ML Assignment – Image Classification

- Jack Allen s3832293 & Vijay Iyer s3797863
-

Assignment Task:

The task was to develop two machine learning models that take an image of a cell as input. One model is required to accurately predict if a cell is cancerous, and the other model can predict the type of cell. Initially both models were approached in a similar way. We began by developing a base model using a traditional neural network. From this we expanded on each model with VGG deep neural networks.

Task 1

In creating ML models that can predict if a cell is cancerous, we decided to use the data labels main data csv file as well as the data labels extra data. We took this approach because the more data we have the better the model can be trained. When conducting the EDA, we began by looking at how the data is distributed. We did this because if we have a lot of data in one area but not others our model may be better at predicting certain things and not others. This was also done to check if the data is distributed well between the train and test data sets. Appendix 1 shows that there are many more epithelial cell types than others in the data and slightly more non-cancerous cells than cancerous. Though the test set has a very similar distribution of data on each cell and whether the cell is cancerous as the train set. We also decided to look at the images of non-cancerous and cancerous cells to see if there is any clear indicator of a cancerous cell that can be picked up by the human eye, these images can be seen in appendix 2(0 are non-cancerous cells and 1 are cancerous). As can be seen in appendix 2, there is no obvious way of telling which cells are cancerous and which are not based on the images.

Now it was time to start the process of creating and testing models. The traditional neural network was used to begin with to establish a base level of accuracy that the other models can be compared to. In creating the base model, we also took the opportunity to test how different numbers of neurons in the hidden layer affect the accuracy. We tested with 256, 512 and 724 neurons. The respective learning curves for these models can be found in appendix 3 in order. All 3 models performed very similarly, thus we concluded that the number of neurons in the hidden layer had little affect on the accuracy. In further testing 724 neurons seemed to perform the best so we used that for further models. The aim was then to build on this with a more complex model and to ultimately achieve a greater accuracy. VGG was used because it is one of the most used image-recognition architectures. The VGG model we created used 3 blocks and 50 epochs. This model outperformed the base model but was overfitting as can be seen in appendix 4. To prevent overfitting, we created a total of 9 models testing different regularization and dropout values and learning rates. After this testing was completed, we ended up with a final VGG model that can be seen in appendix 5. This model had the highest accuracy (0.9035) out of those tested and did not overfit or underfit.

Task 2:

For this task, to create a deep learning network model to predict the cell type of an image, we decided to use the main data csv file for training the model and then using the model, extra data csv file was used for prediction. Then the predicted extra data was used to boost the trained model performance. When conducting the EDA, we began by looking at the different cell types and different representations of the same cell type. As seen in Appendix, inflammatory and other cell type look pretty similar to naked eye but the model might differentiate better. Appendix 7 indicates that there are lot of images for epithelial and images for others is less in comparison to others. This is kind of imbalance, but it is not treated while developing the models as deep learning models can cope up with minor imbalance.

Two approaches were used to implement models to predict cell type for an image. First approach is using simple MLP (Multi-Layer Perceptron) and the second approach is using VGG architecture. In the first approach, the traditional MLP is developed with an input layer of shape 27 * 27 * 3 (length * width * RGB channel of an image) and output layer of shape 4 (as there are 4 target feature). For the hidden layer, hyperparameter tuning was used where the number of neurons was in [256,512,724]. For each number of neurons, model was built and then these models were compared based on the Accuracy, F1-score and Confusion Matrix. Based on the metrics, models with 512 and 724 neurons in the hidden layer outperformed the model with 256 neurons and both had similar accuracy. However, looking at the confusion matrix, model with 512 neurons performed better with an accuracy of 0.6909 and f1-score of 0.558 as shown in appendix 8. Keeping this performance as base performance, VGG architecture was developed to boost the performance of the model. Since our image size is 27 by 27, only 3 blocks of VGG architecture were considered. Using the base VGG architecture with 3 blocks, model's accuracy was boosted to 0.74 and f1-sccore also increased to 0.676. This shows that VGG architecture is already performing better with base architecture, however there was overfitting in the model looking at the train and validation accuracy. To remove overfitting, several techniques were implemented like changing regularization parameters and learning rate. Total of 4 models with different learning rate (0.001 and 0.01) and l2 regularization parameter (0.001 and 0.01) were developed. After this testing, we ended up with a final VGG model with learning rate - 0.001 and regularization parameter – 0.001 that can be seen in Appendix 9. This model gives an accuracy of 0.728 and f1 score of 0. 639. Also, the confusion matrix shown below is also promising for a deep learning model.

Now this best model is used to predict the cell type for extra data. Once predicted, records of cell type classified as "others" is used to merge with the existing train data and then the model is retrained on the new train data. The reason to do this is because, deep learning learns more with more data. In the initial data, number of images for "others" was pretty less. So, adding more images for "others" category would boost the performance of the model in learning "other" cell type. This method is called over sampling. This method ensures that more images are appended for the minority class and model can learn from it. After doing, the same architecture is giving an accuracy of 0.736 and f1 score of 0.679, which is better than the VGG model.

Model Evaluation –

According to the previous research [1], the results obtained and the methodology followed is pretty much the same with major difference in the architecture and availability of data. In the previous research, the dataset available had 20,000 images, which was not the same in our assignment. This is a major factor, as any deep learning model thrives on data, more it gets, the better it performs.

The first and major difference is in the approach. The research is solely focused on the detection and classification of nuclei cell type. For doing the same, research used combination of 2 different model, one for detection and one for classification. This is not the case in the approach we followed for this assignment. For this assignment, we used one model to identify whether the given cell image is classified as isCancerous or not and the second model to predict the type of the nuclei cell.

The second difference is the architecture. Architecture used in the research was SC-CNN (Spatially Constrained CNN) – Input -> Convolution Layer -> Max Pooling -> Convolution Layer -> Max Pooling -> Fully Dense -> Fully Dense -> Parameter Estimation -> Spatially Constrained Layer for the cell type detection and for the isCancerous – Input -> Convolution Layer -> Max Pooling -> Convolution Layer -> Max Pooling -> Fully Dense -> Fully Dense -> Fully Dense. This architecture is different to the one selected for both the task. For both isCancerous and cell type detection, our architecture is – Input -> Convolution -> Convolution -> Max Pooling -> Convolution -> Convolution -> Max Pooling -> Convolution -> Convolution -> Max Pooling -> Fully Dense -> Fully Dense.

Results:

(Available only for cell type detection in the research paper):

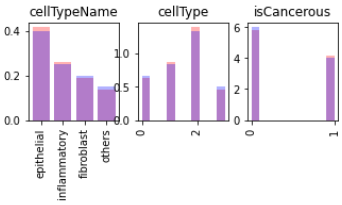
	Our VGG Architecture	Research SC-CNN Architecture
Precision	0.678	0.781
Recall	0.678	0.823
F1-Score	0.676	0.802

References:

1. Korsuk Sirinukunwattana, Shan E Ahmed Raza, Yee-Wah Tsang, David R. J. Snead, Ian A. Cree & Nasir M. Rajpoot 2016, “Locality Sensitive Deep Learning for Detection and Classification of Nuclei in Routine Colon Cancer Histology Images, *IEE Transactions on Medical Imaging,* vol. 35, Issue 5.

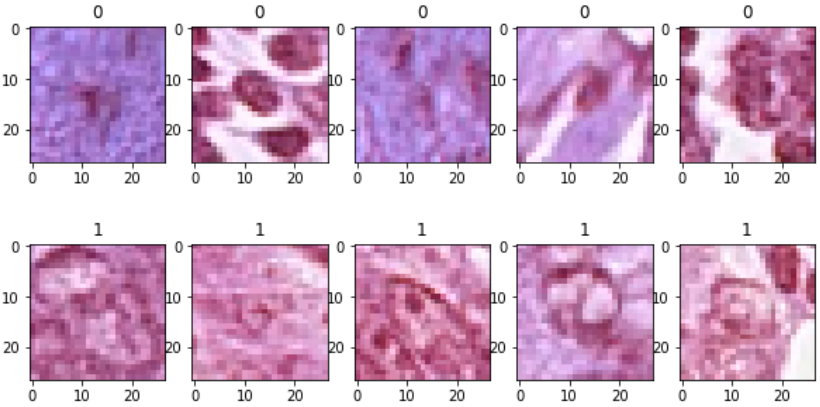
Appendices

1

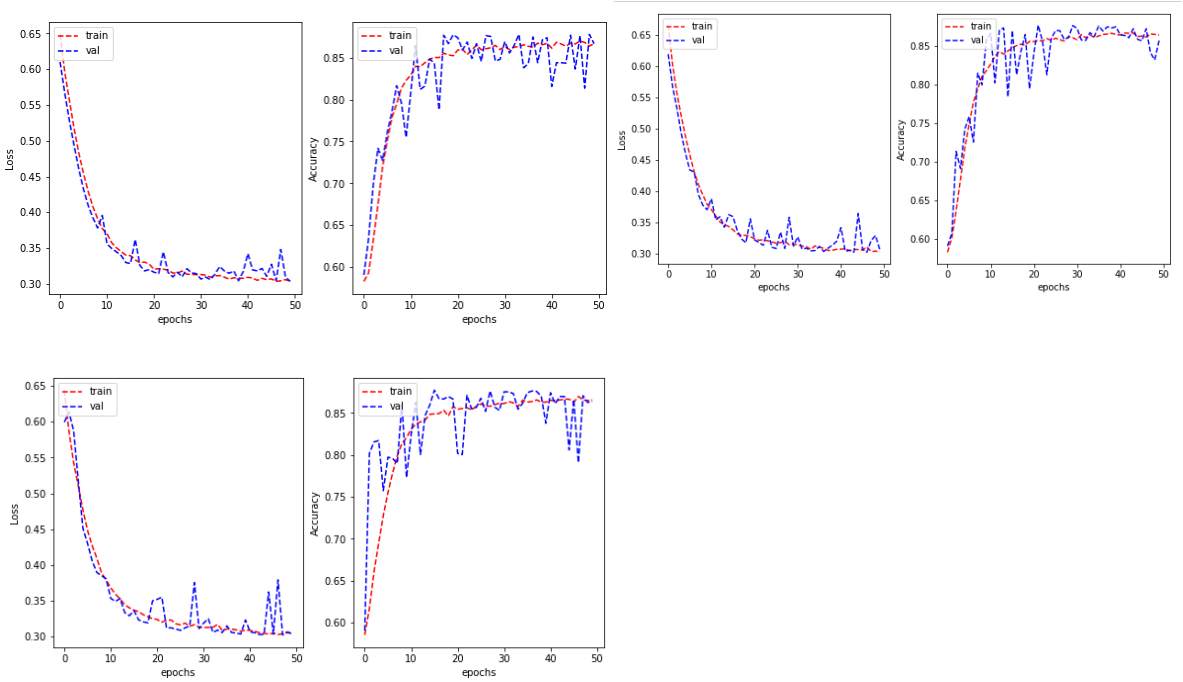


Task1

2

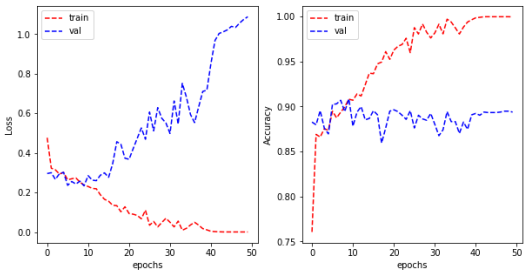


3

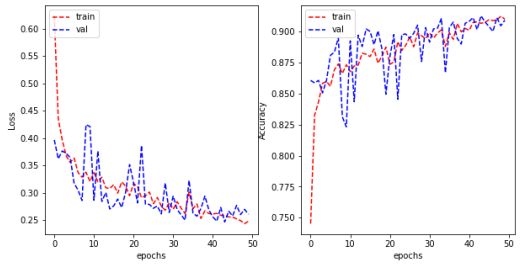


[1.0596505403518677, 0.8929293155670166]

4



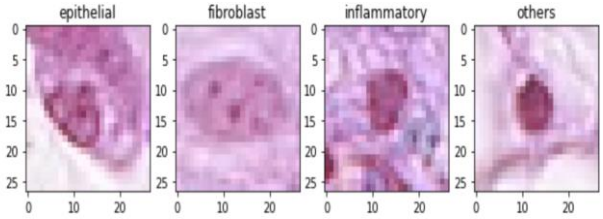
5




```
1980/1980 [==============================] - 2s 1ms/step - loss: 0.2636 - categorical_accuracy: 0.9035
Performance on test data:  [0.2636178135871887, 0.9035353660583496]
```

Task 2:

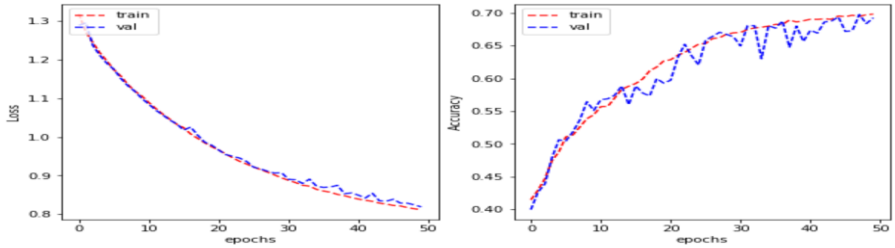
6.



7.

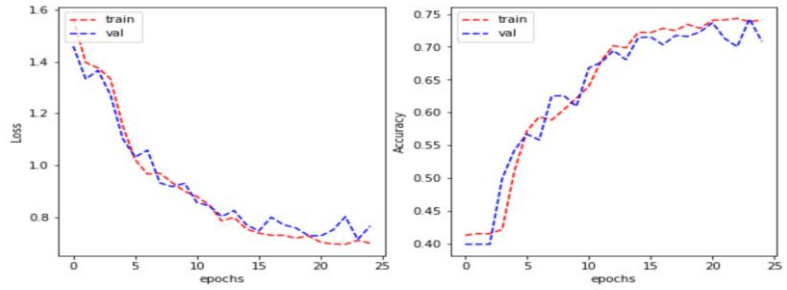
```
: train_data.cellType.value_counts()

: 2    2629
  1    1588
  0    1230
  3     885
  Name: cellType, dtype: int64
```

8.




```
1980/1980 [==============================] - 2s 769us/step - loss: 0.8291 - categorical_accurac
Model Evaluation:  [0.8291257619857788, 0.6909090876579285]
Accuracy score:  0.6909090909090909
F1 score:  0.5580080130357O7
Precision Score:  0.6132678863436061
Recall Score:  0.5748018237074813
[[199  77  70  18]
 [ 22 399  94   8]
 [  8  59 750   1]
 [ 79 128  48  20]]
```

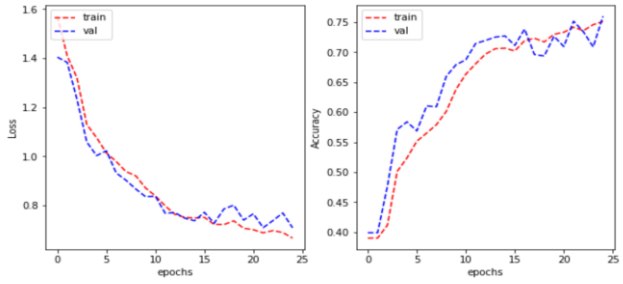
9




```
1980/1980 [==============================] - 5s 3ms/step - loss: 0.7331 - categ
Performance on test data:  [0.733139157295227, 0.728787899017334]
```

```
Accuracy score:  0.7287878787878788
F1 score:  0.6039431725242072
Precision Score:  0.7205990262763426
Recall Score:  0.6119299642234376
[[208  63  87   6]
 [ 22 416  81   4]
 [  5  26 786   1]
 [ 66 129  47  33]]
```

10.




```
Accuracy score:  0.73989898989899
F1 score:  0.6765575840761694
Precision Score:  0.6781478014281556
Recall Score:  0.6783337486330103
[[240  40  28  56]
 [ 23 419  25  56]
 [ 62  42 695  18]
 [ 43 103  19 111]]
```