107062381 楊孝偉

Assignment 3 Report

In this assignment we are going to build a CNN binary image classifier to predict the 30-day mortality of patients.

First I separated the CXR images into two sub folders : 0 (Alive) and 1 (Dead). This separation is for the ease of model training.

Keras ImageDataGenerator makes it easy to label and load the large image datasets. Here I created one object for ImageDataGenerator and rescale the image such that their pixel values are normalized between 0 and 1 without affecting the image quality.

I used data augmentation to increase the amount of data to prevent the model from overfitting. It is neither practical nor efficient to store the augmented data in memory, and that is where the ImageDataGenerator class from Keras comes into play. ImageDataGenerator generates batches of tensor image data with real time data augmentation. The first is setting the shear intensity to 0.2. Shear transformation slants the shape of the image by 0.2 degrees. The second is zoom range to 0.2. A random zoom is obtained and 0.2 is less than 1.0, so it magnifies the image. The third is setting the width shift range to 0.3 which specifies the upper bound of the fraction of the total width by which the image is to be randomly shifted, either towards the left or right. The closest pixel value is chosen and repeated for all the empty values to fill the points where we don't have any value. Horizontal flip is set to true to generate images, which on a random basis, will be horizontally flipped.

Then we use this object to call flow from directory method by specifying our path to Train directory, target size which is important because real-world images can be in different size so what ever the input image size be it will be resized to 150 by 150 image. Then we specify batch size which simply means the number of samples that will be propagated through the network in a given time 32 is the default value for that function. Then here our classification result fall in one of the two classes : Dead or Alive, hence we specify class mode as binary.

Keras has a useful API which makes us easier to define the layers of our neural network. Here the input shape is 150, 150 which is our image size and 3 represents color channel RGB. Conv2D: Neural networks apply a filter to an input image to create a feature map that summarizes the presence of detected features in the input. In our case there are 32, 64 and 128 filters or kernels in respective layers and the size of the filters are 3 x 3 with activation functions as relu. Maxpool2d : Max pooling is a pooling operation that selects the maximum element from the region of the feature map covered by the filter. Flatten: this method converts the multi-dimensional image data array to 1D array.

Then we specify optimizer and loss function for our model and also metrics which we want to visualize while training. The role of optimizer is it measure how good our model predicted output when compared with true output if the loss is high then optimizers are used to change the attributes of your neural network such as weights and learning rate in order to reduce the losses. There are several optimizers and loss functions available in TF. Here we use adam and binary crossentropy.

We can train our model by calling fit generator function which takes our training images as input for training. And also we should specify epochs and steps per epoch, which is the number of train images / batch size.

Finally we use our model to make predictions by using predict classes.