A.4 Fourth Research/Programming Assignment

Deep Learning on COVID CXR Image Dataset

By: Robin Singh

Abstract:

This experiment utilizes the COVID CXR Image Dataset (Research) dataset that I had found on Kaggle. This dataset seems quite relevant today because of the uprising of such a terrible virus. It is also a lot more difficult from the previous computer vision projects as these datasets are of the same body part compared to things that are extremely simple (like the MNIST dataset) or very different from each other (like the CIFAR dataset). All the images consist of human lungs and differentiate between whether the individual had COVID-19, no infection at all, or another infection all together (More specifically, if they had viral Pneumonia). This would probably require a more complex CNN to differentiate between the lungs and their features, and perhaps must go quite deep into learning individual features because of how similar some of the images may be. I will analyze what kinds of neural networks will or will not work on this dataset, and what required features are necessary for the model to learn the differences in the feature set, as well as how long this will take for the model to learn.

Introduction:

This research is conducted to analyze various parameters and features of a neural network to determine what is necessary to work with highly similar data and find extremely small or nuanced differences in the images to differentiate one thing from another. In this case, we will be differentiated between different kinds of lungs, COVID-19 infected, normal lungs, and lungs infected with viral pneumonia. We need to determine how to preprocess the image data from my local device, split the data into train and validation sets, and build the neural networks capable to accepting the data in the processed format and successfully classify them into one of 3 categories in the end. First, we will work with a normal neural network, consisting of various

hidden layers and numbers of nodes within to see how well those will perform. Afterwards we will be working up from extremely simple to more and more complex convolutional neural networks with more hidden layers. The accuracy will be judged by how well the neural network is predicting the set of vectorized images into objects after being trained by the training set and validated by the validation set. This would be used to determine the essential features of an AI system in the medical field to find different illnesses and differentiate between similar illnesses with accuracy. This way, in a real-world scenario, had this occurred, we would know what type of neural network to use and what it takes for that neural network to successfully analyze the precise details and features of the image entered by comparing it with similar images that it had been trained on, what sort of layers the neural network should have, how many hidden layers the neural network should have, how the results should be categorized, and what we need to be weary of when feeding the data in.

Literature Review:

I have seen quite a few examples of training machine learning models on chest x-ray data. One such example is presented in "Deep Learning–Driven Automated Detection of COVID-19 from Radiography Images: a Comparative Analysis" by Sejuti Rehman et al. The article talks about how "[e]xisting diagnostic technologies with high accuracy like RT-PCRs are expensive and sophisticated, requiring skilled individuals for specimen collection and screening, resulting in lower outreach" (Rehman et al). The authors of this article were able to use a DenseNet201 model with a with a Quadratic SVM classifier to achieve an accuracy of 98.16%, which is extraordinarily high for such a similar dataset.

Methods:

I will begin by loading in the data from my computer into the Google Colab environment in which I will be working with on this dataset. This data was downloaded from Kaggle (https://www.kaggle.com/sid321axn/covid-cxr-image-dataset-research) and stored on my local drive. However, due to the size of the folder containing all of the images, I was unable to directly upload it from my computer to the Colab environment and had to instead upload it to my Google Drive and then link my Google Drive to the Colab environment. I used the "os" library, as well as "cv2" to resize the raw image data to 200 by 200 (which I felt retained enough details of the images to conduct a proper analysis for differentiation without running into memory issues) and vectorize the data to feed into the various neural networks I will be working with. After vectorizing the image data, I made sure to shuffle the data so that the training could work effectively, and not just rapidly shift from learning one sort of category to the next. Finally, I was ready to run the data through neural networks.

I had begun working with basic neural networks, starting with one layer, and adding more layers on top of that to see if each individual layer influenced performance.

Next, I moved on to convolutional neural networks, and started with on Conv2D layer and a Max Pooling layer. I then added additional layers as it followed suit to see the improvement in performance.

Results:

Below is a table summarizing the results of the various neural networks. The CNNs have a default setup of a dense layer after the Conv2D layers (unless otherwise noted). They are also run with a batch size of 32.

Neural Network	**Training Set Accuracy**	**Validation Set Accuracy**

Basic neural network with one layer of 512 nodes	0.4129	0.4192
Basic neural network with one layer of 128 nodes	0.3663	0.3671
Basic neural network with four layers of 512, 512, 256, 128 nodes each	0.3663	0.3699
CNN with one Conv2D layer of 256 nodes	1.0000	0.9233
CNN with one Conv2D layer of 512 nodes	0.9856	0.9507
CNN with two Conv2D layer of 256 nodes each	0.9163	0.7973
CNN with two Conv2D layer of 384 and 128 nodes each	0.9060	0.8164
CNN with two Conv2D layer of 512 and 64 nodes each	0.9431	0.8219
CNN with one Conv2D layer of 512 nodes but a dense layer of 72 nodes (instead of default 64)	1.0000	0.9178

CNN with three Conv2D layer of 384, 128, and 64 nodes each	0.9973	0.9205

The best performing model in this case was a single layer CNN with 512 nodes in its hidden layer.

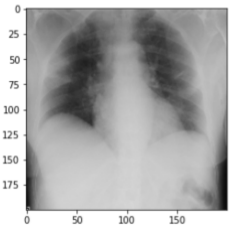
Conclusion:

Starting with the basic neural networks, it seems they are not well suited to learn a complex dataset as this. Even with the CIFAR dataset, the regular neural networks were able to retain some accuracy, but in this scenario that was just not possible. That may be because of the similar nature of all the data within this dataset. It consists of all black and white images of various lungs. Still an interesting factor to point out was that the highest performing model was the model with the greatest number of nodes in the first hidden layer, with this case having 512 nodes. This was also the case in the convolutional neural networks, which performed significantly better. The best CNN had only one layer with 512 nodes. Additional layers with more nodes only hurt the performance of the model, as seen with two- and three-layer CNNs. This could result from overfitting on the training set. Balancing out the nodes between the hidden layers to 512 was also not helpful in this case. The number of nodes in the initial layer is what made the biggest difference, giving us the highest accuracy of 0.9507 on the validation set. I also ran a CNN with a higher node value in the dense layer for classification, but even that had led to overfitting in this case. Because of such a similar dataset, the accuracy was very unstable in some of the models, as you can see in the appendix below, comparing the CNN

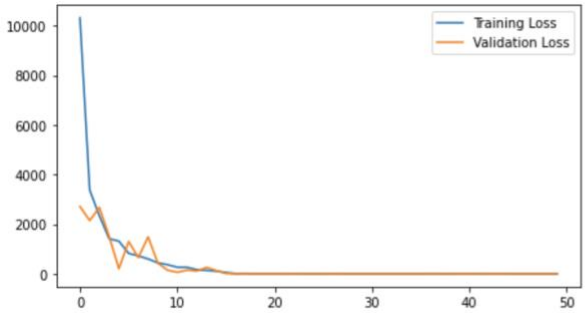
accuracy during the training and the basic neural network accuracy during the training. This was something very unique that I found when working with this dataset and did not occur in the previous experiments I had run.

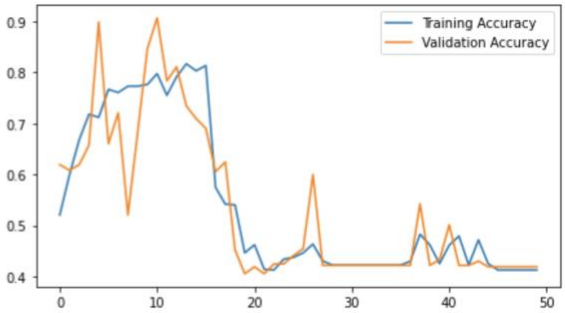
Appendix:

```
[ ]  img_size=200
     new_array=cv2.resize(img_array,(img_size,img_size))
     plt.imshow(new_array, cmap='gray')
     plt.show()
```



*Example of resized image data

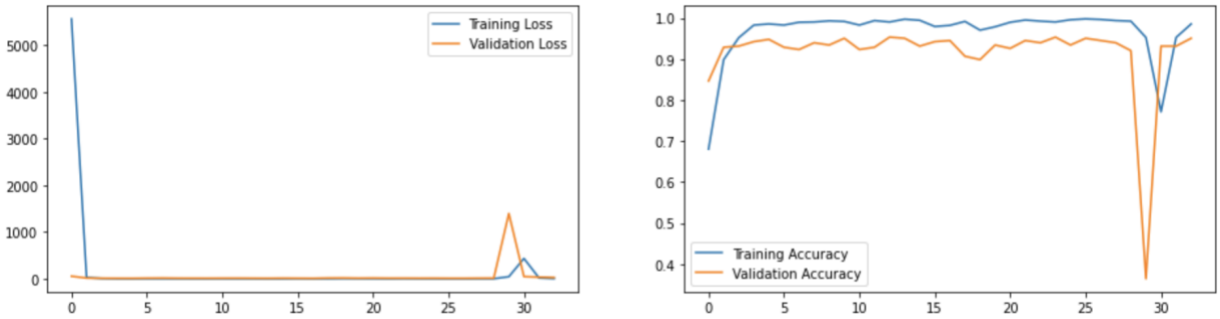




*Loss and accuracy charts of first basic neural network (summarizes the gist of the other basic neural network performance)

	0	1	2
0	25.01%	39.03%	35.97%
1	25.01%	39.03%	35.97%
2	25.01%	39.03%	35.97%
3	25.01%	39.03%	35.97%
4	25.01%	39.03%	35.97%
5	25.01%	39.03%	35.97%
6	25.01%	39.03%	35.97%
7	25.01%	39.03%	35.97%
8	25.01%	39.03%	35.97%
9	25.01%	39.03%	35.97%
10	25.01%	39.03%	35.97%
11	25.01%	39.03%	35.97%
12	25.01%	39.03%	35.97%
13	25.01%	39.03%	35.97%
14	25.01%	39.03%	35.97%
15	25.01%	39.03%	35.97%
16	25.01%	39.03%	35.97%
17	25.01%	39.03%	35.97%
18	25.01%	39.03%	35.97%
19	25.01%	39.03%	35.97%

*Confusion matrix of first basic neural network (summarizes the gist of the other basic neural network performance)



*Loss and accuracy charts of best performing convolutional neural network (had a little error that caused the dip, but it straightened itself out).

	0	1	2
0	100.00%	0.00%	0.00%
1	0.00%	0.00%	100.00%
2	0.00%	0.00%	100.00%
3	0.00%	100.00%	100.00%
4	0.00%	0.00%	100.00%
5	100.00%	0.00%	0.00%
6	0.00%	0.00%	100.00%
7	0.00%	0.00%	100.00%
8	0.00%	0.00%	100.00%
9	0.00%	100.00%	0.00%
10	0.00%	100.00%	0.00%
11	0.00%	0.00%	100.00%
12	0.00%	100.00%	100.00%
13	0.00%	0.00%	100.00%
14	0.00%	100.00%	0.00%
15	0.00%	0.00%	100.00%
16	0.00%	0.00%	100.00%
17	0.00%	100.00%	0.00%
18	100.00%	0.00%	0.00%
19	0.00%	100.00%	0.00%

*Confusion matrix of best performing CNN

Sources:

Rahman, Sejuti, Sujan Sarker, Md Abdullah Al Miraj, Ragib Amin Nihal, A K M Nadimul Haque, and Abdullah Al Noman. “Deep Learning-Driven Automated Detection of COVID-19 from RADIOGRAPHY Images: A Comparative Analysis.” Cognitive computation. Springer US, March 2, 2021. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7921610/.

Siddhartha, Manu. “COVID CXR Image Dataset (Research).” Kaggle, July 19, 2021. https://www.kaggle.com/sid321axn/covid-cxr-image-dataset-research.