# AML Assignment 3 - report

April 27, 2020

Fabio Montello (1834411), Francesco Russo (1449025), Michele Cernigliaro (1869097)

## 1 Overview

In this report we proceed to answer as requested all the questions of the homework 3. For each point we are going to write a brief description using figures and formulas whenever needed.

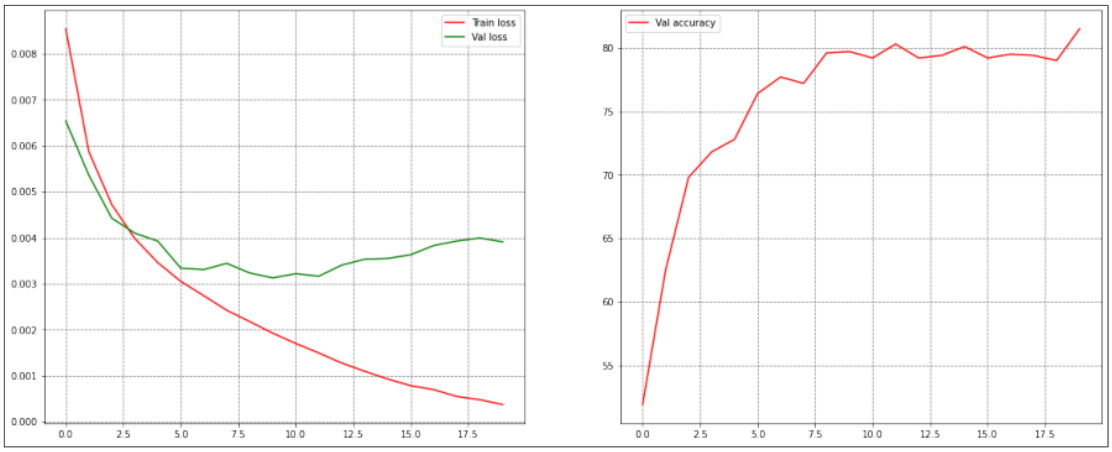
## 2 Question 1a

Here below we present the implementation with pytorch of this question:

```
class ConvNet(nn.Module):
    def __init__(self, input_size, hidden_layers, num_classes, norm_layer=None):
        super(ConvNet, self).__init__()
        layers = []
        # *****START OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)*****
        # First ConvBlock with input size (i.e. C=3) and first hidden layer(i.e. 128)
        layers.append(nn.Conv2d(input_size, hidden_layers[0], kernel_size=3, stride=1, padding=1))
        if norm_layer=="BN":
          layers.append(nn.BatchNorm2d(hidden_layers[0], eps=1e-05, momentum=0.1,
                                       affine=True, track_running_stats=True))
        layers.append(nn.ReLU())
        layers.append(nn.MaxPool2d(kernel_size=2, stride=2))

        # Adding the other blocks
        for Din, Dout in zip(hidden_layers[:-1], hidden_layers[1:]):

          layers.append(nn.Conv2d(Din, Dout, kernel_size=3, stride=1, padding=1))
          if norm_layer=="BN":
            layers.append(nn.BatchNorm2d(Dout, eps=1e-05, momentum=0.1,
                                         affine=True, track_running_stats=True))
          layers.append(nn.ReLU())
          layers.append(nn.MaxPool2d(kernel_size=2, stride=2))

        # stacking convolutional blocks
        self.ConvBlocks = nn.Sequential(*layers)
        self.Dout = hidden_layers[-1]

        # Fully connected layer
        self.Dense = nn.Linear(hidden_layers[-1], num_classes)
```

Notice that we included also the Batch Normalization as requested in the point 2a, but the code

has been executed without any normalization layer set. We will plot the traces of the loss and accuracy values along the epochs:




As final result we obtained 81.5% of accuracy on the validation and 79.3% on the test set. According to the loss graph, it seems that model is slightly overfitting starting from the seventh epoch. We will try to reduce the overfitting in the next steps with data augmentation and dropout, as requested from the homework.

## 3 Question 1b

Implementing the function 'PrintModelSize' we found out that our convnet has 7,678,474 parameters without the batch layers, and 7,682,826 considering also the new scale ($\gamma$) and shift ($\beta$) parameters to be learned in each batch layer.

# 4 Question 1c