Project: COVID-19 Assisted Diagnosis

Frie Van Bauwel[1] Lotte Van de Vreken[2] Marcin Jedrych[3] Xueting Li[4]

[1,2,3,4] Master of Science in statistical data analysis - computational statistics

I. Introduction

Within this project on medical image classification, two deep learning models are trained to classify X-ray images of lungs as COVID-positive or COVID-negative images. Further, the Grad-CAM visualization technique is used to be able to show on which areas of an X-ray image are mainly used by the algorithm to decide whether an image shows a person affected with COVID or a person not affected with COVID. The images used in this project come from [1] and [2].

This report describes first how the data are explored, pre-processed and augmented (section II). It then explains how a baseline model, using a convolutional neural network, was fitted (section III) and how transfer learning was used to make use of the pre-trained ResNetV2 model (section IV). Section V describes in which the Grad-CAM visualisation is used to understand why missclassifications happen in the best performing model trained in the previous tasks. After the conclusion (section VI), the report describes how this project was divided between all group members (section VII). The final section entails our use of generative AI during the project (section VIII).

II. Task 1: Data Exploration, Pre-Processing and Augmentation

As a first step, a data exploration was carried out. The dataset consists out of 1600 training images, 400 validation images and 200 test images. The amount of training images can be considered as quite limited, hence a risk of overfitting exists. Other potential challenges for automatic classification that become clear with having a first look at the images are the subtle differences in COVID versus normal chest X-rays, as well as the variability in brightness, contrast, and the exact part of the lungs shown on the X-ray between pictures. Both challenges are addressed by using normalization and augmentation techniques as described below.

Based on the pixel statistics and label distribution, the datasets appear to be fairly uniformly divided. The mean values and standard deviations of the pixel values are comparable across the sets: training (mean = 0.53, std = 0.25), validation (mean = 0.55, std = 0.25), and test (mean = 0.56, std = 0.26). This finding indicates that the image intensity is consistent. The class distribution in all three sets is also well balanced, as there are an equal amount of COVID compared to the amount of normal images in all three distinct data sets. There are no consistent differences in image quality between the COVID and normal X-ray images. However, it can be observed that some pictures have a much higher intensity distribution than others, creating the need for proper normalization. In some pictures, artefacts like wires could be seen, which can lead to lower model performance.

To reduce the training time and memory usage, the pictures were downsampled from 299x299 to 128x128 resolution. This allows for faster runtimes without losing too much information. Additionally, it makes the picture more general thus reducing the risk that the model will overfit on small artefacts in the pictures. The images were normalized using dataset statistics: the mean and standard deviation of all pixels in the training, and validation dataset were used as normalization statistics. The X-ray images contain black-and-white images thus all three image channels contain the same information and could be normalized using the same statistic.

To augment the data, pictures were slightly shifted in width, rotated with a maximum range of 15 degrees and zoomed with a maximum factor of 0.2. These augmentation patterns were chosen because they represent still realistic X-ray variations, as patients can be positioned slightly more on the side, can be a bit closer or further from the imager or can be slightly tilted. Other augmentations such as vertical or horizontal flip would create unrealistic images that would never occur in real-life examples.

III. Task 2: Building the baseline model

In this task, a neural network including two convolutional layers, one fully connected layer and one dropout layer is trained. During this training, accuracy is used as a metric for model performance. Accuracy is as an appropriate metric as it compares the correctly classified COVID cases and normal cases to the sum of all cases. For the loss function for the model, binary cross-entropy was selected, as it is suitable for binary classification tasks like this one.

The result of the initial model training is shown in figure 2. Looking at the training and validation curves in the initial model's history plot, it is that the learning curve and validation curve are too variable. Therefore, the learning rate should be tuned. The learning rate, together with the optimizer, filters and dropout rate, are tuned using a grid search. The values used for the hyperparameters are given in table I. The filters are tuned to find the ideal scale to generalize the images, the values for this hyperparameter are based on values commonly seen in convolutional networks. The dropout rate was tuned to minimize the risk of overfitting, again, typical values were used. The two tried out optimizers are chosen because these were seen in the course. Finally, the learning rate was tuned to have a smoother learning curve, possible values were chosen

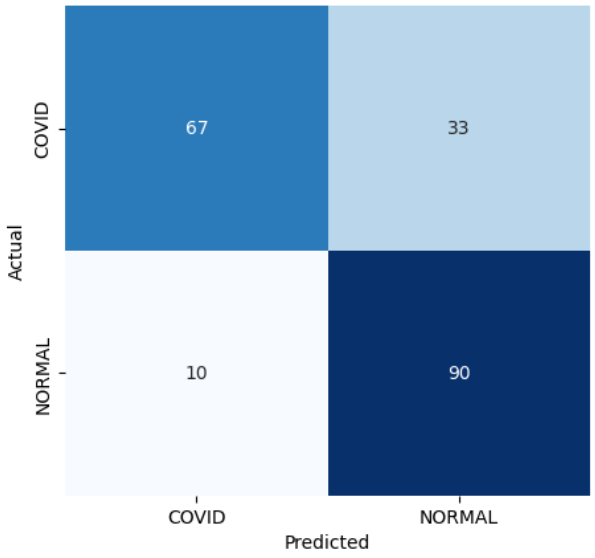

Fig. 1. Confusion matrix of the test data classified by the final baseline model.

based on the initial value. An early stopping mechanism with a patience of 5 is implemented during the grid search.

The best performing combination of hyperparameters turned out to be a filter size equal to 8, a dropout rate of 0.3, a learning rate of 0.001 with the Adam optimizer. The early stopping algorithm stopped this model training after 29 epochs and resulted in a validation accuracy of 0.89. Using these parameters, the model was trained for one final time on the combination of the training and validation data and the performance was evaluated on the test data. The results are shown in figures 3 and 1, the accuracy is equal to 0.79.

In the final model, the training curves indicated significant improvement, with the model reaching a higher accuracy and a more efficient drop in loss compared to the initial training. This suggests that the model has learned to generalize better and is more effective at predicting unseen data.

IV. Task 3: Transfer Learning

Within task 3, a neural network is built using the transfer learning approach. In other words, a pretrained model is used as a basis to built a model, hereby making it possible to use a network with many layers even though a limited set of training data is available. In this case, the ResNet50V2, a convolution neural network consisting of 50 layers that was originally trained on the ImageNet dataset, is used as the pretrained model [3]. After the ResNet50V2 model, four extra layers are added. First, a GlobalAveragePooling2D layer is added to reduce the spatial dimensions. Then, a 128-node dense layer with relu-activation is added in combination with a dropout layer to reduce risk of overfitting. A final single-node dense layer with sigmoid activation was used to get the binary output.

TABLE I
Overview of hyperparameter values used in the grid search during hyperparameter tuning both in task 1 and task 2. In both tasks, a true grid search was carried out, meaning that all possible combinations of the given values were tried out.

Tuned Hyperparameters	**Values Included in Grid Search**	
	Baseline Model	*Transfer Model*
Filters	[8, 32, 64]	Not tuned - not present in manually added part
Dropout Fate	[0.4, 0.3, 0.2]	[0.3, 0.4, 0.5]
Learning Rate	[1e-3, 1e-4, 1e-5]	[1e-2, 1e-3, 1e-4]
Batch Size	not tuned	[32, 64, 128]
Optimizer	[adam, sgd]	Not tuned - adam optimizer used

After setting the model architecture, the hyperparameters of the added layers are tuned using a general grid search. While tuning the hyperparameters, the pretrained parameters in the ResNet50V2-layers remain unchanged. Early stopping based on the validation loss is used with a patience of five steps. The tuned parameters consist of the batch size, learning, and dropout rates. Table I shows the parameter combinations used during the hyperparameter-tuning process. Based on the validation accuracy of each combination tried out during the grid search, the combination of a batch size equal to 64, a dropout rate of 0.3 and a learning rate of 1e-3 is the best performing combination. The accuracy of the model on the validation set was then equal to 0.81, and the model was early stopped after 29 epochs. After this tuning step, a model is trained during 29 epochs based on the combination of the training and validation data. Another, final, model is trained in which the ResNet50V2-layers are unfrozen, hence fine-tuning the weights in these layers is possible, again 29 epochs were used here. The training curves of the model before unfreezing the ResNet50V2-layers is visualized in figure 4, the training curves of the model after unfreezing is visualized in figure 5. The training curves of both trained models within this task look smooth and have a shape coming close to the ideal training curve. The model where the ResNet50V2 model parameters are finetuned shows a higher accuracy and lower loss (respectively 0.99 and 0.02 in the fully trained model) than the model without the extra finetuning (respectively 0.88 and 0.26 in the fully trained version). Adjusting the learning rate to avoid a performance dip during finetuning the model was not necessary to obtain these results. This hints towards better performance of the finetuned model, but might also indicate a bigger chance of overfitting on the training dataset (consisting of the provided training and validation set). As requested in the task, no training of this model was done using the validation set as holdout-samples, hence this is hard to judge from these curves alone. Comparing these curves to the training curve of the final baseline model produced in task 2, it is clear that the curves have similar smoothness, but that the final accuracy and loss of the training curve of the baseline model (i.e. 0.76 and

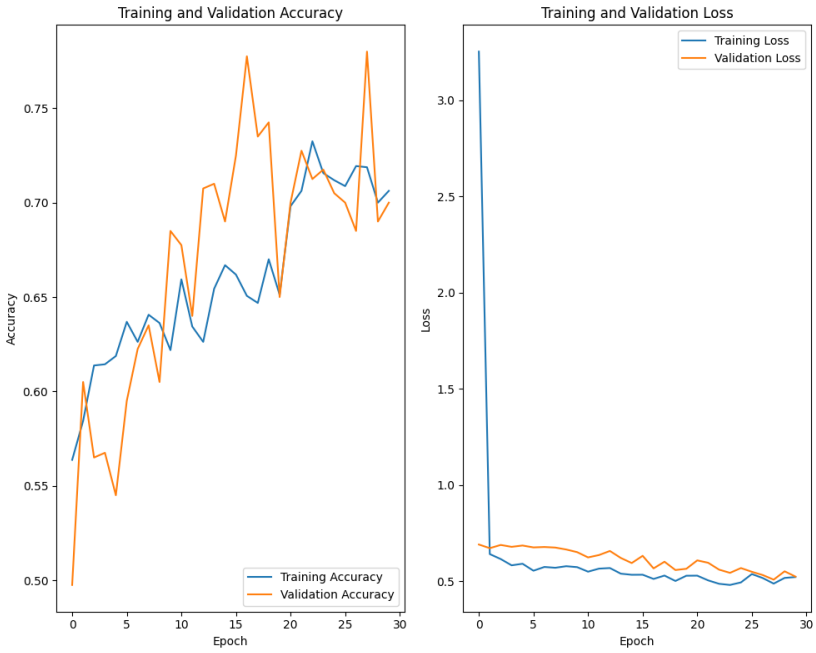

Fig. 2. The training and validation curves of the loss and accuracy of the initial baseline model before the hyperparameter tuning.

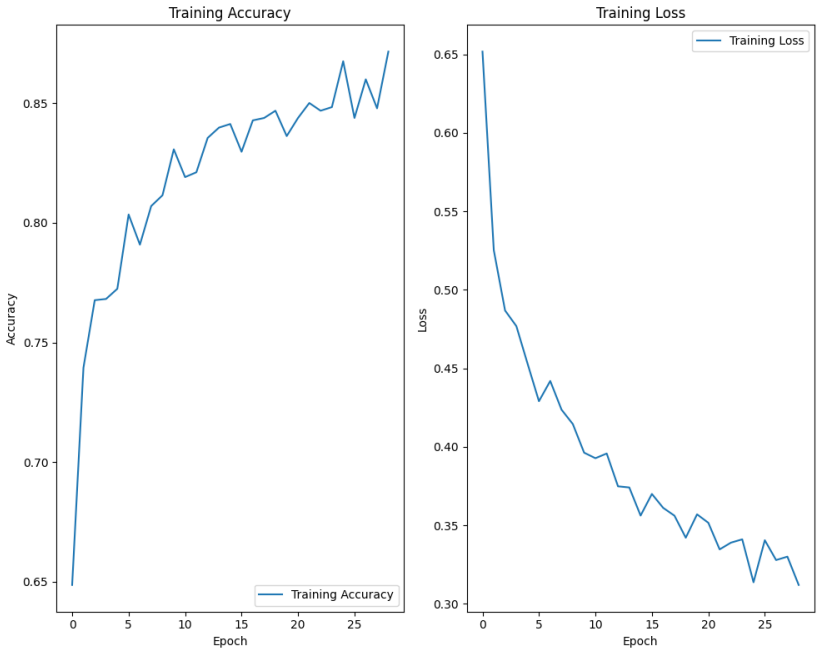

Fig. 3. The training curves of the loss and accuracy of the final baseline model where the model was trained based on the combination of validation and training data.

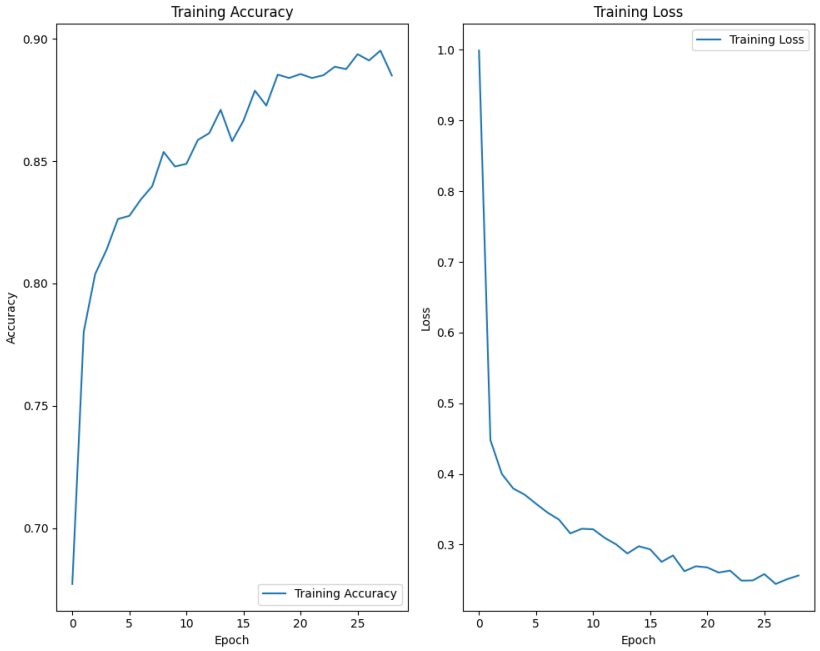

Fig. 4. The training curves of the loss and accuracy of the final transfer model with tuned hyperparameters, but the original weights of the ResNet50v2, where the model was trained based on the combination of validation and training data.

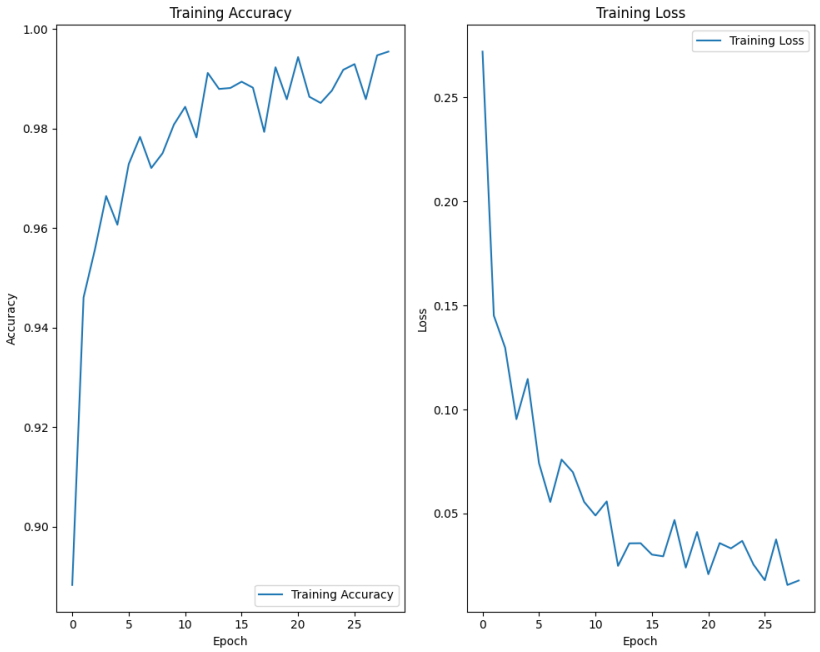

Fig. 5. The training curves of the loss and accuracy of the final transfer model with tuned hyperparameters and weights of the ResNet50v2 where the model was trained based on the combination of validation and training data.

0.41 respectively) are worse than those for the training curves of the model produced in task 3.
As a final step in this task, the finetuned model is evaluated using the test set. Both the confusion matrix (6), where four more images are correctly classified as COVID but 10 less images correctly classified are normal compared to the confusion matrix of the baseline model, and the calculated accuracy of 0.76 show a slightly worse result for this model than for the baseline model.

This result differs from the initial expectations. The transfer model was expected to have an increased performance with only limited increased learning time compared to the base model due to the addition of the pretrained ResNet50v2 model. Instead, the result is a model showing very promising results on the training set with strongly decreased test results. This implies overfitting, which can be due to the relatively limited amount of training data compared to the model complexity.

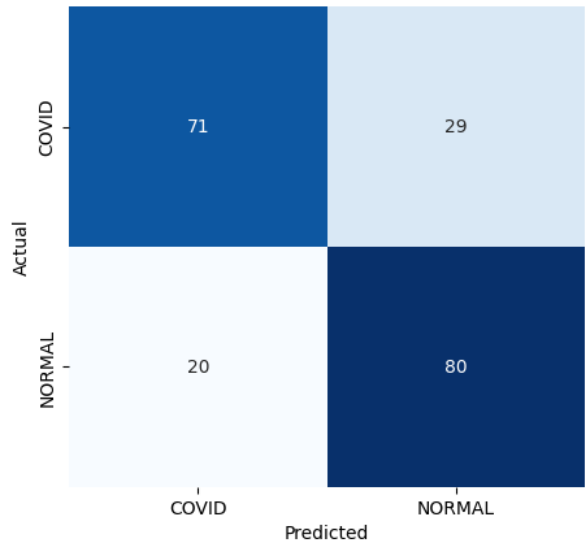

Fig. 6. Confusion matrix of the test data classified by the final transfer model (with tuned hyperparameters and fine-tuned weights of the ResNet50v2 model).

Different ways to alleviate the overfitting problem exist. One option is to use the transfer model before the finetuning of the ResNet50v2 model weights, as this provides a more general model less tailored towards the training data. Other options are introducing bigger transformations in the training set, or adding more training data if possible. Alternatively, starting with a lower resolution can also help to generate a more general model with a lower overfitting risk. Finally, the pre-trained model forming the basis of the transfer model can be changed to a model specifically trained on medical images, like the CheXNet model [4].

V. Task 4: Explainability through Grad-CAM

Grad-CAM (Gradient-weighted Class Activation Mapping), introduced by Selvaraju et al. in 2017, is a technique developed to provide visual insight into the decision-making process of convolutional neural networks (CNNs) [5]. CNNs are powerful models for image classification, but they are often criticized for their ”black box” nature—producing accurate predictions without a clear explanation of how those predictions were formed. Grad-CAM addresses this issue by generating heatmaps that highlight the most relevant regions of an input image with respect to a specific class prediction. These heatmaps are constructed using the gradients of the target class scores with respect to the activations in the final convolutional layers, effectively showing which parts of the image influenced the model's decision. This interpretability is particularly important in domains such as medical imaging or autonomous driving, where understanding why a model made a decision is just as critical as the decision itself. Gradient-based techniques have since become a widely used approach in explainable AI (XAI) [6].

For our project, we followed the Keras tutorial [7] that used a CNN with multiple output classes. As the baseline model outperformed the final transfer function, the Grad-CAM is implemented for the baseline model. We initially adjusted our model to do the same for two classes (COVID and Normal). However, it was discovered later that the algorithm itself could be slightly changed to make it work with a single scalar logit as output, representing the confidence for one class (e.g., COVID). We found that Grad-CAM could be applied by interpreting the scalar logit as the evidence for one class and its negative as the evidence for the other class (since the sigmoid output ensures that the total probability across both classes sums to one). This insight allowed us to use Grad-CAM effectively for a scalar-output binary classification model. Having two class labels (COVID and Normal) or only one scalar output does not affect predictions or interpretability of the Grad-CAM explanations. Whether the model produces two logits or a single scalar, the highlighted regions will be the same, as the underlying decision process remains equivalent.

The Grad-CAM visualizations revealed that, when looking at the COVID-activation, the model focuses particularly on the lung regions, which are medically relevant. It appears to detect opacities or structural anomalies consistent with COVID-19. However, the technique also helped us uncover potential model biases. In some cases, the Grad-CAM showed focus on irrelevant features such as cables or background artefacts, suggesting the model may have learned spurious correlations from the training data. Figure 8 is such an example. This figure also shows a less outspoken activation for the Normal-class, which is generally the case compared to the COVID-activations. Moreover, Grad-CAM proved useful in understanding misclassifications. In several incorrect predictions, we observed that the model was influenced by visual noise such as unusual patient posture, belonging f.e. to infants. If this would have appeared more frequently in the opposite class, the model may have learned to associate these irrelevant features with that class, leading to incorrect predictions.

Based on these insights, if we were to collect more data and retrain the model, we would prioritize minimizing visual biases by standardizing patient positioning, removing non-diagnostic

artifacts (e.g., cables), and ensuring a balanced dataset across age and other demographic factors. These steps would help the model focus more effectively on clinically meaningful features, improving both accuracy and trustworthiness.

Fig. 7. Example Grad-CAM heatmap highlighting lung regions associated with a COVID-positive classification.

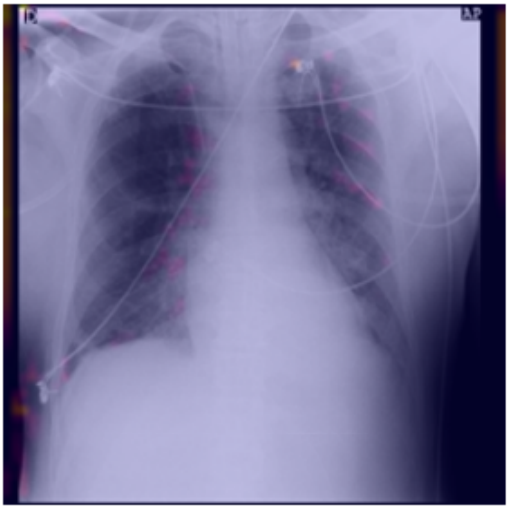

Fig. 8. Example of a Grad-CAM heatmap of a COVID-positive case misclassified as normal case. The lower overall activation for the normal classification is clear here. The little activation that is present, concentrates itself around the artifacts, in this case a part of the wires and it's attachments, and the box in the left upper corner.

VI. Conclusions

In this project, we aimed to classify lung X-ray images as COVID-positive or normal images. Two approaches were used: a custom convolutional neural network tailored to the dataset, and a transfer learning approach using the ResNet50V2 architecture. Both models reached good accuracy with the custom model performing the best (test accuracy = 0.79). This suggests that, for the limited dataset used, a lightweight model specifically trained on lung X-ray images is more effective than a deep pretrained model designed for general-purpose image recognition. To gain insights into model decision making, Grad-CAM was implemented on the custom model. The visualizations showed that for correctly classified images, the model focuses on the lung regions, indicating meaningful feature extraction. For falsely classified images, artefacts such as cables or unusual patient posture were highlighted, underscoring the need for high-quality, artefact-free and balanced datasets in medical image analysis. These findings underscore the value of domain-specific model design and data curation in developing reliable AI tools for healthcare.

VII. Author contributions and collaboration

In first instance, the tasks were divided as follows:

- Task 1: Marcin Jedrych and Frie Van Bauwel,
- Task 2: Marcin Jedrych and Xueting Li,
- Task 3: Lotte Van de Vreken and Frie Van Bauwel,
- Task 4: Marcin Jedrych and Lotte Van de Vreken.

According to this division, everyone made sure there was a first version of the code for their assigned task. Afterwards, questions and unclarities in each task were discussed, after which everyone checked and complemented the first draft of the code of each task. Task 4 was kept until the other tasks were almost done. For the text, first an answer to the questions was formulated by the assigned people to each task, after which everyone Read them and complemented where the need was felt. Some questions were discussed at length before an answer was formulated.

VIII. Use of Generative AI

Generative AI was used to fix bugs in the code for all tasks except task 1. It was also used to get a first version of some parts of the code to plot the images for all tasks except task 1. Suggested code was however always looked at critically and never taken over one-on-one.

References

[1] M.E.H. Chowdhury, T. Rahman, A. Khandakar, R. Mazhar, M.A. Kadir, Z.B. Mahbub, K.R. Islam, M.S. Khan, A. Iqbal, N. Al-Emadi, M.B.I. Reaz, M. T. Islam,"Can AI help in screening Viral and COVID-19 pneumonia" IEEE Access, Vol. 8, 2020, pp. 132665 - 132676.

[2] Rahman, T., Khandakar, A., Qiblawey, Y., Tahir, A., Kiranyaz, S., Kashem, S.B.A., Islam, M.T., Maadeed, S.A., Zughaier, S.M., Khan, M.S. and Chowdhury, M.E., "Exploring the Effect of Image Enhancement Techniques on COVID-19 Detection using Chest X-ray Images". 2020. [Online]. Available: https://arxiv.org/abs/2012.02238

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Identity Mappings in Deep Residual Networks". 2016. [Online]. Available: https://arxiv.org/abs/1603.05027

[4] P. Rajpurkar et al., "CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning". 2017. [Online]. Available: https://arxiv.org/abs/1711.05225

[5] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization," in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), 2017, pp. 618–626.

[6] M. Mersha, K. Lam, J. Wood, A. K. AlShami, and J. Kalita, “Explainable artificial intelligence: A survey of needs, techniques, applications, and future direction,” *Neurocomputing*, vol. 599, p. 128111, Sep. 2024.
[7] Keras Team, “Grad-CAM class activation visualization,” *Keras Documentation*, https://keras.io/examples/vision/grad_cam/, accessed May 7, 2025.