COVID-19 X-Ray Image Classification Using Deep Learning

George Karcevski

Graduate Center, CUNY – Machine Learning CSC 74020

Project Title: COVID-19 X-Ray Image Classification Using Deep Learning

Project Members: George Karcevski

I found that one of the main areas of research in Deep Learning datasets that were readily accessible was COVID-19 X-ray image classification. It became clear that, in addition to frontline responders battling the epidemic, computer and data scientists were striving to effectively detect COVID-19 through X-rays and other forms of medical imaging.

During my research, I frequently came across two periodicals. First, there was Computers in Biology and Medicine, an open access journal that is a companion to Informatics in Medicine Unlocked. It describes itself as "an international gold open access journal covering a broad spectrum of topics within medical informatics, including (but not limited to) papers focusing on imaging, pathology, teledermatology, public health, ophthalmological, nursing and translational." [1] [2]. The publication of Healthcare Engineering, the second, is "a peer-reviewed, Open Access journal publishing fundamental and applied research on all aspects of engineering involved in healthcare delivery processes and systems" [3]. Together, they had written hundreds of studies addressing the question of whether COVID-19 could be predicted from various medical images (such as CT scans, X-rays, ultrasounds, etc.).

I intended to approach this issue in various ways. I first wanted to test if I could build a neural network without utilizing a pre-trained model because I had never worked with deep learning before. This involved learning the fundamentals of a CNN, comprehending what each layer accomplishes, and trying to develop a viable classifier. Second, I wanted to compare my

results to one of the pre-trained models that had been repeatedly referenced in the different articles.

My project's limits were fairly similar to those faced by many others in this field and on this subject: both a shortage of data and computing power. The first Kaggle dataset [4] was only 250 images total for training (111 COVID, 70 Normal, 70 Pneumonia images), 66 images for testing (26 COVID, 20 Normal, 20 Pneumonia images), and run-times for my code were exorbitant (ranging from 5 minutes all the way up to 1.5 hours on my Google Colab Notebook). The second Kaggle dataset [5] was 5216 images total for training (3875 Pneumonia, 1341 Normal), and 624 images for testing (390 Pneumonia, 234 Normal), with equivalent run times from 5 minutes to almost 2 hours.

I used a 3-step process to create my CNNs: first, I created a very simple CNN, then I created a somewhat more complicated CNN, and last, I used the pre-trained model to compare the three. After these models have been run, perform model adjustment in the form of data augmentation and contrast the outcomes. Please see attached Google Collab Notebooks for the code for these models.

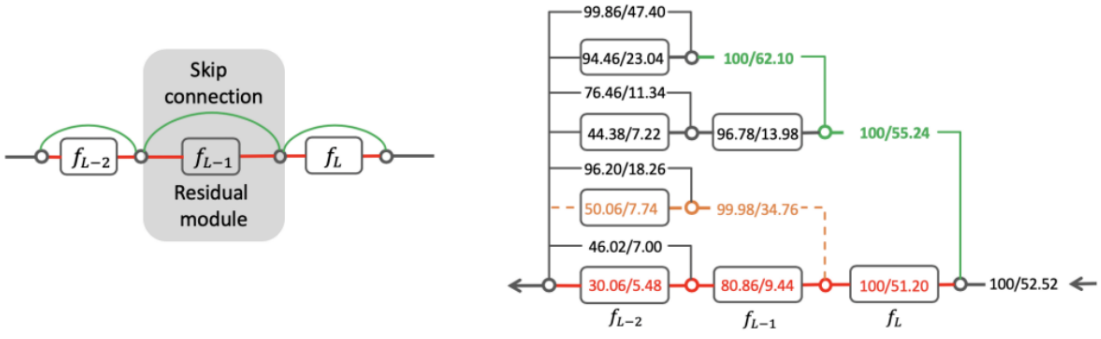
The Simple CNN (SCNN) was a 2-layer Sequential CNN model, with varying Conv2d and MaxPooling filter sizes, but all running the ReLU and Softmax activation function. In building this, the goal was to create something that would have a short run-time, small compute power, and give a basic understanding of CNN's.

The Complex CNN (CCNN) was a 4-layer Sequential CNN model, with varying Conv2d and MaxPooling filter sizes, but all running the ReLU and Softmax activation function. The idea behind constructing this was to build a CNN with exponentially more parameters than the

SCNN, more layers than the SCNN, more filters and kernels, different step and pool sizes, and more inputs than the SCNN.

The pretrained model used was the ResNet50 model from Keras. The ResNet50 model "is a variation of the ResNet model consisting of 50 layers (48 convolution layers, 1 maxpooling, and 1 average pooling layer). The ResNet50 model performs simple training and has many advantages due to its capacity for residual learning directly from images rather than image features." [6] The ResNet infrastructure has won multiple Image Classification competitions and is considered an incredibly strong competition infrastructure along the lines of XGBoost and LightGBM for tabular data. One of its most notable attributes is it's skip connection - "In deep neural networks (DNNs), a skip connection builds a short-cut from a shallow layer to a deep layer by connecting the input of a convolutional block (also known as the residual module) directly to its output. While different layers of a neural network learn different "levels" of features, skip connections can help preserve low-level features and avoid performance degradation when adding more layers. This has been shown to be crucial for building very deep and powerful DNNs such as ResNet, WideResNet, DenseNet, and ResNeXt." [7].

Essentially, it looks something like this:



"Three example backpropagation paths are highlighted in different colors, with the green path skipping over the last two residual modules having the best attack success rate while the red path through all 3 residual modules having the worst attack success rate" [7]. *By maximizing the path of the loss function, this skipping allows the model to pass over nodes that would result in inefficient losses and longer run-times.* The pre-trained ResNet50 model was created with the intention of determining whether an optimized model will yield superior outcomes to the preceding SCNN and CCNN models.

Last but not least, a frequently referenced strategy of model adjustment throughout ResNet and journal article documentation is data augmentation [8] [9]. Data augmentation is the act of using the same collection of images that you already have to train a model, but changing those images by rotating, moving, flipping, etc. and taking pictures of various pixels from various perspectives to help your model maintain a deeper learning.

In the Data Augmented Complex CNN (DACCNN), we used the same exact parameters as the CCNN, but used data augmentation on the images (see code for exact transformation) to see if this form of model tuning does actually produce a better result.

Throughout the Google Collab notebooks, we see that the results of the ResNet50 model sometimes outweigh the simple models, and other times it is simply too complex to train and due to the lack of training time, produces worse results than the other less complex models.

As mentioned in the presentation, the results of my work are simply not great, and the only remotely acceptable model would be the ResNet50 model for the non-paper specific dataset. While the paper absolutely dominated in terms of accuracy, there's a few ways to get better results on *my own* code, and the obvious ones are:

1. **Hyperparameter tuning** – Model hyperparameters like learning_rate, number of epochs, batch_size, pool_size, stride_size, etc…could all be optimized based off the images and data size. This falls out of the scope of this assignment as it would require larger compute power, more compute time, and more resources.

2. **Model Layer Tuning** - With more time and deeper understanding of the Conv2D, MaxPooling, and Dense layers, the various layers could be constructed in an optimized way rather than just using 2- or 4- filters for a CNN.

3. **More/Better Data** – One of the issues with this dataset was the images were labeled, but we knew no other information. In looking at the photos, you could see that there were images of men, women, and children (you could see the smaller frame of children versus the larger frames of males). Without insight into age, sex, previous health conditions, severity of the virus, etc…we cannot expect to make huge gains in deep learning using extremely basic information.

Bibliography

[1] Informatics in Medicine Unlocked, "About the Journal," [Online]. Available: https://www.sciencedirect.com/journal/informatics-in-medicine-unlocked.

[2] Computers in Biology and Medicine, "Aims and Scope," [Online]. Available: https://www.sciencedirect.com/journal/computers-in-biology-and-medicine/about/aimsand-scope.

[3] Journal of Healthcare Engineering, "About This Journal," [Online]. Available: https://www.hindawi.com/journals/jhe/about/#aims-and-scope.

[4] P. Raikote, "Kaggle," Kaggle, [Online]. Available: https://www.kaggle.com/datasets/pranavraikokte/covid19-image-dataset.

[5] P. Mooney, "Kaggle," Kaggle, [Online]. Available: https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia

[6] V. Kumar, A. Zarrad, R. Gupta and O. Cheikhrouhou, "COV-DLS: Prediction of COVID-19 from X-Rays Using Enhanced Deep Transfer Learning Techniques," Journal of Healthcare Engineering, vol. 2022, no. Article Id.

[7] D. Wu, Y. Wang, S.-T. Xia, J. Bailey and X. Ma, "Skip Connections Matter: On the Transferability of Adversarial Examples Generated with ResNets," ArxIv, vol. 1, no. 05990, p. 15, 2020.

[8] M. Rahimzadeh and A. Attar, "A modified deep convolutional neural network for detecting COVID-19 and pneumonia from chest X-ray images based on the concatenation of Xception and ResNet50V2," Informatics in Medicine Unlocked, vol. 19, no. 100360, p. 13, 2020.

[9] L. Perez and J. Wang, "The Effectiveness of Data Augmentation in Image Classification using Deep Learning," 2017. [Online]. Available: https://arxiv.org/abs/1712.04621.

[10] D. M. Ibrahim, N. M. Elshennawy and A. M. Sarhan, "Deep-chest: Multi-classification deep learning model for diagnosing COVID-19, pneumonia, and lung cancer chest diseases," Computers in Biology and Medicine, vol. 132, no. 104348, p. 13, 2021.

[11] M. Mahin, S. Tonmoy, R. Islam, T. Tazin, M. M. Khan and S. Bourouis, "Classification of COVID-19 and Pneumonia Using Deep Transfer Learning," Journal of Healthcare Engineering, vol. 2021, no. Article ID 3514821, p. 11, 2021