Peltarion Assignment Report

Parastu Rahgozar

September 30, 2019

To Peltarion's Machine Learning team,

In addition to the notebook including my code, please find explanations about it below.

1 Description and Workflow

To investigate the problem we need to first look into the data. We have about 10015 images of skin lesions with fixed size of $450 \times 600 \times 3$, where 3 describes the RGB channels. The meta data involves information about gender of the patient, age, localization of the lesion, type of diagnosis, etc. There are 7 different types of lesions. These 7 types will be the labels in this problem.

Looking deeper the dataset is highly imbalanced. Meaning that a big number of images belong to one certain category while other groups do not follow the same distribution. This matter will cause a noticeable bias of our model towards the majority class. There are several approaches that can be applied to overcome this issue, such as data augmentation (resizing, rotating, etc), oversampling the minority class or simply looking for a suitable performance metric which does not focus on accuracy alone (i.e AUC or Precision/Recall curve).

Moreover, in the data source we can see that same patients may have several lesions. This can affect the power of judgment in our model since if each of these lesions end up one in the training set and one in the test set, our model can make a false correct prediction simply because it is more probable that the network will memorize and give an accurate answer with no relation to the actual performance of the model. Therefore doing a subject-wise K-fold cross validation would help solving this issue, while simultaneously also solving the issue of overfitting.

From our point of view the problem can be solved in two approaches. Either segmenting the lesion from the original data or resizing and fitting the whole image with original RGB channels. We chose to go with resizing only, since the time and resources were limited in this assignment, but it would definitely be challenging, yet interesting to investigate the other path.

In this approach, we implemented a simple version of a CNN which I will explain further. I will also reflect on my own work since of course there are always issues regarding coding and solving the obstacles. The code is written in Jupyter Notebook (Python 3) and was run on Google Colab since I had some limitations with memory on my computer. The time limitation of 72 hours also was not enough for the laptop for this challenge.

1. To begin with we first initialize the path for the data and load all needed images and metadata from mentioned paths. There needs to be a dictionary to map names of the lesion categories to abbreviations, which later we will convert to numerical categories as labels.

2. Since this is a large dataset, it is important to check for missing values before proceeding to next steps. In our case, there are only 57 values missing in ”age” column which I replace them with mean value. Although, since we only train our model based on images this step can be skipped.

3. In this step the images are being resized. First we tried resizing to 150×200 although this approach was still heavy for the memory so we tried 75 × 100. The arrays of images are saved as a separate column of a dataframe,

4. As of splitting the data for training and testing, we decided to do a 80%-20% for train and test set respectively, while 10% of training set will be used as a validation set. Due to time limitation unfortunately we did not get to do a K-fold cross validation although that would be a suitable approach.

5. Our implemented CNN first includes a 2D convolutional layer and then a Maxpool layer to reduce the dimension of input volume for the next layers. This is then followed by a dropout to help with overfitting and to remove random nodes. The model repeats the same architecture with a flattened layer to convert to a 1D vector. The last two fully-connected layers are working as classifiers and the ”softmax” activation gives the distribution of probability of each class. The final number of parameters of this model was about 222000.

6. As an optimizer, we decided to apply ”Adam” with learning rate of 0.001 since it is popular for a quick and fast outcome.

7. In our model the training part of the model has not been completely run, due to the limitations mentioned before. Therefore we decided to leave it as it is. Of course, there can be assumptions about the model's performance, while there are some critical points to take into account. Although we tried to cover most of the possible approaches in this report.

2 Discussion

In this section we will discuss the obstacles of this project and ways to overcome them in coming practices.

- Time Limit According to the current setup on GPU from Google Colab, each epoch takes about 2:30 hours which means the training and validation phase would take more than 100 hours to complete if we only have 50 epochs.
- Memory Limit In addition to the above issue, the matter of memory can absolutely affect the model setup such as the filter size, the image size, etc. Therefore it is rather important to take into account such boundaries while preparing the structure.
- Limited Data As it was also mentioned in the assignment, there is a limited number of data for the model to learn from. This matter can be overcome by either data augmentation, or using a pre-trained CNN (For instance trained with ImageNet sources.
- Variation of Networks available Last but not least, there are new neural networks introduced every day in the world of deep learning. It would be an interesting approach to implement AlexNet, VGG or ResNet. Finally, by applying ensemble method and combining some of these methods, better results may be achieved.