Emilio Padrón Molina

Pneumonia Detection through CNNs with VGG Architecture

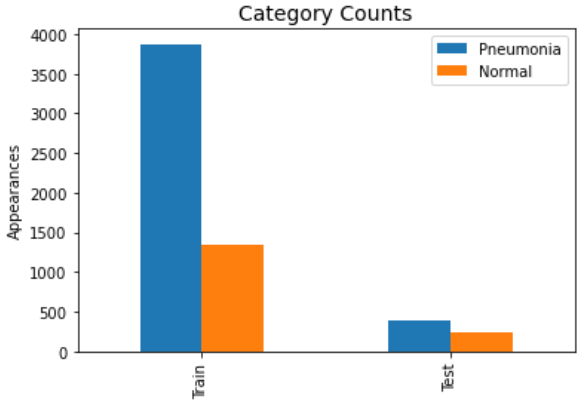
Objectives and Description of the Dataset

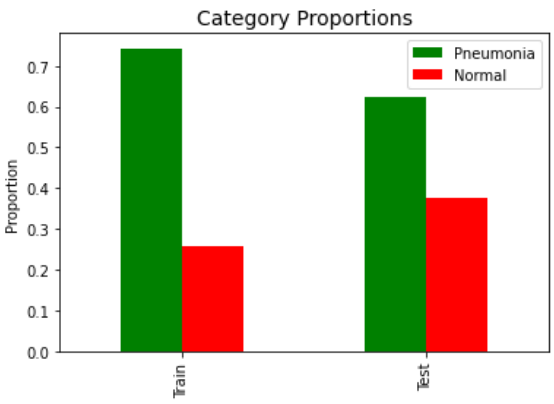
For this report, I decided to work with several chest x-ray images in which some show patients with pneumonia, and the others show healthy patients. With these images I am going to build Convolutional Neural Networks as an attempt to predict whether a person has pneumonia using their chest x-ray. The 5,840 images of this data set were obtained from https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia, and these were already divided into a training and testing splits. Given that these images have different sizes and some of them are composed of three-color channels while others only contain one, all images are passed into a function that standardizes them into a (224,224,1) form. Examples of these are:

Sample 3190: Normal

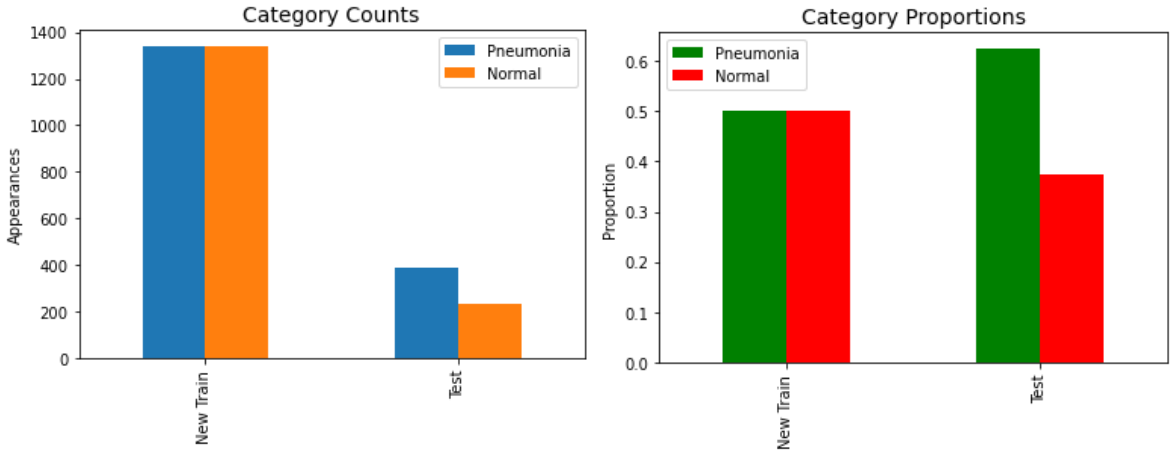
Sample 1409: Pneumonia

After the previous data engineering, the training set contains 5216 samples while the testing set 624. It is worth noting that representation of the classes in these splits are:





This shows that the set is clearly not balanced; therefore, it is necessary to undersample the training set so the model doesn't always predict the pneumonia class. This process gives the training set a new size of 2682 samples and therefore, the new class representation is as follows:



Models

The models created in this section are inspired in the VGG architecture in the sense that only convolutions of a kernel size of 3x3, with a stride of 1, and with padding, were applied to the images. Also, maxpooling layers were applied to the three models in intermediate convolutional steps. The first model had the following structure:

1. Two Convolutional layers of 16 3x3 filters with a stride of 1 and a ReLU activation.
2. A Maxpooling layer of 2x2 size.
3. Two Convolutional layers of 32 3x3 filters with a stride of 1 and a ReLU activation.
4. A Maxpooling layer of 4x4 size.
5. Two Convolutional layers of 64 3x3 filters with a stride of 1 and a ReLU activation.
6. A Maxpooling layer of 4x4 size.
7. A flattening layer.
8. A fully connected layer of 50 units with ReLU activation.
9. A fully connected layer with Softmax activation that results in two outputs.

This model was trained using a batch size of 32, a RMSprop optimizer with a learning rate of 0.0001, a binary-cross-entropy loss, and for 4 epochs.

The second and third models are very similar to the first one. Nonetheless, the main difference between the first and second model is that the latter contains dropout layers, with a 0.25 probability after every maxpooling layer, and with a 0.5 probability in between the two fully-connected layers. Moreover, the second and third model

differ in the type of optimizer given that the third is trained using Adam instead of RMSprop, and the earlier was trained for 10 epochs while the latter for 14. Aside from the already mentioned changes in between models, the rest of the parameters remain the same.

Errors

After running the three described models, the following error results are obtained:

	Accuracy	Precision	Recall	F1
Model 1 (RMSprop)	0.620192	0.623586	0.989744	0.765114
Model 2 (RMSprop with Dropout)	0.618590	0.626667	0.964103	0.759596
Model 3 (Adam with Dropout)	0.618590	0.622581	0.989744	0.764356

Insights and Analysis

Looking at the error results of the three models, one can see that all have similar accuracy. Nevertheless, because of the nature of the problem, one must also consider the recall score given that a good score means that the model doesn't classify many people who have pneumonia as if they don't. That being mentioned, the best model appears to be the first one given that it has the best accuracy and the highest recall.

Nonetheless, it is worth noting that the accuracies of these models could be furthered increased. This could be done by obtaining more data of the normal class to train the model or by changing the images' resolution (either to a higher or a lower one). Additionally, one could change the layers or the hyperparameters to get better results using the same dataset; however, given the numerous possibilities that this action could imply, this could also worsen the model.