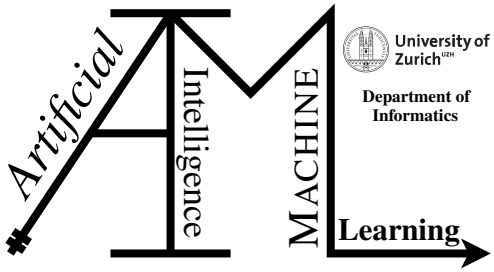
Explaining CNN-Based Active Tuberculosis Detection in Chest X-Rays through Saliency Mapping Techniques

Master Thesis

Özgür Acar Güler
15-705-445

Submitted on
September 1 2023

Thesis Supervisor
Prof. Dr. Manuel Günther
Dr. André Anjos



Master Thesis

Author: Özgür Acar Güler, oezgueracar.gueler@uzh.ch

Project period: March 1 2023 - September 1 2023

Artificial Intelligence and Machine Learning Group
Department of Informatics, University of Zurich

Acknowledgements

First and foremost, I want to express my deepest gratitude to Prof. Dr. Manuel Günther and Dr. André Anjos, without whom this work would not have been possible. Their level of supervision and support has been unparalleled, surpassing any academic guidance I have previously received. Both have generously devoted their time and expertise to ensure that I received the support I needed. They have not only answered my questions and guided me on the correct academic path when I was uncertain, but they have also offered invaluable emotional encouragement, inspiring me to persist and excel in my endeavours. The supervision I received from them is something that every student deserves and aspires to have. I am profoundly grateful for their unwavering faith and for pushing me to become a better researcher.

I would like to extend my heartfelt thanks to my family, especially my mother, father, brother, and Shimba, who have been my emotional cornerstone throughout this journey. Your unconditional love, unwavering support, and sacrifices big and small have not only made this academic endeavour possible but have also shaped the person I am today. Special recognition and love go to my girlfriend, Gan-Erdene Altansukh, whose constant encouragement, emotional support, and unwavering faith in me have been my driving force. To my friends who have stood by me, your encouragement and companionship have been invaluable, and I extend my deepest thanks for your understanding and patience when my academic commitments kept me away. Special thanks to Fabian E. Özpamir for his invaluable support in grammar correction. I am grateful for the strength and resilience that all of your support has bestowed upon me.

Abstract

Tuberculosis (TB) is an infectious disease caused by the bacterium Mycobacterium tuberculosis, which is one of the leading causes of death worldwide. Various Deep Convolutional Neural Network models have gained popularity to help during the TB screening process by detecting patients with active Tuberculosis from their Chest X-Rays. To help with further advancing the research, a new publicly available dataset, TBX11K, has been used to increase the number of samples during training for existing replaceable state-of-the-art models. In the first step, the model's performance was evaluated to see if an improvement through the addition of more TB-related data was observable. It was shown that state-of-the-art replicable binary classifier models could further be improved through the inclusion of more data. Further, there is a lack of focus on generating and evaluating explanations for such models. The preferred methods currently are saliency mapping techniques such as Grad-CAM, to generate visual explanations based on the model's decision-making process, by overlaying heatmaps over the Chest X-Rays. The selected TBX11K dataset includes ground truth bounding box labels, which makes it possible to evaluate if the visualisations were correct. There are various evaluation metrics to evaluate the faithfulness and localisation performance of the saliency mapping techniques according to ground truth labels. Two of them have been identified to be useful, namely RemOve and Debias, and Proportional Energy. RemOve and Debias was used to observe if there is one universal saliency mapping technique that performs well for all models for the task of active Tuberculosis detection. Further, based on these two metrics, a new metric was proposed, ROAD-Normalised PropEng Average, to measure the overall best-performing model and Saliency Mapping Technique combination. From the evaluation with RemOve and Debias, it was concluded that there does not seem to be a universal saliency mapping technique that performs well on all model architectures for the detection of active Tuberculosis. Thus, it is recommended to always consider the underlying model before choosing the optimal saliency mapping technique. Further, through the use of the ROAD-Normalised PropEng Average, it was concluded that one model in combination with a saliency mapping technique offered the best trade-off between faithfulness and correctness of the visualisations. This was the multi-label DenseNet-121 model with Eigen-CAM. To obtain accurate classifications of active Tuberculosis with explainable and correct visualisations, it is recommended to use this model and visualisation technique combination.

Contents

List of Abbreviations

Chapter 1

Introduction

Tuberculosis (TB) is an infectious disease caused by the bacterium *Mycobacterium tuberculosis*. Before the *Coronavirus Disease 2019 (COVID-19)* pandemic, Tuberculosis was responsible for more deaths per year than any other infectious disease. With an estimated 1.6 million deaths in 2021, it still remains one of the top causes of death worldwide.

It is estimated that approximately one-fourth of the world's population lives with a *Latent Tuberculosis Infection (LTBI)* (Houben and Dodd, 2016; World Health Organization, 2022). A latent tuberculosis infection refers to a condition in which a person has been infected with the bacterium, but the infected person is not showing any symptoms. The bacteria remain in the body in a dormant state and the person remains non-contagious. 90% to 95% of the infected people either remain in this state during their lifetime or in some instances, clear the infection. The remaining 5% to 10% develop the *active Tuberculosis (aTB)* disease, typically within the first 5 years of the initial infection. The risk of the disease being developed increases with several factors, including a frail immunological status (e.g., having AIDS, diabetes, being malnourished), smoking, and the presence of alcohol consumption disorders. A person with active TB can transmit the bacterium through the air by coughing, sneezing, or spitting. Most of the time, active TB affects the lungs, in which case it is also called Pulmonary Tuberculosis (PTB), but it can affect other parts of the body too (extra-pulmonary TB) (World Health Organization, 2020, 2022). About 50% of the infected die if the active TB disease is left untreated, and if treated, 85% of them can be cured.

As in the previous years, a majority of the 10.6 million TB infections in 2021 occurred in low- and middle-income countries. In an attempt to combat Tuberculosis globally, the *United Nations (UN)* adopted the *End TB Strategy* developed by the *World Health Organization (WHO)*. The End TB Strategy includes milestones for certain goals, including an absolute reduction in TB-related deaths by 90%, and the reduction of the TB incidence rate (new cases per 100'000 of the population/year) by 80% until 2030. The requirements to reach these goals are divided into 3 pillars, of which one is the *integrated, patient-centred care and prevention* pillar. According to this pillar, TB needs to be screened for, detected, diagnosed early, prevented, and treated until the patient is cured. To perform these tasks and reach the milestones, more domestic and international funding is needed in low- and middle-income countries (World Health Organization, 2022).

Both, LTBI and aTB, can be detected and treated. If a patient has an LTBI, the goal is to stop the patient from developing aTB by starting a preventive treatment. However, the treatments for LTBI and aTB are different from each other, and going through a preventive treatment while already having aTB should be avoided, as such a treatment is not very effective and can even be counterproductive for a patient. In Figure 1.1, the World Health Organization (2020) has created a flowchart that serves as an orientation on how the different risk groups within a population should be tested for LTBI, and when a preventive treatment should be initiated for them. HIV-positive patients and patients who had contact with a household member with TB have their own starting points. For patients within the "Other risk group" or "Household contract" group

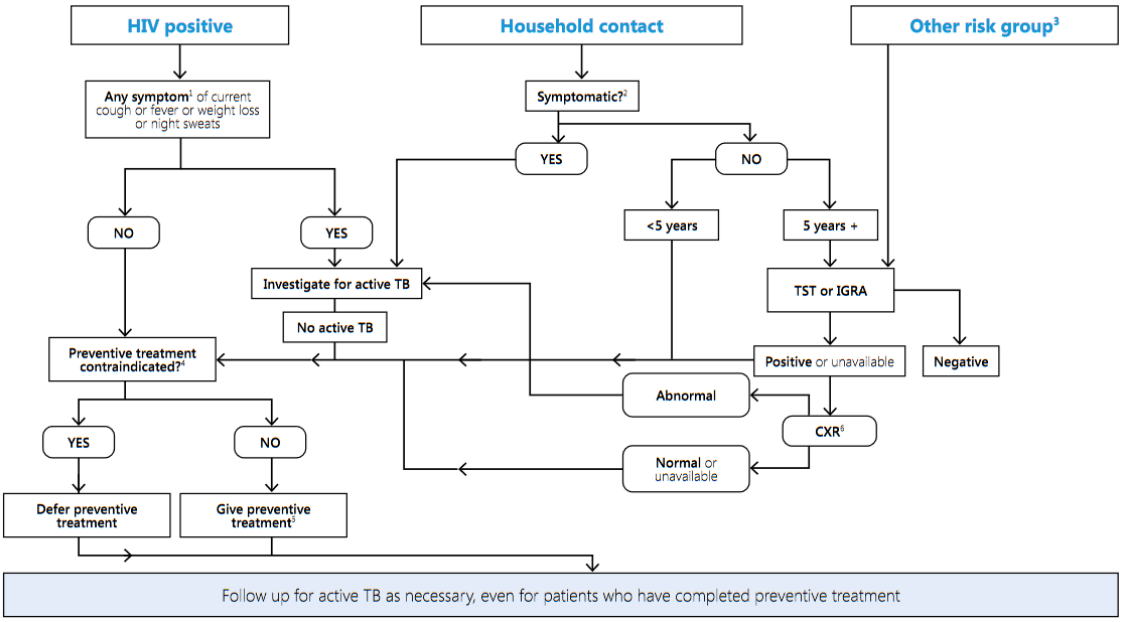
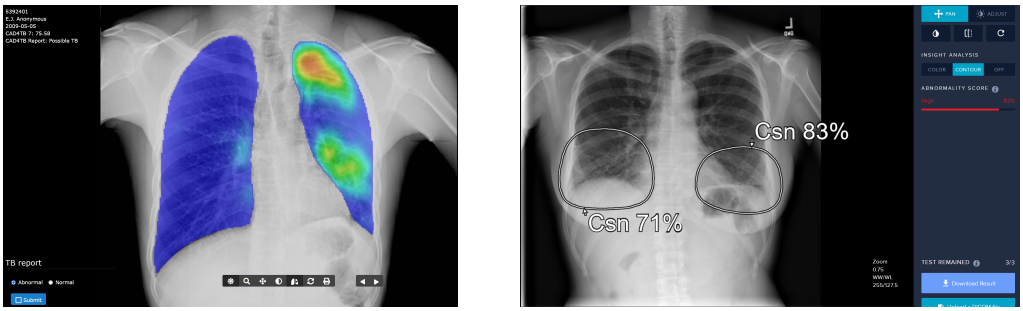


Figure 1.1: ALGORITHM FOR LTBI TESTING AND TB PREVENTIVE TREATMENT IN INDIVIDUALS AT RISK. *This flow chart depicts the decision points when ruling out aTB, testing for LTBI, and deciding if a preventive treatment should be initiated for a patient within a risk group (World Health Organization, 2020).*

that are at least 5 years old, a Tuberculin Skin Test (TST) or Interferon-gamma Release Assay (IGRA) needs to be conducted to test for TB. Both tests cannot distinguish between LTBI or aTB, a positive test could mean the presence of either of them. If the test is positive or not available, it is recommended to use a Chest X-Ray (CXR) to detect any abnormalities to assess if the patient has aTB or not before deciding on a preventive treatment.

As part of the first pillar, the World Health Organization (2021) also recommends using systematic active screenings of selected risk groups to detect aTB. When screening for active (P)TB, a high accuracy for detecting it is achieved by using a CXR screening test as part of the screening algorithm. In a meta-analysis conducted by WHO consisting of 19 cross-sectional studies, a sensitivity of 85% and a specificity of 96% is achieved when diagnosing abnormalities suggestive of TB with the help of CXRs. Depending on screening goals and the cost, in addition to the findings with the demographic and clinical data, a second screening method can be used in combination with CXR screening. The chosen screening algorithm configuration can further affect the sensitivity and specificity of the overall screening process.

All in all, using the rapid imaging technique CXR to detect signs of PTB is a very crucial part of the fight against the TB epidemic (Harris et al., 2019; Shen et al., 2010; World Health Organization, 2020, 2021, 2016), one that has been used for over 100 years (Williams, 1907). Unfortunately, using CXR analysis can be expensive, not only for the provider of the service due to the required equipment and qualified staff but also for the patient if there are no nearby health facilities for taking the CXRs (World Health Organization, 2020, 2021). Further, examining the CXR manually and visually is demanding for trained readers and experienced radiologists, and the number of available radiologists is often limited (Pande et al., 2015; Rajpurkar et al., 2018; Shen et al., 2010; World Health Organization, 2021). Due to these reasons, CXR use is limited in low- and middle-income countries with a high TB burden (Pande et al., 2015), partly attributable to the fact that the



(a) CAD4TB (b) Lunit Insight CXR

Figure 1.2: ABNORMALITY DETECTION BY COMMERCIAL CAD SOFTWARE. *(a) shows an example diagnosis by CAD4TB with the TB abnormality score (not visible in this figure) in the upper left and the heatmap of the TB-affected area (Delft Imaging, 2023) and (b) is an example diagnosis by Lunit Insight CXR, showing the contour of the detected abnormality together with its score (Lunit Insight, 2023).*

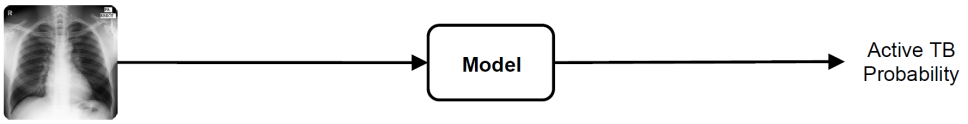
national annual funding is reported to be insufficient to mitigate the TB pandemic according to the End TB Strategy which leads to fewer spending of these funds in screening activities (World Health Organization, 2022, 2021).

To combat the shortage of radiologists, ease access to CXR screenings, and improve the cost, efficiency, and accuracy of CXR readings, the use of Computer-Aided Diagnosis (CAD) technologies has been suggested (Pande et al., 2015; World Health Organization, 2021). Thanks to the resurgence and advancements of Artificial Intelligence (AI) technologies, the CAD tools for detecting active TB in CXRs have been improved (Harris et al., 2019), reportedly sometimes even surpassing the performance of human readers (Qin et al., 2021; Rajpurkar et al., 2018). The World Health Organization (2021) recommends the use of such CAD software for the interpretation of CXR in screenings and triage for aTB, as long as the software performs at least as well as the 3 software on the market that a Guidelines Development Group convened by WHO has reviewed.[1] The results of commercially available software are usually shown as a score between 0 and 100, which is not necessarily a percentage indicating the risk of TB or an abnormality, or as a simple yes/no. Additionally, a heatmap or contour highlighting the areas where the CAD has detected pathologies or conditions is shown, sometimes with auto-generated text describing the location of the findings (Delft Imaging, 2023; Lunit Insight, 2023; Qure.ai, 2021). Figure 1.2 shows two different example diagnoses by two of the commercially available CAD software recommended by WHO. Radiologists can use the output of such CAD to increase their performance and efficiency when diagnosing aTB (World Health Organization, 2021; Rajpurkar et al., 2020).

Traditionally, the detection of PTB by CXR is done by detecting a combination of *radiological abnormalities* (or *radiological signs*, *radiological findings*, *lesions*, or simply *abnormalities)* (Curvo-Semedo et al., 2005). Some abnormalities, like cavities, are highly specific to PTB, while some other abnormalities are not specific to PTB only, and can indicate the presence of other pathologies as well (World Health Organization, 2016). This information is usually combined with other methods and the patient's data to diagnose aTB.

As for AI and Deep Learning (DL) models detecting aTB, Raposo (2021) categorises them into two groups: direct models and indirect models (Figure 1.3). While direct models take the CXR image as input to only predict the aTB probability, indirect models first detect the radiological

[1] Additionally, WHO recommends the use of CAD only for patients aged 15+ and only for antero-posterior or postero-anterior views of CXR for PTB. The respective software products are: CAD4TB v6, from Delft Imaging; Lunit Insight CXR, from Lunit Insight; and qXR v2, from Qure.ai



(a) Direct Detection Model

(b) Indirect Detection Model

Figure 1.3: TYPES OF MODELS FOR DETECTING ATB. *Types of models for detecting aTB* The 2 figures depict the two categorisations of AI models evaluating a CXR for their probability of having aTB (Raposo, 2021). While the direct model (a) directly outputs the probability for aTB, the first sub-model of the indirect model (b) first returns the labels of the detected radiological signs before the second sub-model uses them to output the probability for aTB.

sign probability and detect the probability of aTB based on the N binary labels of the detected radiological signs. According to Raposo (2021), the direct method is the preferred method to predict aTB in the literature, as the author was only able to identify a single study by Rajpurkar et al. (2020) that used an indirect method for aTB prediction. Raposo (2021) uses publicly available data to train and compare 3 types of models against each other. The first is a direct model trained on CXRs only. The second is an indirect model that is a combination of a DenseNet-121 sub-model that was first trained to recognise radiological signs and a second sub-model that predicted aTB based on the binarised radiological sign labels from the first sub-model. The third direct model is a slightly modified and fine-tuned version of the first sub-model of the indirect model. Both the first sub-model of the indirect model and the third model were pre-trained on the *ImageNet* dataset before being trained on the same CXR images again. The indirect model performed better than the basic direct one while showing state-of-the-art results, but it did not outperform the third (direct) model. The author claims that, while performing worse than the third model, the indirect model is more interpretable as it is closer to the clinical workflow of radiologists diagnosing aTB due to the intermediate step. The author continues to argue that similar performance to the third model could be achieved with the indirect model if there were datasets available specifically adapted to PTB detection, labelled and annotated with the radiological signs. Additionally, the precision of the Gradient-weighted Class Activation Mappings (Grad-CAM) or similar high-level visualisation methods could be evaluated thanks to such a dataset.

With CADs based on AI and DL models detecting aTB emerging as commercial software and surpassing human performance (Qin et al., 2021), as with any AI-related models, the question of explainability and interpretability arises (Gilpin et al., 2018). As mentioned, trained physicians and radiologists are looking for abnormalities and the patient's information to diagnose aTB. The diagnosis is made based on the knowledge and experience of the physician. The physician can explain the thought process and answer the questions of the patient to justify the diagnosis. However, when a modern Neural Network (NN) outputs a result, the algorithm or calculations it uses to arrive at the result appear to be more of a black box to the user of the CAD. The inability to look into the model to understand it, due to the "memory being encoded into the strengths of the connections" (Castelvecchi, 2016) of the neurons, just like in a human brain, further exacerbates this issue. Castelvecchi (2016) compares the undertaking of understanding such a model to the opening and dissection of a human brain. Explainability is however needed to ensure reliability,

trust, accountability, and fairness, and it can be used as an additional tool to identify errors and biases in the data (Castelvecchi, 2016; Cutillo et al., 2020; Gilpin et al., 2018).

In the context of this thesis, this poses the question if AI-based computer-aided diagnosis of active TB is explainable. Visualisation techniques like *saliency mapping* or *class activation mappings*, heatmaps, and contours are popular in commercially available CADs and in the literature to highlight the locations of abnormalities, or regions, on which the aTB diagnosis is based on. Such visual explanations not only add interpretability and thus explainability to the model (Cutillo et al., 2020; Gilpin et al., 2018; Lipton, 2018), they have also reportedly increased the performance of radiologists when diagnosing aTB with the help of a CAD software (Rajpurkar et al., 2020) with heatmaps, and the authors suggest to further look into different interpretation methods to improve the performance of the radiologists.

This is what this work is building upon. Firstly, based on the findings and suggestions of Raposo (2021), another publicly available TB dataset named TBX11K with postero-anterior (frontal) view CXRs that are annotated with the locations of abnormalities has been selected. This new dataset will be used to train direct and indirect models similar to Raposo (2021) to evaluate if their performance can be further improved.

Secondly, another aim of this work will be to add interpretability and explainability to CADs based on aTB-related radiological signs to aid physicians in the diagnosis process. According to the recommendations by medical experts, the following will be pursued: newer activation- and gradient-based visualisation techniques for Convolutional Neural Networks will be researched, implemented, and compared to each other by using existing or by developing new appropriate performance measuring techniques, and by using the TBX11K dataset to visualise abnormalities and compare them to the ground truth labels.

1.1 Research Questions

From the previous introduction, the goals of this thesis can be split into two main parts, one dealing with the replication and potential improvement of current existing state-of-the-art models with a new dataset, the other exploring how the explainability of a model based on DL and radiological signs can be established. The following Research Questions (RQ) can be derived:

- **RQ 1**: Can the prediction of the probability of active Tuberculosis by direct and indirect deep learning models be improved through the use of a new dataset specific to active Tuberculosis?
  - **RQ 1.1**: How do the AUC scores from Raposo (2021) compare to the replicated models using the same methods and frameworks?
  - **RQ 1.2**: Does the inclusion of different types of labels in the TBX11K dataset (healthy, latent TB, sick & non-TB) affect the model's ability to discriminate active TB cases?
  - **RQ 1.3**: Does the inclusion of the TBX11K dataset during training affect the generalisability of the models?
  - **RQ 1.4**: How do the AUC scores of the models that included the TBX11K dataset during training compare to the replicated models based on Raposo (2021)?
- **RQ 2**: What novel visualisation methods exist for increasing the explainability of deep learning models trained on a dataset specific to active Tuberculosis by visualising radiological signs and how well do they perform?
  - **RQ 2.1**: What measurement techniques are suited to evaluate the performance of the visualisations?

Chapter 2

Related Work

The purpose of this chapter is to present a brief overview of the past and current research that has been conducted on detecting active Tuberculosis from Chest X-Rays with the help of *Computer-Aided Diagnosis* tools and the explainability methods that have been used to describe the results of such tools. The research gaps are being highlighted based on this research. Section 2.1 starts with general computer-based models and methods that have been used throughout the literature to detect abnormalities in medical images. Section 2.2 focuses specifically on AI models that have been used so far to reach state-of-the-art results in performance with automated detection approaches for Tuberculosis. Section 2.3 follows up with current methods that are being used to add explainability to AIs.

2.1 Computer-aided Diagnosis for Medical Images

Doi (2007) provides an overview of the history of research on CAD tools for medical images. The author mentions that the research on computer-based analysis of medical images started in the 1960s, although rather unsuccessful as they attempted to fully automate the diagnosis. The focus of research shifted to CAD Tools instead of fully automated diagnosis in the 1980s to aid radiologists with mainly breast cancer, cardiovascular diseases, and lung cancer detection. The researches from the 1980s successfully realised three principal ideas that laid the foundation for the development of such tools:

1. The algorithms should be based on the processes the radiologists are using when reading medical images.
2. Measure the performance of such tools by measuring their adoption.
3. Promote and demonstrate the usefulness of CADs to gain wider popularity and acceptance, and thus also to increase the research on the topic.

Since radiologists had trouble detecting e.g. lung nodules in CXR due to anatomical background structures within them, it was assumed that computer algorithms would have trouble too. Thus, the first part of the earlier computerised schemes that were developed was the *difference-image technique*, which attempted to remove or suppress the background "noise" from the actual findings. These methods had a sensitivity of as high as 85%, but also a low specificity due to the large number of false positives. Such a model still seemed to improve the performance of radiologists according to Chan et al. (1990).

After Chan et al. (1990), more researchers reported a statistically significant increase of performance in the detection of different findings within medical images, further fuelling the interest in

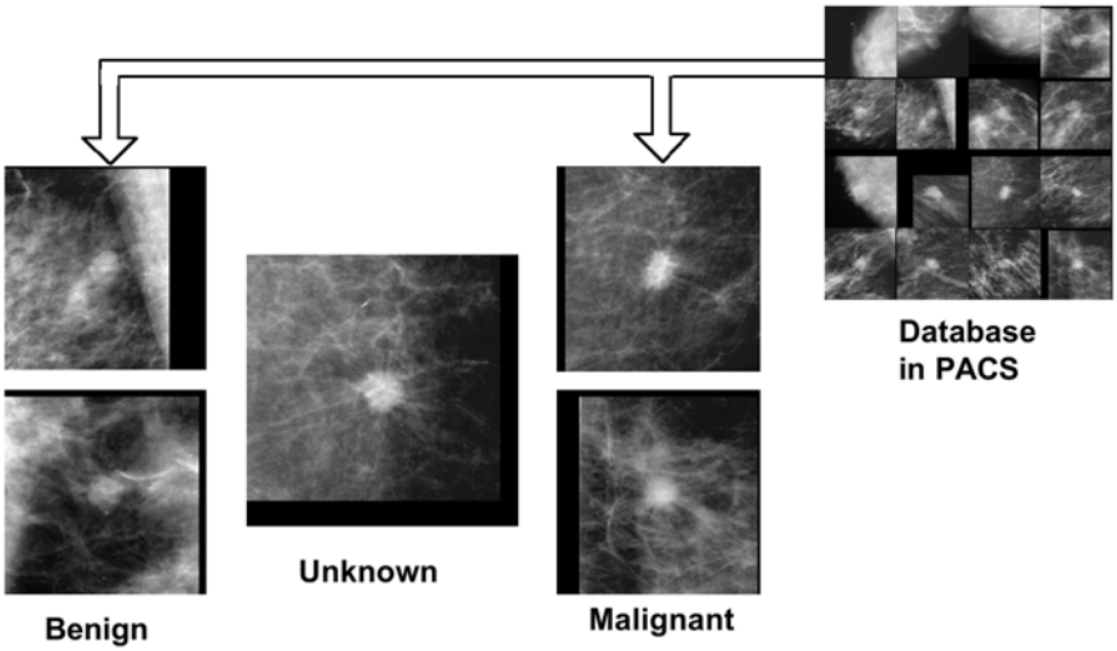


Figure 2.1: EXPLAINABILITY BY SIMILARITY FOR DIFFERENTIAL DIAGNOSIS. *This figure depicts an unknown mammogram case that needs to be diagnosed together with similar looking benign and malignant cases. Radiologists were able to identify the unknown case correctly more often as malignant thanks to this aid (Doi, 2007)*

CAD research and increasing their adoption rates in the US and Europe (Doi, 2007). From 1993 to 2005, some CAD tools have reportedly improved their sensitivity up to 97% while at the same time reducing their false positive rates.

As of 2007, there were CADs available that could detect several abnormalities within CXRs like lung nodules, vertebral fractures, cardiomegaly, and pneumothorax. Some schemes combined postero-anterior CXRs with lateral views of CXR to achieve a sensitivity of up to 86.9% in lung nodule detection.

An specific example of such a CAD tool is one developed by Li et al. (2003) to aid radiologists in lung cancer detection by presenting them computer tomography scan regions of benignant and malignant lung nodules that look similar to an unknown case to be diagnosed (see Figure 2.1). First, three characteristic features were selected from nodules based on similarity ratings by radiologists, and four different techniques (feature-based, pixel-value-difference based, NN based, and cross-correlation based) were examined to calculate the similarity measure of the new unknown case with the existing cases. The feature-based technique has reportedly performed the best by using the 3 selected nodule features to calculate the similarity measure. Such a CAD tool based on an integrated image retrieval technique has improved the performance of radiologists according to a study conducted one year later (Li et al., 2004).

Shen et al. (2010) have used an algorithm to detect and contour cavities in CXR, one of the typical radiological signs for postprimary TB. It uses a clustering technique called *enhanced mean shift segmentation with adaptive thresholding* to identify and mark the initial contours of the TB cavities in the image, and it further refines the contours and segments the abnormalities with a computer model called *GVF snake* with *Dirichlet boundary conditions*. Finally, Bayesian classification and other thresholding methods were used to improve the accuracy of the results. With this, the

authors were able to reach a sensitivity of 82.35%.

While these older techniques helped to improve the performance of radiologists by giving them a "second opinion" on a multitude of studies (Chan et al., 1990; Doi, 2007) when comparing the CADs standalone output diagnoses versus the physician's diagnoses, the CADs did not outperform the physicians on validation datasets or real settings. However, with the emergence of more sophisticated AI and DL models, such as the model performing similarly in detecting radiological signs as radiologists in the study of Rajpurkar et al. (2018), this is no longer necessarily the case today. Additionally, the older methods were usually highly limited and applicable to only specifically chosen abnormalities, while newer models can learn to distinguish multiple abnormalities at once (Rajpurkar et al., 2020; Raposo, 2021). Support Vector Machines (SVMs) (Chauhan et al., 2014) and Convolutional Neural Networks (CNNs) count to the more modern AI approaches (Pasa et al., 2019), although the focus recently has shifted more towards the latter, with one of the reasons being that they have shown more promising results (Harris et al., 2019; Pasa et al., 2019).

2.2 Pulmonary Tuberculosis Detection with AI

NNs have been used in CAD tool research for medical images even before the recent new-found popularity of AI models (Li et al., 2003), although the performance of such AI-based models have been improved significantly with the emergence of more sophisticated DL models (Harris et al., 2019).

The systematic review by Harris et al. (2019) of CADs in PTB detection resulted in an overview of the performance of different models in the literature. In this study, DL models performed better with a median Area Under the Curve (AUC) of 0.91 compared to Machine Learning models with 0.82. Biases in the methodology and data selection in some studies were identified, and the used datasets were mostly from populations with higher TB and HIV prevalence which led to generalisability problems for these models. Another issue with some of the mentioned studies is that they have used the test set during the training, which limits the generalisability of the models. The authors conclude that more research in the development of CAD for detecting PTB with CXR needs to be made and also suggest that more clinical as opposed to development studies are needed.

One of the best-performing models in Harris et al. (2019) is reportedly the one designed by Lakhani and Sundaram (2017). Lakhani and Sundaram (2017) have used pre-trained Deep CNNs to detect PTB on CXR. They have used *AlexNet* and *GoogLeNet* for transfer learning, both Deep CNNs trained on *ImageNet*, and augmented the Montgomery County and Shenzhen (see Section 4) dataset of postero-anterior CXRs by applying contrast changes and rotation to the images during the training. Further, regularisation and dropout were used to make the model more generalisable. They have concluded that an ensemble of the two pre-trained CNNs performed better than untrained models with an (unverifiable according to Raposo (2021)) AUC of 0.99. The authors have also tested a radiologist-augmented approach, where they send the CXRs that the ensemble disagrees on to a radiologist for an overread. This approach resulted in a sensitivity of 97.3% and specificity of 100%.

The second study from Harris et al. (2019), conducted by Heo et al. (2019), achieved the best results by combining the extracted features of CXRs from the pre-trained model *VGG19* (transfer learning) with four demographic variables, with an AUC of 0.9213. The four demographic variables were: weight, age, gender, and height. The inclusion of two demographic variables weight and age (AUC increase by 0.0137) seemed to oddly perform better than the inclusion of three demographic variables weight, age, and gender (by 0.0132), and the inclusion of demographic variables reportedly improved the sensitivity and model stability significantly (by 0.04 on sensi-

tivity).

Another well-performing model from Harris et al. (2019) is the one developed by Santosh and Antani (2018). The authors first automatically segment the left and right lung of each CXR to calculate an 11-dimensional vector with attributes that represent the symmetry differences between them. These symmetry differences between both lung parts are being used as a measure for the existence of abnormalities. The vectors are being fed into a Bayesian Network, a Random Forest and a simple NN and the AUC is calculated based on an ensemble of these 3 models. The authors reach an AUC of up to 0.96. Thus the authors show that the *bilateral lung-field asymmetry* can also be used to detect abnormalities and PTB in CXRs.

Pasa et al. (2019) and Yu-Jen Chen et al. (2015) note that many modern Machine Learning (ML) algorithms are complicated consisting of many steps, like lung segmentation and feature extractions (like the Santosh and Antani (2018) model). Deep CNNs, untrained and pre-trained, can achieve similar performance while being relatively simpler. Pasa et al. (2019) develop a custom CNN inspired by AlexNet's architecture of which the code is publicly available, showing that they can achieve similar results (AUC: 0.925) with fewer parameters when compared to bigger pre-trained models like GoogLeNet. The data was augmented and 5-fold cross-validation was used on the individual datasets. The authors point out that future work should include more data to increase classification accuracy and AUC of their model. Moreover, the authors point out the need for more in-depth research in visualisation techniques for trust and explainability for such DL models (see also Section 2.3).

Rajpurkar et al. (2020) developed a model for detecting PTB in CXRs specifically for co-infected patients with HIV and TB. TB in patients with HIV can often be fatal, and the detection of TB with other testing methods can be aggravated by the fact that HIV patients might not produce enough sputum to conduct them. The authors developed and tested a CAD with radiologists and observed an improvement in the diagnosis accuracy of the radiologists. They further observed that the CAD-assisted radiologists had a worse accuracy than just the standalone model itself. The model called *CheXaid* combines a pre-trained model on non-TB specific CXRs that has been fine-tuned on the TB-specific CXRs with clinical data to diagnose PTB, and the authors report an improvement of the accuracy from 0.61 to 0.79 through the inclusion of the clinical data. The same pre-trained model is also used to detect 6 different radiological signs that also influence the outcome of the overall model. Finally, Class Activation Maps (CAMs) were used to visualize heatmaps for the CXRs.

In an earlier study, Rajpurkar et al. (2018) used a non-TB-specific dataset of radiological signs in CXRs to train a pre-trained CNN and reported that the model performed as well as radiologists on 10 of the signs, better on 1 and worse on 3 in a non-clinical setting. For added interpretability, heatmaps were generated using CAMs. The authors conclude that it can be useful to use CNNs to detect radiological signs, in terms of accuracy and efficiency, as the model takes only 1.5 minutes to evaluate compared to the 240 minutes of a radiologist. The authors mention the limitations of their model, of one which is the lack of use of lateral view when comparing the performance of the model to radiologists. In clinical settings, lateral views are also being used for a diagnosis.

Raposo (2021) has completed a master thesis in the Idiap Research Institute on detecting aTB from CXR, and is in a sense the predecessor for this work. Raposo (2021) points out how research on aTB detection with AI is done with either private datasets or unpublished code, which makes most research irreproducible. It is pointed out that all the top performing DL models in the systematic review of Harris et al. (2019) are not reproducible. To fix this problem, an open-source toolbox called *bob* (as of 2023, renamed to ptbench) written in Python is being used to replicate and extend existing research on aTB detection. The code is made publicly available.[1] The author uses two different approaches, a direct and indirect approach as in Figure 1.3, to train models detecting aTB (see Section 3.1.4 for more details). The direct model type performed better with

[1] https://www.idiap.ch/software/biosignal/docs/biosignal/software/ptbench/main/sphinx/

AUC scores of up to 0.984, while the indirect model reached up to 0.966. The author concludes that an indirect model is naturally more interpretable than the direct model due to the intermediate step of identifying the radiological signs, and hypothesises that using a dataset annotated and labelled with radiological signs specific to aTB to train an indirect model would allow to avoid giving up interpretability for the sake of accuracy as it is with the direct model. The author also uses Grad-CAMs to visualise the results of the indirect model but points out that there is no ground truth data available to verify the correctness of the results. Finally, the author mentions as a limitation that TB-related datasets are small and not necessarily representative, and that TB-related datasets annotated with radiological signs could further help to improve the check the performance of the suggested models.

Liu et al. (2020) point out the lack of publicly available TB-related data and how this has affected progress in improving DL models, and their adoption in CADs. The authors have published the largest publicly available TB-related dataset TBX11K to address the problem. The dataset includes ground truth bounding boxes marking the areas of the CXR that show TB-related radiological abnormalities and also differentiates between *healthy*, *sick & non-TB*, *latent TB*, and *aTB*, compared to previous datasets that tend to distinguish between healthy and TB only. The authors suggest that the additional labels should be used to reduce the number of false positive classifications that can e.g. happen due to *sick & non-TB* cases being classified as TB-positive in real clinical scenarios. The authors propose a multi-branch model that can discriminate between *healthy*, *sick & non-TB*, and *TB* in one branch. The other branch can use that information to further discriminate between *latent TB* and *aTB*, and that branch is also used to generate visualisations. The authors reach AUC scores of up to 93.8% when differentiating between *non-TB* (i.e. healthy and sick & non-TB) and *TB* (i.e. latent and active).

Research in PTB detection with AI has made significant progress within the last decade, with various models showing promising capabilities. Yet, a lot of research gaps remain: reproducibility issues, generalisability issues of the models, interpretability versus accuracy trade-offs, diversity in data labelling, lack of publicly available TB datasets, effects of demographic variables on the models, emphasis on developmental research over clinical research, and lack of research on explainability methods for complex DL models, and more. Among these gaps, this thesis will primarily address the reproducibility and generalisability issues, the inclusion of more TB-related data for data-hungry DL models which also addresses the diversity in data labelling, and most importantly, the lack of research on explainability methods. Raposo (2021) formulates the problem of reproducibility due to the lack of publicly available data (Liu et al., 2020) and unpublished code, making the state-of-the-art results of many studies in Harris et al. (2019) unverifiable. In real-life scenarios, there can be a need to discriminate between more than just *healthy* and *TB* cases. The ability to discriminate and generalise state-of-the-art models when trained with more than two labels, as provided in the TBX11K dataset, also remains unexplored. Lastly, as complex DL models become better in PTB detection, interpreting how these models make their decisions, which is essential for increasing trust and adoption in the medical domain, remains a largely unexplored area in current research.

2.3 Explainability and AI in the Medical Domain

Holzinger et al. (2019) denotes that it is important to have explainable, interpretable, trustworthy, and transparent AI in the medical domain, specifically, a "possibility to understand how and why" the model has arrived at its result, which can also affect the diagnosis of a medical professional. The authors demand *causality* for the learnt *representations* and also introduce the notion of *causability* to measure how well a human achieves a specified level of causal understanding from an explanation. The authors acknowledge that the ground truth cannot always be well defined

when making a diagnosis, not even by medical professionals. So there is a need for a mapping of an AI model to a causal model if the model does not provide causality. Explainability, on the other hand, is only desirable for the decision-relevant parts of a model that contributed to the accuracy of the training set, or one single observation. Cutillo et al. (2020) also mention how trustworthiness, explainability, usability, transparency, and fairness are important factors to consider when using CADs. In particular, to reach explainability, the authors mention the need for visual explanations like saliency or activation maps, score outputs in an interface, and explanation-producing systems. The authors further note that explainability is not always needed if trust can be established. However, trust can only be established over time, so explainability is important for trust and acceptance when new systems are being introduced to the medical domain.

Amann et al. (2020) states that often when AI models are being developed, the focus lies on increasing the accuracy of the predictions to obtain a medical certification. However, AI systems cannot provide perfect accuracy due to various sources of error. The authors note how it is almost impossible to eliminate one of the sources of error despite best efforts, namely AI bias. Due to this, they argue that it is important to introduce explainability to AIs in the medical domain to resolve disagreements between medical experts and AIs. They mention how visual and natural language explanations can help medical experts when evaluating the recommendations of CADs.

Miró-Nicolau et al. (2022) conducted a systematic literature review to investigate the state of explainable AI for CXR analysis. Just as Raposo (2021), they conclude that most methods for CXR analysis were not open, and thus irreproducible. Further, more than half of the studies that included explainable AIs used saliency mapping techniques to achieve it, which are considered to be post-hoc explanations. The quality of the explainability methods (visual and textual) was not measured by most of the studies, which hinders the overall advancement in the field of explainable AI for CXR analysis. Thus, the authors suggest that the focus of research should shift to finding appropriate methods of measuring the quality of the explainability methods.

There have been multiple approaches in the literature to add explainability to the results of CAD tools and to measure the quality of the explainability methods. As one of the more popular and most recommended methods in the medical domain is visual explanations with saliency mapping techniques, there is a greater focus on it in Section 2.3.1. The remaining methods are briefly mentioned in Section 2.3.2.

2.3.1 Saliency Mapping Techniques

CADs showing a visual cue of the detected radiological findings is not only the most popular technique in AI-based CAD tools today (Groen et al., 2022; Miró-Nicolau et al., 2022), it is also one that has been used since the very early days of CAD tools (Chan et al., 1990; Doi, 2007) to aid the radiologists with their diagnoses.

Kundel et al. (1990) have used a computer-assisted approach to show radiologists visually where they might have missed any nodules. This approach improved the Alternative Free Response Operating Characteristic (AFROC) of the radiologists by 16% on average. The AFROC is a metric that is popular in radiology and general medical imaging. It is an extension of Receiver Operating Characteristic (ROC) (see Section 3.4.1) plotting sensitivity against the average number of false positives, which can be helpful when there are multiple abnormalities in one image.

Liu et al. (2020) conducted a human study with radiologists to measure the accuracy of the expert radiologists in detecting TB with the help of CXRs. The experts achieved an accuracy of 84.8% if they only need to recognise TB, and 68.7% if they also need to differentiate correctly between latent TB and aTB. They note how it can be challenging for the human eye to find the important TB areas, and underline the importance of having a model that can localise the TB-related areas to assist radiologists, as they hypothesise that this could improve the accuracy of the radiologists.

Lakhani and Sundaram (2017) point out that it is unreasonable to expect a fully complete explanation for a DL model and instead provide a simple heatmap visualisation of the activations of individual examples that are input into the pre-trained models to add post-hoc interpretability to the results, and to increase the trust into the models. Similarly, Hwang et al. (2019) used a localisation layer within the DL model to generate heatmaps to increase trust in the model.

Rajpurkar et al. (2018, 2020) have used CAMs to visualise heatmaps for the CXRs to add interpretability to their models by localising the areas that are most indicative of a radiological sign. To further test the model and CAMs as one package, they develop a CAD tool as a web interface and test if the performance of radiologists increases when they are assisted with the model's prediction and visualisation output. They show a statistically significant increase in the radiologist's accuracy from 0.6 without assistance to 0.65 with CAD assistance, although the authors do not examine how much of this increase can be attributed to the inclusion of the visual explanation. They further also test the stand-alone model without the radiologists on unseen data and achieve an accuracy of 0.79, which is higher than the assisted radiologist approach (0.65). The authors reason that this might be attributable to the mistrust of the radiologists in the tool, and the overconfidence of the radiologists in their own diagnosis.

Pasa et al. (2019) have used CAMs and Grad-CAMs and believe that such visualisation techniques allow for a visual diagnosis and increase the trust of the medical community in AIs while showing the limitations of the models, too. Raposo (2021) has also implemented Grad-CAMs based on this work to add interpretability to the model's results, however, the performance of the method was not evaluated due to missing ground truth data, just as in the study of Pasa et al. (2019). Both works suggest continuing the research in saliency mapping techniques due to their usefulness.

Groen et al. (2022) and Miró-Nicolau et al. (2022) have both conducted a systematic review of the state of explainability methods for AIs related to radiology. They both conclude that saliency mapping techniques are currently the most popular explainability technique for CADs that use AI, but they also highlight that there is a research gap in measuring the quality of the visualisations produced by saliency mapping techniques in the context of CADs, which slows down the adoption in the medical field.

Saliency mapping techniques for CNNs have evolved over the last decade since the initial basic implementation for CNNs by Simonyan et al. (2014) gained popularity. Various techniques have emerged: some based only on different types of scores calculated with the help of the network (e.g. Score-CAM by Wang et al. (2020)), some based on gradients only (e.g. FullGrad by Srinivas and Fleuret (2019)), some based on CAMs (e.g. Zhou et al. (2016)), or some based on a combination of CAMs and gradients (e.g. Grad-CAM by Selvaraju et al. (2017)), some based on a combination of CAMs and scoring systems (e.g. AblationCAM by Desai and Ramaswamy (2020)), etc. Currently, there is no clear indication that a single technique excels in all image tasks, nor is there an indication that a universally better technique exists specifically for CXR TB-detection tasks (Tomsett et al., 2020). A lack of standardised performance metrics in the accompanying publications of each technique and medical-related publications (Groen et al., 2022; Miró-Nicolau et al., 2022), along with variations in the underlying CNN models, complicates matters. Notably, the CNN model affects the performance of the saliency mapping techniques, making unconfounded comparisons of the performance of the different saliency mapping techniques more challenging. Results in existing literature can thus be inconsistent. The visualisation techniques in the medical field/TB-detection research currently seem to focus on Grad-CAMs or simple CAMs, or older techniques (Hwang et al., 2019; Lakhani and Sundaram, 2017; Pasa et al., 2019; Rajpurkar et al., 2018; Raposo, 2021; Wang et al., 2017) if any at all. The lack of publicly available ground truth data (bounding box annotations) in public datasets to evaluate the performance of such techniques could have been another factor why the progress in this direction was further slowed down (Groen et al., 2022; Liu et al., 2020; Miró-Nicolau et al., 2022; Raposo, 2021).

Since many more techniques have been released by now, it poses the question if these methods offer more or less in terms of explainability, faithfulness to the decision-making process of the models, and localisation performance, especially when being used in combination with models for aTB detection with CXRs.

2.3.2 Other Methods

There are more ways than saliency mapping techniques to add explainability to AI models for CXR analysis. There is a wide variety of explanation methods, some are standardised and some are custom. Although they are not as popular as saliency mapping techniques, overlooking them could mean missing out on potentially more effective or insightful approaches. The research on these methods is relatively scarce, and just as with saliency mapping techniques, the quality of the explanations they provide is not conclusive yet. Some potentially promising methods in the medical domain are briefly presented here. The methods introduced here are by no means exhaustive.

Retrieval of similar images Doi (2007) recommends the use of a system that can retrieve similar-looking benign and malignant regions of medical images to assist the radiologist with visual comparisons, although the recommendation seems to be based on its usefulness as reported by Li et al. (2004) and not specifically for the added explainability. Figure 2.1 depicts how such an explainability method works. Montenegro et al. (2021) mentions how the standard approach today is the *K-Nearest Neighbours* algorithm, and they also show a way of achieving a privatised image retrieval system with the help of Generative Adversarial Network (GAN)s by applying it to a medical dataset.

Human-readable Explanations Krishnamurthy et al. (2020) demonstrates a way to generate automated human-readable explanations (i.e. post-hoc explanations) for CADs with the help of explanation-producing models such that it becomes clear how input features contributed to the final prediction. The authors do this by characterising each node in the model with a set of features and detecting the most relevant ones for the classification through random forest classifiers. They further use these features to capture the classification patterns as a set of decision rules in the form of Boolean logic *sentences*. This way, the overall transparency and interpretability of the models are being increased.

LIME Local Interpretable Model-agnostic Explanations (LIME) are generated through approximation of the more complex AI model with a simpler model e.g. a linear regression. This is done by perturbing the input data, looking at the changes in the output, and using this information to train the simpler model to approximate the complex AI model. Depending on which segments of the image were most important for the model's decision, these sections can be then visualised. Abeyagunasekera et al. (2022) uses LIME on CXR as part of a more complex custom visual explanation method called LISA, consisting of the union of the results of multiple different explanation methods.

Chapter 3

Background

This chapter delves deeper into the important concepts and publications that were used as a basis for the approach and experiments (see Section 5). It can be divided into five main sections. The first one is the more technical Section 3.1 that covers the basic concepts like the architecture and training process of AI and CNN models. Then, Section 3.2 focuses on the details of different saliency mapping techniques that are being used to add visualisations, and thus interpretability and explainability, to the AI models. Afterwards, Section 3.4 focuses on the measurement techniques and metrics that are being used to evaluate the models and visualisations. Section 3.3 briefly mentions the important Tools and Frameworks that were used to train and evaluate the models and visualisations. Last but not least, Section 3.5 briefly establishes a terminology to have a common understanding of the explainability concepts.

3.1 Deep Learning Models for Tuberculosis Detection

Artificial Intelligence (AI) is a branch of computer science that focuses on creating machines capable of intelligent behaviour. It aims to develop systems that can perform tasks that typically require human intelligence and decision-making (Holzinger et al., 2019). *Machine Learning (ML)* is a sub-field of AI that aims to develop systems that automatically extract information and learn from data to improve itself (Harris et al., 2019; Holzinger et al., 2019; Lakhani and Sundaram, 2017) to solve a specific problem. *Deep Learning (DL)* is a sub-field of ML that utilises *Neural Network (NN)s* with multiple hidden layers and (usually) a large amount of data to learn complex patterns and features from them (Bar et al., 2015; Shin et al., 2016; Yu-Jen Chen et al., 2015). Models are considered to be deep due to their increased amount of layers, and thus also model parameters. They have become popular due to the abundance of data, technological progress in computing power, and also the development of newer techniques. DL models are often said to be inspired by the human brain (Harris et al., 2019; Yu-Jen Chen et al., 2015). DL models are known to excel in image classification tasks and are considered to be the state-of-the-art (Harris et al., 2019; Holzinger et al., 2019; Lakhani and Sundaram, 2017; Yu-Jen Chen et al., 2015).

3.1.1 Basic Concepts of Neural Networks

Neural Network (NN)s are a ML (sometimes also DL) technique inspired by the human brain. The goal with NNs is to recognise patterns by learning the relationships between inputs and outputs through a training process. They consist of a large number of *nodes*, also called *perceptrons*, that

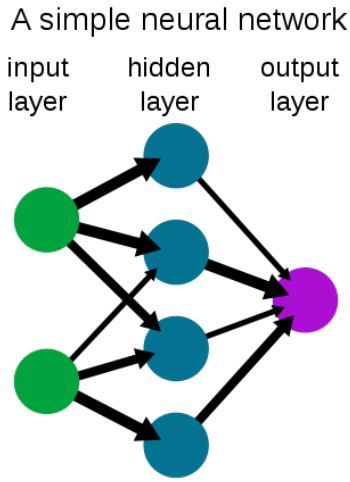


Figure 3.1: EXEMPLARY NEURAL NETWORK. *A very simple Neural Network with one input layer with 2 nodes, one hidden layer (i.e. one fully connected layer) with 4 nodes, and an output layer with 1 node (Wikipedia, 2023).*

are interconnected with each other. NNs are usually divided into *layers* consisting of the nodes. Nodes from a layer take inputs from the previous layer and process them before passing them to the next layer. At the beginning, there is an *input layer* that takes the data into the model, and at the end, there is an *output layer* that provides a result after processing the data from the input layer while it is being passed through the network. The layers in between the input and output layers are called *hidden layers*. If every node of a layer has connections to every node of a previous layer, then that layer is called a *fully connected layer*. The number of hidden layers and nodes in each layer, how the nodes of the layers are connected, and what mathematical operation is applied to the input of each node is customisable and depends on the task (Aggarwal, 2018). Figure 3.1 shows a very simple Neural Network.

Figure 3.2 depicts how a node works in more detail. The inputs from the previous layer are labelled as x_1 to x_N, while their respective weights are labelled w_1 to w_N, with N representing the total number of inputs. It is best to associate the weights with the vectors, and as the input passes through the vector, it gets multiplied by the weight. The node applies a basic linear transformation by calculating the weighted sum of the inputs to get the intermediate result z. Usually, a bias term b is added (not depicted in Figure 3.2), which helps the NN to model more complex relationships:

$$z = \sum_{n=1}^{N} w_n x_n + b \tag{3.1}$$

It would be possible to simply pass the z to the next node without doing any additional calculations, however, the NN would then only be a linear combination of variables. To break up the linearity, and thus to allow the model to learn more complex relationships in the input data, usually an activation function σ is applied to z, resulting in the final output a that gets passed to the next node. There are multiple activation functions, the two popular ones being the Sigmoid and the Rectified Linear Unit (ReLU) activation functions (more on them shortly in Section 3.1.1).

$$a = \sigma(z) \tag{3.2}$$

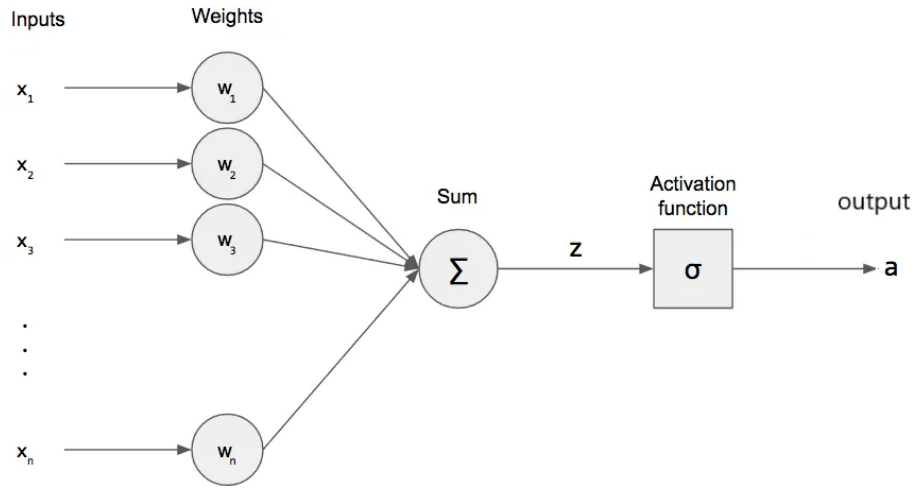


Figure 3.2: EXEMPLARY PERCEPTRON. *A very simple Perceptron, or also simply a Node, that takes inputs x_1 to x_n and calculates the weighted average of them with the respective w_1 to w_n to get z in an intermediate step. An activation function σ is applied to z to get a as the final output of the node (modified version of the figure by Siddharthshah (2020)).*

Each node in the layer passes its output a to the nodes in the next layer, until the node(s) in the output layer is reached and the final estimated output $\hat{y}$ is obtained from the final z.

Training a Neural Network

Training a NN means iteratively adjusting the weights w_1 to w_n and the bias terms b for the nodes in the network to minimise the *loss function*. When the NN is first created, usually these weights and biases are randomly initialised. The loss function is a function that measures how well the actual output y of the input data matches the estimated output $\hat{y}$. The lower the output of the loss function, the better is the network at making predictions. Initially, the NN will have a high loss since the weights and biases are randomly initialised and do not fit the data yet (Aggarwal, 2018).

In the context of binary classification problems, a popular loss function is the Binary Cross-Entropy Loss (BCE) function. If the output label of an input data is either 0 or 1 (e.g. 0 for aTB negative and 1 for aTB positive), the BCE measures how well the output of the network matches the original label.

$$BCE(y, \hat{y}) = -[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})] \tag{3.3}$$

A popular method to iteratively minimise the loss function is called the *gradient descent*. The gradient is a vector of partial derivatives that, in this case, points to the direction of the greatest increase for the loss function. By calculating the gradient of the loss function concerning the weights and biases in the entire NN, it is possible to update the weights and biases iteratively by subtracting the gradient from them. By subtracting the gradient from the weights, they are being adjusted according to the direction of the greatest decrease. The adjusted weights are labelled as $\mathbf{w}_{\text{new}}$ in equation (3.4). The gradients are usually first multiplied by a learning rate η, which

determines how fast the minimum loss is approached. This process is repeated until the loss reaches a minimum.

$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{old}} - \eta \nabla_{\mathbf{w}_{\text{old}}} \text{BCE}(y, \hat{y}) = \mathbf{w}_{\text{old}} - \eta \frac{\partial \text{BCE}(y, \hat{y})}{\partial \mathbf{w}_{\text{old}}} \tag{3.4}$$

To calculate the gradients, first data needs to be input into the model to obtain an estimate on its label. This process of forwarding the input values to generate an output is also called *forward propagation*. The gradients are then calculated layer for layer, from the loss function up to the first layer. Since, through the use of the chain rule in calculus, the calculation of e.g. the second last layer's gradient is based on the gradient of the last layer, which is based on the loss function's output, the calculation of the gradients needs to be done step by step in a backward manner. This process of calculating the gradients for each layer is also referred to as *backward propagation*.

Further, equation (3.4) shows a gradient descent process where the weights are updated after each input data of a dataset. This input data is usually randomly selected, in which case the algorithm is called *stochastic gradient descent*. The opposite, updating the weights by calculating the gradients for every single input data of a dataset, and then averaging them, is called *batched gradient descent*. There are various advantages and disadvantages to both methods. The main one is that the batched gradient descent algorithm is a computationally expensive process. That is why *stochastic gradient descent* can be instead used to approximate the minima. The trade-off however is that the approximation can be less accurate than batched gradient descent. That is why *mini-batch gradient descent* is often used instead, a middle ground between the two. The gradients are instead calculated for a smaller batch of input data that is selected from the dataset (Aggarwal, 2018). B represents the batch size in the following equation:

$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{old}} - \eta \left(\frac{1}{B} \sum_{i=1}^{B} \nabla_{\mathbf{w}_{\text{old}}} \text{BCE}(y^{(i)}, \hat{y}^{(i)}) \right) \tag{3.5}$$

Once every data point in the dataset has been used once to update the gradients, this is called an epoch. Training a NN usually takes multiple epochs. The learning rate, the batch size, and the epoch number are all called hyperparameters. They are additional training parameters that can be adjusted and optimised to achieve better model performance, reduce overfitting (learning the training data too well, including the noise and outliers, such that it performs badly on unseen data), and ensure faster convergence to the minima.

Sometimes, additional techniques like *Batch Normalisation* are used to normalise the outputs (or sometimes also before the activation) of certain layers across a mini-batch. This helps with the vanishing-gradient problem, or also similarly, the *exploding-gradient problem*. This also allows the usage of higher values for the learning rate, increasing the convergence rate to the minimum. Since the batch normalisation adds noise to the outputs, it also helps with overfitting (Ioffe and Szegedy, 2015).

Class Weights Since datasets can often be imbalanced, weights can be assigned to specific classes to counteract the imbalance. Balancing the dataset helps in overcoming the bias towards the majority class and therefore improves the model's generalisability. There are multiple ways of balancing the data. In the ptbench package particularly, which is used in this thesis, this is implemented by calculating the inverse of the class sample count and by providing the weights to the *WeightedRandomSampler*[1] class from the PyTorch package. Below is a summarised step-by-step process on how this exactly works:

[1]For more information, see https://pytorch.org/docs/stable/data.html#torch.utils.data.WeightedRandomSampler

- First, the inverse of the class sample count is calculated for each class label c by using the total class sample count (T_i):

$$v_c = \frac{1}{T_c} \tag{3.6}$$

- The weights v_c are assigned for each sample in the dataset, depending on which class they belong to.
- The list with the sample's weights serves as input for the PyTorch class *WeightedRandomSampler* during training. The sampler uses the weights to determine the probability of selecting each sample while training.

Activation Functions

ReLU is a popular activation function usually used in the nodes of hidden layers, and its purpose is to replace negative outputs with 0. It was shown to perform better in NNs as previously popular activation functions like Sigmoid Glorot et al. (2011). ReLU looks like this:

$$ReLU(x) = \max(0, x) \tag{3.7}$$

ReLU has a multitude of benefits over other activation functions. Its calculation is relatively simple, and it increases the sparsity in the NN due to the 0 values, making training and prediction more efficient. Another advantage over other activation functions like Sigmoid is that it addresses the *vanishing-gradient problem*. When using an activation function like Sigmoid that maps the input to an interval of [0, 1], since in a gradient-based training process the gradients get multiplied with each other when they get back propagated up to the input layers, the gradients can become smaller and smaller until they reach near 0. In especially deeper NNs, this becomes an even bigger issue. When the gradients are near 0, the weights of the network do not get updated and the training process becomes too slow. Since the derivative of ReLU is either 0 or 1, positive gradients cannot vanish anymore (Aggarwal, 2018).

However, this introduces a new problem called the *Dying ReLU* problem, where when a derivative of a node is 0, the weights do not get updated at all. Some other variations of ReLU are sometimes preferred over that standard one like leaky ReLU or *Gaussian Error Linear Units* (GeLU), which try to overcome this problem, but they are not in focus for this work (Aggarwal, 2018).

As briefly mentioned, the Sigmoid activation function is used to re-scale a value to an interval of [0, 1]. This is especially useful when the NN needs to discriminate between two classes (Aggarwal, 2018). For this reason, the Sigmoid function is used in the output layer on the logit z to obtain $\hat{y}$. $\hat{y}$ then represents the probability of the input belonging to either the 0 class or the 1 class:

$$Sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{3.8}$$

3.1.2 Convolutional Neural Networks

Convolutional Neural Network (CNN)s, or also sometimes *Deep Convolutional Neural Networks*, are a DL technique or model, and also a sub-type of NNs consisting of multiple hidden layers (Bar et al., 2015). They perform well in learning the relationships of the values of high(er)-dimensional data like images (Lakhani and Sundaram, 2017), and since they have a large number of parameters and are designed to handle the complexity of high-dimensional inputs, they are usually trained with larger amounts of data compared to traditional feed-forward NNs (Bar et al., 2015; Shin et al.,

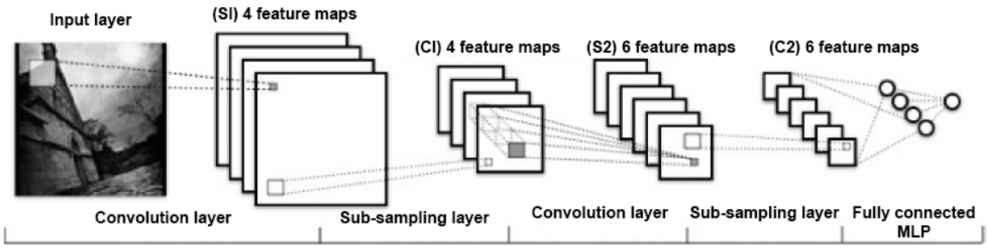


Figure 3.3: EXEMPLARY CONVOLUTIONAL NEURAL NETWORK. *An exemplary Convolutional Neural Network with 2 convolutional layers, each followed by a pooling layer (here called sub-sampling layer) and a fully connected layer in the end before the final output layer (Yu-Jen Chen et al., 2015).*

2016; Yu-Jen Chen et al., 2015). So-called *convolutional layers* are used as hidden layers to perform a *convolution* on the data. This is the process of taking a (small) matrix, called *kernel* or *filter*, and applying an element-wise multiplication on the input data, e.g. a representation of an image, and summing them up to extract features from it. The resulting output matrices are called *feature maps*. The filters themselves consist of the weights, so during training, the parameters that are learned are the weights in the filters. Usually, a convolution layer is followed by other layers like pooling and/or fully connected layers to reduce the dimensionality of the feature maps (Yu-Jen Chen et al., 2015). Fully connected layers are mostly used towards the end of the network. Figure 3.3 depicts a simple exemplary CNN showing these layers. Further, if RGB images are input initially they will have three different channels, meaning three different matrices. The channels are also sometimes referred to as *depth*. The kernel's depth size always depends on the depth of the layer it gets applied to (Aggarwal, 2018). Multiple filters can be applied to one layer to increase the features the network can learn. This further increases the depth of the output of the hidden layers as the image gets fed forward in the CNN, meaning the number of feature maps can be more than the original three channels of the input RGB image.

The convolution operation itself is linear, and there are additional parameters that need to be considered when using them. The *kernel size* defines the dimensions of the filter. The *stride* defines the step size of the kernel as it moves across the input, thus defining the dimension of the feature maps. The filter itself gets applied to the input by sliding the filter across the input from left to right, also sometimes referred to as the *sliding window* algorithm, and the stride defines the step size as the window (i.e. filter) slides. Figure 3.4 depicts this process. *Padding* can be applied to feature maps to control their dimensions by adding 0's around the edges (Aggarwal, 2018).

To calculate the width and height dimensions of the feature maps, equation (3.9) can be used. Assuming the width and height dimensions are the same, it is possible to use the equation to calculate either of the dimensions. O represents the size (width/height) of the feature map, I is the size (width/height) of the input, K is the size (width/height) of the filter, P is the size (width/height) of zero-padding that is added to the edges, and S is the stride or the step size with which the filter is moved across the input:

$$O = \left\lfloor \frac{I - K + 2P}{S} \right\rfloor + 1 \tag{3.9}$$

Usually, after the convolution, ReLU gets applied to each of the values in the feature maps to break up the linearity.

Further, there are different types of pooling layers. Similar to filters, they also have size, stride, and padding parameters, and they get applied to the input in a similar fashion as the kernel, by sliding the window. Unlike filters, there are no parameters to be learned. They are pre-defined mathematical operations that are applied. There are two commonly used pooling operations that

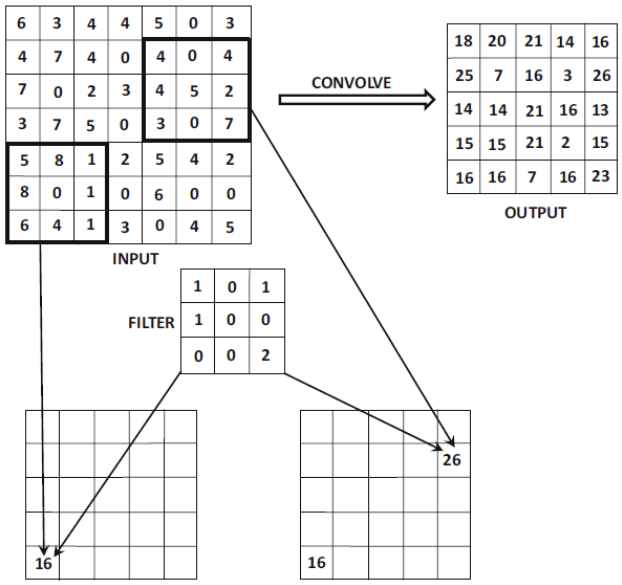


Figure 3.4: EXAMPLE OF FILTER APPLICATION. *An exemplary application of a* $3 \times 3 \times 1$ *filter on a* $7 \times 7 \times 1$ *input matrix through the sliding window algorithm with a stride of 1, and no padding. The output from this operation in the picture is a* $5 \times 5 \times 1$ *feature map (Aggarwal, 2018).*

these layers apply, namely max pooling and average pooling. While the max value gets selected as the output from the values within the window for the max pooling operation, for the average pooling, the average of all the values within the window gets calculated.

The Pasa Model

Pasa et al. (2019) identified the need for a CNN model that is more adapted to the task of TB detection from CXRs. Previous models like *GoogLeNet* and *ResNet* were used for the task of classifying thousands of classes, with the intention of them being trained on millions of images. These models require a lot of memory and computational power to train and predict, and their complexity makes them more prone to overfitting when trained with the small number of publicly available TB-related datasets. With fewer parameters and a smaller architecture, *the Pasa model* is less prone to overfitting with a smaller dataset and less hardware-hungry to train. Figure 3.5 depicts the open-source model's architecture. It consists of 5 *convolutional blocks*, followed by a *Global Average Pooling Layer*, and a *fully connected layer*. A convolutional block consists of two 3×3 convolutions with a stride of 2, followed by ReLU activation function. A 3×3 convolution is applied to the input of the block in parallel before its being summed up with the result of the latter of the two 3×3 convolutional layers. The block ends with a 3×3 max pooling layer with a stride of 2. Batch normalisation is used after each of the 3 convolutions to optimise training and reduce overfitting.

The implementation in ptbench has a single output instead of two, and a Binary-Cross Entropy loss is used during training. This also means that the Sigmoid Function is used instead of Softmax (an activation function typically used for a multiclass classification) to predict the class probability.

A CNN can be constructed from scratch with randomly initialised weights to train it with data like the Pasa Model, but another option is to work with a pre-trained model which comes with weights that have already been trained on a dataset (Shin et al., 2016).

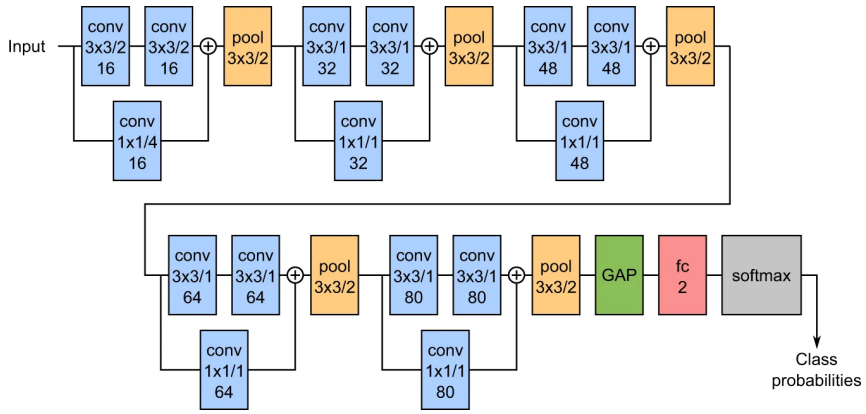


Figure 3.5: THE PASA MODEL. *This figure depicts the architecture of the Pasa Model. It consists of 5 convolutional blocks, followed up by a global average pooling layer, and a fully connected layer at the end. (Pasa et al., 2019).*

3.1.3 Pre-trained Neural Networks

Pre-trained models can be used for *transfer learning*, which is the act of using a model that has already learned the relationships of a general dataset through a pre-training process for another more specific problem. This way, the knowledge of an existing model can be exploited to solve a new problem. For example, the knowledge of a pre-trained model that has learnt the relationships of basic geometric shapes in images can be used for other computer vision-related tasks. One advantage of transfer learning is that it is applicable to problems with otherwise too little data (Bar et al., 2015; Lakhani and Sundaram, 2017; Shin et al., 2016). Other advantages are that such pre-trained models have usually been proven to achieve good performance and that they do not need to be re-trained with a large dataset, which can cost a lot of time and money.

There are generally two ways how pre-trained models can be used for transfer learning (Heo et al., 2019; Shin et al., 2016). With *feature extraction*, the learned representations (of usually latter layers) from the pre-trained model for individual inputs can be used to train another model. With *fine-tuning*, the pre-trained model is modified by (usually) partially *unfreezing* the latter layers to re-train it with new data. Only the unfrozen layer's values (i.e. weights and bias terms) change.

In tasks related to images, earlier layers usually learn the basic geometric shapes and edges that appear in every image, while the latter layers learn the combination of such shapes, thus learning the more complex shapes of more concrete images. That is why the latter layers are usually unfrozen while the earlier ones are kept during a fine-tuning process (Bar et al., 2015; Lakhani and Sundaram, 2017).

As mentioned, some of these pre-trained models are popular due to their well-tested good performance. In the following sections, one of them is explained in more detail: *DenseNet-121*.

DenseNet-121

Dense Convolutional Network (*DenseNet*) is a CNN connecting every feature map from previous layers with succeeding layers in a feed-forward fashion by concatenating the previous feature maps to increase the performance of the network by reducing the vanishing-gradient problem, strengthening feature propagation, enabling feature reuse, and reducing the number of parameters. DenseNet-121 achieves a 6.66% top-5 error rate on the *ImageNet* dataset with 10 crops applied on the test set images, compared to e.g. GoogLeNet's 9.15% on the same number of crops

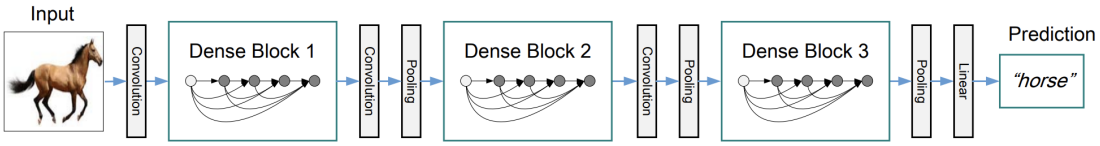


Figure 3.6: DENSENET ARCHITECTURE WITH 3 DENSE BLOCKS. *A DenseNet consisting of 3 dense blocks and 2 transition layers (Huang et al., 2017).*

(Szegedy et al., 2015). ImageNet is a dataset consisting of 14 million images covering more than 20,000 categories. Usually, when referring to ImageNet, it refers rather to its most popular subset called *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC), which consists of over 1.4 million images in total divided into 1000 categories. It is common to test the performance of CNN on this dataset. Each convolution in the DenseNet network is preceded by a batch normalisation and by ReLU. As seen in Figure 3.6, a DenseNet is organised in *dense blocks* and *transition layers*. The dimensionality remains constant within a dense block, and the transition layers reduce the depth (i.e. channels) between the blocks through the 1×1 convolution layer and the general width and height dimensions through the 2×2 average pooling layer. The dense block itself consists of multiple repetitions of a 1×1 convolution layer followed by a 3×3 convolution layer. Depending on the DenseNet variation and the dense block, the number of repetitions changes. *DenseNet-121* specifically consists of 121 layers, starting with a convolutional and average pooling layer, followed by 4 dense blocks and their 3 transition layers in-between, and a final average pooling layer before the fully connected layer and the output layer in the end (Huang et al., 2017).

3.1.4 Direct vs Indirect models

Inspired by Raposo (2021), the models will be distinguished according to the terminology defined in this subsection. The author uses publicly available datasets to train three different types of models (Figure 1.3):

- **Direct Models without pre-training on Radiological Sign labels**: Models with randomly initialised weights that are trained on CXR with PTB
- **Indirect Models with Radiological Sign labels**: Models that consist of 2 parts, first an ImageNet pre-trained DenseNet sub-model that detects radiological signs from CXR by replacing the output layer with a fully connected layer with 14 output nodes for each sign from the NIH CXR14 dataset and pre-training it with the dataset, and a second logistic regression sub-model using the output of the first model to predict PTB on TB-related datasets
- **Direct Models pre-trained with Radiological Sign labels**: ImageNet pre-trained DenseNet sub-model that detects radiological signs from CXR by replacing the final layer with a fully connected layer with 14 output nodes and pre-training it with NIH CXR14 dataset, and then further replacing the multi-class output layer with a randomly initialised output layer for binary classification. The model is then fine-tuned on detecting PTB with various public TB-related datasets

Radiological signs here refer to the radiological abnormalities that are found on the CXR. These signs can be used to detect TB, amongst other diseases. Datasets such as NIH CXR14 consist of CXRs where each X-ray is annotated with various abnormality labels. For each abnormality, either a 0 (abnormality not present) or 1 (abnormality present) is provided. With such a

dataset, it is possible to train a multi-label classification model, meaning it is possible to train a model such as DenseNet-121 to detect the presence of multiple such radiological signs for each CXR.

As mentioned in Section 2.2, the author concludes that an indirect model is naturally more interpretable than the direct model due to the intermediate step of identifying the radiological signs, as this allows us to see what radiological sign labels the model's decision was based on. But the third type of model reached a higher AUC. Raposo (2021) hypothesised that a dataset annotated with radiological sign labels specific to PTB could perform better than the direct model. However, such a dataset was unfortunately not publicly available at the start of this work, so this hypothesis cannot be fully confirmed. Datasets such as NIH CXR14 are not TB-specific, but as Raposo (2021) has shown, this was not necessary. It was still possible to extract useful information from them by constructing the indirect model that can discriminate between *normal* and *PTB*.

Logistic Regression is a statistical model that classifies the input data by outputting the probability of the input data belonging to a (binary) class. Just as with NNs, there are algorithms to optimise the weights of this model to minimise the loss function. The model takes the input data x_1 to x_N and multiplies them similarly with their respective weights (bias term included, too) as in (3.1). To obtain the class probability, the Sigmoid function is used on the result. This is very similar to a single layer fully connected NN, or simply a perceptron as in Figure 3.2, and it can be implemented as such. Raposo (2021) has implemented the logistic regression sub-model from the indirect model as a simple NN with one layer, and it is trained with mini-batch gradient descent to minimise the binary cross-entropy loss.

3.2 Saliency Mapping Techniques

Saliency maps are post-hoc interpretations that explain the outcomes of a model (Lipton, 2018). They can increase the trust of a user in the model as it shows what the model has "observed" when arriving at its output (Pasa et al., 2019). This also means they can highlight where models fail by visually uncovering the biases imposed through the dataset or training procedure (Selvaraju et al., 2017). However, the transparency they provide about the inner workings of a model is considered to be small. They do not necessarily showcase the internal operations of a model, but by highlighting regions in the input data that have most influenced the decision-making process of the model, they are still adding a dimension of interpretability and explainability by presenting certain aspects of the model's behaviour in an understandable way. Through these visualisations that focus on important regions, saliency maps can aid medical experts in diagnosing aTB by striking a balance between interpretability and completeness.

Saliency mapping techniques can differ in how much completeness and usefulness they provide. In light of this, the subsequent subsections delve into detailed explanations of selected techniques.

3.2.1 Grad-CAM

Grad-CAM is one of the earlier techniques that popularised the use of pixel attribution methods for CNNs, invented by Selvaraju et al. (2017). It stands for *Gradient-weighted Class Activation Mapping* (Grad-CAM). It offers a visual explanation by backpropagating the gradient to the last convolutional layer of a model to form a coarse localisation map highlighting the important regions of an input image. What stands out compared to previous techniques is that it can be used for many CNN models without having to make any changes to the model or its architecture, eliminating the interpretability vs. usefulness trade-off that was an issue with the original CAMs tech-

nique (Zhou et al., 2016). The visualisations are class-discriminative for categorical classifications, meaning it is possible to focus the visualisations on a specific target class, due to the tendency of neurons of the last convolutional layer of CNN encoding class-specific information. It can also be used for binary classification with a single output node, in which case the visualisations show the parts of the image contributing to the prediction of the positive class.

This is roughly how Grad-CAM works (Selvaraju et al., 2017):

- Compute the gradient y_c for the class c (before softmax/sigmoid) with respect to the feature map activations A_k for the k feature maps output by the last convolutional layer, e.g. $\frac{\partial \hat{y}_c}{\partial A_k}$. Here, k is the index of a specific feature map (i.e. channel) and ranges from 1 to D, where D is the total number of feature maps produced by the last convolutional layer. The gradients indicate how much influence the individual activations have on the target class score.

- The next step is to calculate how important each of the k feature maps are with respect to target class c. This is done by applying global average pooling to each of the k gradient matrices calculated from the previous step to obtain the weights α_c^k. Here, a feature map has height and width dimensions of $H \times W$, and i and j represent the indices from 1 to H and 1 to W respectively. Z stands for the total number of elements in a feature map A_k (i.e. $H \times W$), and A_k^{ij} represents the activation on the i-th row and j-th column of the k-th feature map:

$$\alpha_c^k = \overbrace{\frac{1}{Z} \sum_{i=1}^{H} \sum_{j=1}^{H}}^{\text{global average pooling}} \underbrace{\frac{\partial \hat{y}_c}{\partial A_k^{ij}}}_{\text{gradients via backprop}} \tag{3.10}$$

- Now, the average of the feature map activations can be calculated by weighing each of the feature maps with its respective weight. On top of that, ReLU is being applied to the final matrix to get rid of negative activations (since this focuses the visualisation on the target class according to Selvaraju et al. (2017)):

$$L^c_{\text{Grad-CAM}} = ReLU \underbrace{\left(\sum_{k=1}^{D} \alpha_c^k A_k \right)}_{\text{linear combination}} \tag{3.11}$$

- Since the feature maps had a lower dimensionality than the original image, the coarse *saliency map* (i.e. heatmap) needs to be upscaled before being overlaid over the original image. Usually, the saliency map is normalised to an interval between 0 and 1 before being upscaled by e.g. *bilinear interpolation*. In addition, a colour map can be applied to the saliency map before being overlaid over the original image.

Amongst various other topics, Selvaraju et al. (2017) also touches on the trade-off between faithfulness and interpretability, and argues that Grad-CAM offers more of both compared to previous visualisation methods. They also show how Grad-CAM visualisations can help to detect biases in the training dataset by visually inspecting failed cases. Figure 3.7 shows how the model they had trained learned to differentiate nurses from doctors by looking at their faces and hairstyles. They were able to fix the bias by adding more images of female doctors and male nurses.

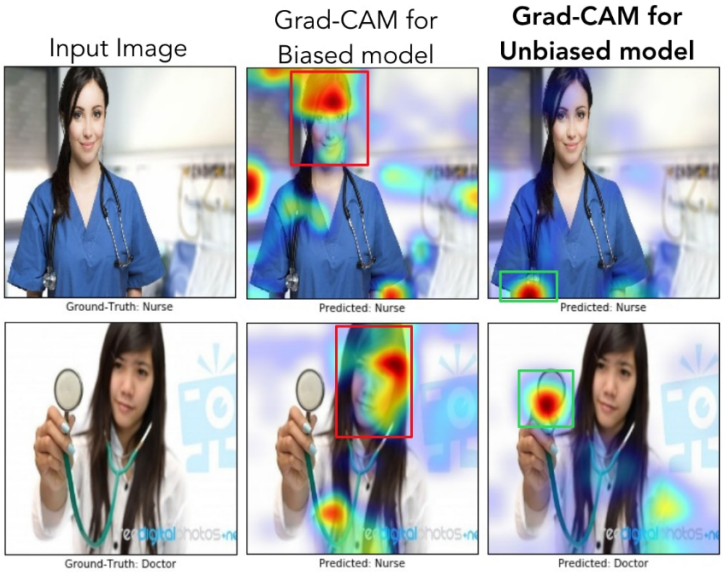


Figure 3.7: BIAS DETECTION. *This figure shows, by utilising Grad-CAMs, the biases learned by a model. It was possible to debias the model by inspecting the biased pictures and adding images of female doctors and male nurses to the training set (figure by Selvaraju et al. (2017)).*

3.2.2 Score-CAM

Wang et al. (2020) raise the concern that gradients might not be the best variable to base the visualisations on, and propose a new gradient-free method called *Score-CAM* that is based on the global contribution of the input features, in contrast to the local contribution of the gradients. The authors argue that gradients are visually noisy due to them being subject to either the vanishing-gradient problem when used with Sigmoid activations in the hidden layers or the ReLU outputting zeroes during back propagation. Further, the authors show that a higher weight α_c^k for one A_k does not necessarily imply that the contribution of that feature map for class c is as important as other weights and that this also might be attributed to either the vanishing-gradient problem or the global average pooling operation on the gradients. The main difference between Score-CAM and Grad-CAM is how the weights α_c^k for the k feature maps are generated. While Grad-CAM uses the global average pooling over the gradients specific to each A_k, Score-CAM calculates the weights through the *increase in confidence* after perturbing the original images with the feature maps and by observing the change in prediction after inputting the perturbed images into the same CNN.

Here is a more detailed description of how Score-CAM works:

- First, similar to Grad-CAM, the feature map activations A_k for the k feature maps output by the last convolutional layer is used. In the first step, they get upscaled and then normalised to a range [0, 1] to obtain the masks M_k. Here, $Up(\cdot)$ refers to the upsampling operation (bilinear interpolation) and $s(\cdot)$ to the normalisation.

$$M_k = s(\text{Up}(A_k)) \tag{3.12}$$

- Then, for each mask k, a perturbed version of the original image G is generated as R_k through element-wise multiplication of the image with the mask:

$$R_k = G \circ M_k \tag{3.13}$$

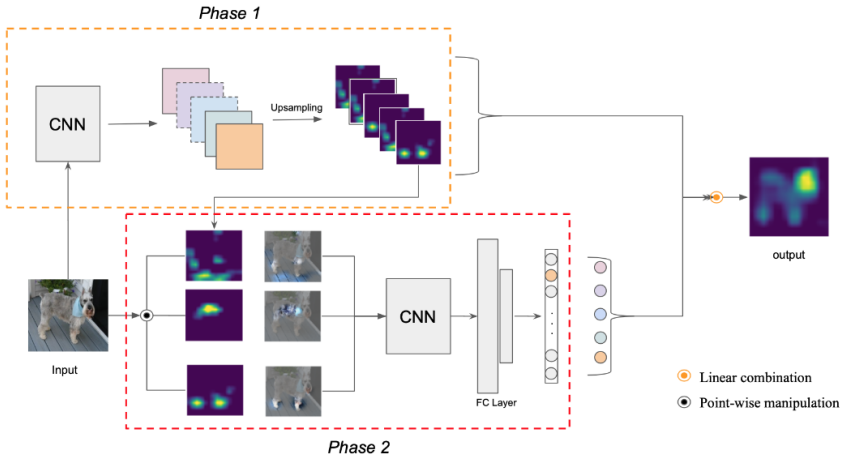


Figure 3.8: SCORE-CAM ALGORITHM. *This figure shows how the visualisations by Score-CAM are generated. The process is divided into two phases. In phase 1, the original image is input in the CNN to get the feature maps. They are then upscaled and normalised. In phase 2, these upscaled masks are element-wise multiplied with the original images, and the resulting perturbed images are input into the CNN again to obtain their predictions, and ultimately scores. The scores are linearly combined with the feature maps from phase 1 to obtain the final saliency map (figure by Wang et al. (2020)).*

- After perturbing the images, they are propagated forwards in the CNN to obtain the output class probability for class c. They are then used to calculate the increase in confidence by comparing the outputs of the perturbed images to the output of the original image. Again, α_c^k stands for the score (i.e. weight) that belongs to the k-th feature map, and it represents the importance of each feature map A_k with respect to target class c. $f(\cdot)$ stands for the forward propagation through the CNN, and $C(\cdot)$ stands for the calculation of the *increase in confidence* score:

$$\alpha_c^k = C(R_k) = f(R_k) - f(G) \tag{3.14}$$

- The weights are then used to calculate the saliency map, similar to Grad-CAM:

$$L^c_{\text{Score-CAM}} = ReLU \underbrace{\left(\sum_{k=1}^{D} \alpha_c^k A_k \right)}_{\text{linear combination}} \tag{3.15}$$

Figure 3.8 also depicts the whole process visually. The authors use various measurement techniques to compare the performance of Score-CAM with Grad-CAM, such as Most Relevant First (MoRF) and Least Relevant First (LeRF) (see Section 3.4.3) and the Proportional Energy score (see Section 3.4.2). They report improved performance for every measurement compared to Grad-CAM and Grad-CAM++.

3.2.3 FullGrad

Similar to Grad-CAM and Score-CAM, *FullGrad* is another saliency mapping technique that aims to calculate importance scores for each feature map (Srinivas and Fleuret, 2019). Unlike Grad-CAM, which only considers the gradient of the last convolutional layers, FullGrad takes into account

the biases and weights throughout the entire CNN to provide a more comprehensive explanation. Specifically, the authors mention the role of saliency maps in explaining the behaviour of the CNN. They differentiate between local and global attribution. *Local attribution* (i.e. weak dependence) refers to the notion of feature importance based on how much changing a particular input feature affects the CNN's output. *Global attribution* (i.e. completeness) refers to the idea that the importance scores for individual input features should collectively explain the CNN's entire output, essentially redistributing the output score back to the individual features. Srinivas and Fleuret (2019) prove mathematically that FullGrad provides both of them.

Here is a simplified version of how FullGrad is calculated:

- First, the gradients are calculated with respect to the inputs. Here, x stands for the input data in the input layer, $f(\cdot)$ stands for the forward propagation through the CNN, and $\nabla_x f(x)$ represents the input-gradients. The input-gradients tell how much of each input feature contributes to the output:

$$\nabla_x f(x) = \frac{\partial f(x)}{\partial x} \tag{3.16}$$

- Consider a single convolutional layer l with filter $z = w * x + b$. The aim is to compute the bias-gradient $f_b^{(l)}(x)$ for that layer. w is the convolutional filter and x is the input to the layer. b is shaped like the input x for that layer, which is achieved by repeating the single scalar b value multiple times. $*$ represents the convolution operation. The gradient of $f(x)$ with respect to z is $\nabla_{z^{(l)}} f(x)$. To compute the bias-gradient for a specific layer, or rather the gradient of $f(x)$ with respect to bias b, the element-wise multiplication of $\nabla_{z^{(l)}} f(x)$ and b needs to be computed:

$$f_b^{(l)}(x) = \nabla_{z^{(l)}} f(x) \circ b^{(l)} \tag{3.17}$$

- To calculate the final FullGrad saliency map $L_{FullGrad}(x)$, it is necessary to weigh the input values x of the neural network with the input-gradients $\nabla_x f(x)$ through element-wise multiplication. Then, for every layer l of all the layers L in the network, and for each channel k of all of the layer's channels D, the bias-gradient $f_b^{(l)}(x)_k$ needs to be summed up. The sum of all bias-gradients needs to be summed up with the element-wise multiplication of the input-gradient and the inputs to get the final full-gradient. To directly obtain $L_{FullGrad}(x)$, post-processing $\psi(\cdot)$ needs to be applied individually as shown below. In Srinivas and Fleuret (2019), this consists of 3 individual post-processing steps: first, the absolute value is taken, then, the gradients are rescaled, and finally bilinear interpolation is used for upsampling. Here is the final equation:

$$L_{FullGrad}(x) = \psi(\nabla_x f(x) \circ x) + \sum_{l \in L} \sum_{k \in D} \psi(f_b^{(l)}(x)_k) \tag{3.18}$$

By incorporating both, the input attributions and the bias attributions, it aims to provide a complete explanation for the network's decisions. This computation, however, is more expensive than other saliency mapping techniques like Grad-CAM or Score-CAM.

3.2.4 Other Saliency Mapping Techniques

This section very briefly introduces the other saliency mapping techniques, although without going into detail about each of them, as the core concepts are all very similar to the previously introduced methods.

Ablation-CAM A gradient-free saliency mapping technique that utilises ablation analysis to determine the weights of the feature maps with respect to the target class (Desai and Ramaswamy, 2020), in an attempt to improve the problems of Grad-CAM (mentioned Section 3.2.2).

Eigen-CAM The principal components of the learned representations within the convolutional layers are being visualised. This method is not class-discriminative, and thus still seems to provide good visualisations without a correct classification by the model (Bany Muhammad and Yeasin, 2021).

EigenGradCAM Similar to Eigen-CAM, but with class discrimination. This is done by taking the first principle component of the element-wise multiplication of the activations and gradients ReLU (Gildenblat, 2023).

GradCAMElementWise Similar to Grad-CAM, but the activations are multiplied element-wise with the gradients before applying ReLU (Gildenblat, 2023).

Grad-CAM++ A variation of Grad-CAM, where instead of taking the global plain average of the gradients of each feature map, the weighted average is taken instead to provide better localisation (Chattopadhay et al., 2018).

HiResCAM Draelos and Carin (2021) aim to increase the faithfulness of the explanations generated by saliency mapping techniques. Instead of averaging the gradients like in Grad-CAM, they are instead element-wise multiplied with the feature maps.

LayerCAM The positive gradients are input into ReLU and used as weights for the activations. As with the other methods, this method, too, attempts to improve the performance of Grad-CAM Jiang et al. (2021).

RandomCAM This is a toy saliency mapping technique implemented in Gildenblat (2023) that creates a saliency map filled with random uniform values in the interval of [-1, 1]. This is used as a baseline, any method performing worse than this baseline is not good.

XGrad-CAM Fu et al. (2020) argue that this method enhances Grad-CAM and is comparatively easier to implement than Grad-CAM++ and Ablation-CAM. The weights for the feature maps are calculated by a normalised weighted combination of the gradients and activations.

3.3 Tools and Frameworks

This section briefly introduces the libraries that were used for Python to train and evaluate the models and the saliency mapping techniques.

3.3.1 ptbench

ptbench is a library specifically for training pre-configured models for aTB detection on CXRs. It provides a fully reproducible experiment cycle, meaning everything from the training up to the evaluation can be done with very few commands. For that, it uses configurations for publicly

available datasets. The library relies mainly on PyTorch to train the models, while the evaluation methods are custom.

The open-source code and the documentation are both available on the GitLab of the Idiap Research Institute.[2]

The *pytorch-grad-cam* package has been integrated into *ptbench* package to produce visualisations with saliency mapping techniques and to evaluate them.

3.3.2 pytorch-grad-cam

pytorch-grad-cam is a library that standardises the application of saliency mapping techniques on CNN models that have been trained with PyTorch. It supports various saliency mapping techniques, including Grad-CAM, Score-CAM, and FullGrad, and thus allows the user to add visual explanations to the models. It also provides evaluation methods like RemOve and Debias (ROAD) to measure the performance of the saliency mapping techniques.

The code and documentation are open-source and they can be found on GitHub.[3]

3.4 Measurements & Metrics for Evaluation

In this section, different established measurement techniques and metrics are introduced. They are necessary to quantify, assess and evaluate the performance of the models, and they not only help to identify improvement points but also allow for comparison of the different models to each other. For each technique, an explanation of how they work and why they were chosen is given.

For RQ1, Area Under the Receiver Operating Characteristic (AUROC) Curve (or also AUC) is the main metric used to assess the performance of the models, and the F1-Score is not only used to select the optimum evaluation threshold for the models before measuring the sensitivity and specificity but also to provide an alternative performance score. See Section 5 for more details on why and how they are used.

For RQ2, ROAD and *Proportional Energy* are the main metrics used to assess the performance of the saliency mapping techniques. See Section 5 for more details on why and how they are used.

3.4.1 Measurements & Metrics for Model Prediction

Confusion Matrix

A confusion matrix visually aids in understanding the metrics used in this section. Table 3.1 shows such a typical confusion matrix with binary labels. The model first generates an output score, which is then translated into a binary decision (positive or negative) based on a selected threshold. The matrix represents the four possible outcomes when comparing the model's binary predictions to the actual labels. The positive and negative classes could for example represent the presence (positive) or absence (negative) of aTB in the patients. The 4 cases are explained in more detail below (Bradley, 1997):

- **True Positive (TP)**: The model predicts positive, and the actual case was positive
- **False Positive (FP)**: The model predicts positive, but the actual case was negative. The model has made an error.

[2] https://www.idiap.ch/software/biosignal/docs/biosignal/software/ptbench/main/sphinx/install.html
[3] https://github.com/jacobgil/pytorch-grad-cam

- **False Negative (FN)**: The model predicts negative, but the actual case was positive. The model has made an error.
- **True Negative (TN)**: The model predicts negative, and the actual case was negative

	Actual Positive	**Actual Negative**
Predicted Positive	True Positive	False Positive
Predicted Negative	False Negative	True Negative

Table 3.1: Confusion matrix for binary classification

Accuracy

Accuracy is a common metric in classification tasks. It quantifies the proportion of instances that are classified correctly out of the total number of instances. It can be calculated by dividing the number of correctly classified predictions (True Positive (TP) and True Negative (TN)) by all predictions (Bradley, 1997):

$$Accuracy = \frac{Nr.\ of\ all\ correct\ predictions}{Nr.\ of\ all\ predictions} = \frac{TP + TN}{TP + FP + FN + TN} \tag{3.19}$$

While it is a straightforward and easy-to-understand metric, it can be problematic for imbalanced datasets, as the prediction performance on the majority class will influence the accuracy the most.

Sensitivity & Specificity

Sensitivity (also known as *recall* or *True Positive Rate*) and *Specificity* (also known as *True Negative Rate*) are two complementary and widely used metrics in the medical and the AI domain (Bradley, 1997). Sensitivity answers the question *"How many of the actual positive cases were correctly classified as such?"*. Specificity answers the question *"How many of the actual negative cases were correctly classified as such?"*

$$Sensitivity = \frac{Nr.\ of\ all\ correct\ positive\ predictions}{Nr.\ of\ all\ actual\ positive\ cases} = \frac{TP}{TP + FN} \tag{3.20}$$

$$Specificity = \frac{Nr.\ of\ all\ correct\ negative\ predictions}{Nr.\ of\ all\ actual\ negative\ cases} = \frac{TN}{TN + FP} \tag{3.21}$$

These metrics are not impacted by imbalanced data as in accuracy, and it is possible to assess how the model performs in predicting each class label.

F1-Score

The F1-Score is the harmonic mean of precision and recall, providing a single metric that balances the trade-off between these two measures. Recall (or Sensitivity) has been defined already, and below is also a definition for precision. It answers the question *"How many of the cases predicted*

as positive were actual positive cases?". It is generally a popular metric that is used in the medical domain (Taha and Hanbury, 2015).

$$Precision = \frac{Nr.\ of\ all\ correct\ positive\ predictions}{Nr.\ of\ all\ positive\ predicitions} = \frac{TP}{TP + FP} \tag{3.22}$$

The F1-Score is defined as:

$$F1 - Score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{3.23}$$

Since it takes account of both false positives and false negatives, the metric can be used for imbalanced datasets. If either of these values performs badly, the F1-Score will be affected. Another advantage is that it is only a single value to account for while comparing and optimising models. However, if sensitivity is preferable over specificity or vice versa, as it often can be in the medical domain depending on the use case (World Health Organization, 2021), the F1-Score might not be suitable as a single measure. The F1-Score is further influenced by the chosen threshold for a model during evaluation. That is why AUROC is also introduced in the next section to obtain a more holistic and theoretical understanding of a model's ability to discriminate the classes.

Area Under the Receiver Operating Characteristic Curve

Compared to the F1-Score, the Area Under the Receiver Operating Characteristic (AUROC) Curve is a threshold-independent metric, because it measures the performance of a model across various classification thresholds. It is another widely used metric in binary classification tasks (Bradley, 1997).

AUROC is a metric for showing the separability of two classes by plotting the True Positive Rate (TPR) (Sensitivity) against the False Positive Rate (FPR) (*"How many of the actual negative cases were wrongly classified as positive?"*).

$$FPR = \frac{Nr.\ of\ all\ wrong\ positive\ predictions}{Nr.\ of\ all\ actual\ negative\ cases} = \frac{FP}{FP + TN} = 1 - Specificity \tag{3.24}$$

To get the AUROC, first, the ROC curve needs to be defined. This is done by calculating the sensitivity and FPR for different thresholds for the measurements in Section 3.4.1. Then, the Area under the plotted Curve is calculated through an integral. This will lead to a value between 0 and 1, which can be considered as the summary of the performance of a model across different thresholds. Figure 3.9 depicts an exemplary AUROC Curve. A classifier that can perfectly separate the two classes will have an AUROC of 1. A classifier whose predictions seem to resemble a random 50/50 coin toss will have an AUROC of 0.5 and its ROC curve will be diagonal, as depicted in the figure.

For the thesis, this metric is sufficient, since it is commonly used and thus allows for easy comparability between different models in the field.

3.4.2 Metrics for Model & Visualisation Performance

IoU and IoDA

Intersection over Union, and *Intersection over Detected Area* are two metrics that can be used to measure the localisation performance of saliency mapping techniques. It is a simple metric that is popular in the medical and AI field (Selvaraju et al., 2017; Wang et al., 2017; Zhou et al., 2016).

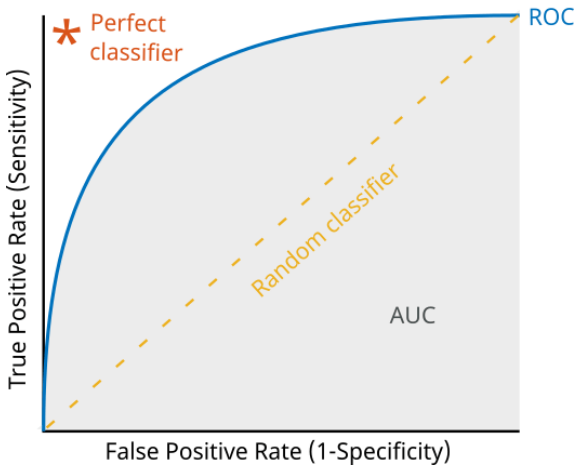


Figure 3.9: AUROC CURVE. *This figure depicts an exemplary AUROC. It also shows where the ROC of a perfect and a random classifier would be located (figure by The MathWorks (2023)).*

Intersection over Union (IoU) can be calculated by dividing the intersection of the ground truth bounding box area with the bounding box of the detected area by their union:

$$IoU = \frac{\text{Area}_{GroundTruthBbox} \cap \text{Area}_{DetectedBbox}}{\text{Area}_{GroundTruthBbox} \cup \text{Area}_{DetectedBbox}} \tag{3.25}$$

Similarly, this is the formula for Intersection over Detected Area (IoDA):

$$IoDA = \frac{\text{Area}_{GroundTruthBbox} \cap \text{Area}_{DetectedBbox}}{\text{Area}_{DetectedBbox}} \tag{3.26}$$

The way the bounding box around the detected area is being generated is how it is being described in the original CAMs publication by Zhou et al. (2016). The pixel values above 20% of the max value are kept in the saliency map. From the remaining pixels, the largest connected component is calculated, and a bounding box is drawn around it. Since Zhou et al. (2016) does not specify which algorithm was used for the largest component calculation and bounding box generation, the default algorithms within the *OpenCV* python library (OpenCV, 2023) are being used. Specifically, for the largest component calculation, *connectedComponentsWithStat* is being used with the default 8-way Spaghetti algorithm (Bolelli et al., 2020). The *boundingRect* function is being used from the same library to draw the bounding box.

Since images can have multiple ground truth bounding boxes, the bounding box that achieves the highest intersection with the detected area bounding box can be used to calculate the final score for each image sample. An alternative is to generate multiple detected bounding boxes, order them by the size of the largest connected components and compare them the the multiple ground truth bounding boxes.

These two metrics can also be used together with a threshold to obtain a binarised metric. If the IoU or IoDA are above a certain threshold, the localisation counts as successful and vice versa. A confusion matrix can be created this way and other metrics can be calculated, such as accuracy, according to that (Wang et al., 2017).

The advantage of these two metrics is that they are easy to understand, commonly used in the field, and have practical relevance, meaning a high score indicates that the correct parts of an image are being highlighted according to the ground truth provided by radiologists. However, the disadvantages outweigh the advantages by far. The introduction of extra calculation steps by

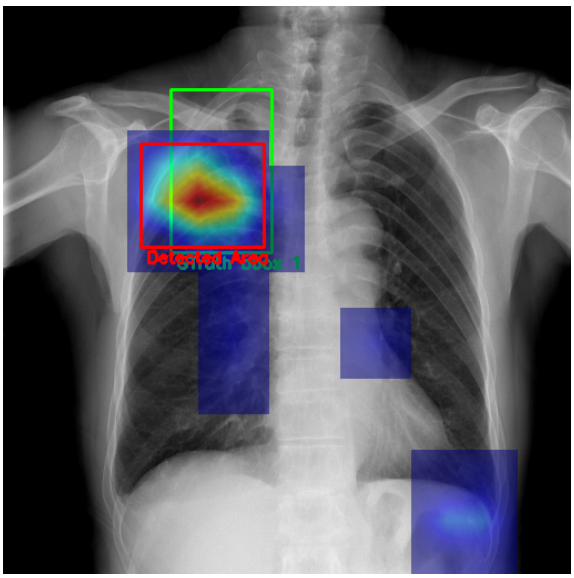


Figure 3.10: DETECTED AREA FOR IOU/IODA. *This image shows the visualisation of saliency maps generated by Grad-CAM. The CXR also includes bounding boxes for the ground truth area of the radiological sign (green) and the bounding box for the detected area (red) by choosing the largest connected activated area*

deliberately choosing a threshold value for the saliency map, by selecting an algorithm for largest component detection, and drawing a rectangular box around it that encompasses pixels that are not relevant, make these metrics more imprecise as they may seem. Another disadvantage, or rather feature, of these metrics is that they not only measure the performance of the model in an isolated way, they also inevitably indirectly get influenced by the model performance. This makes it impossible to distinguish between errors from the visualisation technique and the errors made by the model. This feature further undermines the practical relevance, since in the end, the overall performance of the CAD system is what matters to a medical expert.

The introduction of artificial noise seems fixable by avoiding doing the calculation steps. The next metric solves this problem.

Proportional Energy

Proportional Energy[4] is a localisation metric that is used in Wang et al. (2020) to measure the performance of saliency maps. This is done by summing up the activations of the saliency map that lie within the ground truth bounding box and dividing that by the sum of all activations within the saliency map. This results in a metric that represents the proportion of activations that are "correct":

$$ProportionalEnergy = \frac{\sum_{i,j}(L^c_{i,j} \cdot M^c_{i,j})}{\sum_{i,j} L^c_{i,j}} \tag{3.27}$$

M^c is a mask where the values within the ground truth bounding box equal 1, and the values outside equal 0. This mask is then element-wise multiplied with the saliency map L^c to get the

[4] Wang et al. (2020) call it the *energy-based pointing game*, but since this name does not provide clues on what the metric does, Proportional Energy has deemed to be a more descriptive name for it for this thesis.

numerator. The denominator is simply the sum of all activations within the saliency map.

While IoU and IoDA were implemented to only consider the highest intersecting ground truth box for the metric calculation, the implementation of Proportional Energy in *ptbench* will consider all ground truth bounding boxes when calculating the metric, as the notion of a singular largest component as the detected area gets dropped with this metric.

Contrary to IoU and IoDA, this metric does not introduce any artificial noise to the detected/highlighted area and is more faithful to the result of the saliency mapping technique while keeping the practical relevance advantage. It is also still relatively simple to understand. However, the "property" of the inseparability of the performance of the model and saliency mapping technique remains. Further, this metric in a sense, measures how well the performance of the models and techniques is according to the ground truth provided by radiologists. It assesses the performance of the model to human standards, but even an expert radiologist's ability to recognise patterns has an upper limit due to the limitations of the human body. Thus, if the models and techniques use patterns that a human cannot detect or comprehend, and decides to base its decisions on them, they would inevitably score lower on the Proportional Energy, even if the patterns were correct.

The next metric aims to isolate the performance of the saliency mapping technique from the underlying model, and since it does not use ground truth labels drawn by radiologists, the next metric, while it could measure the influence of radiologists to a certain degree, is not bound by it.

3.4.3 Metrics for Visualisation Method Performance

ROAD

RemOve And Debias (ROAD) is a metric based on pixel perturbations to measure the performance of attribution methods (Rong et al., 2022) specifically, without other external influences on the metric. It is comparatively computationally inexpensive to calculate compared to other sophisticated methods that attempt the same, like RemOve and Retrain (ROAR) or with GANs.

Perturbation methods are not new and have been used previously to measure the performance of attribution methods. Specifically, Chattopadhay et al. (2018) have used this method to evaluate and compare Grad-CAM++ to Grad-CAM. There are multiple approaches in perturbed images. Two popular ones are Most-Relevant-First (MoRF) and Least-Relevant-First (LeRF). Taking MoRF as an example, in this method, a certain percentage of the most relevant pixels according to the saliency map gets removed (pixel value set to 0), and the perturbed new image gets input into the model again to see how the classification of the image has changed by the model. If the confidence of the model dropped, then the original saliency map did a good job of highlighting the most important regions, since after removing these regions, the model was not able to properly identify the ground truth class of the perturbed image. Vice versa, in LeRF, one would expect either no confidence change or an increase in confidence when unimportant parts of an image have been removed. This drop or increase in confidence can be calculated in various ways, but usually, the model's outputs for the original and perturbed image are taken, and their difference is calculated. Figure 3.11 shows an example of how an image might look after a typical LeRF perturbation.

In a sense, MoRF and LeRF measure how well the explanation reflects the prediction of the underlying model, providing fidelity to the visual explanations. However, it is often the case that MoRF and LeRF do not agree with each other. The exact reasons for this are currently unknown, and one strategy might work better for a specific model/saliency mapping technique combination than the other (Rong et al., 2022; Srinivas and Fleuret, 2019; Tomsett et al., 2020). Additionally, some methods may require retraining steps (e.g. Generative Adversarial Networks (GANs), or ROAR by Hooker et al. (2019)), which is computationally expensive. Hooker et al. (2019) show that MoRF and LeRF introduce artefacts, and distribution shifts in the data. Changing the pixels

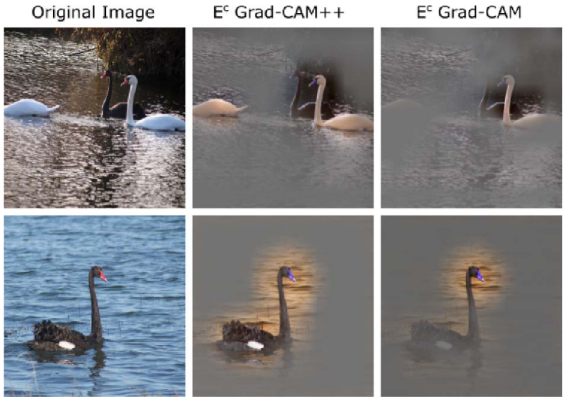


Figure 3.11: LeRF Example. *This figure depicts two different images and how their perturbations look like by multiplying the images with the saliency maps generated by Grad-CAM and Grad-CAM++ after removing the least relevant pixels in them (figure by Chattopadhay et al. (2018)).*

(a) Original (b) Fixed Imp. (c) Noisy Lin. Imp.

Figure 3.12: Noisy Linear Imputation Example. *This figure depicts the original image in (a), a fixes imputation method on 50% of the pixels in (b), and Linear Noisy Imputation on 50% of the pixels in (c) (figure by Rong et al. (2022)).*

of an image to a specific colour like in Figure 3.11 inevitably leaks information - if the confidence of a model drops or increases with such an example, did it happen because the saliency mapping technique was performing well, or because the model has not seen unnatural data like that before? Remove and Retrain by Hooker et al. (2019) tried to fix these shifts by retraining the models with the unnatural data, but this is a very expensive process, as training the models introduced in this work can take up to almost a week with modern GPUs. Aside from that, Rong et al. (2022) also argues how ROAR leaks data since retraining a model with perturbed images could lead to a model that discriminates e.g. from two classes just by looking at the location of an object (i.e. from the locations that were perturbed).

ROAD aims to solve these problems while making the results consistent and the process efficient by reducing the said distribution shift in unnaturally perturbed images. Instead of unnaturally masking the pixels, Rong et al. (2022) suggest using a novel imputation method called *Noisy Linear Imputation*. In short, instead of removing the pixel, the pixel value changes by the value of its direct and diagonal neighbours, and some added noise. Figure 3.12 shows how the noisy linear imputation looks compared to a fixed, unnatural imputation. Since the pixel values depend on each other and calculating these values sequentially would lead to unforeseen biases or inaccuracies, a sparse and linear equation system is formed to calculate the imputations. Calculating this linear equation system is more efficient than using GANs or ROAR.

The ROAD score calculation has been implemented in *pytorch-grad-cam*. The library provides not only LeRF and MoRF, but also their combined average as an additional metric.

The idea of perturbing the images, and reducing class information leakage both lead to a metric that aims to minimise the influence of confounders and the underlying model when measuring the saliency mapping technique performance. However, Rong et al. (2022) never claim that this method perfectly separates the influence of the model from the performance of the saliency mapping technique. The authors are claiming that the information leakage from the imputed image to the model can never be 0, but it can be reduced thanks to their method. This metric has less practical relevance for medical experts and is more a metric intended to measure the performance of a saliency mapping technique as well as possible. To be more precise, since this metric does not use human-generated ground truth labels to assess the performance, it is not bound by the human's ability to recognise patterns. Visualisations that score high on ROAD are expected to reveal correctly what the model is focusing on. Thus, this metric should be effective at revealing which features the model focuses on, but unlike Proportional Energy, it cannot discern whether these features are genuinely useful (unbound by human limitations) for the task at hand and/or indicative of biases in the model.

It should be noted that GANs potentially perform better in imputing images since their imputations are more sophisticated, as mentioned by Rong et al. (2022).

AOPC

Area Over the Perturbation Curve (AOPC) is another slight extension to the existing perturbation methods (Tomsett et al., 2020). Instead of perturbing an image by a set amount of pixels and then calculating the change in confidence, the image is perturbed by multiple set amounts of pixels. So to calculate the AOPC for a single image, first forward the original image within the model to get the prediction, then perturb the image e.g. 4 times by perturbing 20%, 40%, 60%, and 80% of the pixels, input the 4 images into the model to get an output for each perturbed image, calculate the change in confidence for each of them with the help of the output for the original image, and take their mean. As the amount of pixels that get perturbed changes, the change in confidence is expected to grow. AOPC tries to capture that change by calculating the mean through the different perturbed pixel amounts.

In *ptbench*, it is possible to select multiple percentiles when calculating the $AOPC_{MoRF}$, $AOPC_{LeRF}$, and $AOPC_{Combined}$ with ROAD. Here, $AOPC_{Combined}$ is the average of $-AOPC_{MoRF}$ and $AOPC_{LeRF}$.

Calculating the AOPC allows to avoid arbitrariness when choosing which amount of pixels to remove, and it enhances the fidelity of the metric by allowing to capture the change in confidence over different percentiles. There is a fidelity vs. computational cost trade-off dictated by the number of percentiles that are being used. While using more percentiles provides a finer granularity in tracking confidence changes, it also increases the computational cost. This is because, for each percentile, the image must be perturbed and then processed by the model.

3.5 Explainability

3.5.1 Basic Concepts and Definitions

There is a distinction to be made between *interpretability* and *explainability*, and multiple definitions for both of them can be found throughout the literature. There is no one agreed-upon definition, on the contrary, it is agreed upon that there is *no* one definition for interpretability or explainability (Guidotti et al., 2018; Lipton, 2018). This section aims to give an overview of some of the definitions in the context of supervised learning and to identify common important re-occurring concepts that are shaping these definitions.

Generally, interpretability and explainability are considered to be necessary to have accountability, create trust between the user and the model, and create fairness amongst those subjected to its use. They are also instrumental in identifying biases, discrimination, security issues and errors, (Castelvecchi, 2016; Gilpin et al., 2018; Guidotti et al., 2018; Lipton, 2018). In many instances, it can increase the usefulness of the model (Doi, 2007; Li et al., 2004; Rajpurkar et al., 2020).

Lipton (2018) states that interpretability has been understood by some as a means to "engender" *trust* in a model and/or its characteristics like parameters, features or algorithms, with one potential definition of trust being *confident that a model will perform well concerning real-world objectives and scenarios*. Trust can also be defined subjectively, as when a person feels more at ease with a well-understood model. The authors continue by mentioning different desiderata for interpretability: *trust*, *causality*, *transferability*, *informativeness*, and *fair and ethical decision-making*. Interpretability can be used to infer properties or generate hypotheses about the natural world, for example, *inferring causal relationships* from observational data, which is important for fields like medical research. Further, it is desirable to have an interpretation that is *informative/useful*. It is also important to take into account the *transferability* of a model, which is its ability to generalise to unfamiliar situations and in non-stationary environments, and its impact on the environment itself. Interpretability is also a means to ensure that the model is unbiased and fair. Further, Lipton (2018) identifies two groups of properties and techniques that can be used to build interpretable models, namely *transparency* and *post-hoc explanations*. Transparency describes the inner workings of a model. Transparency can be achieved in three levels: *simultability* (the whole model), *decomposability* (for its components), and *algorithmic transparency*. Transparency on the simultability level can be achieved if a human can take the model and some inputs to follow the calculations in a reasonable time. However, what is "reasonable" is highly subjective, such that Lipton (2018) suggests that even simple models are not intrinsically transparent or interpretable. Decomposability refers to the capacity of a model to have each of its components (inputs, parameters, calculations) separately analysed and offer an intuitive explanation. Algorithmic transparency refers to the comprehensibility of a model's learning algorithm. Post-hoc interpretations on the other hand try to explain the results of a model without necessarily opening up the black box, similar to human brains. Textual, visual, local explanations, and *explanations by example* fall into this category. Post-hoc interpretations and transparency are thus contradicting to a certain degree but it does not mean that a post-hoc model cannot also provide some transparency about the inner workings at the same time. Lipton (2018) concludes that:

- linear models are not necessarily more interpretable than DL models, since they have a trade-off between simultability, decomposability and algorithm transparency. Additionally, post-hoc visualisations in DL models could potentially be considered more interpretable.
- How interpretability is intended to be achieved needs to be clearly defined and measured
- Usefulness should not be given up for completeness/transparency
- Post-hoc interpretations can be misleading

Tomsett et al. (2018) adopt the definitions given by Lipton (2018) and categorise post-hoc interpretations by Lipton (2018) as *explanations*. Tomsett et al. (2018) define *explainability* as "the level to which a system can provide clarification for the cause of its decisions/outputs". *Interpretability* is seen as a broader term encompassing transparency and explanations/explainability and how a human can make use of this information.

Guidotti et al. (2018) defines interpretability as "the ability to explain or to provide the meaning in understandable terms to a human". According to the authors, this is implicitly expressing that such interpretations do not need further explanations to be understood by humans. Interpretability is only necessary for something that influences a human decision. Similar to Lipton

(2018), an interpretation can be of 2 categories, it should either open up a black box (increase transparency) or it should explain its output. 3 dimensions were identified:

- *local vs. global interpretability*, similar to the transparency levels in Lipton (2018)
- *time limitation*, depending on if there is a time constraint on the use-case
- *Nature of User Expertise*, background knowledge of the user of the model

Further, a list of desiderata for interpretable models is provided by Guidotti et al. (2018): *interpretability* in terms of *measuring* it and *complexity*, *accuracy*, *fidelity* in terms of how accurately an interpretation can imitate a black box model. Fairness, privacy, usability, reliability, robustness, causality, scalability and generality are also briefly mentioned. Contrary to Lipton (2018), Guidotti et al. (2018) argues that linear models, decision trees, and rules are recognised to be interpretable to humans.

According to Gilpin et al. (2018), interpretability is "the science of comprehending what a model did (or might have done)". Visual cues like heatmaps or contours belong to this category. The authors further state that interpretability alone is not enough, and that explainability is needed for humans to trust black box models:

> [...] models that are able to summarize the reasons for neural network behavior, gain the trust of users, or produce insights about the causes of their decisions. While interp[r]etability is a substantial first step, these mechanisms need to also be complete, with the capacity to defend their actions, provide relevant responses to questions, and be audited.

Explanations thus have 2 components based on which they can be evaluated: *interpretability* and *completeness*, which are difficult to achieve simultaneously, as there is a trade-off between them. Interpretability is about describing the internal workings of a model in an understandable way to humans. Completeness is about describing the workings of a model in an accurate way (Gilpin et al., 2018). Completeness can always be achieved by revealing all mathematical operations and parameters. An accurate and complete mathematical explanation of a model might not be easily interpretable, and it is easy to mislead users into believing that an explanation is simple and complete. Further, explanations can explain the *processing* of the data — answering why a particular input leads to a particular output — or the *representation* of the data inside a DL model. A third approach is to develop *explanation-producing* systems, which are specifically architected to help with interpreting their behaviour, either by revealing key aspects of their internal logic or by generating human-understandable justifications for their operations.

Although there are no agreed-upon definitions for interpretability and explainability, there is still a need to have one (possibly combined) definition for them (Lipton, 2018) and the concepts related to them to have a clear understanding of what these terms mean for the remainder of the thesis. From the presented works in this section that attempted to define interpretability and explainability, it seems unavoidable to have an overlap between the two definitions, and it is often the case that one term encompasses the other. The following definitions aim to provide definitions for the *most important* key concepts mentioned in this section that are relevant to this thesis, based on the mentioned works:

Interpretability is the ability to explain or to present the meaning of a model, including its parameters, features, and algorithms, in understandable terms to a human (Doshi-Velez and Kim, 2017; Gilpin et al., 2018; Guidotti et al., 2018; Lipton, 2018). Interpretability can be achieved through either transparency or post-hoc explanations. An interpretation increases the understanding about *how the model works* (Lipton, 2018).

Transparency is the degree to which the inner workings of a model can be understood and followed by a human. It focuses on making the model's internal operations clear and traceable. Transparency can be achieved in three levels: simultability (ability to follow the model's entire logic), decomposability (understanding the function and contribution of individual components), and algorithmic transparency (clarity about the learning algorithm's mechanism) (Lipton, 2018). Transparency serves as one of the routes to achieving interpretability but is not solely sufficient to facilitate human understanding.

Completeness can be partly achieved by transparency. It refers to how accurately every aspect of a model's inner workings is described. There is a trade-off between interpretability and completeness (Gilpin et al., 2018).

Post-hoc Interpretation is a method of providing an explanation for the output of a model. It does not need to be transparent, meaning it does not need to reveal the inner workings of a model (Lipton, 2018).

Trust is the confidence that a model will perform well with respect to real-world objectives and scenarios. Trust can be defined subjectively, as when a person feels more at ease with a well-understood model. Well-interpretable models increase the trust in them (Lipton, 2018).

Usefulness is the ability of a model to provide useful information to its user. This is also related to the accuracy of a model. Depending on how well a model performs on the chosen accuracy measured, the model might be more or less useful (Lipton, 2018; Guidotti et al., 2018).

Explainability refers to the ability of a model or algorithm to provide transparency, accountability and insights into its decision-making process. It is the capacity of a model to be able to summarise the reasons for its behaviour, gain the trust of users, or produce insights about the causes of its decisions. An explanation balances 2 concurring goals: interpretability and completeness. An explainable model is interpretable, but the reverse is not always true. An explanation increases the understanding about *why the model works* (Gilpin et al., 2018). Explainability with Deep Learning models can be achieved by providing an explanation for how the data is *processed* or how the data is *represented* within the model, or by having an *explanation-producing* model.

There are no standardised methods or blueprints to achieve explainability within AI's (Guidotti et al., 2018), and achieving explainability usually depends on the requirements of the applied domain and its users (Tomsett et al., 2018).

Chapter 4

Data

It is necessary to use datasets consisting of CXR of patients with and without aTB to train the DL models. In general, the availability of such public datasets related to TB is limited (Jaeger et al., 2014; Liu et al., 2020). In this chapter, a multitude of publicly available TB-related datasets is being introduced. For each dataset, a general description, the data split ratios, and their use case in the context of this thesis are provided.

Following dataset introductions in Section 4.1, Section 4.2 describes various (pre-)processing techniques that have been applied to the datasets throughout the experiments.

4.1 Datasets

Table 4.1 is a summary of all the dataset splits that were used to train the models for this work. The TBX11K dataset has two different, while all the other datasets have only one. It is important to notice that for some datasets the labels for the positive and negative classes are labelled differently for each dataset. The authors sometimes do not provide exact terms to define them. The sub-tables use the same labelling as the authors of the datasets have used, however, for this work, all cases belonging to the negative class for all datasets, except the TBX11K dataset, are interpreted as *healthy* and the cases belonging to the positive class are interpreted as aTB.

4.1.1 TBX11K

Description This dataset has been compiled and made public by Liu et al. (2020). It consists of 11'702 CXRs as 24-bit RGB PNG files that have been resized to 512x512 from the original resolution of 3000x3000, which makes this dataset the largest publicly available TB-related dataset. Each sample is from a unique individual. Moreover, the dataset was specifically compiled to use it for CNNs. There are 4 types of labels for the CXRs, *healthy*, *sick & non-TB*, *active TB*, and *latent TB* (LTBI). There is also one additional implicit label in the case a CXR exhibits *active & latent TB* signs simultaneously. Sex and age information for each person is also available.

The dataset is pre-split into a training, validation, and test set according to Table 4.2. No labels have been provided for the test set due to an ongoing online competition. Additionally, cases that could not be identified (*uncertain* cases) have also been put into the test set. While the authors report 104 *latent TB* cases, only 103 were annotated as such, leaving one TB-positive case unannotated. The authors mentioned 2800 test cases, however, the published dataset contains 3302.

For all the aTB or latent TB regions that have been detected in TB-positive cases within the training and validation sets, a bounding box is provided. One image can have multiple bounding

Table 4.1: This table shows a summary of all the dataset splits based on which the models were trained.

(a) TBX11K Custom Split 1 - The publicly available TBX11K dataset split randomly and stratified for the purpose of this thesis. The number of images in each set of each label is listed below.

Label	Training	Validation	Test	**Total**
Healthy	2390	610	800	**3800**
Active TB	377	96	157	**630**
Total	2767 (62.5%)	706 (15.9%)	957 (21.6%)	**4430 (100%)**

(b) TBX11K Custom Split 2 - The publicly available TBX11K dataset split randomly and stratified for the purpose of this thesis. The number of images in each set of each label is listed below.

Label	Training	Validation	Test	**Total**
Non-TB	4864	1239	1636	**7739**
Active TB	377	96	157	**630**
Total	5241 (62.6%)	1335 (16%)	1793 (21.4%)	**8369 (100%)**

(c) Montgomery County Data Split - The publicly available Montgomery County dataset split randomly for the *ptbench* library. The number of images of each label in each set is listed below (table based on Raposo (2021)).

Label	Training	Validation	Test	**Total**
Normal	51	13	16	**80**
TB	37	9	12	**58**
Total	88 (63.8%)	22 (15.9%)	28 (20.3%)	**138 (100%)**

(d) Shenzhen Data Split - The publicly available Shenzhen dataset split randomly for the *ptbench* library. The number of images of each label in each set is listed below (table based on Raposo (2021)).

Label	Training	Validation	Test	**Total**
Normal	207	53	66	**326**
TB	215	54	67	**336**
Total	422 (63.7%)	107 (16.2%)	133 (20.1%)	**662 (100%)**

(e) New Delhi Dataset A Data Split - The publicly available New Delhi Dataset A split randomly for the *ptbench* library. The number of images of each label in each set is listed below (table based on Raposo (2021)).

Label	Training	Validation	Test	**Total**
Non-TB	41	10	26	**77**
TB	42	10	26	**78**
Total	83 (53.55%)	20 (12.90%)	52 (33.55%)	**155 (100%)**

(f) NIH CXR14 CheXNeXt Data Split - The publicly available ChestX-ray14 split according to Rajpurkar et al. (2018) (table based on Raposo (2021)).

	Training	Validation	Test	**Total**
Total	98'637 (90.5%)	6350 (5.8%)	4054 (3.7%)	**109'041 (100%)**

Table 4.2: OFFICIAL TBX11K SPLIT. *The publicly available TBX11K dataset is split by the authors according to this table (Liu et al., 2020). The reported numbers by the authors have a black font, the actual numbers gathered from the published dataset that deviate from the reported numbers are marked red.*

	Label	Training	Validation	Test	**Total**
Non-TB	Healthy	3000	800	1200	**5000**
	Sick & Non-TB	3000	800	1200	**5000**
TB	Active TB	473	157	294	**924**
	Latent TB	104 / 103	36	72	**212** / **211**
	Active & Latent TB	23	7	24	**54**
	Uncertain TB	0 / 1	0	10	**10** / **11**
	Total	6600 (59%)	1800 (16%)	2800 (25%) / 3302	**11'200 (100%)** / **11'702**

boxes if multiple regions where the abnormalities occur have been detected.

Data Collection Method The collected X-rays have been de-identified before they have been annotated with the bounding boxes by radiologists with 5-10 years of experience in TB diagnosis from top hospitals. Another radiologist with 10 or more years of experience then checked the annotations. The bounding boxes are accompanied by a label, either *latent TB* or *active TB*. There is a double-check system in place to ensure that mislabelled CXR are not being diagnosed wrongly (compared to a microbiology/sputum test). Noteworthy is also that it was possible to label a CXR of a patient with both, active and latent TB simultaneously (see Table 4.2) when presumably multiple bounding boxes were identified for a case, however, it is not further discussed how exactly this was achieved and what this exactly means for the final diagnosis.

Data Splits For the purpose of the thesis, the dataset has been split in two ways. One label was missing for a TB-positive case within the training set, thus this file has been removed from the training set.[1] As this dataset is unusual in the sense that it includes 4 or 5 different types of labels, some modifications were made to the labels when splitting the data. The 30 cases diagnosed with *simultaneous active & latent TB* have been removed from all the splits. It is unclear what such a diagnosis means, as usually a patient with TB is either active or latent, but not both at the same time. Since is unclear if they should be handled as aTB or latent TB only, they have been removed. *Latent TB* and *sick & non-TB* have been handled differently depending on the split. Since the labels for the test set are not publicly available, the pre-labelled validation set has been used as the test set and one part of the training set has been used as the new validation set instead. A ratio of training : validation : test of $4 : 1 : 1.25$ was aimed for, similar to the splits in the previous thesis by Raposo (2021) and in the *ptbench* package. The splits were made randomly while it was ensured that the labels were balanced (stratified) accordingly within each set.

The first split consists of the 2 labels *healthy* and *active TB*. The samples with the labels *sick & non-TB* and *latent TB* were fully removed from this split. Table 4.1a describes the split in more detail. The training and validation set consist of 86.4% *healthy* cases and 13.6% *active TB* cases. The test set consists of 83.6% *healthy* cases and 16.4% *active TB* cases.

The second split consists of the 2 labels *non-TB* and *active TB*. The labels *healthy*, *sick & non-TB*, and *latent TB* were merged into one label named *non-TB* for this split. Table 4.1b describes the

[1] likely to be *latent TB*, as only 103 cases were annotated as such instead of the reported 104 in Table 4.2 from the published dataset

split in more detail. The training and validation set consist of 92.8% *non-TB* cases and 7.2% *active TB* cases. The test set consists of 91.2% *non-TB* cases and 8.8% *active TB* cases.

Use Case The advantage of this dataset is that it not only can be used to measure the performance of direct models for RQ1, but also to evaluate the performance of visualisation methods with direct and indirect models for RQ2 due to the provided bounding boxes. Unfortunately, it cannot be used to evaluate the performance of indirect models for RQ1, specifically the first sub-model of such a model that detects specific pre-defined radiological signs.

Another advantage of this dataset is the inclusion of the *sick & non-TB* label. Since there can be an overlap in radiological signs of aTB cases and other diseases (Liu et al., 2020), this additional label could help to avoid introducing unintended biases and false positives by allowing the model to learn the distinction between aTB and other diseases. Future work can make use of split 2 in *ptbench* to investigate its impact in more detail.

4.1.2 Montgomery County

Description The Montogmery County (MC) dataset is one of the two datasets that the U.S. National Library of Medicine has published to help with improving CAD systems for TB detection (Jaeger et al., 2014). The 138 CXRs were published as 8-bit greyscale PNG files. Their resolution is either 4020X4892 or 4892x4020. The CXRs are divided into 2 types of labels, 80 belong to *normal* (healthy) and 58 belong to *TB-positive* class which show manifestations of TB (further details about the type of TB are not specified). For each person, the dataset includes information about their sex and age, and also clinical readings. Some clinical readings include more information than others. In general, a typical reading looks like this:

> Patient's Sex: F
> Patient's Age: 031Y
> cavitary nodular infiltrate in RUL; active TB

Data Collection Method The CXRs were taken by an Eureka stationary X-ray machine. The dataset was de-identified. The lung segmentations were generated by a custom algorithm under the supervision of a medical expert.

Data Splits The split implemented in *ptbench* was generated by Raposo (2021) by randomly splitting the dataset into a training, validation and test set as in Table 4.1c.

Use Case One of the base datasets that have been used to train the direct models for RQ1 and RQ2.

4.1.3 Shenzhen

Description The Shenzhen (CH) dataset is the second dataset that has been published by the U.S. National Library of Medicine (Jaeger et al., 2014) to help with improving CAD systems for TB detection. In total, 662 CXRs were published as 8-bit RGB PNG files of varying sizes, with widths/heights ranging from 948 to 3001 pixels. 326 cases are labelled as *normal* (healthy) and 336 as *TB-positive* class which show manifestations of TB (further details about the type of TB are not specified). As the Montgomery County Dataset (see Section 4.1.2), each PNG file is accompanied by the patient's sex, age, and clinical readings.

The version from Jaeger et al. (2014) is the one implemented on *ptbench*. Later on, a version of the dataset consisting of mask annotations with radiological sign information was published (Yang et al., 2022). This version also includes annotations with information about the location and shape of the radiological signs for TB-positive cases. Additionally, the binary masks were provided as PNG files. A patient can have multiple radiological signs, and thus multiple such PNG files, one for each sign. In total, the CXRs have been screened for 19 different radiological signs. This additional information makes the dataset the first dataset that published pixel-level annotations of PTB-related radiological signs.

Data Collection Method The CXRs have been captured from out-patient clinics with a Philips DR Digital Diagnose system. The dataset was de-identified.

The radiological sign annotation process was conducted by two radiologists from the Chinese University of Hong Kong. A junior radiologist performed the labelling while a senior radiologist checked the labels.

Data Splits The split implemented in *ptbench* was generated by Raposo (2021) by randomly splitting the dataset into a training, validation, and test set as in Table 4.1d.

Use Case One of the base datasets that have been used to train the direct models for RQ1 and RQ2.

In future work, the newly annotated version could be used to test the hypothesis from Raposo (2021) that PTB-specific annotations could elevate the performance of indirect models to those of direct models while remaining more interpretable and having stable predictions. For this work, the annotations and 19 radiological signs are not being used.

4.1.4 New Delhi - Dataset A

Description The two datasets whose CXRs were taken in the National Institute of Tuberculosis and Respiratory Diseases, New Delhi, India were published by Chauhan et al. (2014). They are divided into *Dataset A* and *Dataset B* since two different machines were used to obtain them. The images in the New Delhi Dataset A (IN) were published as 8-bit greyscale JPEG files that were uniformly resized to a resolution of 1024x1024. Dataset A consists of 156 CXRs, with 78 of them labelled as *non-TB*. It is not specified if these are healthy cases, or just cases that do not show signs specific to TB. 78 of the cases are labelled as *TB-positive* class which show manifestations of TB (further details about the type of TB are not specified). The focus in this thesis lies on Dataset A, as this is the one implemented in *ptbench*.

Data Collection Method The CXRs were taken according to standard practices followed by clinicians. They were collected randomly over a period of one year. The CXR in Dataset A were taken by a Diagnox-4050 X-ray machine manufactured by Meditronics and digitised by AGFA CR35-X. The CXRs were selected by the consensus of two highly experienced radiologists from the National Institute of Tuberculosis and Respiratory Diseases, New Delhi, India and Indira Gandhi Medical College, Shimla, India. They independently reviewed the images. Specifically, the selection of the *TB-positive* cases was based on the consensus on the radiological signs the radiologists found in the images.

Data Splits The split implemented in *ptbench* was generated by Raposo (2021) by keeping the test set as it is and splitting 20% of the training set for the validation set. The resulting split is

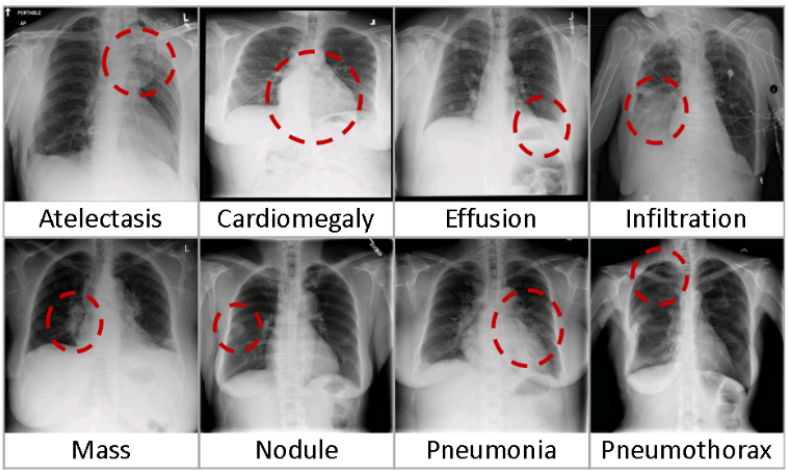


Figure 4.1: RADIOLOGICAL SIGN EXAMPLES. *In this figure, 8 of the 14 different radiological signs from the NIH CXR14 dataset are shown (figure by Wang et al. (2017)).*

shown in Table 4.1e. The split has excluded one of the *non-TB* cases from the initial training set, due to it potentially being a misclassification.

Use Case One of the base datasets that have been used to train the direct models for RQ1 and RQ2.

4.1.5 NIH CXR14

Description The ChestX-ray14 (NIH CXR14) dataset, initially known as *ChestX-ray8*, was published by (Wang et al., 2017) for Deep Learning training and is currently the largest publicly available CXR-related repository. 112'120 images of 30'805 unique patients are being provided as 8-bit greyscale PNG files with a resolution of 1024x1024. With the help of Natural Language Processing techniques, it was possible to mine the occurrence of the 14 different radiological signs as labels for each image. Multi-label annotations are provided, and they can either be 0 (negative) or 1 (positive) for each of the 14 radiological signs to indicate if the abnormality appears in the image or not. The 14 radiological signs are: *Atelectasis, Consolidation, Infiltration, Pneumothorax, Edema, Emphysema, Fibrosis, Effusion, Pneumonia, Pleural Thickening, Cardiomegaly, Nodule, Mass and Hernia*.

Additionally, metadata for each image is being provided: *Image Index, Finding Labels, Follow-up #, Patient ID, Patient Age, Patient Gender, View Position, Original Image Size and Original Image Pixel Spacing*. 984 images also include bounding box information.

The data comes pre-split into a training and test set, with images from the same patient only appearing in either of them.

Rajpurkar et al. (2018) have identified partially incorrect labels in the ChestX-ray14 dataset, and they have used an ensemble of models trained on the original set to relabel the training set. The test set has not been relabelled. They have further split the existing training set into a training and validation set. This version has been implemented in *ptbench* by Raposo (2021).

Data Collection Method The images were collected from the clinical PACS (Picture Archiving and Communication Systems) database of the National Institutes of Health (NIH) Clinical Center, USA.

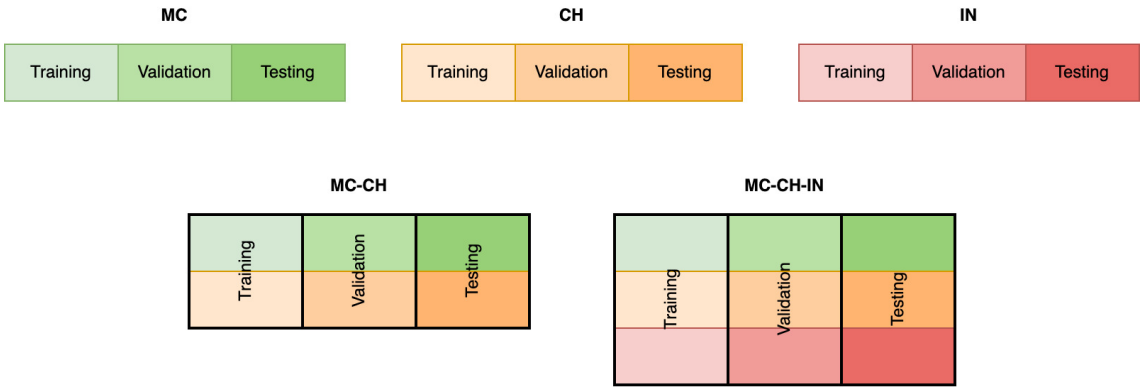


Figure 4.2: DATA AGGREGATION. *This figure depicts how the Montogmery (MC), Shenzhen (CH) and TBXpredict (IN) datasets have been consolidated to create 2 new datasets, Montgomery-Shenzhen (MC-CH) and Montogmery-Shenzhen-TBXpredict (MC-CH-IN) (figure by Raposo (2021)).*

Data Splits The split done by Rajpurkar et al. (2018) and implemented in *ptbench* can be seen in Table 4.1f.

Use Case This is the dataset that has been used to train the first DenseNet-121 sub-model of the indirect model for RQ1 and RQ2.

4.2 Data Processing

This section describes the data processing techniques that have been used for the experiments. Section 4.2.1 outlines the methods that have been used for some of the models for Research Question 1 and Research Question 2 to not only increase the performance, but also to improve generalisation, reduce overfitting & bias, and more.

4.2.1 Data Preprocessing Techniques

This section provides various techniques for preprocessing the data and datasets, either before starting with the training of the models, or also in general while training and for inference. This step is important and necessary as it directly affects the performance of the AI models that have been used in this thesis. For each technique, an explanation of how they work and how they are generally understood to improve the models is provided.

Dataset Aggregation

As mentioned in Section 4, publicly available TB datasets with CXR are rather sparse and usually too small. To increase the performance and generalisability of the models and to evaluate if and how much more data affects the performance, models have been trained on a gradually increasing number of data. The data has been gradually increased by consolidating the different splits of the datasets with each other, as depicted in Figure 4.2. The figure shows how the MC, CH and IN datasets have been aggregated to create the new datasets Montgomery-Shenzhen (MC-CH) and Montogmery-Shenzhen-New Delhi Dataset A (MC-CH-IN).

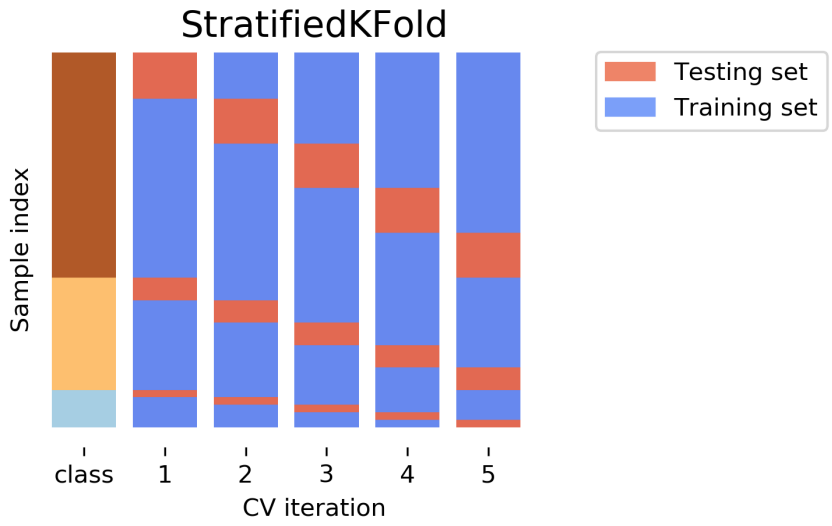


Figure 4.3: STRATIFIED K-FOLD CROSS-VALIDATION. *This figure depicts how the data is split into a training and test set in a 5-fold stratified cross-validation. Important to notice here is how each time the same relative amount of each class is split into training and test set (figure by Müller, Andreas C. (2020)).*

Stratified K-Fold Cross-Validation

To increase the number of available data to train and thus to also stop the model from overfitting (Pasa et al., 2019), stratified k-fold cross-validation is applied.[2] Since the datasets are rather small, different sets might lead to different performances depending on the split, and small changes in these sets could greatly influence the outcome. This method can reduce such variance. For RQ1, the models were trained with cross-validation to compare their performances with each other.

In k-fold cross-validation, the full dataset is split into k different folds, with each fold having different training/validation/test splits. This concept is depicted in Figure 4.3 with the training and test set only, but this concept can be extended to the validation, too. This way, instead of the data being used only once within e.g. a test set, the same data can be used multiple times within training and validation sets too.

In this thesis, 10-fold cross-validation was used only for the models in RQ1, meaning the data is split into 10 such different training/validation/test datasets. Stratified in this context means the ensurance of the folds keeping their original balance of the different class labels. For each fold, one-tenth of the full dataset was split off randomly as a test set. The remaining data of each fold was further split into a training and validation set.

There are two different ways to evaluate the performance of a model after doing a cross-validation, they both lead to the same result as long as the test sets have the same size (which they do). Either way, a separate model was trained for each fold. Their performances can be measured either by aggregating the predictions and using the measurement techniques on the aggregated predictions or by averaging each fold's performance. Both provide an overall understanding of the model's performance on the entire dataset.

[2]Depending on the dataset, it is applied differently, more implementation details can be found within the *ptbench* package

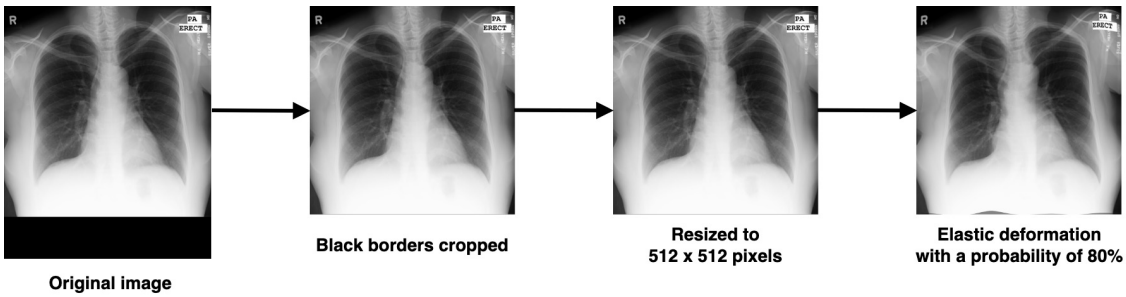


Figure 4.4: DATA AUGMENTATION PIPELINE. *This figure depicts how the CXR images are being preprocessed before being fed into the models for training (figure by Raposo (2021)).*

Data Augmentation

This popular technique also helps with over- & underfitting (the model is not able to capture the patterns in the data) and generalisability, as it makes the models more resilient by introducing artificial noise, distortions, or alterations to the input data while training (Hwang et al., 2019). Such alterations could also happen in images while using the model in a real-world scenario, which is why data augmentation allows models to work with similarly altered images too. Data augmentation can be achieved in different ways such as rotating, flipping, scaling, adding noise to the images, etc..

In this thesis, depending on the dataset and research question, data augmentation (or also called *transformations*) pipelines like depicted in Figure 4.4 were used. Below are explanations for some of them:

- **Removal of the black borders of images**: The removal of the black borders works by cropping out any rows or columns of the images that consist of only pure black pixels.
- **Resizing** In this work, to 512x512 if the image had a higher resolution. The original aspect ratio was disregarded.
- **Elastic deformation with a probability of 80%** Elastic deformation (Simard et al., 2003) was used since both Pasa et al. (2019) and Raposo (2021) have applied them and reported good results with it. It is believed to improve the resilience of the models.

For more details on more of the data augmentations and the pipelines, see Section 5.

Other preprocessing techniques

Z-Normalisation All the models in the *ptbench* package have normalisation implemented to achieve a zero mean and standard deviation of 1 in the distribution of the pixel values of the inputs. This is mainly an optimisation technique to speed up model training (Raposo, 2021):

$$x_{new} = \frac{x - \mu}{\sigma} \tag{4.1}$$

Here, μ stands for the mean and σ for the standard deviation. Some pre-trained models, depending on the dataset (e.g. ImageNet) they were trained on, need a different type of normalisation.

Chapter 5

Approach & Experiments

This chapter utilises the concepts introduced in Section 3 to explain the approach and experiment designs that were used to tackle the research questions introduced in Section 1.1.

It is divided into two main sections Section 5.1 and Section 5.2, one for each research question, and each of the sections is further divided into an approach and experiment design section. For research question 2, there is an additional section that explains one of the metrics that was used to answer it.

5.1 Research Question 1

5.1.1 RQ1 Approach

This section focuses on the approach and experiment design in order to answer Research Question 1. Here is a reiteration of it and its sub-questions:

> **RQ 1**: Can the prediction of the probability of active Tuberculosis by direct and indirect deep learning models be improved through the use of a new dataset specific to active Tuberculosis?
>
> - **RQ 1.1**: How do the AUC scores from Raposo (2021) compare to the replicated models using the same methods and frameworks?
> - **RQ 1.2**: Does the inclusion of different types of labels in the TBX11K dataset (healthy, latent TB, sick & non-TB) affect the model's ability to discriminate active TB cases?
> - **RQ 1.3**: Does the inclusion of the TBX11K dataset during training affect the generalisability of the models?
> - **RQ 1.4**: How do the AUC scores of the models that included the TBX11K dataset during training compare to the replicated models based on Raposo (2021)?

To answer the research question, a baseline for the models needs to be trained. Building upon the previous work of Raposo (2021), the methodology, experiment design, and results of the author were studied to replicate his results for this part. To do that, the newer version of the framework previously utilised by the author, *ptbench* for Python, was used (see Section 3.3.1). In the first step, his results were replicated with the same datasets as in Raposo (2021). In the second step, a newly publicly released dataset, TBX11K, had been selected to further improve these models (Liu et al., 2020). In a third step, the dataset has been implemented into *ptbench* to train new models

similarly to Raposo (2021). The aim is to measure the performance of different types of direct and indirect models for more with and without the new dataset.

Aside from evaluating if a new dataset can further improve the direct and indirect models, this also leads to the replication of the results from the previous work, further validating them. At the same time, this research question serves as a preparatory step for research question 2, as a similar approach can be used to train the models there, too.

Since there is a problem of data scarcity (Section 4), an additional CXR/TB-related dataset was expected to further improve the generalisability and performance of the existing model architectures. Raposo (2021) reported how the lowest validation loss was reached quickly during training, indicating a low data variety, and a new dataset can further help to increase that variety. In addition, previous publicly available datasets only made the distinction between *healthy* and *TB* cases. The new dataset, TBX11K, includes new types of labels, which are expected to further improve the models' generalisability by improving the models' ability to discriminate TB cases from not only *healthy*, but also from *latent TB* and *sick & non-TB*.

Raposo (2021) hypothesised that an indirect model could be further improved by providing the model with a PTB-only based dataset with radiological sign labelling during training. Since such a dataset was not available at the time of writing this thesis, and since training many such models can take up to months, the third category of models mentioned in Section 3.1.4 was skipped. This means the experiments focus only on *Direct Models without pre-training on Radiological Sign labels* and *Indirect Models with Radiological Sign labels*.

The TBX11K dataset not only includes labels for *healthy* and *aTB*-positive as the other previously publicly available TB-based dataset did, but also for *sick & non-TB* and *latent TB*. Due to this, two different data splits were utilised to measure if it makes a difference if the *healthy*, *sick & non-TB* and *latent TB* cases are included in the aTB-negative class, while only *aTB* cases are included in the aTB-positive class, as compared to when there is only a distinction between *healthy* and *aTB* (see Section 4.1.1 for details on the splits). In addition to the class labels, the dataset also includes ground truth bounding box annotations that are based on radiological signs for TB-positive cases, which is especially important for Research Question 2.

Two properties need to be quantitatively measured to compare the models with each other: how often is the model correct in discriminating aTB cases from *healthy* cases, and how well do these models generalise. To measure the first property, it is necessary to make a distinction between practical application and theory. Medical experts, when making a decision on which model to use, which can depend not only on the composition of the population but also on the specific intention of the detection process and the pursued strategy (World Health Organization, 2021), might have different needs on the Sensitivity and Specificity of the models. These will be reported in Section A.1 in the attachment. These two values make it difficult to directly compare the practical performance of the models, which is why the F1-Score is reported additionally as a single metric to do that. The reason why F1-Score is preferred over accuracy is due to the high class imbalance in the datasets, which the F1-Score can counteract to a certain degree. The models return a continuous value between 0 and 1, and a threshold needs to be selected to discriminate between the classes to calculate the F1-Score. E.g. when selecting 0.5 as the threshold, for each sample the model outputs a value under 0.5 would be classified as *healthy*, and vice versa everything that equals 0.5 and above would be classified as *TB-positive*. While it is possible to get an optimal threshold based on which the F1-Score can be calculated for practical purposes, a holistic and theoretical view of the model's overall ability to discriminate the classes is also important to quantify. This can be done through the AUC score. Since the metric is threshold-independent, it is a suitable metric to compare the theoretical overall performance of the models with each other. For the second property, quantification of the generalisability, the same technique used by Raposo (2021) is being applied. The datasets are gradually being aggregated (see Section 4.2.1 to see if the overall AUC performance on the different individual test sets of the datasets improves. As men-

Table 5.1: HYPERPARAMETERS FOR THE MODELS. *This table shows the hyperparameters that were used when conducting the experiments. The model names are specified on the left side, while the hyperparameters Learning Rate, Batch Size, and Training Epochs are specified in the other 3 columns.*

Model	Learning Rate	Batch Size	Training Epochs
Pasa (direct)	4	$8 * 10^{-5}$	500
DenseNet-121 (direct)	8	$5 * 10^{-5}$	2000
DenseNet-121 (indirect)	8	$1 * 10^{-4}$	10
Logistic Regression (indirect)	4	$1 * 10^{-2}$	100

tioned, another piece of information that can potentially be extracted from the TBX11K dataset is whether the two different splits make a difference. The ability to discriminate and generalise the models trained on these splits will also be measured with AUC. In summary, the F1-Scores and the thresholds used to achieve them are presented in Section 6 for each model. To compare them for their ability to discriminate and generalise, the AUC is being used. In total, four main types of comparisons are conducted with AUC:

- Comparing performance of the replicated models with the ones from Raposo (2021)
- Comparing the performance of the models that included the different splits of the TBX11K dataset with each other
- Comparing the performance of the models with different amounts of aggregated dataset
- Comparing the performance of the new models, including those that included the different splits of the TBX11K dataset, with the replicated results

5.1.2 RQ1 Experiment Design

This section focuses on how the experiments for research question 1 were conducted. The process can be split into two parts: one is the training of the models, and one is the evaluation process.

The models are of type *Direct Models without pre-training on Radiological Sign labels* and *Indirect Models with Radiological Sign labels*. Figure 1.3 is a high-level depiction that highlights their main differences.

For the direct models, two different types of CNN architectures were used as they were implemented in *ptbench*: The Pasa Model and DenseNet-121. Both were initialised with random weights before training. The indirect models consist of two sub-models. DenseNet-121 was used for the Multilabel-Classifier sub-model, but it was slightly modified to have a singular fully connected layer with 14 outputs for each of the radiological signs. Another difference is that DenseNet-121 was loaded with the weights provided by PyTorch which were obtained by training the model on the ImageNet dataset. For the second sub-model, a Logistic Regression Classifier was used.

Adam with the PyTorch default parameters (β_1 = 0.9, β_2 = 0.999, and ϵ = 1 * 10^{-8}) was used as the optimiser, and *Binary Cross-Entropy Loss* as the loss function for all the (sub-)models during training. Raposo (2021) did a grid search on the aggregated dataset MC-CH-IN with different hyperparameters to find the best performing (i.e. lowest validation loss) DenseNet-121 and Logistic Regression Classifier configurations, and these two model types in this work were trained using them. For the Pasa model, Raposo (2021) has used the recommended hyperparameters by Pasa et al. (2019), which are also used here. Table 5.1 shows the different hyperparameters that were selected.

After training the models through all the epochs, the version of the model in the epoch with the lowest validation loss was selected to be the final version.

Table 5.2: DATASETS FOR TRAINING. *This table shows the datasets that were used to train the direct Pasa and DenseNet-121 models, and the Logistic Regression Classifier. It includes aggregated datasets.*

Datasets
TBX11K Split 1 (i.e 11k)
TBX11K Split 2 (i.e 11kv2)
MC
MC-CH
MC-CH-IN
MC-CH-IN-11k
MC-CH-IN-11kv2

As for the datasets that were used for training, Table 5.2 shows all the datasets that were used for the Pasa, DenseNet-121 and Logistic Regression Models. For the DenseNet-121 sub-model of the indirect model, NIH CXR14 was used. Since there are 7 different datasets, this leads to 7 different Pasa models and 7 different direct DenseNet-121 models. While the first sub-model of each indirect model is the same DenseNet-121 model trained on NIH CXR14, the logistic regression classifier part was trained on the 7 datasets, also leading to 7 different indirect models.

Just as Raposo (2021), the mentioned preprocessing steps in Section 4.2, namely data augmentation for enhanced generalisability and data variety, stratified 10-Fold Cross Validation for more data variety, class weights to counteract class imbalance, colour space conversions to greyscale to unify the images and make them compatible with the model architecture, and z-normalisations to speed up training, have all been applied, too.

For MC, CH, and IN, the same data augmentation pipeline as in Figure 4.4 was used. The custom function to remove any excess black border from the images was used, in addition to a resizing to 512x512, and Elastic Deformation with a probability of 0.8. For the NIH CXR14 dataset, resizing to 512x512 and random horizontal flips were used. For both TBX11K splits, only Elastic Deformation with a probability of 0.8 was used.

The stratified 10-Fold Cross Validation was done by splitting the datasets into 10 different folds and training a model for each fold. The only exception is the DenseNet-121 sub-model of the indirect model, which was trained without cross-validation and used to derive the Radiological Sign labels from the datasets mentioned in Table 5.2 before being used as input for the Logistic Regression Classifiers with 10-Fold Cross Validation again.

The optimal threshold for each model was selected by using each model on the validation split of the dataset they were trained on to predict the class labels. The threshold leading to the highest F1-Score was selected as the optimum. This was done for each of the 10 sub-models for each of the 10 folds, and the average threshold was selected as the final optimum threshold.

The final model performance was evaluated on the individual, non-aggregated test splits of each dataset. The datasets used for this were MC, CH, IN, TBX11K Split 1 & 2. To clarify more, each of the 10 sub-models predicted the label of the samples belonging to the test set of each of these datasets. For each dataset, the predictions from the 10 sub-models were consolidated into 1 file before the optimum threshold was used to classify the samples into *healthy/non-TB* or *TB-positive*. The final sensitivity, specificity, F1-Score and AUC were calculated based on this classification performance.

5.2 Research Question 2

5.2.1 RQ 2 Approach

This section focuses on the approach and experiment design to answer Research Question 2. Here is a reiteration of it and its sub-question:

> **RQ 2**: What novel visualisation methods exist for increasing the explainability of deep learning models trained on a dataset specific to active Tuberculosis by visualising radiological signs and how well do they perform?
>
> - **RQ 2.1**: What measurement techniques are suited to evaluate the performance of the visualisations?

As mentioned in Section 2, saliency mapping techniques have been identified to be the most promising explainability method for the use case of screening CXRs for aTB, as previous publications in this field focus on such visualisation methods, which are also based on the needs and recommendations of medical experts (Pasa et al., 2019; Rajpurkar et al., 2020). Often, as models become more complicated and accurate, they also tend to become less interpretable. Deep CNNs are considered to be such complicated models due to their sheer size alone (Castelvecchi, 2016; Selvaraju et al., 2017). Generally, Saliency Maps provide post-hoc interpretability by adding transparency to deep CNNs, which increases the interpretability, and thus trust in these models since it becomes possible to understand what the model has based on its (correct/wrong) decisions on (Selvaraju et al., 2017). Depending on the saliency mapping technique, the amount of completeness the explanation provides can vary. While Grad-CAM offers little completeness through local transparency and local attribution, FullGrad offers full completeness through global attribution (Srinivas and Fleuret, 2019). Full completeness is not necessarily a desirable property for a good explanation (Lakhani and Sundaram, 2017), a good trade-off between completeness and interpretability is more important. The usefulness of an explanation, however, is very important. If the explanation is not accurate, the explanation is not useful. So it becomes important to choose faithful metrics, i.e. ones that accurately represent the performance of the explanations, to evaluate the usefulness. This leads back to research question 2, as the amount of explainability a visualisation method offers is directly related to its usefulness, i.e. its performance. Assessing the performance of the visualisation methods in practice and theory also indirectly measures their explainability to a certain extent.

As mentioned in Section 2.3.1, since there are various saliency mapping techniques, and since the evaluation methods are all over the place, there is no clear indication which of these methods would be suited the best. The selection of the techniques was not based on their reported performance metrics, but rather on an attempt to cover a broad spectrum of the principles they are based off. Grad-CAM was selected as a baseline since it is a well-known popular method that has been previously used in the (medical) field. Score-CAM was selected since it is a method based on a scoring system with CAMs, without gradients. Conversely, FullGrad was selected due to its inherent explainability properties (offering completeness and weak dependence), and since it is a model based heavily on gradients. This technique stands in contrast to Score-CAM, covering the other extreme of the spectrum of saliency mapping techniques. The *pytorch-grad-cam* library implemented these three saliency mapping techniques. Since the library is compatible with existing PyTorch-based models, the library has been implemented into *ptbench* to conduct the experiments for Research Question 2. The library also included more saliency mapping techniques than the three mentioned ones. This opportunity was used, and the other techniques have also been incorporated into the experiments. In total, 12 different saliency mapping techniques have been used, listed in Table 5.3.

Table 5.3: SALIENCY MAPPING TECHNIQUES. *This table shows the saliency mapping techniques that were used for this experiment. All these techniques are implemented in the PyTorch-grad-cam library for Python.*

Saliency Mapping Technique
Ablation-CAM
Eigen-CAM
EigenGradCAM
FullGrad
Grad-CAM
GradCAMElementWise
Grad-CAM++
HiResCAM
LayerCAM
RandomCAM
Score-CAM
XGrad-CAM

Previous publications that used AI-based CADs to detect aTB focused, if at all, on a qualitative approach to analyse and evaluate the saliency maps (Lakhani and Sundaram, 2017; Pasa et al., 2019; Rajpurkar et al., 2020; Raposo, 2021). Thanks to the TBX11K dataset and its ground truth bounding box annotations of the locations of radiological signs, it is possible to quantitatively evaluate if the saliency mapping techniques can produce heatmaps located in the same areas that were identified by medical experts. Since the localisation performance of saliency mapping techniques becomes quantifiable, it also becomes possible to compare the performance of the different techniques with each other. A downside of these annotations in TBX11K is that the authors did not provide labels for the detected abnormalities. This means it is not possible to evaluate if the correct radiological signs are being detected by the first sub-model of the indirect model. The evaluation with these labels is limited to the ability of the techniques to correctly localise the area only.

As for the performance metrics for the saliency mapping techniques, two different types of metrics were chosen: metrics that measure the overall performance of the saliency mapping techniques in combination with the underlying CNN, thus having more practical relevance, and one that attempts to measure the saliency mapping technique itself as isolated as possible from the underlying CNN.

As mentioned in Section 3.4.2, IoU and IoDA are two popular metrics that are commonly used in the AI and medical field to evaluate the performance of visualisations (Selvaraju et al., 2017; Wang et al., 2017; Zhou et al., 2016). To use them, some additional processing steps need to be introduced to obtain bounding boxes from the saliency maps. Usually, it is necessary to draw bounding boxes or outline the contour of the detected area based on a threshold. In previous publications (Selvaraju et al., 2017; Wang et al., 2017), these thresholds were selected either arbitrarily or the process of generating them was not discussed at all. This introduction of extra calculation steps leads these metrics to be not faithful in measuring the performance of the visualisation technique. For that reason, Proportional Energy, introduced by Wang et al. (2020), has been selected as the main metric for the performance of the techniques. This metric does not introduce any additional processing steps before evaluation, and it is also threshold-independent, which makes the metric more faithful. While Proportional Energy is a suited metric to evaluate the overall performance of a CAD system in practical settings, it is not suitable to evaluate and compare the performance of the saliency mapping techniques with each other. This metric

is not able to measure the performance of the saliency mapping techniques in an isolated way, separate from the performance of its underlying CNN model. If the underlying model is not good at discriminating *healthy* cases from aTB cases, then there is no reason to believe that the localisation of the important areas based on the calculations made by the faulty model is accurate, either. Since the saliency mapping techniques explain the model's behaviour, by using the Proportional Energy metric, low-scoring visualisations should not be falsely attributed to the visualisation method only. So there is a problem of inseparability between the saliency mapping technique's and the CNN model's performance. This leads to pixel perturbation methods such as the one used in Chattopadhay et al. (2018). By using the saliency map to find the most (MoRF) and least (LeRF) important pixels for example, and using this information to perturb the original image, and then forwarding the original and perturbed image and measuring the change in output, it is possible to somewhat isolate the performance of the model from the one of the visualisation methods by only measuring the saliency mapping techniques ability to highlight important/unimportant pixels (see Section 3.4.3 for a more detailed explanation). But this change in the original image shifts the distribution of the pixel values in the image, so it is still possible that the model is performing worse because the model has never seen unnatural images like that before. ROAD efficiently solves this problem by using a special algorithm to perturb the pixels in a way that they seem more *natural*. By using MoRF and LeRF with ROAD, it is possible to obtain scores that separate the saliency mapping technique's performance from the model's performance, at least to a higher degree than the previous methods. Since all the models use the Sigmoid activation function to predict the class probability, the difference between the class probabilities of the original and the perturbed image is used to calculate the change in prediction. For MoRF, the class probability is expected to drop, while for LeRF, the class probability is expected to either stay the same or increase. The research on this topic is relatively new and it is currently unclear if MoRF or LeRF should be the preferred perturbation method to be used with ROAD (Chattopadhay et al., 2018; Srinivas and Fleuret, 2019; Tomsett et al., 2020). That is why their combined average is taken as the final singular score to make the measurement and comparison of the performances of the saliency mapping techniques simpler. The higher the combined ROAD value, the better the saliency mapping technique. The values the score can take range from -1 to 1:

$$ROAD_{Combined} = \frac{ROAD_{LeRF} - ROAD_{MoRF}}{2} \tag{5.1}$$

AOPC (see Section 3.4.3) is used together with the combined ROAD score to avoid choosing a deliberate threshold for the number of pixels to perturb. Specifically, 20%, 40%, 60%, and 80% have been chosen.

Further, after this section, a new metric is motivated and introduced in more detail in Section 5.2.2, which is called *ROAD-Weighted PropEng* that combines the Proportional Energy and $AOPC_{Combined}$ scores to create one final metric for the evaluations of the visualisations.

For each of the saliency mapping technique and model combinations, the Proportional Energy, $AOPC_{Combined}$, and ROAD-Weighted PropEng are being calculated for each aTB-positive image in the TBX11K test set (since the focus lies on aTB, and since ground truth labels only exist for TB-positive cases). The TBX11K dataset is used to select the best model simply because the ground truth annotations are only available for the TBX11K dataset, and only the previously unseen test set is used to ensure the generalisability of the results. The reason only aTB-positive cases were visualised is because the goal is to visualise regions where TB and radiological signs are present, and not the other way around. It is unclear how visualisations of *healthy* cases are to be interpreted, as the model focuses on detecting the presence of aTB, and not on the non-presence of such signs.

After obtaining the metrics for each image, the median over the dataset for the Proportional Energy and $AOPC_{Combined}$ metrics is computed as the final metric to compare each visualisation

method + CNN model combination with each other. The average is not being used to avoid the influence by outliers, which can be especially skew the results in a small dataset like the TBX11K test set. The *ROAD-Weighted PropEng* metric is an exception since it has been constructed differently. For this metric, the weighted average has been used to normalise the metric over the dataset to allow for better comparability between different visualisation method + model combinations. The final metric acquired through this process is called *ROAD-Normalised PropEng Average*. This is used as a single metric to select the overall most promising saliency mapping technique and CNN model combinations to further focus the qualitative analysis.

In summary, the *Proportional Energy* is used to measure the performance of each such combination as one package. The $AOPC_{Combined}$ based on ROAD attempts to separate the performance of the saliency mapping technique from the model. The *ROAD-Normalised PropEng Average* metric attempts to combine both of these metrics to select the overall best saliency mapping technique and CNN model combinations, and also thus the best visualisations with practical relevance, such that further qualitative analysis can be conducted on the visualisations generated by these combinations.

Aside from the quantitative analysis, after obtaining the three metrics, the 12 visualisations that scored the highest were briefly qualitatively analysed for the $AOPC_{Combined}$ and Proportional Energy metrics. Qualitative analysis in this sense means simply the observation of the visualisations by a data scientist, the author of this work (a non-radiologist or medical expert). Observations help to check if the metrics were able to capture what they intended to, especially when visually comparing the heatmaps to the ground truth labels. This analysis was done for each of the models that were selected for the final qualitative analysis (see Section 5.2.3 for more details), so in total, 8 batches of images were observed to ensure the visualisations worked as intended. The visualisations can be found in Section A.2. Further, after obtaining these results, based on these heatmaps, biases in the DenseNet-121 model trained on MC-CH-IN-11k were detected and the 3 of the 4 selected models were re-trained with more data augmentations, as described in Section 5.2.3. The qualitative analysis was then repeated for the 3 models that were selected (see also Section 7.2 for the discussion and interpretation of the first batch of visualisations). In the result section, only the saliency mapping techniques for the DenseNet-121 model trained on MC-CH-IN-11k that obtained the $AOPC_{Combined}$ and Proportional Energy metrics are being shown, before and after using the new data augmentation pipeline to enable comparison between the visualisations. Afterwards, for the overall highest scoring model and saliency mapping technique combination on *ROAD-Normalised PropEng Average*, the best and the worst performing 12 visualisations have been selected for the qualitative analysis. Additionally, 12 randomly selected visualisations on healthy cases are provided to see what the model focuses on the non-presence of any radiological signs, or generally aTB.

The saliency maps are overlaid as heatmaps over the original image, without any smoothing techniques to avoid the introduction of additional noise and thus to ensure that the explanation remains more faithful. The saliency maps are also not thresholded, meaning the full saliency maps are visualised. Ground truth bounding boxes are also drawn, if applicable, which helps with the qualitative analysis.

5.2.2 Custom Combined Performance Score for Visualisation and Model

Proportional Energy measures the alignment of the saliency mapping technique's and CNN model's output with the radiologist's ground truth labels. $AOPC_{Combined}$ with ROAD measures the fidelity of the saliency mapping technique to the model (i.e. how well it explains what the model bases its decisions on), and thus it measures how well the saliency mapping technique performs at what it is supposed to do: explain the model's decision.

Both metrics are important for a good visualisation. If both metrics agree and show a high score for an individual score, then the model was able to localise the region well according to human observation, and the visualisation accurately represents the decisions the model has made. The opposite is true if the visualisations score low on both metrics.

But there is also the possibility that they do not agree with each other. If the Proportional Energy scores high, but the $AOPC_{Combined}$ scores low, it could mean that the visualisation is capturing the features that agree with the human experts, but the visualisation might not necessarily be faithful to the model's actual decisions. And for the other case that the $AOPC_{Combined}$ is high, and the Proportional Energy is low, the implications become even more unclear. It could simply be the case that the model is not able to localise the region correctly. But it could also mean that the model is focusing on features that humans do not consider important or understandable. This can be either due to the model capturing complex patterns that are beyond human comprehension or simply because the model is biased.

Thus, the new combined metric *ROAD-Weighted PropEng* is being proposed. It weights the Proportional Energy with $AOPC_{Combined}$. $AOPC_{Combined}$ can be used as an adjustment mechanism by weighing the Proportional Energy, giving more emphasis to the localisation performance (according to humans) if the visualisation was more faithful to the model's calculations. Such a visualisation is not only practically relevant but also explainable. Additionally, this new metric also allows for a trade-off between adhering to human limitations vs. allowing potentially beyond-human pattern recognition. This results in a single metric combining both metrics to evaluate the overall performance of the visualisations.

This new metric can be calculated for each sample as:

$$\text{ROAD-Weighted PropEng} = \max(0, AOPC_{Combined}) \cdot ProportionalEnergy \tag{5.2}$$

A negative $AOPC_{Combined}$ is not meaningful since it suggests that the visualisation was not successful. Thus, negative $AOPC_{Combined}$ values are changed to 0 before calculating the ROAD-Weighted PropEng.

To further determine the best performing CNN model and saliency mapping technique combination and make them comparable with each other, the ROAD-Weighted average needs to be computed from the individual ROAD-Weighted PropEng scores of a dataset for each saliency mapping technique and CNN model combination:

$$\text{ROAD-Normalised PropEng Average} = \frac{\sum_m \text{ROAD-Weighted PropEng}_m}{\sum_m AOPC_{Combined[m]}} \tag{5.3}$$

This weighting emphasises the focus on practical relevance. Samples with a higher $AOPC_{Combined}$ will weigh the Proportional Energy more, indicating that a high Proportional Energy was not simply achieved by chance. Model and saliency technique combinations scoring the highest will have the most useful visualisations.

5.2.3 RQ2 Experiment Design

The non-cross-validated *Direct Models without pre-training on Radiological Sign labels* from RQ1 is being used as the underlying CNN models for the visualisations. Since there are 12 saliency mapping techniques in total, and since it would be necessary to train 10 different sub-models for each model if stratified 10-fold cross-validation was being used like in RQ1 for RQ2, this data preprocessing technique has been omitted. With the available hardware, generating visualisations for 12 techniques * 14 models * 10 sub-models, and then evaluating them all with ROAD is not feasible. Aside from that, the 14 models have been trained the same way they were trained in RQ1. To further focus the quantitative analysis, only the 3 models that achieved the highest

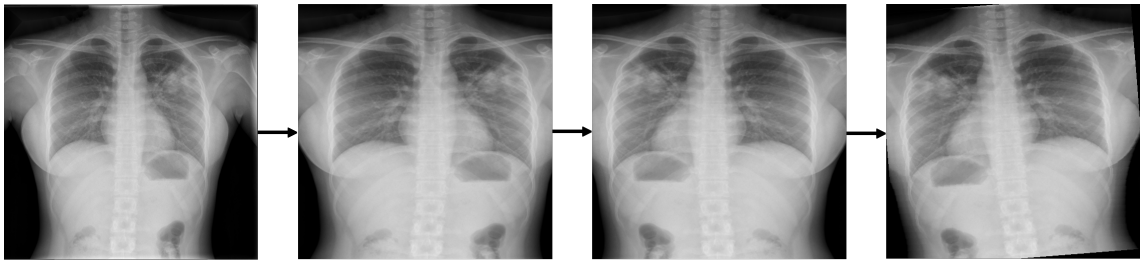
AUC on the TBX11K test set have been selected for the evaluation with the Proportional Energy and $AOPC_{Combined}$. There is also an additional restriction, namely that the Pasa Model is not compatible with the implementation of FullGrad in the *pytorch-grad-cam* library. Thus in total, there will be between 33 - 36 saliency mapping technique + direct model combinations, depending on whether the selected 3 models are all Pasa models or not.

Further, the data augmentations for the selected models were exactly as described in Section 5.1.2. Then, the 3 selected models that achieved the highest on the TBX11K dataset were further selected to be trained with an additional (see Section 7 for more details) different data augmentation pipeline. The intention with this new pipeline was to reduce eventual biases that the models have shown according to the qualitative analysis:

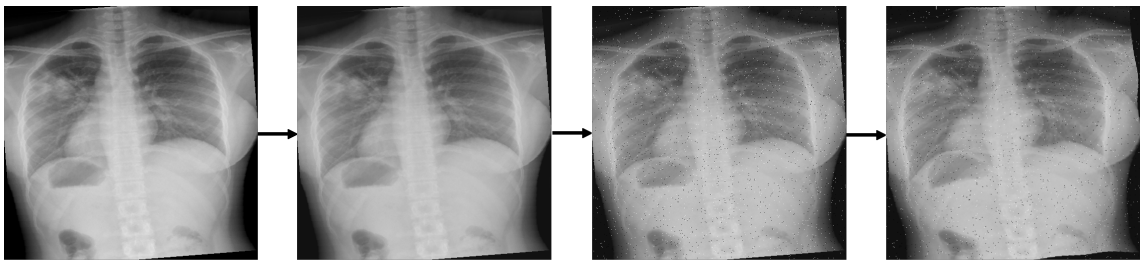
- random resized crops by cropping to 75% - 100% of the original image size (with varying aspect ratios) and resizing it to 512 x 512
- random horizontal flips
- random rotation by 5 degrees
- random changes to the contrast and saturation ranging from factors of 0.8 to 1.2
- random salt-and-pepper noise for 1% of the pixels with a probability of 25%

The random resized crop is only being used for the TBX11K dataset, while the other additional data augmentations are being used for MC, CH, IN, and TBX11K. Figure 5.1 depicts the full new pipeline for TBX11K.

Last but not least, visualisations are also generated by using the Multi-label DenseNet-121 model that was used as the first sub-model of the *Indirect Models with Radiological Sign labels*. Since this model is non-cross-validated and unique, there is no need for retraining or a selection process. This is done to see how well a model trained only on radiological signs performs at localising the important regions of aTB-positive CXRs. This model, even though not trained on TB-specific data, should still be able to correctly localise radiological signs that are present in the TBX11K dataset, since the ground truth annotations are based on them. Since the model has 14 outputs, one for each sign, the saliency maps were only generated for the highest-scoring class(es).



(a) Part 1 - This figure depicts how the CXR images from the TBX11K dataset are being preprocessed before being fed into the models for training for some of the models in Research Question 2.



(b) Continuing from (a), this figure depicts how the CXR images from the TBX11K dataset are being preprocessed further before being fed into the models for training for some of the models in Research Question 2.

Figure 5.1: Data Augmentation Pipeline TBX11K for RQ2

Chapter 6

Results

6.1 Research Question 1

As stated in Section 5.1, the results for the first research question are split into 4 subsections:

- Comparing the performance of the replicated models with the ones from Raposo (2021)
- Comparing the performance of the models that included the different splits of the TBX11K dataset with each other
- Comparing the performance of the models with different amounts of aggregated dataset
- Comparing the performance of the new models, including those that included the different splits of the TBX11K dataset, with the replicated results

Comparing replicated models with original models

Table 6.1 depicts the results of Raposo (2021) for *Direct Models without pre-training on Radiological Sign labels*, while Table 6.2 depicts the replicated results. In general, the absolute AUC score deviations of the replicated results from the results of the thesis of Raposo (2021) lie between 0 and 0.056. The Wilcoxon signed-rank test resulted in a p-value of 0.3.

Table 6.3 depicts the results of Raposo (2021) for *Indirect Models with Radiological Sign labels*, while Table 6.4 depicts the replicated results. The absolute AUC score deviations of the replicated results from Raposo's results lie between 0 and 0.005. The Wilcoxon signed-rank test for the indirect models results in a p-value of 0.17.

Table 6.1: This table depicts the AUC scores that Raposo (2021) obtained for the two types of direct models trained on different datasets. The AUC scores were calculated individually for each test set of the three datasets that were used.

Model	AUC MC Test	AUC CH Test	AUC IN Test
Pasa trained on MC	0.890	0.576	0.642
Pasa trained on MC-CH	0.870	0.893	0.669
Pasa trained on MC-CH-IN	0.881	0.898	0.848
DenseNet-121 trained on MC	0.822	0.607	0.625
DenseNet-121 trained on MC-CH	0.883	0.905	0.672
DenseNet-121 trained on MC-CH-IN	0.860	0.917	0.850

Table 6.2: This table depicts the AUC scores that were obtained through replication for the two types of direct models trained on different datasets. The AUC scores were calculated individually for each test set of the three datasets that were used.

Model	AUC MC Test	AUC CH Test	AUC IN Test
Pasa trained on MC	0.886	0.615	0.631
Pasa trained on MC-CH	0.877	0.902	0.645
Pasa trained on MC-CH-IN	0.905	0.916	0.848
DenseNet-121 trained on MC	0.790	0.551	0.620
DenseNet-121 trained on MC-CH	0.842	0.899	0.649
DenseNet-121 trained on MC-CH-IN	0.881	0.901	0.830

Table 6.3: This table depicts the AUC scores that Raposo (2021) obtained for the indirect models trained on different datasets. The AUC scores were calculated individually for each test set of the three datasets that were used.

Model	AUC MC Test	AUC CH Test	AUC IN Test
Indirect trained on MC	0.966	0.867	0.926
Indirect trained on MC-CH	0.961	0.901	0.928
Indirect trained on MC-CH-IN	0.951	0.895	0.920

Table 6.4: This table depicts the AUC scores that were obtained through replication for the indirect models trained on different datasets. The AUC scores were calculated individually for each test set of the three datasets that were used.

Model	AUC MC Test	AUC CH Test	AUC IN Test
Indirect trained on MC	0.966	0.872	0.928
Indirect trained on MC-CH	0.961	0.901	0.927
Indirect trained on MC-CH-IN	0.950	0.898	0.921

Table 6.5: This table depicts the AUC scores that were obtained for the two types of direct models trained only on the two different splits of TBX11K. The AUC scores were calculated individually for each test set of the five datasets that were used.

Model	AUC MC Test	AUC CH Test	AUC IN Test	AUC 11k Test	AUC 11kv2 Test
Pasa trained on 11k	0.663	0.536	0.576	1	0.873
Pasa trained on 11kv2	0.593	0.501	0.667	0.999	0.986
DenseNet-121 trained on 11k	0.632	0.571	0.605	1	0.808
DenseNet-121 trained on 11kv2	0.626	0.603	0.692	0.999	0.988

Table 6.6: This table depicts the AUC scores that were obtained for the indirect models trained only on the two different splits of TBX11K. The AUC scores were calculated individually for each test set of the five datasets that were used.

Model	AUC MC Test	AUC CH Test	AUC IN Test	AUC 11k Test	AUC 11kv2 Test
LogReg trained on 11k	0.899	0.870	0.881	**0.978**	0.861
LogReg trained on 11kv2	0.888	0.810	0.731	0.949	**0.907**

Comparing TBX11K models with each other

Table 6.5 depicts the obtained AUC scores for *Direct Models without pre-training on Radiological Sign labels*. To reiterate, the (TBX)11k first split consists of 2 classes: the *healthy* cases are in the first class, while the second class consists of *aTB* cases. The (TBX)11kv2 split consists also of 2 classes, however, the *healthy*, *sick & non-TB*, and *latent TB* all belong to the *aTB-Negative* class while the positive class remains the same with only *aTB* cases.

The direct models trained on 11K achieve an AUC of 1 on the 11Kv2 test set, while the direct models trained on the 11kv2 set achieve 0.999 on the same test set. The Pasa model trained on 11k achieves an AUC of 0.873 on the 11kv2 test set, while the Pasa model trained on 11kv2 achieves 0.986 on the same test set. Further, the DenseNet-121 model trained on the 11k dataset achieves an AUC score of 0.808 on the 11kv2 test set, while the DenseNet-121 model trained on the 11kv2 dataset achieves an AUC of 0.988 on the same test set.

Table 6.6 depicts the obtained AUC scores for *Indirect Models with Radiological Sign labels*. The indirect model trained on the 11k dataset achieves an AUC score of 0.978 on the 11k test set and

Table 6.7: This table depicts the AUC scores that were obtained for the two types of direct models trained with an increasing number of datasets through data aggregation. The AUC scores were calculated individually for each test set of the four datasets that were used.

Model	AUC MC Test	AUC CH Test	AUC IN Test	AUC 11k Test
Pasa trained on MC	0.886	0.615	0.631	0.839
Pasa trained on MC-CH	0.877	0.902	0.645	0.833
Pasa trained on MC-CH-IN	0.905	**0.916**	**0.848**	0.824
Pasa trained on MC-CH-IN-11k	**0.910**	0.907	0.838	**0.999**
DenseNet-121 trained on MC	0.790	0.551	0.620	0.726
DenseNet-121 trained on MC-CH	0.842	0.899	0.649	0.612
DenseNet-121 trained on MC-CH-IN	0.881	0.901	**0.830**	0.717
DenseNet-121 trained on MC-CH-IN-11k	**0.894**	**0.929**	0.820	**0.998**

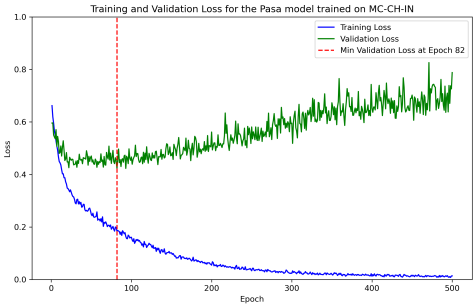
an AUC score of 0.861 on the 11kv2 test set. The indirect model trained on the 11kv2 test set achieves an AUC score of 0.949 on the 11k test set and an AUC score of 0.907 on the 11kv2 test set.

Comparing models trained on different aggregated datasets

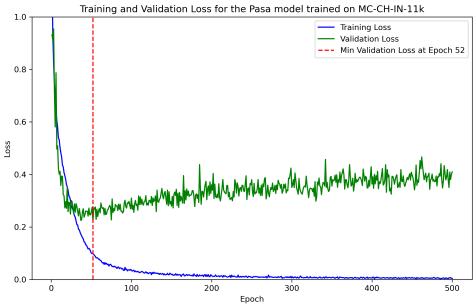
Table 6.7 depicts the obtained AUC scores for *Direct Models without pre-training on Radiological Sign labels* with increasing number of aggregated datasets. Generally, the AUC scores improved as more datasets were aggregated. Adding the fourth dataset did not improve the Pasa models AUC scores for the CH and IN test sets. Looking at the DenseNet-121 model, the AUC scores improved for all test sets except for the CH dataset.

Figure 6.1 shows the training process of the cross-validated Pasa models for MC-CH-IN and for MC-CH-IN-11k. Similarly, Figure 6.2 shows it for the DenseNet-121 models. The minimum loss for the Pasa model trained on MC-CH-IN was obtained at epoch 82, while for the one trained on MC-CH-IN-11k at epoch 52. For the DenseNet-121 models, the minimum loss for the model trained on MC-CH-IN was obtained at epoch 49, and for the one trained on MC-CH-IN-11k at epoch 85.

Table 6.8 depicts the obtained AUC scores for *Indirect Models with Radiological Sign labels* with increasing number of aggregated datasets.

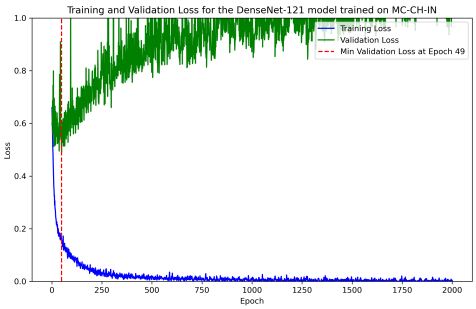


(a) Direct Pasa Model trained on MC-CH-IN

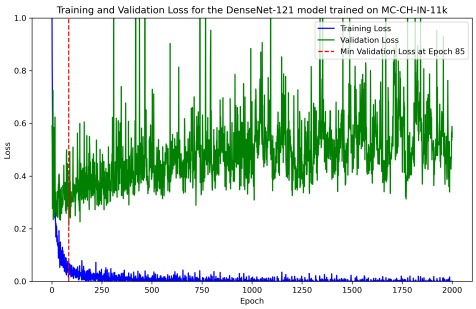


(b) Direct Pasa Model trained on MC-CH-IN-11k

Figure 6.1: DIRECT PASA TRAINING MC-CH-IN VS MC-CH-IN-11K. *These graphs depict the training process of the cross-validated Pasa models, one trained on MC-CH-IN (a) and the other on MC-CH-IN-11k (b). The training and validation loss was calculated based on the mean of the 10 sub-models. The minimum validation loss depicts ultimately the model that was used for the evaluations in this section.*



(a) Direct DenseNet-121 Model trained on MC-CH-IN



(b) Direct DenseNet-121 Model trained on MC-CH-IN-11k

Figure 6.2: DIRECT DENSENET-121 TRAINING MC-CH-IN VS MC-CH-IN-11K. *These graphs depict the training process of the cross-validated DenseNet-121 models, one trained on MC-CH-IN (a) and the other on MC-CH-IN-11k (b). The training and validation loss was calculated based on the mean of the 10 sub-models. The minimum validation loss depicts ultimately the model that was used for the evaluations in this section.*

Table 6.8: This table depicts the AUC scores that were obtained for indirect models trained with an increasing number of datasets through data aggregation. The AUC scores were calculated individually for each test set of the four datasets that were used.

Model	AUC MC Test	AUC CH Test	AUC IN Test	AUC 11k Test
LogReg trained on MC	**0.966**	0.872	0.928	0.890
LogReg trained on MC-CH	0.961	**0.901**	0.927	0.896
LogReg trained on MC-CH-IN	0.950	0.898	0.921	0.895
LogReg trained on MC-CH-IN-11k	0.965	0.885	**0.938**	**0.944**

Comparing if the new models achieved better performance

Compared to the previous Table 6.7, the new Table 6.9 also includes the direct models trained on the MC-CH-IN-11kv2 aggregated dataset. Additionally, the AUC scores for the 11kv2 test splits are reported for each of the models. The reported AUC scores span from 0.497 up to 0.999.

Just as with the direct models, the new Table 6.10 also includes the direct models trained on the MC-CH-IN-11kv2 aggregated dataset. Additionally, the AUC scores for the 11kv2 test splits are reported for each of the models. The reported AUC scores span from 0.839 up to 0.966.

Table 6.11 and Table 6.12 report the optimum thresholds for the models that have been selected based on the highest F1-Score on the validation set of the sets they were trained on. The reported F1-Scores in this table are the ones obtained by using the selected threshold on the test sets of the datasets they were trained on. The highest reported F1-Score for the direct models is 0.91, achieved by the Pasa and DenseNet-121 models that were trained on the MC-CH-IN-11k dataset. For the indirect model, the highest reported F1-Score is 0.90, achieved on the model trained by MC only.

Table 6.9: This table depicts the AUC scores that were obtained for the two types of direct models trained with an increasing number of datasets through data aggregation. Additionally, both versions of the TBX11K splits were used to train different models based on the splits. The AUC scores were calculated individually for each test set of the five data splits that were used.

Model	AUC MC Test	AUC CH Test	AUC IN Test	AUC 11k Test	AUC 11kv2 Test
Pasa trained on MC	0.886	0.615	0.631	0.839	0.684
Pasa trained on MC-CH	0.877	0.902	0.645	0.833	0.689
Pasa trained on MC-CH-IN	0.905	**0.916**	**0.848**	0.824	0.740
Pasa trained on MC-CH-IN-11k	0.910	0.907	0.838	**0.999**	0.905
Pasa trained on MC-CH-IN-11kv2	**0.921**	0.905	0.845	**0.999**	**0.977**
DenseNet-121 trained on MC	0.790	0.551	0.620	0.726	0.625
DenseNet-121 trained on MC-CH	0.842	0.899	0.649	0.612	0.497
DenseNet-121 trained on MC-CH-IN	0.881	0.901	**0.830**	0.717	0.626
DenseNet-121 trained on MC-CH-IN-11k	**0.894**	**0.929**	0.820	**0.998**	0.895
DenseNet-121 trained on MC-CH-IN-11kv2	0.875	0.913	0.763	0.998	**0.983**

Table 6.10: This table depicts the AUC scores that were obtained for indirect models trained with an increasing number of datasets through data aggregation. The AUC scores were calculated individually for each test set of the four datasets that were used.

Model	AUC MC Test	AUC CH Test	AUC IN Test	AUC 11k Test	AUC 11kv2 Test
LogReg trained on MC	**0.966**	0.872	0.928	0.890	0.839
LogReg trained on MC-CH	0.961	**0.901**	0.927	0.896	0.843
LogReg trained on MC-CH-IN	0.950	0.898	0.921	0.895	0.834
LogReg trained on MC-CH-IN-11k	0.965	0.885	**0.938**	**0.944**	0.862
LogReg trained on MC-CH-IN-11kv2	0.964	0.885	0.918	0.928	**0.890**

Table 6.11: This table depicts the obtained mean thresholds of the cross-validated direct models on the respective validation set of the dataset they were trained on. This threshold was used to predict the classes on the respective fold's test sets. The predictions were aggregated from all 10 sub-models before the final F1-Score was calculated for each model.

Model	Threshold	F1-Score
Pasa trained on MC	0.461	0.78
Pasa trained on MC-CH	0.466	0.83
Pasa trained on MC-CH-IN	0.458	0.83
Pasa trained on MC-CH-IN-11k	0.676	0.91
Pasa trained on MC-CH-IN-11kv2	0.864	0.81
DenseNet-121 trained on MC	0.592	0.69
DenseNet-121 trained on MC-CH	0.415	0.81
DenseNet-121 trained on MC-CH-IN	0.315	0.82
DenseNet-121 trained on MC-CH-IN-11k	0.744	0.91
DenseNet-121 trained on MC-CH-IN-11kv2	0.897	0.81

Table 6.12: This table depicts the obtained mean thresholds of the cross-validated indirect models on the respective validation set of the dataset they were trained on. This threshold was used to predict the classes on the respective fold's test sets. The predictions were aggregated from all 10 sub-models before the final F1-Score was calculated for each model.

Model	Threshold	F1-Score
LogReg trained on MC	0.533	0.90
LogReg trained on MC-CH	0.274	0.85
LogReg trained on MC-CH-IN	0.235	0.85
LogReg trained on MC-CH-IN-11k	0.707	0.79
LogReg trained on MC-CH-IN-11kv2	0.912	0.57

Table 6.13: AUC calculated on the first split of the TBX11K test set for direct models (Pasa and DenseNet-121) that have been trained on various dataset combinations. The AUC scores of the three selected models for further analysis are marked as bold.

Training Set	Pasa model AUC on 11k Test	DenseNet-121 model AUC on 11k Test
11k	**1.000**	**1.000**
11kv2	0.997	0.995
MC	0.885	0.900
MC-CH	0.854	0.616
MC-CH-IN	0.771	0.732
MC-CH-IN-11k	0.999	**1.000**
MC-CH-IN-11kv2	0.999	0.995

6.1.1 RQ2 & RQ2.1

The three direct models to be used with the visualisation techniques and for the evaluation with *Proportional Energy*, $AOPC_{Combined}$, and *ROAD-Normalised PropEng Average* have been chosen based on their performance on the test set of the TBX11K dataset. Specifically, the best model was chosen based on the achieved AUC on the test set of the 1st split of the TBX11K dataset For MC, CH, and IN, the data augmentation pipeline depicted in Figure 4.4 was used. For both TBX11K splits, only Elastic Deformation with a probability of 0.8 was used. Although not depicted in, for the first subNIH CXR14 dataset, resizing to 512x512 and random horizontal flips were used. The results can be found in Figure 6.13. The Pasa model trained on 11k achieved a perfect AUC score of 1, together with the two DenseNet-121 models, one trained on 11k, and one trained on MC-CH-IN-11k. In addition, the DenseNet-121 sub-model from the indirect model has been selected as the fourth model. Since no ground truth data exists for calculating the performance of the radiological signs, there is no reported AUC score for this model.

For the 3 selected models, the 12 saliency mapping techniques (11 for Pasa, excluding FullGrad) were applied to obtain their median $AOPC_{Combined}$ and *Proportional Energy*, and the *ROAD-Normalised PropEng Average* for the 11k test set. Table 6.14, Table 6.15, Table 6.16, and Table 6.17 show the metrics for each of the 4 models. The highest score for each metric for each model has been marked in bold.

For the direct Pasa model trained on 11k in Table 6.14, XGrad-CAM and Score-CAM achieved the highest $AOPC_{Combined}$ with 0.125, with XGrad-CAM having a lower Interquartile Range and thus spread according to Q1 and Q3 of the median. The highest Proportional Energy and *ROAD-Normalised PropEng Average* were both obtained for Grad-CAM, with 0.184 and 0.200 respectively.

For the direct DenseNet-121 model trained on 11k in Table 6.15, GradCAMElementWise achieved the highest $AOPC_{Combined}$ with 0.226. The highest Proportional Energy and *ROAD-Normalised PropEng Average* were both obtained for Grad-CAM++, with 0.278 and 0.321 respectively.

For the direct DenseNet-121 model trained on MC-CH-IN-11k in Table 6.16, LayerCAM surpassed GradCAMElementWise's $AOPC_{Combined}$ score by 0.001 with a score of 0.348, and thus obtained the highest $AOPC_{Combined}$ of all the models in this section. The highest Proportional Energy was obtained by Score-CAM with 0.218, and the highest *ROAD-Normalised PropEng Average* was obtained by EigenGradCAM with 0.234.

Table 6.14: This table depicts the median of $AOPC_{Combined}$ and Proportional Energy metrics, and the ROAD-Normalised PropEng Average metric obtained by applying the 11 saliency mapping techniques to the Pasa model trained on 11k.

Visualisation Method	$AOPC_{Combined}$ **((LeRF-MoRF) / 2) Median (Q1, Q3)**	$Proportional Energy$ **Median (Q1, Q3)**	**ROAD-Normalised PropEng Average**
xgradcam	**0.125 (0.073, 0.136)**	0.154 (0.055, 0.201)	0.159
scorecam	**0.125 (0.063, 0.144)**	0.182 (0.081, 0.249)	0.191
ablationcam	0.124 (0.089, 0.134)	0.178 (0.07, 0.25)	0.189
gradcam	0.124 (0.066, 0.132)	**0.184 (0.074, 0.264)**	**0.200**
gradcamplusplus	0.123 (0.054, 0.134)	0.167 (0.069, 0.231)	0.178
hirescam	0.094 (0.0, 0.136)	0.183 (0.077, 0.266)	0.187
randomcam	0.074 (-0.03, 0.134)	0.096 (0.016, 0.144)	0.096
gradcamelementwise	0.026 (-0.001, 0.122)	0.161 (0.071, 0.223)	0.157
layercam	0.016 (-0.002, 0.118)	0.162 (0.071, 0.227)	0.160
eigengradcam	0.013 (-0.035, 0.122)	0.07 (0.015, 0.109)	0.065
eigencam	0.003 (-0.055, 0.098)	0.047 (0.013, 0.067)	0.041

For the first sub-model of the indirect model, the DenseNet-121 model trained on ImageNet and NIH CXR14, LayerCAM again surpassed GradCAMElementWise's $AOPC_{Combined}$ score by 0.001 with a score of 0.203. The overall highest Proportional Energy compared to every model in this section was obtained by HiResCAM with 0.332, and the overall highest *ROAD-Normalised PropEng Average* was obtained by Eigen-CAM with 0.461, making this also the highest score obtained from all the models.

For the qualitative analysis, Figure 6.3 and Figure 6.4 show the top 12 performing visualisations of the 11k test set for the DenseNet-121 model trained on MC-CH-IN-11k. First, based on the $AOPC_{Combined}$ and Proportional Energy metrics, the best performing saliency mapping technique for each metric is selected. Then, for each selected saliency mapping technique, the 12 visualisations that achieved the highest score on the metric they were selected on are presented. The remaining visualisations for the models trained on the first data augmentation pipeline can be found in Section A.2.1.

Table 6.15: This table depicts the median of $AOPC_{Combined}$ and Proportional Energy metrics, and the ROAD-Normalised PropEng Average metric obtained by applying the 12 saliency mapping techniques to the direct DenseNet-121 model trained on 11k.

Visualisation Method	$AOPC_{Combined}$ **((LeRF-MoRF) / 2) Median (Q1, Q3)**	$Proportional Energy$ **Median (Q1, Q3)**	**ROAD-Normalised PropEng Average**
gradcamelementwise	**0.226 (0.11, 0.303)**	0.228 (0.086, 0.323)	0.257
layercam	0.209 (0.167, 0.3)	0.236 (0.086, 0.331)	0.263
hirescam	0.203 (0.118, 0.305)	0.249 (0.088, 0.342)	0.283
eigengradcam	0.202 (0.079, 0.303)	0.189 (0.041, 0.279)	0.238
gradcam	0.194 (0.078, 0.29)	0.265 (0.094, 0.382)	0.307
ablationcam	0.192 (0.078, 0.284)	0.256 (0.086, 0.355)	0.302
gradcamplusplus	0.181 (0.077, 0.273)	**0.278 (0.104, 0.426)**	**0.321**
scorecam	0.178 (-0.045, 0.215)	0.272 (0.089, 0.39)	0.302
xgradcam	0.125 (0.003, 0.25)	0.092 (0.012, 0.096)	0.130
eigencam	0.102 (-0.009, 0.247)	0.183 (0.026, 0.274)	0.242
randomcam	0.077 (-0.149, 0.204)	0.128 (0.019, 0.169)	0.160
fullgrad	-0.116 (-0.125, -0.0)	0.15 (0.067, 0.201)	0.219

Table 6.16: This table depicts the median of $AOPC_{Combined}$ and Proportional Energy metrics, and the ROAD-Normalised PropEng Average metric obtained by applying the 12 saliency mapping techniques to the DenseNet-121 model trained on MC-CH-IN-11k.

Visualisation Method	$AOPC_{Combined}$ **((LeRF-MoRF) / 2) Median (Q1, Q3)**	$ProportionalEnergy$ **Median (Q1, Q3)**	**ROAD-Normalised PropEng Average**
layercam	**0.348 (0.303, 0.386)**	0.191 (0.079, 0.26)	0.184
gradcamelementwise	0.347 (0.303, 0.384)	0.191 (0.079, 0.26)	0.184
gradcamplusplus	0.345 (0.292, 0.377)	0.193 (0.076, 0.262)	0.185
gradcam	0.345 (0.258, 0.397)	0.174 (0.064, 0.243)	0.163
ablationcam	0.336 (0.249, 0.398)	0.166 (0.06, 0.241)	0.159
hirescam	0.335 (0.263, 0.396)	0.176 (0.063, 0.244)	0.166
xgradcam	0.311 (0.22, 0.371)	0.167 (0.05, 0.244)	0.173
fullgrad	0.282 (0.238, 0.354)	0.165 (0.077, 0.215)	0.164
eigengradcam	0.252 (-0.329, 0.358)	0.165 (0.033, 0.267)	**0.234**
eigencam	0.25 (-0.268, 0.357)	0.173 (0.04, 0.243)	0.226
scorecam	0.231 (0.095, 0.312)	**0.218 (0.073, 0.301)**	0.196
randomcam	0.123 (-0.095, 0.286)	0.14 (0.044, 0.196)	0.153

Table 6.17: This table depicts the median of $AOPC_{Combined}$ and Proportional Energy metrics, and the ROAD-Normalised PropEng Average metric obtained by applying the 12 saliency mapping techniques to the first sub-model of the indirect model, namely the DenseNet-121 model trained on ImageNet and NIH CXR14.

Visualisation Method	$AOPC_{Combined}$ **((LeRF-MoRF) / 2) Median (Q1, Q3)**	$ProportionalEnergy$ **Median (Q1, Q3)**	**ROAD-Normalised PropEng Average**
layercam	**0.203 (0.027, 0.399)**	0.23 (0.084, 0.324)	0.302
gradcamelementwise	0.202 (0.027, 0.397)	0.23 (0.084, 0.324)	0.302
gradcamplusplus	0.195 (0.042, 0.4)	0.231 (0.087, 0.329)	0.301
scorecam	0.189 (0.021, 0.373)	0.238 (0.076, 0.327)	0.323
fullgrad	0.178 (0.008, 0.361)	0.216 (0.097, 0.28)	0.283
ablationcam	0.178 (-0.027, 0.4)	0.233 (0.031, 0.381)	0.376
gradcam	0.174 (0.028, 0.365)	0.252 (0.077, 0.38)	0.343
hirescam	0.169 (0.028, 0.386)	**0.322 (0.144, 0.485)**	0.397
eigengradcam	0.12 (-0.003, 0.368)	0.313 (0.062, 0.512)	0.456
xgradcam	0.115 (0.02, 0.315)	0.233 (0.085, 0.362)	0.320
eigencam	0.076 (-0.021, 0.356)	0.285 (0.043, 0.491)	**0.461**
randomcam	-0.022 (-0.108, 0.06)	0.121 (0.008, 0.163)	0.242

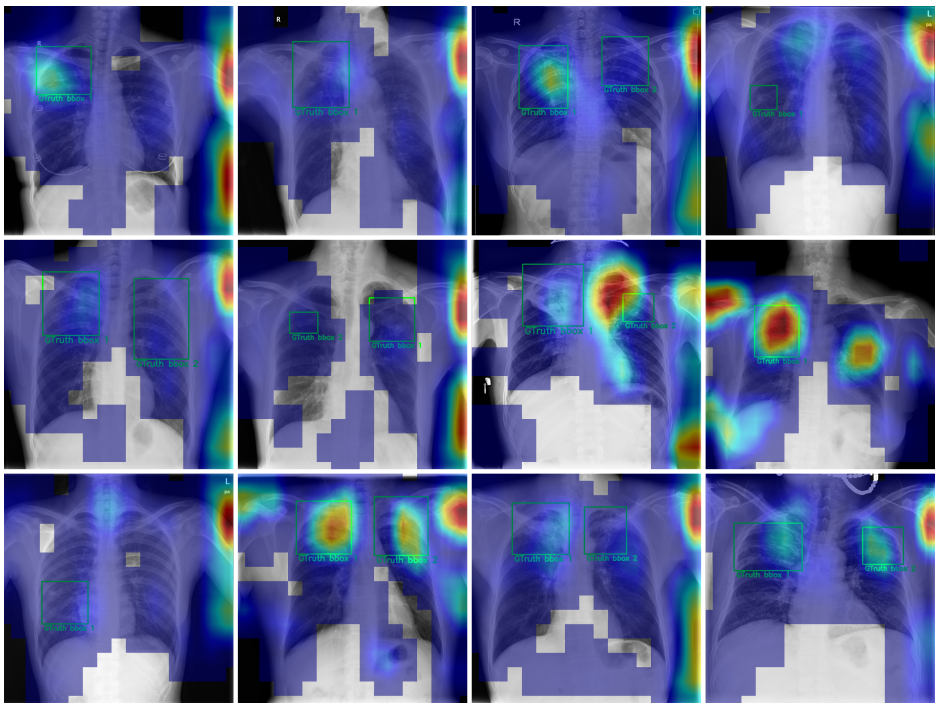


Figure 6.3: DENSENET-121 MC-CH-IN-11K TOP 12 VISUALISATIONS WITH LAYERCAM ACCORDING TO $AOPC_{Combined}$. *The 12 visualisations that achieved the highest score on the* $AOPC_{Combined}$ *are depicted here in a grid, from the highest one on the top left to the twelfth best on the bottom right.*

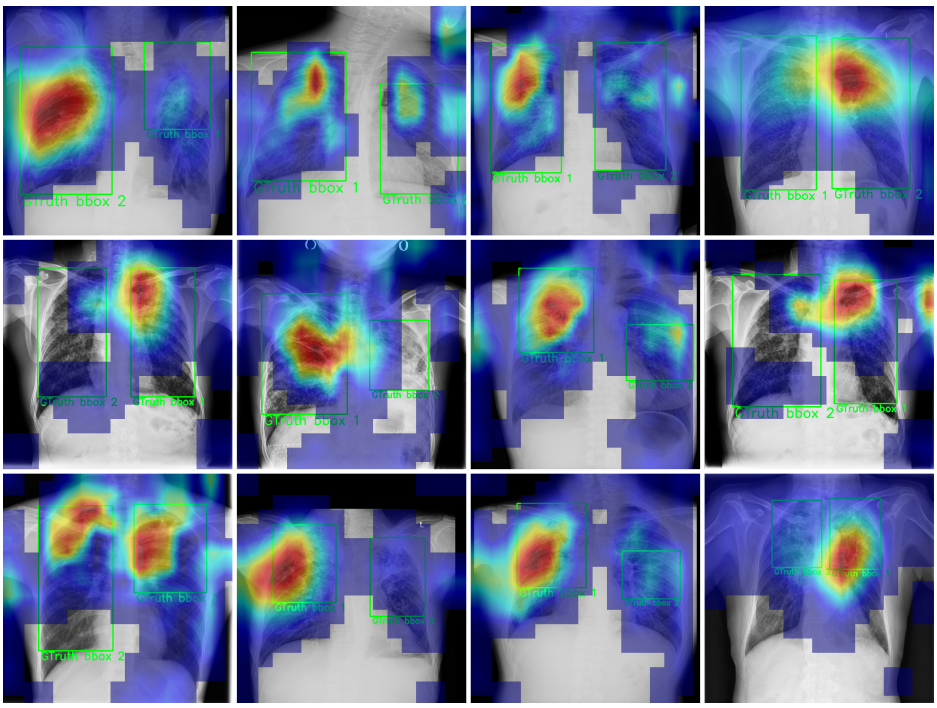


Figure 6.4: DENSENET-121 MC-CH-IN-11K TOP 12 VISUALISATIONS WITH SCORE-CAM ACCORDING TO PROPORTIONAL ENERGY. *The 12 visualisations that achieved the highest score on the Proportional Energy are depicted here in a grid, from the highest one on the top left to the twelfth best on the bottom right.*

Table 6.18: AUC calculated on the first split of the TBX11K test set for selected direct models (Pasa and DenseNet-121) that have been trained on various dataset combinations. These models had a different data augmentation pipeline during training than the ones listed in Table 6.13.

Model Name	AUC on 11k Test
Pasa trained on 11k	0.999
DenseNet-121 trained on 11k	1.000
DenseNet-121 trained on MC-CH-IN-11k	0.999

Further, the AUC scores on the 11k test set of the three selected models that have been trained with the updated data augmentation pipeline from Figure 5.1 are listed in Figure 6.18. The random resized crop was only used for the TBX11K dataset, while the other additional data augmentations were used for MC, CH, IN, and TBX11K. The AUC scores of these models almost stayed the same as in Table 6.13, with the only difference being the Pasa trained on 11k and DenseNet-121 model trained on MC-CH-IN-11k lowering by 0.001 to an AUC on 0.999.

Similarly, for the 3 selected models that have been trained with the new data augmentation pipeline, the 12 saliency mapping techniques (11 for Pasa, excluding FullGrad) were applied to obtain their median $AOPC_{Combined}$ and *Proportional Energy*, and the *ROAD-Normalised PropEng Average* for the 11k test set. Table 6.19, Table 6.20, and Table 6.21 show the metrics for each of the 3 models. The highest score for each metric for each model has been marked in bold.

For the direct Pasa model trained on 11k with the new data augmentation pipeline in Table 6.19, LayerCAM achieved the highest $AOPC_{Combined}$ with 0.102. The highest Proportional Energy was obtained for Grad-CAM++ with 0.181, and the highest obtained *ROAD-Normalised PropEng Average* was 0.184 for Grad-CAM.

For the direct DenseNet-121 model trained on 11k the new data augmentation pipeline in Table 6.20, LayerCAM achieved the highest $AOPC_{Combined}$ with 0.278. The highest Proportional Energy was obtained by Grad-CAM with 0.278, and for *ROAD-Normalised PropEng Average* it was EigenGradCAM, with 0.299.

For the direct DenseNet-121 model trained on MC-CH-IN-11k the new data augmentation pipeline in Table 6.21, XGrad-CAM reached an $AOPC_{Combined}$ score of 0.255. The highest Proportional Energy was obtained by EigenGradCAM with 0.229, and the highest *ROAD-Normalised PropEng Average* was obtained by Eigen-CAM with 0.227.

For the qualitative analysis of the DenseNet-121 model trained on MC-CH-IN-11k that was trained with new data augmentations, Figure 6.5 and Figure 6.6 show the top 12 performing visualisations of the 11k test set. Just as previously, they were selected based on the $AOPC_{Combined}$ and Proportional Energy metrics, however, they were not selected based on the best performing saliency mapping technique with the new data augmentations, but according to the best performing saliency mapping technique with the old ones. This allows for a comparison of the visualisations before and after adding the data augmentations. For each selected saliency mapping technique, the 12 visualisations that achieved the highest score on the metric they were selected on are presented. The remaining visualisations for the models trained on the second data augmentation pipeline can be found in Section A.2.2.

Table 6.19: This table depicts the median of $AOPC_{Combined}$ and Proportional Energy metrics, and the ROAD-Normalised PropEng Average metric obtained by applying the 11 saliency mapping techniques to the Pasa model trained on the 11k by using more data augmentations.

Visualisation Method	$AOPC_{Combined}$ **((LeRF-MoRF) / 2) Median (Q1, Q3)**	$ProportionalEnergy$ **Median (Q1, Q3)**	**ROAD-Normalised PropEng Average**
layercam	**0.102 (0.018, 0.175)**	0.166 (0.077, 0.237)	0.154
hirescam	0.1 (0.049, 0.17)	0.168 (0.062, 0.237)	0.163
gradcamelementwise	0.1 (0.018, 0.173)	0.165 (0.077, 0.237)	0.154
gradcam	0.092 (0.054, 0.15)	0.177 (0.059, 0.264)	**0.184**
gradcamplusplus	0.092 (0.049, 0.157)	**0.181 (0.072, 0.266)**	0.173
ablationcam	0.092 (0.049, 0.144)	0.174 (0.062, 0.261)	0.178
scorecam	0.089 (0.047, 0.167)	0.176 (0.065, 0.265)	0.166
eigengradcam	0.038 (-0.027, 0.086)	0.093 (0.016, 0.13)	0.119
randomcam	0.024 (-0.106, 0.092)	0.105 (0.02, 0.155)	0.132
eigencam	-0.182 (-0.322, -0.046)	0.05 (0.009, 0.068)	0.082
xgradcam	-0.003 (-0.158, 0.049)	0.101 (0.019, 0.149)	0.155

Table 6.20: This table depicts the median of $AOPC_{Combined}$ and Proportional Energy metrics, and the ROAD-Normalised PropEng Average metric obtained by applying the 12 saliency mapping techniques to the direct DenseNet-121 model trained on 11k by using more data augmentations.

Visualisation Method	$AOPC_{Combined}$ **((LeRF-MoRF) / 2) Median (Q1, Q3)**	$ProportionalEnergy$ **Median (Q1, Q3)**	**ROAD-Normalised PropEng Average**
layercam	**0.278 (0.181, 0.373)**	0.262 (0.115, 0.389)	0.237
gradcamelementwise	0.277 (0.18, 0.373)	0.262 (0.115, 0.389)	0.237
gradcamplusplus	0.247 (0.113, 0.293)	0.265 (0.117, 0.399)	0.232
hirescam	0.244 (0.118, 0.25)	0.274 (0.121, 0.423)	0.245
xgradcam	0.238 (0.12, 0.25)	0.269 (0.095, 0.416)	0.244
ablationcam	0.232 (0.102, 0.251)	0.27 (0.11, 0.419)	0.247
gradcam	0.223 (0.124, 0.25)	**0.278 (0.117, 0.429)**	0.251
scorecam	0.205 (0.056, 0.261)	0.247 (0.088, 0.382)	0.223
eigencam	0.097 (0.0, 0.125)	0.256 (0.053, 0.408)	0.263
eigengradcam	0.015 (-0.034, 0.122)	0.247 (0.026, 0.418)	**0.299**
randomcam	-0.052 (-0.306, 0.136)	0.131 (0.009, 0.184)	0.226
fullgrad	-0.005 (-0.115, 0.018)	0.18 (0.083, 0.232)	0.209

Table 6.21: This table depicts the median of $AOPC_{Combined}$ and Proportional Energy metrics, and the ROAD-Normalised PropEng Average metric obtained by applying the 12 saliency mapping techniques to the DenseNet-121 model trained on MC-CH-IN-11k by using more data augmentations.

Visualisation Method	$AOPC_{Combined}$ **((LeRF-MoRF) / 2) Median (Q1, Q3)**	$ProportionalEnergy$ **Median (Q1, Q3)**	**ROAD-Normalised PropEng Average**
xgradcam	**0.255 (0.151, 0.369)**	0.197 (0.078, 0.259)	0.177
hirescam	0.251 (0.159, 0.366)	0.204 (0.087, 0.275)	0.185
gradcam	0.251 (0.154, 0.371)	0.195 (0.082, 0.257)	0.175
ablationcam	0.251 (0.14, 0.364)	0.195 (0.082, 0.269)	0.176
eigengradcam	0.229 (0.215, 0.317)	**0.229 (0.088, 0.316)**	0.226
layercam	0.227 (0.12, 0.326)	0.202 (0.089, 0.275)	0.188
gradcamelementwise	0.227 (0.12, 0.32)	0.202 (0.089, 0.275)	0.188
eigencam	0.219 (0.209, 0.306)	0.226 (0.084, 0.311)	**0.227**
gradcamplusplus	0.209 (0.111, 0.327)	0.194 (0.085, 0.26)	0.175
fullgrad	0.149 (0.095, 0.22)	0.156 (0.076, 0.198)	0.150
scorecam	0.12 (0.017, 0.235)	0.179 (0.075, 0.24)	0.180
randomcam	0.103 (-0.029, 0.198)	0.111 (0.007, 0.165)	0.134

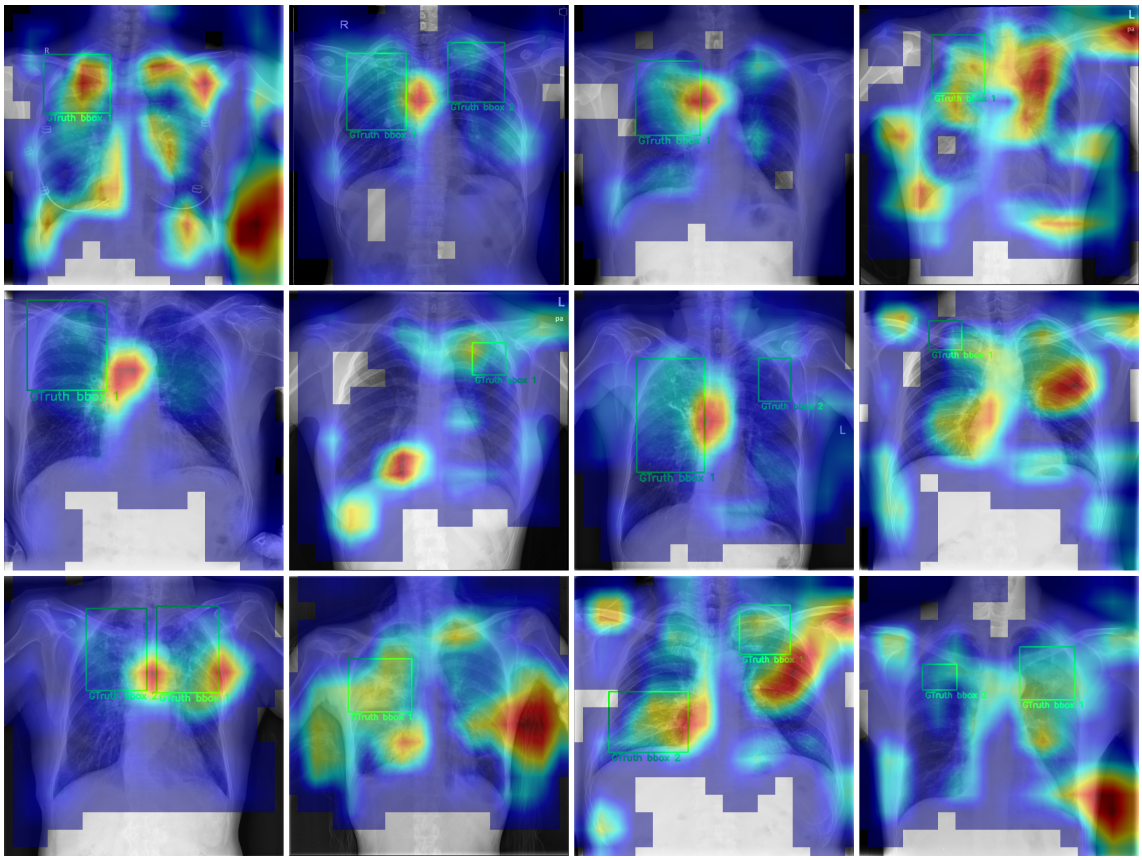


Figure 6.5: DENSENET-121 MC-CH-IN-11K WITH SECOND DATA AUGMENTATION PIPELINE TOP 12 VISUALISATIONS WITH LAYERCAM ACCORDING TO $AOPC_{Combined}$. *The 12 visualisations that achieved the highest score on the $AOPC_{Combined}$ are depicted here in a grid, from the highest one on the top left to the twelfth best on the bottom right.*

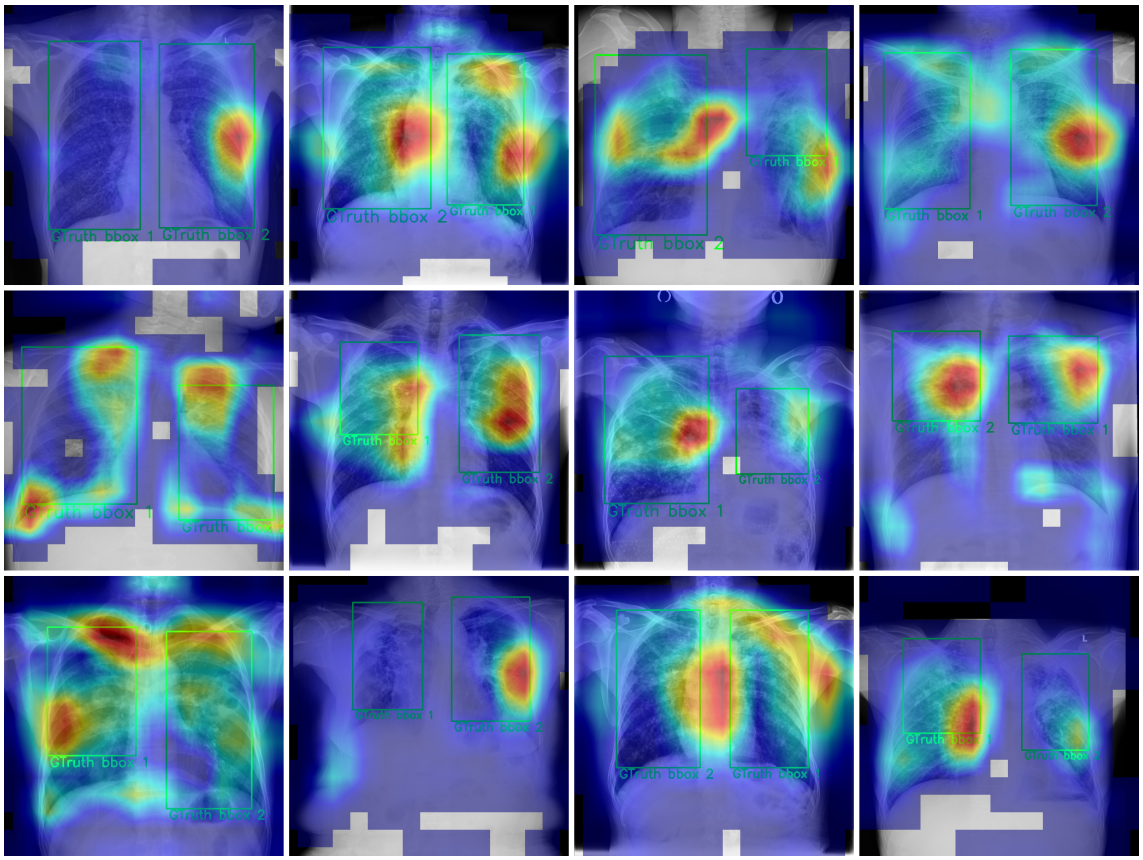


Figure 6.6: DENSENET-121 MC-CH-IN-11K WITH SECOND DATA AUGMENTATION PIPELINE TOP 12 VISUALISATIONS WITH SCORE-CAM ACCORDING TO PROPORTIONAL ENERGY. *The 12 visualisations that achieved the highest score on the Proportional Energy are depicted here in a grid, from the highest one on the top left to the twelfth best on the bottom right.*

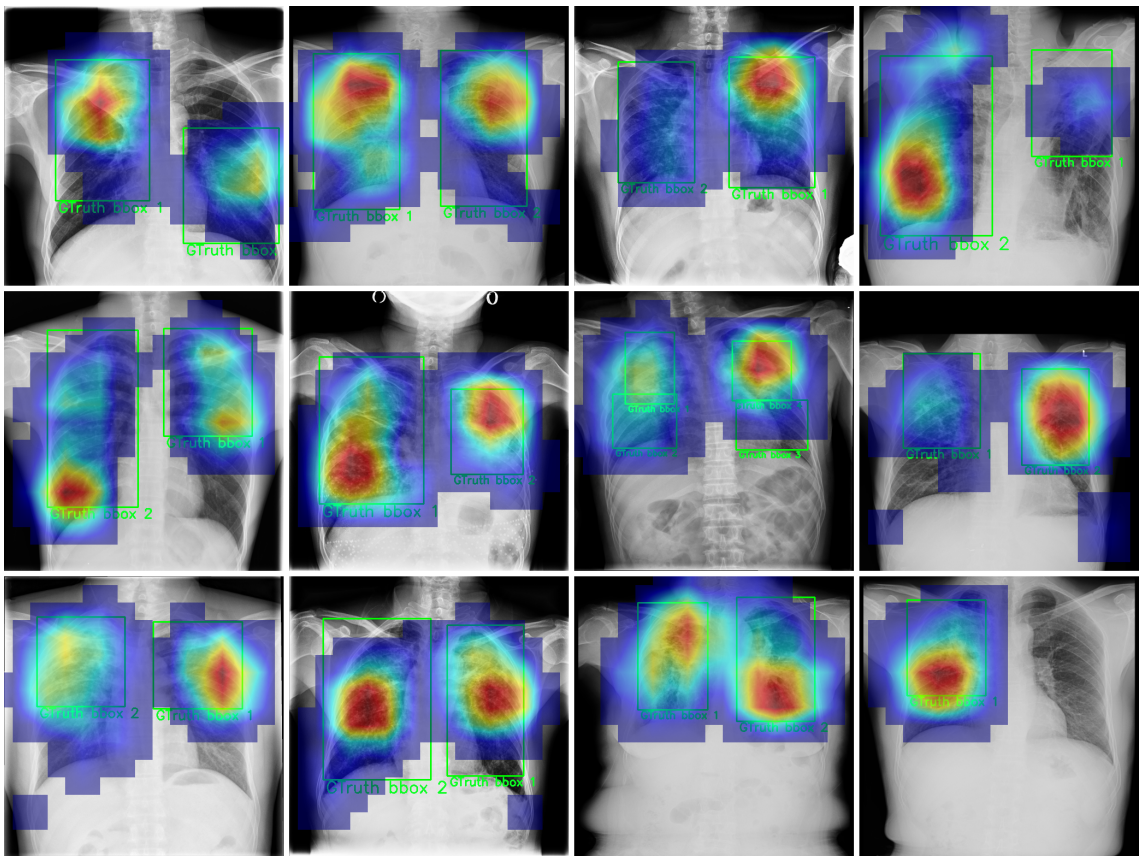


Figure 6.7: INDIRECT DENSENET-121 SUB-MODEL TOP 12 VISUALISATIONS WITH EIGEN-CAM ACCORDING TO ROAD-NORMALISED PROPENG AVERAGE. *The 12 visualisations that achieved the highest score on the $AOPC_{Combined}$ are depicted here in a grid, from the highest one on the top left to the twelfth best on the bottom right.*

Last but not least, the overall highest ROAD-Normalised PropEng Average score was obtained for the DenseNet-121 sub-model of the indirect model by using Eigen-CAM for visualisations. The final images for the qualitative analysis were obtained by selecting the best and worst 12 visualisations for this metric. Additionally, 12 randomly selected healthy images are being shown in Figure 6.9 that were obtained by the same model and saliency mapping technique combination.

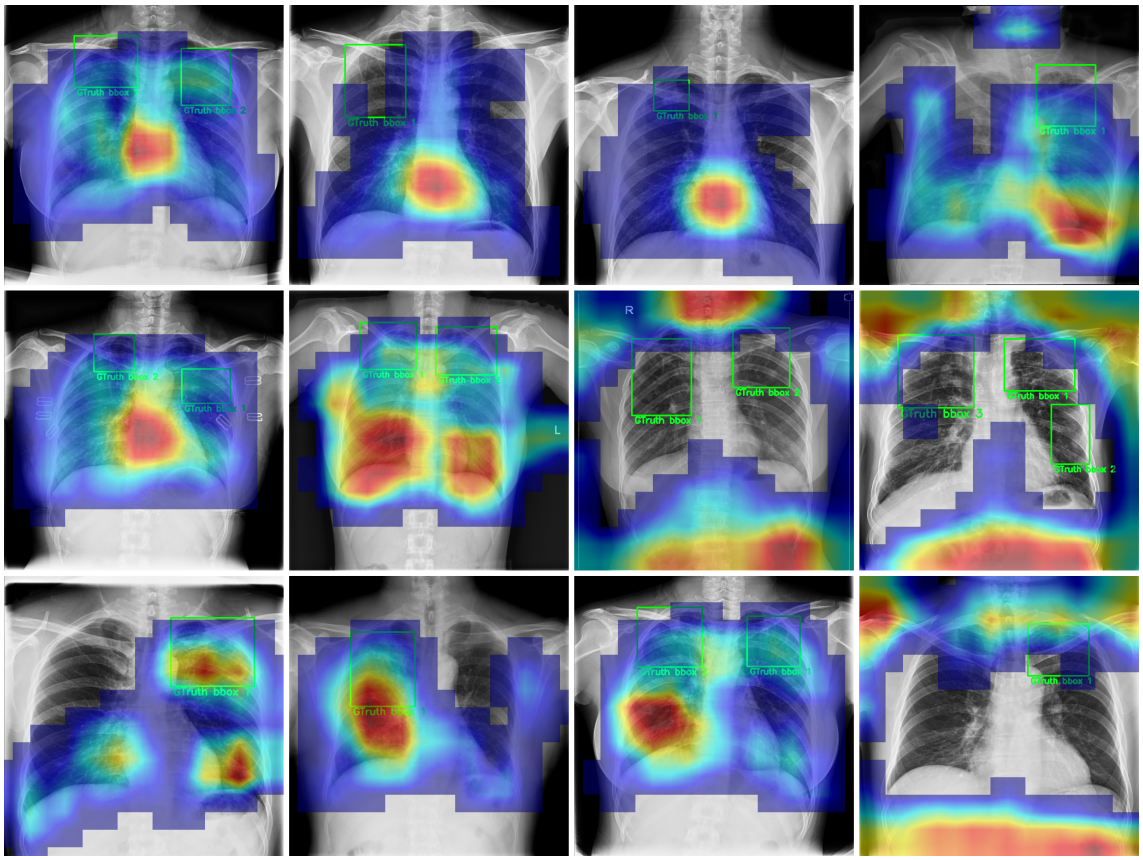


Figure 6.8: INDIRECT DENSENET-121 SUB-MODEL WORST 12 VISUALISATIONS WITH EIGEN-CAM ACCORDING TO ROAD-NORMALISED PROPENG AVERAGE. *The 12 visualisations that achieved the highest score on the Proportional Energy are depicted here in a grid, from the highest one on the top left to the twelfth best on the bottom right.*

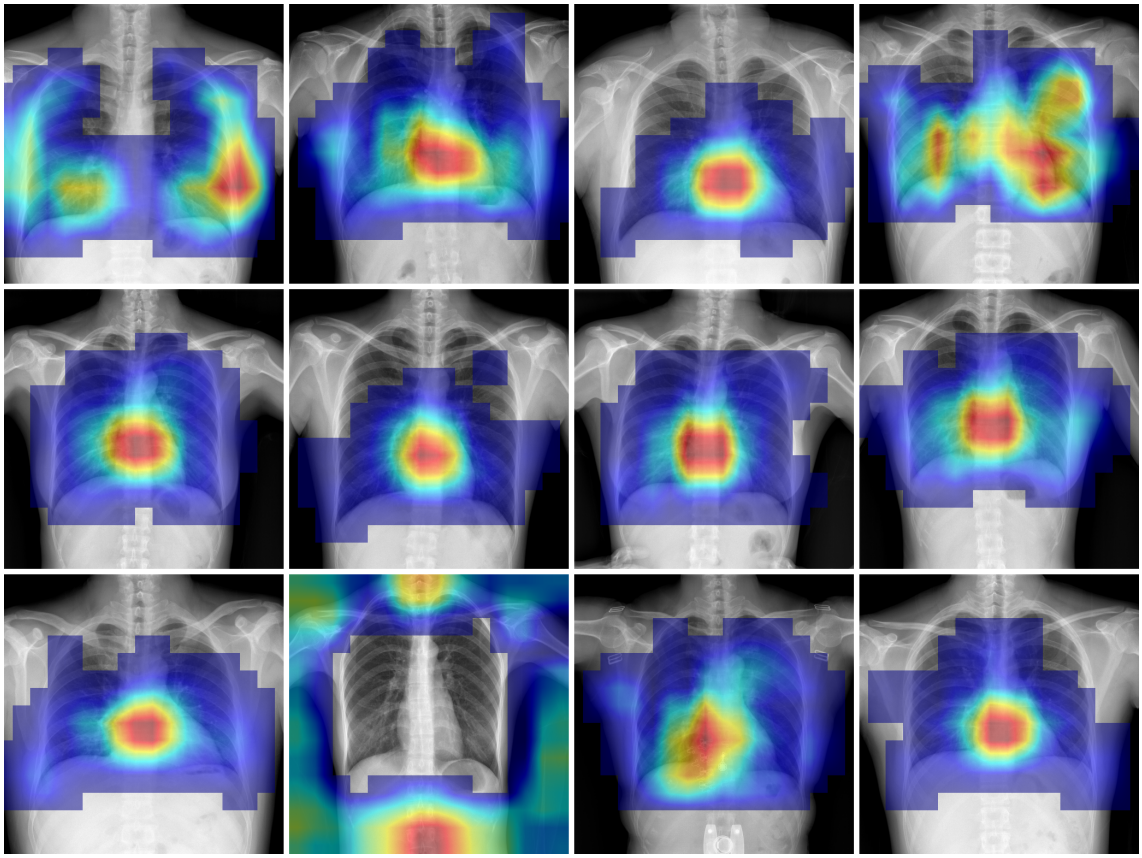


Figure 6.9: INDIRECT DENSENET-121 SUB-MODEL RANDOMLY SELECTED 12 VISUALISATIONS OF HEALTHY SAMPLES WITH EIGEN-CAM. *The 12 visualisations are randomly selected healthy samples.*

Chapter 7

Discussion

The results from Section 6 are first interpreted and discussed in Section 7.1, and the potential limitations of this thesis and future work are mentioned in Section 7.3. Here, again, all the research questions and their sub-questions that the discussion is based around:

- **RQ 1**: Can the prediction of the probability of active Tuberculosis by direct and indirect deep learning models be improved through the use of a new dataset specific to active Tuberculosis?
  - **RQ 1.1**: How do the AUC scores from Raposo (2021) compare to the replicated models using the same methods and frameworks?
  - **RQ 1.2**: Does the inclusion of different types of labels in the TBX11K dataset (healthy, latent TB, sick & non-TB) affect the model's ability to discriminate active TB cases?
  - **RQ 1.3**: Does the inclusion of the TBX11K dataset during training affect the generalisability of the models?
  - **RQ 1.4**: How do the AUC scores of the models that included the TBX11K dataset during training compare to the replicated models based on Raposo (2021)?
- **RQ 2**: What novel visualisation methods exist for increasing the explainability of deep learning models trained on a dataset specific to active Tuberculosis by visualising radiological signs and how well do they perform?
  - **RQ 2.1**: What measurement techniques are suited to evaluate the performance of the visualisations?

7.1 Results Interpretation and Discussion

7.1.1 RQ 1.1

For the *Direct Models without Radiological Sign labels*, assuming the null-hypothesis is that there is no difference between all the AUC scores between the original and replicated results, and assuming a significance level for the p-value of 0.05, the Wilcoxon signed-rank test was applied on all the 18 AUC scores by taking the difference between the replicated and original scores. If the p-value equals or is above 0.05, the null hypothesis is not rejected, if it is below 0.05, the null hypothesis is rejected and it means there is a statistically significant difference between the original and replicated results.

The Wilcoxon signed-rank test resulted in a p-value of 0.3, which means replicated AUC scores lie within the expected and there is no reason to reject the null hypothesis. The difference between the replicated and original AUC scores was not to be found statistically significant ($p > 0.05$). The absolute AUC score difference of up to 0.056 is likely due to the random variability during the training process with PyTorch.

Raposo (2021) noted that the DenseNet-121 results are slightly better than the Pasa results, however, with the replicated results, it looks like that the Pasa model performs better in every configuration (only 1 exception with the IN test set for DenseNet-121 trained on MC-CH). Considering the original and replicated results, it is not possible to infer that one direct model type performs better than the other.

For *Indirect Models with Radiological Sign labels*, the absolute AUC score deviations of the replicated results from the results in the thesis of Raposo (2021) were reported to be between 0 and 0.005, making the replicated model's results even more consistent with those reported in the thesis of Raposo (2021).

Similarly, doing the Wilcoxon signed-rank test for the indirect models results in a p-value of 0.17, which means the null hypothesis is not rejected. The difference between the replicated and original AUC scores was not to be found statistically significant ($p > 0.05$).

These results suggest that it is possible to replicate the results aTB detection models by using the *ptbench* package.

7.1.2 RQ 1.2

The direct models trained on the 11kv2 split consistently perform better on the 11kv2 split test set, while also having almost the same AUC score for the 11k split (0.999 vs. 1). The direct models trained on the 11k split however, while performing well on their own splits test set, do not perform well on the 11kv2 test sets. At first, this would indicate that a model trained on a dataset with not only aTB and *healthy* cases, but also with *latent TB* and *sick & non-TB* is preferable. Looking at the performance of these models on the other three datasets, it seems like adding the ability to distinguish aTB from *latent TB* and *sick & non-TB* does not necessarily increase or decrease generalisability and performance for the simpler task to distinguish *healthy* cases from *aTB* cases. The Pasa model trained on 11kv2 performs worse for the MC and CH datasets, while performing better for the IN dataset, and the DenseNet-121 Model trained on 11kv2 performs worse for the MC dataset, while performing better for the CH and IN datasets. The Pasa Model's ability to capture the nuances of the other types of labels may be restricted due to its size, which could explain why the Pasa model achieves a lower AUC score on 2 datasets and the DenseNet-121 only on one. Unfortunately, from these results, it is not possible to recommend to include or exclude *latent TB* and *sick & non-TB* from the negative class during training for direct models, since there is no clear trend that this would improve or hurt the generalisability of the models. It should also be further added that the AUC on the three other test sets go as low as 0.501, making the models not better than a model that would randomly guess the class. This further lowers the significance of these obtained results.

Similarly, for the indirect models, the AUC scores are better for the split the models were trained on and worse on the other split's test set. Looking at how well they generalise to the other three datasets, the AUC scores the indirect model trained on 11kv2 obtained for every other dataset's test set are lower than the ones the model trained on 11k has obtained. Training the indirect model with *latent TB* and *sick & non-TB* could have led to a worse generalisability on the simpler task of discriminating *healthy* cases from *aTB* cases. This could indicate that this type of model architecture is not suitable to discriminate *aTB* from more than only *healthy* cases, presumably due to the model's underfitting. It should be noted that, contrary to the direct models, the ability to generalise well to other test sets could very likely be attributed to the first DenseNet-

121 sub-model of the indirect model.

7.1.3 RQ 1.3

Generally, there is a trend of the AUC scores improving as more datasets are aggregated for the direct models. For the Pasa model, it seems like there are diminishing results with more data, potentially tied to its simpler architecture (underfitting) compared to DenseNet-121. Other reasons why this could have happened are that the fourth dataset included too much noise or irrelevant patterns, or it could be that the fourth dataset might not be representative of the broader population, or the way CXR are taken. The TBX11K dataset is also larger than the previous three datasets, and there were slightly fewer data augmentations applied to them during training. Looking at how well the models perform on the 11k test set after adding the dataset, this could suggest that this leads to overfitting or bias, especially in the Pasa model, since the training process might have been dominated by the patterns from the TBX11K dataset, even though class weights were applied during training. If these models over- or underfit can be also evaluated by looking at the training process. Observing the training process between the direct models trained on MC-CH-IN vs. MC-CH-IN-11k, the validation loss clearly decreases with the addition of the TBX11K dataset, for both, the Pasa and the DenseNet-121 model. This indicates that the generalisation of both model types improved with the addition of TBX11K, even though the AUC scores were not clearly indicative of it. Since the lowest validation losses were reached quickly, it is expected that more data could still further improve the performance of these models. The decrease of the observed AUC scores in some instances might be attributable to the imbalanced nature of the TBX11K dataset. Potentially, the class weighting during the training process was not enough, or not the right method to compensate for the imbalance. Thus, this could have led the models to overfit the majority class, which might have been responsible for the worse generalisability of the Pasa model. Further, the hyperparameters from the work of Raposo (2021) might not be suitable after changing the composition of the dataset.

The general performance of the indirect models is high, even when training the logistic regression sub-model with only one TB-specific dataset. Data aggregation with TB-related datasets generally seems to have no benefit on the logistic regression sub-model.

The first sub-model was the same for each of the indirect models listed in Table 6.8. That first DenseNet-121 sub-model was trained on the NIH CXR14 dataset. Since the NIH CXR14 dataset is very large (more than 100,000 cases) compared to the largest TB-specific dataset (4430 cases), the DenseNet-121 seems to be more determinative to the final AUC score than the logistic regression sub-models. The logistic regression sub-models might be either too simple to capture the additional patterns to generalise, or the 14 radiological signs might be too limiting to make further generalisations.

7.1.4 RQ 1.4

Compared to Section 7.1.2, these AUC scores are high, so these results are more significant than the models trained only on the 11k or 11kv2 datasets. Notably, when including the 11kv2 split in the aggregated datasets, the Pasa model generalisability seems to have overall improved, while DenseNet-121 generalisability has decreased. Even though the Pasa model with the MC-CH-IN-11kv2 dataset performs slightly worse on CH and IN, it achieves the best performance on 3 of the test sets, although, the high AUC scores on the 11k and 11kv2 test sets, as stated before, are potentially attributable to the overall imbalance of the TBX11K dataset, thus overfitting on these two test sets. The DenseNet-121 model performing worse when trained with the MC-CH-IN-11kv2 split generally seems to indicate that the inclusion of the *latent TB* and *sick & non-TB* labels hurt the performance of this direct model type, although it is unclear why this is the case.

Potentially, the DenseNet-121 model is capable of learning more noise and patterns than the Pasa model that are irrelevant for the other datasets that only include 2 labels, lowering the overall capability of this model to discriminate only between *healthy* and *aTB*.

Overall, the inclusion of the TBX11K dataset has improved the AUC scores of both direct model types, while also keeping their generalisability. The addition of the new *latent TB* and *sick & non-TB* labels allows the models to further differentiate between more nuanced cases, although for the DenseNet-121 model, it comes with the cost of having lower generalisability on the simpler task to discriminate between *healthy* and *aTB* cases. Both models achieve the highest F1-Scores of 0.91 when they are trained on the MC-CH-IN-11 dataset.

As for the indirect models, there is not much to add than in Section 7.1.3. The general performance is tied to the first sub-model, which is the same for all these indirect models. Just as observed with the direct models, the logistic regression sub-model is not able to discriminate the *latent TB* and *sick & non-TB* from the *aTB* cases as well when they were not included in the dataset (judging from the AUC scores on the 11kv2 test set), although they are still able to do it rather well when they were not included, especially when compared with the direct models.

Overall for the indirect models, the new models trained with more TB-related data do not achieve a higher AUC score than they did before, as the F1-Score of 0.90 for the model trained on MC confirms it, too. Judging from the AUC scores for 11kv2 test sets, the inclusion of 11kv2 only slightly benefits the ability of the indirect model to additional discriminate *latent TB* and *sick & non-TB* from *aTB*.

7.1.5 RQ 1

The primary focus of this research question was to investigate whether the prediction of the probability of aTB through direct and indirect models could be improved using a new aTB-related dataset. For RQ 1.1, the AUC scores for direct and indirect models did not show a statistically significant difference when comparing the replicated models to those presented by Raposo (2021). This served as a foundation upon which the new dataset's influence can be measured. Concerning RQ 1.2, the inclusion of multiple label types like *latent TB* and *sick & non-TB* in the training set did not definitively improve or impair the model's discriminatory capability. While direct models trained on a dataset with these additional labels seemed to perform slightly better on that specific dataset, the generalisability across other datasets remained inconclusive. Indirect models, on the other hand, demonstrated decreased performance when trained on the more complex labelling scheme, raising questions about their suitability for more nuanced classifications. In terms of RQ 1.3, data aggregation with the TBX11K dataset appeared to show an improvement in generalisability for the direct models but did not offer any discernible advantage for the indirect models, which seemed to be heavily influenced by the first sub-model (DenseNet-121). Finally, for RQ 1.4, the inclusion of the TBX11K dataset improved the AUC scores for both direct and indirect models. For direct models, the Pasa model benefits in terms of overall generalisability when the 11kv2 split is included in the training dataset. However, the DenseNet-121 model sees a decline in its generalisability, particularly on simpler tasks like discriminating between *healthy* and *aTB* cases. The indirect models do not show a marked improvement in AUC scores with the inclusion of additional TB-related data, although there is a slight benefit in their ability to discriminate new categories like *latent TB* and *sick & non-TB* from *aTB*.

In conclusion, while the introduction of the new dataset may offer some advantages, especially for the direct models, attention must be given to the specific architecture employed and the training process. The findings indicate that further research is warranted to optimise these models, particularly in the domain of generalisability across diverse datasets with more than just binary labels.

7.2 RQ 2

First, when looking at the AUC scores of the first batch of models that were trained with fewer data augmentations, it is noteworthy that the Pasa models trained on MC-CH-IN-11k and MC-CH-IN-11kv2 could also be good candidates for further analysis, too, since they have achieved an AUC score of 0.999, but to avoid overcomplicating the (qualitative) analysis, only the three models with a perfect AUC have been ultimately selected. All the selected three models, the Pasa model trained on 11k, and the two DenseNet-121 models, one trained on 11k, and one trained on MC- CH-IN-11k, had a perfect AUC score of 1. The first DenseNet-121 sub-model is also part of that comparison.

The intention of $AOPC_{Combined}$ was to find the best performing saliency mapping technique for visualisation while minimising the influence of the models on the outcome as much as possible. Looking from Table 6.14 to Table 6.17, for the three DenseNet-121 models, LayerCAM and GradCAMElementWise consistently obtained the highest scores, with the highest being 0.348 by LayerCAM for the DenseNet-121 model trained on MC- CH-IN-11k, suggesting that LayerCAM is the best at being faithful to the model's decisions. Looking at the Pasa model, however, the $AOPC_{Combined}$ is generally lower for all the saliency mapping techniques, and the best performing ones are Score-CAM and XGrad-CAM. These results suggest that while $AOPC_{Combined}$ can be used to find the best performing saliency mapping technique for a specific model, it is not possible to generalise this finding to other model architectures.

Comparing the Proportional Energy of these models, the DenseNet-121 models seem to achieve the highest one, with the indirect multi-label sub-model reaching a 0.322 with HiResCAM in Table 6.17, and with the regular direct binary classifier DenseNet-121 model trained on 11k reaching 0.278 with GradCAM++. Again, it seems to be more difficult for the Pasa model to localise the relevant radiological signs with saliency mapping techniques, as they generally score lower on the Proportional Energy. Overall, HiResCAM combined with the DenseNet-121 model trained on ImageNet and NIH CXR14 seems to be the best at localising radiological signs related to aTB, even though it was not specifically trained with a dataset consisting of radiological signs only related to TB.

As for the ROAD-Normalised PropEng Average, again, the indirect sub-model achieved the highest score in combination with Eigen-CAM in Table 6.17, suggesting that using this combination results in the best trade-off between having faithful explanations versus correct results (according to the ground truth labels). There could be various reasons why this was the best combination. One could be since the ground truth labels are based on radiological signs, and since the indirect model has been trained with a dataset related to radiological signs, this model seems to be able to localise the relevant aTB-related radiological signs the best. Further, one could assume since this is a multi-label model, and Eigen-CAM is a non-class-discriminative saliency mapping technique, that the reason this model performs well is due to the interaction between the model and the visualisation technique. However, looking at Table 6.17, the class discriminative version EigenGradCAM performs only 0.005 worse, suggesting that this is not the only reason this method works well with this model. Further, even though this was the best combination looking at the Q1 and Q3 of $AOPC_{Combined}$ and Proportional Energy, there seem to be very high scoring samples, and it is possible that ROAD-Normalised PropEng Average was skewed due to such samples.

Qualitative Analysis - Models with First Data Augmentation Pipeline Continuing with the qualitative analysis, the focus first lies on the first batch of trained models from Figure A.1 to Figure A.6, and with a special focus on Figure 6.3. For each model, the first batch of visualisations was chosen on the highest $AOPC_{Combined}$, meaning these images will be the most faithful to the model's decision-making. All these models had an AUC of 1, so every CXR was classified

correctly. From this, we can derive the following: When looking at the visualisations with a high $AOPC_{Combined}$, and they do not match with the depicted ground truth labels, then the model made its decisions either on patterns we humans cannot understand or have not observed, or the model is biased and got lucky with the classifications. The second batch of images for each model is based on the highest obtained Proportional Energy, the interpretation of these are more straightforward. The visualisations are expected to be within the ground-truth boxes for these visualisations. Starting with Figure A.1, generally, there seem to be sometimes biases towards the lower left corner of the CXR, and the neck area on which the model bases its decisions on. Looking also further at Figure A.2, visualisations that scored high on Proportional Energy, the saliency mapping techniques combined with this model seem to generally have trouble localising the radiological signs. The decisions this model makes when classifying a CXR as aTB or not seem to be more holistic, but since the visualisations are not faithful in the first place, the significance of this insight is low. Continuing with DenseNet-121 trained on 11k, the visualisations seem to be generally accurate. In Figure A.3, the model sometimes looks at seemingly irrelevant parts, like in the bottom left pictures, looking at the black area of the CXR that contains no relevant information, or also sometimes concentrating below or above the lungs on the right side of the CXR. The visualisations in Figure A.4 also mostly concentrate on the lungs and generally the visualisations match with the radiologist's findings, but sometimes the model focuses on the shoulder above the lungs on the right side of the CXR. The visualisations of the indirect sub-model (DenseNet-121 trained on ImageNet and NIH CXR14) look the most impressive. Since depicts Figure A.5 the most faithful explanations, the model seems to often consider the entire CXR when making its decision, even the black parts. Still, the highest activations in all of the samples are within the lungs and generally match well with the radiologist's findings, with one exception focusing on the neck and armpit. With Figure A.6, the best visualisations according to the Proportional Energy seem very aligned with the radiologist's findings. HiResCAM seems to be indeed able to localise the radiological signs well. Last but not least, the DenseNet-121 trained on MC-CH-IN-11k was kept as the last model for the qualitative analysis, since the results were surprising. Looking at Figure 6.3, almost all the visualisations focus on the right part of the CXR that does not contain any information, suggesting that this model is heavily biased. As mentioned, we can derive this because the model is a perfect classifier and bases its decisions on the visualisations shown in this figure. So we can derive either that the model looks at patterns humans do not understand, or the model is biased. Since the black parts do not contain information, it must be biased. Figure 6.4 does not show such behaviour, and as expected, the visualisations with a high Proportional Energy match the ground truth labels, although there are small biases sometimes similar to the previous cases.

From this analysis, it seems like all three selected direct models contain some sort of bias, especially the DenseNet-121 model trained on MC-CH-IN-11k. It was decided to retrain the three direct models with the mentioned second type of data augmentation pipeline in Section 5.2.3, in an attempt to remove the biases, especially the right side bias with the random horizontal flips.

After the retraining, the AUCs of the models still almost remained at 1, two of them going down to 0.999 according to Table 6.18. The overall ability of the saliency mapping techniques to faithfully represent the model's decision went even further down for the Pasa model trained on 11k, while the Proportional Energy metrics seemed to remain stable. This slightly affects the Road-Normalised PropEng Average too, their score lowered slightly overall in Table 6.19. For the DenseNet-121 model trained on 11k, the overall $AOPC_{Combined}$ increased according to Table 6.20, while the Proportional Energy scores remained relatively stable. The highest Road-Normalised PropEng Average was affected, and it seems like this retraining led to a decrease to 0.299 for EigenGradCAM, decreasing from the previous highest score of 0.321 that Grad-CAM++ obtained. Further, for the DenseNet-121 model trained on MC-CH-IN-11k in Table 6.21, the $AOPC_{Combined}$

decreased for the techniques overall, and the Proportional Energy decreased, too, but a little less than $AOPC_{Combined}$. However, the overall Road-Normalised PropEng Average did not get affected much: EigenGradCAM, which was previously the best performing saliency mapping technique, got beaten by Eigen-CAM by 0.001, making Eigen-CAM the highest scoring visualisation technique on Road-Normalised PropEng Average with 0.227. For both DenseNet-121 models, LayerCAM and GradCAMElementwise still score the highest on $AOPC_{Combined}$, indicating that these visualisations techniques still offer the most faithful explanations for DenseNet-121 models detecting aTB from CXR.

Qualitative Analysis - Models with Second Data Augmentation Pipeline This time, the focus lies on how the previous visualisations in Figure 6.3, best visualisations according to LayerCAM, changed for the DenseNet-121 model trained on MC-CH-IN-11k. Looking at Figure 6.5, it seems like it was possible to reduce the previous biases with the help of data augmentations, however, the model seems to focus more on the right side of the CXRs, and even on the right side of the ground truth visualisations. Further, the model still seems to have other biases that are rather unpredictable, sometimes focusing outside the lungs or even outside the body. Looking at Figure 6.6 and comparing it to the previous visualisations with Score-CAM, it seems like thanks to the additional data augmentations, the biases were generally reduced, as the focus now almost always lies within the lung regions, with some lower activations on other parts like the neck sometimes, indicating that the detection of the bias and counteracting it with more data augmentations was somewhat a successful approach. The other visualisations can be found in Section A.2.2, they will not be further discusees here, as there is not much to be gained from them.

Qualitative Analysis - DenseNet-121 trained on ImageNet and NIH CXR14 - Eigen-CAM Since the indirect sub-model (DenseNet-121 trained on ImageNet and NIH CXR14) in combination with Eigen-CAM scored the highest on ROAD-Normalised PropEng Average, indicating that this setup offers the best combination of faithfulness and the correctness of the localisation, and thus the best trade-off between discovering non-human patterns and adhering to human-made ground truth labels, this model has been selected for the final qualitative analysis. Figure 6.7 shows the top 12 visualisations that obtained the highest ROAD-Normalised PropEng Average with this setup. Looking at the images, the heatmaps seem to generally align to a very high degree with the findings of the radiologists. And since the $AOPC_{Combined}$ part ensures that these explain the model's decision-making process, these visualisations can be trusted. However, it is also interesting to see how the lowest performing visualisations look like, to gain insights about how much trust for the model is warranted, and to find potential biases or problems with the classification and visualisations. Optimally, these would be analysed by radiologists to gain insights. They can be seen in Figure 6.8. Looking at the visualisations, sometimes they agree with the ground truth labels, e.g. the sample on the bottom left, but the sample also includes 2 other regions. It is unclear if the radiologists or the model made an error here in localising the radiological signs. In the other type of failures, the model seems to focus on everything but the lungs. The third type of failure seems to focus on the middle of the CXR, and these look exactly like the visualisations of healthy cases in Figure 6.9, which will be discussed shortly. For the second and third types of failure, it seems like the model classified these cases as healthy, and the error is not necessarily attributable to the saliency mapping technique, but rather the model itself. Looking closer at Figure 6.9, we can see 12 randomly selected healthy cases. The technique and model were never intended to be used on healthy cases, as detecting the non-presence of a sign is difficult to interpret. Nonetheless, they look the same as the failure cases in Figure 6.8. It seems like if the model is classifying something as healthy, it shows a relatively centralised activation, indicating that this is a normal neutral behaviour of the setup when no radiological signs are detected. However, there is a second type of visualisation for healthy cases, within which the model seems

to focus on everything but the lungs.

7.2.1 RQ 2.1

From the discussion of Section 7.2, we can see that the selected three metrics *Proportional Energy*, $AOPC_{Combined}$, and *ROAD-Normalised PropEng Average* were all useful in interpreting the results of the visualisations.

The purpose of the $AOPC_{Combined}$ scores was to measure the performance of the saliency mapping techniques while minimising the influence of the models on the outcome as much as possible. This results in a metric that measures how faithful the visualisations are to the model's decision-making process. Additionally, this metric can be used to select samples for qualitative analysis, potentially also showing non-human understandable patterns in the visualisations that could have contributed to a correct classification. Since it is non-distinguishable for humans if these patterns are correct, or if they are just biases, with the qualitative analysis of high scoring $AOPC_{Combined}$, it was possible to detect biases visually.

The purpose of the localisation metric Proportional Energy was to measure the more practical performance of the best performing saliency mapping technique and model combinations and to see how well each model scored with its best visualisation method by using the ground truth bounding box data from the TBX11K dataset. It accurately shows visualisations that match the human-made ground truth labels, and it is overall a metric that can faithfully capture the algorithm's process. This metric however is limited to the ability of the expert radiologists to detect radiological signs, so this metric will inflict a penalty if the decision of the model is based only on machine-observable patterns.

The combination of these two metrics, ROAD-Normalised PropEng Average, was thus a metric that offered a trade-off between trusting machine-observable patterns and human-generated ground truth labels, while also considering both, the faithfulness and correct localisations of the visualisations. The qualitative analysis of the top performing visualisations for DenseNet-121 trained on ImageNet and NIH CXR14, with the visualisations produced by Eigen-CAM, further shows how well the visualisations that are selected based on this metric look like.

7.3 Limitations & Future Work

RQ1 Limitations & Future Work It was not possible to conclude in Section 7.1.2 if including *latent TB* and *sick & non-TB* is overall preferable or not for the direct models. It remains unclear for the direct models why the Pasa model seems to benefit from the additional labels *healthy* and *sick & non-TB*, and the DenseNet-121 does not. For future work, it is recommended to try different types of splits and model architectures (direct and indirect) to gain more concluding results. It could be worth exploring splits where the TBX11K dataset is split into 4 labels, and the output layer of the models would be adapted accordingly, one for each class. Another possibility is to create a model architecture like in the study of Liu et al. (2020), one that first discriminates between *healthy* and *sick & non-TB* from TB-positive classes, and then on a second step learns to differentiate from *latent TB* and *active TB*. Other methods of counteracting the class balance are potentially helpful to avoid the suspected overfitting of the majority class. Generally, more balanced datasets are desirable, although due to the nature of medical data, the positive cases are usually far more difficult to obtain. For future work, additional hyperparameter tuning is also recommended after the addition of each new dataset for model training, as the previously obtained hyperparameters are not generalisable. Additionally, future research should also focus on finding more suitable data augmentations to improve aTB detection with CXR, and to remove biases. This coincides also with the results obtained in Section 7.2. Further, there are limitations

to the generalisability of the models, as data on children and pregnant are sparse and missing from the used datasets (Qin et al., 2021; World Health Organization, 2020). In clinical settings, usually not only one view of CXR is used, but other views like lateral views were needed for more accuracy too (Doi, 2007; Rajpurkar et al., 2018). Future research could try to train models using both views to potentially further improve the performance of the existing models. Clinical and demographic data of patients increases the accuracy of the models (Rajpurkar et al., 2020), future research should also try to incorporate clinical readings into the aTB detection process. Last but not least, the CH dataset received an update which includes radiological sign annotations now, however, at the start of this thesis, they were not available. Future research could also make use of this additional dataset.

RQ2 Limitations & Future Work According to the results obtained in this work, a saliency mapping technique's performance, i.e. its ability to faithfully visualise the model's decisions, seems to be highly reliant on the model architecture itself, at least for those that are used to detect aTB from CXRs. Further research should focus on an analysis of multiple saliency techniques for each model to ensure that the best technique is selected for the chosen model to generate visual explanations. A limitation of this insight is that is possible that the custom Pasa model was simply an outlier, or that the integrated *pytorch-grad-cam* package is not compatible with all model architectures, which could have led to erroneous results. Further, FullGrad was not compatible with the Pasa model and consistently scored a low $AOPC_{Combined}$, which reinforces these assumptions, but this could be also due to the complexity of this saliency mapping technique. Still, most saliency mapping techniques are advertised to work for all CNN model architectures, so this should theoretically be not a limitation. According to If it is desirable to have explainable models that also perform well on the aTB classification task, then future work should focus on applying saliency mapping techniques to indirect models (according to results from Section 6.1.1), specifically multi-label DenseNet-121 models trained on radiological signs. Future dataset creators should focus on creating TB-related CXR datasets with labels for each detected TB-related radiological sign, and optimally with annotations on the CXRs showing the are they were detected on, such that the produced visualisations can be evaluated.

RQ2.1 Limitations & Future Work All of the three introduced metrics, $AOPC_{Combined}$, Proportional Energy, and ROAD-Normalised PropEng Average, allow for a good comparative analysis, however, their absolute values are difficult to interpret. Is a Proportional Energy of 0.5 a good score or not? Looking at the qualitative analysis even scores up to 0.5 for each of the metrics show successful visualisations. As mentioned in Section 3.4.3, MoRF and LeRF show often different behaviour and results for different saliency mapping techniques (Rong et al., 2022; Srinivas and Fleuret, 2019; Tomsett et al., 2020). They are not well understood yet, and it could make sense to evaluate them separately instead of combining them like with $AOPC_{Combined}$. The qualitative analysis conducted in this work is very limited, as the author of this work is not a radiologist or medical expert. It could make sense in future research to do a more in-depth analysis with radiologists, and potentially a clinical study to see if the visualisations chosen on ROAD-Normalised PropEng Average are useful for radiologists, especially for models with an AUC of 1 producing samples performing high on $AOPC_{Combined}$ while performing low on Proportional Energy, as they could reveal patterns that go beyond those that are revealed by humans, and biases could be observer potentially in a better way. Further, *ROAD-Weighted PropEng* weights both of the metrics by the same amount, however, it might make sense to explore how the ROAD-Normalised PropEng Average changes when the two metrics are weighted differently (e.g. by incorporating entropy into the equation). This could allow for a more nuanced analysis of the trade-off this metric allows, and it would make it possible to choose how much machine-detected patterns should be weighted compared to human-detected patterns. Last but not least, the ROAD-Normalised

PropEng Average can be influenced by outliers, since it is an average rather than a median score. Taking the median over the dataset could potentially make more sense.

Chapter 8

Conclusion

One of the primary focuses of this thesis was to explore whether the prediction of active Tuberculosis aTB by direct and indirect deep learning models could be improved through the utilisation of a new dataset, TBX11K, specific to active Tuberculosis. The study also aimed to examine various aspects related to model performance, generalisation, and architecture. While the Pasa and DenseNet-121 models demonstrated promising results through the inclusion of more datasets, they exhibited varying performance when incorporating additional labels like *latent TB* and *sick & non-TB*. This inconclusive behaviour underscores the need for further investigation. On the other hand, indirect models did not exhibit a significant improvement in performance with the inclusion of the TBX11K dataset, no matter the labels. For future research, investigating alternative data splits, model architectures, and hyperparameter settings could offer more insights into the factors influencing model performance. The incorporation of clinical and demographic data as additional features, and the exploration of different data augmentations, are recommended for improving the model's capabilities and robustness.

As for the second focus, achieving explainability for the deep learning models detecting aTB through the application of saliency mapping techniques, various insights were gained. To analyse the performance of the techniques, two different metrics were selected, $AOPC_{Combined}$ and Proportional Energy, and based on them a new metric called *ROAD-Normalised PropEng Average* was introduced based on which the best performing saliency mapping technique and model combination was detected. A multi-label DenseNet-121 model trained on radiological signs with visualisations produced by Eigen-CAM produced the overall best visualisations. Further, findings imply that there is no universal saliency mapping technique that performs always well for the aTB detection task. It relies always on the model it was used on. This underscores the need for an analysis of the performance of various saliency mapping techniques each time a new model architecture is used.

The research highlighted the significance and complexities of utilising deep learning models for TB detection through CXR. Novel visualisation techniques provide valuable insights but also raise questions about the interplay of model architecture, data, and explanatory methods. As we move towards more advanced and reliable AI diagnostics, addressing these challenges will be critical.

Appendix A

Attachments

A.1 Sensitivity, Specificty, and F1-Scores for Cross-Validated Models from RQ1

Listed below in this section are the tables for each of the models mentioned in Section 6.1. For each of them, their cross-validated Sensitivity, Specificity, and F1-Scores on their test sets are reported. They were obtained by finding the optimum threshold, which is the threshold that achieved the highest F1-Score on their respective validation sets.

Table A.1: These tables depicts the final Sensitivity, Specificity, and F1-Score obtained for the Pasa Models by first finding the optimum threshold on the validation set the model was trained on, and then evaluating the performance with the optimum threshold on the respective test set.

(a) Direct Pasa Model Trained on 11k

F1-Score	Sensitivity	Specificity
0.99	0.99	1

(b) Direct Pasa Model Trained on 11kv2

F1-Score	Sensitivity	Specificity
0.83	0.91	0.98

(c) Direct Pasa Model Trained on MC

F1-Score	Sensitivity	Specificity
0.78	0.78	0.85

(d) Direct Pasa Model Trained on MC-CH

F1-Score	Sensitivity	Specificity
0.83	0.86	0.79

(e) Direct Pasa Model Trained on MC-CH-IN

F1-Score	Sensitivity	Specificity
0.83	0.83	0.82

(f) Direct Pasa Model Trained on MC-CH-IN-11k

F1-Score	Sensitivity	Specificity
0.91	0.93	0.97

(g) Direct Pasa Model Trained on MC-CH-IN-11kv2

F1-Score	Sensitivity	Specificity
0.81	0.85	0.97

Table A.2: These tables depicts the final Sensitivity, Specificity, and F1-Score obtained for the direct DenseNet-121 models by first finding the optimum threshold on the validation set the model was trained on, and then evaluating the performance with the optimum threshold on the respective test set.

(a) Direct DenseNet-121 Model Trained on 11k

F1-Score	Sensitivity	Specificity
0.99	0.99	1

(b) Direct DenseNet-121 Model Trained on 11kv2

F1-Score	Sensitivity	Specificity
0.85	0.94	0.98

(c) Direct DenseNet-121 Model Trained on MC

F1-Score	Sensitivity	Specificity
0.69	0.71	0.75

(d) Direct DenseNet-121 Model Trained on MC-CH

F1-Score	Sensitivity	Specificity
0.81	0.83	0.79

(e) Direct DenseNet-121 Model Trained on MC-CH-IN

F1-Score	Sensitivity	Specificity
0.82	0.85	0.77

(f) Direct DenseNet-121 Model Trained on MC-CH-IN-11k

F1-Score	Sensitivity	Specificity
0.91	0.92	0.98

(g) Direct DenseNet-121 Model Trained on MC-CH-IN-11kv2

F1-Score	Sensitivity	Specificity
0.81	0.85	0.97

Table A.3: These tables depicts the final Sensitivity, Specificity, and F1-Score obtained for the indirect models by first finding the optimum threshold on the validation set the model was trained on, and then evaluating the performance with the optimum threshold on the respective test set.

(a) Indirect Model Trained on 11k

F1-Score	Sensitivity	Specificity
0.85	0.84	0.98

(b) Indirect Model Trained on 11kv2

F1-Score	Sensitivity	Specificity
0.75	0.62	0.99

(c) Indirect Model Trained on MC

F1-Score	Sensitivity	Specificity
0.90	0.88	0.95

(d) Indirect Model Trained on MC-CH

F1-Score	Sensitivity	Specificity
0.85	0.82	0.88

(e) Indirect Model Trained on MC-CH-IN

F1-Score	Sensitivity	Specificity
0.85	0.85	0.86

(f) Indirect1 Model Trained on MC-CH-IN-11k

F1-Score	Sensitivity	Specificity
0.79	0.74	0.97

(g) Indirect Model Trained on MC-CH-IN-11kv2

F1-Score	Sensitivity	Specificity
0.57	0.63	0.93

A.2 Top 12 Visualisations for Models according to AOPC and Proportional Energy

A.2.1 Visualisations for models with First Data Augmentation Pipeline

Figure A.1 to Figure A.6 show the top 12 performing visualisations of the 11k test set for the Pasa model trained on 11k, the DenseNet-121 model trained on 11k, and the DenseNet-121 sub-model of the indirect model. First, based on the $AOPC_{Combined}$ and Proportional Energy metrics, the best performing saliency mapping technique for each metric is shown according to the tables Table 6.14, Table 6.15, Table 6.16, and Table 6.17. Then, for each selected saliency mapping technique, the 12 visualisations that achieved the highest score on the metric they were selected on are presented.

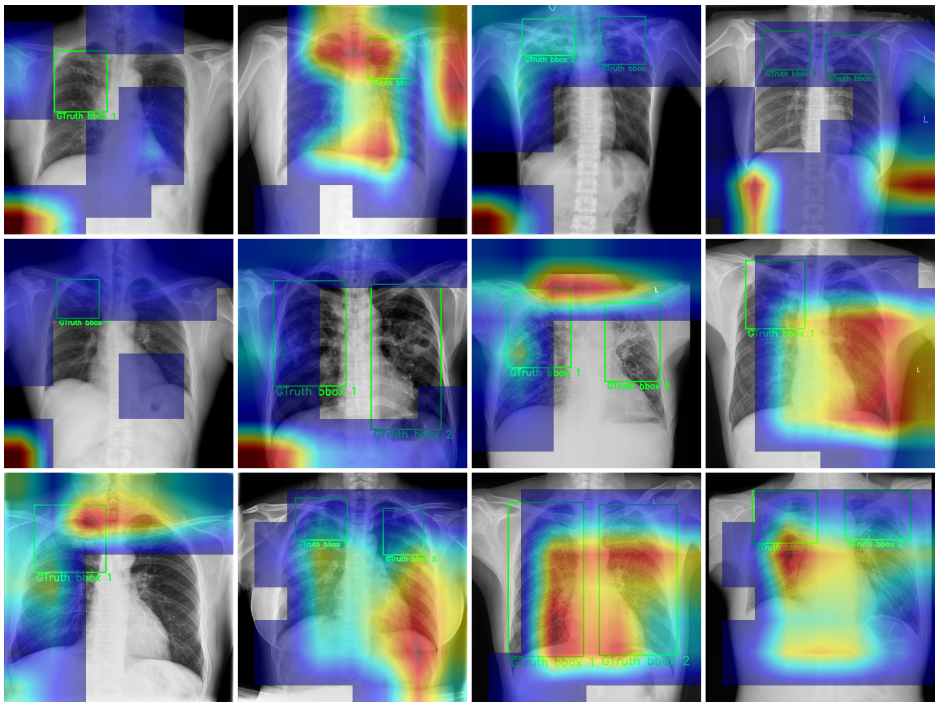


Figure A.1: PASA 11K TOP 12 VISUALISATIONS WITH XGRAD-CAM ACCORDING TO $AOPC_{Combined}$.

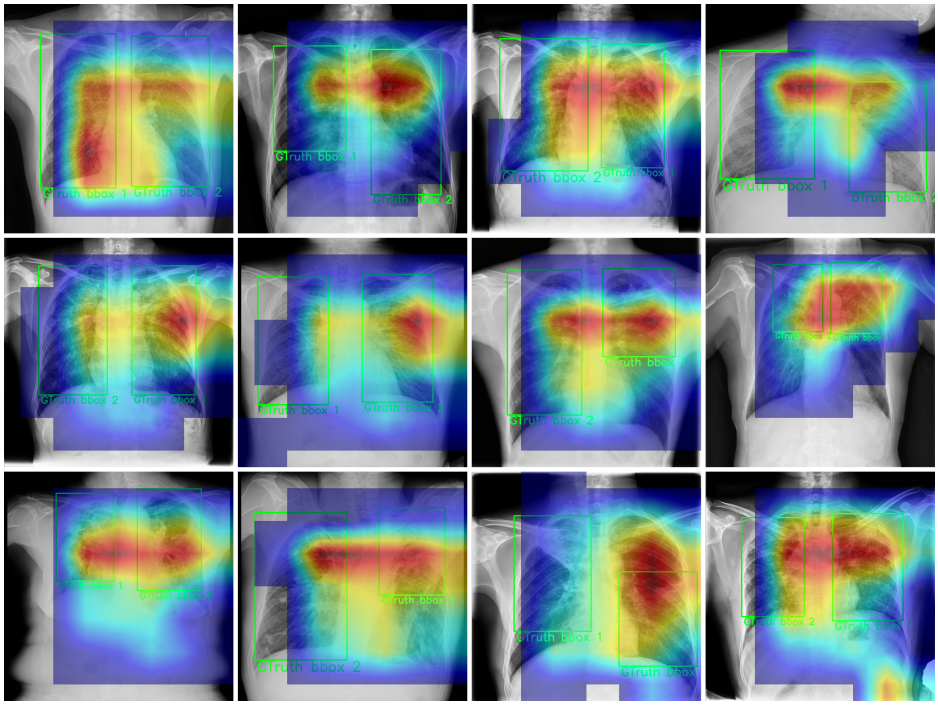


Figure A.2: PASA 11K TOP 12 VISUALISATIONS WITH GRAD-CAM ACCORDING TO PROPORTIONAL ENERGY.

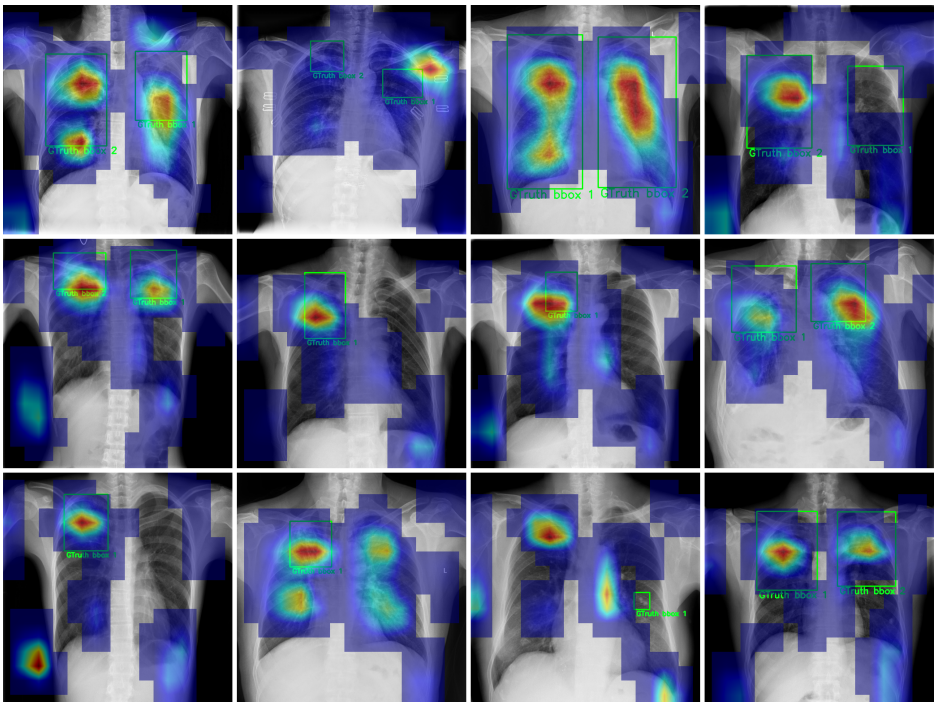


Figure A.3: DENSENET-121 11K TOP 12 VISUALISATIONS WITH GRADCAMELEMENTWISE ACCORDING TO $AOPC_{Combined}$.

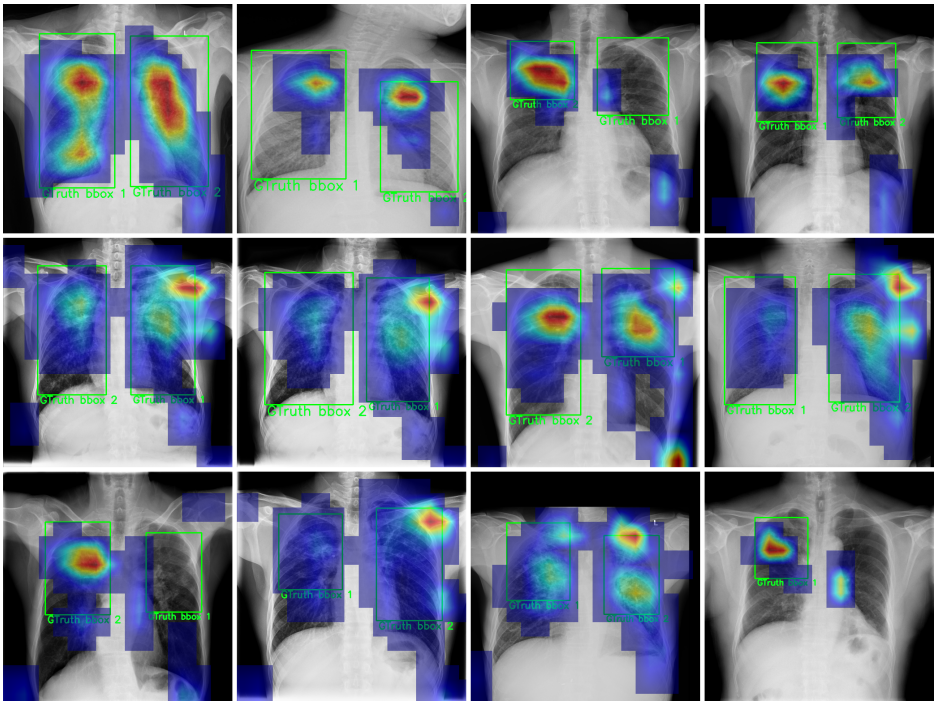


Figure A.4: DENSENET-121 11K TOP 12 VISUALISATIONS WITH GRAD-CAM++ ACCORDING TO PROPORTIONAL ENERGY.

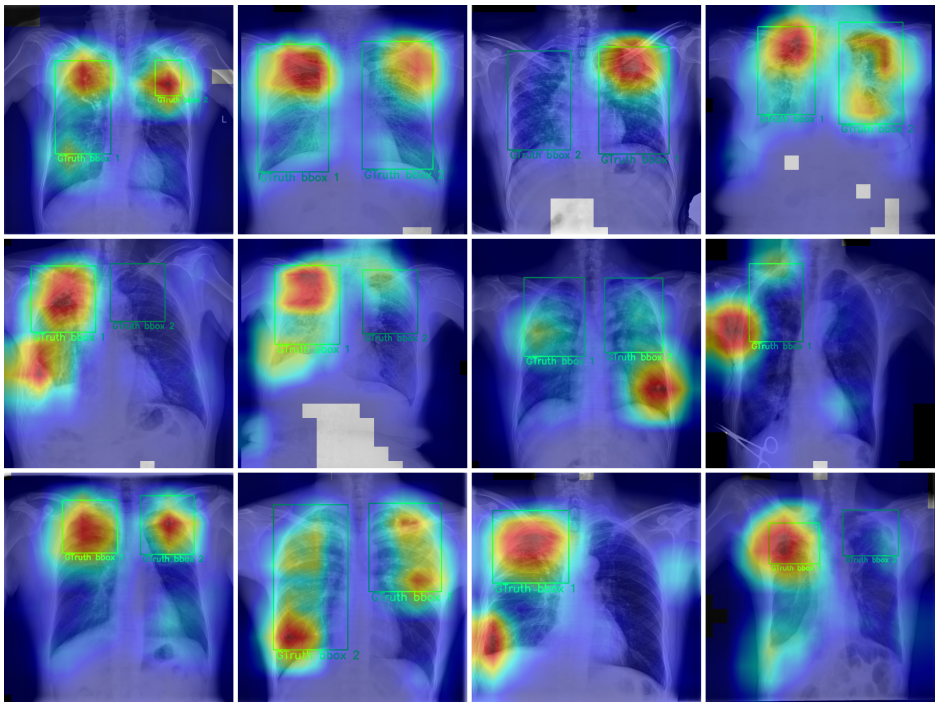


Figure A.5: INDIRECT DENSENET-121 SUB-MODEL TOP 12 VISUALISATIONS WITH LAYERCAM ACCORDING TO $AOPC_{Combined}$.

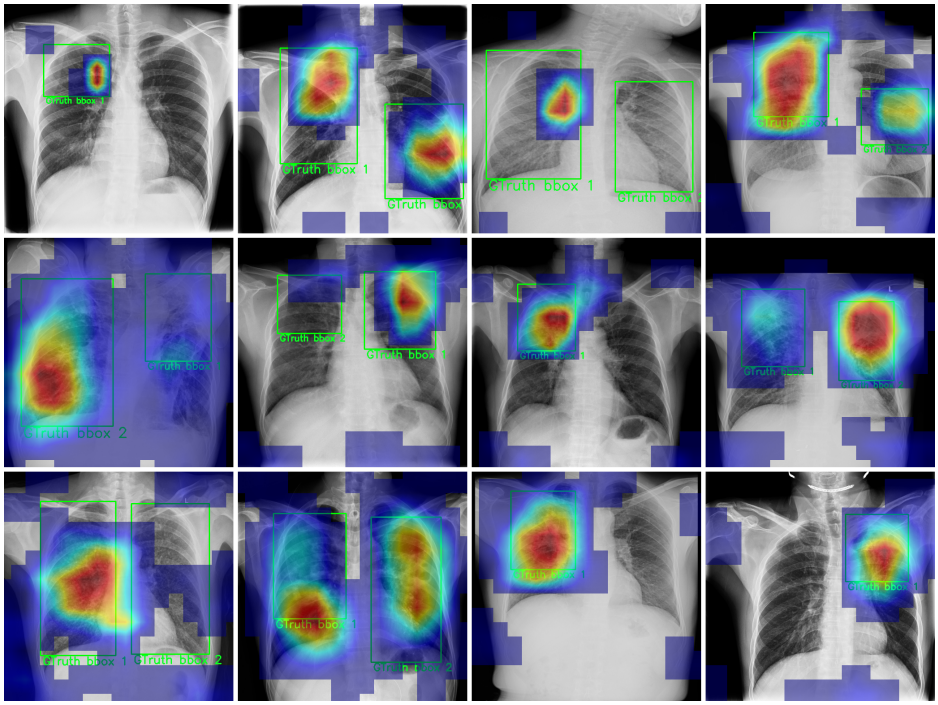


Figure A.6: INDIRECT DENSENET-121 SUB-MODEL TOP 12 VISUALISATIONS WITH HIRESCAM ACCORDING TO PROPORTIONAL ENERGY.

A.2.2 Visualisations for models with Second Data Augmentation Pipeline

For the qualitative analysis for the models that were trained with new data augmentations, Figure A.7 to Figure A.12 show the top 12 performing visualisations of the 11k test set for the Pasa model trained on 11k, the DenseNet-121 model trained on 11k, and the DenseNet-121 model trained on MC-CH-IN-11k. First, based on the $AOPC_{Combined}$ and Proportional Energy metrics, the best performing saliency mapping technique for each metric is selected according to the tables Table 6.19, Table 6.20, and Table 6.21. Then, for each selected saliency mapping technique, the 12 visualisations that achieved the highest score on the metric they were selected on are presented.

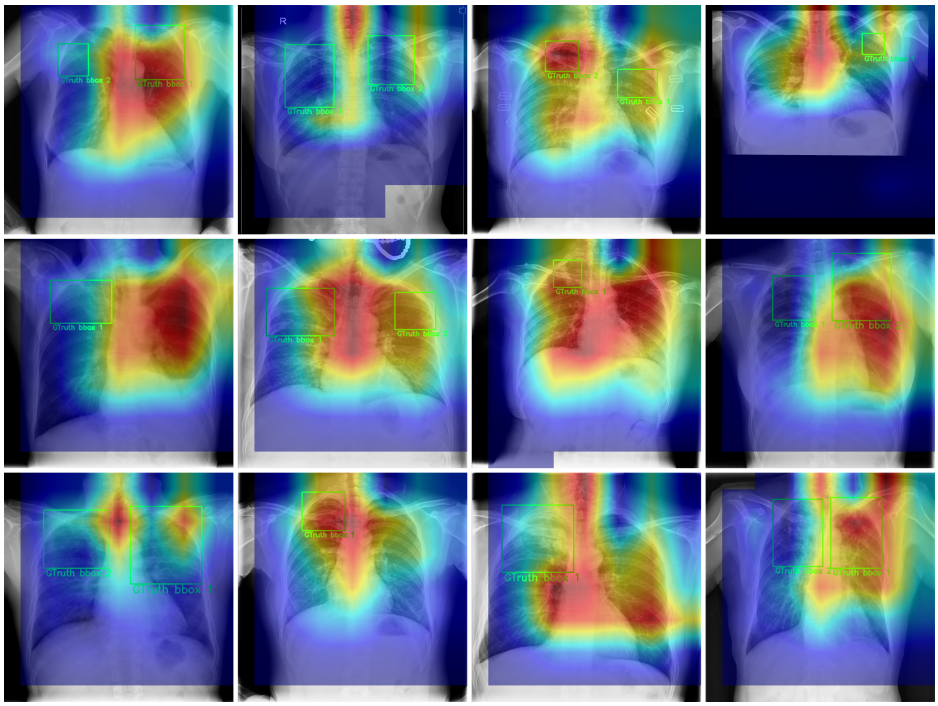


Figure A.7: PASA 11K WITH SECOND DATA AUGMENTATION PIPELINE TOP 12 VISUALISATIONS WITH LAYERCAM ACCORDING TO $AOPC_{Combined}$.

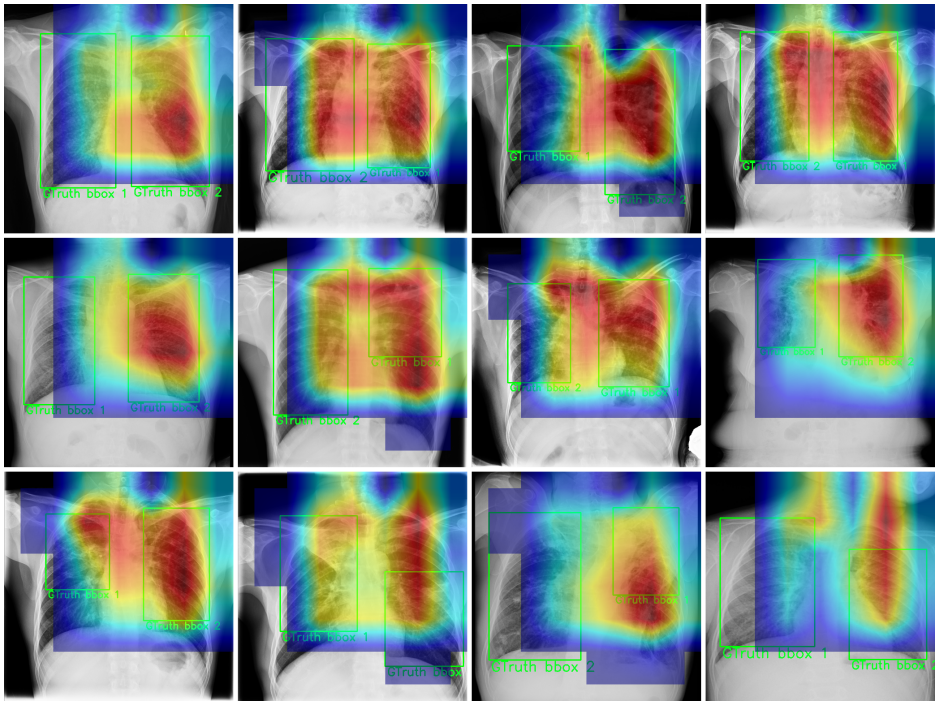


Figure A.8: PASA 11K WITH SECOND DATA AUGMENTATION PIPELINE TOP 12 VISUALISATIONS WITH GRAD-CAM++ ACCORDING TO PROPORTIONAL ENERGY.

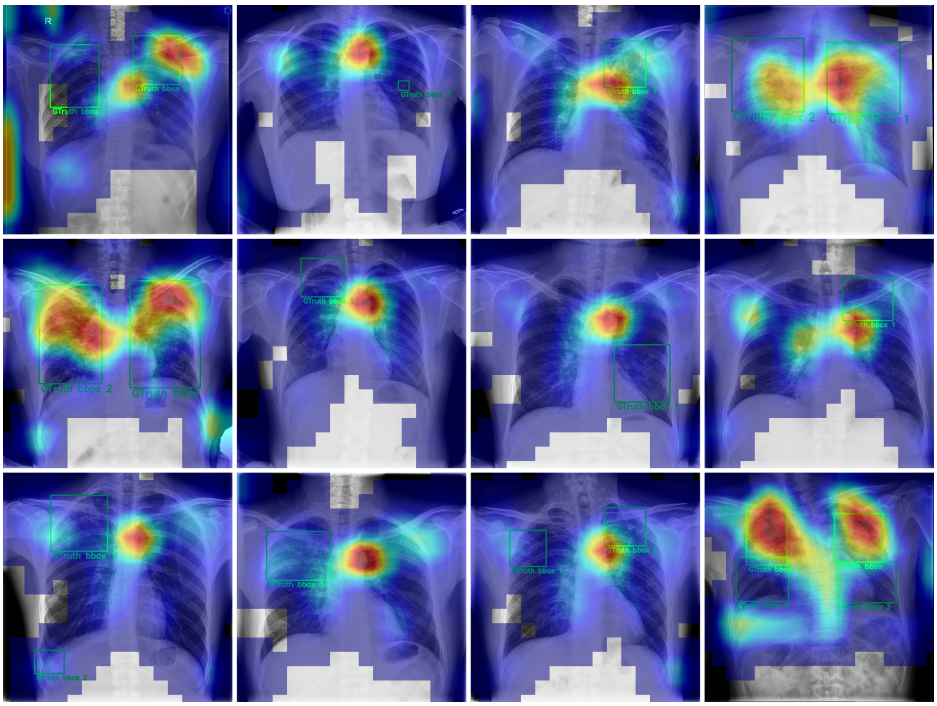


Figure A.9: DENSENET-121 11K WITH SECOND DATA AUGMENTATION PIPELINE TOP 12 VISUALISATIONS WITH LAYERCAM ACCORDING TO $AOPC_{Combined}$.

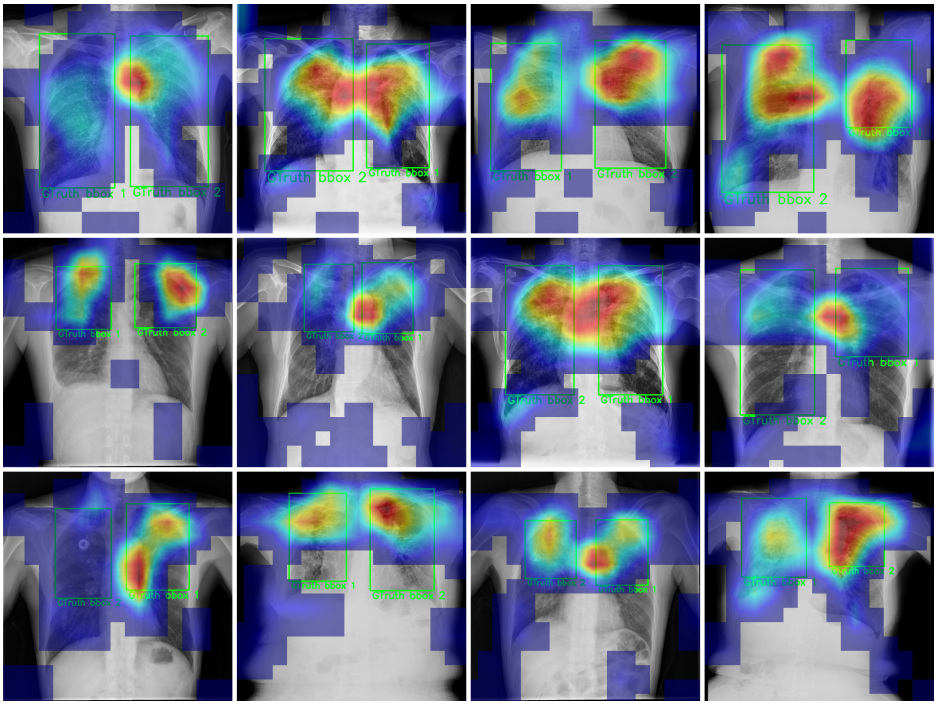


Figure A.10: DENSENET-121 11K WITH SECOND DATA AUGMENTATION PIPELINE TOP 12 VISUALISATIONS WITH GRAD-CAM ACCORDING TO PROPORTIONAL ENERGY.

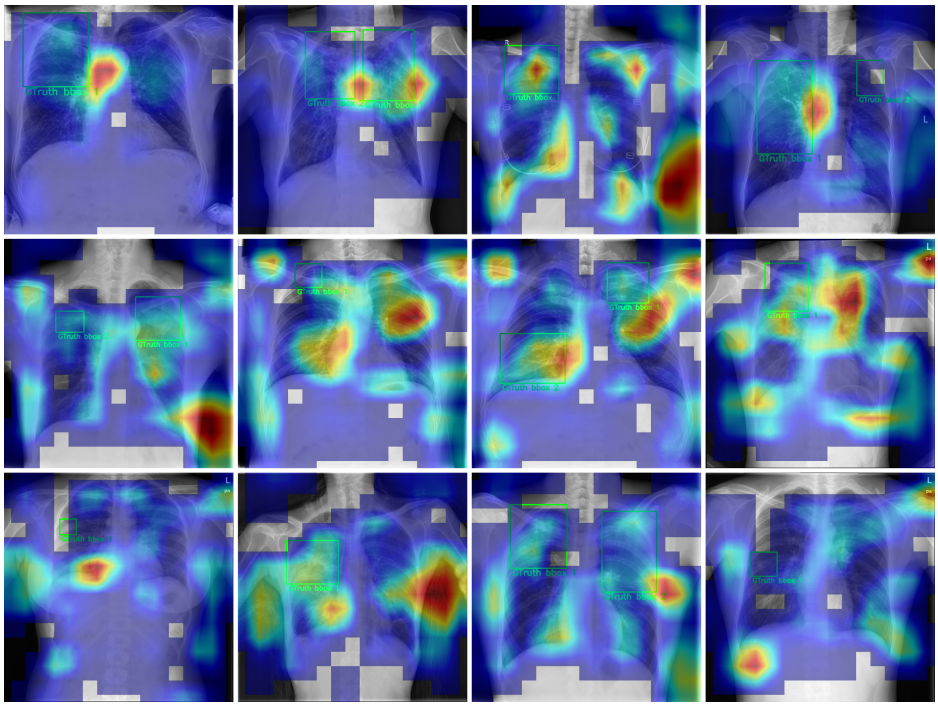


Figure A.11: DENSENET-121 MC-CH-IN-11K WITH SECOND DATA AUGMENTATION PIPELINE TOP 12 VISUALISATIONS WITH GRAD-CAM ACCORDING TO $AOPC_{Combined}$.

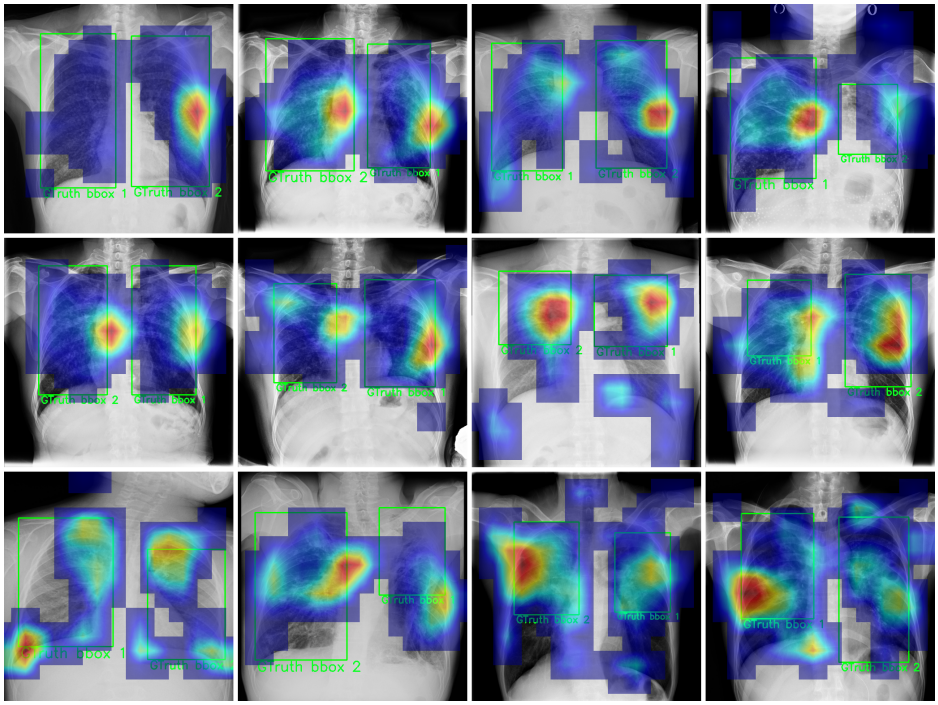


Figure A.12: DENSENET-121 MC-CH-IN-11K WITH SECOND DATA AUGMENTATION PIPELINE TOP 12 VISUALISATIONS WITH EIGENGRADCAM ACCORDING TO PROPORTIONAL ENERGY.

List of Figures

List of Tables

Bibliography

Abeyagunasekera, S. H. P., Perera, Y., Chamara, K., Kaushalya, U., Sumathipala, P., and Senaweera, O. (2022). LISA : Enhance the explainability of medical images unifying current XAI techniques. In *2022 IEEE 7th International conference for Convergence in Technology (I2CT)*, pages 1–9.

Aggarwal, C. C. (2018). *Neural Networks and Deep Learning: A Textbook*. Springer International Publishing, Cham.

Amann, J., Blasimme, A., Vayena, E., Frey, D., Madai, V. I., and the Precise4Q consortium (2020). Explainability for artificial intelligence in healthcare: a multidisciplinary perspective. *BMC Medical Informatics and Decision Making*, 20(1):310.

Bany Muhammad, M. and Yeasin, M. (2021). Eigen-CAM: Visual explanations for deep convolutional neural networks. *SN Computer Science*, 2(1):47.

Bar, Y., Diamant, I., Wolf, L., Lieberman, S., Konen, E., and Greenspan, H. (2015). Chest pathology detection using deep learning with non-medical training. In *2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI)*, pages 294–297, Brooklyn, NY, USA. IEEE.

Bolelli, F., Allegretti, S., Baraldi, L., and Grana, C. (2020). Spaghetti labeling: directed acyclic graphs for block-based connected components labeling. *IEEE Transactions on Image Processing*, 29:1999–2012. Conference Name: IEEE Transactions on Image Processing.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159.

Castelvecchi, D. (2016). Can we open the block box of AI? *Nature*, 538:20–23.

Chan, H.-P., Doi, K., Vybrony, C. J., Schmidt, R. A., Metz, C. E., Lam, K. L., Ogura, T., Wu, Y., and Macmahon, H. (1990). Improvement in radiologists' detection of clustered microcalcifications on mammograms: The potential of computer-aided diagnosis. *Investigative Radiology*, 25(10):1102–1110.

Chattopadhay, A., Sarkar, A., Howlader, P., and Balasubramanian, V. N. (2018). Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 839–847.

Chauhan, A., Chauhan, D., and Rout, C. (2014). Role of Gist and PHOG Features in computer-aided diagnosis of tuberculosis without segmentation. *PLOS ONE*, 9(11):e112980. Publisher: Public Library of Science.

Curvo-Semedo, L., Teixeira, L., and Caseiro-Alves, F. (2005). Tuberculosis of the chest. *European Journal of Radiology*, 55(2):158–172.

Cutillo, C. M., Sharma, K. R., Foschini, L., Kundu, S., Mackintosh, M., and Mandl, K. D. (2020). Machine intelligence in healthcare—perspectives on trustworthiness, explainability, usability, and transparency. *npj Digital Medicine*, 3(1):1–5. Number: 1 Publisher: Nature Publishing Group.

Delft Imaging (2023). CAD4TB. https://www.delft.care/cad4tb/ (Last accessed on 30.08.2023).

Desai, S. and Ramaswamy, H. G. (2020). Ablation-CAM: Visual explanations for deep convolutional network via gradient-free localization. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 972–980, Snowmass Village, CO, USA. IEEE.

Doi, K. (2007). Computer-aided diagnosis in medical imaging: Historical review, current status and future potential. *Computerized Medical Imaging and Graphics*, 31(4-5):198–211.

Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *stat*, 1050:2.

Draelos, R. L. and Carin, L. (2021). Use HiResCAM instead of Grad-CAM for faithful explanations of convolutional neural networks. arXiv:2011.08891 [cs, eess].

Fu, R., Hu, Q., Dong, X., Guo, Y., Gao, Y., and Li, B. (2020). Axiom-based Grad-CAM: Towards accurate visualization and explanation of CNNs. arXiv:2008.02312 [cs, eess].

Gildenblat, J. (2023). Advanced AI explainability for PyTorch. https://github.com/jacobgil/pytorch-grad-cam (Last accessed on 30.08.2023).

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., and Kagal, L. (2018). Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89. IEEE.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In Gordon, G., Dunson, D., and Dudík, M., editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA. PMLR.

Groen, A. M., Kraan, R., Amirkhan, S. F., Daams, J. G., and Maas, M. (2022). A systematic review on the use of explainability in deep learning systems for computer aided diagnosis in radiology: Limited use of explainable AI? *European Journal of Radiology*, 157:110592.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., and Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM Computing Surveys*, 51(5):93:1–93:42.

Harris, M., Qi, A., Jeagal, L., Torabi, N., Menzies, D., Korobitsyn, A., Pai, M., Nathavitharana, R. R., and Ahmad Khan, F. (2019). A systematic review of the diagnostic accuracy of artificial intelligence-based computer programs to analyze chest X-Rays for pulmonary tuberculosis. *PLOS ONE*, 14(9):e0221339.

Heo, S.-J., Kim, Y., Yun, S., Lim, S.-S., Kim, J., Nam, C.-M., Park, E.-C., Jung, I., and Yoon, J.-H. (2019). Deep learning algorithms with demographic information help to detect tuberculosis in chest radiographs in annual workers' health examination data. *International Journal of Environmental Research and Public Health*, 16(2):250. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute.

Holzinger, A., Langs, G., Denk, H., Zatloukal, K., and Müller, H. (2019). Causability and explainability of artificial intelligence in medicine. *WIREs Data Mining and Knowledge Discovery*, 9(4):e1312. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1312.

Hooker, S., Erhan, D., Kindermans, P.-J., and Kim, B. (2019). A benchmark for interpretability methods in deep neural networks. *Advances in neural information processing systems*, 32.

Houben, R. M. and Dodd, P. J. (2016). The global burden of latent tuberculosis infection: a re-estimation using mathematical modelling. *PLoS medicine*, 13(10):e1002152.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269. IEEE Computer Society.

Hwang, E. J., Park, S., Jin, K.-N., Kim, J. I., Choi, S. Y., Lee, J. H., Goo, J. M., Aum, J., Yim, J.-J., and Park, C. M. (2019). Development and validation of a deep learning–based automatic detection algorithm for active pulmonary tuberculosis on chest radiographs. *Clinical Infectious Diseases*, 69(5):739–747.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France. PMLR.

Jaeger, S., Candemir, S., Antani, S., Wáng, Y.-X. J., Lu, P.-X., and Thoma, G. (2014). Two public chest X-Ray datasets for computer-aided screening of pulmonary diseases. *Quantitative Imaging in Medicine and Surgery*, 4(6):475–477.

Jiang, P.-T., Zhang, C.-B., Hou, Q., Cheng, M.-M., and Wei, Y. (2021). Layercam: Exploring hierarchical class activation maps for localization. *IEEE Transactions on Image Processing*, 30:5875–5888. Conference Name: IEEE Transactions on Image Processing.

Krishnamurthy, P., Basak Chowdhury, A., Tan, B., Khorrami, F., and Karri, R. (2020). Explaining and interpreting machine learning CAD decisions: An IC testing case study. In *Proceedings of the 2020 ACM/IEEE Workshop on Machine Learning for CAD*, MLCAD '20, pages 129–134, New York, NY, USA. Association for Computing Machinery.

Kundel, H. L., Nodine, C. F., and Krupinski, E. A. (1990). Computer-displayed eye position as a visual aid to pulmonary nodule interpretation. *Investigative Radiology*, 25(8):890–896.

Lakhani, P. and Sundaram, B. (2017). Deep learning at chest radiography: automated classification of pulmonary tuberculosis by using convolutional neural networks. *Radiology*, 284(2):574–582.

Li, F., Aoyama, M., Shiraishi, J., Abe, H., Li, Q., Suzuki, K., Engelmann, R., Sone, S., Macmahon, H., and Doi, K. (2004). Radiologists' performance for differentiating benign from malignant lung nodules on high-resolution CT using computer-estimated likelihood of malignancy. *AJR. American journal of roentgenology*, 183(5):1209–1215.

Li, Q., Li, F., Shiraishi, J., Katsuragawa, S., Sone, S., and Doi, K. (2003). Investigation of new psychophysical measures for evaluation of similar images on thoracic computed tomography for distinction between benign and malignant nodules. *Medical Physics*, 30(10):2584–2593.

Lipton, Z. C. (2018). The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57.

Liu, Y., Wu, Y.-H., Ban, Y., Wang, H., and Cheng, M.-M. (2020). Rethinking computer-aided tuberculosis diagnosis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2646–2655.

Lunit Insight (2023). Lunit Inc. https://www.lunit.io/en/products/cxr (Last accessed on 30.08.2023).

Miró-Nicolau, M., Moyà-Alcover, G., and Jaume-i Capó, A. (2022). Evaluating explainable artificial intelligence for X-ray image analysis. *Applied Sciences*, 12(9):4459. Number: 9 Publisher: Multidisciplinary Digital Publishing Institute.

Montenegro, H., Silva, W., and Cardoso, J. S. (2021). Privacy-preserving generative adversarial network for case-based explainability in medical image analysis. *IEEE Access*, 9:148037–148047. Conference Name: IEEE Access.

Müller, Andreas C. (2020). Data Splitting Strategies — Applied Machine Learning in Python. https://amueller.github.io/aml/04-model-evaluation/1-data-splitting-strategies.html (Last accessed on 30.08.2023).

OpenCV (2023). Opencv. https://github.com/opencv/opencv (Last accessed on 30.08.2023).

Pande, T., Pai, M., Khan, F. A., and Denkinger, C. M. (2015). Use of chest radiography in the 22 highest tuberculosis burden countries. *European Respiratory Journal*, 46(6):1816–1819. Publisher: European Respiratory Society Section: Agora.

Pasa, F., Golkov, V., Pfeiffer, F., Cremers, D., and Pfeiffer, D. (2019). Efficient deep network architectures for fast chest X-Ray tuberculosis screening and visualization. *Scientific Reports*, 9(1):6268.

Qin, Z. Z., Ahmed, S., Sarker, M. S., Paul, K., Adel, A. S. S., Naheyan, T., Barrett, R., Banu, S., and Creswell, J. (2021). Tuberculosis detection from chest X-Rays for triaging in a high tuberculosis-burden setting: an evaluation of five artificial intelligence algorithms. *The Lancet Digital Health*, 3(9):e543–e554.

Qure.ai (2021). qXR | AI for detection of abnormal findings on a Chest X-ray. https://qure.ai/product/qxr/ (Last accessed on 30.08.2023).

Rajpurkar, P., Irvin, J., Ball, R. L., Zhu, K., Yang, B., Mehta, H., Duan, T., Ding, D., Bagul, A., Langlotz, C. P., Patel, B. N., Yeom, K. W., Shpanskaya, K., Blankenberg, F. G., Seekins, J., Amrhein, T. J., Mong, D. A., Halabi, S. S., Zucker, E. J., Ng, A. Y., and Lungren, M. P. (2018). Deep learning for chest radiograph diagnosis: A retrospective comparison of the CheXNeXt algorithm to practicing radiologists. *PLOS Medicine*, 15(11):e1002686.

Rajpurkar, P., O'Connell, C., Schechter, A., Asnani, N., Li, J., Kiani, A., Ball, R. L., Mendelson, M., Maartens, G., van Hoving, D. J., Griesel, R., Ng, A. Y., Boyles, T. H., and Lungren, M. P. (2020). CheXaid: deep learning assistance for physician diagnosis of tuberculosis using chest X-Rays in patients with HIV. *npj Digital Medicine*, 3(1):115.

Raposo, G. (2021). Active tuberculosis detection from frontal chest X-Ray images. Idiap-com, Idiap Research Institute.

Rong, Y., Leemann, T., Borisov, V., Kasneci, G., and Kasneci, E. (2022). A consistent and efficient evaluation strategy for attribution methods. In *International Conference on Machine Learning*, pages 18770–18795. PMLR.

Santosh, K. C. and Antani, S. (2018). Automated chest X-Ray ccreening: can lung region symmetry help detect pulmonary abnormalities? *IEEE Transactions on Medical Imaging*, 37(5):1168–1177. Conference Name: IEEE Transactions on Medical Imaging.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. (2017). Grad-CAM: visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626.

Shen, R., Cheng, I., and Basu, A. (2010). A hybrid knowledge-guided detection technique for screening of infectious pulmonary tuberculosis from chest radiographs. *IEEE Transactions on Biomedical Engineering*, 57(11):2646–2656.

Shin, H.-C., Roth, H. R., Gao, M., Lu, L., Xu, Z., Nogues, I., Yao, J., Mollura, D., and Summers, R. M. (2016). Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE Transactions on Medical Imaging*, 35(5):1285–1298.

Siddharthshah (2020). Perceptron Learning Algorithm. https://medium.com/analytics-vidhya/perceptron-learning-algorithm-7ae7c4b90eb2 (Last accessed on 30.08.2023).

Simard, P., Steinkraus, D., and Platt, J. (2003). Best practices for convolutional neural networks applied to visual document analysis. In *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, pages 958–963.

Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Deep inside convolutional networks: Visualising image classification models and saliency maps.

Srinivas, S. and Fleuret, F. (2019). Full-gradient representation for neural network visualization. *Advances in neural information processing systems*, 32.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions.

Taha, A. A. and Hanbury, A. (2015). Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool. *BMC Medical Imaging*, 15:29.

The MathWorks (2023). Compare Deep Learning Models Using ROC Curves - MATLAB & Simulink - MathWorks Switzerland. https://ch.mathworks.com/help/deeplearning/ug/compare-deep-learning-models-using-ROC-curves.html (Last accessed on 30.08.2023).

Tomsett, R., Braines, D., Harborne, D., Preece, A., and Chakraborty, S. (2018). Interpretable to whom? A role-based model for analyzing interpretable machine learning systems. *arXiv:1806.07552 [cs]*. arXiv: 1806.07552.

Tomsett, R., Harborne, D., Chakraborty, S., Gurram, P., and Preece, A. (2020). Sanity checks for saliency metrics.

Wang, H., Wang, Z., Du, M., Yang, F., Zhang, Z., Ding, S., Mardziel, P., and Hu, X. (2020). Score-cam: Score-weighted visual explanations for convolutional neural networks.

Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M., and Summers, R. M. (2017). ChestX-Ray8: hospital-scale chest X-Ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3462–3471, Honolulu, HI. IEEE.

Wikipedia (2023). Neural network. https://en.wikipedia.org/w/index.php?title=Neural_network&oldid=1168729270 (Last accessed on 30.08.2023).

Williams, F. H. (1907). The use of X-Ray examinations in pulmonary tuberculosis. *The Boston Medical and Surgical Journal*, 157(26):850–853. Publisher: Massachusetts Medical Society _eprint: https://doi.org/10.1056/NEJM190712261572602.

World Health Organization (2016). *Chest radiography in tuberculosis detection: summary of current WHO recommendations and guidance on programmatic approaches*. World Health Organization. Number: WHO/HTM/TB/2016.20.

World Health Organization (2020). *WHO operational handbook on tuberculosis: Module 1 Prevention: Tuberculosis preventive treatment*. World Health Organization, Geneva.

World Health Organization (2021). *WHO operational handbook on tuberculosis: Module 2: Screening: Systematic screening for tuberculosis disease*. World Health Organization, Geneva.

World Health Organization (2022). Global Tuberculosis Report 2022.

Yang, F., Lu, P. X., Deng, M., Wáng, Y. X. J., Rajaraman, S., Xue, Z., Folio, L. R., Antani, S. K., and Jaeger, S. (2022). Annotations of lung abnormalities in the shenzhen chest X-Ray dataset for computer-aided screening of pulmonary diseases. *Data*, 7(7):95. Number: 7 Publisher: Multidisciplinary Digital Publishing Institute.

Yu-Jen Chen, Y.-J., Hua, K.-L., Hsu, C.-H., Cheng, W.-H., and Hidayati, S. C. (2015). Computer-aided classification of lung nodules on computed tomography images via deep learning technique. *OncoTargets and Therapy*, page 2015.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. (2016). Learning deep features for discriminative localization.