HeartLab AI Technical Test

SETUP

The task presented was to classify images of chest X-rays to determine if the image showed signs bacterial pneumonia, Viral pneumonia or if it was healthy.

The task involved classification of images. Although I had familiarity with classification tasks in the past, they were for speech signal data. I employed many things that I learnt during that project to this one, while also applying some new methods that learnt during this task. I chose to use Keras as I was previously familiar with its implementation.

Additionally, implementation and testing were done on googles cloud based Jupyter notebook service Google Colab and virtual GPU. However, they were then later brought over to PyCharm and made to work with the given format. I have also given the. ipynb file if you wish to run it on Google Colab.

STRUCTURING

The first issue that was encountered was the structure and format of the data. The data was structured

chest_xray /

> ***NORMAL /***
>
> ***PNEUMONIA /***

Within the Pneumonia folder, each file name indicated which variation of Pneumonia it was.

Initially, I took the contents of both folders, created a Dataframe using pandas, applied labels based on its type, as well as added a column for its absolute path, combined and then split into training testing and validation splits. However, as the Dataframe only contained a reference to the files absolute path and not the files themselves, accessing each proved more complicated down the road. Therefore, I decided to modify the file structure to make it easier to classify. In the notebook file, I make a new folder called BACTERIAL, loop through all the files in the PNEUMONIA folder and mover any with the word 'bacteria' to the BACTERIAL folder. Then rename the PNEUMONIA folder to VIRAL. The updated structure looks like:

chest_xray /

> ***NORMAL /***
>
> ***VIRAL /***
>
> ***BACTERIAL /***

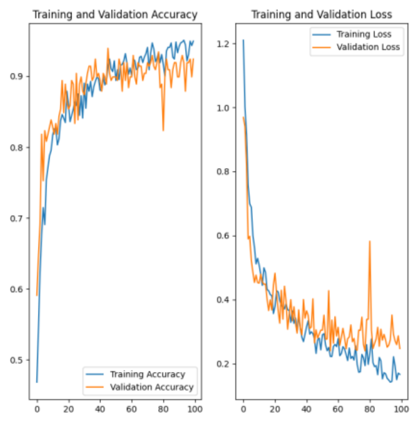
PRE-PROCESSING

From here a batch size was selected and an image size was specified. The batch size chosen for this task was 32. Although this dataset is relatively small, and testing was done with a size of 64 and 16, 32 gave the best results when the training time was increased to 100 epochs. The image size was set to 180 by 180 pixels. 180 was chosen as it provided enough detail with the input images. However, any input smaller than 180x180 will cause issues. It was assumed that the input images would not be smaller than 180x180. The data was then split into a 80/20 training-testing split and pre-processed by Keras's "image_dataset_from_directory" function. The data was then normalised.

MODEL

The model used was a Keras sequential CNN with 4 layers. 3 convolution layers and 1 dense layer. The problem is a multiclass classification problem, and the problem involved the classification of images. The Convolutional layers were used to address image classification. Although I experimented with changing the output layer's activation function to 'SoftMax', it tended to cause large fluctuations on the validation accuracy and loss.

The optimizer was also experimented with, but optimizers such as SGD and RMSprop resulted in extremely early saturation where the local minimum was achieved at around 50%. Thus Adam was used. Below are the accuracy and loss curves for one instance of training.



RESULTS

Although the above graph does not show explicit signs of over or underfitting, its possible as the amount of epoch are increased, the model will become too familiar with the input set and result in overfitting. At 100 epochs, the accuracy reaches to the 90%'s. Such a high accuracy could be a sign that the model has adapted too well to the input data. To combat this, either more data needs to be provided or more augmentation methods can be used.

I also included a few unit tests to ensure that various parts of the execution occur correctly. This includes the directory restructuring, file types, Directory names, number of identified classes and correct labels.

Finally, there is a section in the code to test various inputs of chest Xray's.

I have had various results from testing this model. In some iterations it can correctly identify 4/6 inputs, having the most difficulty with viral infections. However, the images I used for input are from google and I could not confirm that each of the input images are what I thought they were. Additionally, I noticed that the model's classification of each image varied between different times that the model was created, likely due to the random training-testing split taken.