Tuberculosis Detection Model using Machine Learning

Ayman Husain

Dec 2025

1 Abstract

This project proposes the development of a supervised deep learning model to detect tuberculosis from chest X-ray images. It does so by classifying the images as either TB-positive or TB-negative by using the feature extractor to highlight key outliers in an image. In this project, I am building three models. One of these models we have built from scratch, and the other two models are pre-trained model specialized in disease detection called Densenet121 and Resnet. I will be finetuning the Densenet and Resnet to make it more suitable for this project's purpose. Finally, I will be conducting a comparative analysis on how my model compares to the transfer learning-based model through several performance metrics detailed below. The project also includes a Streamlit-based web application allowing real-time prediction from uploaded X-ray images.

2 Introduction

Tuberculosis (TB) remains one of the deadliest diseases in the world, with the WHO reporting A total of 1.25 million people died from tuberculosis (TB) in 2023 alone. Furthermore, they have set out on a mission to end the TB epidemic by the year 2030 [1]. As a result, there is a current rise in solutions to early and accurate detection of this disease so that it can be treated quickly and/or prevent transmissions. Traditional methods for testing for TB consists popularly of sputum microscopy and culture tests, which have proven to be time-consuming and an inefficient use of resources, especially for those in less fortunate environments. In this vacuum of cheaper and more efficient solutions, TB detection models using deep learning to analyze chest X-rays, has arisen. Below, I have reviewed several pieces of literature, all using similar methodologies of building a deep convolutional neural network (CNN) to achieve this feat, while also comparing them with my own model to evaluate efficacy and ease of implementation should the results prove fruitful.

3 Current Approaches

It is worth discussing briefly, the present state of tuberculosis detection in terms of methods and their limitations. Commonly, early detection of pulmonary TB is done through a process of microscopic examination of sputum cultures and chest X-rays[2]. There also exist strains that are drug-resistant, requiring a drug susceptibility test (DST), as the culturing will not work on them. As a result, the processes have proven to be tedious and time-consuming, often taking a longer time for pre-diagnosis processes and fail to provide reports at a fast enough rate, which is critical to early detection. In response to this, many immunoassay techniques have been developed by the medical industry, which take less time and have a higher sensitivity. However, they are faced with the other side of the coin's issues, having a high cost and need for well-established infrastructures. While these are simplifications of the different TB detection processes that exist now, a more detailed table overview can be found in M. Singh's paper.

They go on to discuss newer methods that are being introduced to com-

bat this issue of early TB detection, mostly focusing on different AI techniques that are still being experimented with. The foundations for these different CNNs that are suggested are classified into three different methods, being: unsupervised learning, semi-supervised learning and supervised learning. For this project, I intend to create a deep learning CNN model trained by supervised learning. This means that I will be feeding my model labelled data as input and outputs to find common patterns, as opposed to having one or both sets of unlabeled data like in the other methods. The author also mentions the use of transfer learning,that is for when I am trying to transfer knowledge bases between different models, as I am doing with the Densenet-121 model.

In most cases, they have applied different versions or models of CNN to detect TB using CXR images. Wong et al. [3] proposes a deep CNN model with a self-attention mechanism called TB-Net. This model showed much better performance compared to standard CNNs, showing results in accuracy, sensitivity and specificity of 99.86%, 100.0% and 99.71% respectively. It is particularly useful due to its ability to capture spatial dependencies in medical images, making it more efficient at classifying the test result as negative or positive for TB. However, it is worth mentioning that this model is almost too powerful or power hungry, in that it is not suitable for resource-constrained settings. With all that being said, my project is influenced by this research model, due to its high figures in results and due to its self-attention layer which will hopefully allow it to better locate the tuberculosis affected areas and learn more efficiently from the practice datasets.

Sharma et al. [4] focused on integrating deep learning with visualization techniques such as Grad-CAM to highlight key regions in CXR images contributing to TB classification. This method enhances model interpretability, which is critical for medical applications where decision transparency is essential. Another significant approach is presented by Jaeger et al. [5], who employed a UNet-based segmentation model for TB detection. While UNet is effective for biomedical image segmentation, its computational complexity and reliance on large training datasets pose challenges for real-world implementation.

On the other hand, Sharma aimed their paper at being more about visualization techniques relating to deep learning. For example, the most important technique they refer to is Grad-CAM to highlight key regions in CXR images, allowing it to better classify an input as TB positive or negative. However, a significant downside of this approach is that it relies on high-resolution images to be doing this, which again, may not be feasible in an underprivileged setting.

A commonality between all the papers is their emphasis on preprocessing techniques and transfer learning, but there is one paper that puts it on a pedestal more than the others, being Norval[6]. They mention methods such as lung ROI extraction and contrast enhancement that basically isolate the different regions of the lung from the X-ray and then improve the image to improve visibility of key features or points of interest. They also introduce a unique hybrid method which combines traditional computer-aided detection (CAD) with

deep learning and has shown a model accuracy of 92.54%. It does however, add a whole new level of complexity to the model by adding this much preprocessing and could slow the system down. Transfer learning has therefore become a common strategy. Pretrained models such as DenseNet121 and ResNet50 have shown strong performance on medical imaging tasks due to their deep feature hierarchies and ability to generalize well with relatively small datasets.

4 Preliminary Design

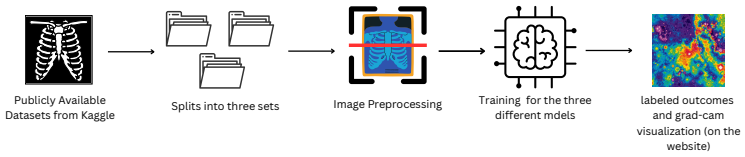


Figure 1: Graphical Abstractfor TB detection model.

The system I am proposing will follow a supervised learning approach, making use of labeled chest X-ray (CXR) datasets where each image is annotated as TB positive or TB negative.

4.1 Data Collection and Pre-processing

The dataset contains chest X-ray images that have been collected from publicly available datasets from Kaggle. The dataset contains a total of about 6,200 chest X-ray images. The images belong to two categories which are Normal and Tuberculosis. These images are placed in separate folders inside a raw data directory. A Python script is used to randomly split the images into training, validation, and testing sets. This script ensures that each set has a fair mix of both classes which helps the models learn patterns without bias.

Each image is converted to grayscale because X-ray information does not require color. The grayscale image is then duplicated across three channels because pretrained models

expect three channel inputs. Every image is resized to a fixed resolution of 224 by 224 pixels which keeps the input shape consistent for all models. The pixel values are normalized using values from the ImageNet dataset so the models can train more steadily since the inputs follow a familiar numeric range.

4.2 Models Used

As mentioned above, I intend to build three models; one from the ground up and another two using transfer learning from a CNN model pre-trained in medical imaging tasks. While these are my three main models on which I will be conducting a thorough comparative analysis, I will also be using the base metric results from the models mentioned in my current approaches section. The models in that section are also, as explained, specialized specifically for Tuberculosis image classification, but vary in their methods and inputs needed. So, the models used in my research overall are:

Custom CNN:The custom CNN is built from scratch with five convolutional blocks. Each block contains two layers that slide filters across the image to detect patterns. These filters learn basic shapes in early layers such as edges or bright regions. They learn more complex patterns in deeper layers such as shadows or textures that suggest infection. Each convolution is followed by batch normalization which stabilizes training by keeping the numbers inside the network from getting too large. Each convolution also has a ReLU activation which forces negative values to zero so the model can focus on strong features. A max pooling layer follows the first four blocks. Pooling makes the image smaller as it keeps only the strongest value in each region. This step helps the network focus on important areas and ignore small noise. The last block uses an adaptive average pool which squeezes the feature maps into a single value per channel. This gives the classifier a clean summary of what the filters found. The classifier contains a flatten layer which turns the feature maps into a vector. A fully connected layer then combines these values to learn relationships between them. ReLU is used again to help the model learn non-linear patterns. A dropout layer is added to prevent the model from memorizing the training set because it removes some values at random during training. The final linear layer outputs two values which correspond to the two classes. Cross entropy loss is used so the model learns to produce a high value for the correct class.

DenseNet-121: DenseNet121 is a deep convolutional network that was pretrained on a large image dataset. It is called DenseNet because each layer passes its output to every future layer inside a block. This helps the model reuse features and prevents the network from forgetting important details. DenseNet121 can find patterns in X-rays with fewer parameters than other models because the layers build on each other in a very efficient way. To adapt DenseNet121 for TB detection the original classifier is removed. A new classifier is added that outputs two values for the two classes. This classifier has a fully connected layer that reduces the large feature vector into a smaller representation. It then uses ReLU and dropout to improve learning stability. The final linear layer outputs two logits. This structure al-

lows the model to use its deep pretrained features while learning the specific differences between Normal and TB images.

ResNet50: Resnet50 is another pretrained model that uses residual connections. These connections allow the signal to skip layers which makes training more stable. They prevent gradients from shrinking as they move backward through the network. This helps the model learn deeper patterns in the image. Each block in ResNet50 learns features from the input and adds them to the original signal which gives the network a strong ability to capture fine details.

The original ResNet50 classifier is removed and replaced with a new two class classifier. This classifier has a fully connected layer that reduces the feature vector. ReLU is used to add non-linearity. Dropout is used to lower the chance of overfitting. The final fully connected layer outputs class scores. This lets ResNet50 transfer its strong pretrained knowledge to the TB dataset.

Streamlit Website: A Streamlit application is built to demonstrate how the models can be used in practice. The app allows a user to upload an X-ray and select one of the three models. The image is processed with the same steps used during training so the output stays consistent. The chosen model predicts the class and displays the result with a confidence score. This gives a clear view of how the model behaves outside the training environment.

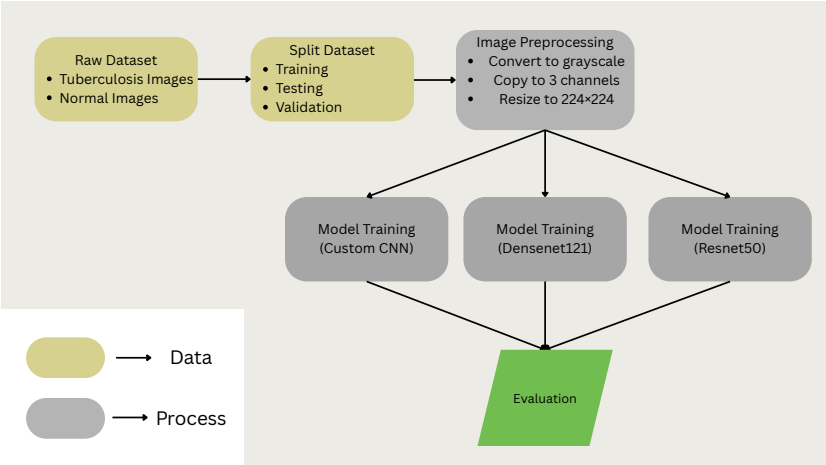


Figure 2: Data architecture for Custom TB detection model.

4.3 Training

The models are trained on the processed dataset using a consistent training pipeline. Each model receives the same input format so the comparison is fair. The process begins with images being loaded in small batches.

These batches help the model learn patterns without using too much memory. The images are passed through the model which produces two output values. These values represent the confidence for each class.

The model uses cross entropy loss which compares the predicted values to the true labels. This loss tells the model how wrong it is on each batch and gives a direction to improve. The Adam optimizer updates the weights after every batch. Adam is chosen because it adapts its learning rate which helps the model train smoothly on this dataset.

Each training run lasts for a fixed number of epochs. An epoch means the model has seen every training image once. The model evaluates itself on the validation set after each epoch. This step checks if the model is learning general patterns rather than memorizing the training images. The validation loss guides the saving of the best checkpoint. The checkpoint stores the model weights that achieve the highest accuracy on the validation set. This ensures that the final model used for testing is the best version seen during training.

Training is done on a CPU environment which slows the learning speed. The batch sizes and image sizes are kept moderate so the process remains stable. No early stopping is used because the checkpoint system already prevents overfitting. The training loop also collects losses and accuracies for both the training and validation sets. These values allow the creation of learning curves which help analyze how each model behaves over time.

The custom CNN trains from scratch. DenseNet and ResNet begin with pretrained weights and only the classifier is new. This difference changes how each model learns. The pretrained models start with very strong feature extraction which means they converge faster. The custom CNN must learn every feature from the ground up which takes more time. The training procedure remains the same across all models so the comparison stays fair.

5 Evaluation Metrics

The evaluation metrics measure how well each model performs on the unseen test set. These metrics help determine if the model can identify tuberculosis in new images. The main goal of the evaluation is to show the strengths and weaknesses of each model in a clear and simple way.

Accuracy is used as the first metric. Accuracy shows the percentage of test images that the model labels correctly. This value gives a quick view of overall performance. Accuracy alone is not enough for medical tasks because a model can score high accuracy by predicting the majority class often. The dataset has more normal images than TB images which means accuracy must be supported by other measures to avoid misleading results.

Precision is used to measure how many of the images predicted as TB are actually TB. This helps detect if the model gives too many false alarms. A high precision score means the model avoids labeling normal images as TB. This is important because false positives can cause unnecessary worry for a patient.

Recall is used to measure how many true TB images the model detects.

This value shows how sensitive the model is to the disease. A high recall score means the model finds most of the TB cases which is important for medical screening. Missing a positive case is dangerous therefore recall must be strong.

The F1 score combines precision and recall into one number. It gives a balanced measure when the dataset has uneven class sizes. This score helps show how well the model performs when both precision and recall are important. The F1 score is useful for TB detection because both false positives and false negatives can cause harm.

A confusion matrix is also created for each model. The confusion matrix shows the number of correct and incorrect predictions for each class. This gives a detailed view of the types of mistakes that the model makes. A separate normalized confusion matrix is also generated which shows the same information in percentages. This makes it easy to compare model performance even when the classes have different counts.

A classification report is generated for each model. This report contains precision, recall, and F1 scores for both classes. It gives a full summary that supports the confusion matrix. These evaluation results help guide the comparison between the three models and show how well each model handles the TB detection task.

6 Results

Model	**Accuracy**	**Precision (N)**	**Precision (TB)**
Custom CNN	0.9954	0.9950	0.9958
DenseNet121	0.9548	0.9341	0.9843
ResNet50	0.9493	0.9807	0.9146

Table 1: Accuracy, precision, and recall for all three models. N = Normal. TB = Tuberculosis.

Model	**F1 (Normal)**	**F1 (TB)**	**Recall (N)**	**Recall (TB)**
Custom CNN	0.9958	0.9947	0.9966	0.9937
DenseNet121	0.9606	0.9470	0.9883	0.9128
ResNet50	0.9531	0.9450	0.9270	0.9771

Table 2: F1 scores and Recall for the Normal and Tuberculosis classes for each model.

6.1

Model	True Normal	False Normal	False TB	True TB
Custom CNN	601	2	3	479
DenseNet121	596	7	42	440
ResNet50	559	44	11	471

Table 3: Confusion matrix counts for each model. True values represent correct predictions while false values represent misclassifications.

Custom 9 Layer CNN The custom CNN achieves the highest overall performance among the three models. The model reaches a test accuracy of 99.54% which shows that it rarely makes mistakes. The precision for the normal class is 0.9950 and the recall is 0.9966 which means the model correctly identifies almost every normal image. The precision for the TB class is 0.9958 and the recall is 0.9937 which shows that the model also detects TB cases very well.The confusion matrix confirms this strong performance. The model predicts 601 normal images correctly and misclassifies 2. The model also predicts 479 TB images correctly and misclassifies 3. The normalized matrix shows values of 1.00 for normal and 0.99 for TB which means the model maintains high precision and recall in both classes.

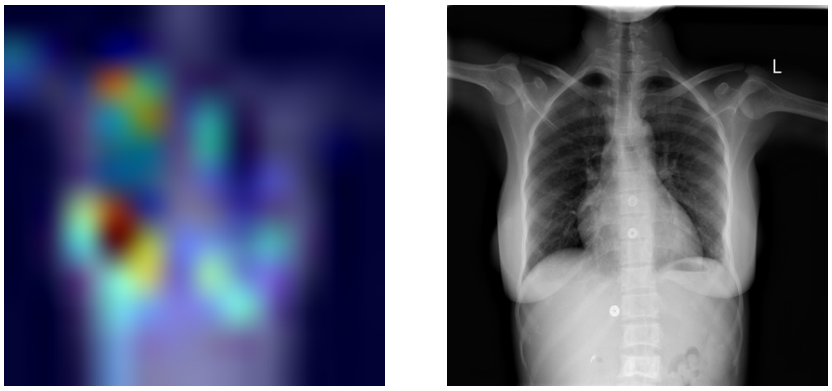
6.2 DenseNet121

DenseNet121 achieves a test accuracy of 95.48%. The precision for the normal class is 0.9341 and the recall is 0.9883 which means the model identifies most normal images but produces more false positives. The precision for the TB class is 0.9843 and the recall is 0.9128 which means the model detects most TB cases but still misses some. These values come from the DenseNet metrics file. The confusion matrix shows the model predicts 596 normal images correctly and misclassifies 7. The model predicts 440 TB images correctly and misclassifies 42. The normalized matrix shows 0.99 for normal and 0.91 for TB which means the model struggles more with TB detection. DenseNet121 performs well on normal images but it shows a drop in TB recall. This pattern suggests the model may rely heavily on broad features that do not always capture the subtle patterns present in TB cases.

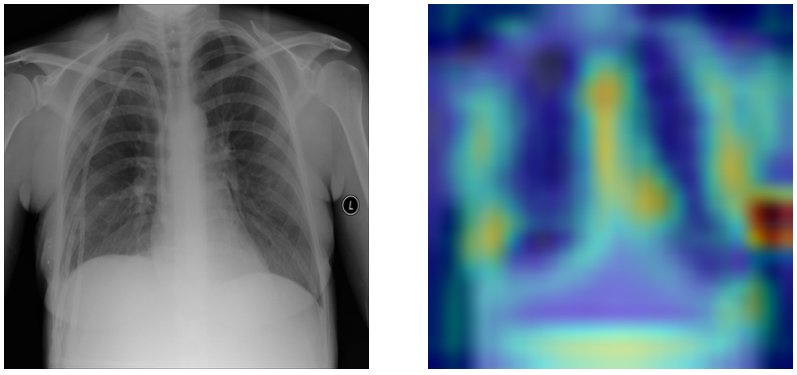
6.3 ResNet50

ResNet50 achieves a test accuracy of 94.93%. The precision for the normal class is 0.9807 and the recall is 0.9270 which shows the model has fewer false positives but misses more normal images. The precision for the TB class is 0.9146 and the recall is 0.9771 which means the model detects TB cases well. These values come from the ResNet metrics fil. The confusion matrix shows the model predicts 559 normal images correctly and misclassifies 44. The model predicts 471 TB images correctly and misclassifies 11. The normalized matrix shows 0.93 for normal and 0.98 for TB which means the model is more sensitive to TB cases than DenseNet121. ResNet50 performs well when identifying TB images because it captures deep features that match patterns seen in TB. It performs weaker on normal images which indicates a tendency to over detect TB.

6.4 Grad-Cam Inconsistency



Resnet50 Grad-CAM image Vs Test Image



Custom CNN Grad-CAM image Vs Test Image

Grad-CAM is used to understand which regions of the X-ray influence each model's prediction. The method works by looking at the gradients inside the last convolutional layer. These gradients highlight the parts of the image that have the strongest impact on the final decision. The resulting heatmap is then placed on top of the X-ray so the important regions become visible to the user. In my case, the Grad-CAM works on the custom CNN and ResNet50 models, but it does not work on DenseNet121 because my hardware does not support it. This limitation prevents a full comparison across all models in this aspect. Furthermore, the overall quality of the Grad-CAM images is low. This issue is caused by the small input resolution and the limited capacity of the hardware. Low resolution reduces the clarity of the heatmap because the model cannot form sharp spatial details. The heatmaps therefore appear blurry. The low

quality makes it difficult to draw significant conclusions from these visualizations. However, they are still available to view in the website along with the prediction.

6.5 Discussion

Now that we have our results we can begin to draw some connections and understand some shortcomings. The extremely high scores of the custom CNN are unusual for a medical imaging task because models with simple architectures rarely outperform deep pretrained networks on complex patterns found in X-rays. This raises a clear concern about generalization because the model may be overfitted in a way that inflates performance. An overfitted model memorizes repeated patterns that appear often in the dataset which means it performs well on similar images but fails to understand the real features of tuberculosis. The dataset contains many images with similar lighting and structure which can push the model to learn surface level cues instead of disease features. This can create accuracy that looks strong but does not reflect real clinical behavior. The transfer learning models show lower accuracy which matches expectations because this task is difficult and usually requires deeper feature extraction to handle subtle medical patterns. DenseNet121 performs well on normal images but shows a weaker result on TB images. This difference suggests that DenseNet focuses on broad structural features within the X-ray. These features may be more common in normal images which can make TB detection harder for this model. DenseNet therefore misses a notable group of TB cases. ResNet50 shows a different pattern. The model detects TB cases with strong accuracy but misses more normal images. This outcome suggests that ResNet focuses on fine details that often appear in TB cases. These details may also appear in some normal images which leads to false positives. The model becomes more sensitive to TB but loses balance between the two classes. This pattern highlights the challenge of detecting subtle disease features with high consistency. These differences suggests that the custom CNN performance should be interpreted with caution.

6.6 Future Work

Future work will focus on improving the models and addressing the limits found in this study. The first step is to expand the dataset because medical models need a wide range of examples to learn features that generalize well. A larger dataset would reduce the chance of inflated accuracy and would help the models handle more variation in lung structure and image quality. Stronger data augmentation should also be added because real clinical images show many lighting and positioning differences. The custom CNN can also be improved by adding more layers or by changing the filter sizes. These changes can help the model learn deeper features from the images. Future work can also test regularization methods that lower the chance of overfitting. These steps would help confirm if the model can maintain strong results on new data. The use of Grad-CAM should also be improved. The heatmaps are low quality because the current

environment limits the resolution of the visualization. The models should be tested on hardware with better support for interpretability tools. The Grad-CAM procedure should be run on higher resolution images so the highlighted regions become clear. A model used in medical tasks must show that it focuses on the correct lung areas. Better interpretability would allow stronger trust in the decisions made by the models. Finally, The web application can be expanded with features that support clinical use. These features can include multi image comparison and user feedback options. The app can also be adapted to handle new models with minimal changes. This would let future work test the pipeline quickly and safely.

7 Conclusion

This project builds an end to end system for tuberculosis detection using three deep learning models. The results show that the custom CNN performs the best on the test set although this performance may not reflect real world behavior. DenseNet121 and ResNet50 show different strengths which highlights the value of comparing multiple architectures. The evaluation process and the Grad-CAM analysis reveal that the models need improvement before they can handle real clinical data. The project demonstrates a full workflow that includes data preparation, model training, evaluation, and deployment through a web app. This work provides a strong base for future improvements and shows how deep learning can support medical image analysis when used with care and critical review.

References

[1] World Health Organization. Tuberculosis fact sheet, n.d.

[2] M. Singh, G. V. Pujar, S. A. Kumar, M. Bhagyalalitha, H. S. Akshatha, B. Abuhaija, A. R. Alsoud, L. Abualigah, N. M. Beeraka, and A. H. Gandomi. Evolution of machine learning in tuberculosis diagnosis: A review of deep learning-based medical applications. *Electronics*, 11(17):2634, 2022.

[3] A. Wong, J. R. H. Lee, H. Rahmat-Khah, A. Sabri, and A. Alaref. Tb-net: A tailored, self-attention deep convolutional neural network design for detection of tuberculosis cases from chest x-ray images. arXiv preprint arXiv:2104.03165, 2021.

[4] V. Sharma, S. Nillmani, S. K. Gupta, and K. K. Shukla. Deep learning models for tuberculosis detection and infected region visualization in chest x-ray images. *Intelligent Medicine*, 4(2):104–113, 2024.

[5] S. Jaeger, A. Karargyris, S. Candemir, L. Folio, J. Siegelman, F. Callaghan, X. Zhiyun, K. Palaniappan, R. K. Singh, S. Antani, G. Thoma, W. Yi-Xiang, L. Pu-Xuan, and C. J. McDonald. Automatic tuberculosis screening using chest radiographs. *IEEE Transactions on Medical Imaging*, 33(2):233–245, 2014.

[6] M. Norval, Z. Wang, and Y. Sun. Pulmonary tuberculosis detection using deep learning convolutional neural networks. In *Proceedings of the 2019 3rd International Conference on Video and Image Processing, ICVIP 2019*, ACM International Conference Proceeding Series, pages 47–51. Association for Computing Machinery, 2019.

[7] C. J. Liu, C. C. Tsai, L. C. Kuo, et al. A deep learning model using chest x-ray for identifying tb and ntm-ld patients: A cross-sectional study. *Insights into Imaging*, 14:67, 2023.

[8] S. Kant and M. M. Srivastava. Towards automated tuberculosis detection using deep learning. In *Conference Paper*, n.d.

[9] S. Hansun, A. Argha, S. T. Liaw, B. G. Celler, and G. B. Marks. Machine and deep learning for tuberculosis detection on chest x-rays: Systematic literature review. *Journal of Medical Internet Research*, 25:e43154, 2023.

[10] L. Lin-Sheng, L. Yang, L. Zhuang, Y. Zhao-Yang, Z. Wei-Guo, and W. P. Gong. From immunology to artificial intelligence: Revolutionizing latent tuberculosis infection diagnosis with machine learning. *Military Medical Research*, 10:1–37, 2023.

[11] Kaggle. Chest x-ray lungs segmentation, n.d.

[12] Kaggle. Tuberculosis (tb) chest x-ray dataset, n.d.