Data Mining – Professor Joe Burdis

Semester – Fall 2022, CSC 84040

Student: George Karcevski

Data Mining Final Project Documentation

When looking into datasets that were widely available, and were on the topic of Deep Learning, I found that COVID-19 X-Ray image classification was a major topic that was being worked on throughout the world. It seems that not only were doctors and healthcare workers on the frontlines fighting the pandemic, but computer and data scientists were tackling the issue of quickly and easily detecting COVID-19 through X-rays.

In my research online, I often times stumbled upon two journals. First, *Computers in Biology and Medicine*, an open access Journal that is companion to *Informatics in Medicine Unlocked*, “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 second, *Journal of Healthcare Engineering*, “a peer-reviewed, Open Access journal publishing fundamental and applied research on all aspects of engineering involved in healthcare delivery processes and systems” [3]. Combined, they had hundreds of research papers that focused on solving the issue: could you predict COVID-19 from various medical imagery (CT scans, X-rays, ultrasounds, etc…).

I wanted to tackle this problem from a few angles. First, as having never done any Deep Learning work, I wanted to see if I could create a neural network without using a pre-trained model. This meant understanding the basics of a CNN, understanding what each layer does, and attempting to create a classifier that was usable. Second, I wanted to use one of the pre-trained models consistently cited in the various journals, and see what sort of improvement that model gave me over my own.

My constraints on this project were very similar to the constraints many others in this field and on this topic face: the lack of data, and the lack of compute power. The Kaggle dataset [4] was only 250 images total for training (111 COVID, 70 Normal, 70 Pneumonia images), and 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).

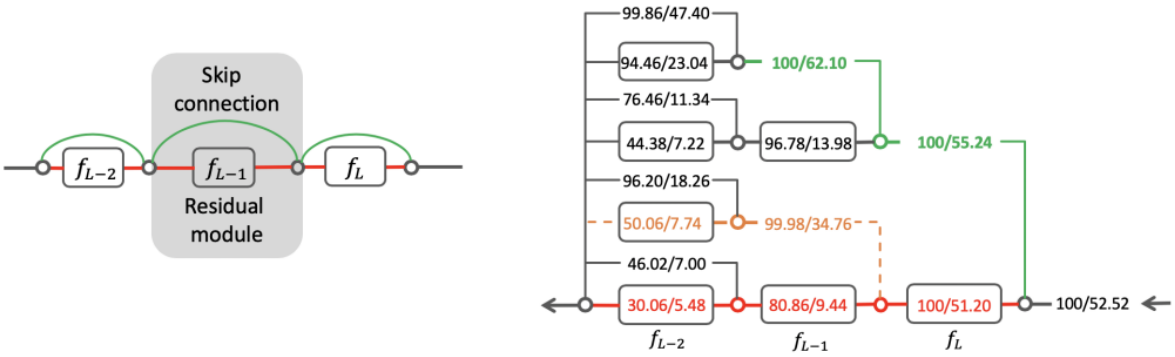
In building my CNN's, I took a 3-step approach: build a very simple CNN, build a slightly more complex CNN, and then use the pre-trained model and compare the three. After running these models, conduct model tuning in the form of data augmentation and compare results across all four. For these models, please see PowerPoint presentation for image and Google Collab Notebook for code.

The Simple CNN (SCNN) was a 2-layer Sequential CNN model. The Input Layer of size (224, 224), *Conv1* having 4-filters with a MaxPooling layer, *Conv2* having 8 filters with a MaxPooling layer, 2 *Dense* layers of 16 and 8 using "ReLU" activation, and the output layer of 3 having "SoftMax" activation. 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. The Input Layer of size (224, 224), *Conv1* having 32 filter with MaxPooling, *Conv2* having 64 filters with MaxPooling, *Conv3* having 128 filters and MaxPooling, *Conv4* having 256 filters and MaxPooling, 2 *Dense* layers of 128 and 64 using “ReLU” activation, and the output layer of 3 having “SoftMax” activation. In building this, the goal was to create a CNN that had more layers than the SCNN, more filters and kernels, varying step and pool sizes, and have exponentially more parameters than the SCNN, with the hope that more inputs/parameters = better results.

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.” [5] 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.” [6].

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" [6]. 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.

In building the pre-trained ResNet50 model, the goal was to see does an already optimized model provide better results than creating the SCNN and CCNN models from earlier.

Lastly, throughout ResNet and journal article documentation, it is often mentioned that one method of model tuning is data augmentation [7] [8]. Data augmentation is the process of training a model on the same set of images you have, but altering the images through rotating, shifting, flipping, etc…capturing different pixels at different angles such that your model retains 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 notebook, we see that the results of the ResNet50 model far outweigh the SCNN, CCNN, and the model tuned DACCNN.

Model	**Accuracy**	**F1**
Simple CNN	70%	50%
Complex CNN	65%	29%
ResNet50	90%	63%
Data Augmented CCNN	70%	39%

As mentioned in the presentation, the results here are simply not great, and the only remotely acceptable model would be the ResNet50 model. There's a few ways to get better results, and the 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/aims-and-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] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.