FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Assignment of master's thesis

Title:	Explainability in deep learning-based medical image analysis
Student:	Bc. Martin Lank
Supervisor:	Ing. Magda Friedjungová, Ph.D.
Study program:	Informatics
Branch / specialization:	Knowledge Engineering
Department:	Department of Applied Mathematics
Validity:	until the end of summer semester 2023/2024

Instructions

Mean diffusivity (MD) and fractional anisotropy (FA) obtained with diffusion magnetic resonance imaging (dMRI) have been associated with cell density and tissue anisotropy across tumors, but it is unknown whether these associations persist at the microscopic level. The MD/FA can be predicted from histology patches using convolutional neural networks [1]. However, neural networks are often used as black-boxes. The aim of this thesis is to apply methods of explainable artificial intelligence in deep learning-based medical image analysis focused on application in magnetic resonance imaging (MRI) and histology.

1. Survey common and state-of-the-art methods of explainable artificial intelligence used in deep learning-based medical image analysis [2]. Focus on application using MRI and histology.
2. Get familiar with data provided by supervisor. Using these data, design and train neural network to predict MD/FA from histology patches.
3. Implement or use existing at least three surveyed algorithms from Step 1 and perform them using network from Step 2.
4. Present (i.e. visualize) and discuss results from each algorithm. Choose and discuss the best method in sense of interpretability, suitable for used dataset.

References

[1] https://www.biorxiv.org/content/10.1101/2022.12.20.521068v1
[2] https://www.sciencedirect.com/science/article/pii/S1361841522001177

Electronically approved by Ing. Karel Klouda, Ph.D. on 6 February 2023 in Prague.

Czech Technical University in Prague

Faculty of Information Technology

Department of Applied Mathematics

Master's thesis

Explainability in Deep Learning-based Medical Image Analysis

Bc. Martin Lank

Supervisor: Ing. Magda Friedjungová, Ph.D.

January 9, 2024

Acknowledgements

I would like to express deep gratitude to my thesis supervisor, Ing. Magda Friedjungová, Ph.D., for her guidance, fast reactions to my thoughts, and sometimes even for her motivation. It couldn't be better. Many thanks also go to MUDr. Jan Brabec, MSc., Ph.D., for the discussions about the results.

Just as important was the support of my close family. I am so grateful for my parents, who never questioned my skills, gave me opportunities and supported me all the way through my life up to this last stage of my study era. Thank you!

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60(1) of the Act.

In Prague on January 9, 2024 .....

Czech Technical University in Prague

Faculty of Information Technology

Citation of this thesis

LANK, Martin. *Explainability in Deep Learning-based Medical Image Analysis*. Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2024.

Abstract

In this work, we apply Grad-CAM++, LayerCAM and SmoothGrad explainability methods to the proposed EfficientNetV2-based convolutional neural network fine-tuned on microscopic histology imaging. The network predicts mean diffusivity (MD) and fractional anisotropy (FA) obtained from diffusion tensor imaging. The aim of the work was to reveal which histology features tend to increase MD and FA. The proposed network achieved more than 98.5% R^2 on all train, validation and test sets, surpassing the network proposed in the preceding work by tens of percentage points in R^2. Nevertheless, the explainability methods applied to microscopy imaging were less valuable than anticipated. They indicate certain nuclei influence; however, the details about the relationship remain undiscovered.

Keywords explainability, convolutional neural networks, medical imaging, histology imaging, microscopic imaging, mean diffusivity, fractional anisotropy

Abstrakt

V této práci aplikujeme metody vysvětlitelnosti Grad-CAM++, LayerCAM a SmoothGrad na konvoluční neuronovou síť založenou na EfficicientNetV2 a doučenou na mikroskopických histologických snímcích. Tato neuronová síť predikuje průměrnou difuzivitu (MD) a frakční anizotropii (FA), původně získanou technikou difuzního tenzorového zobrazování (DTI). Cílem této práce je odhalit, které histologické vlastnosti mají vliv na zvýšení MD a FA. Naše síť dosahuje více než 98.5 % R^2 na trénovací, validační i testovací množině, čímž o desítky procentních bodů překonává síť navrženou v předešlé práci. Aplikované metody vysvětlitelnosti na mikroskopické snímky se ukázaly být méně užitečné, než jsme předpokládali. Sice naznačují určitý vliv buněčných jader, nicméně detaily tohoto vztahu zůstávají i nadále nejasné.

Klíčová slova vysvětlitelnost, konvoluční neuronové sítě, medicínské snímky, histologické snímky, mikroskopické snímky, průměrná difuzivita, frakční anizotropie

Contents

List of Tables

List of Figures

Introduction

Explainability in machine learning (ML) has been an active area of research and continues to gain popularity due to its necessity and importance. The field, referred to as explainable artificial intelligence (XAI), focuses on increasing ML models' trustworthiness by understanding the behaviour, gaining more confidence in their predictions, comprehending decisions and detecting biases that may have been unnoticed in the training data. The level of this knowledge of a model can play a crucial role in the selection of a suitable ML approach for a given application. There are ML methods that are well interpretable by design and thus explainable, such as linear regression or decision tree-based algorithms. However, nowadays, the state-of-the-art (SOTA) approaches across fields are typically based on an artificial neural network (ANN), which are poorly interpretable compared to the traditional ML methods [1]. Consequently, the lack of transparency makes it difficult to explain specific decisions. Still, these so-called "black box" models are heavily used in production and businesses, and people rely more on them daily than ever. Sadly, numerous examples can be found when such models failed horribly, as they were unsafe and biased towards sex and race [2, 3, 4]. The risk could have been reduced if not avoided, provided the creators could comprehend the models' behaviour and detect issues at first glance.

Nevertheless, there are still areas where the explainability of ML models outputs are critical. Thus, interpretable models are preferred over black box models even when they achieve worse results. One example for all is the

medical field, where doctors cannot prescribe a treatment based on a model result without justification for ethical and legal reasons. One of them is the "right to explanation", as stated in the EU regulation known as the General Data Protection Regulation (GDPR) [5]. It enables clients to require an explanation of how a given conclusion was approached. That could involve questions about the choice of certain medications over another or why a mortgage application was denied. It makes complete sense, yet regrettably, we do not utilize the full potential of SOTA approaches.

This thesis builds on the research conducted by Brabec et al. [6], which used magnetic resonance imaging (MRI), one of the most heavily used medical technologies. MRI non-invasively probes tissue, offering doctors insights into tissue structure and organisation at various scales [7]. This is possible thanks to the presence of water in our bodies. When exposed to short electromagnetic pulses, the protons in water molecules are magnetised in the direction of the magnetic field. As they gradually return to their equilibrium state, electromagnetic signals are emitted. This signal is detected and used to create the MRI image. By analysing the signal attenuation caused by the Brownian motion of water molecules, we can infer the diffusion of water molecules in our system. This is particularly useful for gaining information about areas with restricted diffusion, such as within the brain's white matter or tumours. A specific technique in diffusion MRI, known as diffusion tensor imaging (DTI), provides a 3D tensor that describes the full diffusion process. From this tensor, we can derive two key measurements: mean diffusivity (MD), which describes the average diffusion in all directions, and fractional anisotropy (FA), which represents the degree of anisotropy of the diffusion process [8].

Brabec et al. [6] studied the relationship between histological features and intra-tumour MD and FA in meningioma tumours. The study was conducted on 16 excessively collected tumour samples. Each sample was scanned on the DTI to get the ground truth MD and in-plane FA, which were later predicted using two methods. The features used for prediction were based on histology images taken with a light microscope. In the first approach, they calculated cell density (CD) and image anisotropy (IA) features from histology patches and performed polynomial regression. In the second approach, they trained a convolutional

neural network (CNN) directly on the histology patches. The CNN used pre-trained EfficientNetV2 [9] as a backbone and was fine-tuned individually for each sample with the mean square error (MSE) loss function. In both cases, the coefficient of determination (R^2) metric was used to analyze the results. Also, residual maps were generated between ground truth and the predicted MD.

They discovered an across-tumours relationship between CD and MD ($R^2 = 0.58$) and between IA and FA ($R^2 = 0.82$). Also, CNN performed systematically better than the regression method. The comparison of residual maps suggests that CNN learned new features that account for MD and FA changes.

This work uses the same data as Brabec et al. [10] in the study mentioned above[a]. The work aims to uncover the additional learned features by CNN. Considering the size of the trained model with over 117M trainable parameters, we utilise several SOTA methods from the XAI field. We start with training a suitable CNN model, and then we utilise three XAI post-hoc methods.

Structure of the Thesis

The thesis is organised as follows. In Chapter 1, we introduce the XAI concepts, terminology, and XAI methods used in this work. Chapter 2 defines the task and describes the methodology, including the dataset origin and preprocessing. Chapter 3 covers the design of experiments and implemented approaches. Chapter 4 presents and discusses the results. Finally, in the Conclusion Chapter, we review our contributions and propose directions for future work.

Further reading of this work assumes substantial knowledge of the machine learning concepts, which are not described here. We refer to other literature to get more familiar with the problematics, such as the book by Murphy [11].

[a] https://github.com/jan-brabec/microimaging_vs_histology_in_meningeomas

Chapter 1

Explainable AI

Traditionally, there has always been a trade-off between models with high interpretability but low accuracy and models with low interpretability but high accuracy. The goal of XAI is to make the models more understandable and, ideally, without decreasing the models' accuracy. As we can tell, it is a very broad field, which is partly the reason that there is no universally accepted definition [12]. However, one of the widely adopted, receiver-focused definitions of XAI was proposed by Arrieta et al. [13]. According to it, XAI is "one that for a given audience produces details or reasons to make its functioning clear or easy to understand". The part regarding the target audience is important. What is understandable for developers will undoubtedly be different to end-users or lawmakers.

1.1 Explainability vs Interpretability

In this Section, we would like to clarify the terms explainability and interpretability in the context of artificial intelligence (AI), as they are often misused and interchanged. We adopt the definition from the recent (8/2023) outstanding work conducted by S. Ali et al. [12], in which they surveyed and reviewed 410 articles to accurately reflect the current methodologies, directions and approaches.

Explainability refers to "the process of elucidating or revealing the decision-making mechanisms of models". It is connected with the ability to comprehend and explain to users *why* given AI model made certain decisions. It gives

humans the ability to "interpret and describe the inner workings of an AI system".

On the other hand, interpretability "enables developers to delve into the model's decision-making process, boosting their confidence in understanding where the model gets its results". Opposed to explainability, this is related to the ability to comprehend *how* given AI model makes decisions. It is a characteristic of a model. The term is more common among the ML community.

Generally, models with high interpretability (white box) are also well explainable. Those include, e.g. linear/logistic regression, decision trees and k-means. Conversely, ensemble methods, ANNs, and transformers lack interpretability due to their complexity and require additional explainability. That is called post-hoc explainability, as further explained in the next Section. However, as the authors claim, the explainability can be used anytime during development.

1.2 XAI Taxonomy

Various taxonomies have been proposed to date. The most recent XAI taxonomy was also proposed by Ali et al. [12], where they formed four categories.

(i) Scoop-based, analysing the feature importance to determine how model inputs affect the outputs. It is further categorised into local and global, focusing either on a specific instance or the full dataset. Methods from this group are used for data explainability.

(ii) Complexity-based, where the degree of interpretability is proportional to the number of trainable parameters. Simple models form the intrinsic group, where the interpretability is achieved intrinsically by design. Complex models form a post-hoc explainability group, which requires additional methods to provide explanations.[a]

(iii) Model-based, where we group methods based on the targeted models. Model-specific methods are tailored to a given model or a type of model such as CNN. Model-agnostic strategies do not make any assumptions about the internal structure and can be used on any model in a post-hoc fashion.

[a] Post-hoc interpretability also exists and involves building a new model next to the original one with better interpretability, such as a decision tree based on a ANN model.

(iv) Methodology-based approaches can be either gradient-based or perturbation-based. Perturbation-based methods involve modifying the inputs to alter the feature set of a given data point and then observing the corresponding changes in the network's output. This process is akin to data augmentation, with techniques including feature masking (for instance, obscuring certain parts of images) or employing generative algorithms to produce new samples. On the other hand, gradient-based methods are more sophisticated. They typically emphasise influential data segments by computing the gradient of the loss function with respect to the input data and propagating it from the output back to the input. This gradient provides information about the sensitivity of output changes to the input, allowing us to understand which parts of the input are most influential in the model's predictions. Some variations of gradient-based methods aim to identify the most influential layer by analysing the propagated gradients across different layers of the network.

In the survey of related works presented in Section 1.3, we focus exclusively on post-hoc model-specific approaches, which generally perform better than model-agnostic methods. We further drive our attention to CNN explainability methods, which are the most relevant to our study.

1.3 Related Work

In this section, we survey related works that propose CNN-model-specific XAI methods and also methods that were previously applied to medical imaging.

In a medical XAI survey conducted by van der Velden et al. [14] in 2021, the authors reviewed 178 papers published between the years 2017–2020. They focused on the used XAI method, anatomical location and the data modality (such as histology, MRI or X-ray). Table 1.1 shows usages for each method, regardless of the modality and anatomical location.

We can see that one of the most used methods is CAM [15]. It was used in 67 papers. It is also one of the oldest methods for post-hoc CNN explainability. The second most used method is its successor, the improved Grad-CAM [16], which was used in 54 papers. If we only consider histology modality (22 papers), then the CAM and Grad-CAM were used in 7 and 10 papers, respectively. Trainable

Table 1.1: Summarized results from the medical XAI survey conducted by van der Velden et al. [14].

method	usage count	usage [%]
CAM	67	37.6
Grad-CAM	54	30.3
Trainable Attention	13	7.3
Backpropagation	7	3.9
Multiple instance learning	7	3.9
LRP	6	3.4
Guided Backpropagation	5	2.8
Other 9 methods in total	19	10.7

Attention [17] was used in two works. Guided Grad-CAM [18], LRP [19], and Occlusion sensitivity [20] were only used in one work each.

In 2022, Zeineldin et al. [21] proposed a so-called NeuroXAI framework with seven implemented SOTA XAI methods that they considered most relevant and useful for making deep-learning models more transparent, especially in brain imaging analysis from MRI. Those methods involved the Vanilla Gradient [22], Guided Backpropagation [23], Integrated Gradients [24], Guided Integrated Gradients [24], SmoothGrad [25], Grad-CAM [16] and Guided Grad-CAM [18]. Their showcase for the methods was brain classification and glioma sub-region segmentation. As stated by the authors, Grad-CAM, Guided Grad-CAM, and SmoothGrad generated visualization maps with the least noise. In addition, SmoothGrad showed overall the best feature maps, highlighting the key discriminative parts of the input images.

SmoothGrad and Guided Integrated Gradient were also used by Rguibi et al. [26]. The authors applied these methods on CNNs trained on two datasets regarding brain tumour and pneumonia classifications, concluding the SmoothGrad provided satisfying results.

Grad-CAM++ [27], an enhanced version of Grad-CAM, was successfully used in a lung cancer detection task with the primary goal of increasing radiologist trust for better model adoption [28].

In a paper proposed by Rahman et al. [29], the authors trained a COVID-19 disease classifier from X-ray chest imaging. Score-CAM [30] was then used to

reveal that the classifier uses non-lung areas to make decisions. Similarly, [31] trained a CNN to classify lung diseases from X-ray imaging into three classes (COVID-19, pneumonia and normal). They utilised LayerCAM [32] to validate and explain the model behaviour.

Although we did not limit our survey to XAI usage in the medical field, we are not surprised that the majority of the works fall into this category. The medical field is indeed one of the fields which can benefit from XAI applications most. We further looked into the methods we examined to see how the performance of the methods is compared. Also, we wanted to understand the potential downsides and benefits of the methods before we select three to be later utilised.

While pixel-attribution methods such as Vanilla Gradient and SmoothGrad are mostly compared via qualitative analysis, the CAM-based methods are typically compared via the ILSVRC [33] object localisation benchmark. The authors of LayerCAM (Jiang et al. [32]) used the ImageNet [33] validation set counting 50k images to compare LayerCAM, ScoreCam, Grad-CAM++ and Grad-CAM. In essence, they binarise the activation maps with a given threshold and calculate the smallest rectangle bounding box, including all activated pixels. Then, the intersection over the union is calculated and compared. Based on that benchmark, LayerCAM outperforms all other mentioned CAM-based methods.

1.4 Selected XAI Methods

After a closer look into the methods we examined, we decided to pick the follofing three XAI methods which we believe would have the most potential in addressing our problem.

We chose Grad-CAM++ [27] because it is a direct enhancement of the heavily used Grad-CAM. Secondly, we picked LayerCAM [32]. Although it was used way less frequently than Grad-CAM++, perhaps for its novelty, we think it has the potential to provide better results than Grad-CAM++. Lastly, we picked SmoothGrad, a pixel attribution method commonly used even in recent works that seem to achieve satisfying results. In the next sections, we provide a detailed description of each method.

1.4.1 Grad-CAM++

This method was proposed by Chattopadhyay et al. [27] as an enhanced, generalized version of Grad-CAM [16]. The authors claim that their enhanced version provides more accurate visualisations of the predictions in terms of improved object localisation. Also, they stated the proposed approach can better handle multiple occurrences of the objects. That is very important for our case, as the potential features, such as the cell nuclei, are overly repetitive in the data.

To better understand Grad-CAM++, we start with the concepts of class activation maps (CAM) and Grad-CAM. Consider a CNN model that classifies objects into several classes. The CNN has a typical structure of several convolutional layers, followed by several fully connected layers, where the last one is the classification layer. The basic CAM method takes the convolutional layers and adds the global average pooling (GAP) layer and the classification layer. The classification layer is then retrained.

Finally, the CAM for a given image of class c is produced by weighting feature maps from the last convolutional layer with the weights $\boldsymbol{w}$ between the GAP layer and the neuron corresponding to the class c. The intuition behind this is that these weights represent the contribution of each feature map to the final output for a specific class. Figure 1.1 depicts an overview of the process with three feature maps.

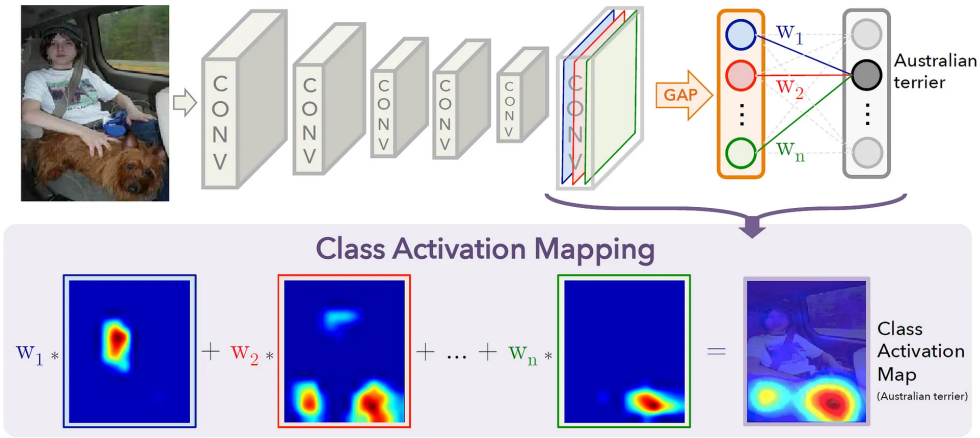

Figure 1.1: CAM using GAP. Source: [15]

Mathematically, we can write it as:

$$Y^c = \sum_k w_k^c \frac{1}{Z} \sum_i \sum_j A_{ij}^k \tag{1.1}$$

where Y^c is a class score, w_k^c are channel-wise weights for class *c* and *k*th feature map, $\frac{1}{Z}\sum_i \sum_j$ is GAP (*Z* is the total number of pixels of the feature maps), and $A_{i,j}^k$ is the pixel at (i, j) of the *k*th feature map.

The Grad-CAM stands for gradient-weighted CAM. It replaces the GAP-weighting with gradient-weighting with respect to the output class score Y^c. The weights are then calculated as follows:

$$w_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\delta Y^c}{\delta A_{ij}^k}, \tag{1.2}$$

where $\frac{\delta Y^c}{\delta A_{ij}^k}$ is the gradient. If we plug it into 1.1, we get:

$$L^c = \sum_k w_k^c A^k = \sum \frac{1}{Z} \sum_i \sum_j \frac{\delta Y^c}{\delta A_{ij}^k} A^k, \tag{1.3}$$

where L^c is the saliency map. However, the authors also proposed wrapping it into the rectified linear unit (ReLU) function to keep only the features that positively contribute to the output. The final saliency map would then be:

$$L^c = \text{ReLU}(\sum_k w_k^c A^k) = \text{ReLU}(\sum \frac{1}{Z} \sum_i \sum_j \frac{\delta Y^c}{\delta A_{ij}^k} A^k). \tag{1.4}$$

Finally, the Grad-CAM++ introduces pixel-wise gradient weighting to compensate for the area-dependent feature map contributions that were previously uniformly scaled (averaged). The authors propose weighting more significant pixels with higher weights. That significantly improves the detection of multiple object instances. The idea is visually presented in Figure 1.2.

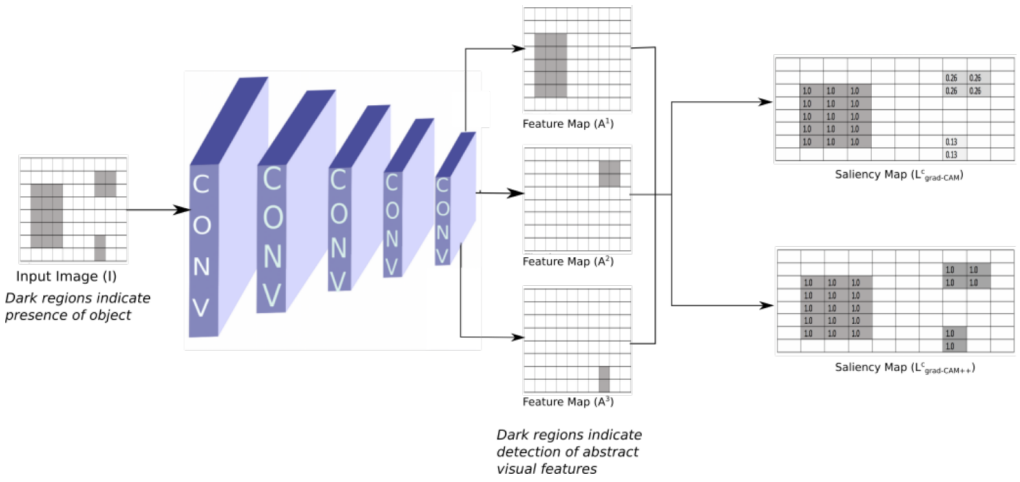

Figure 1.2: Grad-CAM++ pixel-wise weighting intuition in a binary classification case. In Grad-CAM, smaller instances of the same class get lower scores in the saliency map. Grad-CAM++ improves this by pixel-wise weighting. Source: [27]

Mathematically, the authors reformulated Eq. 1.2 by adding weighting coefficient α to the formula:

$$w_k^c = \sum_i \sum_j \alpha_{ij}^{kc} \text{ ReLU}(\frac{\delta y^c}{\delta A_{ij}^k}). \tag{1.5}$$

The α_{ij}^{kc} is the weighting coefficient for pixel (i, j) in the *k*th layer for class *c*.

Consider a binary object classifier with A^k as the *k*th feature map visualisation of the last convolutional layer. Each A^k is expected to be triggered by a visual pattern, for which $A_{ij}^k = 1$ if the pattern is detected and $A_{ij}^k = 0$ otherwise. Consequently, derivatives are expected $\frac{\delta y^c}{\delta A_{ij}^k} = 1$ if $A_{ij}^k = 1$ and vice versa. To achieve equal importance of the objects in feature maps in this example, the pixel-wise weights α_{ij}^{kc} for $\frac{\delta y^c}{\delta A_{ij}^k} = 1$ would be calculated as:

$$\alpha_{ij}^{kc} = \frac{1}{\sum_{l,m} \frac{\delta y^c}{\delta A_{lm}^k}}. \tag{1.6}$$

For $\frac{\delta y^c}{\delta A_{ij}^k} = 0$, the α_{ij}^{kc} would be 0.

The authors of this method proposed a closed form for a general neural network, which we do not present here. We refer to the original paper [27] to learn more about its derivation.

1.4.2 LayerCAM

As stated by the authors Jiang et al. [32], LayerCAM is a simple yet effective method that outperforms Grad-CAM++. The main difference to the Grad-CAMs is that LayerCAM combines (often complementary) semantic knowledge from multiple layers, while Grad-CAMs only use the last convolutional layer. They argue that the shallower layers provide more fine-grained object details and coarse spatial position. Importantly, the method is also directly applicable to any CNN. There is no need to change its architecture as in the case of CAM.

In LayerCAM, the class activation maps are first generated for each or a subset of layers. That is done by pixel-wise class-specific gradient weighting of feature maps. As opposed to Grad-CAM++ where the authors add pixel-wise term α_{ij}^{kc} to the weight calculations, the LayerCAM authors define pixel-wise weight w_{ij}^{kc}. The weight is defined simply as a positive gradient or zero otherwise.

Using the same notation as above, it would formally be:

$$w_{ij}^{kc} = \text{ReLU}(\frac{\delta y^c}{\delta A_{ij}^k}). \tag{1.7}$$

The map M_l^c of the given layer l for class *c* is then calculated by linearly combining the weighted maps and filtering the negative values using the ReLU function.

$$M_l^c = \text{ReLU}(\sum_k w_{ij}^{kc} \delta A_{ij}^k) \tag{1.8}$$

Finally, all the layer maps are fused into one by taking an element-wise maximum. Thus, the pixel at location (i, j) in the final map $L_{i,j}^c$ is calculated as:

$$L_{i,j}^c = \max_{\text{all layers } l}(M_{l,i,j}^c), \tag{1.9}$$

where $M_{l,i,j}^c$ is a pixel located at (i, j) in the map of the given layer l for class *c*.

1.4.3 SmoothGrad

The previous methods generated class activation maps by weighting feature maps. The SmoothGrad method, proposed by Smilkov et al. [25], utilises a slightly different approach. However, it is still a gradient-based method.

SmoothGrad extends the so-called Image-Specific Class Saliency [22] method, known as Saliency Maps or Vanilla Gradient, which is one of the first-pixel attribution methods. To avoid confusion, we refer to this method as Vanilla Gradient, as saliency maps are generally outputs from all the described methods.

To understand SmoothGrad, we first introduce the Vanilla Gradient and the issues associated with it. Consider a CNN that classifies input image I into C classes. Given input I, the network produces output score S_c for each class c. Vanilla Gradient calculates the gradients of the S_c with respect to the pixels of the image I. The gradients are then directly used to form the saliency map. The intuition behind this is that the map represents how big a change in the final score S_c would be if a given pixel changed. In other words, the saliency map is a measurement of pixel influence. Formally speaking, the map is calculated as:

$$M_c(I) = \frac{\delta S_c(I)}{\delta I}. \tag{1.10}$$

Vanilla Gradient has several disadvantages. One of the most significant is that the produced maps are very noisy and hard to interpret for humans. The next major issue is saturation, causing the gradients to become small or completely vanish. If we follow the intuition stated above, we can imagine a cats and dogs classifier. Cat's whiskers can be a strong feature for the classifier. In that case, a small change in the whiskers, such as shape or length, would not have a significant impact on the output score (it is saturated). Consequently, the gradient would be small, and the feature would not be highlighted in the saliency map. That is a problem because the whiskers were a highly relevant feature, which we would expect to see highlighted in the saliency map. Mathematically, saturation occurs when the output of a given non-linear activation function is in its "flat region" (near its minimum or maximum,

e.g. near 0 or 1 for sigmoid), causing the gradients to become very small or vanish.

The SmoothGrad authors address these issues by applying Gaussian noise $\mathcal{N}(0, \sigma^2)$ to the image. Afterwards, they utilize the Vanilla Gradient with the new noisy sample. Generally, they do this n times and then average the results. As we can see, the method comes with two hyperparameters: the number of samples n and the standard deviation σ of the noise, the "noise level".

Formally, the SmoothGrad saliency map can be written as follows:

$$\hat{M_c}(I) = \frac{1}{n} \sum_{1}^{n} M_c(I + \mathcal{N}(0, \sigma^2)), \tag{1.11}$$

where M_c is the saliency map from Vanilla Gradient.

The authors observed best results with hyperparameters $n <= 50$ and $\sigma \in [0.1, 0.2]$, claiming that $n > 50$ caused diminishing returns. We take this information into account when utilising this method in the next chapter.

Chapter 2

Methodology

In this chapter, we describe our methodology. First, we define the aims of this work and provide the necessary definitions. Next, we describe how the dataset was created and preprocessed. Finally, several examples from the dataset are presented.

2.1 Task Definitions

The work is composed of two consecutive tasks. The first one is a regression task. The goal is, given the dataset described in the next section, to train a CNN that would predict MD and FA from the histology patches. After such a model with reasonable performance is trained, we move to the second task – explainability.

It is still unknown how and which histology features influence MD and FA values. We aim to change this by applying explainability methods to the trained model and performing qualitative analysis. We want to infer patterns that lead to increased MD and FA. In Section 3.3, we describe how we used them and in 3.4 how we performed the qualitative analysis.

On a side note, we also train a separate network that predicts the cell nuclei count (CC). The reason for this is to gain more trust in the XAI methods, as it is the only target where we know what features the network should focus on.

2.2 Dataset

This Section describes the dataset that we use in the work. It also explains how the data were collected and preprocessed prior to running experiments.

2.2.1 Data Origin

As the introduction mentions, this work is a direct follow-up to the research done by Brabec et al. [6]. Inherently, we also use the same dataset that was proposed there.

The collection of the dataset was not trivial. It required substantial knowledge of MRI technology, experience in histology and access to numerous specialised medical equipment, such as microtome, MRI scanner or light microscope. The description here is given in a more high-level fashion to fit the scope of this work. Therefore, we refer to the original work to get more in-depth information.

The first step in the data collection was to obtain tissue samples with brain tumours. That was done using neurosurgical excision from 16 patients. Each tissue was then cut into smaller pieces with an approximate size of $35 \times 20 \times 2\ mm^3$ and placed into a 3D-printed holder.

When all the samples were prepared in the holder, an MRI scanner was utilised to scan each sample with a fine resolution of $0.2 \times 0.2 \times 0.2\ mm^3$, allowing the capture of the microstructure. The scanning was done in six different directions and three b-values ($100, 1000, 3000 s/mm^2$) in order to reliably calculate the diffusion tensor for each voxel using the DTI technique. In this context, voxel is a 3D representation of a segment of a given tissue in the scanner. After the diffusion tensors were calculated, they could further derive MD and FA, which are the ground truths of the target variables in the dataset.

Given the diffusion tensor $\boldsymbol{D}$, MD and FA are defined as follows [6]:

$$\text{MD} = \frac{Tr(\boldsymbol{D})}{3} = \frac{\lambda_1 + \lambda_2 + \lambda_3}{3}, \tag{2.1}$$

$$\text{FA} = \sqrt{\frac{3}{2}} \cdot \sqrt{\frac{(\lambda_1 - \text{MD})^2 + (\lambda_2 - \text{MD})^2 + (\lambda_3 - \text{MD})^2}{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}}, \tag{2.2}$$

where $Tr(\boldsymbol{D})$ is the trace of the diffusion tensor.

After scanning, a $4 - 5$ µm thick slice was cut from each sample. To highlight the microscopic features that allow better tissue differentiation, each slice was stained with Hematoxylin & Eosin (H&E). That is a commonly used stain in histopathology, which increases the tissue contrast by mainly colouring cell nuclei to purple and cytoplasm to shades of pink. Finally, the specimens were scanned using a light microscope with 0.5×0.5 µm resolution.

To match the histology images with the MD/FA maps, a technique called coregistration was employed. Their approach was based on non-rigid landmarks, which means the overlays can be stretched for better matches. This process is generally very challenging and error-prone as it maps images from different modalities and scales. To mitigate this, the authors smoothed the MRI maps using Gaussian kernel.

Finally, the QuPath[a] software was utilised to calculate cell nuclei in each histology sample. That is the last target variable CC in our dataset.

To sum up, this dataset has three target variables: MD, FA and CC. The ground truth data was obtained via DTI from MRI scans and the QuPath program. The training image data were obtained using a light microscope. Both are based on the 16 initial tissue samples.

2.2.2 Data Preprocessing

At this point, we had the MRI maps and histology matched. However, the data were still unsuitable for fitting a CNN. The histology images from the light microscope had too high resolution and contained varying backgrounds. A region of interest was introduced for each sample to remove the backgrounds and edges, reducing the noise in the data. To address the high-resolution problem, we cut the images into smaller patches and resized them to 360×360 px.

This procedure was applied to all 16 samples, resulting in roughly 64k image patches. After considering our computational resources, we trained only on samples 3 and 4. However, the generalisation abilities were evaluated on all of them. Samples 3 and 4 accounted for 4335 and 3908 images, respectively. In

[a] https://qupath.github.io/

total, our training dataset contains 8243 images. Several examples can be seen in Figure 2.1.

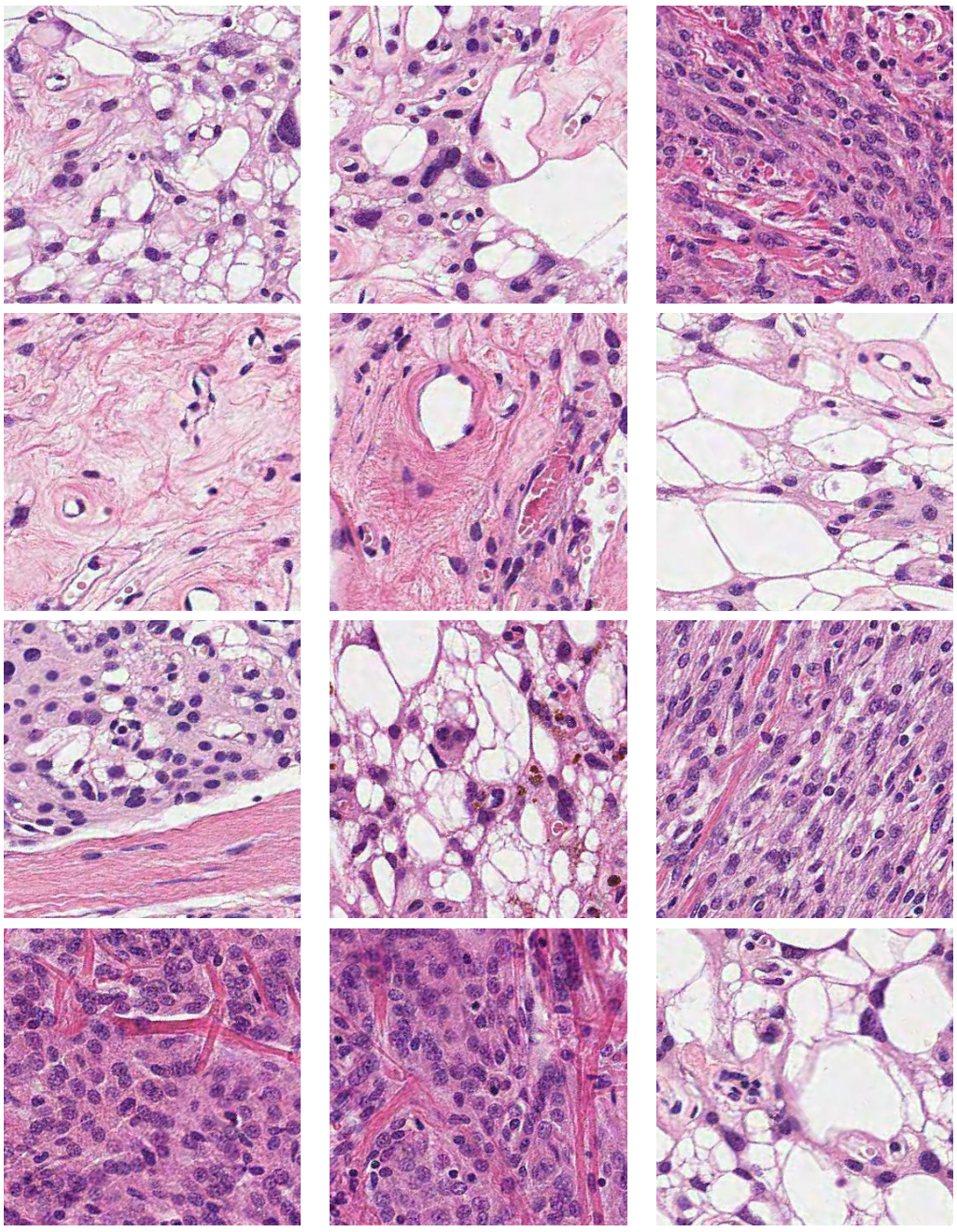

Figure 2.1: This table shows several examples from the dataset. Individual images are histology patches derived from the H&E-stained samples. The purple blobs are cell nuclei, the pink areas are cytoplasm, and the white areas are air spaces.

Chapter 3

Implementation and Experiments

This chapter describes the key software, libraries, and approaches implemented in our research. All computations were carried out within a Docker virtual environment. The host machine, running on Windows 10 22H2, leveraged the WSL2 (Windows Subsystem for Linux, version 2) technology to operate the Docker environment on Ubuntu 22.04.3 LTS. The host machine's hardware consisted of a Core i7-9700k processor, an NVIDIA GeForce RTX 2080 Ti graphics processing unit, and 32 GB of RAM.

3.1 Software

One of the fundamental technologies used in this research was Docker and Docker Compose [34]. This open-source platform enables the encapsulation of applications and their dependencies within self-contained containers. Docker Compose further streamlines this process by allowing for the efficient management of multi-container Docker applications. These tools collectively simplify the setup process for scientific computing and ensure reproducibility on any system supporting Docker.

We utilised Python[a], a widely recognised programming language, for its extensive support of data science tools and libraries, making it a popular choice in machine learning. Its clear syntax and readability contribute to its widespread use. The specific version employed in our work was Python 3.11.

[a] https://www.python.org

Our machine learning models were built using the latest TensorFlow 2.15 [35], a popular framework developed by Google that offers many tools and algorithms for creating neural networks. However, as TensorFlow can be somewhat low-level, we utilised the Keras library [36], allowing us to write even more concise code and set up experiments faster.

In the early stages of our research, we used Jupyter notebooks[a] for prototyping. This interactive web application allows for executing Python code in cells, visualising graphs and images, and including textual notes for explanation and clarification. After setting up working prototypes, we moved to standalone parametrised Python scripts, which enabled us to experiment with different hyperparameters more effectively.

Nevertheless, choosing the best hyperparameters based on numerous experiments is challenging. Hence, we utilised TensorBoard[b], a visualisation toolkit that provides an interactive interface for real-time monitoring of metrics like loss. That helps in identifying model convergence and potential training issues such as overfitting. Furthermore, TensorBoard allows the comparison of multiple training runs, which is particularly beneficial for hyperparameter tuning.

Lastly, we used Streamlit[c] open-source framework to build an interactive web app for quantitative analysis of the results.

To wrap up, we created a custom Docker container based on the official latest Tensorflow image with GPU support and bundled Jupyter notebook [d] and then further extended it with the mentioned dependencies.

3.2 Prediction Model Development

This Section provides an overview of the CNN training process regarding selecting an architecture and hyperparameters. Our model implementation is highly configurable, making the exploration of many architectures easier. It allowed us to experiment with different augmentation techniques, the number of neurons in hidden layers, the number of neurons in feature layers, dropout

[a] `https://jupyter.org`
[b] `https://www.tensorflow.org/tensorboard`
[c] `https://streamlit.io/`
[d] `https://hub.docker.com/r/tensorflow/tensorflow/`

Table 3.1: Abstract architecture of the implemented model. HP=hyperparameter, N=number of neurons. The prefix "X-" means the layer is present for every target in the case of a multi-task network.

#	Layer
1	Augmentations (HP)
2	Pre-trained backbone (HP)
3	GlobalAveragePooling2D
4	Dropout (HP)
5	Dense (HP), Linear
6	X-features: Dense (HP), Linear
7	X-dropout: Dropout (HP)
8	X-output: Dense (N1), Linear

rates, or even multiple training outputs to employ multi-task learning. The architecture overview can be seen in Table 3.1.

We used the standard MSE loss, which is widely used for regression tasks and is defined as:

$$MSE = \sum_{i=1}^{D} (y_i - \hat{y}_i)^2 \tag{3.1}$$

where D is the dimension of training data, y_i is the i-th element of the ground truth vector Y and $\hat{y}_i$ is the i-th item of the predicted vector $\hat{Y}$.

To compare the network performance, we used the R^2, which tells us how much variation in the training data is explained by the model.

$$R^2 = 1 - \frac{var(Y - \hat{Y})}{var(Y)} \tag{3.2}$$

The R^2 was also used as an additional training metric.

Each network was first trained with frozen backbone weights until an early stop interrupted the training. Then, the whole network was fine-tuned for a given amount of epochs with the early stop employed. To ensure the reproducibility of the results, a random seed was set before every experiment and passed to all relevant functions, such as data shuffling.

3.2.1 Training Data

The training data consists of merged samples 3 and 4, as mentioned in Section 2.2.2. Before splitting the data into train, validation and test sets, the data were fully shuffled to minimize the effect of distribution bias. We used 20% of the data for the validation set and another 20% for the test set. The validation set was used to tune the hyperparameters during training, and the test set was used in the interpretation and analysis.

3.2.2 Augmentations

With the nature of the data and the goal of this work in mind, we implemented augmentations that could positively impact the predictions and robustness of the network.

Firstly, we implemented random flips, both vertical and horizontal. The good thing about this augmentation is that we do not lose any details, as all pixels in the image are preserved.

Similarly, the second implemented augmentation technique – random rotation by $k \times 90°$. Thanks to the right angle multiples, no corners are cut, and the details are also preserved. That is not the case for the third augmentation technique, which was random rotation by a given degree. However, we only explored small degrees (up to $10°$), as every extra degree dramatically reduces the details of the picture due to interpolation.

3.2.3 Backbones

Considering our computational resources, we selected eight well-performant backbones available in the Keras Applications[a], including MobileNet{1, 2, 3-Small, 3-Large}, ConvNext-{Tiny, Small}, DenseNet-121 and EfficientNet2-Small. All of these were used with weights trained on the ImageNet[b] dataset. Additionally, we experimented with DenseNet-121 weights trained on histopathology, known as Kimianet, proposed by Riasatian et al. [37].

3.2.4 Multi-task Learning

Since our task requires us to predict several target variables from the same inputs, and two of them have a similar value range (MD, FA), it is rational to

[a] `https://keras.io/api/applications/`
[b] `https://www.image-net.org/`

Table 3.2: Summary of hyperparameters to set in our network implementation. HP=hyperparameter, LR=learning rate. Nested HP are hyperparameters of hyperparameters, e.g. AdamW comes with Weight Decay HP. The values of these HP are in the "nested values" column.

HP	**values / nested HP**	**nested values**
Batch Size	4, 8, 16, 24, 32	
Epochs	[50-100]	
LR	1e-05, 3e-05, 5e-05, 1e-04	
LR scheduler	none/Cosine Decay	
	Warmup Epochs	[5-20]
	Decay Epochs	[50-100]
	Initial LR	1e-05, 3e-05, 5e-05, 1e-04
	Warmup target LR	1e-03, 3e-03, 5e-04
Optimizer	Adam/AdamW	
	Weight Decay	[1e-3 - 1e-2]

think about multi-task learning. Instead of two models of the same size and perhaps slightly different architectures, we would have one model predicting both targets, with about the same size as a single output model. Moreover, we would save twice as much time on hyperparameter tuning and training.

Even though multi-task learning sometimes fails and performs worse than individual networks, we decided to experiment with it. To our advantage, we empirically discovered that multi-task learning does not reduce prediction performance compared to two single-output networks. Furthermore, it seemed the multi-task approach positively influenced the performance. Therefore, most experiments with these targets were done with the multi-task approach.

The third target CC was trained on a separate network due to large differences in the training loss. The loss of CC was much bigger than the MD/FA loss. Even appropriate static loss-weighting did not improve the convergence of the multi-task network with all three targets. Therefore, we trained a separate single-task network to predict the CC. The rest of the network architecture remained the same, including the hyperparameters.

Table 3.2 presents all the other hyperparameters we fine-tuned, including the values we explored.

3.2.5 Implementation Details

We used several techniques to make the training more resource-efficient. Firstly, we enabled the mixed-precision[a], which means that almost all the variables in the network are 16-bit floating types. Only the last layer is kept with a 32-bit floating type to avoid the prediction quality loss due to numerical instability. This has led up to $3\times$ faster training.

Another employed technique was the accelerated linear algebra (XLA), which, simply put, precompiles the model graph and optimizes it by reducing the number of operations. That can be done, for example, by replacing compound operations with single operations supported by hardware. This optimization technique not only speeds up the training but also allows larger batch sizes for training. Unfortunately, not all operations in the model graph are usually supported. In our case, it was necessary to implement our own XLA-ready random rotation layers and the R^2 metric.

Undoubtedly, the total speedup of $3 - 4\times$ allowed us to explore hyperparameters significantly faster, resulting in a broader explored space.

3.2.6 Final Models

To proceed to the XAI techniques, we needed to pick the best model based on the R^2 validation metric of the MD, FA and CC outputs. First, we describe the final hyperparameters and then the performance.

The best-performing multi-task model uses EfficientNet2-Small as a backbone and was trained **without** any augmentation techniques. It turned out that even the flips and rotations without detail loss negatively impacted the performance. The AdamW [38] optimizer and the Cosine Decay Learning rate scheduler were used in the training. The hyperparameters of the final model are shown in the Table 3.3. Dropout rates were set to 0.3. The single-task model predicting CC performed very well with the same hyperparameters as the best multi-task. Therefore, we kept it simple and picked a model with the same hyperparameters.

Regarding the evaluation metric, the models perform very well. Table 3.4 shows the R^2 for all data splits. As we can see, the results do not suggest any overfitting

[a] `https://www.tensorflow.org/guide/mixed_precision`

Table 3.3: Best hyperparameters of our implementation. HP=hyperparameter, LR=learning rate. Nested HP are hyperparameters of hyperparameters, e.g. AdamW comes with Weight Decay HP. The values of these HP are in the "nested values" column.

HP	**values / nested HP**	**nested values**
Batch Size	16	
Epochs	100	
LR scheduler	Cosine Decay	
	Warmup Epochs	17
	Decay Epochs	60
	Initial LR	1e-5
	Warmup target LR	5e-5
Optimizer	AdamW	
	Weight Decay	0.008

Table 3.4: R^2 of the best models, trained on the train set derived from merged samples 3 and 4. The table shows the results for all data splits. We can see that all the data splits have nearly the same R^2, supporting no overfitting issue. Note that the MD and FA were trained together in a multi-task model. The CC was trained separately.

R^2 [%]	MD	FA	CC
train set	98.60	99.19	99.54
validation set	98.50	99.12	99.51
test set	**98.65**	**99.20**	**99.52**

issues. All sets achieve almost identical, and yet very high, scores. We also evaluated the models on all other patient samples. These results are described in Chapter 4.

3.3 Applying XAI Techniques to the CNN

A brief research about the available implementations was done before considering our implementation. To our advantage, we found a suitable implementation of the chosen methods in the Keras-Vis framework [39], which works with regression mixed-precision Keras models. Although we have encountered several minor issues in the initial stage of making the framework work with our models, we have managed to use the framework succesfully.

One of the crucial things that was necessary to define for each visualisation technique was a so-called *score* function. This function specifies how the model output (respectively, the last convolutional layer) influences the visualisation. In other words, it is what the visualisation should focus on. In the case of a classification task, it may be which features contribute most to the given class, such as dogs or cats. In our regression case, it can be the focus on higher values – the higher the model output, the higher the contribution. Alternatively, we could visualise what makes the predictions low. Finally, we could visualise what contributes to one specific target – the closer the prediction to the target value, the more pronounced the visualisation. We have concluded that in our case, the most suitable score function is the first mentioned – the higher the prediction, the higher the contribution to the visualisation.

After getting familiar with the library and being able to produce all desired visualisations for single images, we wrote scripts to generate visualisations for the whole test set, along with metadata for each image *i*. Those included ground truth, predicted value, their absolute difference Δ and relative difference rΔ.

$$\Delta_i = \hat{y}_i - y_i \tag{3.3}$$

$$r\Delta_i = (\frac{\hat{y}_i}{y_i} - 1) \cdot 100 \tag{3.4}$$

The rΔ would then be interpreted in percentages.

3.4 Application for Qualitative Analysis

One of the main goals of this work is to try to infer valuable insights from the visualisations. That can be best done by qualitative analysis, where we compare multiple images of the same kind. In this context, the same kind is a similar target value, i.e. either MD or FA. This Section describes the web application we built with the Streamlit framework to make the analysis possible.

We start with the list of functional requirements we declared before implementation. The app should enable us to:

1. view histology patches with XAI visualisation overlay in a grid,
2. switch between visualisation techniques,
3. filter patches based on target type (MD/FA/CC),
4. filter patches based on ground truth, Δ and rΔ value,
5. adjust alpha channel of the visualisation overlay.

In addition to the initial functional requirements, we implemented several other features, which we found later to be useful. Those include changing overlay colourmaps, histograms for all the metadata, pagination or selecting different hyperparameters of the visualisation techniques.

In total, we created four pages. The *Main* page provides basic information about the web app, the models and the visualisation. The other three pages contain the aforementioned settings in a collapsible sidebar and a grid with image patches that meet the filter criteria. Each image has an overlay, also determined by the configuration in the sidebar. The grid size is adjustable, too. Each image has a title containing the following information (in order): image ID in the test set, y, $\hat{y}$, Δ, $r\Delta$. Figure 3.1 shows a screenshot of the application.

The other three pages differ as follows. The *Single Target* page allows comparing a lot of different image patches with a given visualisation type for a given target. The *Multiple Targets* page allows comparing visualisations for different targets of the same image patch. The *Multiple Methods* page allows comparing different visualisation methods of the same image patch for a given target.

Lastly, the application was put behind a login wall, wrapped into its own Docker container and deployed to the Google Cloud Run platform[a]. That allowed us to discuss and consult the results with the authors of the initial study [6], which this work directly extends. We do not disclose this website; it was built only for internal purposes.

In the next chapter, we look at the results in detail and discuss the visualisations.

[a] `https://cloud.google.com/run`

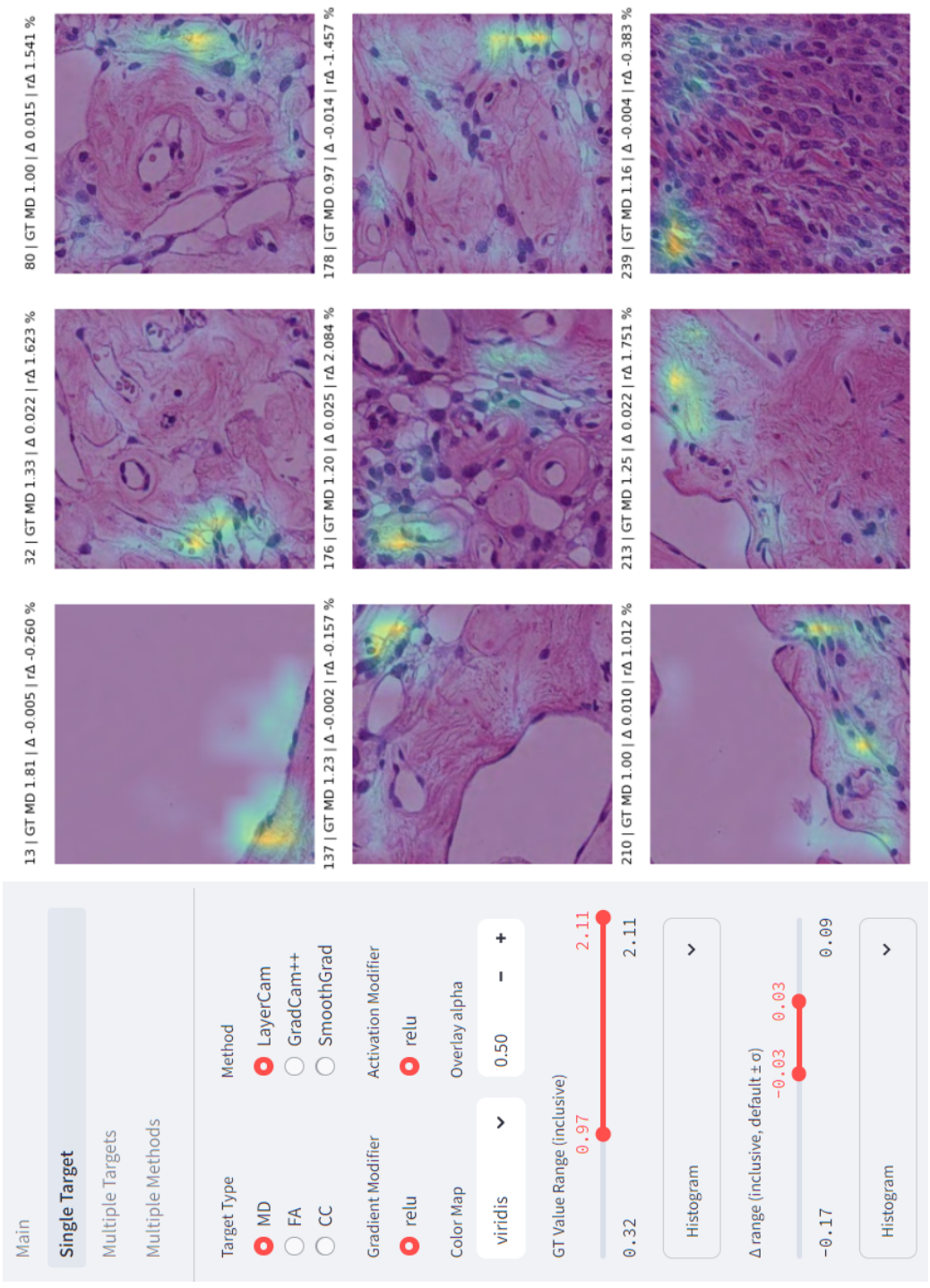

Figure 3.1: Webapp for qualitative analysis

Chapter 4

Results and Discussion

This chapter introduces the results of the best trained CNNs, then dives into the visualisation methods regarding possible interpretations and discussion.

4.1 CNN models

In this Section, we present the full results of the best models on all 16 patient samples in the dataset, including the train set, validation set and test set. We put all the results in the Table 4.1.

The table shows the R^2 score for all samples and targets. The MD and FA were predicted by the multi-task model, and the CC by a separate network. Firstly, we can see that the R^2 scores of MD, FA, CC on the test set (98.65%, 99.2%, 99.52%), validation set (98.5%, 99.2%, 99.54%) and the train set (98.6%, 99.19%, 99.54%) are all very high and similar. That is a very important finding as it suggests no overfitting issue in the networks.

The second key finding is that the model performs reasonably well on other samples which were extrapolated. That suggests the network has learned features that are general enough to extrapolate new unseen samples. Regarding the MD, out of 14 extrapolated samples, 11 have a higher score than 70%, and nearly eight samples have a higher score than 80%. FA performs slightly worse, but the scores are still solid. Eight samples have scored above 70%. It is an immensely good result compared to the scores achieved in the initial

Table 4.1: The table shows model R^2 [%] evaluations on all patient samples for all targets. MD and FA are predicted by the multi-task model, CC is predicted by the single-task model with the same hyperparameters and inputs. The highlighted samples denote data that were part of the training process. The rows that are not highlighted denote samples that were extrapolated.

	multi-task		**single-task**
data	**MD [%]**	**FA [%]**	**CC [%]**
sample 1	79.89	75.11	87.03
sample 2	76.80	73.84	92.80
sample 3	**99.59**	**98.29**	**99.46**
sample 4	**99.51**	**99.69**	**99.82**
sample 5	83.75	72.06	82.46
sample 6	85.26	79.00	84.87
sample 7	80.92	62.43	97.07
sample 8	80.82	86.35	83.46
sample 9	82.81	78.49	87.20
sample 10	79.56	58.10	84.29
sample 11	70.69	52.21	95.00
sample 12	63.52	58.38	85.75
sample 13	39.13	55.83	94.28
sample 14	74.77	60.54	78.36
sample 15	68.76	86.36	86.23
sample 16	81.31	70.88	51.53
train set	**98.60**	**99.19**	**99.54**
validation set	**98.50**	**99.12**	**99.54**
test set	**98.65**	**99.20**	**99.52**

study [6]. Our multi-task model vastly outperforms their per-sample[a] models even though our model is extrapolating. Regarding MD, their highest test set score was 60% on sample 13. All the other samples had scores below 40%. As for FA, all of their test-set scores were below 40%. To put it in perspective, our test scores achieved 98.65%, respectively 99.2%. The results from the initial can be seen in Figure 4.1.

In terms of CC, we cannot directly compare it with the previous study, as they did not use the target. They used cell density, which is per-sample normalized

[a]Per-sample model means that the model was trained using only data of the given sample. I.e. the train, validation and test sets were derived only from the one given sample.

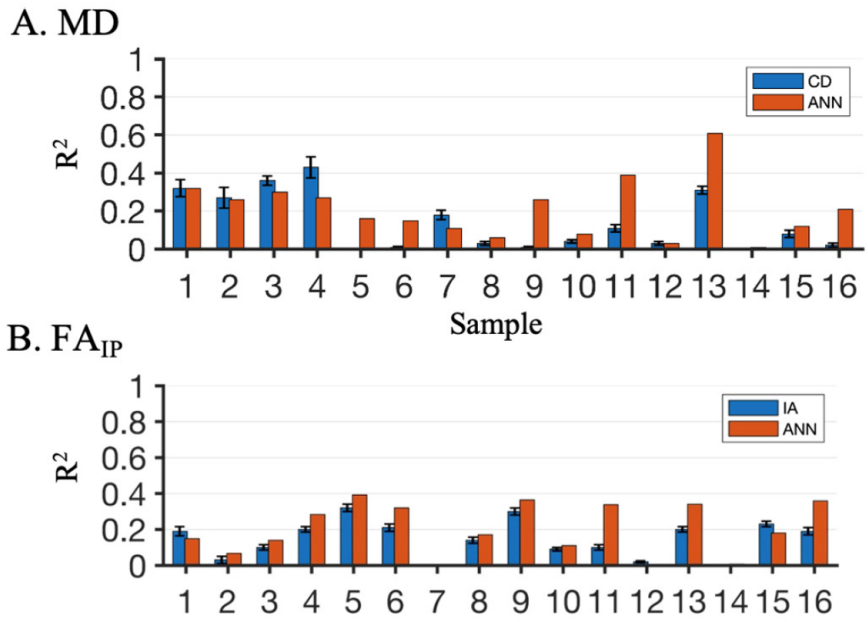

Figure 4.1: This figure depicts the results from the initial study [6]. Only the orange bars denoting ANN are relevant to us. Their ANNs are trained per-sample, i.e. a model was trained for each sample separately. The results presented are evaluated on test sets. Source: [6]

CC that we use. Therefore, using cell density as a target is only appropriate in per-sample models.

Nevertheless, the CC scores are even better than with MD – 12 out of 14 extrapolated samples have scores above 82%, and eight extrapolated samples have higher scores than 85%. Four extrapolated samples have scores even above 92%.

We attribute such improvement in the results to the following points. Primarily, we combined samples 3 and 4 for training. More data means more potential for generalization capabilities. Next, we utilised multi-task learning as opposed to the initial study. Multi-task learning can further improve the model's generalisation as it works as a regularization technique. Lastly, thanks to our focus on optimizations for resource-efficient training, we extensively explored the hyperparameter space by running more than 250 experiments.

After analysing the results, we were fairly confident with our models' capabilities and motivated to proceed with XAI methods.

4.2 XAI Methods Outcomes

This Section presents visualisations generated by the utilised methods, which we described in depth in 1.4. We first start with the CC target that was predicted solely for the purpose of validating and getting confident with the visualisations, as we know what features the network should focus on. Then, we proceed to the main targets MD and FA. For each target, we show and comment on outcomes from all three XAI methods. We recommend zooming in on the examples for maximum detail.

4.2.1 Target CC

Here, we describe the XAI outcomes for CC target. Several examples can be seen in Figure 4.2. More examples can be seen in Appendices B.1, B.2, B.3 and B.4.

The CAM-based methods did not produce satisfying results. Grad-CAM++ produced mostly blank visualisations. In cases where they were not blank, the highlighted areas are around the image edges. Only few patches are highlighted across the whole patch, as expected.

LayerCAM, on the other hand, did not produce any blank visualisations. However, the vast majority of overlays look very similar – highlighted areas are around the edges. We explain this by the effect of an overlay normalisation. If the gradient is small but similar for all pixels, the feature should have the same intensity. Provided that there is some noise or systematic bias in the image patch that has a bigger gradient, it could easily dominate. The normalisation would then effectively remove the real features from the highlighted areas. It could also be that the CAM-based methods are designed for macroscopic objects with one to lower units of occurrences, while in our case, we have higher tens to several hundreds of occurrences of the given object.

The SmoothGrad method seems more convincing. We can see the nuclei are fully highlighted in a lot of cases. In some cases, only a part of them is highlighted. Sometimes, it is only visible after zooming. We think this is not an issue, as the feature detector could learn the shape of the nuclei and only use part of that information for detecting cell nuclei. We need to keep in mind that the model has very high R^2 on the test set; in other words, the model evidently

learned to detect the nuclei accurately. This is also supported by the fact that it is hard to find some incorrectly highlighted features.

We can also compare two different hyperparameters of SmoothGrad, which, in this case, produce slightly different highlighted areas. Most of the areas seem to be the same, but setting $\sigma = 0.3, n = 30$ seems to highlight a bit more. Note that we only present results with the most relevant hyperparameters as the other explored did not bring any novelty.

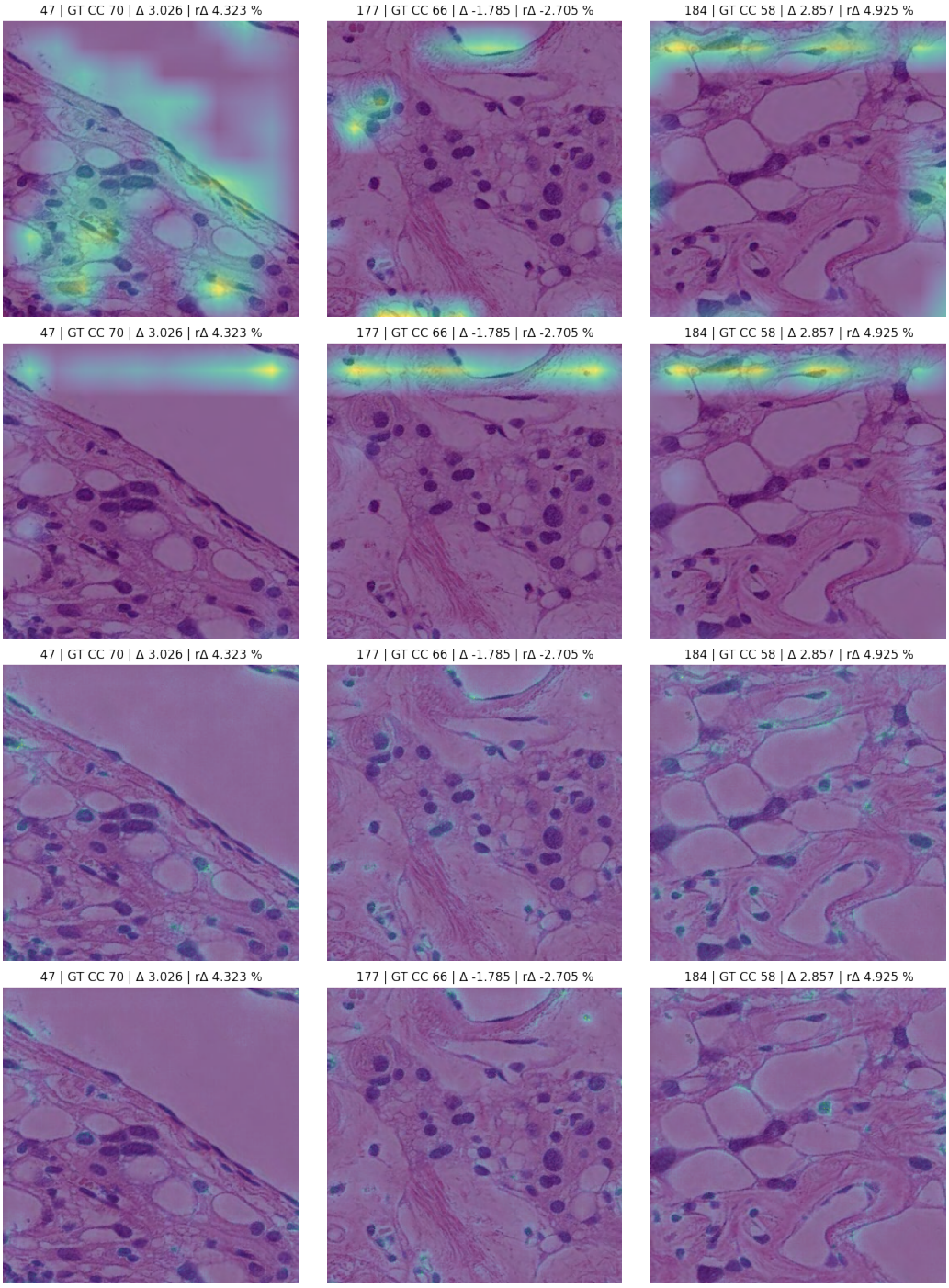

Figure 4.2: This figure shows three randomly picked patches with a XAI visualisation overlays (alpha=0.6, colour map=*viridis*) for CC target. Rows represent different methods. From top: Grad-CAM++, LayerCAM, SmoothGrad ($\sigma = 0.3, n = 30$), SmoothGrad ($\sigma = 0.2, n = 30$). The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

4.2.2 Target MD

Although the CAM-based visualisations for the CC target did not work well, it seems more plausible with the MD target. The examples are visible in Figure 4.3. More examples can be seen in Appendices B.5, B.6, B.7 and B.8.

There are still sometimes repetitive patterns around the edges for small values of MD (≤ 0.3). In those cases, the normalisation causes the highlighting of unimportant features, as explained above. Besides, none of the visualisations is blank. They all seem distinct for high-enough MD values, and they are usually spread out across the whole patch.

Both Grad-CAM++ and LayerCAM produced similar visualisations, but we can conclude that LayerCAM provided more detail and usually highlighted more areas. Also, the methods rarely highlighted empty areas of the histology patch, which increases our confidence in the methods (see patch 210 in Fig. 4.3).

SmoothGrad visualisations are again visible in with two sets of hyperparameters. Here, the difference between them is more observable than with CC. Each set of hyperparameters seems to be highlighting slightly different areas. Unfortunately, this decreases the trust in the method, as we do not know which visualisation is more reliable and which should be taken into account. Perhaps fusing the maps could be beneficial (or just interpreting them together).

Nevertheless, we can see the cell nuclei play a strong role in the prediction as many of them are highlighted. At the same time, they are by far less important than with the CC. In addition to nuclei, we can see highlighted areas of other histology features, such as the border between white space and a histology feature.

However, we were unable to find any exact general patterns that positively correlate with increased MD. For example, we are still unsure whether the position, shape or size of nuclei matters or whether the local density of the cells plays a role. We were able to find representatives for all the mentioned, making it very difficult to interpret and make conclusions.

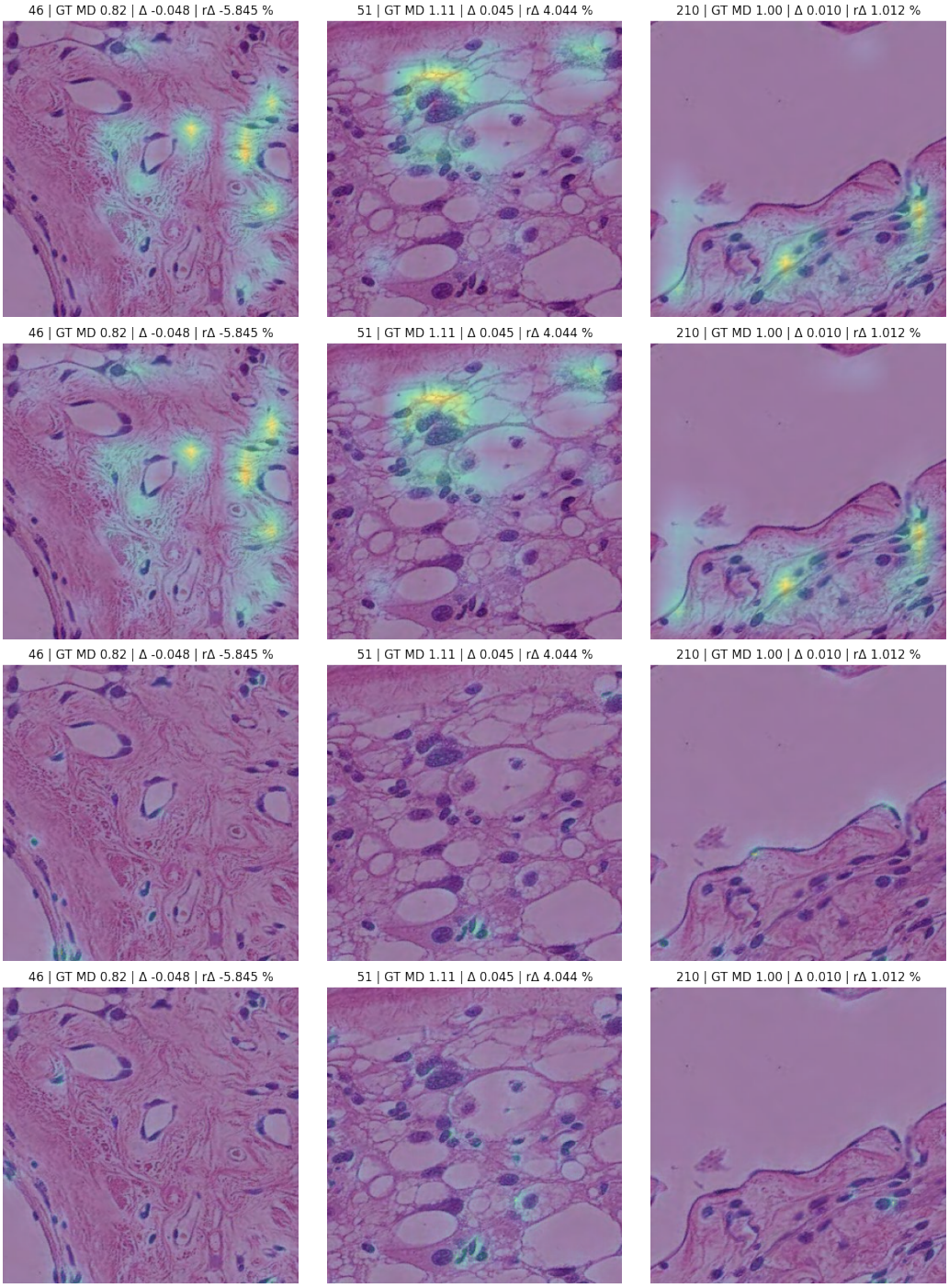

Figure 4.3: This figure shows three randomly picked patches with a XAI visualisation overlays (alpha=0.6, colour map=*viridis*) for MD target. Rows represent different methods. From top: Grad-CAM++, LayerCAM, SmoothGrad ($\sigma = 0.3, n = 30$), SmoothGrad ($\sigma = 0.2, n = 30$). The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

4.2.3 Target FA

The examples of the visualisation for target FA can be seen in Figure 4.4 and more in Appendices B.9, B.10, B.11 and B.12.

We can see the CAM-based methods, again, produce similar visualisations. We observe that unlike with MD, the visualisations tend to have repetitive patterns near the edges, primarily for higher values of FA ($\geq \quad 0.4$). That is visible on patches 101 and 124. In contrast, this does not occur on lower values of FA on patch 47. That is surprising, and we came up with the following justification.

When we compared multiple image patches with high FA in the qualitative analysis (consider example patches 101 and 124), we noticed higher density of the nuclei in most images, as well as the prolonged shape of the nuclei. Since this is mostly consistent across the whole image patch, it would make sense if the model subsampled the features. That would explain the highlighted areas along the image border. It could also be a similar normalisation consequence as mentioned above.

For higher FA, the SmoothGrad seems to focus not only on the nuclei but also on the space between the nuclei. We can notice the highlighted lines in the narrow space among nuclei. That is more emphasised with the $\sigma = 0.2, n = 30$ setting.

Low values of FA tend to visualise similarly as MD – focusing on nuclei and the close area around. Furthermore, as with MD, there is still a lot of uncertainty in the interpretation. It seems the shape, size, and position of the nuclei could play a role as they could form some higher-level histology features. If we recall the test set R^2 score of the model, both MD and FA were very high, meaning the model has learned some general patterns. However, the utilised methods did not fully reveal them.

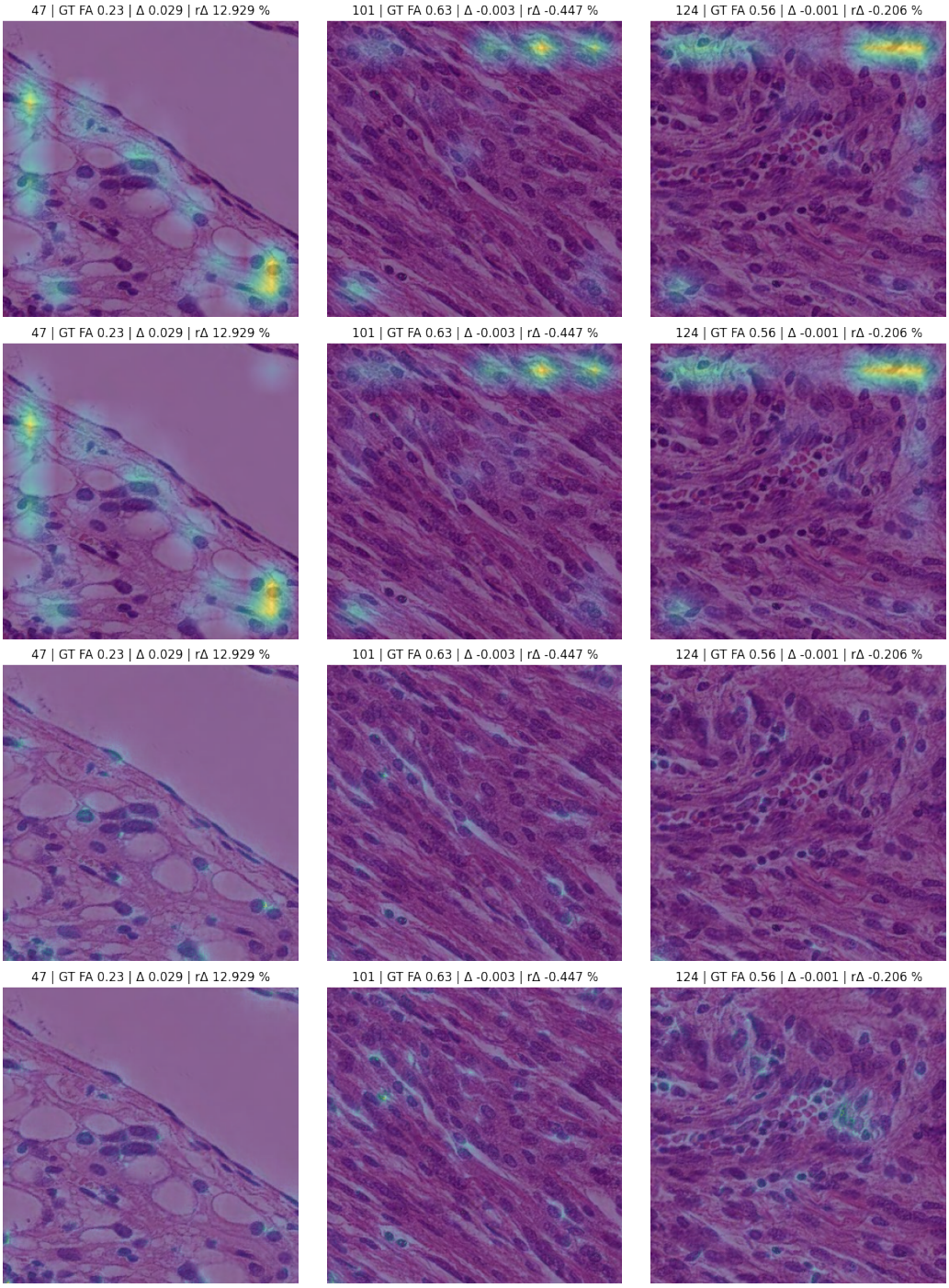

Figure 4.4: This figure shows three randomly picked patches with a XAI visualisation overlays (alpha=0.6, colour map=*viridis*) for FA target. Rows represent different methods. From top: Grad-CAM++, LayerCAM, SmoothGrad ($\sigma = 0.3, n = 30$), SmoothGrad ($\sigma = 0.2, n = 30$). The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

Conclusion

This thesis followed up on research conducted by Brabec et al. [6] where they noticed the CNNs could better predict MD and FA from histology patches than from CD and IA. There were two objectives of this work. First, to use the same data as in the initial study and train a CNN that predicts MD and FA reasonably well. Second, to utilise three SOTA methods from the XAI field and try to reveal what histology features are most influential in increasing MD and FA. That is still unknown among medical scientists, and any new hints would be valuable.

Our approach was to combine two histology samples and utilise multi-task learning. A single network with outputs MD and FA turned out to be beneficial for two reasons. It worked as a regularisation and contributed to achieving outstanding results. Secondly, it sped up the training process as we only fine-tuned one network instead of two.

For the XAI part, based on a conducted survey, we picked three heavily used and well-performant methods for the CNN explanation: Grad-CAM++, LayerCAM and SmoothGrad. We built a web application for qualitative analysis to compare visualisations from different methods and for different targets. To gain more confidence in the visualisations before interpretation, we also trained a network to predict CC, where we know exactly what histology features the model should take into account.

Contributions

We managed to train a network without overfitting or underfitting issues and still with excellent performance in terms of the R^2 evaluation metric. Achieving more than 98% on both MD and FA targets on all training data splits (train, validation, test) is a considerable success. Moreover, our proposed CNN significantly outperforms all the models presented in the initial study conducted by Brabec et al. [6] despite our model extrapolates. Their approach was to train a network for each histology sample separately. In contrast, we combined two of them and the rest was predicted in an out-of-sample fashion. The observable improvement of the R^2 was generally by tens of percentage points, suggesting decent generalisation capabilities of our network.

The second contribution is the explored XAI visualisations applied to microscopic histology imaging. Even though the utilised methods turned out to work less on microscopic objects rather than macroscopic ones, we are confident to conclude that the core nuclei play a role in increased MD and the cell density in FA. Nonetheless, the details of the role, such as the shape, size, and position of the nuclei, remain uncovered.

Future Work

As mentioned above, the utilised XAI methods (especially CAM-based) seem to work very well on macroscopic objects with fewer instances but rather fail on microscopic with hundreds of object instances. Future works could explore additional methods that would not suffer from this issue, which could unveil additional details of the cell-MD/FA relationship.

Bibliography

1. MOLNAR, Christoph. *Interpretable Machine Learning*. Leanpub, 2020. ISBN 9780244768522. Available also from: https://books.google.cz/books?id=jBm3DwAAQBAJ.
2. CASEY ROSS, Ike Swetlitz. *IBM's Watson supercomputer recommended "unsafe and incorrect" cancer treatments, internal documents show*. 2023. Available also from: https://www.statnews.com/2018/07/25/ibm-watson-recommended-unsafe-incorrect-treatments/.
3. DASTIN, Jeffrey. *Amazon scraps secret AI recruiting tool that showed bias against women*. Thomson Reuters, 2018. Available also from: https://www.reuters.com/article/us-amazon-com-jobs-automation-insight-idUSKCN1MK08G.
4. VINCENT, James. *Google "fixed" its racist algorithm by removing gorillas from its image-labeling tech*. The Verge, 2018. Available also from: https://www.theverge.com/2018/1/12/16882408/google-racist-gorillas-photo-recognition-algorithm-ai.
5. COUNCIL OF THE EUROPEAN UNION, European Parliament. *Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA relevance)*. 2016. No. 119.

6. BRABEC, Jan; FRIEDJUNGOVÁ, Magda; VAŠATA, Daniel; ENGLUND, Elisabet; BENGZON, Johan; KNUTSSON, Linda; SZCZEPANKIEWICZ, Filip; VAN WESTEN, Danielle; SUNDGREN, Pia C.; NILSSON, Markus. Meningioma microstructure assessed by diffusion MRI: An investigation of the source of mean diffusivity and fractional anisotropy by quantitative histology. *NeuroImage: Clinical*. 2023, vol. 37, pp. 103365. ISSN 2213-1582. Available from DOI: https://doi.org/10.1016/j.nicl.2023.103365.

7. BASSER, Peter J; MATTIELLO, James; LEBIHAN, Denis. MR diffusion tensor spectroscopy and imaging. *Biophysical journal*. 1994, vol. 66, no. 1, pp. 259–267.

8. STEJSKAL, Edward O; TANNER, John E. Spin diffusion measurements: spin echoes in the presence of a time-dependent field gradient. *The journal of chemical physics*. 1965, vol. 42, no. 1, pp. 288–292.

9. TAN, Mingxing; LE, Quoc. Efficientnetv2: Smaller models and faster training. In: *International conference on machine learning*. 2021, pp. 10096–10106.

10. BRABEC, Jan; ENGLUND, Elisabet; BENGZON, Johan; SZCZEPANKIEWICZ, Filip; WESTEN, Danielle van; SUNDGREN, Pia; NILSSON, Markus. Coregistered H&E- and VEGF-stained histology slides with diffusion tensor imaging data at 200 μm resolution in meningioma tumors. 2023. Available from DOI: 10.23698/AIDA/MICROMEN.

11. MURPHY, Kevin P. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012. Adaptive Computation and Machine Learning series. ISBN 9780262018029. Available also from: https://books.google.cz/books?id=NZP6AQAAQBAJ.

12. ALI, Sajid; ABUHMED, Tamer; EL-SAPPAGH, Shaker; MUHAMMAD, Khan; ALONSO-MORAL, Jose M.; CONFALONIERI, Roberto; GUIDOTTI, Riccardo; DEL SER, Javier; DÍAZ-RODRÍGUEZ, Natalia; HERRERA, Francisco. Explainable Artificial Intelligence (XAI): What we know and what is left to attain Trustworthy Artificial Intelligence. *Information Fusion*. 2023, vol. 99, pp. 101805. ISSN 1566-2535. Available from DOI: https://doi.org/10.1016/j.inffus.2023.101805.

13. ARRIETA, Alejandro Barredo; DÍAZ-RODRÍGUEZ, Natalia; SER, Javier Del; BENNETOT, Adrien; TABIK, Siham; BARBADO, Alberto; GARCÍA, Salvador; GIL-LÓPEZ, Sergio; MOLINA, Daniel; BENJAMINS, Richard; CHATILA, Raja; HERRERA, Francisco. *Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI*. 2019. Available from arXiv: 1910.10045 [cs.AI].

14. VAN DER VELDEN, Bas H.M.; KUIJF, Hugo J.; GILHUIJS, Kenneth G.A.; VIERGEVER, Max A. Explainable artificial intelligence (XAI) in deep learning-based medical image analysis. *Medical Image Analysis*. 2022, vol. 79, pp. 102470. ISSN 1361-8415. Available from DOI: https://doi.org/10.1016/j.media.2022.102470.

15. ZHOU, B.; KHOSLA, A.; LAPEDRIZA, A.; OLIVA, A.; TORRALBA, A. Learning Deep Features for Discriminative Localization. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, 2016, pp. 2921–2929. ISSN 1063-6919. Available from DOI: 10.1109/CVPR.2016.319.

16. SELVARAJU, Ramprasaath R.; DAS, Abhishek; VEDANTAM, Ramakrishna; COGSWELL, Michael; PARIKH, Devi; BATRA, Dhruv. Grad-CAM: Why did you say that? Visual Explanations from Deep Networks via Gradient-based Localization. *CoRR*. 2016, vol. abs/1610.02391. Available from arXiv: 1610.02391.

17. JETLEY, Saumya; LORD, Nicholas A.; LEE, Namhoon; TORR, Philip H. S. *Learn To Pay Attention*. 2018. Available from arXiv: 1804.02391 [cs.CV].

18. SELVARAJU, Ramprasaath R.; COGSWELL, Michael; DAS, Abhishek; VEDANTAM, Ramakrishna; PARIKH, Devi; BATRA, Dhruv. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. *International Journal of Computer Vision*. 2019, vol. 128, no. 2, pp. 336–359. ISSN 1573-1405. Available from DOI: 10.1007/s11263-019-01228-7.

19. BACH, Sebastian; BINDER, Alexander; MONTAVON, Grégoire; KLAUSCHEN, Frederick; MÜLLER, Klaus-Robert; SAMEK, Wojciech. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*. 2015, vol. 10, no. 7, pp. e0130140.

20. ZEILER, Matthew D; FERGUS, Rob. Visualizing and understanding convolutional networks. In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*. 2014, pp. 818–833.

21. ZEINELDIN, Ramy A.; KARAR, Mohamed E.; ELSHAER, Ziad; COBURGER, Jan; WIRTZ, Christian R.; BURGERT, Oliver; MATHIS-ULLRICH, Franziska. Explainability of deep neural networks for MRI analysis of brain tumors. *International Journal of Computer Assisted Radiology and Surgery*. 2022, vol. 17, no. 9, pp. 1673–1683. ISSN 1861-6429. Available from DOI: 10.1007/s11548-022-02619-x.

22. SIMONYAN, Karen; VEDALDI, Andrea; ZISSERMAN, Andrew. *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*. 2014. Available from arXiv: 1312.6034 [cs.CV].

23. SPRINGENBERG, Jost Tobias; DOSOVITSKIY, Alexey; BROX, Thomas; RIEDMILLER, Martin. *Striving for Simplicity: The All Convolutional Net*. 2015. Available from arXiv: 1412.6806 [cs.LG].

24. SUNDARARAJAN, Mukund; TALY, Ankur; YAN, Qiqi. *Axiomatic Attribution for Deep Networks*. 2017. Available from arXiv: 1703.01365 [cs.LG].

25. SMILKOV, Daniel; THORAT, Nikhil; KIM, Been; VIÉGAS, Fernanda; WATTENBERG, Martin. SmoothGrad: removing noise by adding noise. 2017.

26. RGUIBI, Zakaria; HAJAMI, Abdelmajid; ZITOUNI, Dya; ELQARAOUI, Amine; BEDRAOUI, Anas. CXAI: Explaining Convolutional Neural Networks for Medical Imaging Diagnostic. *Electronics*. 2022, vol. 11, no. 11. ISSN 2079-9292. Available also from: https://www.mdpi.com/2079-9292/11/11/1775.

27. CHATTOPADHYAY, Aditya; SARKAR, Anirban; HOWLADER, Prantik; BALASUBRAMANIAN, Vineeth N. Grad-CAM++: Generalized Gradient-based Visual Explanations for Deep Convolutional Networks. *CoRR*. 2017, vol. abs/1710.11063. Available from arXiv: 1710.11063.

28. JOSHUA, Eali Stephen Neal; CHAKKRAVARTHY, Midhun; BHATTACHARYYA, Debnath. Lung Cancer Detection Using Improvised Grad-Cam++ With 3D

CNN Class Activation. In: SAHA, Sanjoy Kumar; PANG, Paul S.; BHATTACHARYYA, Debnath (eds.). *Smart Technologies in Data Science and Communication*. Singapore: Springer Singapore, 2021, pp. 55–69.

29. RAHMAN, Tawsifur; KHANDAKAR, Amith; QIBLAWEY, Yazan; TAHIR, Anas; KIRANYAZ, Serkan; ABUL KASHEM, Saad Bin; ISLAM, Mohammad Tariqul; AL MAADEED, Somaya; ZUGHAIER, Susu M.; KHAN, Muhammad Salman; CHOWDHURY, Muhammad E.H. Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images. *Computers in Biology and Medicine*. 2021, vol. 132, pp. 104319. ISSN 0010-4825. Available from DOI: `https://doi.org/10.1016/j.compbiomed.2021.104319`.

30. WANG, Haofan; WANG, Zifan; DU, Mengnan; YANG, Fan; ZHANG, Zijian; DING, Sirui; MARDZIEL, Piotr; HU, Xia. Score-CAM: Score-weighted visual explanations for convolutional neural networks. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*. 2020, pp. 24–25.

31. LIU, Shangwang; CAI, Tongbo; TANG, Xiufang; ZHANG, Yangyang; WANG, Changgeng. COVID-19 diagnosis via chest X-ray image classification based on multiscale class residual attention. *Computers in Biology and Medicine*. 2022, vol. 149, pp. 106065. ISSN 0010-4825. Available from DOI: `https://doi.org/10.1016/j.compbiomed.2022.106065`.

32. JIANG, Peng-Tao; ZHANG, Chang-Bin; HOU, Qibin; CHENG, Ming-Ming; WEI, Yunchao. LayerCAM: Exploring Hierarchical Class Activation Maps for Localization. *IEEE Transactions on Image Processing*. 2021, vol. 30, pp. 5875–5888. Available from DOI: `10.1109/TIP.2021.3089943`.

33. RUSSAKOVSKY, Olga; DENG, Jia; SU, Hao; KRAUSE, Jonathan; SATHEESH, Sanjeev; MA, Sean; HUANG, Zhiheng; KARPATHY, Andrej; KHOSLA, Aditya; BERNSTEIN, Michael; BERG, Alexander C.; FEI-FEI, Li. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*. 2015, vol. 115, no. 3, pp. 211–252. Available from DOI: `10.1007/s11263-015-0816-y`.

34. MERKEL, Dirk. Docker: lightweight linux containers for consistent development and deployment. *Linux journal*. 2014, vol. 2014, no. 239, pp. 2.

35. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Available also from: https://www.tensorflow.org/. Software available from tensorflow.org.

36. CHOLLET, François et al. *Keras* [https://keras.io]. 2015.

37. RIASATIAN, Abtin; BABAIE, Morteza; MALEKI, Danial; KALRA, Shivam; VALIPOUR, Mojtaba; HEMATI, Sobhan; ZAVERI, Manit; SAFARPOOR, Amir; SHAFIEI, Sobhan; AFSHARI, Mehdi; RASOOLIJABERI, Maral; SIKAROUDI, Milad; ADNAN, Mohd; SHAH, Sultaan; CHOI, Charles; DAMASKINOS, Savvas; CAMPBELL, Clinton JV; DIAMANDIS, Phedias; PANTANOWITZ, Liron; KASHANI, Hany; GHODSI, Ali; TIZHOOSH, H.R. Fine-Tuning and training of densenet for histopathology image representation using TCGA diagnostic slides. *Medical Image Analysis*. 2021, vol. 70, pp. 102032. ISSN 1361-8415. Available from DOI: https://doi.org/10.1016/j.media.2021.102032.

38. LOSHCHILOV, Ilya; HUTTER, Frank. *Decoupled Weight Decay Regularization*. 2019. Available from arXiv: 1711.05101 [cs.LG].

39. KUBOTA, Yasuhiro. *tf-keras-vis*. 2023. Version 0.8.6. Available also from: https://keisen.github.io/tf-keras-vis-docs/.

Appendix A

List of Acronyms

AI	artificial intelligence
ANN	artificial neural network
CAM	class activation maps
CC	cell nuclei count
CD	cell density
CNN	convolutional neural network
DTI	diffusion tensor imaging
FA	fractional anisotropy
GAP	global average pooling
IA	image anisotropy
MD	mean diffusivity
ML	machine learning
MRI	magnetic resonance imaging
MSE	mean square error
R^2	coefficient of determination
SOTA	state-of-the-art
XAI	explainable artificial intelligence
XLA	accelerated linear algebra

Appendix B

XAI visualisations

In the following pages, we present 12 randomly picked image patches for each target in four figures. Each figure has an overlay with a different XAI method.

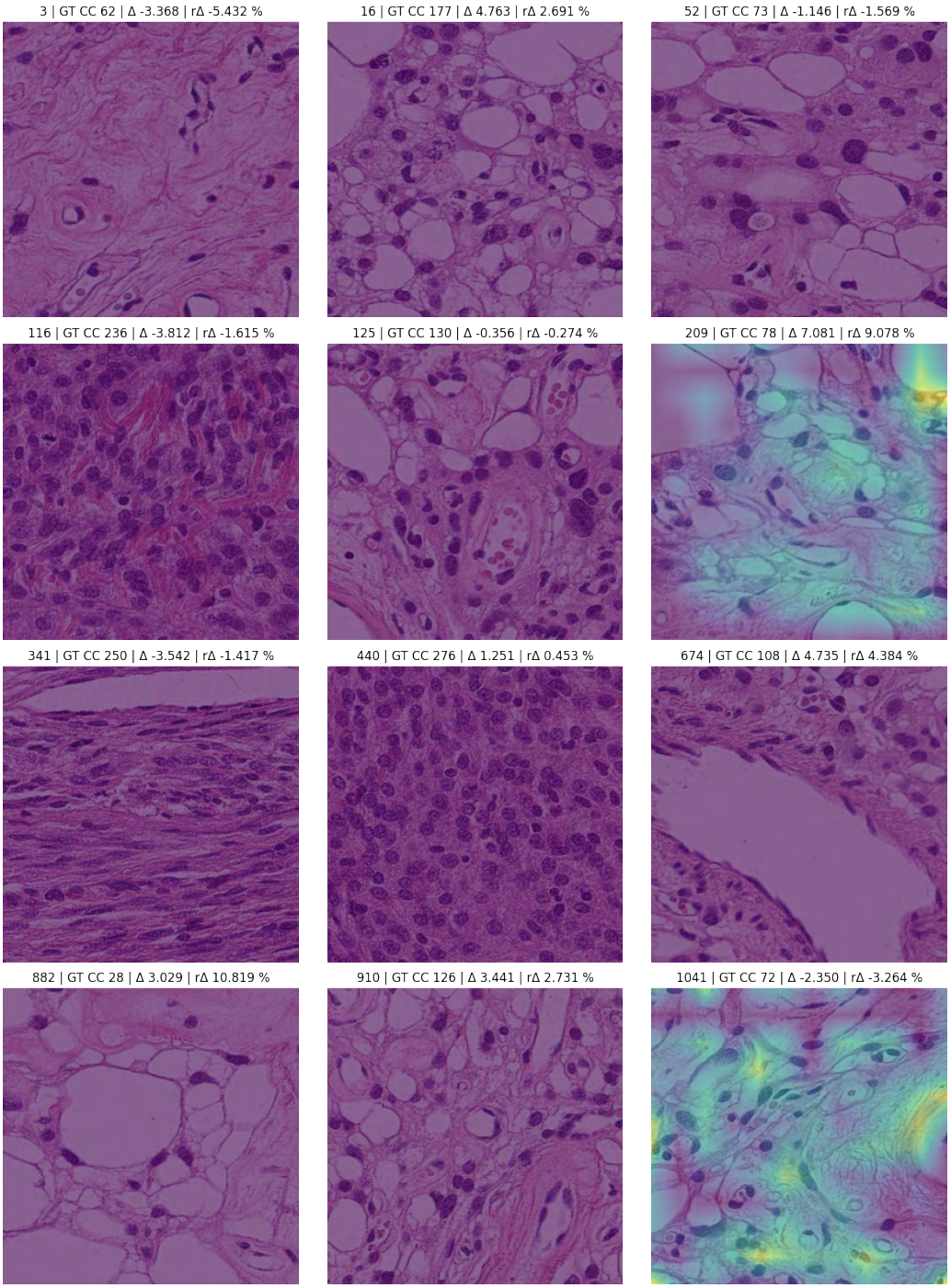

Figure B.1: This figure shows 12 randomly picked patches with a Grad-CAM++ visualisation overlay (alpha=0.6, colour map=*viridis*) for CC target. The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

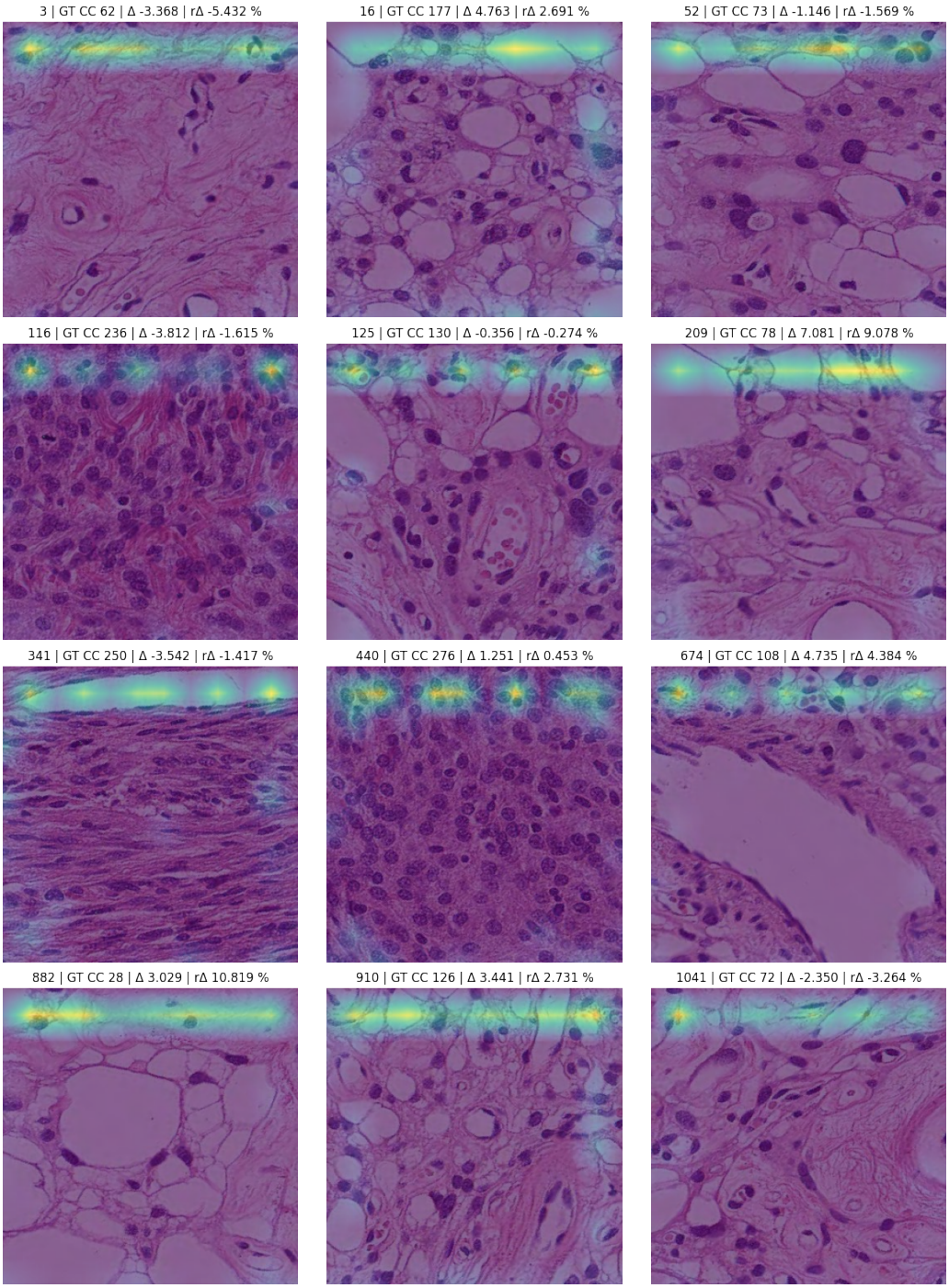

Figure B.2: This figure shows 12 randomly picked patches with a LayerCAM visualisation overlay (alpha=0.6, colour map=*viridis*) for CC target. The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

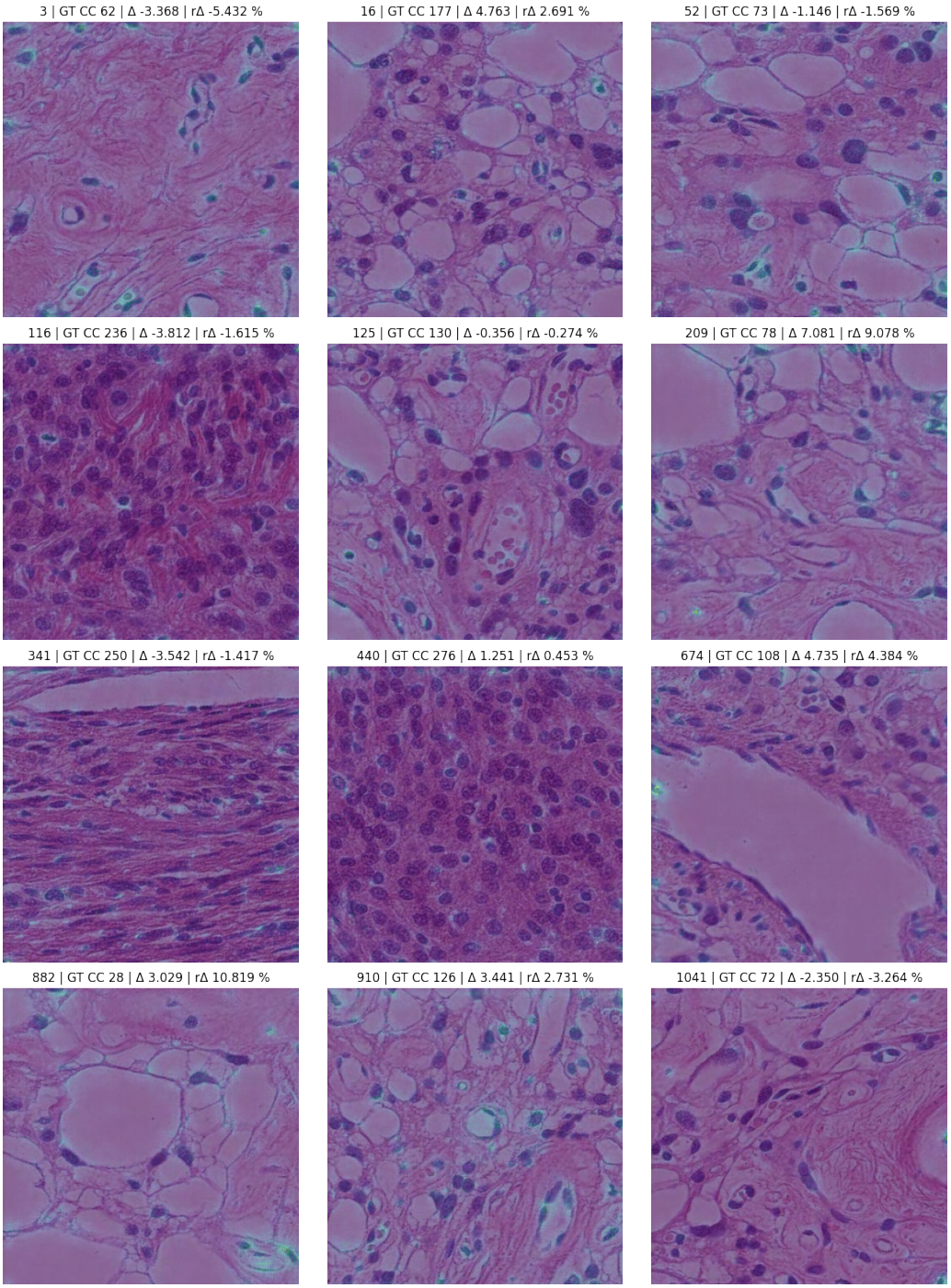

Figure B.3: This figure shows 12 randomly picked patches with a SmoothGrad ($\sigma = 0.2, n = 30$) visualisation overlay (alpha=0.6, colour map=*viridis*) for CC target. The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

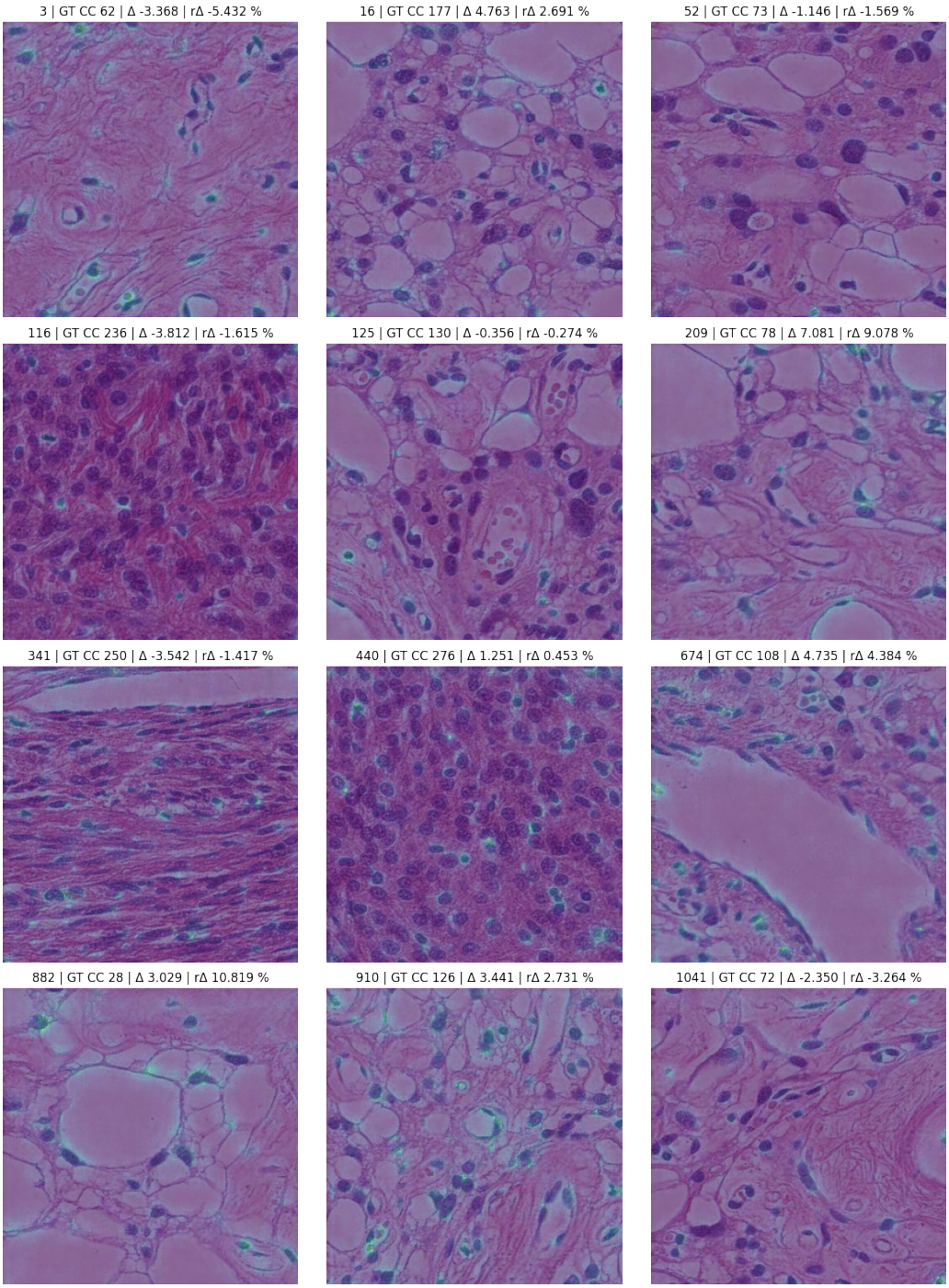

Figure B.4: This figure shows 12 randomly picked patches with a SmoothGrad ($\sigma = 0.3, n = 30$) visualisation overlay (alpha=0.6, colour map=*viridis*) for CC target. The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

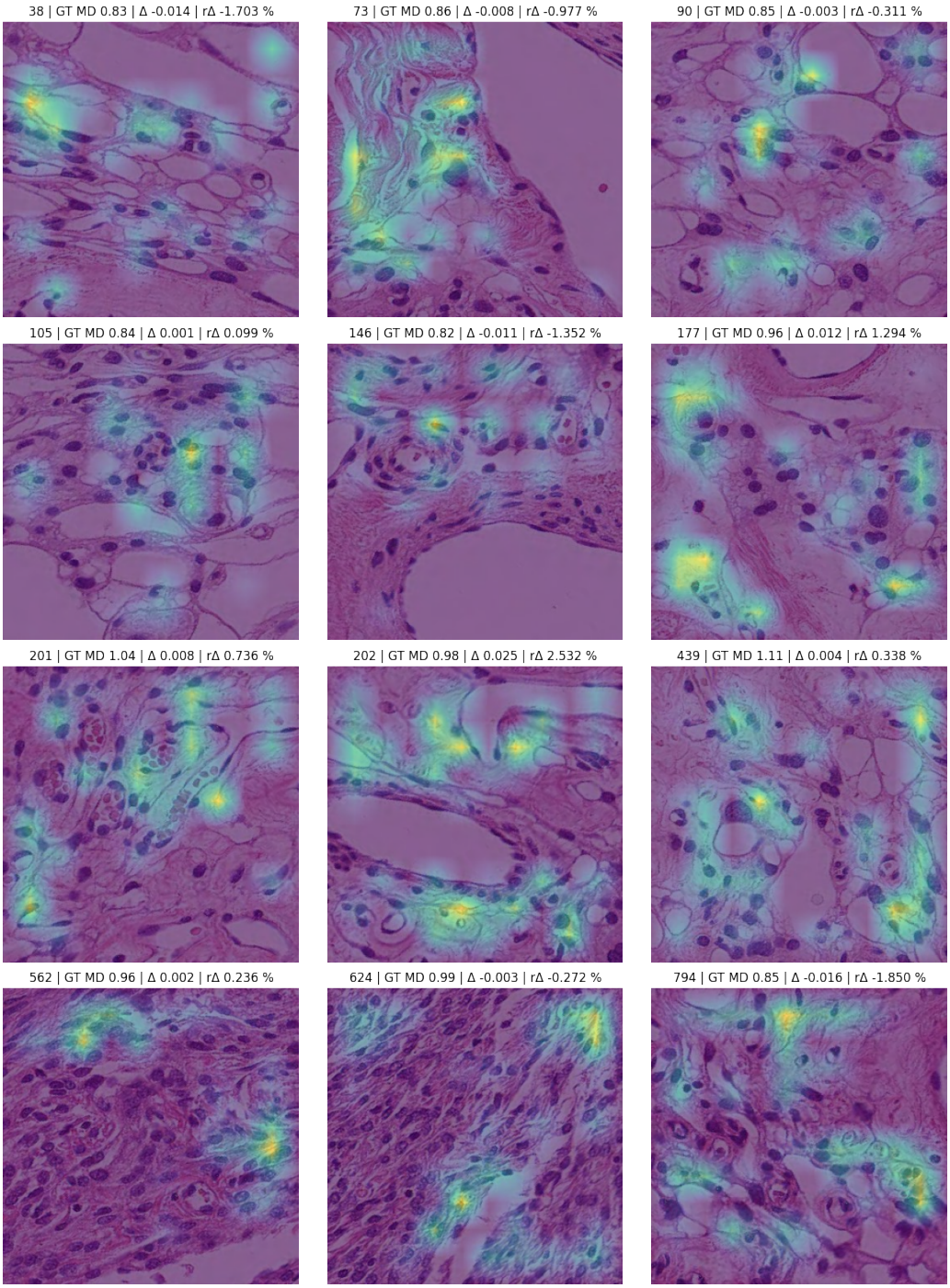

Figure B.5: This figure shows 12 randomly picked patches with a Grad-CAM++ visualisation overlay (alpha=0.6, colour map=*viridis*) for MD target. The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

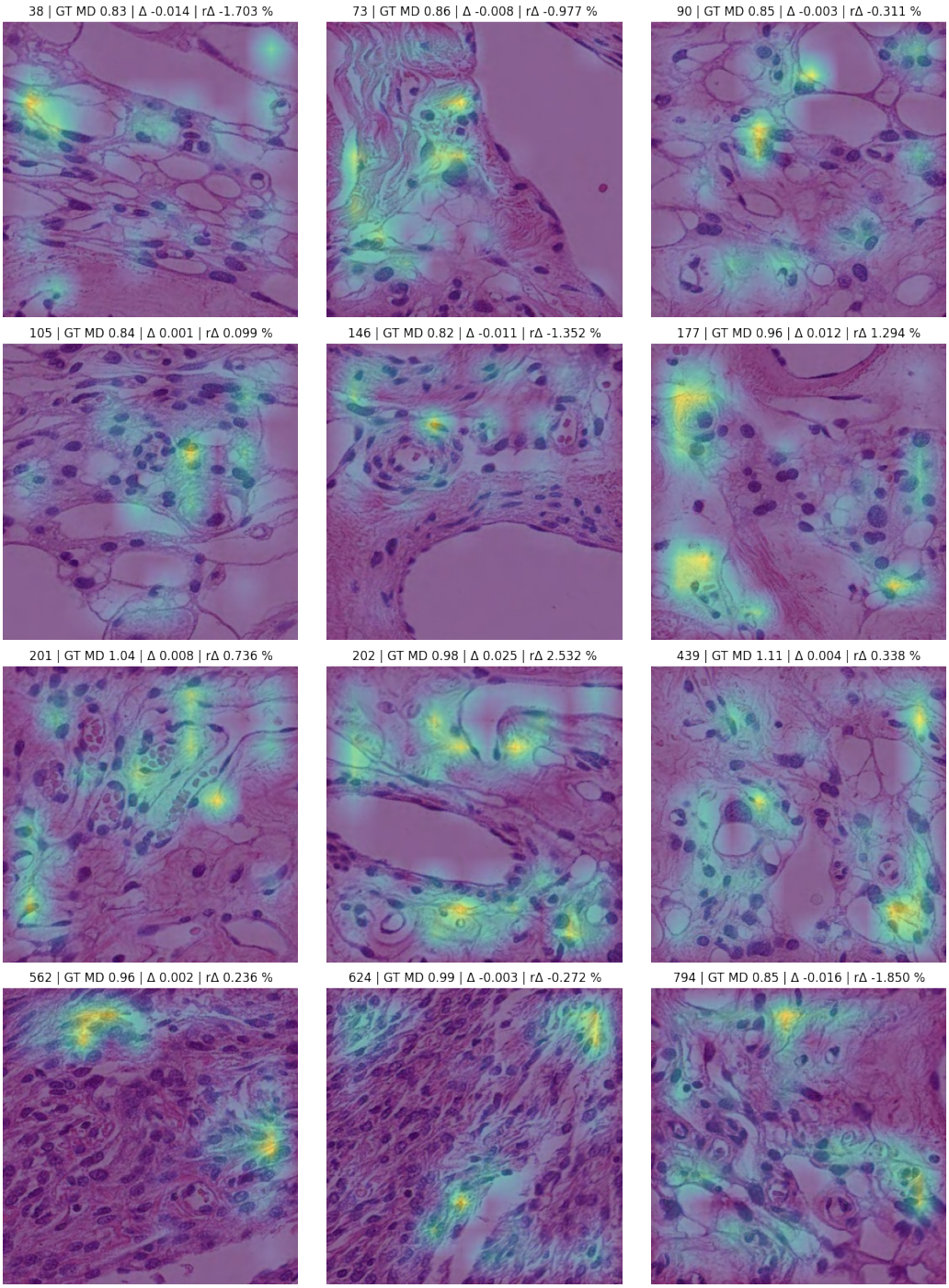

Figure B.6: This figure shows 12 randomly picked patches with a LayerCAM visualisation overlay (alpha=0.6, colour map=*viridis*) for MD target. The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

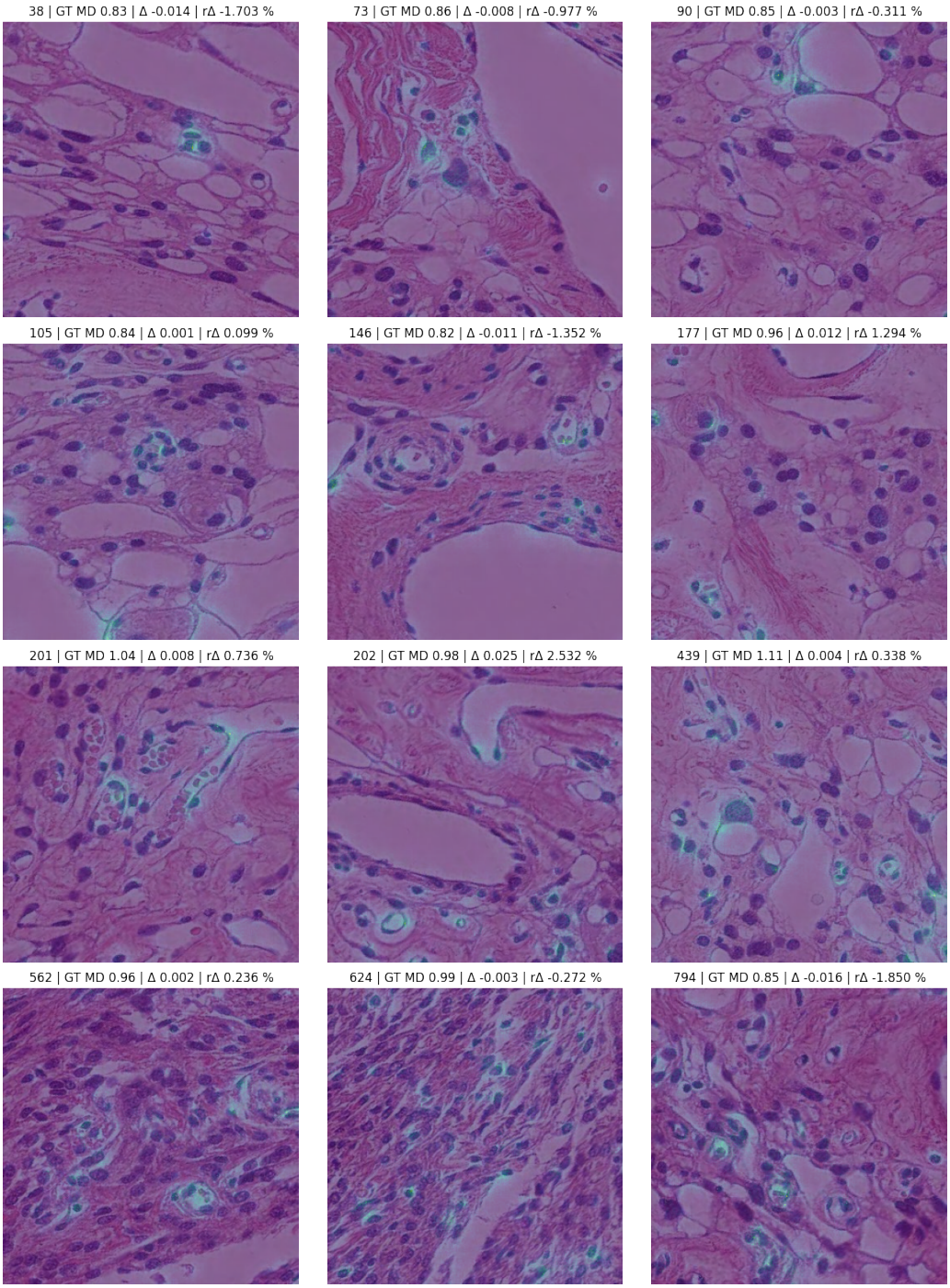

Figure B.7: This figure shows 12 randomly picked patches with a SmoothGrad ($\sigma = 0.2, n = 30$) visualisation overlay (alpha=0.6, colour map=*viridis*) for MD target. The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

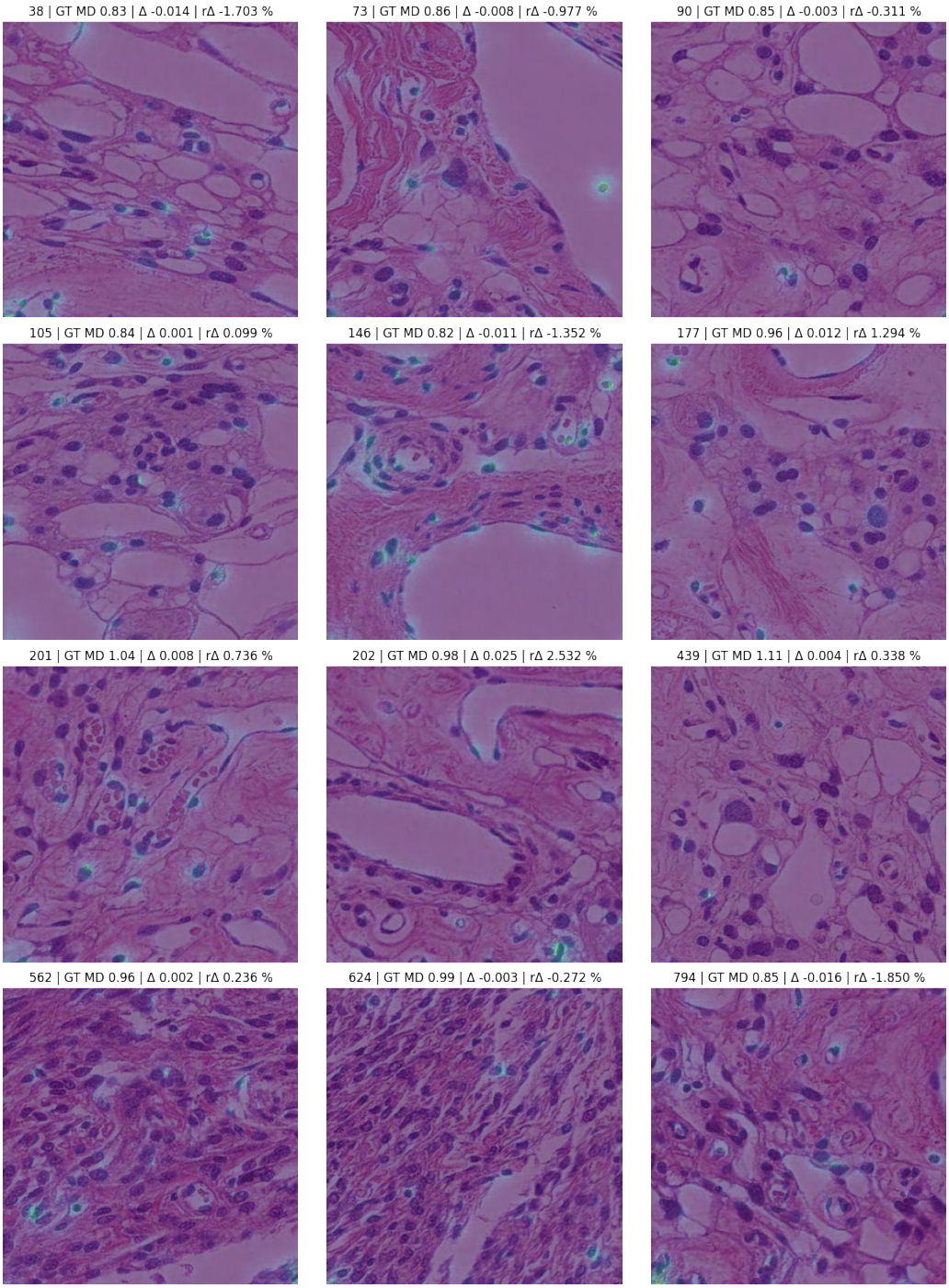

Figure B.8: This figure shows 12 randomly picked patches with a SmoothGrad ($\sigma = 0.3, n = 30$) visualisation overlay (alpha=0.6, colour map=*viridis*) for MD target. The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

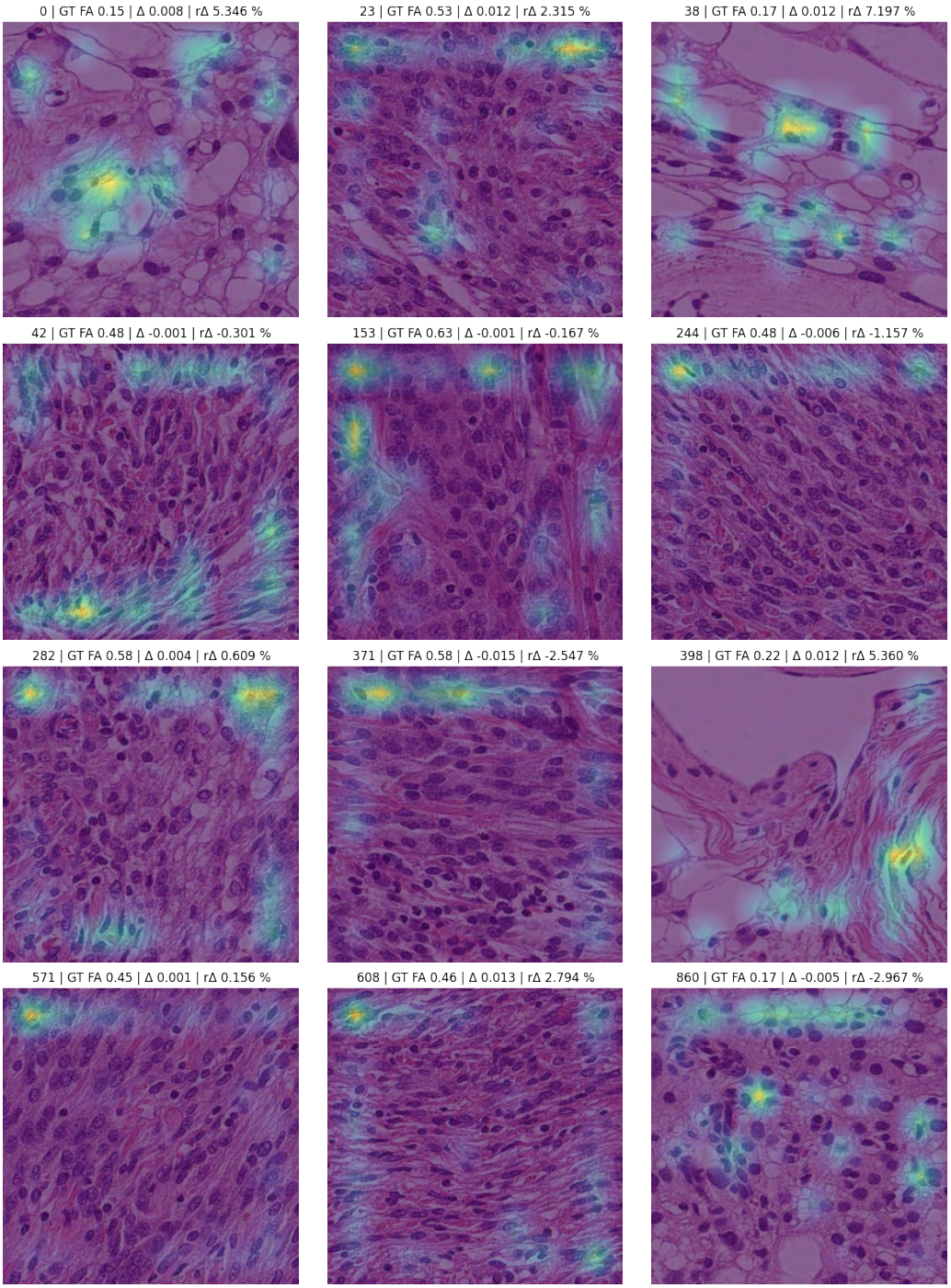

Figure B.9: This figure shows 12 randomly picked patches with a Grad-CAM++ visualisation overlay (alpha=0.6, colour map=*viridis*) for FA target. The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

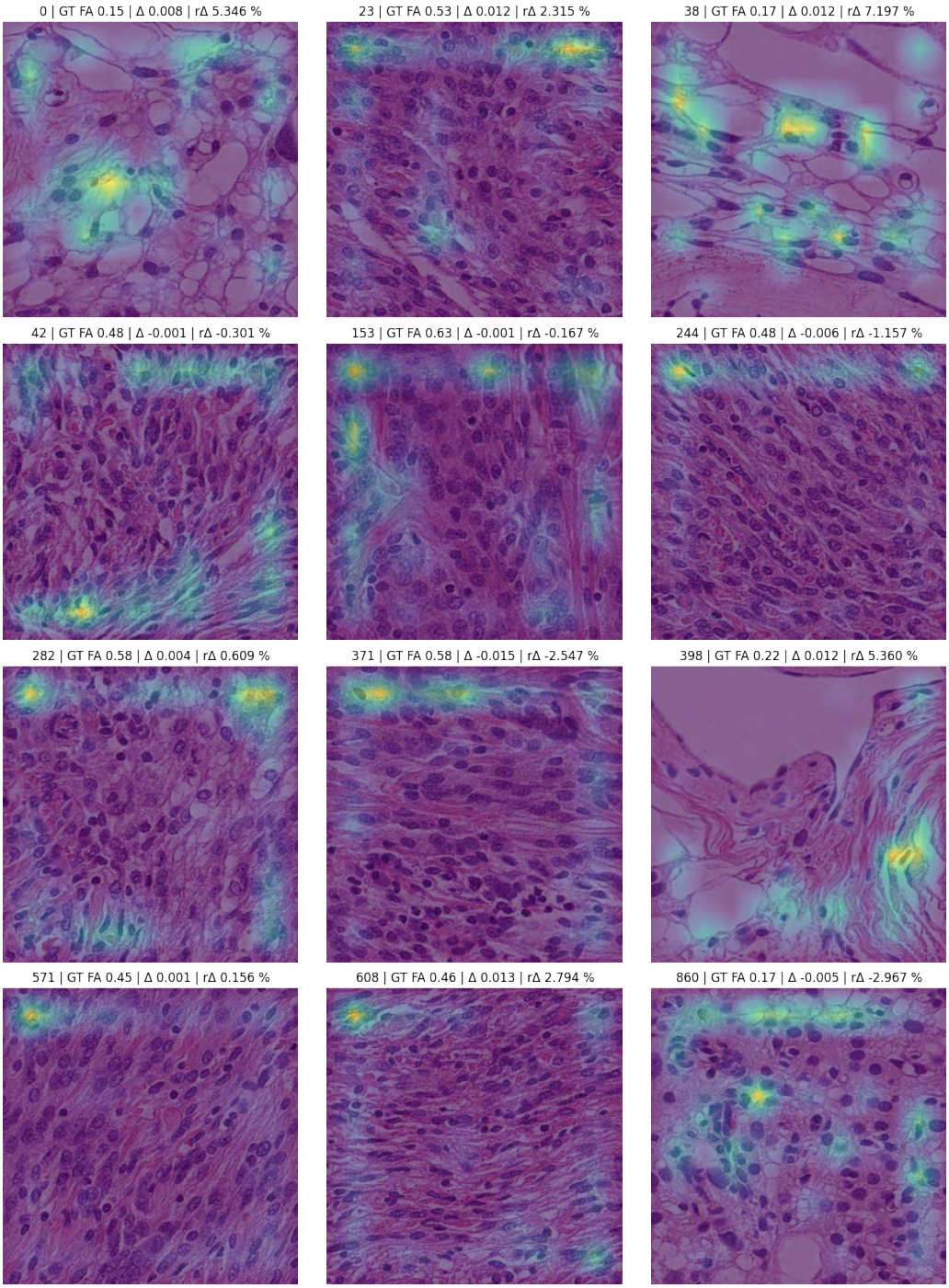

Figure B.10: This figure shows 12 randomly picked patches with a LayerCAM visualisation overlay (alpha=0.6, colour map=*viridis*) for FA target. The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

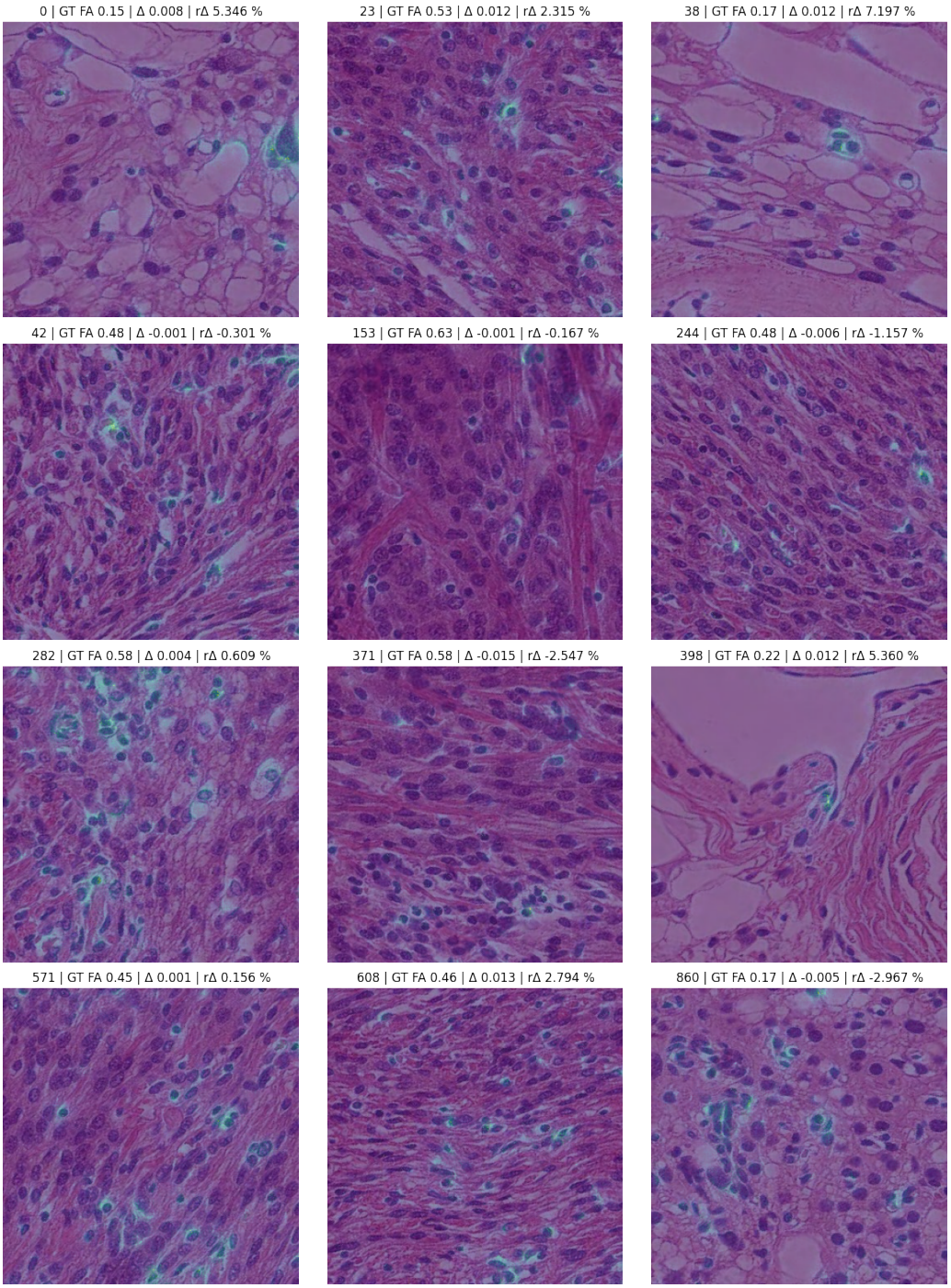

Figure B.11: This figure shows 12 randomly picked patches with a SmoothGrad ($\sigma = 0.2, n = 30$) visualisation overlay (alpha=0.6, colour map=*viridis*) for FA target. The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.

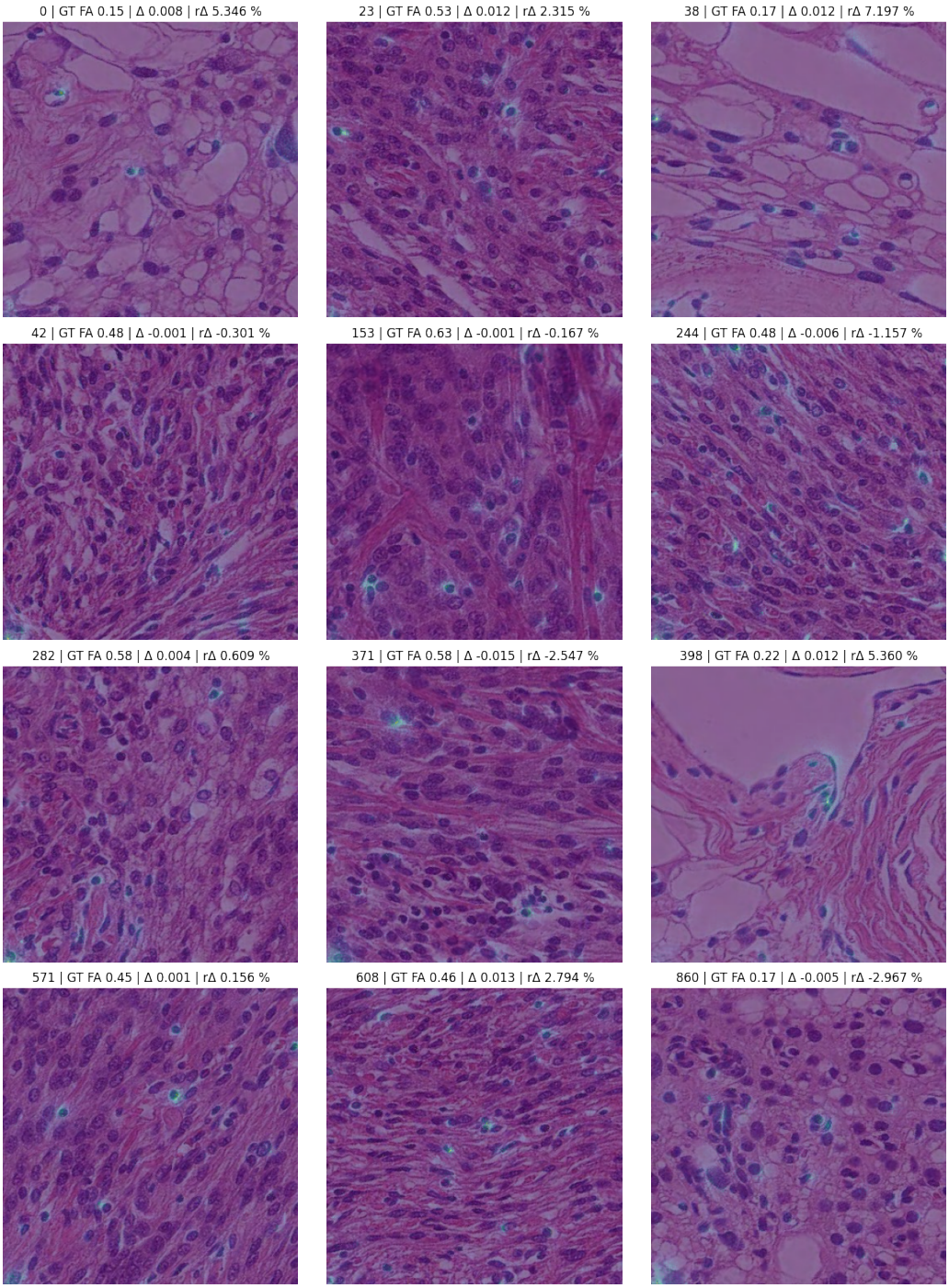

Figure B.12: This figure shows 12 randomly picked patches with a SmoothGrad ($\sigma = 0.3, n = 30$) visualisation overlay (alpha=0.6, colour map=*viridis*) for FA target. The header of each image denotes the index in the test set, ground truth value, and absolute and relative error of the predicted value.