the dark parts of the image — background with noise included. The generator can accept a single value (to define the parameters for one Gaussian distribution) or a range of values (to sample from different Gaussian distributions) for each noise parameter. It may be necessary to measure these parameters several times in different parts of the image to define an approximate range, depending on the dataset. The mean intensity must be measured inside the foreground object, the mean and standard deviation of the noise — on the background free from the study sample.

The following definition of the SNR was used: the ratio of the mean intensity of the foreground to that of the noise:

$$\mathrm{SNR} = \frac{I_{\mathrm{foreground}}}{I_{\mathrm{noise}}} \tag{3.2}$$

3.3.4 Data generator with phantoms

This modification is a part of my study with Eugene W. Myers [109]. We passed a crop from the raw unaberrated image into the generator where the crop was convolved using synthetic PSFs to include the information about the object structure. An unaberrated image in our synthetic data experiments was the raw image that was downloaded. An unaberrated image in an experimental setup with real data acquired with an AO microscope would be one that was free of aberrations caused by the deformable mirror or a spatial light modulator. In contrast to the created aberrations that emerge in images only when the AO microscope is told to add them, an unaberrated image is supplied to guarantee that the only PSF present is the residual PSF of the microscope, which is observed in all data coming from this microscope.

In addition to Gaussian noise generated by the generator, our method for producing images from actual images seeks to incorporate prior information regarding the shape of the object, the residual PSF of the microscope, and other noise originating from the microscope.

The 3D images of the study samples were generated with synthetic aberrations through the following steps: cropping the input image within the generator, convolving the crop with random synthetic PSFs, and adding random Gaussian noise. The input image entered into the

generator is denoted as a **phantom**. Training in which two or more images were passed as phantoms was referred to as **training with multiple phantoms**.

3.3.5 Restoration via deconvolution

In our study [109], we showed image restoration with the PhaseNet predictions via deconvolution.

The aim of predicting aberrations is to achieve high-quality restoration of affected images, which can be done via deconvolution. The purpose of this operation that is inverse to convolution, is to restore a convolved image to its state before it was convolved with the respective filter (kernel). The PSFs (the synthetically generated PSFs and the residual ones) take the role of the convolutional filters. To restore aberrated images, they can be deconvolved with the predicted PSFs.

In our work [109], a Python-based library Flowdec [129], for 3D deconvolution via the Richardson–Lucy algorithm [130, 131] was used for restoration. Compared to commonly used DeconvolutionLab2 [132], Flowdec performs faster, accepts the synthetically generated PSFs, and can be called directly from Python scripts or Jupyter notebooks. The comparison of Flowdec with DeconvolutionLab2 [132] by Eric Czech et al. [129], revealed that Flowdec was 55 times faster than DeconvolutionLab2: a sample task was accomplished in 729 and 40,263 ms, respectively. The tests were completed by the authors under the following conditions: Jupyter Notebook, 64 GB RAM, Intel Xeon CPU (x2), and Nvidia GTX 1080 GPU (x2).

The described approach of training with phantoms along with the restoration task is schematically shown in Figure 3.2.

3.3.6 Convolution with the “zero” synthetic PSF

This novel method for preprocessing experimental data, first proposed in this dissertation, has not been previously reported.

Synthetic data for training the CNN and experimental data used for testing it may belong to different domains and have significant visual differences. This issue can be addressed by:

1. Making synthetic data so realistic that synthetic images cannot be distinguished from real ones;

2. Making real data look like synthetic data.

The first solution is hard to implement in practice, whereas the second one is relatively straightforward.

For each image, the data generator computes a random PSF within the range defined by the user. In addition, the theoretical residual PSF is calculated according to the microscope’s characteristics, which is the same for all generated images.

In the **“zero” convolution method**, real test images are passed to the generator along with the microscope’s parameters and the vector of introduced amplitudes set to zero. The generator calculates only the residual PSF, because all other Zernike polynomials are set to zero, and then convolves it with the real input images. As a result, the output images look like synthetic ones.

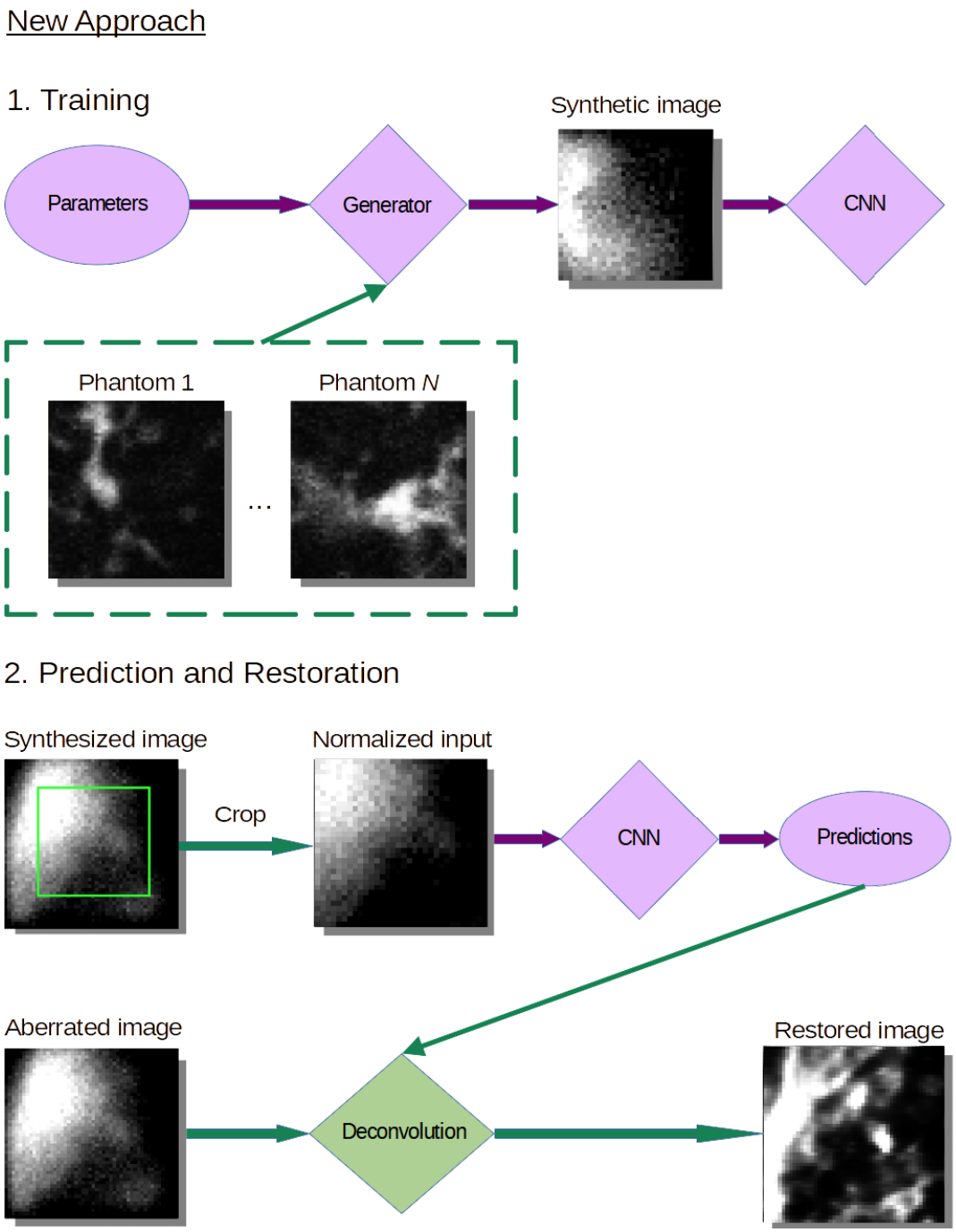

Figure 3.2: Schematic of CNN training with phantoms and the data generator followed by image restoration. Training procedure: the generator receives as inputs the phantoms (unaberrated 3D images of the specimens), data generating parameters, and microscope's parameters. It convolves the PSF with a randomly selected phantom's 3D crop at each iteration, producing an aberrated image along with the ground truth that is then fed to the CNN for supervised training. Restoration: given an aberrated image of the sample, the trained CNN predicts the vector of the amplitudes for each aberration mode. The PSF, which can be utilized as a deconvolution kernel in restoration, is created from the CNN predictions. The Python library Flowdec was the tool for completing the restoration task. Our modifications of PhaseNet are shown in green.

3.4 Data

The simplest way to prove or refute the extendability of PhaseNet to objects of a more complex shape than beads could be to computationally introduce optical aberrations to an

image of a complex object and build a corresponding dataset. The alternatives are to find a dataset with images experimentally aberrated on an AO microscope or to acquire such images in own study. As such a dataset was not publicly available during the time of this research (January 2021 – July 2022), I used computational simulation of aberrations (astrocytes data) on publicly available data [133], images from an unpublished in-house AO microscope (*Drosophila* data) acquired by Debayan Saha, and unpublished data from external collaborators Na Ji and Qinrong Zhang at the University of California, Berkeley (neurons data). In addition, another set of images of beads was acquired by Debayan Saha on the in-house AO microscope (fluorescent beads data) to test some hypotheses. The microscope was built by my colleagues Debayan Saha, Nicola Maghelli, and Sergei Klykov. Debayan Saha instructed me on the sample preparation and data acquisition protocols, according to which I attempted to collect datasets myself. However, the images acquired by D. Saha are used in this work because of their superior quality.

3.4.1 Astrocytes (synthetic data)

To introduce aberrations by generating them synthetically, 3D images with known characteristics of the microscope (voxel size, refractive index, numerical aperture (NA), and light wavelength λ) are required. Suitable data were found in Cell Image Library, a public collection of bioimages. The project by Bushong et al. [133] includes a large number of images of protoplasmic astrocytes in the hippocampus of *Rattus norvegicus*, ranging the age of specimens and the image resolution, acquired using two fluorescent dyes. For the experiments described in this chapter, two images [134, 135] were selected (Figure 3.3). To ensure that a model trained on one image can be run on the other one, the selected images have an identical voxel size and show specimens taken from the rats of the same age and injected with one dye.

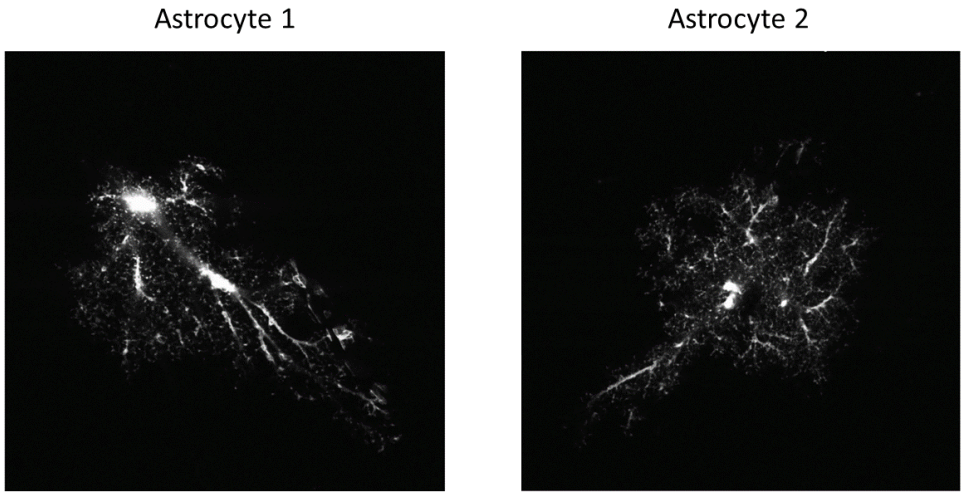

Figure 3.3: Central slices of the original images: astrocyte 1 [134] and astrocyte 2 [135].

The two selected images of astrocytes, having a voxel size of (0.068519, 0.068519, 0.2) μm in the (x, y, z) coordinates, are of a 4-week-old young adult rat's hippocampal area CA1 injected with Lucifer yellow [136] ($\lambda = 488$ nm) intracellularly. A single photon confocal microscope

Biorad Radiance2000 with NA = 1.4 and oil immersion medium (refractive index = 1.5) was used for the data acquisition. The details of the sample preparation and data acquisition are available in refs [133, 134, 135].

The astrocytes occupied only about 20–25% of the image area. To ensure the presence of the object during training, the images were cropped (Figure 3.4).

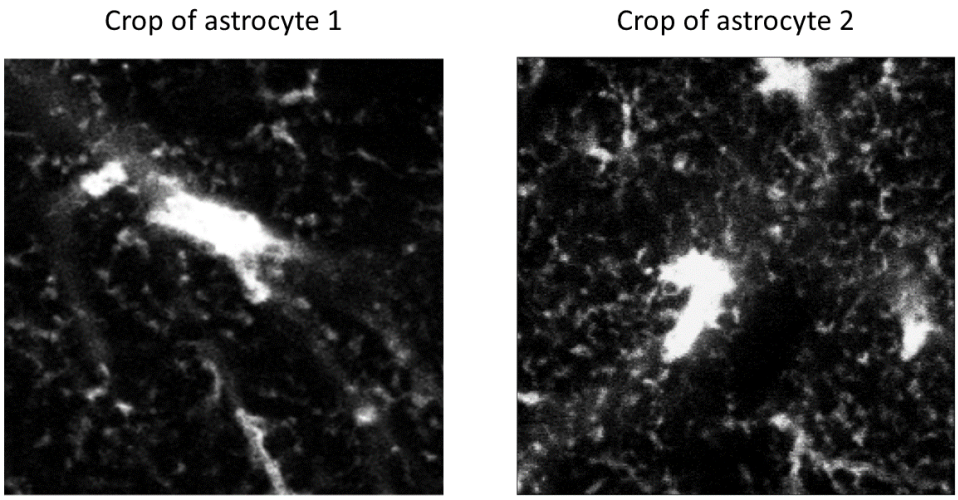

Figure 3.4: Central slices of the cropped images: crop of astrocyte 1 [134] and of astrocyte 2 [135].

The data were simulated with the data generator using these two cropped images as phantoms. Aberrations of 11 Zernike modes up to the fourth-order polynomials were introduced to the images in the simulation. The trivial modes (piston, x-tilt, y-tilt, and defocus) and aberrations of higher orders were excluded, similar to the PhaseNet research. The images for training and validation had combinations of all modes. The test image set consisted of 11 series of images for each aberration mode. In each of these series, only the selected mode was present, with the other modes set to zero to compare the prediction plots with those from the PhaseNet study.

3.4.2 Fluorescent beads

3D images of beads exhibiting fluorescence when exposed in the light with a wavelength of about 515 nm (the green spectrum) were acquired using a custom-designed light-sheet AO microscope with deformable mirrors constructed by Debayan Saha, Nicola Maghelli, and Sergei Klykov at the Max Planck Institute of Molecular Cell Biology and Genetics (MPI-CBG, unpublished work). Deformable mirrors were incorporated to allow introduction of monochromatic aberrations with controlled types and amplitudes. The dataset was intentionally limited to the Zernike polynomials up to the fourth order, as has been done in the PhaseNet study, because of a low impact of the higher-order polynomials on the image quality. The first-order aberrations and defocus were excluded because of their triviality: piston is a constant aberration, and x-tilt, y-tilt, and defocus are dispositions of an object along the x, y, and z axes, respectively.

The beads dataset consists of 177 images, each of which has dimensions of 128×128×60 pixels in the (x, y, z) coordinates. One image was acquired without the introduced aberrations.

The other 176 images contained single-mode aberrations: 16 amplitude values ranging from –0.15 to 0.15 μm with a step of 0.02 μm for each of the 11 Zernike modes: oblique astigmatism, vertical astigmatism, vertical trefoil, vertical coma, horizontal coma, oblique trefoil, oblique quadrafoil, oblique secondary astigmatism, primary spherical, vertical secondary astigmatism, and vertical quadrafoil.

Figure 3.5 shows an example of a slightly out-of-focus z plane near the middle of an aberrated image visualized in Fiji software [128]. The middle plane, which is in focus, is not shown because astigmatism is not observable exactly in the focus.

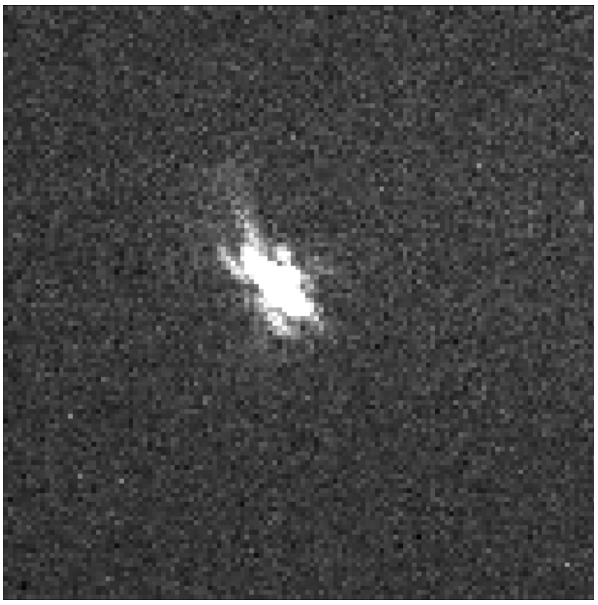

Figure 3.5: Example of a close-to-the-middle section in the z axis of an aberrated image: oblique astigmatism with an amplitude of 0.09 μm (Zernike mode 3 in the ANSI nomenclature).

3.4.3 *Drosophila* embryo (live sample)

In developmental biology, the fruit fly (*Drosophila melanogaster*) is a broadly used model organism that gained popularity because of its rapid reproduction and uncomplicated genetics with only four pairs of chromosomes [137]. Rapid development of fruit flies makes it particularly interesting to image them alive under a microscope.

A *Drosophila melanogaster* embryo, genetically modified to contain a fluorescent cell membrane marker membrane-mCherry [138], was imaged on the same microscope as the Fluorescent beads presented in Subsection 3.4.2. The embryo was at an early development stage (within 3 hours after the egg was laid) and kept at room temperature (24–25 ºC). The sample was mounted perpendicular to the light beam and to the camera, thus allowing imaging of the surface of the longest and flattest part of the body. The embryo was alive and continued to develop in the microscope's chamber filled with water.

The *Drosophila* dataset was collected for six Zernike polynomials of the second and third order, excluding the first-order aberrations and defocus because of their triviality, as described in Subsection 3.4.2. In the first step of this research, the dataset was limited to six aberration modes that affect the image quality the most (oblique astigmatism, vertical astigmatism, vertical trefoil, vertical coma, horizontal coma, and oblique trefoil) because acquisition of a

full dataset is time-consuming. A single embryo cannot survive the whole acquisition process because of exposure to the laser radiation, which is harmful for living organisms and leads to photobleaching of the fluorescent dye.

For each mode, 16 3D images were acquired, with amplitudes ranging from -0.15 to 0.15 μm with a step of 0.02 μm. The size of each image was 512×512×100 pixels in the (x, y, z) coordinates. In total, 97 images were obtained, including an image without the introduced aberrations.

Figure 3.6 shows the middle section of the full-size 3D image of a *Drosophila* embryo without introduced aberrations. The 192×192×100-pixel central crop of the image with a very small vertical trefoil aberration having an amplitude of 0.01 μm is shown in Figure 3.7. It is the 42nd image by the acquisition time, obtained after the first unaberrated image, 32 images with oblique astigmatism and vertical astigmatism, and 8 images of the stack for vertical trefoil with negative amplitudes. The quality of the images (the noisiness and sharpness of the membranes) decreased with time and the number of laser exposures. Both images were visualized in Fiji software [128] with an autocorrection of contrast.

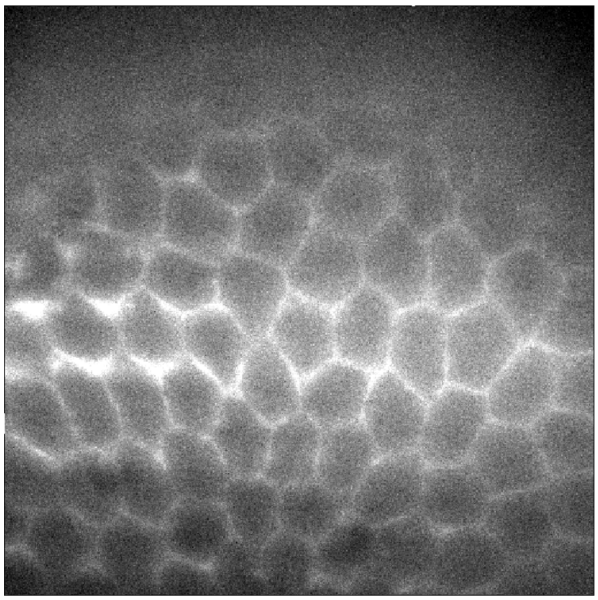

Figure 3.6: Unaberrated full-size *Drosophila* image, the middle section in the z axis.

3.4.4 Neurons (fixed sample)

Imaging of neuronal structures, especially in vivo, is essential to study the functions of the neural system and communication between the cells. Widefield fluorescence microscopy can be used for in vitro imaging of thin slices of the brain, but for in vivo studies, advanced modifications of this technique such as optical sectioning structured illumination microscopy are more suitable [139]. Neuronal structures are also interesting for studies of optical aberrations.

The sample preparation and data acquisition were done by Qinrong Zhang and Na Ji at the University of California in Berkeley. The widefield AO microscope is described in Figure S1b in ref [108], the sample preparation technique — in ref [139]. The samples were thin slices of the fixed mouse brain, where fixed means that the samples do not lose the ability to emit fluorescence after death. The slices were taken from the cortical tissue of a transgenic mouse: a

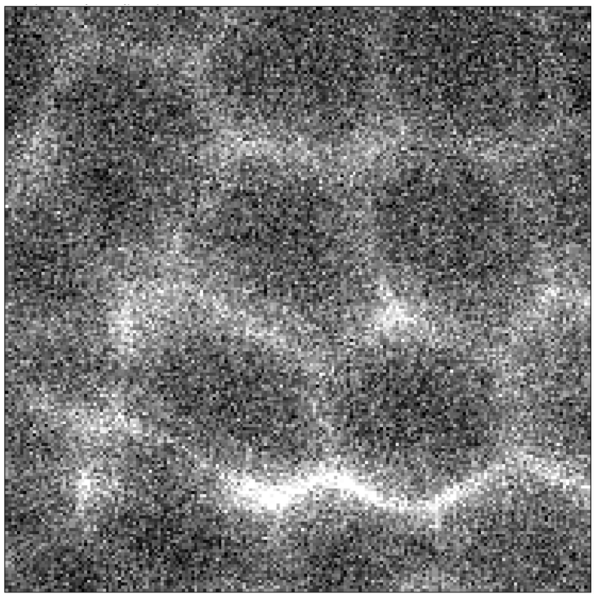

Figure 3.7: Example of the *Drosophila* data: a 192×192×100-pixel crop of the middle section in the z axis with a vertical trefoil aberration having an amplitude of 0.01 μm (Zernike mode 6 in the ANSI nomenclature).

Thy1–GFP (green fluorescent protein) line M mouse [140], as described in detail in the section "Fixed mouse brain slices preparation" in ref [139].

The images having a size of 1024×1024×201 pixels with four aberration modes (oblique astigmatism, horizontal coma, oblique trefoil, and primary spherical) were acquired. In the acquisition stack for oblique trefoil, the neuronal structure appeared to be very thin and observable only in a couple of consequent slices. Training of PhaseNet on this stack showed that the network was unable to recognize the aberrations in this data, therefore this stack was not used in the work presented in this thesis. Three other acquisition series (oblique astigmatism, horizontal coma, and primary spherical) were used further. The neurons were not clearly observed everywhere across the full-size images. As was done with the other datasets, the images were cropped so that the study subjects were visible in the crop.

The 96×96×96-pixel crops of an unaberrated image and of an image with oblique astigmatism having an amplitude of 0.1 μm (Zernike mode 3 in the ANSI nomenclature) are shown in Figures 3.8 and 3.9, respectively.

This dataset of the images of cortical neuronal structures is referred to as neurons in this dissertation.

3.5 Results

3.5.1 Astrocytes

The results presented in this section were published in my article "Estimation of Optical Aberrations in 3D Microscopic Bioimages" [109] coauthored with Eugene W. Myers, my PhD supervisor, in 2022 at the 7th International Conference on Frontiers of Signal Processing (ICFSP), IEEE. In the article, the plots with the results were shown for the horizontal coma aberration. In this thesis, the example plots are shown for aberration mode 5, vertical

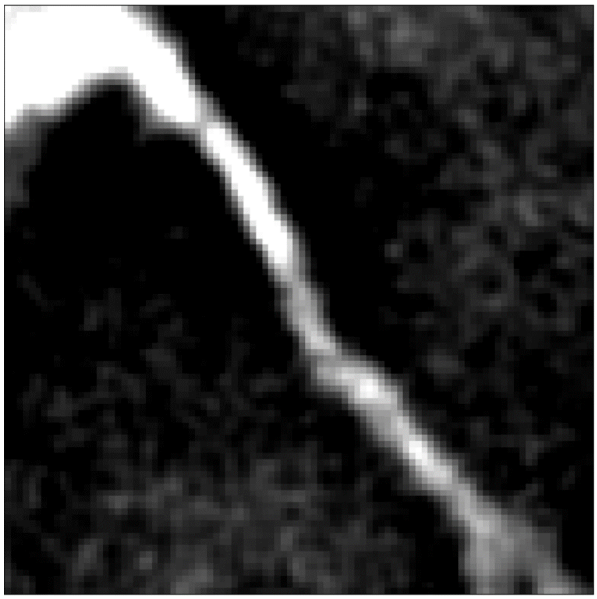

Figure 3.8: In-focus z section no. 60 of a cropped image (96×96×96 pixels) of a neuron from one series of acquisitions for oblique astigmatism. The image does not contain aberrations.

astigmatism. The result of restoration is shown in a different representation as a figure consisting of three images, including the generated PSF. The schematics are reproduced in a different color scheme.

Applying PhaseNet

PhaseNet was retrained using updated microscope parameters also needed for correct synthetic data generation. Due to the network's requirement for a specific pixel size (in microns), the pretrained weights from the initial PhaseNet publication were not applicable to the new data containing objects of an alternative scale. Furthermore, in accordance with the microscope configuration, we modified the noise parameters in the data generator (Section 3.3.3). The model underwent training using synthetic pointlike objects (representing PSFs) besides adjusting the pixel size, microscope parameters, and noise. No additional changes were made to the model or its architecture. The training methodology is illustrated in Figure 3.1.

Following this, aberrations were added to the astrocyte images through the process of cropping the original image and convolving the result with synthetic PSFs chosen at random for each crop (Section 3.3.4). The creation of such image series replicated the acquisition of the dataset analogous to the one from the PhaseNet publication, which included 11 test image series for every aberration mode. On this new dataset, we evaluated the predictions of retrained PhaseNet using the mean squared error (MSE) calculated as an average of all predictions. The model demonstrated an inability to forecast aberrations, as evidenced by the MSE of 0.001656 for the central crop in the initial image, 0.001718 for the crops randomly extracted from this image, and 0.002125 for the crops extracted from the second source image. One potential explanation is overfitting to the geometry of the training objects: the network acquired knowledge of the appearance of aberrated point sources but was unable to extrapolate this knowledge to nonspherical objects present in the test set. An alternative explanation could be that PhaseNet may be incapable of comprehending aberrations on complex geometries; this

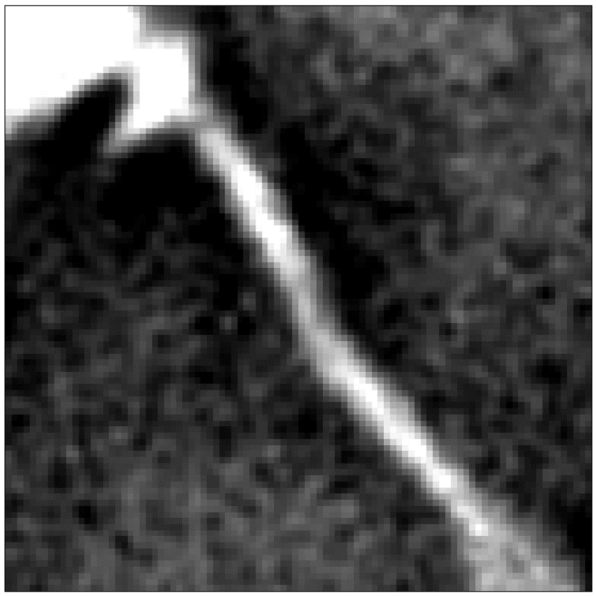

Figure 3.9: Z section no. 60 of a cropped image (96×96×96 pixels) of a neuron. The image has oblique astigmatism with an amplitude of 0.1 μm.

is examined in the subsection "Training with a phantom image."

Reading the prediction plots

For a better understanding of the results, they are visualized in a consistent manner across this chapter. The results of applying the retrained PhaseNet mentioned above to the test series with the aberration mode 5 in ANSI nomenclature, vertical astigmatism, are shown in Figure 3.10. The plot displays the intended **ground truth (GT)** predictions for the present aberration mode as **blue stars along the diagonal line**. The **predicted values** are represented by **yellow dots**. The **statistical measures** of predictions for all modes, namely the median, minimum, maximum, first and third quartiles, and outstanding data points, are presented in the **inset** located at the bottom right. Textbfpredictions for modes other than the experimentally introduced one are wished to have **minimal variation and be centered around 0 μm**.

Training with a phantom image

In order to examine whether PhaseNet is capable of learning aberrations on a nontrivial-shaped object, the model was trained using a phantom: an unaberrated image [134]. The generator was directed to crop out the center of the phantom; consequently, the simulated aberrations varied while the undelying object in the images remained constant. The findings demonstrated that the model is capable of learning how aberrations look like when applied to complex objects, such as an astrocyte part, provided that the training data contains pertinent information regarding the object (Figure 3.11).

To ascertain whether the model overfitted to the knowledge about the representation of the aberrations applied to one particular object (from the phantom), random cropping technique (jitter = "true", refer to Section 3.3.2) was applied to the first [134] and the second images

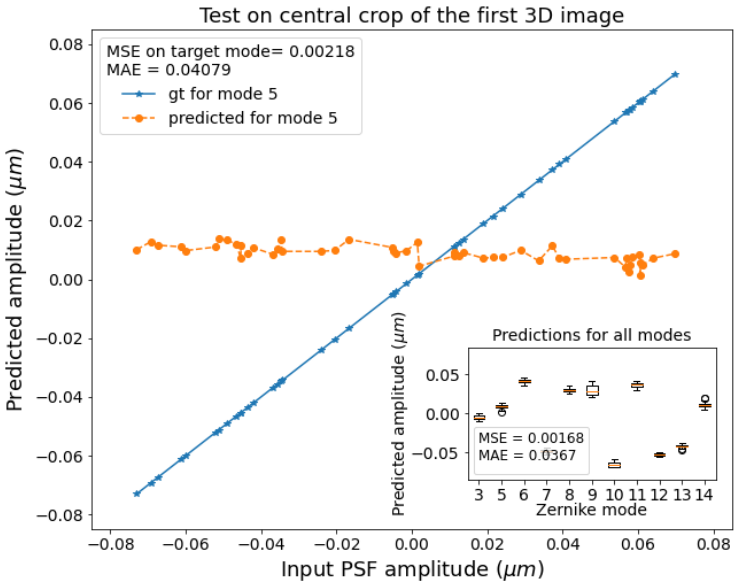

Figure 3.10: Results of PhaseNet trained on points and tested on astrocytes. The GT for the present aberration mode (mode 5, vertical astigmatism) is plotted with the blue stars on the diagonal line, the predicted amplitudes - by yellow dots. The statistics for all modes are presented in the inset. MSE is the mean squared error, MAE is the mean absolute error.

[135], which the model did not encounter during training. Performance dropped substantially from MSE = 0.000027 on the central crop to 0.000299 on random crops and 0.002186 on unseen data, demonstrating that if the geometry of the object does not change during training, the model can severely overfit to it.

Training with a phantom and random cropping

Here, the ability of the model to acquire an understanding of aberrations in the presence of object shape variation was investigated. The training dataset was enlarged by random cropping (jitter = “true”, see Section 3.3.2). The results of this experiment demonstrated (see Figure 3.12) that the model can effectively handle aberrations on objects of different shapes when it was exposed to them during training (see the "Random crop" entry in Table 3.4). Nevertheless, the accuracy of the predictions was marginally reduced compared to the results obtained from training on a stationary object shape (specifically, the central crop from the prior experiment).

Training with multiple phantoms

Two phantoms were provided to the generator: one of which was utilized in prior experiments, and the other of which was cropped from the second image. During each training iteration, the generator picked a phantom at random from the provided list of images, which was also randomly cropped. The mean square error (MSE) obtained from testing on random crops of the first and second images with random aberrations was nearly identical: 0.000129 on the first set and 0.000131 on the second. Upon observing the objects in the second dataset

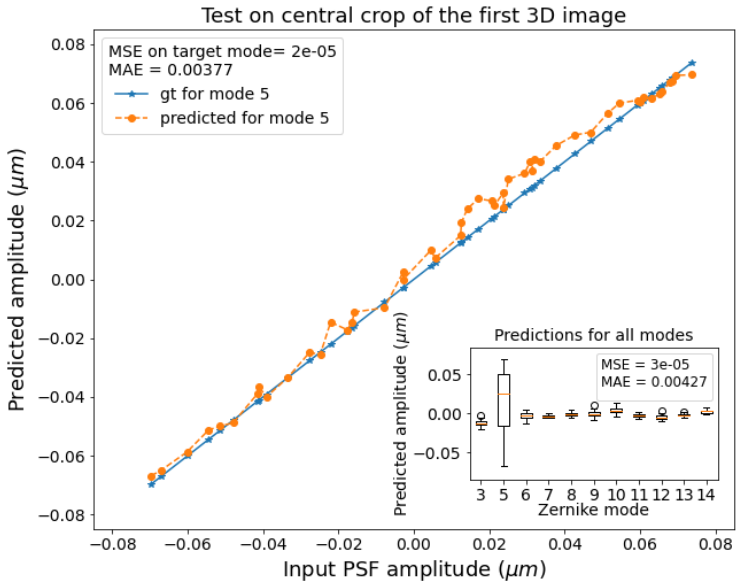

Figure 3.11: Results of training on the phantom with central cropping and testing on the central crop convolved with random PSFs. The GT for the present aberration mode (mode 5, vertical astigmatism) is plotted with the blue stars on the diagonal line, the predicted amplitudes - by yellow dots. The statistics for all modes are presented in the inset. MSE is the mean squared error, MAE is the mean absolute error.

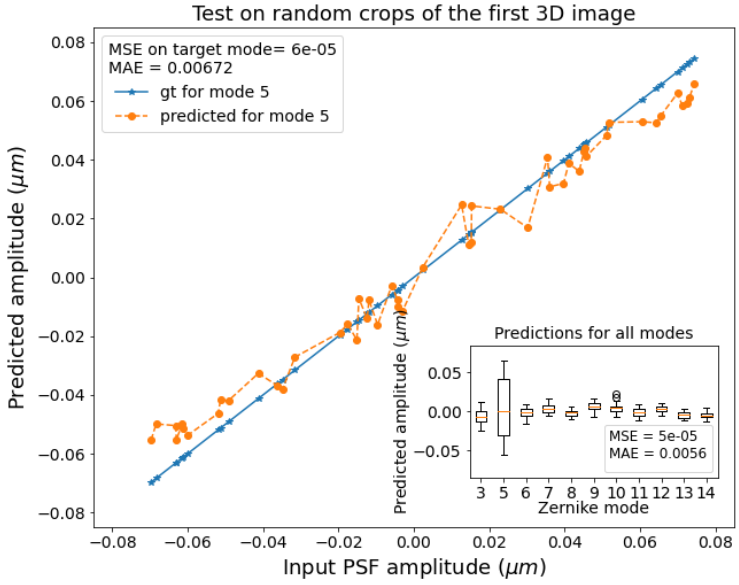

Figure 3.12: Results of training on random crops of the phantom and testing on random crops. The GT for the present aberration mode (mode 5, vertical astigmatism) is plotted with the blue stars on the diagonal line, the predicted amplitudes - by yellow dots. The statistics for all modes are presented in the inset. MSE is the mean squared error, MAE is the mean absolute error.

during training, it was not surprising that this model outperformed the preceding CNNs on this dataset.

In order to provide additional evidence that the model overfits to the morphologies of the objects in the training set, the model was evaluated on a distinct region of the second image

that was not intersecting with the training and validation crops. A substantial increase in the MSE to 0.001554 was observed. In spite of this, training on a greater variety of data resulted in a reduced MSE in comparison to prior experiments.

Furthermore, training was attempted on the central region of the first image, validation was performed on the central region of the second image, and testing was conducted on another region of the second image. The observation of overfitting was noticed after 400 epochs, which provides the support of the assumption that the network overfits on the seen object geometries.

Table 3.4 summarizes the results of all experiments. Training of the models lasted 50,000 epochs, with five steps per epoch and the batch size of two images. Training of each model was carried out on a computer with an Intel Core i7-8850H CPU, single NVIDIA Quadro P3200 GPU with 8 GB memory, and 64 GB RAM taking 30 hours.

Table 3.4: Results of PhaseNet on astrocytes

Training on	Mean MSE of testing on:		
	Central crop from image 1	Random crops from image 1	Random crops from unseen area of image 2
Points (III-A)	0.001656**	0.001718**	0.002125
Central crop (III-B)	0.000027	0.000299	0.002186
Random crop (III-C)	0.000040	0.000045	0.001793
Two phantoms* (III-D)	0.000090	0.000129	0.001554

*Training with multiple phantoms experiment, two were supplied.
**All crops were unseen, the model was trained on points.

Restoration

The workflow includes restoration of aberrated 3D images (Figure 3.2). A three-dimensional crop of an aberrated image is provided to the model, which uses the cropped image to predict the amplitudes of the aberration modes. The PSF is constructed utilizing predictions. Then, the input 3D image of the original size is deconvolved with the PSF.

Enhancing the quality of the source image can be achieved through deconvolution using the predicted PSF, when the model is trained using the identical image as a phantom. To accomplish this, the unprocessed image must be cropped to the dimensions supported by the network. The network then outputs the vector of amplitudes for the residual PSF for the cropped image. The PSF is computed according to the predicted amplitudes and employed as a deconvolutional kernel.

The raw image [134], the restored version of it, and the predicted PSF are shown in Figure 3.13.

3.5.2 Conclusions on the results for astrocytes

An unaberrated image (called phantom) is important to allow PhaseNet to perform well on nontrivial objects such as astrocytes. PhaseNet trained on simulated aberrations on a point light source cannot be directly applicable to bioimages of a specimen with a nonspherical

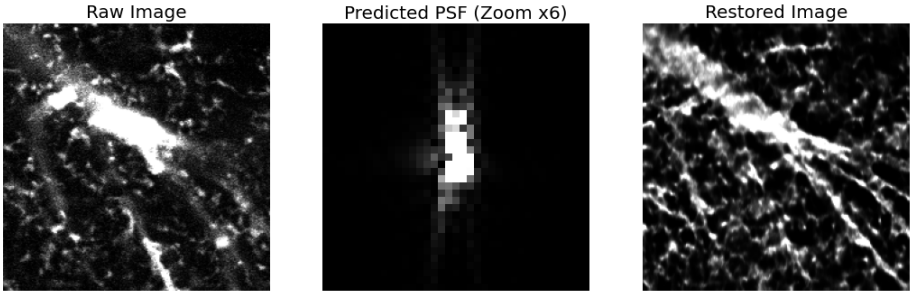

Figure 3.13: Restoration of a raw 3D image via deconvolution with the predicted 3D PSF. The central 2D planes are shown.

shape. The generator should be supplied with an unaberrated image of the specimen to include information about the shape of the object into the synthesized images, which allows usage of PhaseNet to estimate aberrations on images of nontrivial and nonspherical biological samples.

The results suggest that PhaseNet trained on synthetically aberrated images with one crop of an astrocyte as a phantom overfits on the seen object. For datasets with a large variety of object morphologies, it is recommended to include maximum of the available information about the shape variety, which can be done by providing the generator with multiple phantom images that represent different shapes. The problem of generalization should be addressed in future works.

This study indicates that PhaseNet predictions can be used for image restoration. The suggested modifications for the training and testing procedures also allow restoration of source images without additional synthetic aberrations. The model has to be retrained on new data with a phantom and the microscope's parameters (voxel size, refractive index, numerical aperture, and light wavelength) for each new dataset.

3.5.3 Fluorescent beads

The benefits of training with an unaberrated image as a phantom also for trivial objects and of convolving the experimental test data with the residual synthetic PSF are presented in this subsection. The problem of extendability of PhaseNet to amplitudes larger than 0.1 μm is also addressed here.

The network expects the pixel size and the parameters of the microscope as an input, therefore the pretrained networks from the original PhaseNet experiments cannot be reused for beads of a different size imaged on another microscope. In the training configuration, the noise parameters in the data generator were also changed according to the microscope system: they were derived using Fiji software directly from the image that contained only the residual PSF but no experimentally introduced PSF. Different preprocessing techniques and parameters for the data generator were explored. Comparisons of the models trained for 1000 epochs with the same CNN architecture, train batch size, steps per epoch, learning rate, train validation split, and optimizer are shown in Figures 3.14, 3.15, 3.16, and 3.17. Increasing the number of training epochs incrementally reduces the error. The models for the parameter search were

trained for only 1000 epochs because the training is time- and resource-consuming.

Training with a phantom, deconvolution, "zero" convolution

Phantom. To make the synthetically generated training data more realistic, the convolution with synthetic PSFs was performed on a bead image (phantom) instead of a point. The approach is described in Section 3.3.4. Generating images using an actual image of the bead should add prior information about the object structure, the microscope noise that cannot be approximated by Gaussian noise (which can be added by the data generator when the training and validation images are formed), and about the residual PSF. It will be investigated in this subsection whether adding prior information about the object structure leads to performance improvement on bead images.

If a phantom is used, the data generator outputs a product of the convolution of the phantom with a synthetic PSF. As a result, the generated image was "convolved" twice: with the microscope's PSF from the phantom (during the acquisition of the phantom) and with the synthetic PSF. Afterward, additional synthetic Gaussian noise was added. The test dataset of the experimental images was "convolved" once during the acquisition, after which the real noise was "added" by the camera. Hypothetically, performing convolution twice adds too much blur.

To bridge this gap between the training and test data, two approaches were introduced:

- (a) **Gaussian deconvolution:** removing the blur of one convolution from the phantom image via deconvolution.
- (b) **"Zero" convolution:** adding the synthetic PSF to the test data as described in Section 3.3.6.

In approach (a), the deconvolution was performed using the Richardson–Lucy iterative algorithm [130, 131] in DeconvolutionLab2 plugin [132] within the Fiji software [128], assuming that the images contain standard blur, which can be approximated by the Gaussian function [141, 142]. The results shown below were obtained with the following parameters of the Gaussian PSF: PSF shape of 64×64×64 pixels, sigma of 5×5×5 pixels, and mean intensity of 102. After varying the parameters, this set was chosen according to visual observations of the deconvolution result.

Multiple models were trained for 1000 epochs using the combinations of the preprocessing approaches (a) and (b) and training without a phantom (the approach from the original PhaseNet study). In Figures 3.14 and 3.15, a comparison of six approaches is shown:

(1) Training with the phantom deconvolved using the Gaussian approach and testing on the raw test data (red stars in the left column);

(2) Training with the phantom deconvolved using the Gaussian approach and testing on the test data convolved with the synthetic "zero" PSF (blue dots in the left column);

(3) Training with the raw phantom and testing on the raw test data (red stars in the middle column);

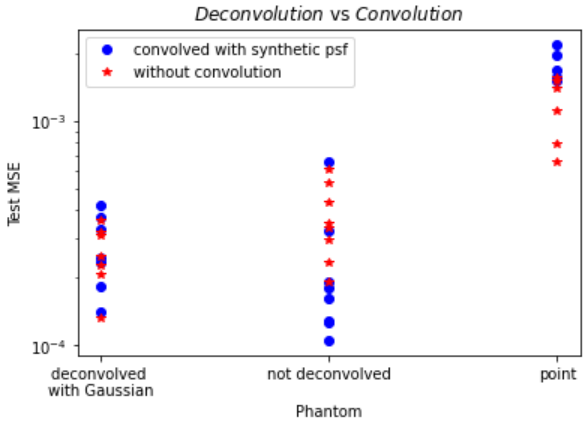

Figure 3.14: Comparison of the training–testing pipelines of the models trained for 1000 epochs on six Zernike modes (oblique astigmatism, vertical astigmatism, vertical trefoil, vertical coma, horizontal coma, and oblique trefoil). Blue dots indicate the predictions after convolving the test images with the synthetic "zero" PSF, red stars — the predictions on the raw test data.

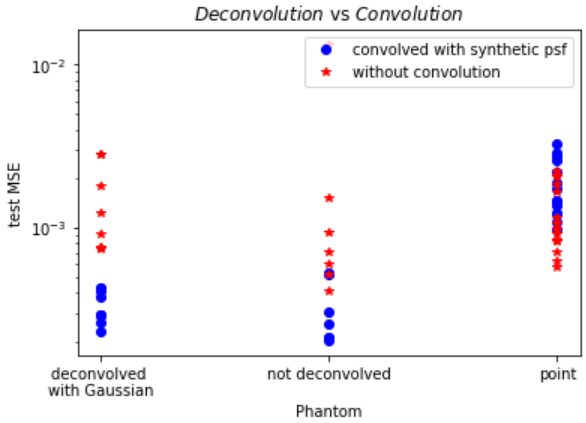

Figure 3.15: Comparison of the training–testing pipelines of the models trained for 1000 epochs on 11 Zernike modes (from oblique astigmatism to vertical quadrafoil). Blue dots indicate the predictions after convolving the test images with the synthetic "zero" PSF, red stars — the predictions on the raw test data.

(4) Training with the raw phantom and testing on the test data convolved with the synthetic "zero" PSF (blue dots in the middle column);

(5) Training without a phantom (original PhaseNet) and testing on the raw test data (red stars in the right column);

(6) Training without a phantom (original PhaseNet) and testing on the test data convolved with the synthetic "zero" PSF (blue dots in the right column).

Blue dots in Figures 3.14 and 3.15 correspond to the "zero" convolution preprocessing of the test images (approach (b)), red stars — to testing on the raw images. The left column corresponds to the Gaussian deconvolution (approach (a)).

The results demonstrate that:

1. Introducing the information (phantom) about the object and microscope to the generator significantly reduces the MSE (by a factor of 10 in the case of training on six modes, Figure 3.14);

2. Passing an unaltered raw unaberrated image as the phantom to the synthetic data generator during training and subsequently convolving the test images using the synthetic PSF during testing gives the best results. The performance gain with the suggested approach (b) is the best if training on 11 modes (Figure 3.15).

Random cropping, training range of amplitudes

Next, it was investigated whether random cropping of an image (jitter = "true" in 3.3.2) is better than central cropping. The PhaseNet generator takes a large image as an input, convolves it using a synthetic PSF, and crops to a smaller image that the CNN takes as an input. When the jitter parameter of the generator is set to true, cropping is shifted from the center of the image. When the jitter is set to false, the crop is taken from the center. For these experiments, the networks were trained with the raw image as a phantom, and the test images of the full range (± 0.15 μm) were convolved with the "zero" PSF, as suggested by the previous results.

The results show that off-center cropping leads to a performance gain for the beads dataset (Figures 3.16 and 3.17).

The original PhaseNet has been proven to work on images of beads when trained on the synthetically aberrated images with the amplitude range from -0.075 to 0.075 μm [108]. In the dataset considered here, the amplitude range of the acquired test images is twice as large, and it was necessary to determine the range within which PhaseNet is applicable. The models were trained on different amplitude ranges and the resulting MSEs were compared. The calculations showed that training on ranges smaller than the full range present in the test data (-0.15 to 0.15 μm) yields a significant improvement.

Among the models trained on six Zernike modes, one trained on the range from -0.1 to 0.1 μm significantly outperformed the others trained on ± 0.075 μm and on ± 0.15 μm (Figure 3.16). For the models trained on 11 Zernike modes, using the smallest studied range

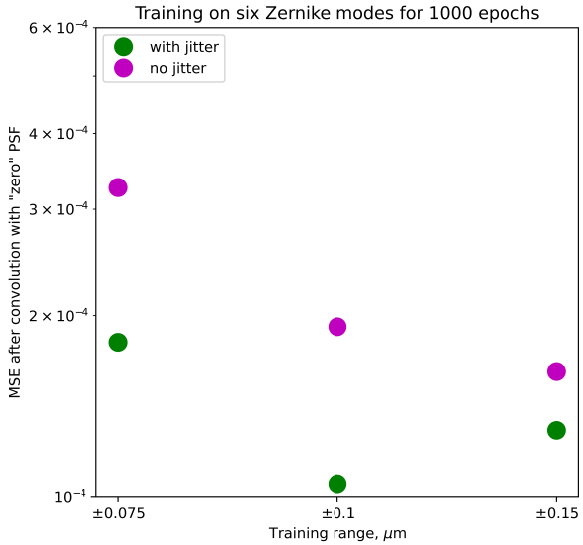

Figure 3.16: Comparison of the amplitude range and jitter training parameters of the models trained for 1000 epochs on six Zernike modes and tested on the images of the full range of ±0.15 μm with the "zero" convolution applied. Green dots indicate the predictions of the models trained with jitter = true (off-center cropping), magenta dots — those of the models trained with jitter = false (center cropping).

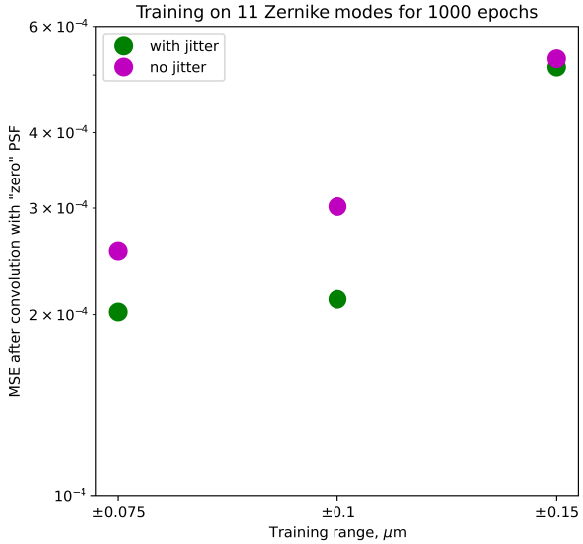

Figure 3.17: Comparison of the amplitude range and jitter training parameters of the models trained for 1000 epochs on 11 Zernike modes and tested on the images of the full range of ±0.15 μm with the "zero" convolution applied. Green dots indicate the predictions of the models trained with jitter = true (off-center cropping), magenta dots — those of the models trained with jitter = false (center cropping).

(−0.075 to 0.075 μm) leads to only a slight improvement compared to training on the range from −0.1 to 0.1 μm (Figure 3.17). Training on the range from –0.1 to 0.1 μm should also be suitable. This suggests a hypothesis that the generated synthetic data with the aberrations of large amplitudes are not sufficiently realistic. The reason for this could be a decrease in the intensity of the object (bead) on the experimental images when the amplitudes of aberrations are large.

The best model among those trained on 11 Zernike modes for 1000 epochs achieved the MSE = 0.000202 on the test data (full range from −0.15 to 0.15 μm) after the "zero" convolution. The MSEs of testing on other ranges with and without the "zero" convolution are presented in Table 3.5.

Table 3.5: Results of the best model trained on Fluorescent beads on 11 Zernike modes for 1000 epochs

	MSE on test data range			MSE on synthetic data
Postprocessing	±0.075 μm	±0.1 μm	±0.15 μm	±0.075 μm
Raw test data	0.000488	0.000527	0.000716	0.000113
"Zero" convolution	0.000082	0.000106	0.000202	

The best parameters of the generator and the model trained on 11 Zernike modes for 1000 epochs are:

- phantom: raw unaberrated image (not deconvolved);
- jitter: true;
- amplitude range: −0.075 to 0.075;
- noise mean: [70.0, 130.0]; noise sigma: [5.0, 10.0]; SNR: [20.0, 40.0]; Gaussian blur sigma [0.5, 1.0];
- numerical aperture: 0.9; wavelength: 0.515 μm; immersion refractive index: 1.33; PSF pixel size: [0.2, 0.113, 0.113];
- kernel size: [3, 3, 3]; pooling size: [1, 2, 2]; batch size: 8; crop shape: [50, 50, 50]; steps per epoch: 5;
- loss: MSE; optimizer: Adam (standard Keras parameters); learning rate: 0.0003.

In the next subsection, the dataset was used with only six Zernike modes, on which additional models were trained. Among the models trained for 1000 epochs on these six modes, the best performing model had the same parameters as the best model trained on 11 modes, with two exceptions:

- amplitude range: −0.1 to 0.1;
- noise mean: 102; noise sigma: 5; SNR: 31; no Gaussian blur.

This model achieved the MSE = 0.000105 after 1000 epochs on the test data in the full range of amplitudes with "zero" convolution applied (Table 3.6).

Number of epochs

The same model (referenced in Subsection 3.5.5) was retrained for 10,000 epochs with:

- phantom: raw unaberrated image (not deconvolved);
- jitter: true;
- amplitude range: −0.1 to 0.1;
- noise mean: 102; noise sigma: 5; SNR: 31; no Gaussian blur;
- numerical aperture: 0.9; wavelength: 0.515 μm; immersion refractive index: 1.33; PSF pixel size: [0.2, 0.113, 0.113];

- kernel size: [3, 3, 3]; pooling size: [1, 2, 2]; batch size: 8; crop shape: [50, 50, 50]; steps per epoch: 5; epochs: 10,000;

- loss: MSE; optimizer: Adam (standard Keras parameters); learning rate: 0.0003.

The training resulted in the MSE = 0.000082 (Table 3.7), which is only 0.000023 smaller than that of the training for 1000 epochs, thus being of approximately the same order of magnitude. The predictions of the network trained for 10,000 epochs are shown in the plot of the predicted amplitude versus the amplitude of the input PSF (Figures 3.18 and 3.19). The predictions for mode 3 are about 10 times more accurate than those for mode 6. This behavior is discussed in the next chapter.

Table 3.6: Results of the best model trained on beads on six Zernike modes for 1000 epochs

	MSE on test data range			MSE on synthetic data
Postprocessing	± 0.075 μm	± 0.1 μm	± 0.15 μm	± 0.1 μm
Raw test data	0.000310	0.000311	0.000352	0.000038
"Zero" convolution	0.000038	0.000049	0.000105	

Table 3.7: Results of the best model trained on beads on six Zernike modes for 10,000 epochs

	MSE on test data range			MSE on synthetic data
Postprocessing	± 0.075 μm	± 0.1 μm	± 0.15 μm	± 0.1 μm
Raw test data	0.000049	0.000069	0.00015	0.000009
"Zero" convolution	0.000032	0.000041	0.000082	

The models performed best on the test data within the smallest range of amplitudes. This suggests that PhaseNet produces the most reliable results on the images with only small aberrations. If the goal is to use PhaseNet on the images with the six aberration modes with the amplitudes of ± 0.15 μm, the network should be trained on the amplitude range of ± 0.1 μm (Figure 3.16) where the lowest MSE was achieved. In the case of using PhaseNet on the 11 modes with the amplitudes of ± 0.15 μm, the training range should be set to ± 0.075 μm (Figure 3.17). Overall, the results on the 11 aberration modes with the amplitudes of ± 0.15 μm are worse than those on smaller test ranges and on fewer aberration modes.

3.5.4 Conclusions on the results for fluorescent beads

In the next two subsections, the following conclusions will be used:

1. Supplying the generator with a phantom improves the results compared to the original PhaseNet setup where the synthetic data were generated on spheres.

2. An unprocessed unaberrated image is a better choice for the phantom compared to an image preprocessed using deconvolution with the Gaussian PSF.

3. Applying the "zero" convolution to test images improves the results.

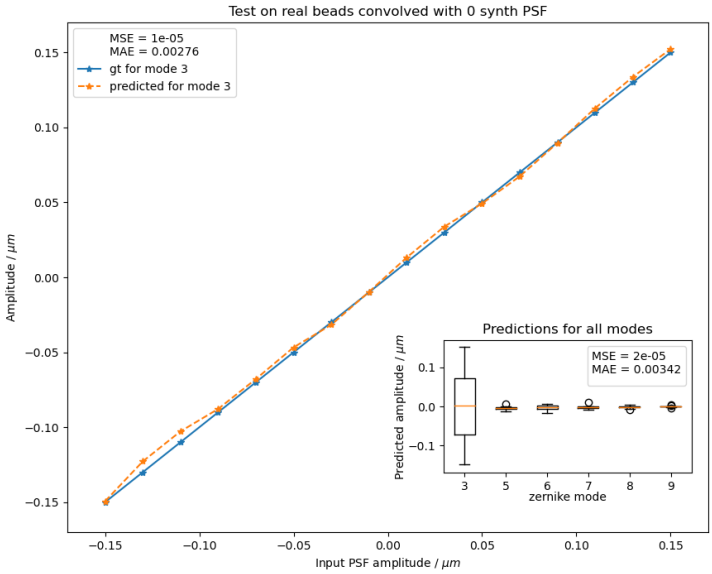

Figure 3.18: Results of the best model trained on beads on six Zernike modes for 10,000 epochs display an excellent performance on mode 3. The ground truth (GT), where the prediction is equal to the input PSF, is shown in blue.

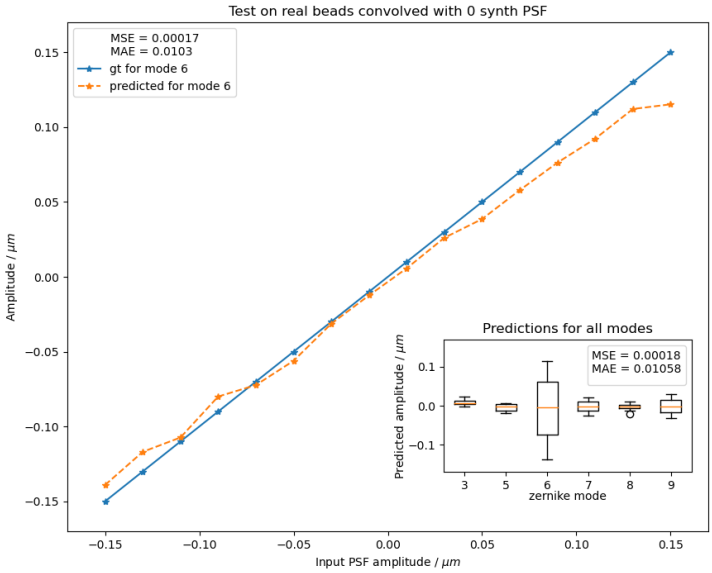

Figure 3.19: Results of the best model trained on beads on six Zernike modes for 10,000 epochs display a lower performance on mode 6. The ground truth (GT), where the prediction is equal to the input PSF, is shown in blue.

4. Random cropping of the phantom inside the data generator also helps the model learn better.

5. Training for 1000 epochs should yield sufficiently accurate results, increasing the number of epochs may lead to incremental improvements.

6. For the test set with 11 modes and the full amplitude range of ±0.15 μm, training on the smaller range of ±0.075 μm led to the lowest MSE, whereas using the training range of ±0.1 μm yielded very close results of the same order of magnitude.

7. For the test set with six modes and the full amplitude range of ±0.15 μm, the lowest MSE was achieved by training on the amplitude range of ±0.1 μm.

3.5.5 *Drosophila* embryo

Applying PhaseNet trained on beads to the *Drosophila* dataset

To start exploring applicability of PhaseNet to a new type of experimental data, it was first checked whether the network trained on beads can perform on a new object. The required parameters of the microscope system for this dataset are the same as those for the Fluorescent beads dataset.

The best model trained on beads on six Zernike modes for 10,000 epochs (see Subsection 3.5.3) was used in this experiment. The model was unable to predict accurately: the MSE achieved on the test *Drosophila* images with and without the "zero" convolution was 0.002913 and 0.002312, respectively. The predictions looked similar to Figure 3.10 and did not show any diagonal trend seen in Figures 3.18 and 3.19. These results lead to a conclusion that the network did not learn the generic representation of the aberration modes from the bead images. Instead, it likely learned how the aberrations look like on point light sources.

Training on the *Drosophila* dataset

At the next step, the same model was trained with the same parameters on a phantom that was the unaberrated *Drosophila* image.

First, it was checked if the CNN can predict the aberrations correctly on the synthetic data, which means that it could learn from the designed task. The calculations showed that the network can perform well on the synthetic data: the predictions are close to the GT line (Figure 3.20).

The network was then tested on each subset with the images in which only one aberration mode was present. On the test set with the "zero" convolution, a very large error (MSE = 0.004784) was observed. On mode 3, the network showed an approximate, although rather poor, diagonal trend (Figure 3.21), failing on all other modes (Figure 3.22) despite performing well on the synthetic data (Figure 3.20) with MSE = 0.000026.

New noise parameters for the data generator were retrieved from the *Drosophila* images, considering the membranes as the foreground and the cells as the background. The parameters were set to the following values: noise mean = 800.0, noise sigma = 36.0, and SNR = 1.27. The resulting MSE was 0.008580, MSE after the "zero" convolution was 0.005027, and MSE on the synthetic data was 0.000185. The errors were higher than in the previous experiment with the noise mean = 102, sigma = 5, and SNR = 31, but the MSEs after the "zero" convolution were of the same order of magnitude.

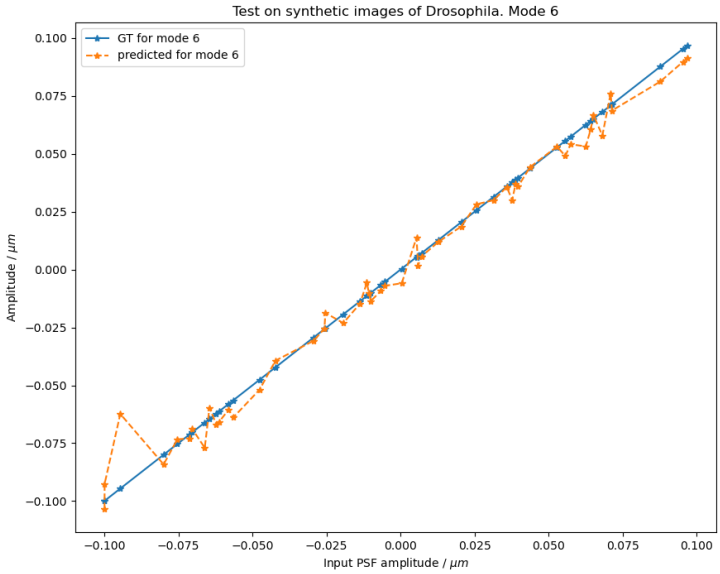

Figure 3.20: Predictions of the model trained on six Zernike modes and applied to synthetic single-mode (mode 6) *Drosophila* images. The diagonal trend indicates the model's ability to learn the representations of aberrations on this nontrivial object.

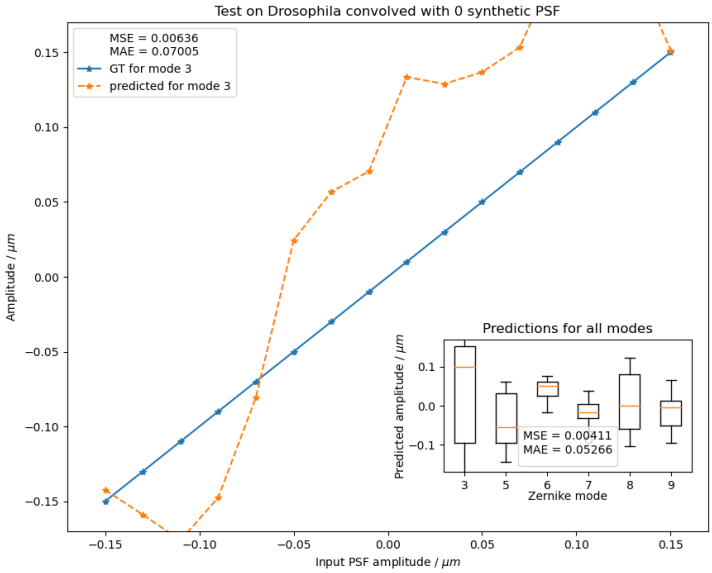

Figure 3.21: Predictions of the model trained on six Zernike modes and applied to mode 3 of the *Drosophila* dataset show a weak approximate trend.

Training on the full set of aberration modes did not work out of the box. To study the underlying mechanism, the problem was simplified.

Single-mode training

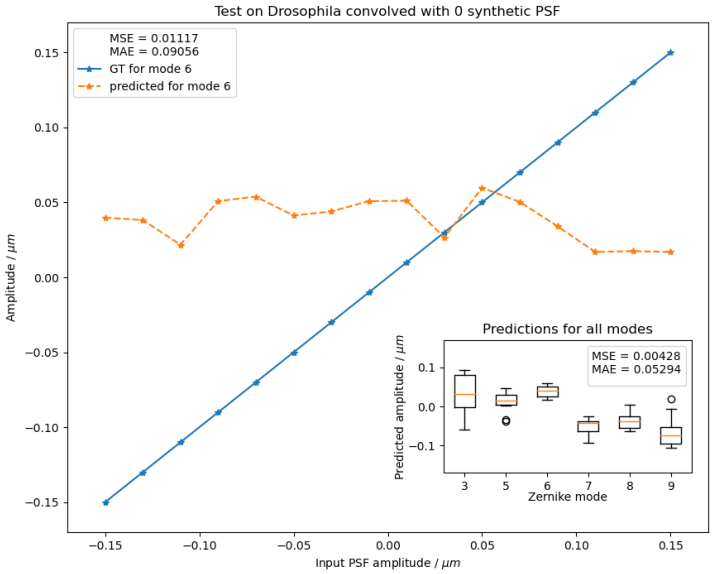

Figure 3.22: Predictions of the model trained on six Zernike modes and applied to mode 6 of the *Drosophila* dataset show no diagonal trend.

To reduce the complexity of the problem, multiple models were trained on a single aberration mode separately. This resulted in an improvement in the performance on mode 3 with an approximately diagonal trend (Figure 3.23). On other modes, the networks did not perform significantly better than in the previous experiment (Figure 3.24) despite the low MSE on the synthetic test data (e.g., MSE = 0.000084 for mode 6).

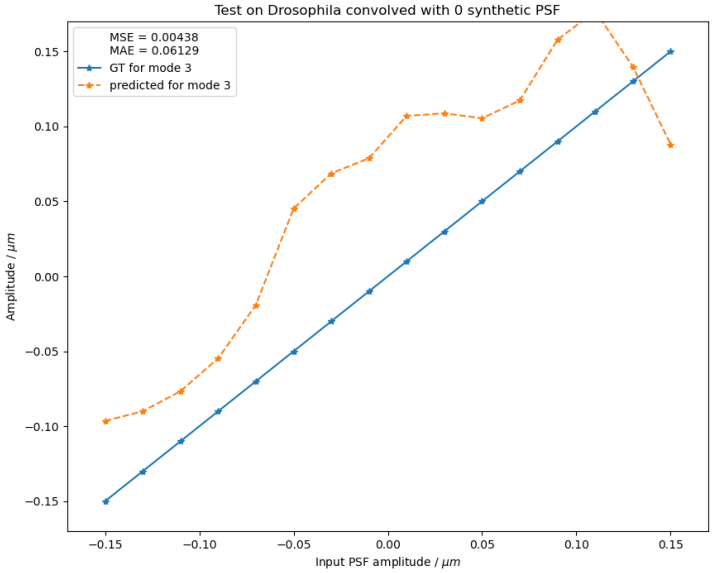

Figure 3.23: Predictions on the *Drosophila* dataset of the model trained only on mode 3. A diagonal trend can be observed, although the predictions are relatively far from the GT.

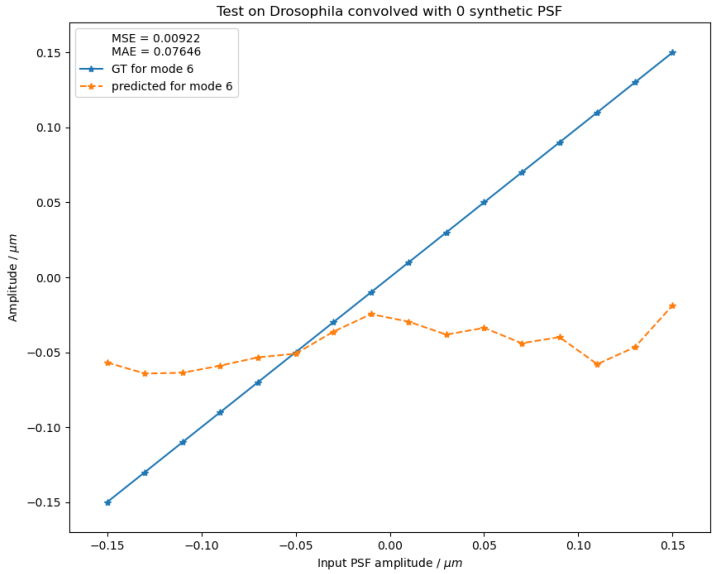

Figure 3.24: Predictions on the *Drosophila* dataset of the model trained only on mode 6 show no diagonal trend. The CNN training on the single mode was unsuccessful.

Training according to the acquisition time

The reasonable performance on mode 3 can be potentially explained by the time of acquisition. The unaberrated image was acquired first. Then, the stack for mode 3 was imaged in the following amplitude order: –0.15, –0.13, . . . , –0.01, 0.01, . . . , 0.15 μm. Afterwards, the stack for mode 5 was acquired in the same order. By this time, the sample may have changed because of the loss of the image quality (e.g., because of photobleaching due to phototoxicity), development of the embryo (the cellular shape can change pretty quickly, which is best observed during the gastrulation), or movement of the embryo that led to rotation of the membranes that were in the field of view or to displacement out of focus. Observations show that the most probable reasons are the loss of the image quality or the live embryo's movements. The quality of the image is lower, the image is noisier, and the contrast of the membranes with the background (cells) is lower in Figure 3.7 compared with the image in Figure 3.6. Thus, the net was trained on one embryo but tested on "another": displaced, photobleached, or maybe even more developed one.

To validate these assumptions, the image with mode 5 having an amplitude of 0.01 μm (almost negligible and invisible to the human eye) was passed as a phantom to the generator and the model was trained to predict the amplitudes on this mode (single-mode training). The same procedure was repeated for mode 6 (Figure 3.7). The results prove that PhaseNet can predict the amplitudes if the structures in the object on the phantom image do not differ significantly from those of the test objects (Figure 3.25). Convolving the test data with the "zero" synthetic PSF improved the predictions (Figure 3.26), the same behavior was observed for the predictions after the single-mode training on mode 5.

The parameters of the best model trained on mode 6:

- amplitude range: ± 0.1 μm;

- phantom: image from the target acquisition stack with $a_6 = 0.01$ μm;
- activation functions (except for the last layer with the linear activation): tanh;
- input shape (crop shape): [50, 50, 50];
- training loss function: MSE;
- epochs: 1000;
- steps per epoch: 5;
- learning rate: 0.0003;
- batch size: 8.

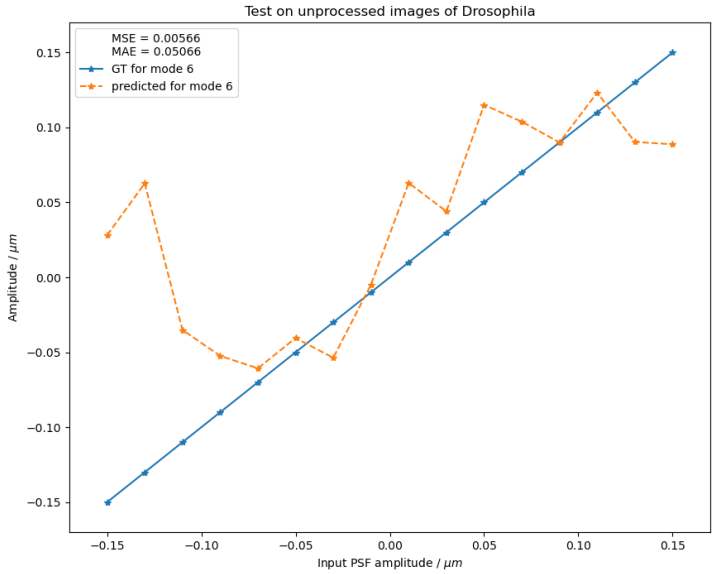

Figure 3.25: Results of training on a phantom (a *Drosophila* image with mode 6 having an amplitude of 0.01 μm) show a good performance in the middle of the acquisition stack.

The quality of predictions drops drastically on the amplitudes with large absolute values. Can this also be explained by the time of acquisition and subsequent changes in the sample or by inability of PhaseNet to perform well on aberrations with large amplitudes? This question is addressed in Subsection 3.5.7 with the usage of fixed biosamples.

3.5.6 Conclusions on the results for *Drosophila* embryo

Providing the generator with an unaberrated image that represents the information about the object and the optical system allows extendability to nontrivial samples. The results show that training on the range of synthetic amplitudes from -0.1 to 0.1 μm that is smaller than the full range present in the test set (from -0.15 to 0.15 μm) increases the quality of predictions for these data. The underlying reasons require further investigation.

The method might not be directly transferable to moving and developing organisms. In the next subsection, PhaseNet is applied to fixed (stained and dead) samples to eliminate the

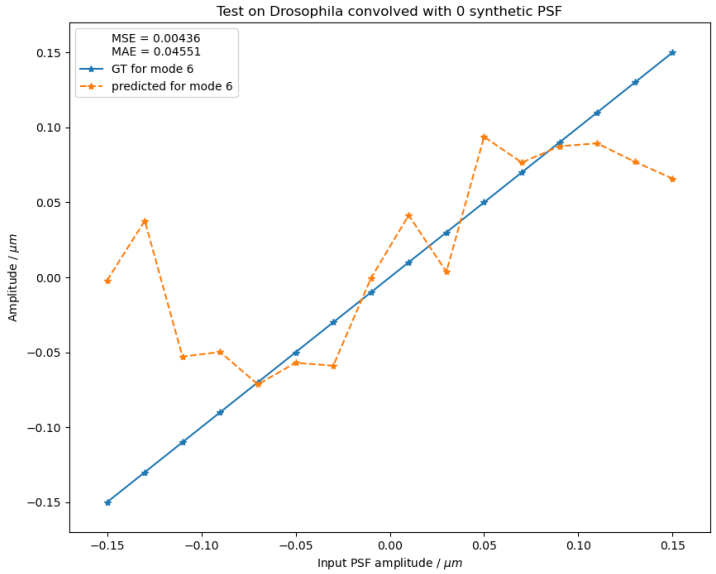

Figure 3.26: Applying the "zero" convolution to the test data after training on a phantom (a *Drosophila* image with mode 6 having an amplitude of 0.01 μm) shows an improved performance.

problems of their developmental changes and movements. In addition, different samples are used to prevent severe photobleaching caused by the phototoxicity of the laser light. If a single sample is used for the complete acquisition, the fluorescent label loses its brightness. After a certain cumulative time of laser exposure, fluorescent markers reach their limit of allowed exposures, the so-called photon budget, and degrade.

3.5.7 Neurons

This section presents the final results achieved building on the study of astrocytes, fluorescent beads, and a *Drosophila* embryo described in previous subsections. The work on this data started in January 2021 and included bleaching correction with histogram matching and exponential fit methods, deblurring via deconvolution with a Gaussian kernel, percentile normalization, and testing various values for PhaseNet parameters. The results of early, rather unsuccessful, attempts are not included in this thesis; only the final solution and the path to it are presented.

To achieve these results, the following **information from the previously described experiments** was used:

1. The generator must be supplied with phantom(s) representing the shape(s) of the study samples.

2. Supplying the generator with multiple phantoms improves generalizability during training and allows PhaseNet to work on all test acquisition sets.

3. Random cropping ("jittering" around the center) of the phantoms inside the generator improves the generalizability of PhaseNet and prevents early overfitting.

4. The phantoms must be larger than the network's input shape to allow random cropping inside the generator and to "cut off" the boundary effects of convolution with the PSF.
5. Training PhaseNet on synthetically aberrated images with the amplitude range up to $[-0.1, 0.1]$ μm gives reliable results on the test data within this range and may give acceptable results in the intervals from -0.15 to -0.1 μm and from 0.1 to 0.15 μm. Training on the amplitude range from -0.15 to 0.15 μm leads to a significant drop in performance.
6. 1000 training epochs are sufficient for achieving reasonable performance on the test data.
7. Increasing the training batch size to eight images led to improvements.
8. Convolution of the test images with the "zero" synthetic PSF, which makes the test data look similar to the training data, significantly improves the results.

Using these recommendations, two networks with the same parameters were trained:

1. PhaseNet was trained on two modes — mode 3 (oblique astigmatism) and mode 8 (horizontal coma) — with two phantoms: one unaberrated image from the acquisition series for mode 3 and the other — from the series for mode 8.
2. PhaseNet was trained on six modes — mode 3 (oblique astigmatism), mode 5 (vertical astigmatism), mode 6 (vertical trefoil), mode 7 (vertical coma), mode 8 (horizontal coma), and mode 12 (primary spherical) — with three unaberrated images passed as phantoms from the corresponding series for modes 3, 8, and 12.

The following **network parameters**, which led to the best results on the *Drosophila* dataset, were used to train these networks:

- training amplitude range: ± 0.1 μm;
- phantom: unaberrated images from the target acquisition series with $a_i = 0$ μm;
- activation functions (except for the last layer with the linear activation): tanh;
- input shape (crop shape): [50, 50, 50];
- training loss function: MSE;
- epochs: 1000;
- steps per epoch: 5;
- learning rate: 0.0003;
- batch size: 8.

Training on two modes with two phantoms

The predictions were made for the images with experimentally introduced oblique astigmatism (mode 3) and horizontal coma (mode 8) aberrations, acquired on the AO microscope (Figure 3.27). The raw images did not contain any other aberrations except for the inevitable residual PSF of the microscope, which is present in all images acquired on this microscope. The raw images were convolved with the "zero" synthetic PSF as described in Subsection 3.3.6.

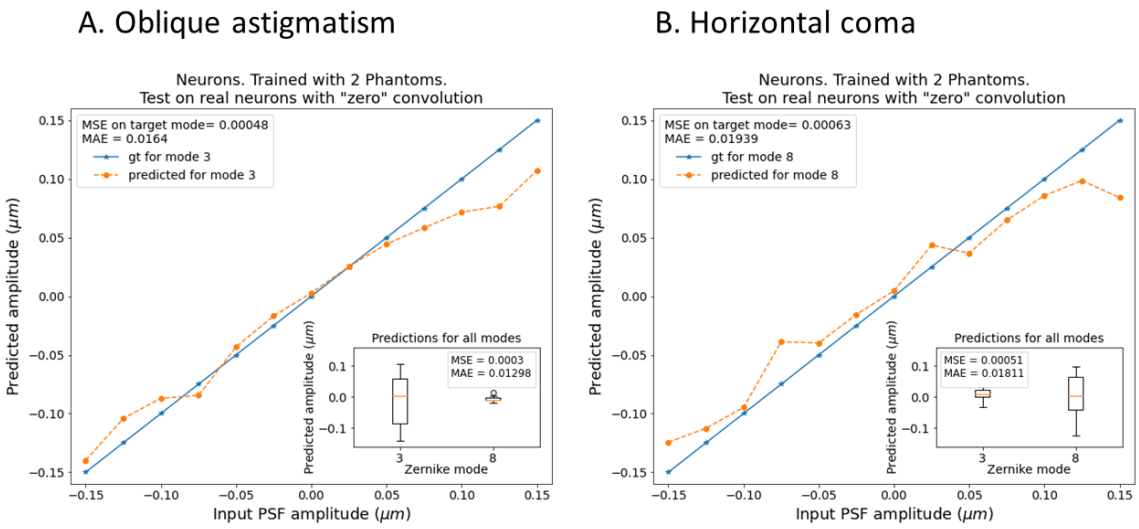

Figure 3.27: Predictions for the images of neurons with experimentally introduced oblique astigmatism (mode 3) and horizontal coma (mode 8) aberrations, acquired on the AO microscope. The model was trained on modes 3 and 8 with two phantoms (neurons) and tested on the real data with the "zero" convolution.

The results demonstrate that:

1. PhaseNet was able to pick up a relatively unsophisticated task of predicting the amplitudes for two Zernike modes on the images of a nontrivial object.
2. It worked on the fixed sample with observable difference in intensity between the objects and the background.
3. The "zero" convolution method helped to reach a reasonably low error.
4. The predictions for amplitudes with large absolute values (0.125 and 0.15 μm) may not be as reliable as for smaller amplitudes.
5. PhaseNet correctly predicted the amplitude values around 0 μm for the absent aberration mode on the images with introduced modes 3 and 8.

The network was tested on a series of images with varying amplitudes of mode 12. For these images, the network was expected to predict $a_{3,8} = 0$; the real predicted value was

± 0.027 μm, resulting in the MSE = 0.000932. The predicted amplitudes were not consistent across the series: for nontarget modes, they could not belong to the residual PSF of the microscope or to the calibration offset. This suggests that the model did not generalize well.

Training on six modes with three phantoms

The second network was given a more sophisticated task of quantifying six types of aberrations. It was trained with the same parameters as the first one but with three phantoms.

The loss function (Figure 3.28) entered plateau between 855 and 1000 epochs, showing that the CNN reached its maximal performance on the validation set at 854 epochs. Therefore, training this architecture for more epochs is not necessary.

The prediction on the raw images with primary spherical aberrations (mode 12) experimentally introduced on the AO microscope is shown in Figure 3.29. A significant improvement in the quality of predictions was observed after application of the “zero” convolution method (Figure 3.30).

The results demonstrate that:

- PhaseNet was able to predict the amplitudes for six Zernike modes on the images of a fixed biosample.
- Making the test data look more similar to the synthetic training data by applying the “zero” convolution significantly improves the quality of predictions.
- PhaseNet reached its performance limit within 1000 epochs, therefore longer training is not necessary.
- The performance decreases with increasing complexity.
- The results are reliable only on small amplitudes within the amplitude range up to ± 0.1 μm.
- The CNN architecture may not be sufficiently advanced to learn all the necessary patterns in the data to complete this task.

Training on six modes with two phantoms

It is important to acknowledge that in the experiment, where the model was trained on six modes with two phantoms (modes 3 and 8) but was tested on the acquisition series for mode 12, the performance dropped, although demonstrating a rough diagonal trend after applying the "zero" convolution (Figure 3.31. As in the previous example, the "zero" convolution helped. Nevertheless, this result indicated a lack of generalizability. This can be overcome by providing the model with as many phantoms as available for the dataset. Potentially, additional augmentation techniques could help.

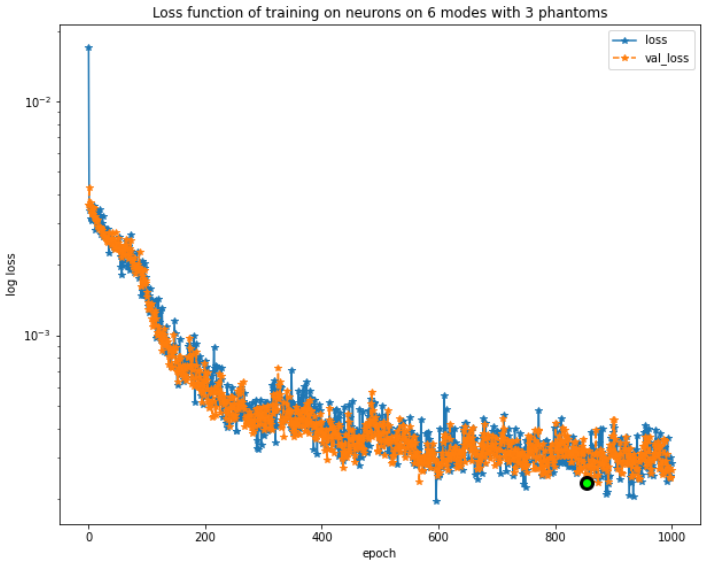

Figure 3.28: Loss function of training on the images of neurons with six modes (3, 5, 6, 7, 8, and 12, in ANSI nomenclature) with three phantoms (mode 3, oblique astigmatism; mode 8, horizontal coma; and mode 12, spherical). The lowest value of the validation loss at epoch 854 during the training for 1000 epochs is shown by a green dot.

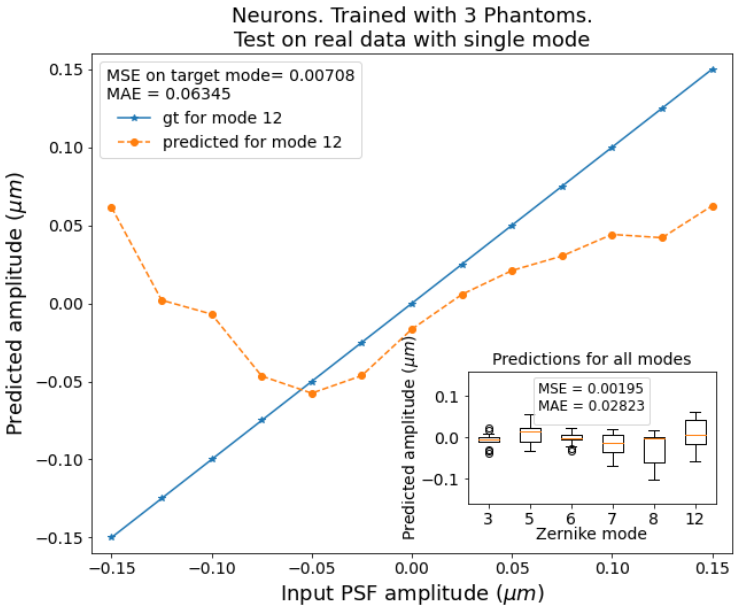

Figure 3.29: Predictions of PhaseNet on the raw images with primary spherical aberrations (mode 12) after training the model on six modes with three phantoms.

Training ResNet architecture

The previous results suggested trying a more advanced CNN architecture. A modified ResNet model referred to as PhaseResNet here, with the structure summarized in Tables 3.2 and 3.3, was trained. The best performance was achieved after training it for 2291 epochs with the learning rate of 0.00035 (Table 3.8). Other learning rate values led to similar performance

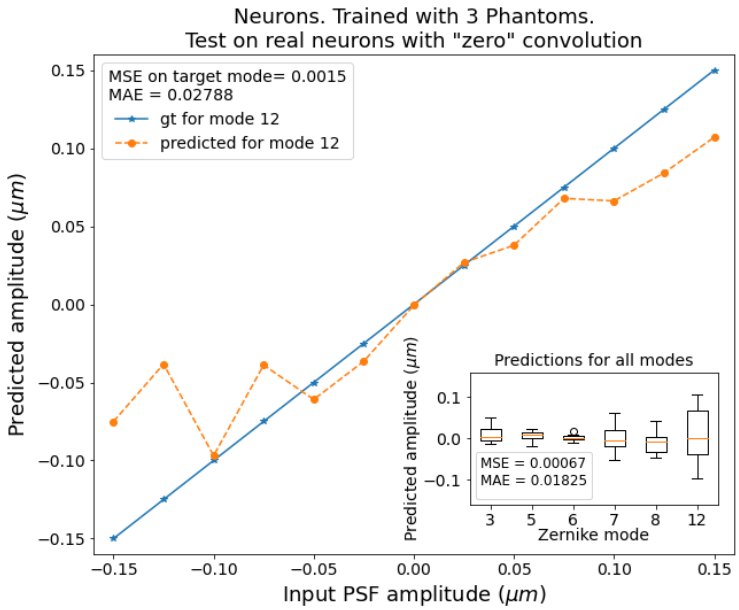

Figure 3.30: Predictions of PhaseNet on the images with primary spherical aberrations (mode 12) convolved with the "zero" synthetic PSF. The model was trained on six modes with three phantoms. The "zero" convolution was key to achieving a good result in the ±0.1 μm range.

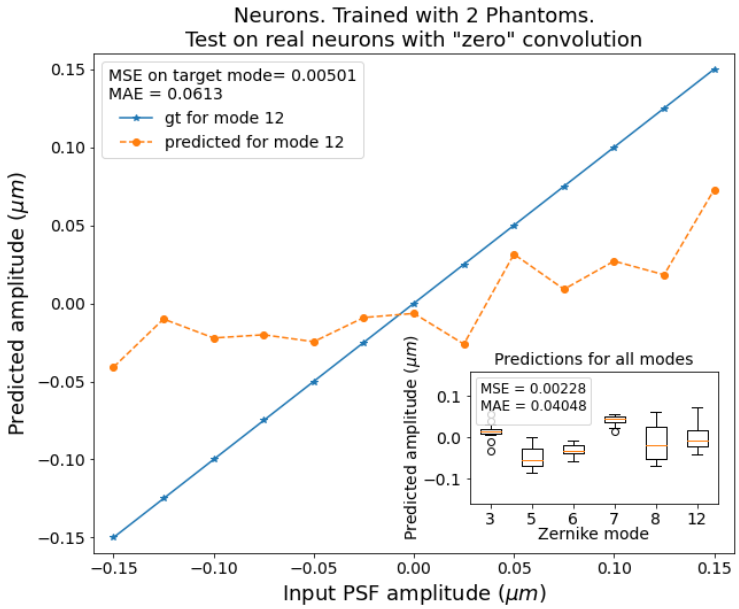

Figure 3.31: Predictions of PhaseNet on the images with primary spherical aberrations (mode 12) convolved with the "zero" synthetic PSF. The model was trained on six modes with two phantoms (modes 3 and 8). The shape of the object in this acquisition series was not seen before.

according to the MSE on the test sets with the "zero" convolution applied. A ten times higher learning rate of 0.0045 was tried, but the validation loss reached only 0.001800 at 600 epochs and was not progressing. The optimal learning rate should lie between 0.0003 and 0.0005. The rate of 0.00035 is considered to be suboptimal because other values were not tried rigorously. Rigorous fine-tuning could improve the results incrementally only. Overall, changing the learning rate within the range from 0.0003 to 0.0005 did not lead to significant differences.

Training of one PhaseResNet model for 2500 epochs (until the model has likely reached its performance limitation, as suggested by early stopping according to validation MSE) took on average 13 hours 15 minutes, whereas training of PhaseNet for 1000 epochs (until the validation loss reached a plateau) lasted 5 hours 18 minutes on a single GPU with 8 GB memory. Although training of PhaseResNet took 2.5 times longer, the huge advantage of it was that usage of PhaseResNet instead of LeNet-based PhaseNet resulted in approximately 2.3 times lower MSE. Compared to the PhaseNet predictions (Figure 3.30), PhaseResNet showed a better result on the test set with the "zero" convolution applied (Figure 3.32). Nevertheless, the predictions for the images with the largest amplitudes of aberrations are less accurate than for those within the range of ± 0.1 μm. Attempted training on synthetic images with aberrations in the range of ± 0.15 μm showed a higher MSE = 0.000639 on testing (Table 3.8).

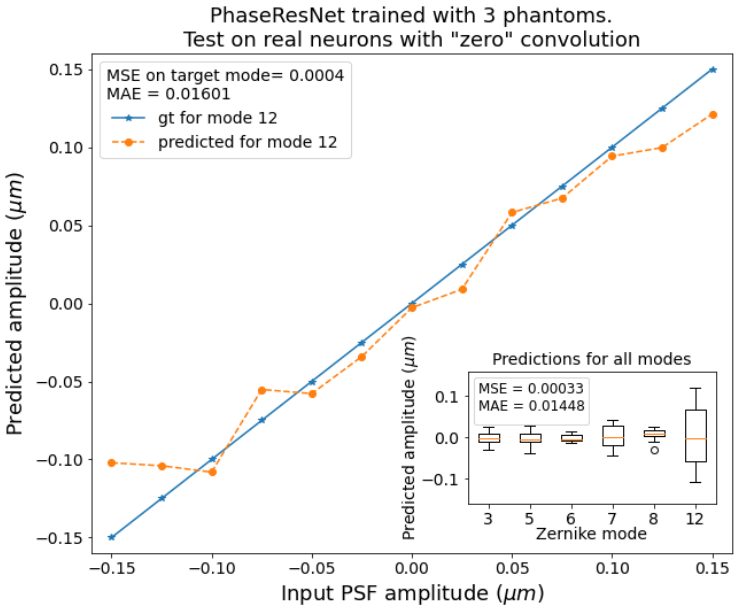

Figure 3.32: Predictions of PhaseResNet on the images with primary spherical aberrations (mode 12) convolved with the "zero" synthetic PSF. The model was trained on six modes with three phantoms.

Restoration

The predicted amplitudes were used for image restoration as described in Subsection 3.3.5 (Figure 3.34). The "zero" convolution trick was needed only for prediction of the PSF; the restoration was performed on the raw image. The images in the figure are normalized with min = 1%, max = 98% with the percentile normalization function from the CSBDeep library [94]. The predicted PSF for the image of a neuron with the amplitude of -0.1 μm, mode 12 (spherical) had the following parameters (Zernike mode in the ANSI nomenclature : amplitude in μm): $\{3 : 0.016003516; 5 : -0.008984767; 6 : -0.010572113; 7 : 0.017577238; 8 : -0.0029398804, 12 : -0.1000473\}$.

Taking into account the results obtained on the synthetic data (astrocytes), the CNN was used to determine the PSF on the in-focus image (the residual PSF of the microscope)

Table 3.8: Comparison of PhaseResNet and PhaseNet models trained on the images of neurons with six Zernike modes

PhaseResNet (ResNet-based) with $a_{j,\text{train}} = \pm 0.1$ μm			
Learning rate	Lowest validation MSE	Epoch	Test MSE
0.00025	0.000109	2492	0.000483
0.0003	0.000099	2374	0.000446
0.00035	0.000100	2291	0.000430
0.0005	0.000106	2362	0.000437
PhaseResNet with $a_{j,\text{train}} = \pm 0.15$ μm			
0.00035	0.000354	2326	0.000639
PhaseNet (LeNet-based) with $a_{j,\text{train}} = \pm 0.1$ μm			
0.0003	0.000233	854	0.000676

and to restore the raw image with this PSF (Figure 3.33). The predicted residual PSF was $\{3 : 0.006032576; 5 : -0.009988198; 6 : -0.010923643; 7 : -0.0013584846; 8 : 0.005916034; 12 : -0.0022144802\}$.

The predictions include only those six modes that the model was trained to recognize. Future studies should include optimization of the network's architecture and parameters for at least 11 modes up to the fourth-order Zernike polynomials.

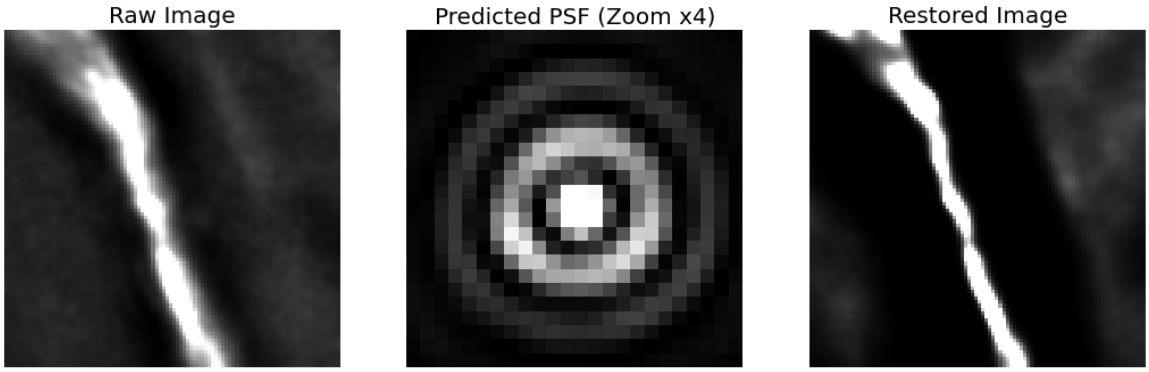

Figure 3.33: Restoration of an experimental raw image of a neuron with mode 12 (spherical) having the amplitude of -0.1 μm (middle z planes are shown). The zoomed-in predicted PSF (middle panel), in addition to the spherical aberration, also contains small amplitudes for other modes.

For the **final training and restoration, PhaseResNet** model with the following parameters was used:

- training amplitude range $a_{j,\text{train}} = \pm 0.1$ μm;
- three phantoms: unaberrated images of 96×96×96 pixels from the target acquisition series with $a_i = 0$ μm for $i = 3, 8, 12$;
- activation functions (except for the last layer with the linear activation): tanh;
- network input shape (crop shape): [50, 50, 50];
- random cropping: true;

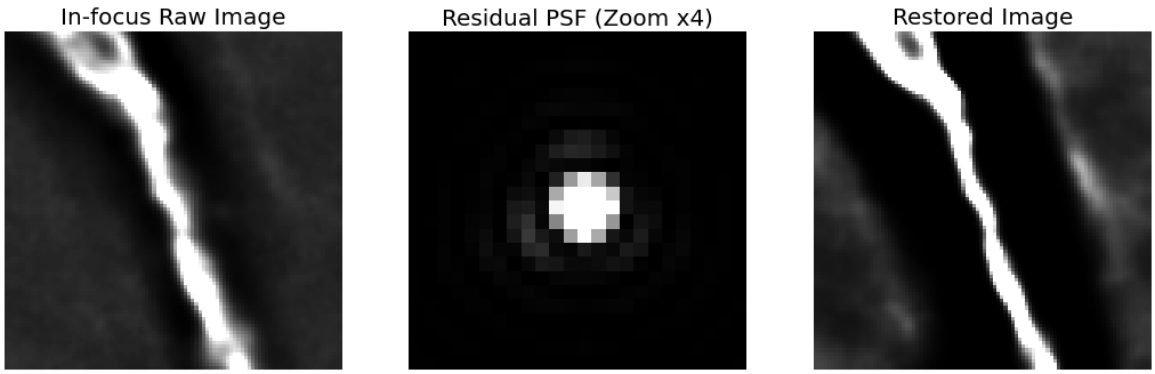

Figure 3.34: Restoration of an experimental in-focus raw image of a neuron (the middle z planes are shown). The zoomed-in predicted residual PSF (middle panel) contains small amplitudes of up to ± 0.01 μm.

- epochs: 2291 (best weights);
- steps per epoch: 5;
- learning rate: 0.00035;
- batch size: 8.

The "zero" convolution was applied to the test images to obtain more accurate predictions. The raw images were restored via deconvolution with the predicted PSFs.

3.6 Conclusions

This study proves potential applicability of PhaseResNet (ResNet-based version of PhaseNet) to estimation of optical aberrations on images of fixed biosamples having clearly observable difference between the sample and the background and with well-defined structure of the sample. The predicted estimations can be used in the deconvolution algorithm to improve the quality of images affected by aberrations.

It was demonstrated that the deep learning-based approach can accurately estimate synthetic aberrations on objects of complex shapes and is able to achieve a decent quality on experimental data only under certain conditions that constrain its practical reliability. The issue of bridging the gap between the synthetically generated training data and the acquired experimental data was partially solved by making the real data more similar to the synthetic data.

In the current state of research, the method may be applicable only to images of relatively thin fixed tissues, where the sample's shape does not vary significantly. The problem of generalizability to varying object shapes (e.g., in cases of moving or developing biosamples, or if the imaged structures are highly diverse) should be worked on before applying PhaseResNet in practice. Generalizability is a common problem in deep learning that can be later addressed by improving the preparation of the training data by, for example, introducing augmentations

to the synthetic training data. Some possible data augmentation techniques for this type of data are mentioned in the next chapter in the context of another research question.

A suboptimal generalizability solution can be achieved by providing the network with the shapes of the objects under study by sampling the phantoms — unaberrated images for generation of synthetic training data — so that they represent the variety of shapes. As shown on the *Drosophila* data, the point spread function on substantially aberrated experimental images cannot be accurately determined without the proper phantom. The requirement to provide such unaberrated (or minimally aberrated) images reduces the number of applications where PhaseNet may be used.

In addition, the improvement of the image quality after performing image restoration by deconvolution with the predicted point spread function requires further quantification. The proposed method may also not work reliably for amplitudes of aberrations larger than ± 0.1 μm. It is difficult to foresee the maximum possible amplitudes in an experimental imaging setup, which further limits practical applicability of the method.

Overall, it was shown in this chapter that PhaseResNet can be used to estimate optical aberrations under the described conditions. However, future work should focus on extending the conditions of applicability and on improving the practicality of this method for sensorless aberration correction.

4

Explainable Multitarget Image Regression

4.1 Abstract

In the previous chapter, I described an approach to quantitative evaluation of monochromatic aberrations in 3D microscopic images — multitarget image regression. Predicting continuous values for 11 classes is a complex task, and to the best of my knowledge, at the start of the work on this chapter in May 2022, no interpretation method existed for multitarget image regression.

Considering that approaches to explaining classification CNNs and their decisions are well-established, it was decided to turn the task of multitarget image regression into that of a multiclass classification. Explainability of a classification CNN was considered the starting point for the development of a new method for interpretation of image regression because classification and regression networks have essentially the same architecture. The main differences between classification and regression consist in the preparation of the ground truth for training, in the activation functions placed in the last layer, the loss functions to optimize, and the metrics to evaluate the performance.

A method called Local Interpretable Model-agnostic Explanations (LIME) [39] was chosen to explain the classification of aberrations because it has received high recognition by the community, is model-agnostic, and outputs both positively and negatively contributing features. The latter means that the method can answer a twofold question, "What parts of image x support prediction y, and what vote against it?" The applicability of LIME to aberrated 3D images was confirmed, as its output aligned with the human understanding of aberrations. Finally, LIME was extended to multitarget 3D image regression. The method was named Image-Reg-LIME and presented at the 1st World Conference on eXplainable Artificial Intelligence (xAI-2023) in July 2023 [143].

In this chapter, using the examples of PhaseNet (see Chapter 3), I demonstrate how to use LIME and Image-Reg-LIME on 3D images and how to ask Image-Reg-LIME the right questions to explain predictions of multitarget regression CNNs.

4.2 Related Work

The multitarget (multi-output) image regression, especially in 3D, is a rather rare task. One of the common tasks is the human pose estimation, for example, by predicting the rotation angles of the body parts in the 3D environment [58]. To the best of my knowledge, no explainable AI methods have been applied to this problem. In a problem of adversarial attacks in 6D object pose estimation, Seg-Grad-CAM (see Chapter 2) has been applied to the segmentation network involved in the pose estimation [144]. The decisions of the convolutional long short-term memory model trained on the daily temperature and precipitation maps of the catchment area of the river to predict the streamflow were explained by visualizing important regions in these maps using a gradient-based technique [89]. Detection of the crack tip and determining the spatial position of the tip can be seen as image segmentation with subsequent calculation of the tip position based on the segmentation mask. The task of segmentation can be interpretable by applying an XAI method post hoc or by using a specially designed explainable model for this task, such as a U-Net-based CNN with a parallel path with GAP and afterward retrieval of gradient-weighted explanations [91].

The estimation of the brain age on MRI scans is a single-target regression problem in 3D. The architecture of U-Noise [81], which was originally used for pancreas segmentation, has been adapted to the brain age regression problem [145]. The authors extended it to 3D and used an encoder–decoder pretraining to make training faster. To visualize the explanations, they showed the brain regions used by the model to predict the age, then normalized and averaged the relevance (intolerance to noise) maps across the dataset. Agreement of the output explanations with the relevant medical studies has been reported.

Another model-specific method for brain age prediction, interpretable classification and regression with feature attribution mapping (ICAM-reg), was published in November 2022 [146], extending the method called interpretable classification via disentangled representations and feature attribution mapping (ICAM) [147] designed for classification to regression. ICAM is a variational autoencoder with a generative adversarial network (VAE–GAN) with feature attribution (FA) maps retrieved from the latent space. ICAM was tested on three binary classification tasks: (1) determining whether the brain contains lesions, (2) detecting Alzheimer's disease, and (3) brain age classification into young (45–60 years) and old (70–80 years). ICAM-reg is a modification of ICAM with an additional regression module (fully connected layer) placed in the encoder to enable training of VAE–GAN for regression tasks. The FA maps from the layers in the middle of VAE–GAN are used to retrieve the network's attention as it was done in ICAM. ICAM-reg was also trained for the binary classification task of lesion detection like ICAM. Saliency maps of several gradient-based XAI methods were compared against FA maps of ICAM and ICAM-reg on how their produced maps match with the positions of lesions (ground truth segmentation masks were available). According to this comparison, ICAM showed the most accurate attribution maps, slightly outperforming

the ICAM-reg architecture. The quality of the FA maps of ICAM-reg on the regression task was not quantified because the chosen comparison metric was applicable only to the binary classification task with the ground truth for lesion locations, whereas similar ground truth masks for brain aging were unavailable.

Explaining estimations of aberrations

The studies of AI usage for recognition of optical aberrations are constrained by the lack of publicly available data. The task of evaluating aberrations and the related problem of sensorless correction of aberrations do not demand explainability because they neither target applications regulated by law nor make decisions crucial for humans' lives. Therefore, XAI has not received much attention in this domain. Nevertheless, XAI is beneficial for any deep learning application to confirm that the DL model makes correct decisions for right reasons.

The preprint by Hu et al. [125] that appeared in January 2023 indicates an interest in interpretable neural networks designed to estimate optical aberrations. In their study, the CNN was designed to determine the aberrations from precomputed pseudo-PSFs calculated as an inverse Fourier transform of the fraction of the Fourier transform of one image with one PSF divided by the Fourier transform of another image with another PSF (eq 3.1).

Central crops of 32×32 pixels from M pseudo-PSFs were stacked as an input to the CNN. Each image was passed through four convolutional layers with trainable 3×3 kernels, each followed by a local 2×2 max pooling. The input layer and the four convolutional layers had additional connections to the concatenation fully connected layer (see Figure S1 in ref [125]), analogous to shortcut connections in U-Net [63]. These five layers were connected to the concatenation fully connected layer through the global max pooling. The concatenation fully connected layer was followed by the intermediate fully connected layer and then by the final fully connected layer for the output regression. The weights between the concatenation fully connected layer and intermediate fully connected layer were used for interpretation: the proportional weights of each block of features coming from a shortcut connection to each of five convolutional layers (see Figure S1 in ref [125]) are summarized in Table 1 in ref [125].

The interpretation of the CNN globally was achieved by measuring the percentage of the weight for each block, which can be perceived as a sort of "statistics" of which parts of the network were more important during training than the others. The authors also showed the distribution of weights for manually designed input images and manually chosen (not trained) convolutional kernels. This approach aimed at constructing an explainable by design gray box model with increased trustworthiness as opposed to classical black boxes used for estimation of aberrations.

This method has the following drawbacks:

1. It is model-specific and ante hoc (works only for the selected architecture predesigned in this special way);
2. It may not be applicable to explaining individual decisions;
3. Its explanations are not visual and may not be directly human-understandable.

4.3 Methods

The method proposed in this dissertation is based on Local Interpretable Model-Agnostic Explanations (LIME) and has the following properties:

1. It is model-agnostic and post hoc (works for any model and is applied after training);
2. It explains individual decisions;
3. Its explanations are visual and are mapped to a human-understandable domain: input images.

However, it cannot be directly used to explain models globally, a disadvantage typical for model-agnostic methods. Nevertheless, the user can understand how the model makes decisions in general by investigating multiple examples of the input data and respective explanations of the predictions for these examples.

4.3.1 LIME

Concept

LIME is an algorithm designed for explaining the predictions of any black-box classifier and of black-box regressors trained on tabular data. LIME supports tabular, image, and text input data for classifiers.

It works by approximating the behavior of the complex model locally by learning a white box model (such as a linear or logistic regression) around the prediction made for a specific instance. This is done by randomly perturbing the input instance and observing the changes in the predictions made by the complex model. LIME then fits the simple white box model to these perturbed instances as input data with the predictions of the black box model as the ground truth for supervised training. This fitted simple model is a surrogate that locally approximates the behavior of the complex model of interest. In this context, locally means that it approximates the behavior of the complex model only for the data points in close proximity (within the prediction space) to the input instance, therefore this approximation is not transferable to other predictions and cannot explain all predictions and the model globally.

LIME retrieves explanations from the surrogate model, which is chosen to be interpretable by design due to its simplicity (e.g., linear regression). The resulting explanation is a set of feature importances that describe the weights of each feature in the simple model, and these weights reflect the impact of these features on the prediction made by the complex model.

Optimization problem

In mathematical terms, a white box model (e.g., a decision tree or a linear regression) $g \in G$, where G is the class of models interpretable by design. The choice of g is constrained by the measure of complexity $\Omega(g)$, which can be, for example, the depth of the decision tree. The prediction of the complex model of interest on the input data instance x is denoted as $f(x)$. LIME requires $f(x)$ to be the probability function of the prediction of the class. To find only

the local data samples z around x, the distance between z and x is defined as $\pi_x(z)$. The loss function $\mathcal{L}(f, g, \pi_x)$ measures how far the predictions of g are from the predictions of f on the local area π_x around x. Finding the explanation $\xi(x)$ for the input x becomes the optimization problem of minimizing the loss $\mathcal{L}$, with the complexity of the model g being constrained by $\Omega(g)$:

$$\xi(x) = \arg\min_{g \in G}(\mathcal{L}(f, g, \pi_x) + \Omega(g)) \tag{4.1}$$

Because the user of LIME has to select the class of models G and set the complexity of the model $\Omega(g)$ by defining the number of features K to be used by g, the optimization problem in eq 4.1 is reduced to finding the loss $\mathcal{L}$:

$$\xi(x) = \arg\min_{g \in G}\mathcal{L}(f, g, \pi_x) + \Omega(g) \tag{4.2}$$

The loss function $\mathcal{L}$ is defined in LIME as the locally weighted square loss:

$$\mathcal{L}(f, g, \pi_x) = \sum_{z, z' \in \mathcal{Z}} \pi_x(z)(f(z) - g(z'))^2 \tag{4.3}$$

The distance measure $\pi_x(z)$ was defined as an exponential kernel with the width σ on the distance function D: $\pi_x(z) = \exp(-D(x, z)^2/\sigma^2)$. For images, the authors stated $D = L2$, which is the Euclidian distance in n-dimentional feature space. The user can select other metrics for D, such as a more "advanced" version of $L2$ — cosine similarity, popular in natural language processing applications.

Input perturbation

To make the perturbations meaningful, groups of pixels were used instead of individual pixels. Changing individual pixels can lead to changes in predictions, but such adversarial attacks are barely understandable by humans. The pixels were grouped according to semantic meaning, which was achieved by segmentation of the input image, as a preprocessing step, with an unsupervised segmentation algorithm. In the LIME study, such segments are called superpixels [39].

The input perturbation begins with selecting superpixels, which are small regions of similar color or texture within an image. The superpixels are then treated as features and "turned off" by occlusion of the region with a color (defined by the parameter "hide color" set by default to the mean value of the segment) or "turned on" by keeping the segment present. N perturbed images with some segments "on" and others "off" form the new dataset Z. First, the complex model f makes predictions on this new dataset. Second, the interpretable model g is trained on the new data with the predictions $f(Z)$. The quality of the approximation $g(Z) \approx f(Z)$ depends on the size N of the dataset. In practice, a good value for N is 1000 data samples. The image is segmented only once at the preprocessing step, and the positions and shapes of the superpixels are fixed for each $z \in Z$. Because of this, $g(Z)$ can learn the impact of each segment.

Last, the prediction of $g(x)$ for image x is explained: the weights of the segments (features on which g was trained) are mapped back to the image, forming a heat map. The resulting

explanation is essentially a set of feature importances indicating the impact of each superpixel on the prediction of the image classifier. The impact can be both positive (features support the prediction) and negative (features vote against the prediction of the class).

4.3.2 Superpixel algorithms

The selection of superpixels in LIME is effectively a form of image segmentation. Besides aiming at semantically meaningful segmentation, the purpose of dividing an image into superpixels is to reduce the number of features to be considered while still preserving enough information. Superpixels should not be tiny (otherwise the features are too small and not meaningful enough) or very large (to avoid too generic explanations). By breaking down the image into smaller segments, LIME can generate a more precise explanation. The number of segments sets a tradeoff: it is set by the user, and the rule of thumb is to try several values on a couple of input images to determine which number gives sufficiently precise and, at the same time, meaningful explanations.

The superpixel segmentation can be performed with any classical image segmentation algorithm such as Quick Shift [148] from Scikit-image library [99] implemented in LIME by default. Quick Shift accepts only RGB images or those converted into the $L^*a^*b^*$ color space [149]. The original implementation of LIME for image data was designed for explaining the classification of natural RGB and grayscale 2D images. As inputs, LIME requires a 2D image (with an additional input dimension for RGB images) and the model's prediction function that can calculate predictions on the input image.

To extend LIME to 3D input, Quick Shift, which works only on 2D images, had to be replaced with another algorithm. The aberrated images in this study and point spread functions (PSFs) are three-dimensional, hence the model's decisions have to be explained in 3D. Downscaling the task to 2D does not make sense: as has been shown by Debayan Saha et al. [108], a single z plane is insufficient for high-quality predictions for all 11 Zernike modes.

In this work with 3D images, Quick Shift was replaced with Simple Linear Iterative Clustering (SLIC) [150], a K-means-based clustering algorithm in Color-(x, y, z) space. Specifically designed as a superpixel algorithm, SLIC executes K-means in the 5D space of color information and image location and has many similarities with Quick Shift. The clustering approach is highly efficient due to its simplicity. To produce appropriate results, SLIC is recommended to be used in the $L^*a^*b^*$ color space instead of RGB for color images. As with Quick Shift, the compactness parameter in SLIC trades off color similarity and closeness of points within the segments, whereas the number of segments determines the number of centers for K-means.

4.3.3 LIME for 3D image classification

Replacement of Quick Shift with SLIC allowed extendability of LIME to 3D images of fluorescent beads. The last activation function was chosen so that it outputs a probability vector, which is the form of predictions that LIME operates with. The network was trained to assign a single class (Zernike mode) to 3D experimental images, into which only one single mode was intentionally introduced with an adaptive optics (AO) microscope.

The architecture of the network is summarized in Table 4.1. This CNN is similar to PhaseNet (Table 3.1) in Chapter 3. More details on training and the layers that differ from the PhaseNet architecture are provided in Subsection 4.4.3.

Table 4.1: Model summary for 3D classification CNN

Model: 3D classification CNN		
Layer (type)	Output shape	Parameters
X (InputLayer)	(None, 32, 32, 32, 1)	0
conv1 (Conv3D)	(None, 32, 32, 32, 8)	224
conv2 (Conv3D)	(None, 32, 32, 32, 8)	1736
maxpool1 (MaxPooling3D)	(None, 32, 16, 16, 8)	0
conv3 (Conv3D)	(None, 32, 16, 16, 16)	3472
conv4 (Conv3D)	(None, 32, 16, 16, 16)	6928
maxpool2 (MaxPooling3D)	(None, 32, 8, 8, 16)	0
conv5 (Conv3D)	(None, 32, 8, 8, 32)	13,856
conv6 (Conv3D)	(None, 32, 8, 8, 32)	27,680
maxpool3 (MaxPooling3D)	(None, 32, 4, 4, 32)	0
conv7 (Conv3D)	(None, 32, 4, 4, 64)	55,360
conv8 (Conv3D)	(None, 32, 4, 4, 64)	110,656
maxpool4 (MaxPooling3D)	(None, 32, 2, 2, 64)	0
conv9 (Conv3D)	(None, 32, 2, 2, 128)	221,312
conv10 (Conv3D)	(None, 32, 2, 2, 128)	442,496
batchnorm (BatchNormalization)	(None, 32, 2, 2, 128)	512
maxpool5 (MaxPooling3D)	(None, 16, 1, 1, 128)	0
flat (Flatten)	(None, 2048)	0
dropout1 (Dropout)	(None, 2048)	0
dense1 (Dense)	(None, 128)	262,272
dropout2 (Dropout)	(None, 128)	0
dense2 (Dense)	(None, 64)	8256
Y (Dense), Softmax activation	(None, 11)	715
Total parameters: 1,155,475 Trainable parameters: 1,155,219 Nontrainable parameters: 256		

The operation of LIME for 3D image classification is shown in Figure 4.1 on the example task of classification of the optical aberration modes (Zernike modes). An already trained network with convolutional layers and three dense layers at the end (other layers are omitted to simplify the schematic) is used to produce predictions. The last layer outputs the probability vector. To make the final class prediction, the index of this vector with the highest probability score is taken. LIME uses the "raw" probabilities and does not need the prediction itself.

LIME uses the prediction function of the network to construct a new dataset (as described in Subsection 4.3.1). During the preparation of the new dataset, the superpixel algorithm SLIC divides the input 3D image into the user-defined number of volumetric segments. These segments are either occluded (replaced with black pixels instead of gray set up by default) or kept to measure their impact on the prediction.

After the surrogate model is trained, the user can choose the class prediction to explain, which could be any class, not only that with the highest probability. For example, if the model

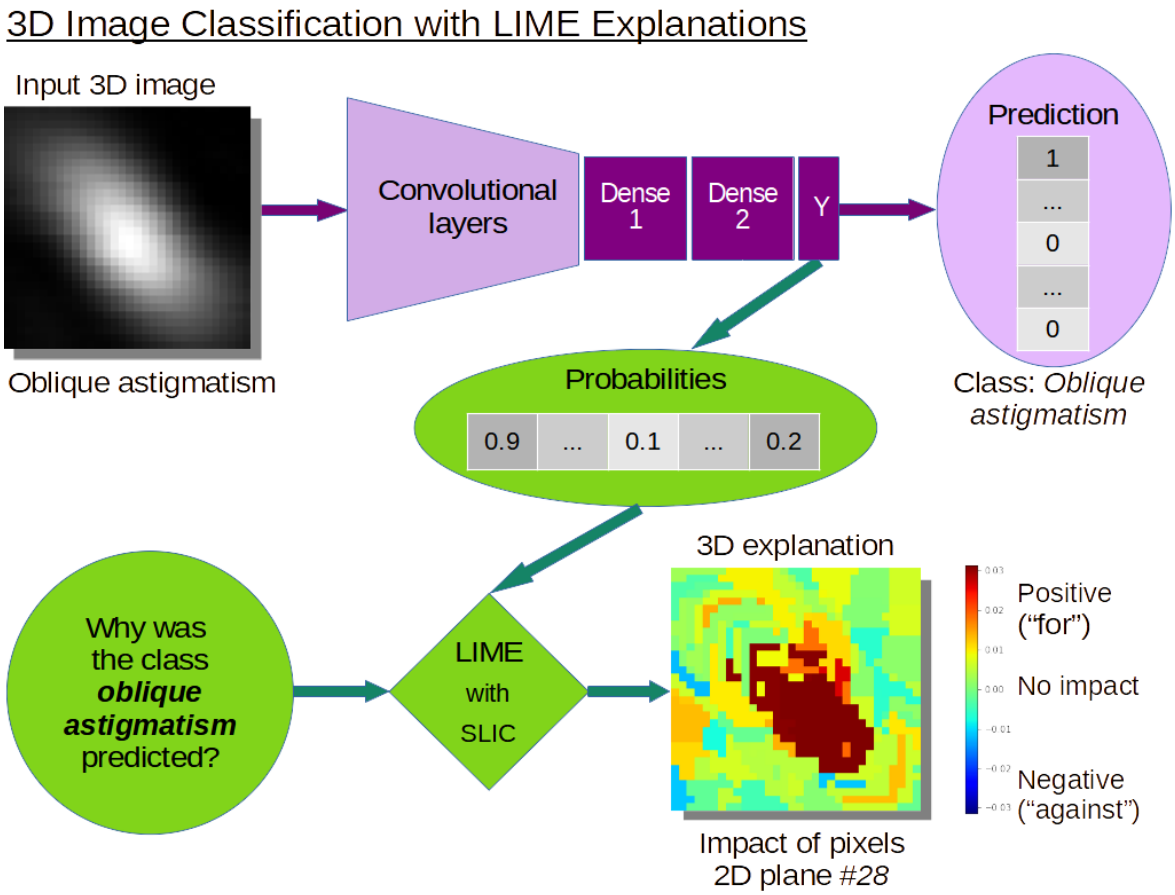

Figure 4.1: Schematic of LIME explanations for 3D image classification. The convolutional neural network with dense layers is trained on 3D images (the training procedure is not shown). The prediction function of the CNN that produces the probability vector (the output of the last dense layer without thresholding) is passed to LIME. The input 3D image is segmented with SLIC. A new dataset for training a surrogate model is generated: several segments are occluded to form the images and the prediction function produces the ground truth. The user defines the class prediction to be explained. LIME explains the prediction of the surrogate model by retrieving the weights of each segment for this prediction, producing a 3D explanation heat map. The input image and explanation have a size of 32×32×32 pixels, plane 28 from the z stack is shown for both in 2D view.

predicted class A incorrectly and the user wants to understand why, this can be done by selecting one class and asking LIME:

- Why was class A predicted?

  Then, the user can observe what input features speak for predicting class A (shown in red and orange on the heat map at bottom right).

- Why was class B predicted?

  Posing this question about the ground truth class B that was actually not predicted forces LIME to generate a heat map where the segments with a positive impact would mean that they support the low value predicted for B, whereas the segments with a negative impact would speak against B, meaning that because of them class B was not predicted with the highest probability.

The output explanation consists of weights (impact) of each segment for one particular prediction for which an explanation was requested. This result is volumetric and can be visualized as a series of 2D planes.

LIME has an option to visualize only top n positive or negative features, where n is defined by the user. This works particularly well for natural images and can be beneficial for images with optical aberrations when LIME is asked, "Why was this class not predicted?"

4.3.4 Image-Reg-LIME: LIME for 3D image regression

The original LIME was designed for 2D image classifiers with a probability activation function for class predictions. An approach to handling the problem of higher dimensionality is described in the previous subsection.

The original LIME produces explanations based on the predicted class probabilities (e.g., the probability of the class that the user wishes to explain). In the case of classifiers with one-hot encoded outputs (0 and 1), the predictions are derived prior to the last activation function (some classifiers possess a method "predict probabilities," from others an intermediate output should be retrieved manually). If the classifier outputs probabilities by design, its raw output is used by LIME. For classifiers with thresholding, the output with probabilities has to be used.

In the previous subsection, the probabilities of classification of aberrations were required for LIME. The last activation function was softmax, producing probabilities with the respective output range of (0, 1):

$$\mathrm{softmax}(t_i) = \frac{\mathrm{e}^{t_i}}{\Sigma_j \mathrm{e}^{t_j}} \tag{4.4}$$

where t is the vector of outputs of this layer of the network, t_i is the output of the neuron i, e is the exponential function, and Σ_j is the sum over all elements $j \in N$ of this layer.

Sigmoid is another function commonly used in dense layers that outputs the probabilities:

$$\mathrm{sigmoid}(t_i) = \frac{1}{1 + \mathrm{e}^{-t_i}} \tag{4.5}$$

In PhaseNet [108], the second to last activation function for regression was the hyperbolic tangent:

$$\tanh(t_i) = \frac{\mathrm{e}^{t_i} - \mathrm{e}^{-t_i}}{\mathrm{e}^{t_i} + \mathrm{e}^{-t_i}} \tag{4.6}$$

The function tanh behaves similarly to softmax by clipping the input–output function into an "S" shape, but the output range of tanh is $(-1, 1)$, and it is mostly used as an intermediate activation function.

The last activation function of PhaseNet was linear:

$$\mathrm{linear}(t_i) = t_i \tag{4.7}$$

The output range for PhaseNet should include both negative and positive values because aberrations can have positive and negative amplitudes. In addition, the output cannot be treated as probabilities summing up to 1 because the presence of one aberration mode with a large amplitude does not exclude a chance that other modes can also have large amplitudes.

Therefore, replacing the last linear activation in PhaseNet with a function that outputs a probability vector was not an option.

The most important for LIME is to receive predictions on a continuous scale to determine changes in the predictions precisely when the input image is manipulated. Because the CNN is considered to be a "black box," there are no additional requirements for the differentiability of the activation functions.

In this research, the activation functions and model architecture remained the same (tanh for the second to last layer and linear activation for the last layer).

Usage of Image-Reg-LIME for multitarget image regression in 3D is shown in Figure 4.2. The working principle is similar to that of LIME for 3D image classification (Figure 4.1). In addition to the standard question, "What speaks for and against the prediction?" the proposed method offers an option to compare the predicted value with a reference value for the same class and answer the question, "Why was the reference value for the target class not predicted?" The ground truth value for the target mode is 0.093 μm, an amplitude of 0.088 μm was predicted, which is very close to the ground truth amplitude. Because of the neglectable difference between the ground truth and the prediction, the explanation shows barely interpretable features, thus, another reference value was selected for better visualization. For the shown task of estimation of optical aberrations, this question may sound as, "Why the predicted amplitude for oblique astigmatism is not 0.050 μm?"

The key contributions of this research are:

1. To explain the predicted values ("Why was the value a_i for the target class i predicted?"), the prediction function f of a regression CNN can be taken as is;

2. To answer the question, "Why the value a_i^* was not predicted instead of a_i for the class i for the input image x?" LIME requires a modified prediction function f^* to select the correct dataset Z for training of the surrogate model $g(Z)$. The function f^* for the perturbed image $z \in Z$ is defined as

$$\text{for } z \neq x\text{:}\quad f^*(f, z, i, a_i, a_i^*) = f(z) \tag{4.8}$$

$$\text{for } z = x\text{:}\quad f^*(f, z, i, a_i, a_i^*) = \begin{cases} f(x), & \text{where } j \neq i \\ a_i^*, & \text{where } j = i \end{cases} \tag{4.9}$$

   Index j is an index in the N-dimensional vector a of the output prediction, and $z = x$ is the input image x with all segments present, the predictions for which are used as the reference point to determine the most relevant perturbed images Z.

The prediction for the input image is shifted to assign more similarity to perturbed images with the prediction close to the target value a_i^*, so that the segments responsible for the prediction of the desired value receive higher weights, according to the cosine similarity distance metric (Figure 4.3).

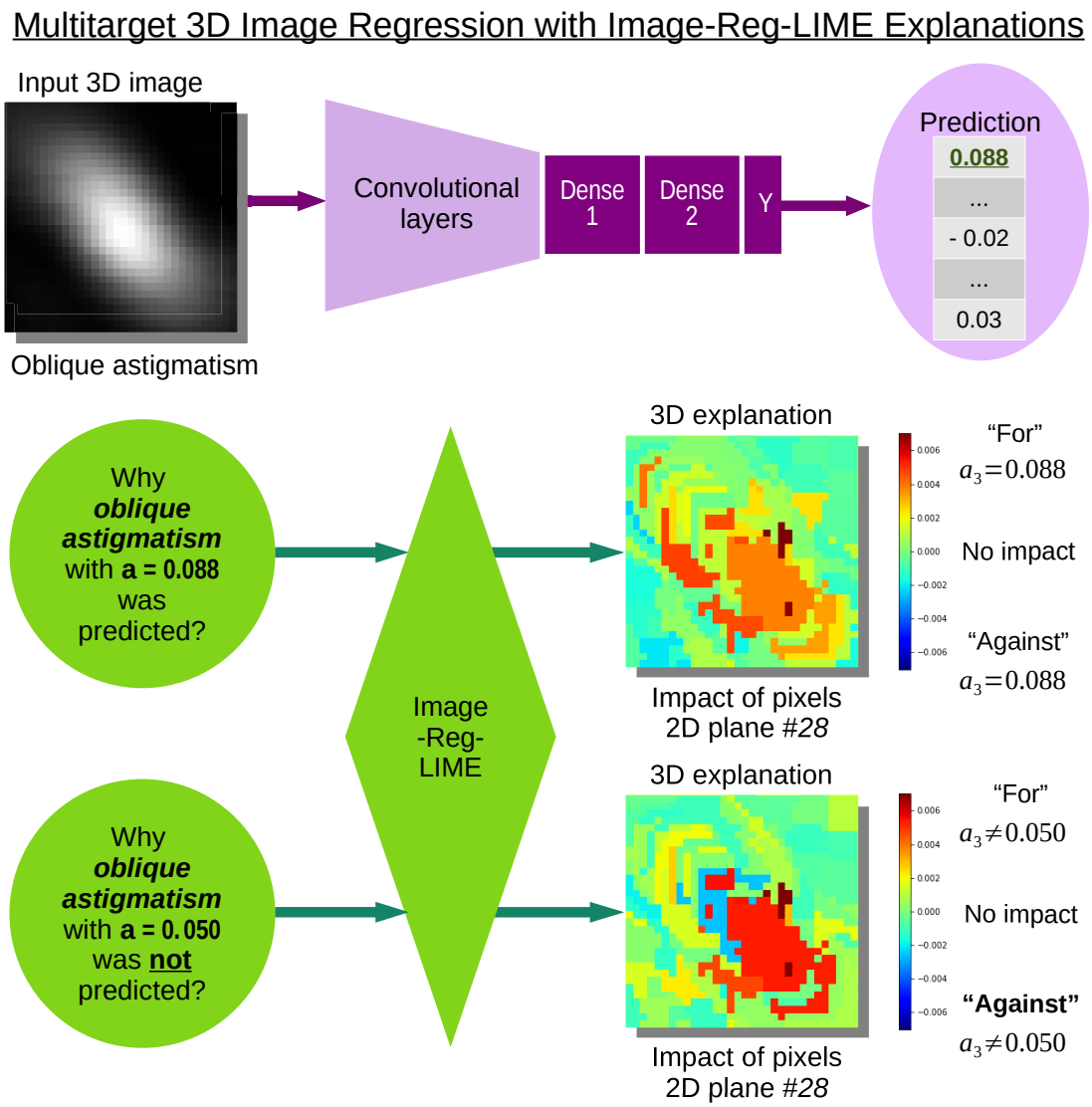

Figure 4.2: Schematic of Image-Reg-LIME explanations for 3D image regression. The convolutional neural network with dense layers is trained on 3D images (the training procedure is not shown) to predict a 1D array of continuous values. The wrapper (a modified function accepting the value of interest) around the prediction function of the CNN is passed to Image-Reg-LIME. The input 3D image is segmented with SLIC. A new dataset for training a surrogate model is generated: several segments are occluded to form the images, the prediction function produces the ground truth. The user defines the target (an index in the prediction array) and the value to be explained. Image-Reg-LIME explains the prediction of the surrogate model by retrieving the weights of each segment for this prediction, producing a 3D explanation heat map. The input image and explanation have a size of 32×32×32 pixels, plane 28 from the z stack is shown for both in 2D view.

4.4 Results: Classification of Aberrations

Data

The experiments described in this and the next sections were performed on the dataset with single-mode aberrations on images of fluorescent beads from the PhaseNet publication

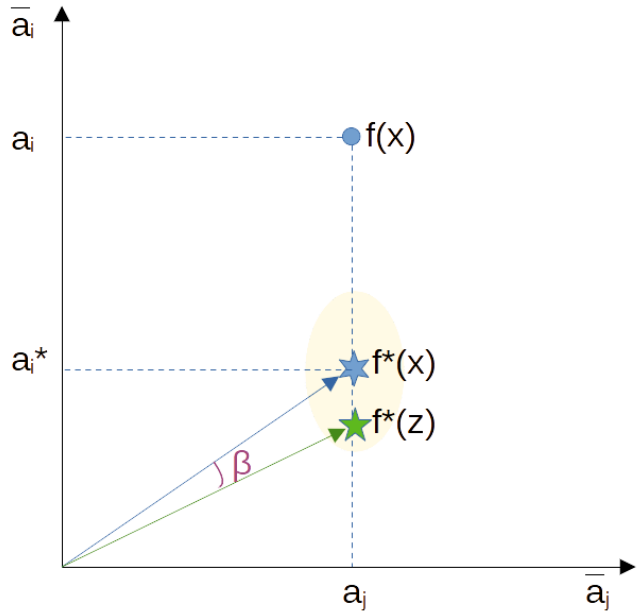

Figure 4.3: Schematic of the modified prediction function f^*. 2D section of N-dimensional prediction space is shown. Here, x is the original unperturbed image, z is the perturbed image, $f(x)$ is the prediction of the CNN for x, $f^*(x)$ is the shifted prediction for x, $f^*(z) = f(z)$ is the prediction for z, i is the target class (aberration mode), j is another class ($j \in N, j \neq i$), a_i is the predicted value in $f(x)$ for class i, a_i^* is the target value for class i, a_j is the predicted value for class j, β is the angle between the vectors for $f^*(x)$ and $f^*(z)$ in the prediction space (the measure of similarity between $f^*(x)$ and $f^*(z)$ is cos β). Vertical axis shows the predicted values for class i, horizontal axis — the predicted values for class j. Locally similar predictions with the largest weights in LIME's surrogate model are shown in yellow. Function f^* shifts the area of predictions considered the most relevant (yellow ellipse) close to the value a_i^* being "explained."

[108], different from the Fluorescent beads dataset used in Chapter 3. This dataset was chosen since it was usable for a classification task besides the original regression one.

4.4.1 Transforming the regression task into classification

PhaseNet was chosen as the baseline architecture because it was proven to perform well on the regression task (prediction of amplitudes for each aberration mode) on this dataset. The results of the search for a more optimal architecture are described in Subsection 4.4.3.

PhaseNet's synthetic data generator takes the microscope's parameters, a list of aberration modes, and the maximally allowed absolute value of amplitudes (the allowed range of amplitudes). It outputs a synthetic image with mixed aberrations and a vector of amplitudes (one float value per mode). Such a generator is not designed for preparing a classification training dataset. To do this, either the generator has to be changed to output synthetic images with a single aberration mode and a single integer value of the amplitude, or the real data has to be used. For the latter approach, data augmentation is necessary to prevent overfitting because the number of images in the dataset is relatively small for training a neural network.

Training on the real data with augmentations was preferred for two reasons: (1) there is a discrepancy between the synthetic and real data (Figure 4.4), and (2) this approach, compared

to using synthetic images, provides a better control over the data distribution from which the images for training, testing, and validation are sampled.

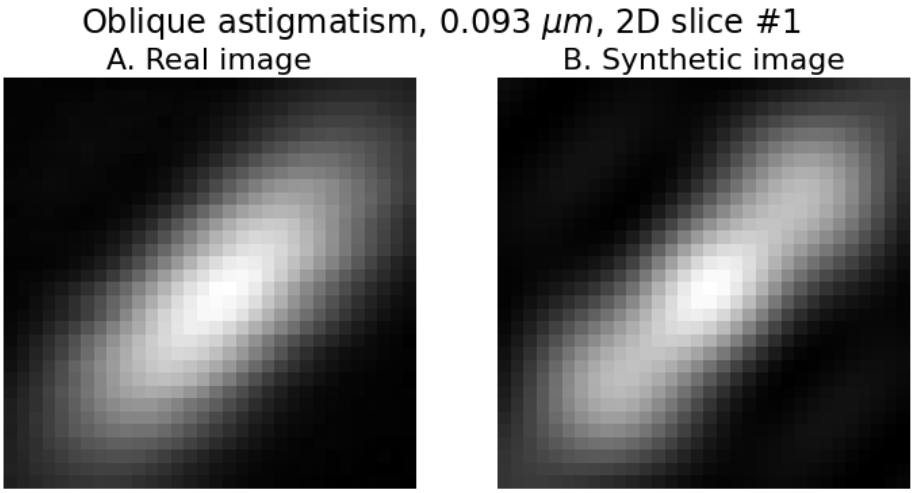

Figure 4.4: Acquired image of a real bead[108] and a synthetically generated aberrated image.

4.4.2 Data augmentation

The dataset was split into the training, validation, and test sets containing 110, 44, and 44 images, respectively. The training set contained 10 images for each of 11 modes, the validation and test sets contained four images for each mode. The images were randomly selected to ensure a variety of amplitudes, but the aberration modes were present in equal proportions to avoid the class imbalance. Identical sets were used to train multiple models and evaluate their performance.

To enlarge the training set, each raw image from it was passed to the augmentation block of the code a certain number of times, and this parameter was configurable. It was set to 100, meaning that the networks were trained on 11000 augmented images instead of 110 raw images. Enlarging the dataset more than 100 times did not improve the quality of predictions, therefore the enlargement factor was kept at 100.

The nature of the data limited the list of suitable data augmentation techniques. Standard computer vision augmentations [151] such as horizontal or vertical flips, rotations, distortions, blurring, and sharpening are unsuitable for this type of data. Flips and rotations in the (x, y) plane change the direction of some aberrations (e.g., pairs: oblique astigmatism and vertical astigmatism, horizontal and vertical coma, etc.) or have no effect on symmetrical aberrations such as primary spherical. Distortions change the size and shape of an object and the magnitude of aberrations, and may change the nature of aberrations. Blurring and sharpening may make small-amplitude aberrations invisible in the images.

In this work, the following data augmentation methods were used:

1. Translation

2. Brightness change
3. Contrast stretching
4. Adding noise (Gaussian, Poisson, salt-and-pepper)

The first augmentation technique, translation, is random shifting of the object up, down, right, or left. The images, having a size of 32×32×32 pixels, were padded to the size of 60×60×60 pixels with a constant intensity value randomly sampled from the range of $(10^{-11}, 10^{-6})$ for each image. This range was selected according to the intensity of background pixels. Each padded image was randomly cropped, with the output size of 32×32×32 pixels, which made most of the images contain only a part of the object and have it located in different parts of the image.

To change the brightness, a random float number between -0.5 and 0.5 was added to the images.

Contrast stretching was performed using skimage.exposure.rescale_intensity method from Scikit-image library [99]. The stretching range was randomized by sampling the upper bound from the range of $(0.6, 1.4)$.

The functions for adding Gaussian, Poisson, and salt-and-pepper noise were taken from the module skimage.util.random_noise.

In the final step, the augmented images were normalized using the percentile normalization function csbdeep.utils.normalize [94]. The normalization method included the parameter "clip to [0, 1]" for rescaling the images to the value range of $[0, 1]$. Its effectiveness was also tested.

4.4.3 Parameter search

To compare the augmentation techniques mentioned above, the model with the following parameters was selected on the basis of a preliminary search:

- dropout layers between the dense layers with the amount of dropout equal to 0.3;
- last activation: softmax;
- learning rate: 0.0002;
- added batch normalization prior to the dense layers;
- the "flatten" layer of PhaseNet was replaced by the global average pooling layer;
- epochs: 100;
- batch size: 50.

The other training parameters and the CNN architeture remained the same as in PhaseNet.

During the second iterative parameter grid search, four models achieved an equal classification accuracy of 0.8636. They differed in the noise-adding parameters: whether Poisson noise or salt-and-pepper noise was added. One of the models had clip = true (percentile normalization was clipped to the range of [0, 1]), in the other models, clip = false. The model that showed the slowest overfitting (according to the loss function and validation accuracy) was chosen as the baseline for the subsequent parameter search. It was trained on the data with translation, brightness, contrast, and Poisson noise augmentations. The impact of Poisson and salt-and-pepper noise was investigated further during the architecture search (final iteration of

the parameter grid searche) because these parameters differed across the winning models. The investigated and finally selected parameters are summarized below.

In total, the following parameters and values were tested (the values of the best-performing model are shown in bold):

1. Add augmentation translation: [**true**, false];
2. Randomized brightness change: [**true**, false];
3. Randomized contrast stretching: [**true**, false];
4. Add Gaussian noise: [true, **false**];
5. Add Poisson noise: [true, **false**];
6. Add salt-and-pepper noise: [**true**, false];
7. Clip pixel intensities to [0, 1] after normalization: [true, **false**];
8. Dropout of weights before the dense layers (fraction): [none (no dropout), 0.2, **0.3**, 0.4, 0.5];
9. Intermediate activation functions: [**tanh**, softmax, sigmoid, relu];
10. Last activation function: [**softmax**, sigmoid];
11. Number of convolutional blocks (two convolutional layers with one max pooling): [4, **5**];
12. lr: [0.0001, **0.0002**, 0.0003, 0.0004];
13. Batch normalization after the convolutional layers: [**true**, false];
14. Dimensionality reduction layer before the dense layers: [global average pooling, **flattening**];
15. dense1 : [2048, 1024, 256, **128**, 64];
16. dense2 : [2048, 1024, 512, 256, 128, **64**, 32, 16];
17. Kernel regularization in the first convolutional layer of each convolutional block (L2 regularization factor): [**none**, 0.01, 0.05];
18. Kernel regularization in the second convolutional layer of each convolutional block (L2 regularization factor): [**none**, 0.01, 0.05].

This model was trained to classify 11 Zernike aberration modes and achieved the following results on the test set:

- accuracy: 95.45%,
- precision: 93.18%,
- recall: 95.35%.

These metric values were considered sufficient. Further parameter search was not necessary for exploring the applicability of LIME to this type of data. The architecture of this model is summarized in Table 4.1. The model was used in the XAI research presented in Subsection 4.4.5.

4.4.4 Clustering of 3D images

LIME [39] is a powerful model-agnostic explainer, but its default superpixel algorithm Quick Shift [148] used for unsupervised segmentation, or clustering, has to be replaced with SLIC [150] to work on 3D input images.

In this study, the following parameters of SLIC were changed from their default values in the Scikit-image [99] implementation:

1. The number of segments: $8 \times 8 \times 8 = 512$. The input images had a size of 32×32×32 pixels, and the number of segments was chosen as a rescaling factor of the input size;
2. Compactness: 0.01. The low value allowed segments of irregular shapes besides cubic segments (when the compactness is large, the proximity between pixels strongly influences the clustering result);
3. Maximal number of iterations: 50. The maximum number of iterations for K-means algorithm was increased to help refine the clusters with only an insignificant increase in the calculation time;
4. Channel axis: None, the images were grayscale;
5. Random seed was fixed to ensure consistency in segmentation and reproducibility.

In Figure 4.5, two neighboring z planes from the 3D clustering result produced by SLIC are shown. Two other non-neighboring z planes selected from the segmented volume are shown in Figure 4.6. The input image of a fluorescent bead, taken from ref [108], has oblique astigmatism with an amplitude of 0.093 μm. The first z plane of the input image is shown in Figure 4.4.

4.4.5 Explanations of classification

To validate LIME's explanations for the estimation of optical aberrations on these 3D data, the first step is to ensure adequate performance in the simplest scenario: explaining the classification of aberration modes on the images of point source objects, where a human can clearly identify the aberration types. The types shown in the examples are easy to understand without a deep background in optics: oblique astigmatism (Figure 4.4), vertical astigmatism (similar to oblique astigmatism but the direction of distortions is vertical or horizontal, as shown in Figure 4.7), and primary spherical (looks like bright and dark concentric rings around the point in the (x, y) plane). The sources of these aberrations were described in Subsection 1.4.1.

Ridge regression model was used as the surrogate and fitted into 1000 perturbed data samples. The procedure is shown in Figure 4.1.

An explanation for the question, "Why was the correct class label predicted?" (Figure 4.7) shows:

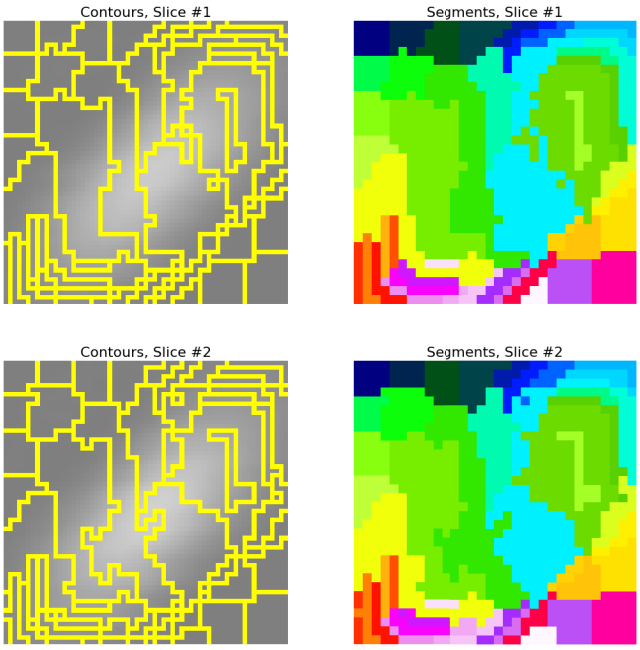

Figure 4.5: Clustering using SLIC in 3D. Two neighboring z planes are shown.

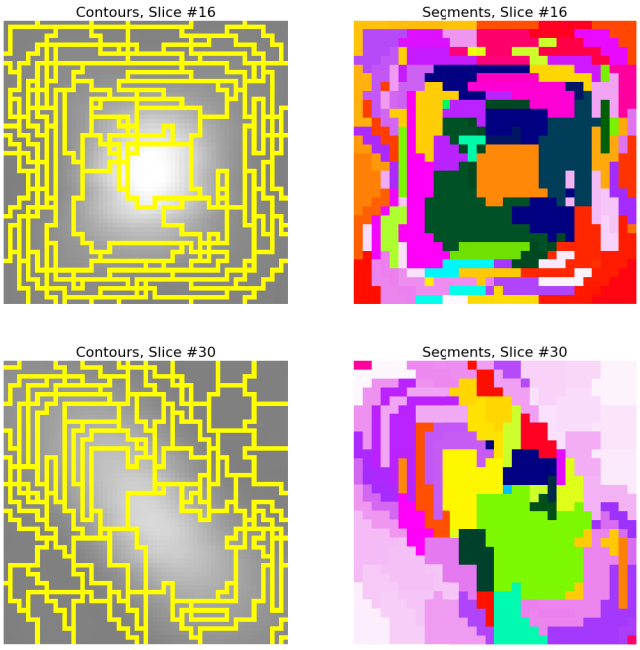

Figure 4.6: Clustering using SLIC in 3D. Two non-neighboring z planes (from the middle and from the end of the stack) are shown.

1. How vertical astigmatism looks like: the aberration with the positive amplitude $a = 0.093$ μm stretched the point in the vertical direction ((x, y) plane) at the beginning of the 3D acquisition stack along the z plane (at the end of the stack, the direction is horizontal).

2. How to read LIME explanations: the heat map corresponds to the impact of each segment (clustered with SLIC) on the CNN's decision. Positive values (yellow, orange, and red) are assigned to the segments supporting the prediction, values around zero (green) are assigned to the neutral segments not influencing the decision, and negative values (blue spectrum) correspond to the clusters speaking against the prediction. The color scale depicts the importance values on the heat map.

3. The CNN correctly learned the features of the class: the elongated shape of the point spread function (PSF) in the vertical direction was picked as the most positively important part of the image for the prediction of the class "vertical astigmatism".

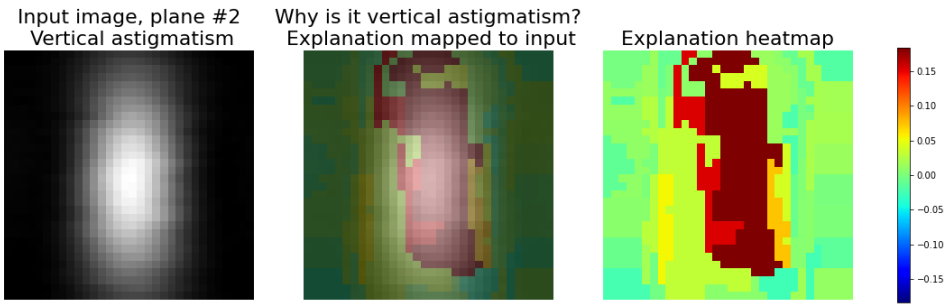

Figure 4.7: LIME explanation for the correct classification of an aberrated image with vertical astigmatism (left). The heat map represents the influence of each part of the input image (segmented by SLIC) on the CNN's prediction. Positive values (yellow, orange, and red) are allocated to the segments supporting the prediction, neutral values (green) are assigned to the segments not affecting the classification, and negative values (blue spectrum) correspond to the clusters arguing against the prediction. The color scale indicates the impact of each segment on the heat map. The explanation with LIME answers the question, "Why was the class 'vertical astigmatism' predicted by the CNN for the input image?" The image and explanation are three-dimensional, plane 2 is shown for both in 2D view.

In Figure 4.8, LIME answers the question, "Why was oblique astigmatism predicted, not vertical astigmatism?" for an image with experimentally introduced oblique astigmatism with the positive amplitude $a = 0.093$ μm. In the middle image, the segments having a positive influence on the correct prediction (oblique astigmatism) are highlighted in red. In the right image, the segments with a negative impact on the other class, vertical astigmatism, which earned a low probability score and was not output in the final prediction, are shown in blue. This figure suggests that the network "paid attention" to the diagonal distortion typical for oblique astigmatism (middle image), did not find the vertical stretch typical for vertical astigmatism (right image), and decided that the image contains oblique astigmatism and no vertical astigmatism.

Figure 4.8: LIME explanation for classification: segments with a positive impact on the correct prediction (oblique astigmatism) and segments with a negative impact on the class that received a low probability prediction (vertical astigmatism). Plane 2 from the z stack of the experimental raw image with oblique astigmatism is shown on the left. The image in the middle represents the parts of the input (in red) that positively influenced the prediction of oblique astigmatism. The image on the right shows the segments (in blue) that negatively impacted the predicted probability of vertical astigmatism in this input, which means that because of these parts of the image, the class "vertical astigmatism" was not predicted.

4.4.6 Conclusions on the results for classification

The regression task of estimation of aberrations with their amplitudes was reduced to the classification of 11 types of aberrations. The CNN was trained on the augmented data, and its training parameters and architecture were optimized. The decisions were explained with LIME XAI-method using SLIC superpixel algorithm working in 3D for image preprocessing (segmentation into reasonable clusters). The operating parameters of SLIC were adjusted to improve reliability with only a minor decrease in speed. The explanations of LIME were proven to be realistic.

The following conclusions can be made from this research:

1. Training on the enlarged dataset with randomized augmentations reached high accuracy on the raw test data.

2. Augmenting the training data by translation, brightness change, and contrast stretching was beneficial to the quality of predictions.

3. Among the tested noise types — Gaussian, Poisson, salt-and-pepper, and their combinations, — adding purely salt-and-pepper noise improved the test accuracy.

4. Clipping the pixel intensities to the range of [0, 1] after the percentile normalization on the training data was not beneficial to the quality of predictions.

5. Two dropout layers and batch normalization improved the model's accuracy.

6. The final activation function softmax produced better results than sigmoid.

7. The optimized CNN architecture, proposed data augmentation, and replacement of Gaussian noise with salt-and-pepper noise **may improve the work of PhaseNet** in the future.

8. **SLIC** with the described parameters **segments** the 3D grayscale images of beads **appropriately**.
9. **LIME is suitable** for the selected task: it highlights the input features relevant for classification according to human notion.
10. LIME explanations locate features with a positive as well as negative impact on the prediction, which also **aligns with the human perception** of aberration types.

The initial check on the classification task is thereby passed. The extendability of LIME to the initial multitarget 3D image regression is explored in the next chapter.

4.5 Results: Explainable Regression of Aberrations

The most straightforward way of explaining decisions of a multitarget regression network is to select a single target in the prediction vector and explain the value assigned to this target.

In the context of estimation of optical aberrations with PhaseNet [108], the prediction vector consists of 11 amplitudes for 11 Zernike aberration modes, the targets are the aberration modes, and the amplitudes (positive and negative floating point numbers) are the values to be explained. For the task of estimation of the aberrations, it is not meaningful to explain the whole prediction vector: it would be similar to explaining the whole prediction for all classes in a multilabel classification, where multiple classes are allowed to be assigned to a single image. In real applications, aberrations of different types appear independently from each other, and multiple aberration types can be present in a single image. Then it would be unclear which image features contribute to the prediction of which label. Because of that, only the value of a single aberration is explained in each experiment later in this chapter.

In this section, example explanations of the values of a single target are presented as heat maps with fixed colors, all answering the question, "Why was exactly this value predicted for the selected target?"

The explanation heat maps represent the influence of each part of the input image, segmented by SLIC, on the CNN's prediction. Positive values (yellow, orange, and red) are allocated to the segments supporting the prediction of a particular amplitude for the chosen target in the prediction vector, neutral values (green) are assigned to the segments not affecting the decision, and negative values (blue spectrum) correspond to the clusters arguing against the prediction of this value. The color scale indicates each segment's contribution on the heat map.

Informativeness across the z planes

In the examples shown in Figures 4.9, 4.10, and 4.11, PhaseNet significantly underestimated the positive amplitude. These explanations suggest that the CNN focused on the typical features of vertical astigmatism and its reasoning made sense. Such an underestimation of 0.025 μm could be caused by a discrepancy between the synthetic training data and real test data, shown in Figure 4.4.

The 3D explanation contains 32 planes in the z stack, three of which are shown in the examples: plane 2 (Figure 4.9), plane 16 in the middle (Figure 4.10), and plane 27 (Figure 4.11). The explanation maps differ across the z stack, providing the least informative explanations in the middle planes of the stack. This behavior has a natural cause: the point in the middle plane is in the optical focus, therefore the aberrations there are barely observable. In this particular example, plane 27 appeared to be more informative than plane 2. Plane 2 was chosen for consistency with the previous subsection. As described in Introduction (Subsection 1.4.1), optical aberrations occur when the light rays do not focus in one point. Considering the above, it was decided to visualize the performance of Image-Reg-LIME later in this chapter on the planes taken from either the beginning or the end of the z stack. Further results are demonstrated on planes 2 and 27 for consistency and a more objective comparison. Please note that not all color shades and boundaries between superpixels on the heatmaps may be visible due to compression of the document.

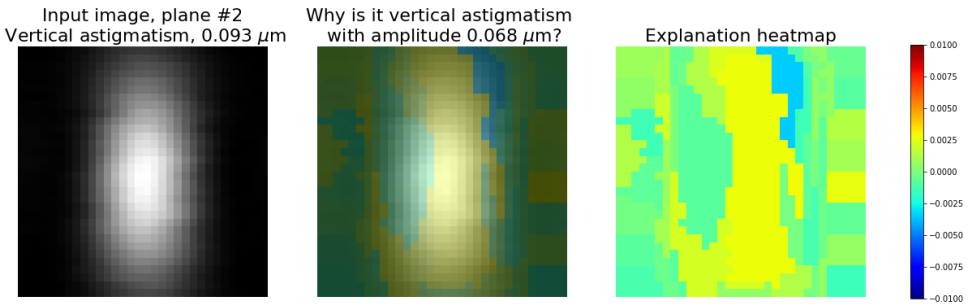

Figure 4.9: Image-Reg-LIME explanation for the value of a single target: the predicted amplitude of 0.068 μm for the vertical astigmatism aberration. The input image (left) is identical to the input in Figure 4.7 and contains an introduced aberration with an amplitude of 0.093 μm. The Image-Reg-LIME explanation answers the question, "Why was the amplitude of 0.068 μm predicted for the vertical astigmatism mode?" The image and explanation are three-dimensional, plane 2 is shown for both in 2D view.

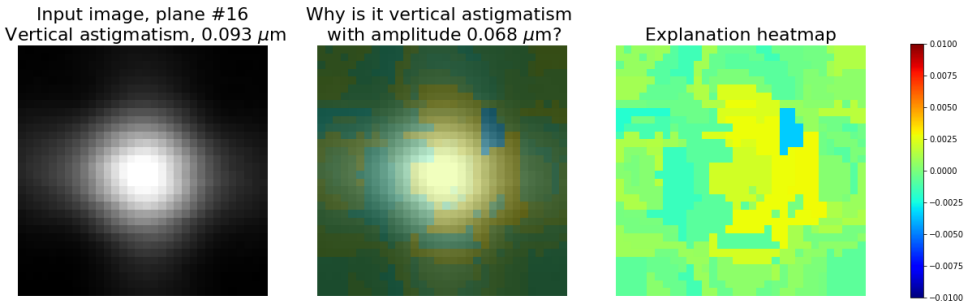

Figure 4.10: Image-Reg-LIME explanation for the value of a single target: the predicted amplitude of 0.068 μm for the vertical astigmatism aberration. The input image (left) contains an introduced aberration with an amplitude of 0.093 μm. The Image-Reg-LIME explanation answers the question, "Why was the amplitude of 0.068 μm predicted for the vertical astigmatism mode?" The image and explanation are three-dimensional, plane 16 is shown for both in 2D view. This middle plane shows the least informative explanation, compared to two other planes, as the image does not contain target-specific features.

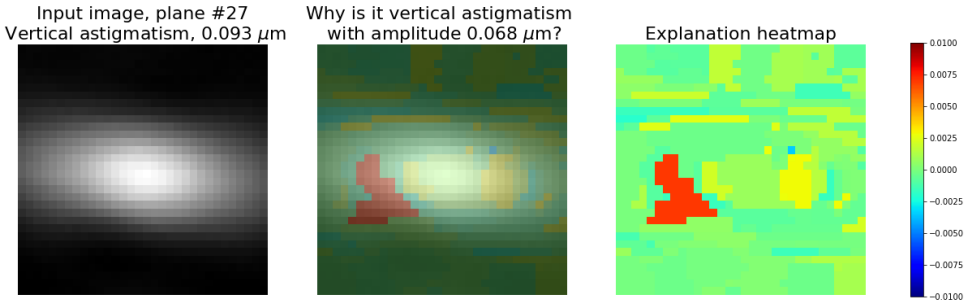

Figure 4.11: Image-Reg-LIME explanation for the value of a single target: the predicted amplitude of 0.068 μm for the vertical astigmatism aberration. The input image (left) contains an introduced aberration with an amplitude of 0.093 μm. The Image-Reg-LIME explanation answers the question, "Why was the amplitude of 0.068 μm predicted for the vertical astigmatism mode?" The image and explanation are three-dimensional, plane 27 is shown for both in 2D view. This plane from the end of the z stack contains a segment (in red) that has a strong impact on the prediction.

Residual aberrations

In Figure 4.11, the elliptic shape of the distortion is not exactly horizontal, and it can be presumed that the image is slightly affected by oblique astigmatism, which can be due to the residual PSF of the microscope containing this aberration. PhaseNet predicted the amplitude of 0.012 μm for oblique astigmatism. The explanation (Figure 4.12) shows that the top left part of the distortion was important to find slight oblique astigmatism.

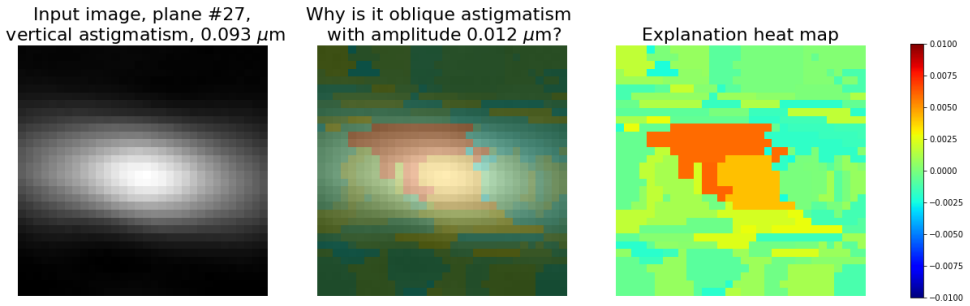

Figure 4.12: Image-Reg-LIME explanation for the value of a single target: the predicted amplitude of 0.012 μm for the **oblique astigmatism aberration that was not experimentally introduced**. The input image is identical to the input in Figure 4.11 with introduced vertical astigmatism. The Image-Reg-LIME explanation answers the question, "Why was the amplitude of 0.012 μm predicted for the oblique astigmatism mode?" The image and explanation are three-dimensional, plane 27 is shown for both in 2D view. This plane from the end of the z stack contains a segment (in red) that captures a portion of the oblique shift of the distortion and has a strong impact on the prediction.

To determine why PhaseNet significantly underestimated the amplitude for the image with vertical astigmatism (introduced value — 0.093 μm, predicted — 0.068 μm, see Figures 4.9, 4.10, and 4.11), an experiment with a reference value was conducted. As pointed out earlier, XAI methods are especially helpful to understand the reasoning behind wrong decisions.

4.5.1 Explanations with a reference value

To find out why the amplitude value was incorrectly predicted by PhaseNet, LIME was modified to answer questions with a reference value as described in Subsection 4.3.4 (Figure 4.2).

The result of asking Image-Reg-LIME, "Why did the model **not** predict the amplitude of 0.093 μm?" is shown in Figures 4.13 and 4.14 (planes 2 and 27 from the z stack). The positive impact (in the yellow-red spectrum) stands for the decision "**not 0.093** μm" and the segments with a negative impact (in blue) are **against** this decision, meaning that they are actually supporting the opposite decision of predicting **0.093** μm. The segments supporting the decision "**not 0.093** μm" outweighed those against it, therefore the final decision, dictated by the features with positive weights, was "**not 0.093** μm".

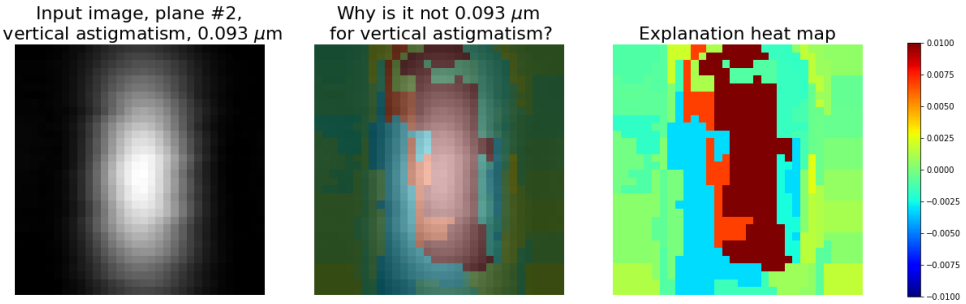

Figure 4.13: Image-Reg-LIME explanation for the ground truth value that was not predicted: 0.093 μm for vertical astigmatism. The input image (left) contains an introduced amplitude of 0.093 μm, but PhaseNet predicted 0.068 μm. The explanation answers the question, "Why did the model **not** predict the amplitude of 0.093 μm for the vertical astigmatism mode?" The segments with a positive impact (in the yellow-red spectrum) support the decision "**not 0.093** μm" and the segments with a negative impact (in blue) are **against** this decision. The latter effectively vote for 0.093 μm but are in the minority. Because of the highly impactful positive features, the decision "**not 0.093** μm" won. Hypothetically, the intensity of the input pixels in the red segments was insufficient (the larger the amplitude, the bigger and brighter the distortions are), and therefore a lower amplitude was predicted. Plane 2 from the z stack is shown.

Predictions on synthetic data

The reason for underestimation of the aberration amplitudes may be hidden in the training procedure: PhaseNet was trained on the synthetically generated images, which may not reflect the real conditions (Figure 4.4). To investigate this, the predictions made on synthetic images were explained with Image-Reg-LIME. The PhaseNet's predictions were accurate for small amplitudes, but starting from ± 0.070 μm, the absolute values were also underestimated.

To determine the characteristics that an image must possess to receive the prediction of an amplitude of 0.093 μm, synthetic images with varying ground truth amplitudes were generated. The image for which PhaseNet predicted an amplitude of 0.093 μm was selected for the analysis. The ground truth of the generated amplitude was 0.107 μm.

The Image-Reg-LIME explanations shown in Figures 4.15 and 4.16 answer the question without a reference value, "Why was the amplitude of 0.093 μm predicted for vertical astigmatism?" and demonstrate two 2D planes of the 3D explanation heat map. In Figure 4.16,

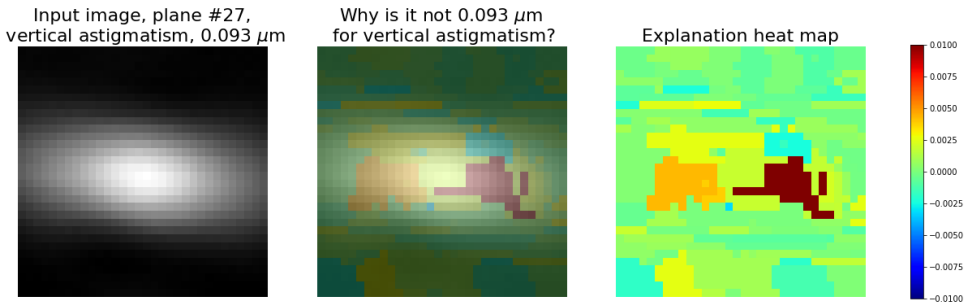

Figure 4.14: Image-Reg-LIME explanation for the ground truth value that was not predicted: 0.093 μm for vertical astigmatism. The input image (left) contains an introduced amplitude of 0.093 μm, but PhaseNet predicted 0.068 μm. The explanation answers the question, "Why did the model **not** predict the amplitude of 0.093 μm for the vertical astigmatism mode?" The segments with a positive impact (in the yellow-red spectrum) support the decision "**not 0.093** μm" and the segments with a negative impact (in blue) are **against** this decision. The features with a positive impact heavily prevail. Hypothetically, the intensity of the input pixels in the dark red and orange segments was too low, and therefore a lower amplitude was predicted. Plane 27 from the z stack is shown.

the highest positive impact was assigned to the pixels with a high pixel value, which align horizontally in the center of the image. The object looks like a rectangle with rounded corners. The real images contained a sphere whose pixel value decreased from the center to the periphery (Figure 4.14). This observation may indicate that PhaseNet has learned the aberration features typical for the synthetic dataset but absent in the set of the real data.

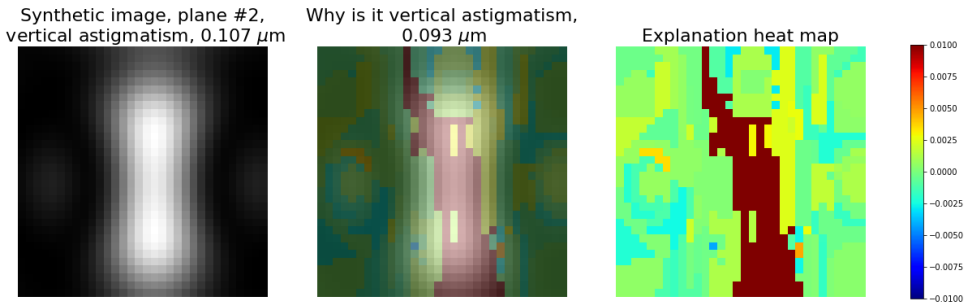

Figure 4.15: Image-Reg-LIME explanation of the prediction on the synthetic data: an amplitude of 0.093 μm was predicted for vertical astigmatism. The input image (left) contains a synthetic amplitude of 0.107 μm. The explanation answers the question without a reference value, "Why was the amplitude of 0.093 μm predicted for vertical astigmatism?" The high pixel value of the pixels in the bottom center and center of the image and the vertical shape of the object were important for this decision. 2D plane 2 is shown.

4.5.2 Validation of explanations

In the example of underestimation of the amplitude of vertical astigmatism (Figure 4.14), the amplitude of 0.068 μm was predicted instead of 0.093 μm. Presuming that the aberration features in the synthetic data can differ from those in the real data, it is interesting to search for these differences, and a straightforward way for this is subtracting a synthetic image with one amplitude from the real image.

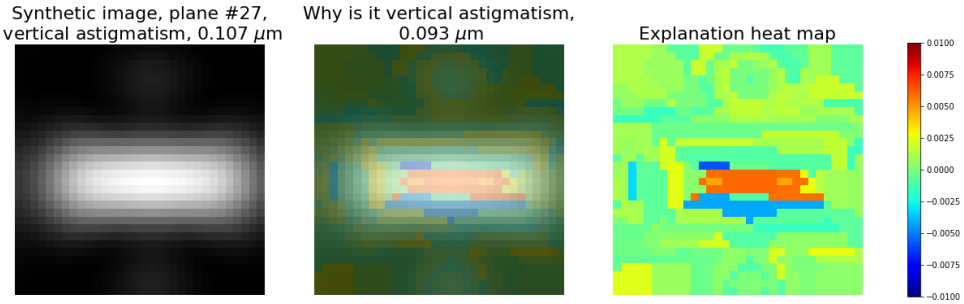

Figure 4.16: Image-Reg-LIME explanation of the prediction on the synthetic data: an amplitude of 0.093 μm was predicted for vertical astigmatism. The input image (left) contains a synthetic amplitude of 0.107 μm. The explanation answers the question without a reference value, "Why was the amplitude of 0.093 μm predicted for vertical astigmatism?" The constantly high pixel value of the pixels aligned horizontally in the center was very important for this decision. 2D plane 27 is shown.

Two synthetic images with aberration amplitudes of 0.068 and 0.093 μm and the result of subtraction of the second image from the first one are shown in Figure 4.17. The result is the difference in the pixel values which points to the parts of the input that should be important to differentiate these two amplitudes. These are the visual features that PhaseNet has learned from the synthetic training data to distinguish these two amplitudes. The CNN also observed in the training data that the images with an amplitude of 0.093 μm contain a rectangle-like object, whereas the synthetic image with an amplitude of 0.068 μm contains an elliptic object with a bright sphere in the middle.

The experimental image (Figure 4.14) with the ground truth amplitude of 0.093 μm for which PhaseNet predicted 0.068 μm looks more similar to the synthetic image in Figure 4.17B, with the predicted amplitude of 0.068 μm, than to the image in Figure 4.17A with that of 0.093 μm.

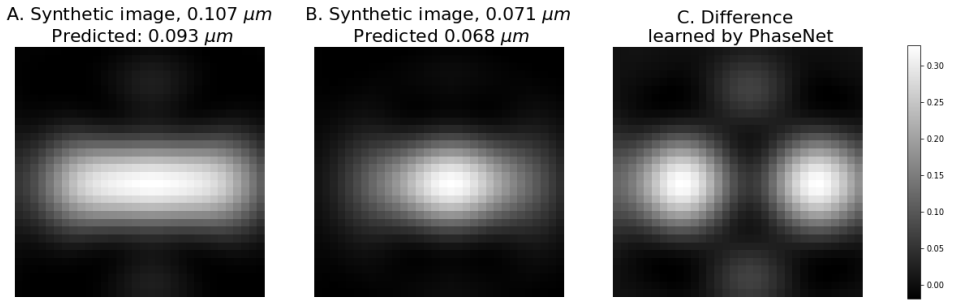

Figure 4.17: Difference between two synthetic images, plane 27. The synthetic images A and B are normalized, their difference A – B = C is not normalized to visualize the contrast. The scale shows the pixel value. In the right panel, zero difference in the center of the image indicates that the pixel value there is the same in both images. Bright spheres on the left and right demonstrate that the largest difference is at the ends of the object. Gray areas around the centers of the spheres indicate a gradual increase in the pixel value.

The network expects that vertical astigmatism with an amplitude of 0.093 μm has a certain appearance. On the other hand, the input image was not predicted to have an aberration amplitude of 0.093 μm. The discrepancy in the pixel values between the synthetic image with

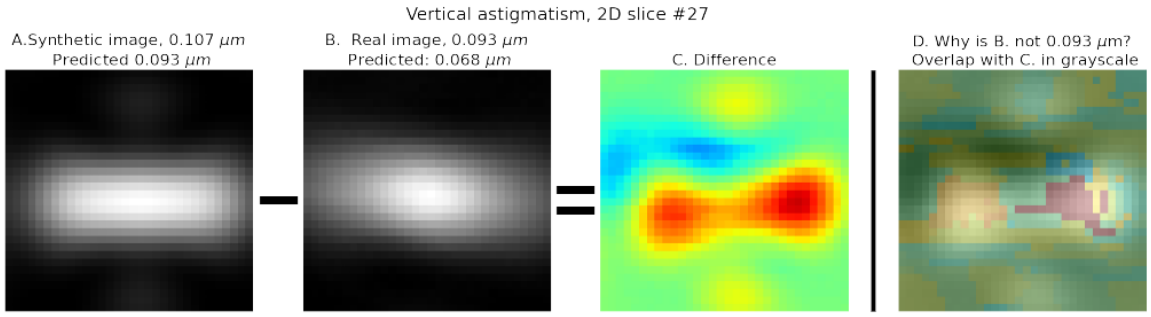

Figure 4.18: Difference between the synthetic and real images compared with the Image-Reg-LIME explanation, plane 27. Synthetic image A and real image B are normalized and the result of their subtraction A – B = C is visualized in color, where the negative difference is shown in blue (pixel value in A < pixel value in B), values around zero — in green, and positive difference — in yellow-red, with lower values in yellow, and larger ones in red. This color scheme is used for explanations throughout this chapter. Because the absolute values of the difference and explanation weights are not essential for the conclusions, they are omitted for space and simplicity. In image D, the explanation of why the amplitude of 0.093 μm was not predicted for the real image B is plotted on top of the difference C converted to grayscale. This shows agreement of the Image-Reg-LIME explanation with the actual discrepancy between the input features which lead to the desired prediction (0.093 μm) and the input that did not receive the desired prediction. This experiment confirms the rationality of the Image-Reg-LIME explanation.

the predicted amplitude of 0.093 μm and the real image with the ground truth amplitude of 0.093 μm was visualized to determine whether the Image-Reg-LIME explanations point to the differences between the network's expectation and the input image.

The synthetic image (Figure 4.18A) has the predicted aberration amplitude of 0.093 μm, the real image (Figure 4.18B) — the ground truth amplitude of 0.093 μm. The result of their subtraction (Figure 4.18C) points to the parts of the input that should be responsible for predicting the wrong amplitude for the real image. The red-orange barbell-like area in the middle with a yellow boundary shows the location where the pixel value in the synthetic image (which the CNN expects to possess the features of 0.093 μm amplitude) is larger than that in the real image.

To compare the localization of these differences with the explanation of why the amplitude of 0.093 μm was not predicted for this real image (Figures 4.14 and 4.18B), the explanation was mapped to the difference plot (Figure 4.18C) converted to grayscale. The mapping is shown in Figure 4.18D. Because only spatial localization and relative differences in the pixel values are necessary to validate the explanation of the network's reasoning, the absolute values of the differences and explanation weights are omitted for space and simplicity.

In Figure 4.18D, the areas with the largest positive weights in the explanation (orange and dark red segments near the center) overlap with the locations of the largest difference in Figure 4.18C (bright spots in Figure 4.18D). This means that, according to Image-Reg-LIME, the CNN did not predict the desired value exactly because of this discrepancy (Figure 4.18C) between the real input image (Figure 4.18B) and synthetic input image (Figure 4.18A), which the CNN needs for making the desired prediction of 0.093 μm for the selected target (vertical astigmatism) in the prediction vector.

This result proves that the explanations of Image-Reg-LIME highlight reasonable input features and make sense from the domain-specific perspective.

4.6 Conclusions

The research presented in this chapter proves that LIME is applicable to 3D image classification of optical aberration modes and Image-Reg-LIME - to multitarget 3D image regression of amplitudes of aberrations. For the latter task, Image-Reg-LIME with the modified prediction function can explain incorrect predictions by showing what was missing in the input image to make the prediction correct.

Both tasks were in 3D, and to enable LIME to work with 3D grayscale images, SLIC was chosen as the superpixel segmentation algorithm. The parameters of SLIC that lead to image partitioning suitable for LIME explanations are data-specific and must be carefully defined before applying LIME and Image-Reg-LIME.

The regression task was reduced to classification to investigate whether LIME can reasonably explain the differences in the aberration types. Data augmentation of the training set and the network architecture proposed for classification may be used to improve the results of PhaseNet and PhaseResNet investigated in Chapter 3. The explanations of LIME for classification agreed with the distinctive visual features of the aberration modes. Thereby, LIME, with SLIC as a segmentation algorithm, is suitable to explain the 3D image classification of aberrations.

Next, Image-Reg-LIME was used to explain the decisions of PhaseNet. The explanations of Image-Reg-LIME for multitarget 3D image regression of amplitudes of aberrations answering the question, "Why was the value a predicted for the regression target i?" agreed with the distinctive visual features of the aberration modes and their magnitudes.

A novel modification to LIME (the key feature of Image-Reg-LIME) was proposed — explanations with a reference value — that allows explaining why the value a^* (amplitude) was not predicted for the regression target i (aberration type) in the prediction vector of amplitudes for each aberration mode. The rationality of this novel type of explanations was proven by demonstrating that Image-Reg-LIME highlights the differences between the input image, for which a^* was not predicted, and another image, for which a^* was predicted. This is an important step in understanding the sources of inaccurate predictions.

Overall, LIME and Image-Reg-LIME are valid methods for explaining the classification and the estimation of optical aberrations in 3D images, respectively. To allow applicability to 3D images, SLIC should be used as the superpixel algorithm and its parameters should be defined specifically for the new data. To explain incorrect predictions of individual target values in the output of a regression network, Image-Reg-LIME can be supplied with a modified prediction function with a reference value. This may provide new insights into data biases or discrepancies between the training and test domains, and thereby help to improve construction of the training dataset.

Image-Reg-LIME has the potential to be used for other multitarget 3D image regression tasks in other application domains, which could be a scope of future work. The insights about the synthetic training data gained from Image-Reg-LIME and proposed data augmentations can be used to improve PhaseNet's and PhaseResNet's performance in the future.

5

Conclusions and Outlook

Answering the questions proposed in Introduction, I would like to share a view on the current state of explainable artificial intelligence (XAI) and make final conclusions about the conducted research in this broad context.

- Can artificial intelligence be explainable?

Yes, AI can be explainable. Moreover, any weak AI should become explainable one day, regardless of the task it was designed for. As the current XAI research concerns only weak AI, a prognosis for the explainability of artificial general intelligence cannot be made. However, the conversational AI ChatGPT, which had become one of the most discussed breakthroughs immediately after its release in November 2022, came as close as never before to what is defined as artificial general intelligence. Nevertheless, ChatGPT is still believed by the research community to belong to weak AI. To my question of whether it can explain its decisions, it replied positively. When I asked for advice on a casual topic and requested an explanation of why it gave me this advice, I expected a disclosure of information sources or some Internet statistics. Instead, ChatGPT argued like a human would. This explanation was indeed in the human-understandable domain, but did the AI explain its own decision, or did it learn how to argue with humans by "reading" similar conversations?

- What does it mean for artificial intelligence to be able to explain its decisions?

It means that such an AI model is explainable by design (see Subsection 1.2.2), and thereby its decisions are transparent to a certain degree. Normally, models of this kind are designed so that explanations are easy to retrieve and no additional tools are needed. But this does not necessarily mean that the model provides explanations along with predictions; instead, it waits for a human to ask for explanations.

- Can we explain AI's decisions and the underlying algorithms in detail?

We humans have the tools in our hands to explain the decisions and underlying algorithms of certain types of weak AI. The underlying AI algorithms can be explained in detail on the

theoretical level by providing an understanding of the architecture and optimization procedure (e.g., how convolutional layers process input images and how loss functions are optimized). On the practical level, **what** the network has learned can be explained using global XAI methods that provide advanced techniques for retrieving information learned by individual units during training. For explaining individual decisions made on specific data examples, local XAI methods are useful. A range of methods is already available for practitioners, but, in the context of computer vision, most of them are designed for image classifiers. As the research field moves forward, more types of AI models become explainable, and the quality of explanations increases. However, the notion of quality remains mostly subjective, or the defined quality metric is specific to a single problem and a single type of data, which complicates comparison of XAI methods.

In AI research in general, the lack of available training data is one of the major limitations. The number of AI publications about training techniques has been skyrocketing during the past decade, libraries for various programming languages offer tools for model design and even pretrained models, and necessary hardware for calculations is accessible through cloud providers. Nevertheless, the research of some AI applications is limited by data availability or quality. For example, in sensorless adaptive optics, research is only possible with microscopes with special sensors, constraining the scientists not having access to such devices to computational work on limited publicly available image data. Making more experimental data accessible would facilitate progress in solving the problems of computational estimation and correction of optical aberrations.

For XAI research, the lack of specific data — expert-labeled ground truth for explanations — is a limiting factor. Such ground truth would be invaluable for comparing XAI explanations with human decision-making in tasks where expertise in the topic is required to validate the plausibility of explanations.

An alternative to the expert-labeled ground truth for explanations is creation of a toy task where the features responsible for the decisions are obvious. For evaluation of XAI for image regression, such a toy task is fairly easy to design by training a model with a regression objective to count objects or to estimate their size, and then check if the explanation heat maps highlight the expected objects' locations.

As was proven in this thesis, the existing tools for explainable image classification are powerful enough to be extendable to other tasks, such as image segmentation and image regression. However, to realize such extension, the methods for explainable image classification require adaptation and understanding of the underlying explanation processes and of the data that they are applied to.

References

[1] *Neuchâtel Museum of Art and History. The Jaquet-Droz Automata.* https://www.mahn.ch/en/expositions/automates-jaquet-droz. Accessed: 2022-12-18.

[2] D. Crevier. *Ai: The Tumultuous History Of The Search For Artificial Intelligence*. Basic Books, 1993, p. 49. ISBN: 9780465029976. URL: https://books.google.de/books?id=QJNQAAAAMAAJ.

[3] A. M. TURING. "I.—COMPUTING MACHINERY AND INTELLIGENCE". In: *Mind* LIX.236 (Oct. 1950), pp. 433–460. ISSN: 0026-4423. DOI: 10.1093/mind/LIX.236.433.

[4] Henry Shevlin et al. "The limits of machine intelligence: Despite progress in machine intelligence, artificial general intelligence is still a major challenge". In: *EMBO reports* 20.10 (2019), e49177.

[5] David Silver et al. "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play". In: *Science* 362.6419 (2018), pp. 1140–1144. DOI: 10.1126/science.aar6404.

[6] *The AI companion who cares.* https://replika.com/. Accessed: 2022-12-22.

[7] Nick Bostrom. *Superintelligence: Paths, Dangers, Strategies*. Oxford University Press, 2014.

[8] A. L. Samuel. "Some Studies in Machine Learning Using the Game of Checkers". In: *IBM Journal of Research and Development* 3.3 (1959), pp. 210–229. DOI: 10.1147/rd.33.0210.

[9] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[10] J. Deng et al. "ImageNet: A Large-Scale Hierarchical Image Database". In: *CVPR09*. 2009.

[11] Simon Crase and Suresh N Thennadil. "An analysis framework for clustering algorithm selection with applications to spectroscopy". In: *Plos one* 17.3 (2022), e0266369.

[12] Nan Zhao et al. "Determining Effects of Non-synonymous SNPs on Protein-Protein Interactions using Supervised and Semi-supervised Learning". In: *PLoS computational biology* 10 (May 2014), e1003592. DOI: 10.1371/journal.pcbi.1003592.

[13] Khansa Rasheed et al. "Explainable, trustworthy, and ethical machine learning for healthcare: A survey". In: *Computers in Biology and Medicine* 149 (2022), p. 106043. ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2022.106043.

[14] Alun Preece et al. “Stakeholders in explainable AI”. In: *arXiv preprint arXiv:1810.00184* (2018).

[15] Alejandro Barredo Arrieta et al. “Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI”. In: *Information Fusion* 58 (2020), pp. 82–115. ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2019.12.012.

[16] Zohaib Salahuddin et al. “Transparency of deep neural networks for medical image analysis: A review of interpretability methods”. In: *Computers in biology and medicine* 140 (2022), p. 105111.

[17] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. “Explainable AI: A Review of Machine Learning Interpretability Methods”. In: *Entropy* 23.1 (2021). ISSN: 1099-4300. DOI: 10.3390/e23010018.

[18] Sherin Mary Mathews. “Explainable Artificial Intelligence Applications in NLP, Biomedical, and Malware Classification: A Literature Review”. In: *Intelligent Computing*. Ed. by Kohei Arai, Rahul Bhatia, and Supriya Kapoor. Cham: Springer International Publishing, 2019, pp. 1269–1292. ISBN: 978-3-030-22868-2.

[19] Amina Adadi and Mohammed Berrada. “Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)”. In: *IEEE Access* 6 (2018), pp. 52138–52160. DOI: 10.1109/ACCESS.2018.2870052.

[20] Alexandra Kirsch. “Explain to whom? Putting the User in the Center of Explainable AI”. In: *Proceedings of the First International Workshop on Comprehensibility and Explanation in AI and ML 2017 co-located with 16th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2017)*. Bari, Italy, 2017. URL: https://hal.archives-ouvertes.fr/hal-01845135.

[21] Nicholas Diakopoulos. “Accountability in Algorithmic Decision Making”. In: *Commun. ACM* 59.2 (Jan. 2016), pp. 56–62. ISSN: 0001-0782. DOI: 10.1145/2844110.

[22] Andreas Holzinger. “Explainable AI (ex-AI)”. In: *Informatik-Spektrum* 41 (Apr. 2018), pp. 1–6. DOI: 10.1007/s00287-018-1102-5.

[23] Simon Hoffmann. *LIME ein vielseitiges Erklärermodell - auch für Machine-Learning-Laien. Seminar KI: gestern, heute, morgen. Computing in the Humanities, Universität Bamberg*. https://cogsys.uni-bamberg.de/teaching/ws1718/sem_m2/Simon_Hoffmann_LIME.pdf. Accessed: 2022-12-24.

[24] U Schmid, S Scheele, and C Distler. *Erklärbares Maschinelles Lernen für Ingenieurwissenschaften*. https://learn.ki-campus.org/courses/erklaerbareki2020. 2020.

[25] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. “Methods for interpreting and understanding deep neural networks”. In: *Digital Signal Processing* 73 (2018), pp. 1–15. ISSN: 1051-2004. DOI: https://doi.org/10.1016/j.dsp.2017.10.011.

[26] Grégoire Montavon et al. “Explaining nonlinear classification decisions with deep Taylor decomposition”. In: *Pattern Recognition* 65 (2017), pp. 211–222. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2016.11.008.

[27] Iam Palatnik de Sousa, Marley M. B. R. Vellasco, and Eduardo Costa da Silva. "Explainable Artificial Intelligence for Bias Detection in COVID CT-Scan Classifiers". In: *Sensors* 21.16 (2021). ISSN: 1424-8220. DOI: 10.3390/s21165657.

[28] Kristina Preuer et al. "Interpretable Deep Learning in Drug Discovery". In: *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Ed. by Wojciech Samek et al. Cham: Springer International Publishing, 2019, pp. 331–345. ISBN: 978-3-030-28954-6. DOI: 10.1007/978-3-030-28954-6_18.

[29] Paul Jacob et al. "STEEX: steering counterfactual explanations with semantics". In: *European Conference on Computer Vision*. Springer. 2022, pp. 387–403.

[30] Surya Mattu Julia Angwin Jeff Larson and Lauren Kirchner. *Machine bias. There's software used across the country to predict future criminals. And it's biased against blacks.* https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing. Accessed: 2023-01-14. 2016.

[31] *§ 1002.9 Notifications. 12 CFR Part 1002 (Regulation B)*. https://www.consumerfinance.gov/rules-policy/regulations/1002/9. Accessed: 2023-01-14.

[32] Chris Olah et al. "Zoom In: An Introduction to Circuits". In: *Distill* 5 (Mar. 2020). DOI: 10.23915/distill.00024.001.

[33] Christoph Molnar. *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable*. 2nd ed. Accessed: 2023-01-15. 2022. URL: https://christophm.github.io/interpretable-ml-book.

[34] Muhammad Salar Khan et al. "Explainable AI: A Neurally-Inspired Decision Stack Framework". In: *Biomimetics* 7.3 (2022), p. 127.

[35] Masoud Aliramezani, Armin Norouzi, and Charles Koch. "A grey-box machine learning based model of an electrochemical gas sensor". In: *Sensors and Actuators B Chemical* 321 (Oct. 2020), p. 128414. DOI: 10.1016/j.snb.2020.128414.

[36] Emmanuel Pintelas, Ioannis E. Livieris, and Panagiotis Pintelas. "A Grey-Box Ensemble Model Exploiting Black-Box Accuracy and White-Box Intrinsic Interpretability". In: *Algorithms* 13.1 (2020). ISSN: 1999-4893. DOI: 10.3390/a13010017.

[37] Eduardo Almeida Soares. "Explainable-by-Design Deep Learning". PhD thesis. Lancaster University, 2022.

[38] Kira Vinogradova, Alexandr Dibrov, and Gene Myers. "Towards interpretable semantic segmentation via gradient-weighted class activation mapping (student abstract)". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 10. 2020, pp. 13943–13944.

[39] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. ""Why Should I Trust You?": Explaining the Predictions of Any Classifier". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. 2016, pp. 1135–1144.

[40] Matthew D Zeiler and Rob Fergus. "Visualizing and understanding convolutional networks". In: *European conference on computer vision*. Springer. 2014, pp. 818–833.

[41] Ruth C. Fong and Andrea Vedaldi. "Interpretable Explanations of Black Boxes by Meaningful Perturbation". In: *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), pp. 3449–3457.

[42] Bolei Zhou et al. "Learning deep features for discriminative localization". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2921–2929.

[43] Ramprasaath R Selvaraju et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 618–626.

[44] Sebastian Bach et al. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation". In: *PloS one* 10.7 (2015), e0130140.

[45] Uwe Schmidt et al. "Cell detection with star-convex polygons". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2018, pp. 265–273.

[46] Coleman Broaddus et al. "Removing structured noise with self-supervised blind-spot networks". In: *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*. IEEE. 2020, pp. 159–163.

[47] Luiz Galvao et al. "Pedestrian and Vehicle Detection in Autonomous Vehicle Perception Systems—A Review". In: *Sensors* 21 (Oct. 2021), p. 7267. DOI: 10.3390/s21217267.

[48] Sardar Waqar Khan et al. "Anomaly Detection in Traffic Surveillance Videos Using Deep Learning". In: *Sensors* 22.17 (2022). ISSN: 1424-8220. DOI: 10.3390/s22176563.

[49] Alexey M. Fartukov, Gleb A. Odinokikh, and Vitaly S. Gnatyuk. "Approaches and Methods to Iris Recognition for Mobile". In: *Smart Algorithms for Multimedia and Imaging*. Ed. by Michael N. Rychagov, Ekaterina V. Tolstaya, and Mikhail Y. Sirotenko. Cham: Springer International Publishing, 2021, pp. 397–422. ISBN: 978-3-030-66741-2. DOI: 10.1007/978-3-030-66741-2_16.

[50] *Image Classification on ImageNet Leaderboard*. https://paperswithcode.com/sota/image-classification-on-imagenet. Accessed: 2023-01-01.

[51] Jiahui Yu et al. *CoCa: Contrastive Captioners are Image-Text Foundation Models*. 2022. DOI: 10.48550/ARXIV.2205.01917.

[52] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[53] Christian Szegedy et al. "Rethinking the Inception Architecture for Computer Vision". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2818–2826. DOI: 10.1109/CVPR.2016.308.

[54] Mingxing Tan and Quoc Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, Sept. 2019, pp. 6105–6114. URL: https://proceedings.mlr.press/v97/tan19a.html.

[55] Jordi de la Torre, Domenec Puig, and Aida Valls. "Weighted kappa loss function for multi-class classification of ordinal data in deep learning". In: *Pattern Recognition Letters* 105 (2018). Machine Learning and Applications in Artificial Intelligence, pp. 144–154. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2017.05.018.

[56] Raphael Angulu, Jules R. Tapamo, and Aderemi O. Adewumi. "Age estimation via face images: a survey". In: *EURASIP Journal on Image and Video Processing* 2018.1 (June 2018). DOI: 10.1186/s13640-018-0278-6.

[57] Yao Xue et al. "Cell Counting by Regression Using Convolutional Neural Network". In: *Computer Vision – ECCV 2016 Workshops*. Ed. by Gang Hua and Hervé Jégou. Cham: Springer International Publishing, 2016, pp. 274–290. ISBN: 978-3-319-46604-0.

[58] Xiabi Liu et al. "3D head pose estimation with convolutional neural network trained on synthetic images". In: *2016 IEEE International Conference on Image Processing (ICIP)* (2016), pp. 1289–1293.

[59] Nyeong-Ho Shin, Seon-Ho Lee, and Chang-Su Kim. "Moving Window Regression: A Novel Approach to Ordinal Regression". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 18760–18769.

[60] Cho-Ying Wu, Qiangeng Xu, and Ulrich Neumann. "Synergy between 3dmm and 3d landmarks for accurate 3d facial geometry". In: *2021 International Conference on 3D Vision (3DV)*. IEEE. 2021, pp. 453–463.

[61] Marius Cordts et al. "The Cityscapes Dataset for Semantic Urban Scene Understanding". In: *29th IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)*. Las Vegas, NV, USA: IEEE Computer Society, 2016, 3213?3223. URL: https://publications.cispa.saarland/1846/.

[62] Pedro M.M. Pereira et al. "Dermoscopic skin lesion image segmentation based on Local Binary Pattern Clustering: Comparative study". In: *Biomedical Signal Processing and Control* 59 (2020), p. 101924. ISSN: 1746-8094. DOI: https://doi.org/10.1016/j.bspc.2020.101924.

[63] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Cham: Springer International Publishing, 2015, pp. 234–241. ISBN: 978-3-319-24574-4.

[64] Haotian Yan, Chuang Zhang, and Ming Wu. "Lawin transformer: Improving semantic segmentation transformer with multi-scale representations via large window attention". In: *arXiv preprint arXiv:2201.01615* (2022).

[65] James Wyant and Katherine Creath. "Basic Wavefront Aberration Theory for Optical Metrology". In: *Applied Optics and Optical Engineering* 11 (Jan. 1992).

[66] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Nature, 2022.

[67] Ryuji Matsuoka et al. "Evaluation of correction methods of chromatic aberration in digital camera images". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* I-3 (July 2012), pp. 49–55. DOI: 10.5194/isprsannals-I-3-49-2012.

[68] Eugene Hecht. *Optics*. 2017.

[69] Steven H Schwartz. *Geometrical and Visual Optics*. McGraw-Hill, 2013.

[70] *Convolution illustrated: longitudinal (XZ) central slice of a 3D image acquired by a fluorescence microscope.* `https://commons.wikimedia.org/wiki/File:Convolution_Illustrated_eng.png`. Licence: Public domain. Accessed: 2023-01-16.

[71] *Zernike-Polynome bis zur 4. Ordnung und ein Beispiel 6. Ordnung.* `https://commons.wikimedia.org/wiki/File:ZernikePolynome4.png`. Licence: Public domain. Accessed: 2022-12-23.

[72] F Zernike. "Beugungstheorie des schneidenver-fahrens und seiner verbesserten form, der phasenkontrastmethode". In: *physica* 1.7-12 (1934), pp. 689–704.

[73] Joshua Cook. *The Zernike Polynomials*. `https://github.com/joshuacook/zernike`. Dec. 2014.

[74] Robert J. Noll. "Zernike polynomials and atmospheric turbulence∗". In: *J. Opt. Soc. Am.* 66.3 (Mar. 1976), pp. 207–211. DOI: `10.1364/JOSA.66.000207`.

[75] Larry N Thibos et al. "Standards for Reporting the Optical Aberrations of Eyes". In: *Journal of Refractive Surgery* 18.5 (2002), S652–S660. DOI: `10.3928/1081-597X-20020901-30`.

[76] Sourangsu Banerji et al. "Imaging with flat optics: Metalenses or diffractive lenses?" In: *Optica* 6 (June 2019), p. 805. DOI: `10.1364/OPTICA.6.000805`.

[77] Nahian Siddique et al. "U-net and its variants for medical image segmentation: A review of theory and applications". In: *Ieee Access* 9 (2021), pp. 82031–82057.

[78] Adrianna Janik, Kris Sankaran, and Anthony Ortiz. "Interpreting black-box semantic segmentation models in remote sensing applications". In: (2019).

[79] Lukas Hoyer et al. "Grid saliency for context explanations of semantic segmentation". In: *Advances in neural information processing systems* 32 (2019).

[80] Max Losch, Mario Fritz, and Bernt Schiele. "Semantic bottlenecks: Quantifying and improving inspectability of deep representations". In: *International Journal of Computer Vision* 129 (2021), pp. 3136–3153.

[81] Teddy Koker et al. "U-Noise: Learnable Noise Masks for Interpretable Image Segmentation". In: *arXiv preprint arXiv:2101.05791v3* (2021).

[82] Awadelrahman MA Ahmed and Leen AM Ali. "Explainable Medical Image Segmentation via Generative Adversarial Networks and Layer-wise Relevance Propagation". In: *Nordic Machine Intelligence* 1.1 (2021), pp. 20–22.

[83] Deepa Darshini Gunashekar et al. "Explainable AI for CNN-based prostate tumor segmentation in multi-parametric MRI correlated to whole mount histopathology". In: *Radiation Oncology* 17.1 (2022), pp. 1–10.

[84] Longxi Zhou et al. "An interpretable deep learning workflow for discovering subvisual abnormalities in CT scans of COVID-19 inpatients and survivors". In: *Nature Machine Intelligence* 4.5 (2022), pp. 494–503.

[85] Jixiang Lei. “Interpretation of Semantic Urban Scene Segmentation for Autonomous Vehicles/Author Jixiang Lei”. In: (2022).

[86] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. “Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps”. In: *CoRR* abs/1312.6034 (2013).

[87] Daniel Smilkov et al. “Smoothgrad: removing noise by adding noise”. In: *arXiv preprint arXiv:1706.03825* (2017). Presented at Workshop on Visualization for Deep Learning, ICML.

[88] Aditya Chattopadhay et al. “Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks”. In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (Mar. 2018). DOI: 10.1109/wacv.2018.00097.

[89] Lennart Schmidt et al. *Spatially-distributed Deep Learning for rainfall-runoff modelling and system understanding*. Tech. rep. Copernicus Meetings, May 2020, 20736, p. 20736. DOI: 10.5194/egusphere-egu2020-20736.

[90] Christina Humer et al. “Interactive attribution-based explanations for image segmentation”. In: *Johannes Kepler University Linz: Linz, Austria* (2022).

[91] David Melching et al. “Explainable machine learning for precise fatigue crack tip detection”. In: *Scientific Reports* 12.1 (2022), p. 9513.

[92] Olga Russakovsky et al. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[93] Dan Ciresan et al. “Deep neural networks segment neuronal membranes in electron microscopy images”. In: *Advances in neural information processing systems* 25 (2012).

[94] *CSBDeep - a deep learning toolbox for microscopy image restoration and analysis*. URL: https://csbdeep.bioimagecomputing.com/.

[95] Pavel Iakubovskii. *Segmentation Models*. https://github.com/qubvel/segmentation_models. 2019.

[96] *GridSaliency-ToyDatasetGen - the repository that provides code to generate the toy datasets used in "Grid Saliency for Context Explanations of Semantic Segmentation"*. URL: https://github.com/boschresearch/GridSaliency-ToyDatasetGen.

[97] Li Deng. “The mnist database of handwritten digit images for machine learning research”. In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142.

[98] *SciPy library, interpolation function 'zoom'*. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.ndimage.zoom.html.

[99] Stéfan van der Walt et al. “scikit-image: image processing in Python”. In: *PeerJ* 2 (June 2014), e453. ISSN: 2167-8359. DOI: 10.7717/peerj.453.

[100] Ade Irma Suryani et al. “Lung Tumor Localization and Visualization in Chest X-Ray Images Using Deep Fusion Network and Class Activation Mapping”. In: *IEEE Access* 10 (2022), pp. 124448–124463.

[101] Jia Wu et al. “BA-GCA net: boundary-aware grid contextual attention net in osteosarcoma MRI image segmentation”. In: *Computational Intelligence and Neuroscience* 2022 (2022).

[102] Andreas M Kist et al. “A single latent channel is sufficient for biomedical glottis segmentation”. In: *Scientific Reports* 12.1 (2022), p. 14292.

[103] Junbong Jang et al. “A deep learning-based segmentation pipeline for profiling cellular morphodynamics using multiple types of live cell microscopy”. In: *Cell reports methods* 1.7 (2021), p. 100105.

[104] Yi Zhou et al. “Slot-vps: Object-centric representation learning for video panoptic segmentation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 3093–3103.

[105] Indu Joshi et al. “Sensor-invariant fingerprint roi segmentation using recurrent adversarial learning”. In: *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2021, pp. 1–8.

[106] Indu Joshi et al. “Data uncertainty guided noise-aware preprocessing of fingerprints”. In: *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2021, pp. 1–8.

[107] Yunlong Zhang et al. “Harmonizing Pathological and Normal Pixels for Pseudo-Healthy Synthesis”. In: *IEEE Transactions on Medical Imaging* 41.9 (2022), pp. 2457–2468.

[108] Debayan Saha et al. “Practical sensorless aberration estimation for 3D microscopy with deep learning”. In: *Optics express* 28.20 (2020), pp. 29044–29053.

[109] Kira Vinogradova and Eugene W Myers. “Estimation of Optical Aberrations in 3D Microscopic Bioimages”. In: *2022 7th International Conference on Frontiers of Signal Processing (ICFSP)*. IEEE. 2022, pp. 97–103.

[110] H. W. Babcock. “The possibility of compensating astronomical seeing”. In: *Publications of the Astronomical Society of the Pacific* 65.386 (Oct. 1953), p. 229. DOI: 10.1086/126606.

[111] F Fischer and H Thiemann. “Theoretische Betrachtungen über ein neues Verfahren der Fernsehgrossprojektion [Theoretical considerations on a new method of large-screen television projection]”. In: *Schweiz. Archiv Angew. Wiss. Technik* 8 (1942), pp. 135–143.

[112] Johannes Hartmann. *Objektivuntersuchungen*. Springer, 1904.

[113] Junzhong Liang et al. “Objective measurement of wave aberrations of the human eye with the use of a Hartmann–Shack wave-front sensor”. In: *JOSA A* 11.7 (1994), pp. 1949–1957.

[114] Hai Gong et al. “Optical path difference microscopy with a Shack–Hartmann wavefront sensor”. In: *Opt. Lett.* 42.11 (June 2017), pp. 2122–2125. DOI: 10.1364/OL.42.002122.

[115] Sophia Imperato et al. “Single-shot quantitative aberration and scattering length measurements in mouse brain tissues using an extended-source Shack-Hartmann wavefront sensor”. In: *Opt. Express* 30.9 (Apr. 2022), pp. 15250–15265. DOI: 10.1364/OE.456651.

[116] Paolo Pozzi et al. “High speed wavefront sensorless aberration correction in digital micromirror based confocal microscopy”. In: *Opt. Express* 25 (Jan. 2017), pp. 949–959. DOI: 10.1364/OE.25.000949.

[117] Hanley et al. “An optical sectioning programmable array microscope implemented with a digital micromirror device”. In: *Journal of Microscopy* 196.3 (1999), pp. 317–331. DOI: https://doi.org/10.1046/j.1365-2818.1999.00602.x.

[118] Jacopo Antonello et al. *Interferometric calibration of a deformable mirror*. Mar. 2020. DOI: 10.5281/zenodo.3714951.

[119] Ivan Vishniakou and Johannes D Seelig. “Differentiable model-based adaptive optics for two-photon microscopy”. In: *Optics Express* 29.14 (2021), pp. 21418–21427.

[120] L. Murray et al. “Wavefront correction of extended objects through image sharpness maximisation - art. no. 60181A”. In: *Proceedings of SPIE - The International Society for Optical Engineering* 6018 (Dec. 2005). DOI: 10.1117/12.669380.

[121] Allen Jong-Woei Whang et al. “Zernike coefficient prediction technique for interference based on generation adversarial network”. In: *Applied Sciences* 11.15 (2021), p. 6933.

[122] Minzhao Liu, David N. Lopez, and Gabriel C. Spalding. “Experimental implementation of wavefront sensorless real-time adaptive optics aberration correction control loop with a neural network”. In: *Emerging Topics in Artificial Intelligence 2020*. Ed. by Giovanni Volpe et al. Vol. 11469. International Society for Optics and Photonics. SPIE, 2020, 114691S. DOI: 10.1117/12.2569647.

[123] Qinghua Tian et al. “DNN-based aberration correction in a wavefront sensorless adaptive optics system”. In: *Opt. Express* 27.8 (Apr. 2019), pp. 10765–10776. DOI: 10.1364/OE.27.010765.

[124] Qi Hu, Martin Hailstone, and Martin J Booth. “Efficient and versatile aberration correction through sensorless adaptive optics”. In: *Three-Dimensional and Multidimensional Microscopy: Image Acquisition and Processing XXIX*. Vol. 11966. SPIE. 2022, pp. 21–26.

[125] Qi Hu et al. “Universal adaptive optics for microscopy through embedded neural network control”. In: *arXiv preprint arXiv:2301.02647* (2023).

[126] Eduard Durech et al. “Wavefront sensor-less adaptive optics using deep reinforcement learning”. In: *Biomed. Opt. Express* 12.9 (Sept. 2021), pp. 5423–5438. DOI: 10.1364/BOE.427970.

[127] Y. Lecun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. DOI: 10.1109/5.726791.

[128] Johannes Schindelin et al. “Fiji: an open-source platform for biological-image analysis”. In: *Nature methods* 9.7 (2012), pp. 676–682.

[129] Eric Czech et al. “Cytokit: A single-cell analysis toolkit for high dimensional fluorescent microscopy imaging”. In: *bioRxiv* (2018). DOI: 10.1101/460980.

[130] William Hadley Richardson. “Bayesian-based iterative method of image restoration”. In: *JoSA* 62.1 (1972), pp. 55–59.

[131] Leon B Lucy. "An iterative technique for the rectification of observed distributions". In: *The astronomical journal* 79 (1974), p. 745.

[132] Daniel Sage et al. "DeconvolutionLab2: An open-source software for deconvolution microscopy". In: *Methods* 115 (2017), pp. 28–41.

[133] Eric A Bushong, Maryann E Martone, and Mark H Ellisman. "Maturation of astrocyte morphology and the establishment of astrocyte domains during postnatal hippocampal development". In: *International Journal of Developmental Neuroscience* 22.2 (2004), pp. 73–86.

[134] Bushong, Eric and Martone, Maryann and Ellisman, Mark. *CCDB:1065, rattus norvegicus, protoplasmic astrocyte. CIL. Dataset.* Accessed: 2022-20-07. 2002. URL: https://doi.org/doi:10.7295/W9CCDB1065.

[135] Bushong, Eric and Martone, Maryann and Ellisman, Mark. *CCDB:1073, rattus norvegicus, protoplasmic astrocyte. CIL. Dataset.* Accessed: 2022-20-07. 2002. URL: https://doi.org/doi:10.7295/W9CCDB1073.

[136] Pavel V Belichenko and Annica Dahlström. "Studies on the 3-dimensional architecture of dendritic spines and varicosities in human cortex by confocal laser scanning microscopy and Lucifer yellow microinjections". In: *Journal of neuroscience methods* 57.1 (1995), pp. 55–61.

[137] J. H. Sang. *In Reeve E.C. (ed.). Encyclopedia of Genetics*. Fitzroy Dearborn, 2001, pp. 155–157. ISBN: 9781884964343. URL: https://books.google.de/books?id=JjLWYKqehRsC.

[138] Adam Martin et al. "Integration of contractile forces during tissue invagination". In: *The Journal of cell biology* 188 (Mar. 2010), pp. 735–49. DOI: 10.1083/jcb.200910099.

[139] Ziwei Li et al. "Fast widefield imaging of neuronal structure and function with optical sectioning in vivo". In: *Science Advances* 6.19 (2020), eaaz3870. DOI: 10.1126/sciadv.aaz3870.

[140] Guoping Feng et al. "Imaging Neuronal Subsets in Transgenic Mice Expressing Multiple Spectral Variants of GFP". In: *Neuron* 28.1 (2000), pp. 41–51. ISSN: 0896-6273. DOI: https://doi.org/10.1016/S0896-6273(00)00084-2.

[141] Sjoerd Stallinga and Bernd Rieger. "Accuracy of the Gaussian Point Spread Function model in 2D localization microscopy". In: *Opt. Express* 18.24 (Nov. 2010), pp. 24461–24476. DOI: 10.1364/OE.18.024461.

[142] Bo Zhang, Josiane Zerubia, and Jean-Christophe Olivo-Marin. "A study of Gaussian approximations of fluorescence microscopy PSF models". In: *Three-Dimensional and Multidimensional Microscopy: Image Acquisition and Processing XIII*. Ed. by Jose-Angel Conchello, Carol J. Cogswell, and Tony Wilson. Vol. 6090. International Society for Optics and Photonics. SPIE, 2006, 60900K. DOI: 10.1117/12.645650.

[143] Kira Vinogradova and Gene Myers. "Local Interpretable Model-Agnostic Explanations for Multitarget Image Regression". In: *Joint Proceedings of the xAI-2023 Late-breaking Work, Demos and Doctoral Consortium co-located with the 1st World Conference on eXplainable Artificial Intelligence (xAI-2023)*. Vol. 3554. CEUR Workshop Proceedings (CEUR-WS.org), 2023, pp. 47–52.

[144] Jinlai Zhang et al. *Adversarial samples for deep monocular 6D object pose estimation.* 2022. arXiv: 2203.00302 [cs.CV].

[145] Kyriaki-Margarita Bintsi et al. "Voxel-Level Importance Maps for Interpretable Brain Age Estimation". In: *Interpretability of Machine Intelligence in Medical Image Computing, and Topological Data Analysis and Its Applications for Medical Data: 4th International Workshop, IMIMIC 2021, and 1st International Workshop, TDA4MedicalData 2021, Held in Conjunction with MICCAI 2021, Strasbourg, France, September 27, 2021, Proceedings*. Strasbourg, France: Springer-Verlag, 2021, pp. 65–74. ISBN: 978-3-030-87443-8. DOI: 10.1007/978-3-030-87444-5_7.

[146] Cher Bass et al. "Icam-reg: Interpretable classification and regression with feature attribution for mapping neurological phenotypes in individual scans". In: *IEEE Transactions on Medical Imaging* (2022).

[147] Cher Bass et al. "ICAM: Interpretable Classification via Disentangled Representations and Feature Attribution Mapping". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 7697–7709. URL: https://proceedings.neurips.cc/paper/2020/file/56f9f88906aebf4ad985aaec7fa01313-Paper.pdf.

[148] Andrea Vedaldi and Stefano Soatto. "Quick Shift and Kernel Methods for Mode Seeking". In: *Computer Vision – ECCV 2008*. Ed. by David Forsyth, Philip Torr, and Andrew Zisserman. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 705–718. ISBN: 978-3-540-88693-8.

[149] CIE Colorimetry. "Report No: CIE Pub No 15". In: *Vienna: CIE Central Bureau* (2004).

[150] Radhakrishna Achanta et al. "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.11 (2012), pp. 2274–2282. DOI: 10.1109/TPAMI.2012.120.

[151] Connor Shorten and Taghi M. Khoshgoftaar. "A survey on Image Data Augmentation for Deep Learning". In: *Journal of Big Data* 6 (2019).