Dataset Description:

The dataset was obtained from Kaggle, The dataset consisted of 2 classes, ‘covid’ and normal in which batches were made dynamically. The ‘covid’ class had 69 images feed into it all of chest xrays of various sizes. The normal class had around 50 images, 15 images provided from the dataset and rest were taken from a different dataset consisting of non-covid patients. Apart from this, for testing two images from personal contacts were also used. The links to the datasets can be found in the references column.

Model description for X-Ray Covid-19 detection:

All the images were feed into the model for training with the shape of (200,200,3) , This was to find a uniformity between all the images and also to make sure the images were not too small to acquire any kind of information.

The number of layers used in the model were made with the compatibility of DPU acceleration kept in mind. That might have reduced the range of layers, but we used various other techniques to compensate for that. The layers and the model breakdown are somewhat like this:

The model consists of Conv2d layers to perform CNN. This layer creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs. In image processing kernel is a convolution matrix or masks which can be used for blurring, sharpening, embossing, edge detection and more by doing a convolution between a kernel and an image. The pooling is being done with the use of max\_pooling2d layers. Max pooling operation for 2D spatial data. Dropout layer was used to reduce the extra nodes to avoid overfitting. This allowed us to increase the number of conv2d, pooling pair so we can avoid underfitting as well. The output of the dropout was then flattened by adding an extra dimension using the flatten layer. The output of the flatten layer is feed into the dense layer pair in the end of the model which then makes the logic to distinguish the input images into two classes of COVID-19 and non-covid.

The original plan for the model was to utilize the batch normalization layers but they had some sort of compatibility issues forcing me to use the layer’s impact on the CPU. Due to lack of time we shifted to using maxpooling instead. Batch normalization was preferred in this scenario due to the vast difference in the number of images in both classes in the original dataset. This was overcome by adding additional data and finding a balance between classes.