Artificial Neural Networks and Deep Learning 2020

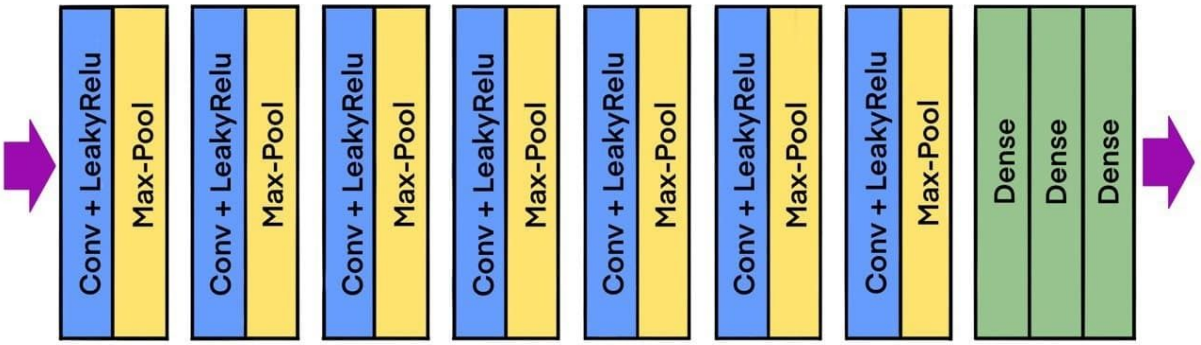
Homework 1 - Image Classification

Team: StavoCorLibbanese

Amorosini Francesco - Awad Yasmin - Fioravanti Tommaso

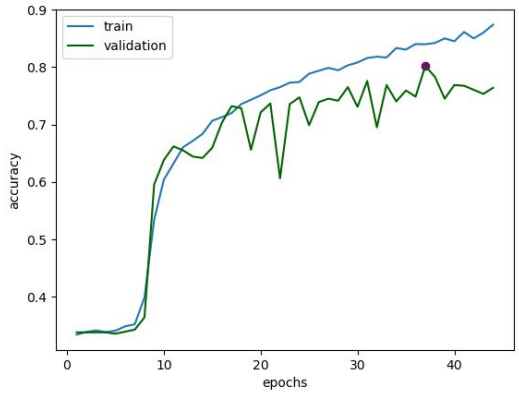
As Neural Networks suitable for solving the proposed challenge, two structures have been created, one of which with the use of Fine Tuning.

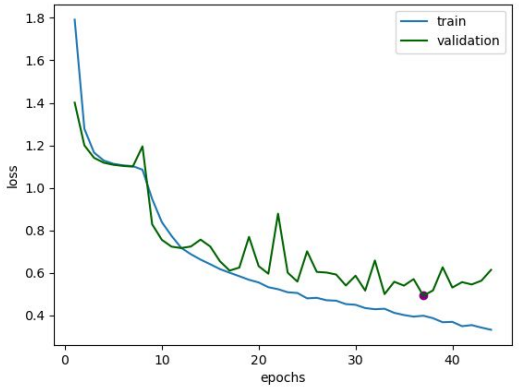
The first one we present is the result of numerous Hyperparameter Tuning experiments, which concerned the change of batch size, type of augmentation, depth, number of filters and layers, type of activation function and so on. Starting from the examples we tested during lab classes, we ended up with a structure like the one in the following figure:



Dropout and Augmentation were chosen as generalization methods: the former was inserted after each of the first two Dense layers with a probability of 0.3, whereas the latter considered a *rotational_range* of 20, a *height_shift_range* of 0.2 and *horizontal_flip*. The dataset was divided into training and validation sets with an 85:15 ratio.

In the Network training the *early stopping* method with the validation loss as value to monitor and a *patience* of 7 was used, for which we obtained a stop at the epoch 44 and best value of validation loss and validation accuracy at the epoch 37. The weights of the 37th epoch were used to apply the Network on the test set, achieving a test accuracy of 80.00%.





The second approach we propose involves using a pre-built model as a feature extractor, which we completed with customized top layers for the classification task.

Our best run was achieved by using VGG16 as a feature extractor, in which the top of the network was composed by a Flatten layer, followed by two dense layers with 512 units and a 0.001 L2 regularization factor interleaved with two dropout layers with 0.3 probability. We also apply data augmentation with *rotation_range*=10, *horizontal_flip*=True and *zoom_range*=0.2. For what concerns other hyperparameters as the learning rate, the batch size and the number of epochs, we have found after numerous tuning experiments that the best value was *1e-4* for the first and 120 for the last two. As optimization parameters we have adopted a *Categorical Crossentropy* as loss function, the *Adam* optimizer and as validation metrics the *accuracy.*

The VGG16 has a very large number of parameters, thus the top of the network underwent a meticulous tuning procedure in order to find the right balance between overall training time and overfitting. Since the pretrained weights of the VGG16 refers to the imagenet problems and since our classification task is different compared to the previous problem, we perform a fine-tuning approach where we freeze the train of the VGG16 network until the 15 layer. Also in this second procedure we have adopted the early stopping procedure (also here with validation loss as value to monitor and patience =7) and we obtained a stop at epoch 23, with the best weights at epoch 16. This procedure resulted in a test accuracy of 86.66%.

A similar approach has been applied to other pre-built models, namely Inception, Alexnet and Resnet, but they scored a lower test accuracy with respect to VGG16. Here we reported the summary of the model used in the second approach.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Functional)           (None, 8, 8, 512)         14714688
_________________________________________________________________
flatten (Flatten)            (None, 32768)             0
_________________________________________________________________
dense (Dense)                (None, 512)               16777728
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               262656
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 3)                 1539
=================================================================
Total params: 31,756,611
Trainable params: 24,121,347
Non-trainable params: 7,635,264
_________________________________________________________________
```

In the zip file you will find the notebooks of the two networks described in this paper, plus our attempt of using the Resnet as a feature extractor.