Detecting COVID-19 indicators in lung X-rays using transfer learning and gradient-based localization

Jón R. Balvinsson
jrba@kth.se

Hólmfríður Elvarsdóttir
helv@kth.se

Kristmundur Á. Jónsson
krisjon@kth.se

Leifur D. Sigurðarson
ldsi@kth.se

Abstract

In early December 2019, patients with pneumonia of an unknown cause were first reported in the city of Wuhan in China. These cases have now been linked to the infectious disease COVID-19 which has since been declared a global pandemic placing tremendous stress on health-care systems around the world. A large factor in fighting the spread of the virus is finding efficient ways to diagnose patients, preferably with minimal contact between patients and health-care workers. Researchers all around the world have been finding clever ways to utilize AI tools for diagnostic purposes. The objective of this paper is to study if it is possible to design a deep learning model to be used as a reliable detector for COVID-19 indicators in lung X-ray images.

1 Introduction

The novel coronavirus, also known as COVID-19, is an infectious disease that primarily affects the lungs. This paper aims to analyze how transfer learning can be used with a Convolutional Neural Network (CNN) to implement a diagnostic tool for detecting COVID-19 in lung X-ray images. CNNs are commonly used for image classification and recognition because of their high accuracy. The network that we propose will use weights from a pre-trained model as well as retraining the last layer of the network on COVID-19 images. In order to get a deeper understanding of how the network locates the COVID-19 indicators in the X-ray images, we use Gradient-weighted Class Activation Mapping (Grad-CAM) [1], a gradient-based localization technique. Grad-CAM, as a localization technique, is specifically suited to a family of CNN architectures and thus fitting for our purposes. The localization allows us to determine the areas in a given image that influence the discriminative process in our network and allows for an empirical analysis of what areas contribute to specific classes in our data set. We model the problem as two binary classification problems, one with classes COVID-19 and Non-COVID-19 and another with classes COVID-19 and Pneumonia. The data set consists of 247 COVID-19, 4273 pneumonia, and 1583 healthy X-ray images.

2 Related work

Currently, there are two main types of methods to detect COVID-19. The most common method is detecting the presence of the virus with various swab tests such as RT-PCR testing, which is a precise detection method but not always accurate. If the test results are positive, it is almost certain that the patient has COVID-19, but there is still a 30% chance of a false negative [2]. An alternative method is detecting indicators of the virus in lung X-rays. Similarly to the swab tests, there is also a chance of false negatives with this method since developing lung symptoms is one of the more severe

Project report for DD2424

complications of COVID-19. Many patients who exhibit milder symptoms will not have any visible signs of COVID-19 in their lung X-ray images [3]. Even so, the analysis of the X-ray images is being performed by radiologists daily. The diagnosis can often be difficult to confirm, which leaves room for human error since the COVID-19 indicators can vary on how prominent they are one the scans. Additionally, this process can be very time-consuming, so it gives the incentive to find alternative solutions to speed up and increase the accuracy of these images.

Since the COVID-19 virus first surfaced, researchers have paved the way for developing more intelligent and efficient detection methods. There have been many efforts to produce AI-based COVID-19 detection tools using Deep Learning on both X-ray and CT images. The results of these efforts have been promising with regards to accuracy and efficiency in detecting COVID-19 through radiography imaging. Currently, the main issue is the lack of available X-ray data online from COVID-19 patients, so most of the networks presented have only trained on a small data set compared to general sizes of data sets used for training CNNs in general. One of the most recent and promising networks is the open-source COVID-Net developed by Linda Wang, Zhong Qiu Lin, and Alexander Wong. COVID-Net is a CNN designed to detect COVID-19 in chest X-ray images [4]. The data set that they put together is composed of 13,975 CXR (Chest X-ray) images, where approximately 358 of those images are from COVID-19 patients. The published results from their experiments show that COVID-Net produced a 93.3% test accuracy. The network was trained on 13675 images and tested with 100 images from each class; COVID-19, Pneumonia, and Healthy. COVID-Net is transparent in the way that it produces a visual aid to validate that the network is making decisions based on the relevant COVID-19 indicators. Due to the lack of available data and the fact that COVID-Net is quite a new tool, it is by no means a production-ready solution, but it most certainly shows much promise for the future.

3 Data

The data we used was from the publicly available data sets "COVID-19 image data collection" from GitHub and "Chest X-ray Images (Pneumonia)" from Kaggle. We retrieved chest X-ray images of patients from these data sets from the following classes: Bacterial pneumonia, Viral Pneumonia, COVID-19, and Healthy. This combined data set only contains 247 samples of COVID-19 classed images, so to generate more images, we performed data augmentation.

To augment the photos the ImageDataGenerator package from Keras was used. We allowed shifting of 20% in each direction, zooming in by 10%, and rotating the image up to 50 degrees in the augmentation process. We did not allow further commonly used augmentation, such as flipping the images, ZCA whitening or playing around with the brightness due to the complicated nature of X-ray images. All the COVID-19 images used for training were augmented once, which allowed us to double the amount of COVID-19 images that we have for training. The images used for validation and testing are not augmented. Table 1 shows the data we collected with respect to the data source.

Table 1: Data with respect to source

	Nr. of data points		
Source	COVID-19	Pneumonia	Healthy
Github	247	0	0
Kaggle	0	4273	1583
Augmented	142	0	0

Our data set is flawed since many of the images have artifacts such as arrows, letters, or symbols placed on the images. In most cases, the artifacts are indicators of disease, and we must take care that they do not skew the results, as our network may use them to classify the images. Since there is such a shortage of openly available COVID-19 X-ray images, we must try to work with what we have.

The images in the data set are of many different sizes, as can be seen in Figure 1, which shows the distribution of the resolution of the images. We chose to scale the images to a resolution of 224×224 since this size gives a good performance on the data indicating that the down-scaling does not cause noticeable information loss and minimizes the need for upscaling images. By decreasing the sizes of the images, the computing time is also reduced.

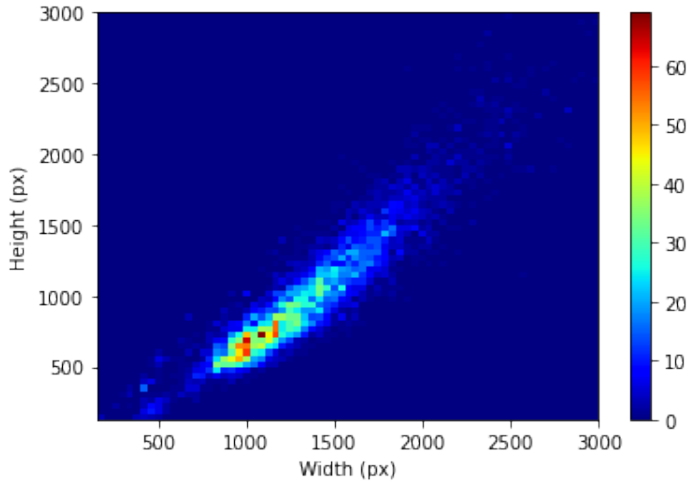


Figure 1: Distribution of image resolution from collected data. The majority of the images have a resolution greater than 500×500.

4 Methods

The network that we propose is a CNN that will use weights from a pre-trained model on the ImageNet data set. The final layer of the network is trained on our X-ray data set, and we use GradCAM to analyze the results. We model the problem as a binary classification problem where the classes are either COVID-19 and Non-COVID-19 or COVID-19 and Pneumonia. The following sections go into further detail on how we designed our network.

4.1 Image classification with CNNs

Detecting COVID-19 cases from X-ray images is considered an image classification problem. Convolutional neural networks (CNN's) have been leading the field in image classification problems over the past few years, making them the obvious candidate to tackle this problem. CNN's are different from conventional feed-forward neural networks due to their convolutional layers and pooling layers. A CNN has many of these convolutional and pooling layers that allow the network to learn feature maps from the input. The network also has a fully connected output layer to give the final probabilities for each class. One of the main difficulties of using CNNs is balancing the trade-off between the size of the network and the time it takes to train the network. A larger model gives better results than a smaller model, but given limited time and computation power, a trade-off needs to be made if we want to train a new model from scratch. For this reason, we decided on using transfer learning.

4.2 Transfer Learning

Transfer learning involves taking a pre-trained model for a specific task and partially or entirely reusing it in a different model designed to solve another task. Therefore, the pre-trained model acts as the expert in our system for which we wish to learn how to exploit. We discard the final layer of the pre-trained network, replace it with a new final layer, and train the new network on the data for the new task while leaving the pre-trained weights frozen. This process allows the use of very complicated models without having to waste computation time to train it from scratch. Optionally, we can then unfreeze the pre-trained weights and fine-tune the whole network further, in hopes of improving the accuracy of our model. [5].

4.3 Model selection

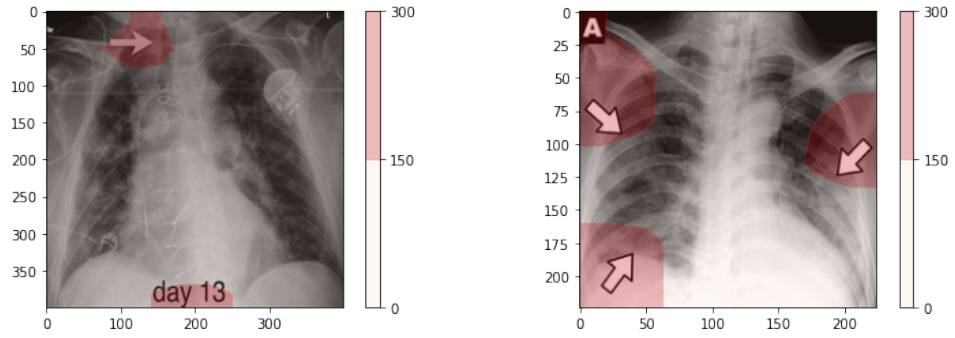
Many pre-trained CNN models exist for image classification, some of which have been trained on a large variety of data sets and their weights made open for public access. These models have been shown to perform very well with transfer learning methods. We selected our pre-trained model by

performing cross-validation with a few different models to determine which one performed best on the task, given a low number of training steps. Additionally, we used cross-validation to decide on the number of hidden nodes in the final layer. These models all have different architecture—the "shallowest" being of depth 23 while the deepest has a depth of over 500. The parameters of the networks also vary from 20.000.000 to 143.000.000. Training such deep networks with millions of parameters would take weeks. However, with transfer learning fine-tuning the network on our small data set only takes a few minutes.

When creating the convolutional neural network and utilizing transfer learning, we used the Tensorflow version of Keras. All the models that we tested with transfer learning are available from Keras [6]. We chose to use Adam optimizer in conjunction with our model. Adam is an adaptive learning rate optimization algorithm that uses stochastic gradient descent with momentum. This optimizer is commonly used to train Deep Neural Networks due to its computational efficiency, low memory requirements, and it being well suited for networks with a large number of parameters [7]. We used early stopping when training the networks. With the early stopping method, the model monitors the validation loss of the network, and once it has not decreased over a predefined number of epochs, the network stops training. When the validation loss starts increasing, the model is overfitting on the training data. Early stopping combats this and increases the generalization capability of the network [8].

4.4 Grad-CAM

As an empirical explainability measure, we employed the gradient-based localization technique Grad-CAM, which allows us to visualize in what areas of the image the network is basing its classification. The benefits of using Grad-CAM are that it both lends insight into failure modes of the model and helps identify data set bias. In general, we want the model to focus on lung tissue areas and not specifically on the artifacts, as can be seen in Figure 2 where the red area shows the innermost contour of the heatmap.



(a) (b)

Figure 2: Grad-CAM showing our network when it classifies data using artifacts on the image

The Grad-CAM technique uses gradient information propagated to the last convolutional layer of our network and assigns importance values to the individual neurons contributing to the network's decision. We compute the importance values α_k^c as a global-average-pool of the gradients flowing back from a convolutional layer with respect to its activation map A^k, thus given a class score y^c for a class c the importance value becomes

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}. \tag{1}$$

Finally, we compute in Equation 2 a heatmap H^c as a linear combination of the forward activation maps and their importance values. The heatmap can be scaled to the dimension of our original image and will contain the highest temperatures in the areas of interest, furthermore the $ReLU$ activation

function is applied as we only care about positive influences on the network's discriminative behaviour.

$$H^c = ReLU\left(\sum_k a_k^c A^k\right) \tag{2}$$

The method is general, i.e., we can use it to explain activations in any layer of our network. However, for the sake of the empirical explainability of the model's decisions, we are interested only in the output layer. Our implementation of the Grad-CAM algorithm is based on the source code given in the online tutorial *"Grad-CAM: Visualize class activation maps with Keras, TensorFlow, and Deep Learning"* with some adjustments to fit our network [9].

5 Experiments

In this section, we perform various experiments to achieve a high performing model and to make claims about our network's generalization capabilities. Our claims will be based solely on the network accuracy and empirical evidence obtained from the gradient-based localization. In order to avoid overfitting our models to artifacts in the data, such as the ones shown in Figure 2, we crop the images. As most of the artifacts are located in the corners or close to the edges, we cropped the images by 8% on each side. First, we scale the images to a 300×300 resolution and then cropped them down to 224×224 with a margin of 38 pixels on each side. All of our experiments run on the cropped data set.

5.1 Comparing model performance

The first task was to choose a model architecture to use for transfer learning. We selected a few candidate models from the pre-trained models made available by Keras, trained on the ImageNet data set. These models are trained on a large number of samples, as the ImageNet data set consists of over 14 million labeled images. In order to find the most suitable model, we performed the following comparison. Using 5 different pre-trained models (VGG16, VGG19, InceptionResNetV2, InceptionV3, Xception) and 2 different values for number of hidden nodes in the last layer (256, 512), 10 models were created.

We trained the models on both balanced and unbalanced data sets but chose to use the unbalanced data set since it, in general, did not affect the model accuracy by much and allowed for more regularization due to the amount of data in our training set. We trained each model using 1347 images, thereof 142 COVID-19 images. The validation and test sets contained the remainder of our COVID-19 images (52 images each), and 398 Non-COVID images. The results from our comparison using 4-fold cross-validation are shown in Table 2.

Table 2: Mean test accuracy for 4-fold cross-validation, with and without further fine-tuning (ft)

	Hidden nodes			
Model	256	512	256 (ft)	512 (ft)
VGG16	0.9734	0.9650	**0.9834**	0.9734
VGG19	0.9778	0.9800	0.9734	**0.9839**
InceptionV3	0.7550	0.7300	0.9244	**0.9278**
Inception ResNetV2	0.8822	0.8822	0.9678	**0.9722**
Xception	0.8689	0.6272	**0.9656**	0.9500

From our comparison, VGG19, with 512 hidden nodes in its final layer, had the best predictive powers, yielding a 98% accuracy on the test set with only three misclassifications of COVID images. However, the high accuracy, in this case, came with inadequate explainability measures concerning our localization techniques, i.e., the network did not predict COVID images based on areas on lung tissue but instead on the artifacts in the image. With further analysis using Grad-CAM we found that the best performing model is InceptionResNet2 with 512 hidden nodes in the last layer. The results from the Grad-CAM analysis are shown in section 5.2. The trade-off between model accuracy and explainability is about 0.8% in the test accuracy, i.e., the InceptionResNetV2 gave a 97.22% accuracy on the test set and misclassified 10 of the 53 COVID images as Non-COVID images.

5.2 Classification with data augmentation

The VGG19 and InceptionResNetV2 models with 512 hidden nodes were trained as binary classifiers for COVID and Non-COVID cases. We chose to use VGG19 since it showed the best accuracy in the model comparison and InceptionResNetV2 since it provided better explainability as compared to VGG19. Training with the additional augmented data resulted in higher test accuracy for the VGG16 model, but lower test accuracy for the InceptionResNetV2 model, seen in Table 3.

Table 3: COVID vs Non-COVID test accuracy,
with and without data augmentation and fine-tuning (ft)

Model	No augmentation	No Augmentation (ft)	Augmentation	Augmentation (ft)
VGG19	0.9734	0.9800	0.9777	**0.9867**
InceptionResNetV2	0.8822	**0.9755**	0.8822	0.9555

5.3 COVID vs. viral Pneumonia

Since COVID-19 causes a type of viral pneumonia, we decided to train a network to classify between COVID-19 and other viral pneumonia specifically. As before, we tried augmenting the data, doubling COVID cases in our test set, and also tried further fine-tuning of all layers. As can be seen in table 4, both VGG19 and InceptionResNetV2 showed a significant increase in accuracy, though this may be contributed to the test set being of smaller size. The InceptionResNetV2 model classified all COVID cases correctly, misclassifying 6 viral pneumonia as COVID, shown in Figure 3.

Table 4: COVID vs viral pneumonia test accuracy,
with and without data augmentation and fine-tuning (ft)

Model	No augmentation	No Augmentation (ft)	Augmentation	Augmentation (ft)
VGG19	0.9741	0.9799	0.9799	**1.0000**
InceptionResNetV2	0.8736	0.9799	0.8736	**0.9828**

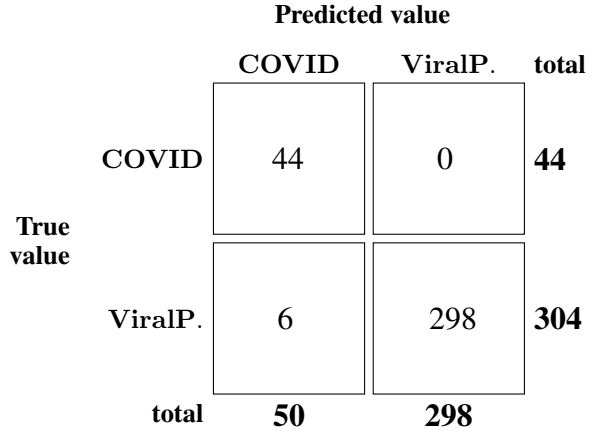


Figure 3: Confusion matrix for COVID vs viral pneumonia (ViralP.) test set predictions.
InceptionResNetV2 with data augmentation and fine-tuning

5.4 Analysing the results

Data augmentation increased the prediction accuracy in most instances, though there are some cases where it was not beneficial, as seen in Table 3. This decrease may be due to the model becoming biased to the augmented data, resulting in less performance on the unseen test data. Fine-tuning also proved very beneficial, often increasing prediction accuracies significantly. It is important to note that it was essential to use low learning rates for the fine-tuning, or it would risk severe overfitting[10].

In Figure 4, we show the results of the gradient-based localization after training on the InceptionResNetV2 with data augmentation. The results indicate that the network learns to identify more complex features within the X-ray images, rather than solely basing its decision on artifacts or other non-lung tissue attributes. Figure 5a shows the same image as in Figure 2a, where a network bases its decision only on the artifacts in the image, however, these artifacts are not a contributing factor in our network decision. Further, in Figure 5d, we can see that some of the artifacts point in the direction of the hotspots, suggesting that our network may be detecting similar indicators as the human expert that placed the artifacts on the image.

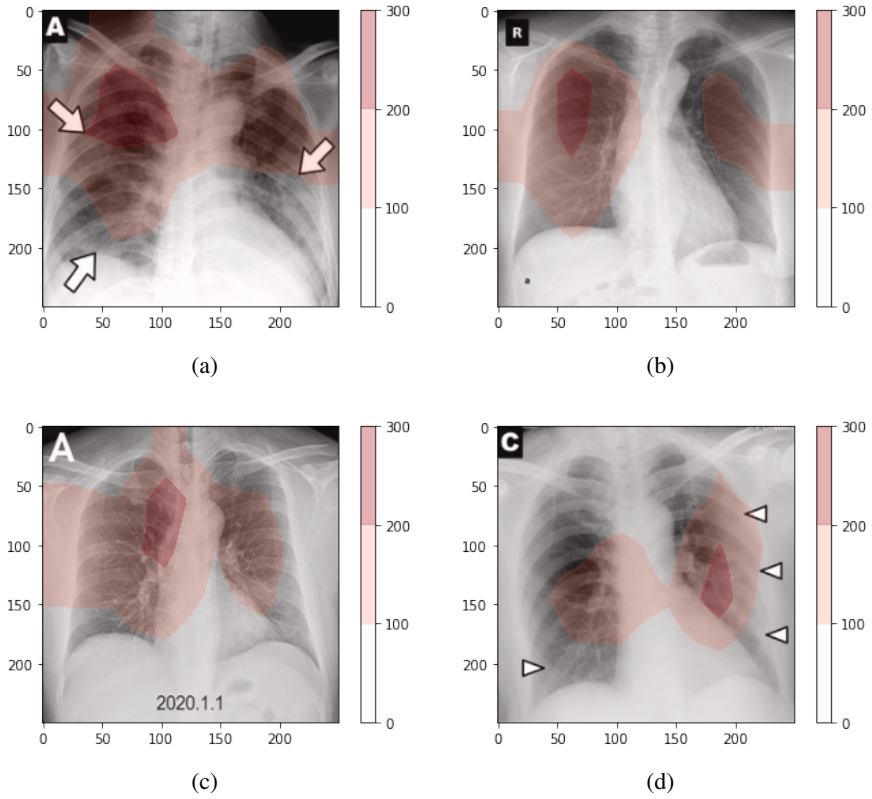


Figure 4: Resulting gradient-based localizations from InceptionResNet2

VGG19 performed best in terms of accuracy in all tests, though looking at the gradient-based localization, it seems it does not learn features relevant to lung tissue. Grad-CAM shows that in most cases, VGG19 determines classification according to non-lung areas, shown in figure 5. We speculate that the model may be overfitting to individual's anatomy, or other objects in the image, such as medical equipment. Since we only have a handful of COVID positive images, and many of these share the same individuals, the network may have learned features of the few individuals that are infected with COVID rather than features of the pneumonia.

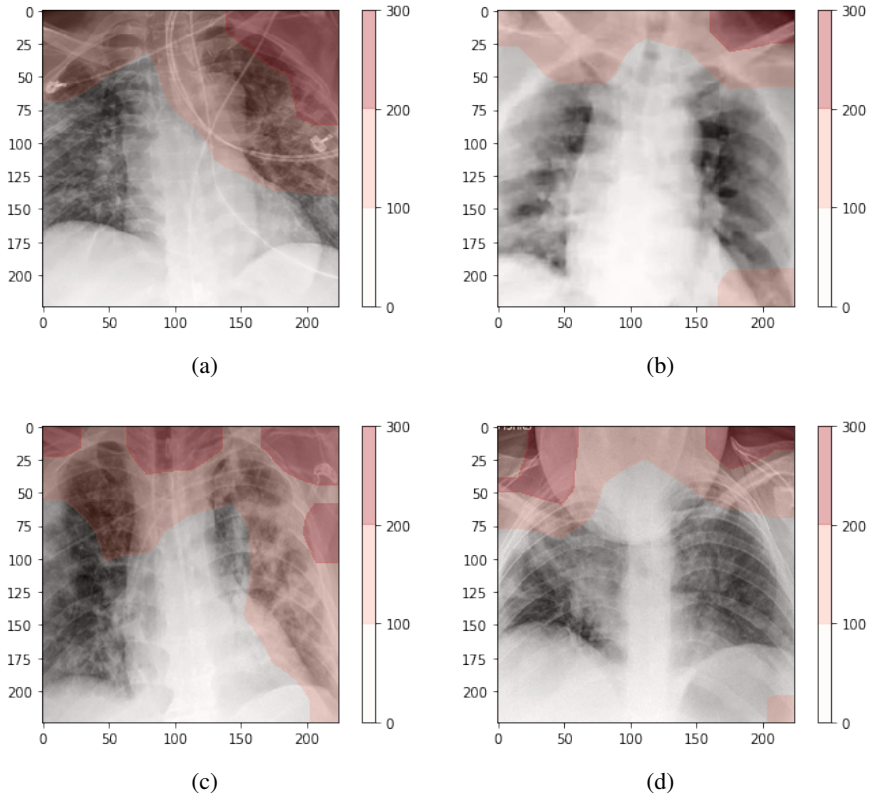


Figure 5: Resulting gradient-based localizations from VGG19

6 Conclusion

Our experiments showed just how important it is to have a good data set when training neural networks. A network will always find the easiest way to classify images, which can lead to strange results when we have artifacts or other shared features between data classes. Overall, the model we trained performed very well on the task at hand. However, given how little data was available to us and the flawed nature of it, no conclusions can be drawn whether our model can be accurately used to diagnose people with COVID-19. We can say that this paper demonstrates that there is some possibility of using Deep Learning as a diagnostics tool as long as sufficient resources are in place. The use of Grad-CAM gave us the much-needed insight into being able to analyze our results and allowed us to detect that our best model was not the model with the highest training accuracy, but instead, we were able to show the model's diagnostic ability through the Grad-CAM results.

References

[1] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization," *CoRR*, vol. abs/1610.02391, 2016. [Online]. Available: http://arxiv.org/abs/1610.02391

[2] N. Lanese, "Even if you test negative for COVID-19, assume you have it, experts say," 2020, Accessed May 15, 2020. [Online]. Available: https://www.livescience.com/covid19-coronavirus-tests-false-negatives.html

[3] W. J. Palmer, "Chest x-ray isn't reliable in detecting covid-19 lung infections," 2020, Accessed May 15, 2020. [Online]. Available: https://www.diagnosticimaging.com/covid-19/chest-x-ray-isnt-reliable-detecting-covid-19-lung-infections

[4] L. Wang and A. Wong, "Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images," 2020. [Online]. Available: https://arxiv.org/abs/2003.09871

[5] J. Brownlee, "A Gentle Introduction to Transfer Learning for Deep Learning," 2017, Accessed May 15, 2020. [Online]. Available: https://machinelearningmastery.com/transfer-learning-for-deep-learning

[6] Keras, "Keras applications," 2020, Accessed May 16, 2020. [Online]. Available: https://keras.io/api/applications/

[7] TensorFlow, "Adam," 2020, Accessed May 16, 2020. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/keras/optimizers/Adam

[8] ——, "Early stopping," 2020, Accessed May 16, 2020. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/keras/callbacks/EarlyStopping

[9] A. Rosebrock, "Grad-CAM: Visualize class activation maps with keras, tensorflow, and deep learning," 2020, Accessed May 15, 2020. [Online]. Available: https://www.pyimagesearch.com/2020/03/09/grad-cam-visualize-class-activation-maps-with-keras-tensorflow-and-deep-learning/

[10] Keras, "Transfer learning fine-tuning," 2020, Accessed May 17, 2020. [Online]. Available: https://keras.io/guides/transfer_learning/