Explainability pipelines for an end-to-end understanding of 3D Covid-19 CT scans

Master of Research project industry-sponsored by AstraZeneca
August 2021
(Standard project)

Author:
Filip Makraduli

Supervised by:
Elsa Angelini, Arijit Patra

Statement of originality

This is to certify that to the best of my knowledge, the content of this thesis is my own work. This thesis has not been submitted for any degree or other purposes. Any other work or assistance has been properly and clearly cited. I would like to acknowledge Aakash Kumar Nain, F.I.Tushar (https://github.com/fitushar/3D-GuidedGradCAM-for-Medical-Imaging) and Tiba Razmi (https://medium.com/@tibastar) for their work which helped in my understanding and development of certain aspects of my projects. As well I would thank Google Colab's free cloud computing services and Keras's library free 3D imaging tutorials. Furthermore, a small portion of the introductory topics on explainability has been also discussed in my Grant proposal student paper from the same year.

Table of contents

Abstract 2
Introduction 3
What is explainability and why it is needed
Image explainability and COVID-19 detection
Novelty and hypothesis in this project
Algorithm and methods 9
Model backends, pre-processings
Model explainability
Project workflow
Results 16
3D explainability results
Evaluations of heatmaps
2D explainability results
Model backend results
Discussion 24
Strengths and weaknesses of the methods
Impact to other literature and novelty
Conclusion
Future work 28
Bibliography **28**
Appendix A **30**

Abstract

In the field of medical diagnostics, deep learning approaches have proven to be accurate tools that yield valuable findings. They've been employed by medical experts as a form of assistive technology, particularly in imaging segmentation and classification challenges. In some diagnostic tasks, they have also been able to surpass human's performance. The 'black box' structure of these models is of major concern. The goal of explainability research is to determine which characteristics have the greatest influence on a model's choice. This project implements a pipeline of model agnostic explainability techniques both in 2D and 3D for medical images. The use-case data is a collection of CT chest scans as 3D nifti files from the MosMed COVID-19 publicly available dataset. The task being explained in 2D explainability is an implementation of the SHAP algorithm (SHapley Additive exPlanations). The 3D explainability is an implementation of a gradient class activation map algorithm using Tensorflow and Keras. This algorithm generates heat maps that highlight class activation regions. Two backend models are used for testing the explainability methods. One is a 3D CT binary classification model detecting COVID-19, used for generating 3D heatmaps, and the other is a pneumonia VGG16 based model calculating relevant image patches using the SHAP algorithm. By analyzing the resulting heat maps and outputs, this explainability pipeline aids the efforts of model understanding, debugging, and trust in AI systems across various stakeholders. This has been shown through the results, discussing many insights (including lung abnormality activations) that would otherwise be missed if an explainability component was not used. If applied, this pipeline limits bias, leads to higher-quality model development, and documenting. As more and more AI systems are applied in society, the importance of adding explainability as a common task in the development of these systems will only get greater. The project tackles an urgent issue such as COVID-19 detection and appeals to the effort of making COVID-19 research more applicable in clinical settings.

1. Introduction

In this introductory chapter, there are three sections. In the first section, the background of the field of explainability and interpretability is presented. It includes some important definitions and concepts. The second section looks into literature and scientific work done in imaging explainability as this area is in the narrower scope of the project. The third section describes the novelty and what we have done in our work. It also includes the hypothesis and aims of my project with added background about the data used.

1.1 What are explainability and interpretability and why are they needed ?

In several areas, deep learning and machine learning technologies have shown to be useful and accurate tools. These techniques have proven to be effective in fields such as medical, technology, law, and finance. Deep learning models naturally make decisions like a “black box” with input and output, without too much detail of what is going on in between. This is a major source of concern. Humans find it difficult to visualise predictions since they are typically non-intuitive for humans. This lack of transparency and knowledge of predictions can result in deep learning models being misconstrued or in failure and unexpected real-world performance. In sensitive medical use cases, a circumstance like this can have an impact on stakeholders, business decision-making, customers, and even patients.

Explainable artificial intelligence (XAI) is a field in machine learning which addresses the understanding of decision making of artificial intelligence systems. This is an active research topic, especially the explanations for the aforementioned ‘black box’ models. Sometimes the term interpretable artificial intelligence is used interchangeably with explainable artificial intelligence, although there is a slight difference between the two. Interpretability tends to explain models which already have an existing decision-based structure such as decision trees, linear regression, random forests etc. Explainability studies the explanations of “black box” models such as deep neural networks. The study of Explainable AI (XAI) looks to recognize and understand ,how models make their decisions. A deeper insight into the naming schemes, terminology and taxonomy of explainability methods is mentioned in (Singh et al., 2020). In Figure 1.1 the main taxonomy of

Explainable AI methods is visualized. As this research is new, this terminology is not "set in stone" and is sometimes used interchangeably.

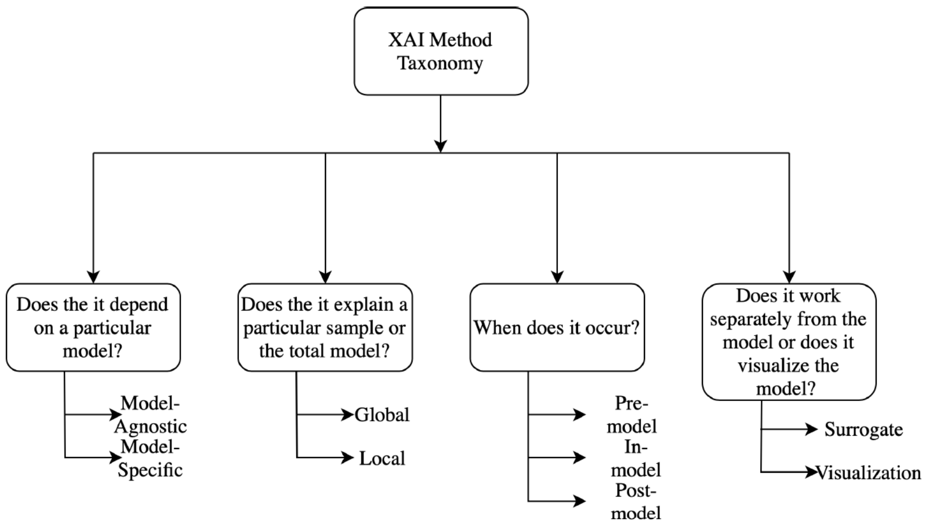
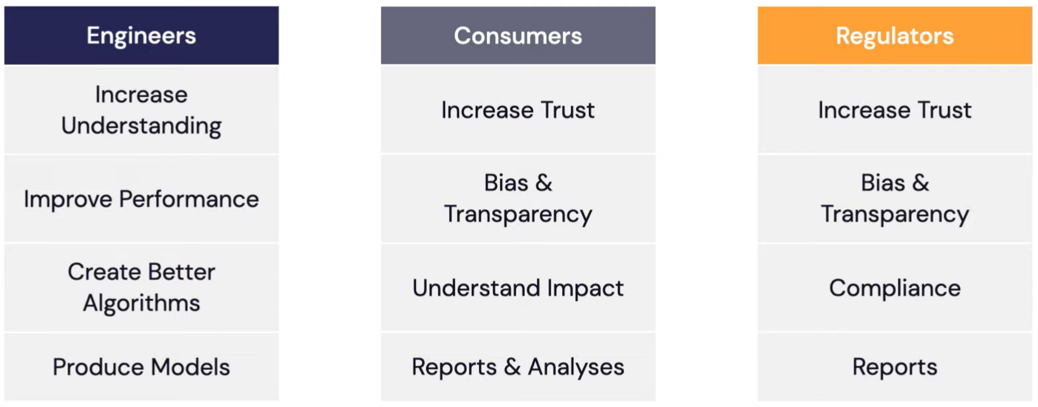

XAI Method Taxonomy
Does the it depend on a particular model?
Model-Agnostic
Model-Specific
Does the it explain a particular sample or the total model?
Global
Local
When does it occur?
Pre-model
In-model
Post-model
Does it work separately from the model or does it visualize the model?
Surrogate
Visualization

Fig 1.1 - Explainable AI main taxonomy from (Singh et al., 2020). The graph shows the types of explainability techniques in regards to their function and interaction with the model they are explaining. The techniques are divided into four groups.

Research in explainable AI is not new and has been present in the past. However, there has been a recent spike in research interest in explainable AI from 2015 onwards. This is attributed to the increasing numbers of applied machine learning systems deployed in various areas. As the inclusion of such systems becomes a bigger part of society, the role of explainability will get more recognition. Another broader survey of explainability methods is done in (Das & Rad, 2020).

Why is explainability needed?

Explainable AI has many use cases and roles. Figure 1.2 shows the three main stakeholder groups of explainable AI. This division is in terms of the role of explainability in different stakeholder groups of the AI systems.



Engineers	Consumers	Regulators
Increase Understanding	Increase Trust	Increase Trust
Improve Performance	Bias & Transparency	Bias & Transparency
Create Better Algorithms	Understand Impact	Compliance
Produce Models	Reports & Analyses	Reports

Fig 1.2. - Roles of explainable AI with different stakeholders. Engineers mainly use explainability to further improve the algorithms and better understand the inner workings of the systems. Consumers and Regulators are more concerned about trust and transparency for reports generation for compliance and impacts, source: Google ML tech talks.

The division of use cases is broader and explainability can have numerous use cases. Below is the outline of some notable use cases relevant to most AI systems:

- The main use case in the explanation of model predictions is to **support human decision-making** processes. More and more AI systems include humans as part of the decision making processes especially in medical applications (Fauw et al., 2018). Having model decisions that are reliable and transparent becomes a key aspect both for clinicians and patients alike.
- For engineers, a sizable use case of explainable AI is **debugging**. Understanding model decisions can help in **corrective actions** and adjustments to the algorithms. Explainability can also help in understanding dataset biases and skews thus aiding in data collection and model validation tasks (Xie et al., 2020).
- Explainable AI can have a meaningful role in the adoption of these technologies. This is the case with law regulators, who need to understand model decisions but also such understandings can enlist the **trust** and approval of end-users. This leads to increased approval of AI systems and more scenarios where AI systems can be applied.

Explainability and interpretability are becoming common steps in the development of machine learning applications. They are starting to take an important role on par with privacy, security, and other best practices when creating machine learning-based applications. This is notable in human first applications of deep learning (*Healthcare AI Systems That Put People at the Center*, 2020).
There is no clear rule on the extent of interpretability and explainability needed for AI systems to be classified as explainable. This extent is defined by the application and the use case. The research area of our project is medical imaging explainability, combining different elements of the explainability use cases mentioned above. Its purpose is to offer insights to a wider group of stakeholders.

1.2 Image explainability and COVID-19 detection

COVID-19 detection

Deep learning algorithms have been extensively used in medical imaging and showcased great performance for specific tasks for example, the detection of COVID-19 (Dong et al., 2021) but also as computer-aided diagnostic tools for humans (Beede et al., 2020),(Raumviboonsuk et al., 2019). These satisfactory performances of computer vision models have caused developments in this research field (S. M. Lundberg et al., 2018). Multimodal neuroimage data has been used for Altzheimer's detection methods, a general state-of-the-art review of some of these methods is available in (Jo et al., 2019). An active research area is present around chest scans. A big set of algorithms tackle lung cancer problems using datasets of computer tomography imaging. Methods of detecting and inspecting abnormal lung tissue growth are highlighted in (Hua et al., 2015). In (Nayak et al., 2021) a review of methods for detecting lung tissue lesions characteristic of the COVID-19 virus are shown. The detection is done using a varied set of data types but most include imaging data. All these models show great accuracy and performance, however, they lack the explainability component hence making their decisions difficult to explain and/or visualize. This can be a considerable drawback in practically applying these models to the real world or including them in high stakes real-world decision making.

SHAP (Shapley Additive exPlanations)

This is an explainability technique established on concepts in cooperative game theory (Shapley, 1951). The technique calculates importance values that are assigned to a feature of the data. These values are calculated by averaging the marginal contributions of features considering all possible combinations of contributions. SHAP unifies multiple different explainability techniques (Lundberg & Lee, 2017). Another advantage of SHAP is that it builds on and improves on some flaws of LIME (Local Interpretable Model-agnostic Explanations) (Ribeiro et al., 2016). The biggest differentiating impact is that SHAP, unlike LIME, has a game-theoretical assurance that guarantees local accuracy and consistency

Gradient activation based explainability methods

One of the first ideas of explaining the 'black box' nature of neural networks was proposed with Saliency maps using deconvolutions (Zeiler & Fergus, 2013). They calculate the neuron's absolute value of the partial derivative of the output. However, this technique is unable to distinguish the differences between classes. Class activation maps (CAMs) are another group of commonly used techniques. Proposed in (Zhou et al., 2016), the algorithm calculates the gradient of the output with respect to some other network parameter. These other parameters are the input parameters in many cases. A problem with the classic CAM implementation is the need for a model architecture with a global average pooling layer after the last convolutional layer. Should this not be the case, re-training is required. There are many papers that discuss improvements of CAM. One is (Selvaraju et al., 2020) with Grad-CAM and Guided Grad-CAM (introduced in 2016, but kept updated). Others like (Sayres et al., 2019) with Grad-CAM++ and (Sattarzadeh et al., 2021) propose some minor improvements for multi-class distinction problems. Further improvements are in (Wang et al., 2020) with Score CAM. Even though many versions exist, they are all based on calculating gradients like in GradCAM. GradCAM calculates the gradient with respect to the feature maps of the last convolutional layer. The Grad CAM algorithm generates a 'heatmap' that when resized visualizes parts of the image that are "activated" when the model makes a decision of classifying an image to a given class. This technique is model agnostic for all differentiable neural networks. It can also be applied to some reinforcement learning models and to also include captioning as a form of textual explanation. No re-training of models is required to implement this explainability technique. The code in the project follows the fundamentals of Grad-CAM but is extended to 3D. It is based on a Grad-CAM implementation by F.I.Tushar with changes in the backend models, pre-processing, functionality, visualizations, and gradient mapping. Other explainability techniques for imaging tasks are reviewed in-depth in (Singh et al., 2020).

1.3 Novelty and hypothesis in this project

The work in this project is based on creating explainability pipelines for COVID detection for 3D computer tomography scans. We noticed that gradient-based explainability techniques were not as present with models that use 3D data, especially with Keras/Tensorflow libraries. Having this in mind, combined with the recent rise of explainability as a subfield of machine learning (Wu et al., 2021), an opportunity to test novel ideas in this project had risen. The urgency of COVID-19 and the impactful role of imaging techniques are mentioned in (Dong et al., 2021). When creating this pipeline, certain objectives were taken into account. The explainability pipeline should be model agnostic, require no re-training or adjustments of already existing models (anyone with a trained model for that task can use the system as a "plug and play" solution) and offer explanations for multiple stakeholders in the explainability pipeline. That is why, besides the 3D explainability using gradient-based methods, 2D explainability using perturbation techniques were implemented. This offers a more "intuitive" understanding but also overcomes some shortcomings of the 3D explainability techniques. Most of the Grad CAM implementations are on 2D images with models performing object localization. Pytorch implementations of 3D Grad CAM methods such as M3D CAM (Gotkowski et al., 2020) is a full library for 3D explainability. Other implementations tied to specific use cases are available like using temporal convolutions (Ras et al., 2020), for Alzheimer's disease (Yang et al., 2018) and disease detection in plants (Nagasubramanian et al., 2019). At the moment of writing this thesis, there were no working implementations for Grad CAM on 3D images in Tensorflow and Keras. This is one of the motivations for novel impact through this project, besides the impact of explainability in general. These gradient-based methods are developed and kept up to date and their usage is still explored, with many being reviewed again after publishing.

Entire explainability systems, not just methods, are being built such as in (Spinner et al., 2019). Furthermore, workflows that offer visually-assisted approaches are presented in (Sacha et al., 2019). The importance of explainability in recent years is further proved by their inclusion of some type of explainability in Covid-19 deep learning imaging methods. An apparent example for this are (Wu et al., 2021) and (Nayak et al., 2021).

Hypothesis and aims

Hypothesis: A pipeline with 2D and 3D explainability techniques would offer visualized insights to various stakeholders in the AI model building process.

Aim 1: Create model agnostic explainability techniques that do not require re-training and model architecture changes.

Aim 2: Implement 3D explainability techniques by expanding 2D explainability concepts using the Tensorflow and Keras libraries.

Aim 3: Using an explainability pipeline better understand model decisions and potential biases.

Aim 4: Use 2D explainability to overcome some shortcomings of 3D explainability.

Data exploration

The dataset used in this project is the MosMed dataset with Chest CT Scans with COVID-19 Related Findings (Morozov et al., 2020). Although only a subset of this dataset was used, it is truly a useful resource for COVID-19 related projects. The dataset contains the resulting scans of computer tomography of chest organs. It contains data with radiological indications of viral pneumonia induced by COVID-19 and data without any indications of COVID-19.

The scans consist of 3-dimensional pixels which are called voxels. The voxel is the smallest discrete element of the image. The spatial resolution is also an important concept as it affects the ability of viewers to distinguish between structures in the scans that are in close proximity. The higher the resolution is, the greater this ability is (Huda & Slone, 2003). The CT image is mathematically reconstructed from numerous views obtained when the X-ray tube is rotated 360 degrees around the patient. The voxel values are expressed in Hounsfield units (HU). They signify the rate at which tissue absorbs or attenuates radiation. The reference point of 0 is water. These values are usually converted to grayscale images and are used in radiological imaging. Value of -1024 HU is black and represents air in the lungs. Fat tissue is close to -100 HU, whereas muscle is around 100 HU. Different types of bone go from 200 HU (trabecular and spongious bone) to about 2000 HU (cortical bone). Metal implants are usually capped at the maximum value because they have very high Housfield unit values. These concepts were important to understand in order to explore the data. Some of the artefacts detections and visualizations were done having this knowledge in mind. Besides the Python packages Matplotlib and OpenCV, the software package ITK Snap was used for data exploration, voxel histogram manipulation, and overall data understanding. A screenshot of ITK Snap is shown in figure 1.3.3.

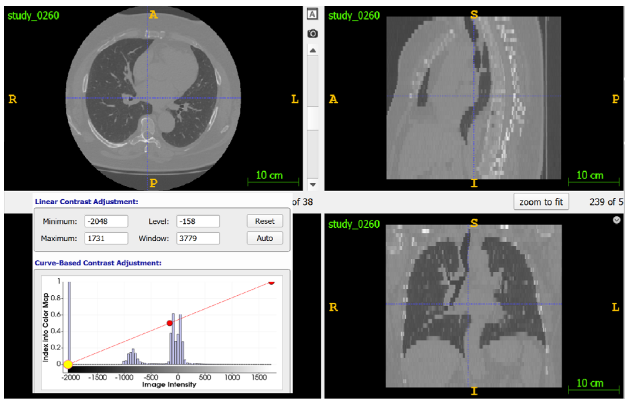

Linear Contrast Adjustment:
Minimum: -2048 Level: -158
Maximum: 1731 Window: 3779

Figure 1.3.3 - ITK Snap screenshot of the study_0260 from the MosMed dataset. The image shows slices of the 3D CT scans from axial (top left), coronal (bottom right), and sagittal (top right) views. The image on the bottom left shows the windowing feature which is commonly used in CT scan explorations as it enables noticing more nuanced details. Windowing is performed when the intensity of the voxels is confined to a range of values.

2. Algorithms and methods

In this section, the methods for creating the explainability pipeline are explained. This section has three parts. The first part elaborates on the data pre-processing and the model backends. Although the explainability techniques can be used with any differentiable neural network architecture, it is important to understand the details of the models which are explained. The second reviews the explainability algorithms in detail. The third is on the pipeline of explainable techniques and the general workflow of this project. The Github repository with a link to all the related files, code, and outputs is available at: https://github.com/fm1320/ICL/tree/main/AstraZeneca (more in Appendix A)

2.1 Model backends and pre-processings

This section gives information on the deep learning models used for testing out the explainability methods. Details on the training process and pre-processings are given to better understand certain biases or anomalies that might occur in the explainability pipelines.

Pneumonia detection model (used for 2D explainability)

Data Quality

The subset of the dataset "Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification" by (Kermany et al., 2018) is used and the model is based on a Kaggle competition notebook by Aakash Kumar Nain from 2018 but with different training seeds. The dataset consists of anterior and posterior chest X-ray images selected from cohorts of pediatric patients of one to five years old from Women and Children's Medical Center, Guangzhou. The scans were part of the routine patient's checkup and unreadable and low-quality scans were removed. The annotation was done by two separate physician experts and a third one made a grading to account for possible errors. The subset data chosen consists of 5863 images in total, with 3875 NORMAL cases and 1341 PNEUMONIA cases. The data is highly imbalanced. The situation of having scans from children and young women might incur bias to the model's decisions when the model is tested on scans of fully adult patients.

Data pre-processing and transformations

The images are converted to RGB space. Augmentation is implemented in the training data for the under-represented class (the NORMAL class) to further balance the data and generate more examples of the NORMAL class even though this class is still under-represented. The *imgaug* library was used to implement 3 types of augmentation: random brightness change, rotation, and horizontal flipping. Transfer learning on the first 4 layers of the VGG16 model was used with weights from Imagnet. The other layers are left to be fine-tuned to the task.

<u>Covid-19 3D CNN detection model (used for 3D explainability)</u>

Data Quality

A smaller subset of the MosMedData, Chest CT Scans with COVID-19 Related Findings (Morozov et al., 2020) is used. This dataset has lung CT scans with COVID-19 findings, but also CT scans without COVID-19. The CT scan findings were used as labels in creating a classifier that tries to predict the presence of viral pneumonia. As a result, the problem is a binary classification one. The scans are divided into five categories CT0, CT1, CT2, CT3, and CT4. CT0 consists of non-covid samples, while CT1 to CT4 consist of Covid samples with increasing severity. In this project, the most severe group CT4 was not used as earlier detection makes more practical sense.

Data pre-processing and Augmentations

The files are in Nifti format(.nii) from the MosMed dataset (Morozov et al., 2020). The preprocessing functions use Keras's official library and 3D imaging code examples. The nibabel package is used to read the scans. For such files, Hounsfield units are used to store raw voxel intensity values from CT images (HU). In this dataset, the range is from -1024 to above 2000. Because we are only interested in structures inside the lung, 400 HU is utilized as a higher bound. The overall threshold for the HU values is between -1000 and 400. The CT scans are pre-processed according to the following steps:

- Scaling (normalization) of the HU values between 0 and 1.
- Resizing in width, height, and depth dimensions (128x128x64). Using the ndarray.zoom function in python, If the size of the input is smaller than (128x128x64) in any dimension, Spline interpolation of the first order is performed to have the scan in the preferred size. This approach also preserves the distances between voxels.

Lastly, the dataset is split into train and validation subsets. For example, only 200 scans are used for training, not the full dataset. An augmentation function that randomly rotates volumes at different angles is used while defining the training and validation data loader. The scans are rotated by a random angle between -20 and 20 degrees. Both the training and validation data are already rescaled and normalized. This yields similar but different results when replicating the model as the set-up parameters are used at random. A random seed is not specified before training too, which influences the exact replicability of the model's results.

2.2 Model explainability

There are two categories of explainability techniques used in this project. The first category is gradient-based explainability techniques (Guided class activation maps) and the second is about perturbation based techniques like Shapley additive explanations (SHAP) based on Local Interpretable Model-agnostic Explanations (LIME).

Class activation maps - gradient-based explainability methods

In order to understand the adapted version of Grad CAM (Selvaraju et al., 2020) implemented in this project, a description of the general class activation mapping (CAM) algorithm will be given followed by the Grad CAM algorithm and then the steps implemented in this project. The theory behind the classic class activation mapping is the following as expressed by equation 1. Let:

$f_k(x, y)$- *k-th feature map with height x, width y of the last convolutional layer, and*

$F_k = \sum_{x,y} f_k(x, y)$- *the k-th averaged pooled value. The score for a given class (S_c) is computed as:*

$$S_c = \sum_k w_k F_k = \sum_k w_k \sum_{x,y} f_k(x, y) = \sum_{x,y} \sum_k w_k f_k(x, y)$$

Equation 1, Source: Weizmann Institute of Science 2021, 4182 Deep Learning for Computer Vision: Fundamentals and Applications Spring 2021

Figure 2.2.1 shows a visual illustration of this concept involving the Global average pooling of the neural network. This is based on the concept that each feature map captures some important feature from the image (Bengio et al., 2013). The process of generating a class activation map is explained visually in Figure 2.2.1.

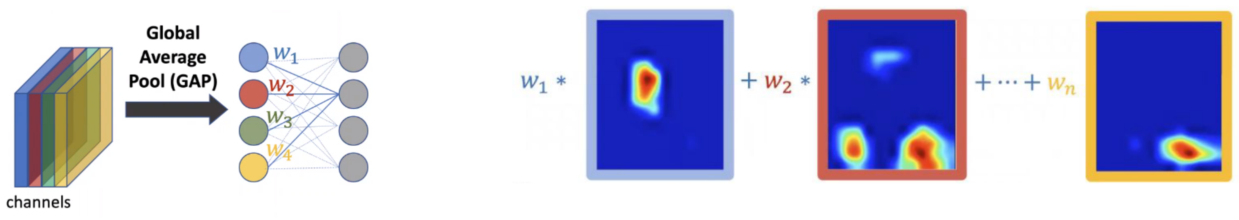


Figure 2.2.1 Each feature map is reduced to a weight (w1,w2 etc.) by a global average pooling layer. To calculate the heatmap, a linear combination between the feature maps and weights is computed. When feature maps are multiplied with weights, the "contribution" of each weight is returned. The score for class C is computed as a linear combination of the feature maps with the learned weights.
(Zhou et al., 2016). Source: Weizmann Institute of Science 2021, 4182 Deep Learning for Computer Vision: Fundamentals and Applications Spring 2021

Grad CAM is a generalized version of the already mentioned CAM in Figure 2.2.1 and the previous passage. The difference now is that, instead of summing a linear combination between the feature maps and the learned weights, they propose a new term. This term is α_k which takes into account the global average pooling of the gradients with respect to the feature maps. S_c is the target class score. This term α_k signifies neuron "importance". The expression for this term is given as follows:

$$\alpha_k = \frac{1}{Z}\sum_x\sum_y \frac{\partial S_c}{\partial f_k(x,y)},$$ *where* $\partial f_k(x, y)$ *are the gradients computed via backpropagation*

global average pooling

Equation 2, Source: Weizmann Institute of Science 2021, 4182 Deep Learning for Computer Vision: Fundamentals and Applications Spring 2021

In the Grad CAM paper (Selvaraju et al., 2020) it is proven that CAM is a special case of grad CAM where $\alpha_k = w_k$ when the last layer is a global average pooling layer. Also, it is stated that adding a ReLU activation to the final expression adds better performance (Selvaraju et al., 2020). In Equation 3 a comparison of the heatmaps calculated by CAM and Grad CAM is shown in equation 3.

$$L^c_{CAM} = \sum_k w_k f_k \qquad L^c_{Grad\,CAM} = ReLU\left(\sum_k \alpha_k f_k\right)$$

Equation 3, Source: Weizmann Institute of Science 2021, 4182 Deep Learning for Computer Vision: Fundamentals and Applications Spring 2021

Having in mind the theoretical bases of the mentioned gradient-based algorithms, here are the steps implemented in the project (adapted from *F.I.Tushar's acknowledged implementation)*:

1. A model is created that maps the input CT scan to the activations of the last 3D convolutional layer as well as the output predictions of the model. The aim is to "record" the gradients when an input scan is forward passed through an already trained model. A function is written to select the last convolutional layer. The last layer is chosen as the most specific features of the class are captured by that layer (this is also dependent on the backend model of choice).
2. Tensorflow's gradient tape function is used to calculate the gradient of the top predicted class with respect to the feature map of the last convolutional layer.
3. A guided ReLu is performed. This step offers suppression of the negative gradient activations giving a "smoother" visual map (Springenberg et al., 2015).

4. A weighted map of filters (according to the gradient's importance to the top predicted class) is multiplied by each channel in the feature map. Then, all channels are summed to obtain the activations of the heatmap.

5. Heatmap is normalized to have voxel intensity values between 0 and 1

This was implemented in Python using Keras and Tensorflow frameworks. The details of the versions and requirement packages are available at the GitHub repository of this project available in the Appendix A section.

Perturbation based methods - SHAP and LIME

The SHAP explainability was implemented taking inspiration from the blog post by Tiba Razmi and the official repository of the SHAP project but with different preprocessing, backend models, classification task, and test data. SHAP is based on LIME, and although LIME was also tested (just as a demonstration following the python library) SHAP was chosen because of its game-theoretical background (Section 3 in S. Lundberg & Lee, 2017) and its better alignment with human intuition (Section 5 in (S. Lundberg & Lee, 2017). A "black box" model example of the intuition behind SHAP is shown in figure 2.2.2.

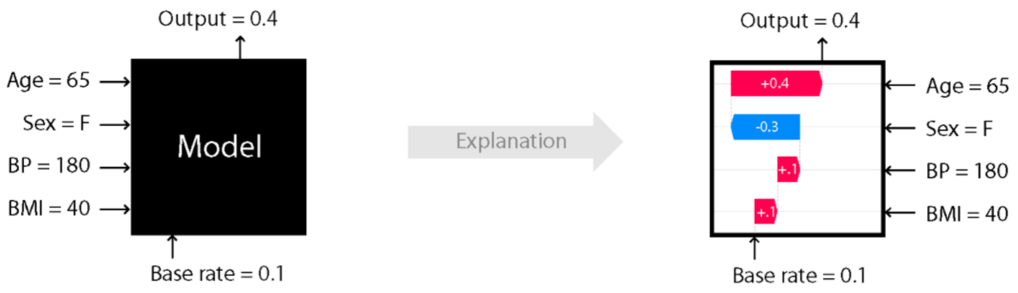


Figure 2.2.2 - A visualization of how SHAP works for a generic black-box model. The left-hand side shows Age, Sex, BP, BMI as input features and the output is an arbitrary prediction (0.4). The right-hand side shows the SHAP values colour-coded in terms of how much a feature contributes negatively (blue) or positively (red) for the given prediction. The SHAP values can be interpreted as the impact of having a certain value for a given feature in comparison to the same prediction if that feature had a baseline value.

In the case of the project, this intuition still applies, however, the parameters such as Age, Sex, BMI etc. from figure 2.2.2 are image patches, and the explanation offers a map of the input image divided into patches with the contribution of each patch being visualized. The model is trained on 2D pneumonia scans and the inference is made on the 2D slices of the 3D scans.

The implementation steps are the following:

1. Extract a 2D slice of interest from the 3D CT scan. There is an option to do this manually or it can be done automatically by a certain rule (voxel values, lung masks, gradient activations etc.)

2. Segment the image into superpixels in python using the simple linear iterative clustering (SLIC) algorithm (Achanta et al., 2012) which adapts a k-means clustering approach to generate superpixels (other image segmentation algorithms can be used too)

3. Explain each superpixel using the Kernel SHAP python library

4. Get the predictions of the 2D chest classification model for the given image and extract the top 2 predictions (in the case of the project the two possible predictions are pneumonia and non-pneumonia but the model can be extended to multiple class classification problems)

5. Generate SHAP values and visual explanations signifying which superpixel contributes by how much towards the given class.

2.3 Project workflow

In diagram 2.3.1 the explainability pipeline (which includes the already mentioned explainability components) is drawn. There are two types of explanations as outputs. One is a 3D explainability method and the other is a 2D explainability method.

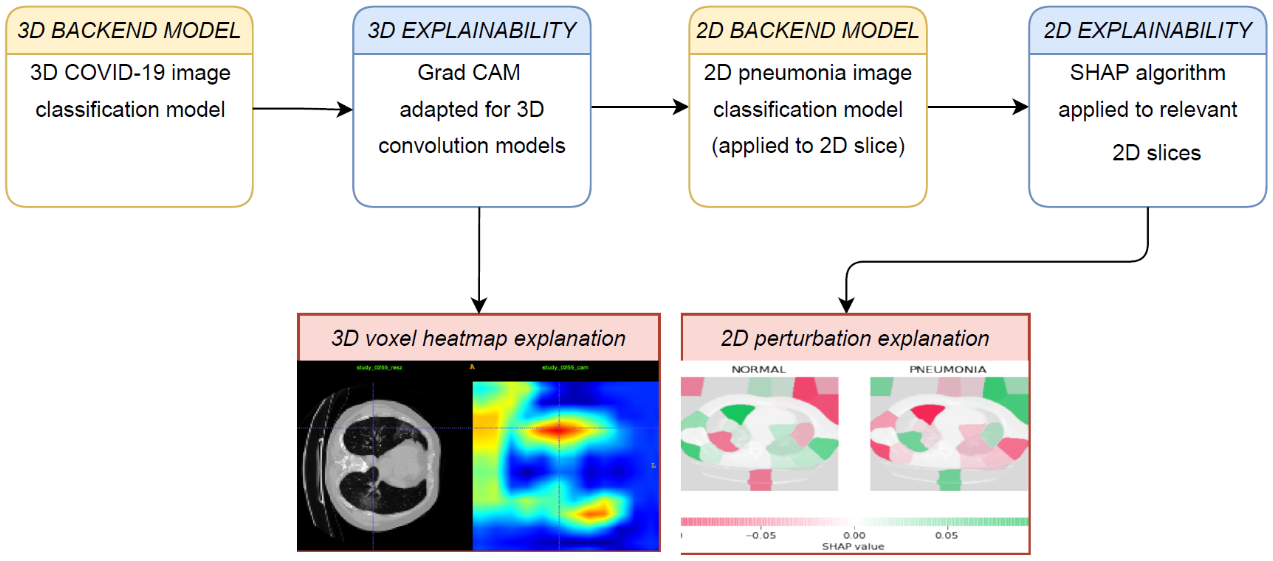

3D BACKEND MODEL
3D COVID-19 image classification model
3D EXPLAINABILITY
Grad CAM adapted for 3D convolution models
2D BACKEND MODEL
2D pneumonia image classification model (applied to 2D slice)
2D EXPLAINABILITY
SHAP algorithm applied to relevant 2D slices
3D voxel heatmap explanation
2D perturbation explanation
NORMAL
PNEUMONIA
−0.05
0.00
0.05
SHAP value

Diagram 2.3.1 - The diagram shows the processing of the 3D CT scans. Firstly the 3D backend model generates a prediction which is explained by the adapted 3D Grad CAM based algorithm. This generates the first explanation which is a 3D heatmap of class activations. Afterwards, relevant 2D slices from the coronal viewpoint are extracted from the 3D CT scans (This is done either manually or by selecting regions with high activations). Then a 2D pneumonia detection backend model is used to generate a prediction for the 2D slice. After the SHAP algorithm is applied and the perturbation maps for 2D are outputted. The green regions highlight the regions which contribute positively to the given class, whereas the red regions the negative contributions for the given class. This pipeline generates explanations that can be used both for debugging purposes but also for visualization purposes and understanding of the model.

3. Results

This section shows the results from the explainability methods as well as some biases which could be noticed. Also, the 3D heatmaps are evaluated compared to annotated masks by experts. In the end, the evaluations and results of the backend models are presented. The outputs discussed in this section, and other more examples, are uploaded on the GitHub repository of this project.

3D explainability results

The generated output is in the form of 3D heatmaps with normalized voxel intensities saved as nifti files. An example is shown in figure 3.1. The scan number is shown on the top left in each image. In the jet colour map, the “warmer” (red, orange) the colour, the higher the gradient. Higher gradients signify regions of interest with a greater contribution towards the decision of the model. Studies starting with 02XX have moderate COVID-19 pneumonia cases, those with 00XX have no COVID-19 present, and 11XX are severe disease cases. A lot more of these examples and visualizations are available on the GitHub repository of this project.

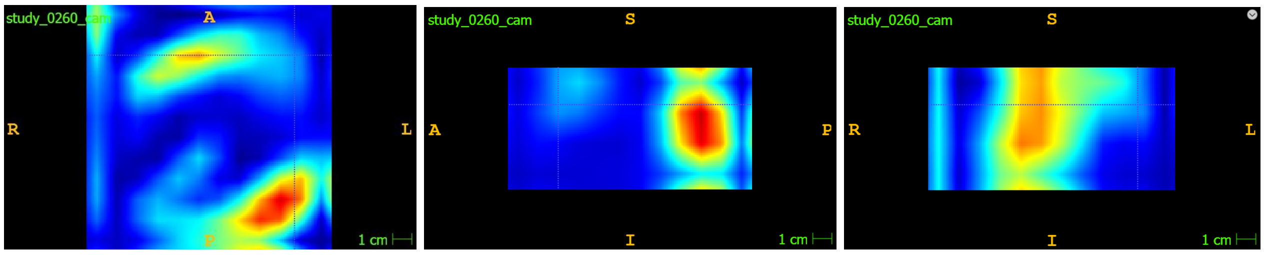


Figure 3.1 - A visualization of a slice of a 3D heatmap for study_0260 processed to 128x128x64 in ITK snap with the Jet colour map. From left to right (axial, coronal, sagittal view). The axial heatmap is rotated by 90 degrees in this and the other jet map visualizations. In this scan, it is apparent that there is a region of interest and activation in the patient’s left lung. There are some activations in the right lung, but these are not that evident in these slices.

The heatmaps are more intuitive for humans when they are combined with the original scans and not just the heatmaps themselves. Figure 3.2 visualizes jet colour heatmaps with their ct scan slice accordingly.

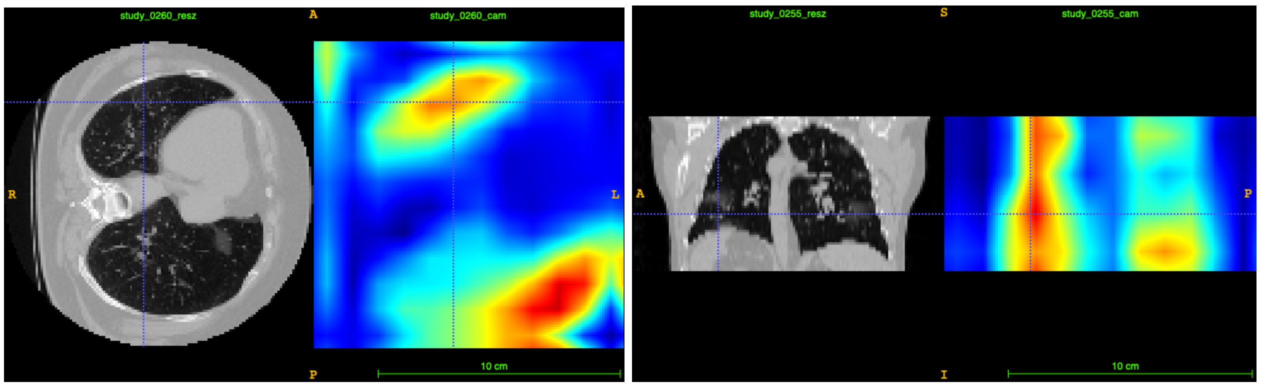


Figure 3.2 - 3D heatmap slices with their 2D counterparts. Here there is an activation around the right lung of the patient, but also small activations are present in the lower part of the left lung.

The scans can be also visualized with a different colour map. Viridis was also chosen in this project because of its better representation of the colour differences of slight gradient changes. A key thing to note is that all heatmaps when presented in 2D are slices of a 3-dimensional heatmap. This seems less intuitive to the eye compared to directly generated 2-dimensional heatmaps which are used in most of the implementations of gradient-based algorithms. These heatmaps should be looked at not as exact localizations of COVID-19 lesions but as “clouds” which go “above” and “below” the presented slices. One example to illustrate this is in Figure 3.3 where the slices moving from the abdominal towards the thoracic area are visualized.

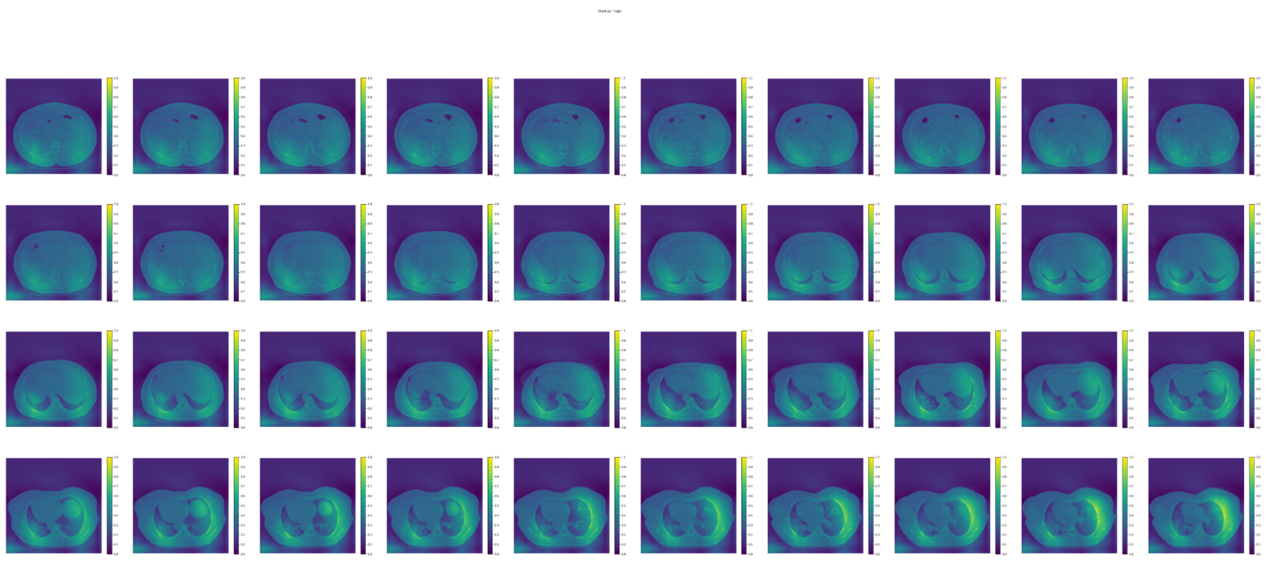


Figure 3.3 - A set of 40 slices of study 1109. Moving from the abdomen (upper left corner) towards the mid-region of the chest (bottom right corner) the activations around the left lung gradually increase. This illustrates the “cloud” concept of the 3D heatmaps.

This can create a situation where activations can be viewed on one slice but not due to some characteristic of the image at that given slice but as an effect from a feature above or below the visualized slice. That is why 3D heatmaps might be less intuitive for humans. Figure 3.4 shows 3D heatmaps of a severe COVID-19 slice by mapping the activations opposite of the dominant class. This is to show that Grad CAM can visualize counterfactual examples meaning any class can be visualized if the backend model was for example classifying multiple classes. The higher the visual difference between classes the more noticeable these maps are.

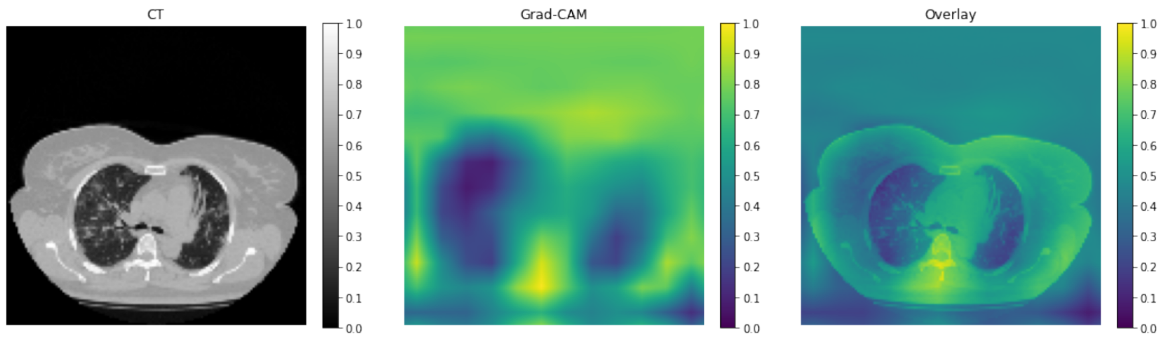


Figure 3.4 - This is study 1109, from the CT4 subset with severe ground glass opacifications and consolidations. From the middle subplot, it can be seen that the activation for the NORMAL class is around the lungs, which signifies that the details in the lungs are what contribute to the class ABNORMAL which is present in this image.

Figure 3.5 shows a COVID-19 scan from the subset of NORMAL scans CT0. Here the activation for the NORMAL class is mostly centred near the spine of the scan.

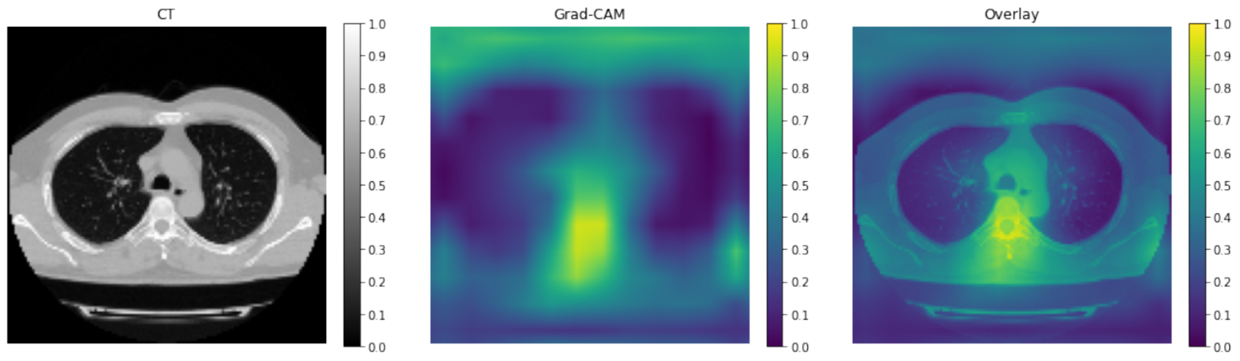


Figure 3.5 - A NORMAL CT scan without the presence of COVID-19. It is noticeable how the activations are around the spine of the scan and how confined around these regions they are compared to the evidently ABNORMAL scan of Figure 3.4.

The 3D explainability offers some insights into biased decisions by the model. This is one of the advantages of having an explainability method as part of a machine learning deployment pipeline. Figure 3.6 highlights some recurring biases that were noticed throughout some portions of the data.

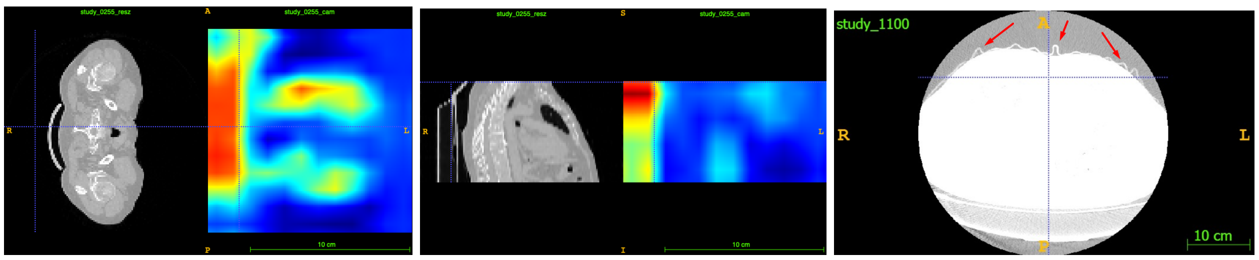


Figure 3.6 - The figure shows examples of biases that arise in some of the data. The leftmost heatmaps show high activation in the part of the image near the table of the CT scan. The same view but from a different viewpoint is shown in the middle picture. This points to the fact that the model can concentrate on certain artefacts in the image caused by the sensors or apparatus of the CT scan machines. Another example is the rightmost picture which after some thresholding, shows a rippling artefact on top of the patient. This is usually a cover or blanket used for patients when the CT scans are taken. Such artefacts in the data can cause biases and can skew the backend model.

Evaluation of heatmaps

Evaluation of these heatmaps is a difficult task as there is an element of perceived subjective value. One way of evaluation is using manually annotated masks. The MosMed dataset contains a small number of scans that have been manually annotated with binary pixel masks depicting regions of ground-glass opacity and consolidation as regions of interest. We generated overlays of the expert's annotations and our heatmaps to understand the overlap and how our heatmaps correspond to the expert annotators. Figure 3.7. (a) shows this. Tables 3.7. (b) and 3.7. (c) show the similarity between our activation mask and the mask annotated by experts. They are quantified using the Dice coefficient (F1-score) and MSE (Mean square error). Different thresholds of binarization were used (Lower MSE and higher Dice coefficient mean greater similarity).

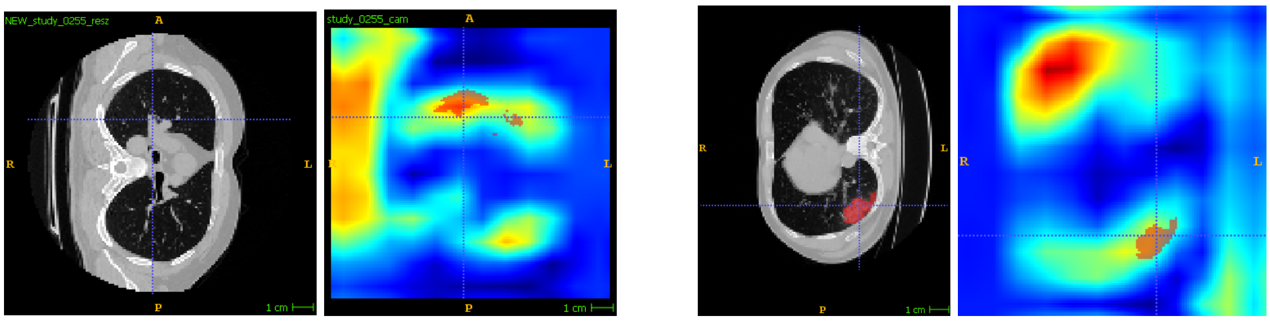


Figure 3.7. (a) - Manually annotated regions of interest that contain COVID-19 signatures study 0255 (left) and study 0258 (right). Each CT scan has the appropriate overlaid heatmap with the blue crosshair placed at the same point. The coarse red pixels signify the expert's annotations and the jet is our output. From the left-most pair, it can be seen that the model correctly overlaps with the expert's annotations and is more responsive to consolidations. The ground glass opacities are usually more dispersed and are not big connected regions. Our heatmap activates on consolidations. In the right pair, it can be seen that the heatmaps activate on the consolidation in the scan, however with less intensity due to the artefact bias which generates high-intensity activations. These insights open the way for non-max suppression and heatmap intensity value casting for further improving and debugging the model.

Study ID:	study_0255	study_0256	study_0258	study_0259	study_0260	study_0265
Threshold of binarization is: >0.7 (Red/ Dark orange)	MSE: 0.00277 DICE: 0.95435	MSE: 0.00321 DICE: 0.95294	MSE: 0.0002 DICE: 0.995	MSE: 0.00047 DICE: 0.97742	MSE: 0.00136 DICE: 0.95453	MSE: 0.00226 DICE: 0.97168

Table 3.7. (b) - Both masks were binarized and resized to the same format. The threshold used for the intensity was above 0.7 for our heatmaps. This way only the highest activations are preserved. Then the two masks are added and the region of overlap is compared to the experts annotated binary mask. PSNR, MSE, and SSIM are calculated. These calculations give a quantified result of the similarities between our maps and the experts. However, this is not a fully accurate technique as there can be discrepancies with sizing, dimensions, and choice of backend model. This would be ideal for a segmentation task, but the role of the heatmaps is not to segment COVID-19 but to offer an understanding of model behaviour. Nevertheless, this approach offers consistent quantified metrics but needs to be coupled with visual feedback for a more thorough evaluation.

Study ID:	study_0255	study_0256	study_0258	study_0259	study_0260	study_0265
Threshold of binarization is: 0.3 - 0.7 (yellow/orange)	MSE: 0.00017 DICE: 0.99554	MSE: 0.00105 DICE: 0.97931	MSE: 0.00001 DICE: 0.99878	MSE: 0.00012 DICE: 0.99359	MSE: 0.00039 DICE: 0.98505	MSE: 0.00124 DICE: 0.98663

Table 3.7. (c) - Both masks were binarized and resized to the same format. The threshold used for the intensity was between 0.3 and 0.7 for our heatmaps. These results lead towards the assumption that the heatmaps are affected by high peaks (caused by biases or artefacts) and the real lung abnormalities are highlighted in the middle to higher ranges. This paves the way for further development of models with regions of interest, multiple classes for specific lung abnormalities, and max-suppression of images.

2D explainability results

In this subsection, the results from the 2D explainability are presented. Firstly, LIME is used just as a demonstration (medium example by Cristian Arteaga with different inputs), and afterwards, SHAP was implemented with a 2D pneumonia detection backend model (the inference is on the 2D slices, the 3D classification is not part of this section). Figure 3.8 and Figure 3.9 present demo versions of LIME and SHAP with arbitrary classes and backend models. Figures 3.10 and 3.11 depict 2D explainability with SHAP using an appropriate pneumonia detection backend model. The 2D slice used is an example of moderate severity, but any 2D slice can be used.

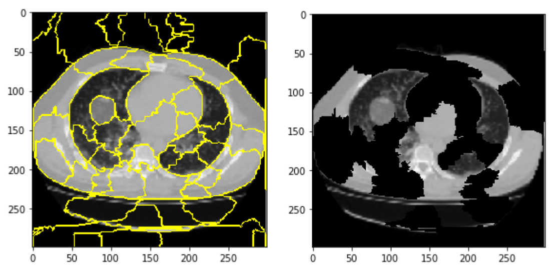


Figure 3.8 - A LIME prototype demo example. This example is tested on a model that recognizes objects, not a specific disease. In order to practically use this approach, a specific model should be trained that recognizes a relevant class (e.g pneumonia). This was implemented using a "random" backend model just to test out the explainability technique.

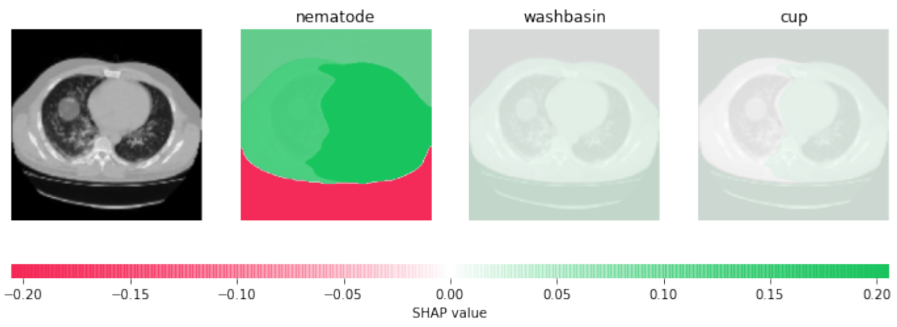


Figure 3.9 - A SHAP prototype demonstration example with ImageNet classification model. The top 3 imagnet classes are chosen (nematode, wash bin, cup) and a Superpixel segmentation algorithm with a low number of segments is chosen. This demonstration was modified in Figures 3.10 and 3.11 to include a backend model that recognizes pneumonia.

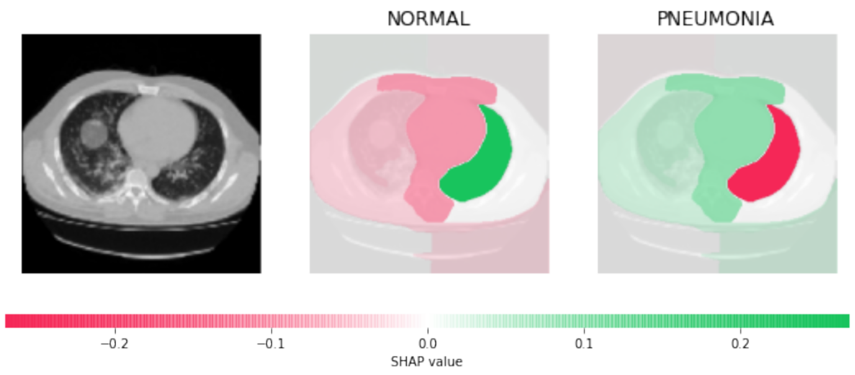


Figure 3.10 - A SHAP 2D explainability using a pneumonia detection backend model. This model recognizes two classes: NORMAL and PNEUMONIA, however a multi-class backend model that detects multiple chest conditions can be also used. The colour green is indicative of a positive influence of a given image patch on the class. Accordingly, red patches are indicative of a positive influence on the second class. Since the problem is binary, the generated images are complementary to each other.

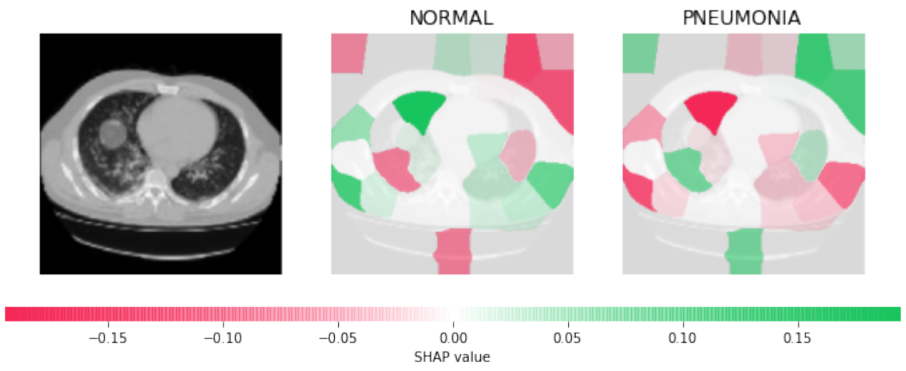


Figure 3.11 - A SHAP 2D explainability using a pneumonia detection backend model. The difference to figure 3.10 here is that there are more generated superpixels in the image segmentation which offers insights to smaller patches of the image. The number of superpixels can also be altered to include a different number of patches. The higher number of patches means that the insights can be more localized now, compared to 3.10. This technique can be used to segment regions of interest for further investigation by engineers or even other explainability techniques.

Backend models results

2D Backend - model architecture, training, and evaluation

The model takes an input of (224,224,3), follows the VGG16 model with the added batch normalization layers and 2D separable convolutions. The model is trained for a binary classification task. The dataset is divided into three sets: 1) train set 5223 images 2) validation set 16 images and 3) test set 624 images. The test set for external model evaluation is from the same dataset but an unseen section of the data. The remaining imbalance of the data (not solved by the augmentation) is corrected by adding class_weight={0:1.0, 1:0.4} for the classes when calculating the loss function. The model was trained for 20 epochs and the training and validation curve are shown in Figure 3.12.

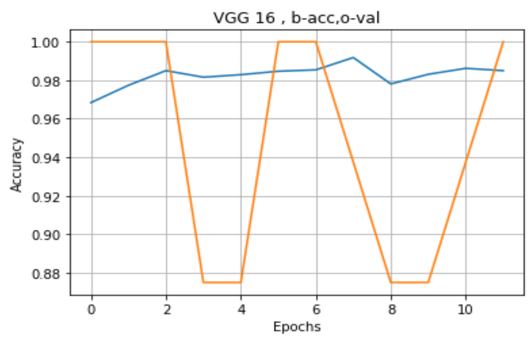
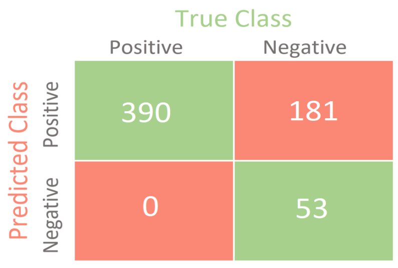


Fig 3.12 - The blue graph shows training accuracy and the orange validation accuracy. The sudden drastic differences in validation accuracy might be due to the small size of the validation dataset. Due to the small size, there are really easy and really difficult examples for the mod

The model achieves a 98% training set accuracy, however only 70% accuracy on the testing set. Using the confusion matrix generated, the model has a high recall in multiple training sessions with different random seeds (0.95 - 1.00) and a recall between (0.68 - 0.80). Precision and Recall follow a trade-off as they cannot both simultaneously increase. The acceptable trade-off should be dependent on the task at hand. In the case of explainability methods, this is not a key concept but is important to document the model behaviour. In figure 3.13 the confusion matrix of the model is given on the test set of 624 images.



Predicted Class	True Class: Positive	True Class: Negative
Positive	390	181
Negative	0	53

Figure 3.13 - It can be seen the level of false negatives is really low but this comes at a sacrifice of the overall accuracy and the high number of false positives

Some tests were done on higher input resolution images with more details preserved. Then the overall accuracy improves, however the number of false negatives increases. This is further proof of the Precision/Recall tradeoff mentioned before.

3D Backend - model architecture, training, and evaluation

The architecture is based on the 3D CNN architecture (Zunair et al., 2020). The evaluation of a 3D convolutional neural network (CNN) to detect viral pneumonia in CT scans is presented here. 3D CNNs are useful for learning representations for 3D data, as they take as input a 3D volume or a series of 2D slices. It is worth noting that the sample size is quite small (200), and random seed is not used. As a result, some variation can be expected in the outcomes. The complete dataset, which includes over 1000 CT scans, is available on the dataset link provided at the beginning of this paragraph. The authors of the Keras example attained 83% validation accuracy using the entire dataset. In both situations, there is a 6-7% variation in the performance of the classifier. Three versions of the model were saved, each with different accuracy on unseen data from the same dataset. The one used for the explainability pipelines has an accuracy of 75% on the validation set in the best training epoch (using early stopping). The accuracy curves of our training are shown in Figure 3.14

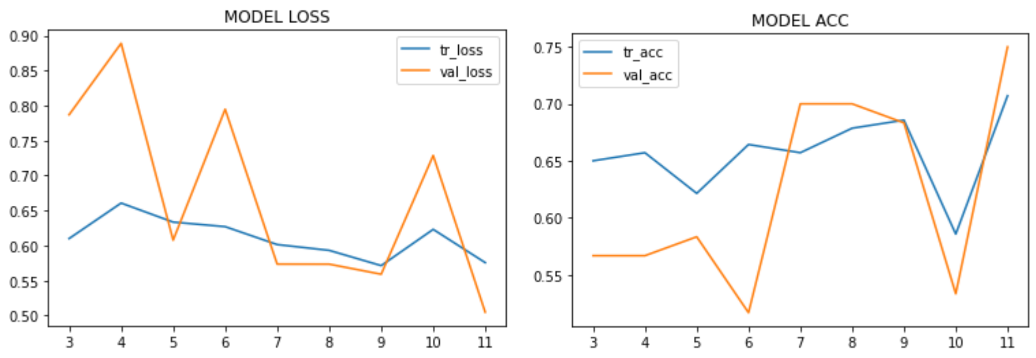


Fig 3.14 - (Left) Model loss in training (blue) and on validation data (orange). (Right) Model accuracy in training (blue) and on validation data (orange). The X-axis is the number of epochs.

An evaluation with a “conventional” confusion matrix for binary classification was not used here because of the varying difficulties of the subsets in the data. This means that choosing different severity subsets for the positive test class would give varying confusing matrix results. This is also noticeable with the validation test data. If it was predominantly from the subset CT3 (which has more advanced cases of COVID-19) it would be easier for the model to detect them compared to CT2. If the validation set was predominantly from CT1, the validation accuracy would have decreased. Table 3.15 shows an analysis of the model tested on each severity subgroup from the full MosMed dataset (not the full subset, but several scans from each subset were used). For CT4 there were only a few examples available and the model correctly predicted all of them.

Subset (severity)	CT0 (normal)	CT1(mild)	CT2 (moderate)	CT3(advanced)
No. of scans	100	100	125	45
Accuracy	86.0%	32.0%	50.4%	82.9%

Table 3.15 - Backed model evaluation on subsets with varying severities

4. Discussion

The visualizations from our results show that certain biases and characteristics of both the models and the data can be identified with the help of the explainability pipeline. Adding to this, the analysis of the backend models and their performance helps in shaping up the understanding of the decision making of these “black box” models. This section will discuss the drawbacks of the methods and experimental design with possible reasons, solutions, and mitigations. Furthermore, the argument about the impact and importance of this work concerning the current trend in medical machine learning imaging will be stated.

Strengths and weaknesses of the methods

When using explainability methods the explanations must be related to the parameters the model has learned during training. In (Adebayo et al., 2020) and (Nie et al., 2020) gradient-based models are tested by doing model parameters and data randomization. The authors of (Adebayo et al., 2020) have tested various gradient-based explainability techniques and Grad CAM, the method used in this project, passed their randomization tests. The quick breakdown of their experiment is shown in figure 4.1.

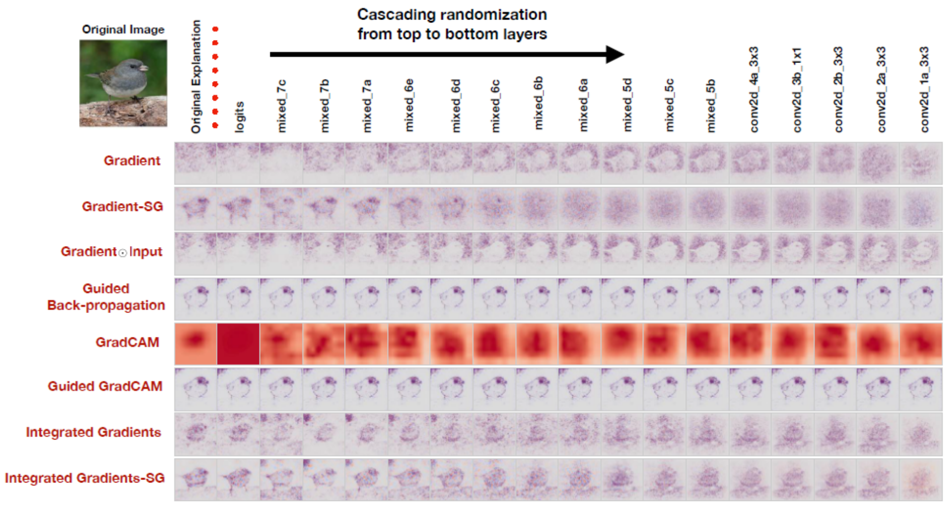


Figure 4.1 - Each row corresponds to a different gradient explainability mapping method, and the randomization gets bigger from top to bottom layers. It is expected that the further the randomization, the more the explanation will break. If that is not the case, the authors infer that those models are not related to the trainable parameters of the model and act effectively as edge detectors.
Source: Figure 2 from (Adebayo et al., 2020)

These findings suggest that although Grad CAM might not be the most precise (it is not a COVID-19 segmentation tool) and visually convenient method, Grad CAM can offer explanations that hold more value, compared to other similar more visually "appealing" techniques.

One drawback of Grad CAM methods is their bad differentiation of multiple occurrences of the same class (Chattopadhyay et al., 2018) in an image while saliency methods (Simonyan et al., 2014) are entirely class non-discriminative. Class differentiation is improved in Guided Grad CAM (Selvaraju et al., 2020). However, Guided Grad CAM has its faults as mentioned in figure 4.1. Another drawback (besides class differentiation) that inspired Score CAM (Wang et al., 2020) to include perturbation based explainability techniques is the false confidence of the Grad CAM models. They claim that activation maps with higher weights do not lead to more decision-relevant regions. Regions highlighted with smaller weight may contribute more to the class confidence score. This is explained in Figure 4.2 using a passage from their paper.

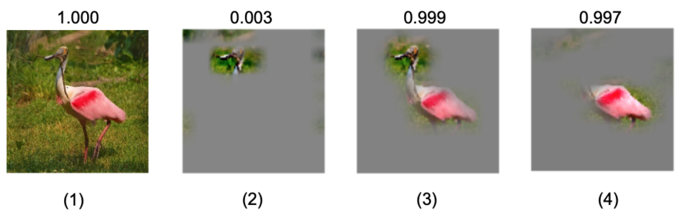


Figure 4.2 -(1) is the input image, (2)-(4) are generated by masking input with upsampled activation maps. The weights for activation maps (2)-(4) are 0.035, 0.027, 0.021 respectively. The values above are the increased on-target score given (1)-(4) as input. As shown in this example, (2) has the highest weight but causes less increase on-target score. Source: (Wang et al., 2020)

Some possible improvements on the class contribution issue can be seen in (Sattarzadeh et al., 2021). In this project, instead of implementing perturbation based explainability as part of the gradient-based method like mentioned, the perturbations using SHAP and LIME are implemented on 2D slices. This tackles the drawback of Grad CAM but also offers a different perspective and versatility of the explainability pipeline. It is also noteworthy to mention the influence of the backend models. A more accurate backend model and a different architecture would lead to different 3D heatmaps. Although the model works with any convolutional network, the visual results may vary. Many of the implementations of Grad CAM (mentioned in the introduction section) do 2D explainability using multi-label classification models such as Xception for example (Chollet, 2017). Although this does not prove, it possibly implies that Grad CAM based methods are suited better for object localization with multiple possible classes in 2D (like ImageNet or MNIST for example) unlike the binary classification tasks in 3D in this project. Additionally, the visual intuition in 3D is different to 2D heatmaps as explained in the Results section.

Knowing all of this about the 3D explainability, it should not be expected of 3D Grad CAM to exhibit equal results to 2D explainability, and even with an imperfect model, there is a lot to learn.

SHAP is a 2D explainability technique that has its downsides too. Although it offers a theoretical basis for its approach, it has some mathematical and human-centric issues as explained in (Kumar et al., 2020). In (Fernández-Loría et al., 2020) besides the other drawback mentioned, there is an analysis of counterfactual examples, where an important feature for the model might have low SHAP scores. The examples from their work are in the area of law and finance, but the same principles apply in medical applications. Another problem of LIME is that their solutions are not unique, meaning that re-running the model multiple times might yield similar but non-unique solutions. A key issue with perturbation techniques like SHAP is the choice of image patches. It is impossible to choose the “best” number of image patches as this choice is also subjective. The explainability being dependent on the backend model is also an issue here. For these reasons, SHAP and LIME are not advised to be used on their own. This is why we chose to combine these techniques with other explainability and pave the way towards human-centred and versatile AI systems. Explainability in general has an issue of evaluation and validation because they naturally involve a subjective human-centred design. Many of the mentioned papers offer survey-based approaches for evaluation usually combined with some quantitative metrics (similar to Dice coefficient in segmentation tasks). Another question to pose is “How much explaining is enough to say that something is an explainable AI system?" This is determined by the extent of the applied scenario. As long as something can be learned from the explanation - the explainability is useful.

The explainability pipeline can work with other backend models too as long as they are based on a convolutional neural network. Other implementations of these algorithms are available, which with some adjustments can offer explainability for other machine learning tasks beyond convolutional neural networks. The SHAP explainability paper (S. Lundberg & Lee, 2017) highlights some of these possibilities while others are well documented in the official SHAP python package documentation.

Impact concerning other literature and novelty

Explainability components are becoming instrumental in building AI systems. Full explainability AI systems and assistive workflows are being designed (Spinner et al., 2019), (Sacha et al., 2019) while current deep learning methods for COVID-19 have explainability components (Wu et al., 2021) and (Nayak et al., 2021). The repercussions of AI systems have been mostly studied in law, but recently similar studies (Banerjee et al., 2021) have uncovered that some COVID-19 detection methods can detect the patient's race just by looking at chest scans. Although this study has not been peer-reviewed, model decision explanations would still be useful in such scenarios. The novelty of this project comes not only from the 3D explainability with TensorFlow and Keras but also from the use case of explainability. A survey of COVID-19 imaging detection methods was released by (Roberts et al., 2021). They have identified that out of 2212 COVID-19 imaging detection models, none are sufficient for clinical use due to methodological faults and biases. In their conclusion, they offer improvements such as *"(1) informing the clinician of which features in the data most influenced the prediction of the model, (2) linking the prognostic features to the underlying biology and (3) overlaying an activation/saliency map on the image to indicate the region of the image that influenced the model's prediction"*. The pipeline in this project tackles and contributes towards each of the 3 improvement suggestions.

Conclusion

The failure of COVID-19 imaging research to translate to applied clinical settings is an alarmingly worrying fact for the whole medical imaging field. Even more so, for urgent and impactful medical emergencies like COVID-19. It is of the highest importance to address these issues when building AI systems. By confirming the hypotheses and realizing the aims, the project's explainability pipeline tackles key concerns and is a step in the right direction towards building AI systems. Although it has its downsides and should not be used solely for high stakes decisions, It helps in understanding model decision making to certain lung characteristics associated with COVID-19, data and model biases. The explainability is not only for debugging purposes but paves the way for

AI systems that would be: applicable in practice, human-centred, and trustworthy towards multiple stakeholders in the development of AI systems.

5. Future work

A deep understanding of metrics for evaluating explainability methods would be useful as the explainability techniques become more and more used. An approach of combining subjective and quantitative metrics should be explored. Also, if more annotated data for specific lung abnormalities is available, it can be utilized to improve the contextualization of the explanations.

A potential explainability technique that could be added to the pipeline is Neural-backed decision trees (Wan et al., 2021). This method tries to make deep learning models interpretable rather than explain the "black boxes". With some alterations they make neural networks work as decision trees not sacrificing accuracy for interpretability. To achieve this, Instead of the last linear layer, a surrogate loss and differentiable sequence of decisions are added. Although there have been proven advances in explainability, more research is needed on the effect of these methods on the end-users, especially in medical environments with patients and medical professionals. This is recognized as an issue in the conclusion of (Singh et al., 2020). The efforts of this project could be extended to aid this issue by assessing the quality of the outputs through both qualitative and quantitative means. Lastly, this project can be further developed into a full Python package for the convenience of use and versioning (from a software development aspect).

Acknowledgements

I would like to thank my mentors Elsa Angelini and Arijit Patra for the opportunity to do this project with AstraZeneca but also for their support, creative ideas, and critical thought throughout the duration of the project.

Bibliography

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., & Süsstrunk, S. (2012). SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(11), 2274–2282. https://doi.org/10.1109/TPAMI.2012.120

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., & Kim, B. (2020). Sanity Checks for Saliency Maps. *ArXiv:1810.03292 [Cs, Stat]*. http://arxiv.org/abs/1810.03292

Banerjee, I., Bhimireddy, A. R., Burns, J. L., Celi, L. A., Chen, L.-C., Correa, R., Dullerud, N., Ghassemi, M., Huang, S.-C., Kuo, P.-C., Lungren, M. P., Palmer, L., Price, B. J., Purkayastha, S., Pyrros, A., Oakden-Rayner, L., Okechukwu, C., Seyyed-Kalantari, L., Trivedi, H., … Gichoya, J. W. (2021). *Reading Race: AI Recognises Patient's Racial Identity In Medical Images*. https://arxiv.org/abs/2107.10356v1

Beede, E., Baylor, E., Hersch, F., Iurchenko, A., Wilcox, L., Ruamviboonsuk, P., & Vardoulakis, L. M. (2020). A Human-Centered Evaluation of a Deep Learning System Deployed in Clinics for the Detection of Diabetic Retinopathy. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (pp. 1–12). Association for Computing Machinery. https://doi.org/10.1145/3313831.3376718

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828. https://doi.org/10.1109/TPAMI.2013.50

Chattopadhyay, A., Sarkar, A., Howlader, P., & Balasubramanian, V. N. (2018). Grad-CAM++: Improved Visual Explanations for Deep Convolutional Networks. *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 839–847. https://doi.org/10.1109/WACV.2018.00097

Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. *ArXiv:1610.02357 [Cs]*. http://arxiv.org/abs/1610.02357

Das, A., & Rad, P. (2020). Opportunities and Challenges in Explainable Artificial Intelligence (XAI): A Survey. *ArXiv:2006.11371 [Cs]*. http://arxiv.org/abs/2006.11371

Dong, D., Tang, Z., Wang, S., Hui, H., Gong, L., Lu, Y., Xue, Z., Liao, H., Chen, F., Yang, F., Jin, R., Wang, K., Liu, Z., Wei, J., Mu, W., Zhang, H., Jiang, J., Tian, J., & Li, H. (2021). The Role of Imaging in the Detection and Management of COVID-19: A Review. *IEEE Reviews in Biomedical Engineering*, *14*, 16–29. https://doi.org/10.1109/RBME.2020.2990959

Fauw, J. D., Ledsam, J. R., Romera-Paredes, B., Nikolov, S., Tomasev, N., Blackwell, S., Askham, H., Glorot, X., O'Donoghue, B., Visentin, D., Driessche, G. van den, Lakshminarayanan, B., Meyer, C., Mackinder, F., Bouton, S., Ayoub, K., Chopra, R., King, D., Karthikesalingam, A., … Ronneberger, O. (2018). Clinically applicable deep learning for diagnosis and referral in retinal disease. *Nature Medicine*, *24*(9), 1342–1350. https://doi.org/10.1038/s41591-018-0107-6

Fernández-Loría, C., Provost, F., & Han, X. (2020). Explaining Data-Driven Decisions made by AI Systems: The Counterfactual Approach. *ArXiv:2001.07417 [Cs, Stat]*. http://arxiv.org/abs/2001.07417

Gotkowski, K., Gonzalez, C., Bucher, A., & Mukhopadhyay, A. (2020). M3d-CAM: A PyTorch library to generate 3D data attention maps for medical deep learning. *ArXiv:2007.00453 [Cs]*. http://arxiv.org/abs/2007.00453

Healthcare AI systems that put people at the center. (2020, April 25). Google. https://blog.google/technology/health/healthcare-ai-systems-put-people-center/

Huda, W., & Slone, R. M. (2003). *Review of Radiologic Physics*. Lippincott Williams & Wilkins.

Kermany, D. S., Goldbaum, M., Cai, W., Valentim, C. C. S., Liang, H., Baxter, S. L., McKeown, A., Yang, G., Wu, X., Yan, F., Dong, J., Prasadha, M. K., Pei, J., Ting, M. Y. L., Zhu, J., Li, C., Hewett, S., Dong, J., Ziyar, I., … Zhang, K. (2018). Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning. *Cell*, *172*(5), 1122-1131.e9. https://doi.org/10.1016/j.cell.2018.02.010

Kumar, I. E., Venkatasubramanian, S., Scheidegger, C., & Friedler, S. (2020). Problems with Shapley-value-based explanations as feature importance measures. *ArXiv:2002.11097 [Cs, Stat]*. http://arxiv.org/abs/2002.11097

Lundberg, S., & Lee, S.-I. (2017). *A Unified Approach to Interpreting Model Predictions*. https://arxiv.org/abs/1705.07874v2

Lundberg, S. M., Nair, B., Vavilala, M. S., Horibe, M., Eisses, M. J., Adams, T., Liston, D. E., Low, D. K.-W., Newman, S.-F., Kim, J., & Lee, S.-I. (2018). Explainable machine-learning predictions for the prevention of hypoxaemia during surgery. *Nature Biomedical Engineering*, *2*(10), 749–760. https://doi.org/10.1038/s41551-018-0304-0

Morozov, S. P., Andreychenko, A. E., Pavlov, N. A., Vladzymyrskyy, A. V., Ledikhova, N. V., Gombolevskiy, V. A., Blokhin, I. A., Gelezhe, P. B., Gonchar, A. V., & Chernina, V. Y. (2020). MosMedData: Chest CT Scans with COVID-19 Related Findings Dataset. *MedRxiv*, 2020.05.20.20100362. https://doi.org/10.1101/2020.05.20.20100362

Nagasubramanian, K., Jones, S., Singh, A. K., Sarkar, S., Singh, A., & Ganapathysubramanian, B. (2019). Plant disease identification using explainable 3D deep learning on hyperspectral images. *Plant Methods*, *15*(1), 98. https://doi.org/10.1186/s13007-019-0479-8

Nie, W., Zhang, Y., & Patel, A. (2020). A Theoretical Explanation for Perplexing Behaviors of Backpropagation-based Visualizations. *ArXiv:1805.07039 [Cs]*. http://arxiv.org/abs/1805.07039

Ras, G., Ambrogioni, L., Haselager, P., van Gerven, M. A. J., & Güçlü, U. (2020). Explainable 3D Convolutional Neural Networks by Learning Temporal Transformations. *ArXiv:2006.15983 [Cs]*. http://arxiv.org/abs/2006.15983

Raumviboonsuk, D. P., Krause, J., Chotcomwongse, D. P., Sayres, R. A., Raman, R., Widner, K., Campana, B., Phene, S., Hemarat, K., Tadarati, M., Silpa-Archa, S., Limwattanayingyong, J., Rao, C., Kuruvilla, O., Jung, J., Tan, J., Orprayoon, S., Kangwanwongpaisan, C., Sukumalpaiboon, R., … Webster, D. (2019). Deep learning versus human graders for classifying diabetic retinopathy severity in a nationwide screening program. *Nature Partner Journal (Npj) Digital Medicine*. https://www.nature.com/articles/s41746-019-0099-8

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why Should I Trust You?": Explaining the Predictions of Any Classifier. *ArXiv:1602.04938 [Cs, Stat]*. http://arxiv.org/abs/1602.04938

Roberts, Roberts, M., Driggs, D., Thorpe, M., Gilbey, J., Yeung, M., Ursprung, S., Aviles-Rivero, A. I., Etmann, C., McCague, C., Beer, L., Weir-McCall, J. R., Teng, Z., Gkrania-Klotsas, E., Rudd, J. H. F., Sala, E., & Schönlieb, C.-B. (2021). Common pitfalls and recommendations for using machine learning to detect and prognosticate for COVID-19 using chest radiographs and CT scans. *Nature Machine Intelligence*, *3*(3), 199–217. https://doi.org/10.1038/s42256-021-00307-0

Sattarzadeh, S., Sudhakar, M., Plataniotis, K. N., Jang, J., Jeong, Y., & Kim, H. (2021). Integrated Grad-CAM: Sensitivity-Aware Visual Explanation of Deep Convolutional Networks via Integrated Gradient-Based Scoring. *ArXiv:2102.07805 [Cs]*. http://arxiv.org/abs/2102.07805

Sayres, R., Taly, A., Rahimy, E., Blumer, K., Coz, D., Hammel, N., Krause, J., Narayanaswamy, A., Rastegar, Z., Wu, D., Xu, S., Barb, S., Joseph, A., Shumski, M., Smith, J., Sood, A. B., Corrado, G. S., Peng, L., & Webster, D. R. (2019). Using a Deep Learning Algorithm and Integrated Gradients Explanation to Assist Grading for Diabetic Retinopathy. *Ophthalmology*, *126*(4), 552–564. https://doi.org/10.1016/j.ophtha.2018.11.016

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2020). Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. *International Journal of Computer Vision*, *128*(2), 336–359. https://doi.org/10.1007/s11263-019-01228-7

Shapley, L. S. (1951). *Notes on the N-Person Game — II: The Value of an N-Person Game*. https://www.rand.org/pubs/research_memoranda/RM0670.html

Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *ArXiv:1312.6034 [Cs]*. http://arxiv.org/abs/1312.6034

Singh, A., Sengupta, S., & Lakshminarayanan, V. (2020). Explainable deep learning models in medical image analysis. *ArXiv:2005.13799 [Cs, Eess]*. http://arxiv.org/abs/2005.13799

Spinner, T., Schlegel, U., Schafer, H., & El-Assady, M. (2019). explAIner: A Visual Analytics Framework for Interactive and Explainable Machine Learning. *IEEE Transactions on Visualization and Computer Graphics*, 1–1. https://doi.org/10.1109/TVCG.2019.2934629

Springenberg, J. T., Dosovitskiy, A., Brox, T., & Riedmiller, M. (2015). Striving for Simplicity: The All Convolutional Net. *ArXiv:1412.6806 [Cs]*. http://arxiv.org/abs/1412.6806

Wan, A., Dunlap, L., Ho, D., Yin, J., Lee, S., Jin, H., Petryk, S., Bargal, S. A., & Gonzalez, J. E. (2021). NBDT: Neural-Backed Decision Trees. *ArXiv:2004.00221 [Cs]*. http://arxiv.org/abs/2004.00221

Wang, H., Wang, Z., Du, M., Yang, F., Zhang, Z., Ding, S., Mardziel, P., & Hu, X. (2020). Score-CAM: Score-Weighted Visual Explanations for Convolutional Neural Networks. *ArXiv:1910.01279 [Cs]*. http://arxiv.org/abs/1910.01279

Xie, Y., Nguyen, Q. D., Hamzah, H., Lim, G., Bellemo, V., Gunasekeran, D. V., Yip, M. Y. T., Lee, X. Q., Hsu, W., Lee, M. L., Tan, C. S., Wong, H. T., Lamoureux, E. L., Tan, G. S. W., Wong, T. Y., Finkelstein, E. A., & Ting, D. S. W. (2020). Artificial intelligence for teleophthalmology-based diabetic retinopathy screening in a national programme: An economic analysis modelling study. *The Lancet Digital Health*, *2*(5), e240–e249. https://doi.org/10.1016/S2589-7500(20)30060-1

Yang, C., Rangarajan, A., & Ranka, S. (2018). Visual Explanations From Deep 3D Convolutional Neural Networks for Alzheimer's Disease Classification. *AMIA Annual Symposium Proceedings*, *2018*, 1571–1580.

Zeiler, M. D., & Fergus, R. (2013). Visualizing and Understanding Convolutional Networks. *ArXiv:1311.2901 [Cs]*. http://arxiv.org/abs/1311.2901

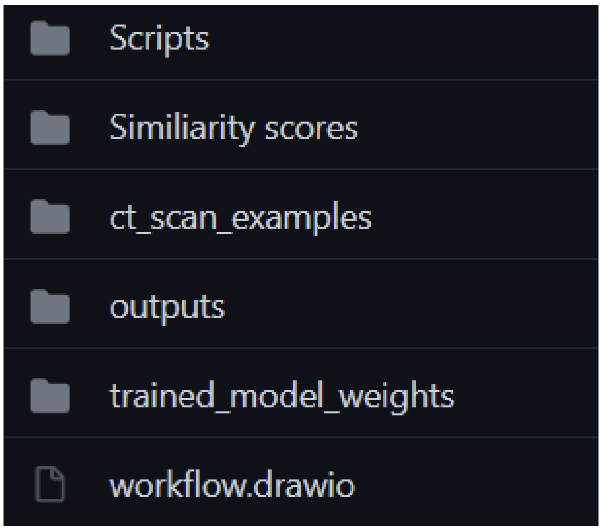
Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning Deep Features for Discriminative Localization. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2921–2929. https://doi.org/10.1109/CVPR.2016.319

Zunair, H., Rahman, A., Mohammed, N., & Cohen, J. P. (2020). *Uniformizing Techniques to Process CT scans with 3D CNNs for Tuberculosis Prediction*. https://arxiv.org/abs/2007.13224v1

Appendix A

Details on the Github repository

The code files are available at: https://github.com/fm1320/ICL/tree/main/AstraZeneca
There are multiple scripts and multiple folders which have different generated outputs in them.


Scripts
Similiarity scores
ct_scan_examples
outputs
trained_model_weights
workflow.drawio

"Scripts" folder has the python scripts that are used for different tasks in the project
"Similarity scores" has the binary maps from our masks and the expert's masks as well as the full version of the expert's masks
"Ct_scan_examples" has examples of a full resolution CT scan
"Outputs" has screenshot visualizations of some scans and their activations, overlays, as well as 3D nifti files of the class activation maps
"trained_model_weights" the trained model weights for the 3D classification
"Workflow.drawio" is a diagram of the explainability pipeline