Matteo D'Ospina s252973

Homework 3 – Deep Learning

Purpose of project:
Train several network on a large dataset (CIFAR 100). Firstly a standard Neural Network. Secondly a Convolutional Neural Network (CNN) and finally a ResNet. Taking note about measure of loss and accuracy, in order to understand the behaviour at every change of the network.

CIFAR-100:

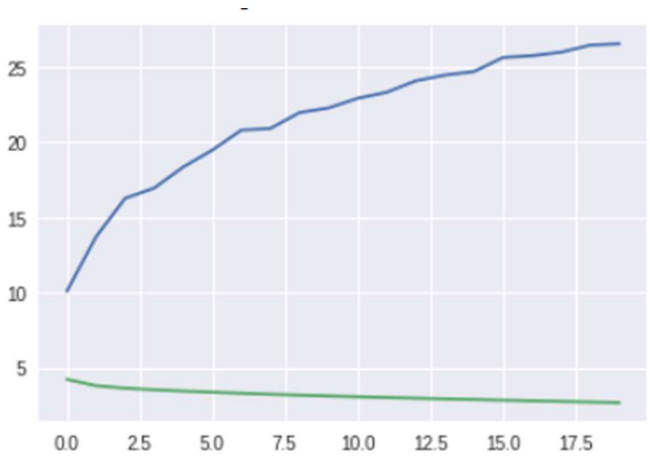
This dataset is large, consisting of 100 image classes, with 600 images per class.
Each image is 32x32x3 (3 color), and the 600 images are divided into 500 training, and 100 test for each class.

Setting of Nets:
For each net we are using a fix configuration that is a value of epochs at 20, batch size at 256 and learning rate at 0.0001.
So the model has a batch size is smaller then training sample. So at each epoch the model updates the weights of the network for 196 times, in this way we have the advantage to use less memory during the training time.

POINT 1: OLD NEURAL NETWORK

In this case it is used a standard neural network with 2 hidden layer and final full connected with 4096 for each.
At 20th epoch this network has an accuracy of 26% and a loss of 2.67 in 4 minutes. These are good values considering that dataset has 100 classes and only 500 training samples for each class. After all the classifier works well because the randomly value is 1%.
In this net is not present a strong overfitting in fact, increasing the epochs number, the network will have a better accuracy value.

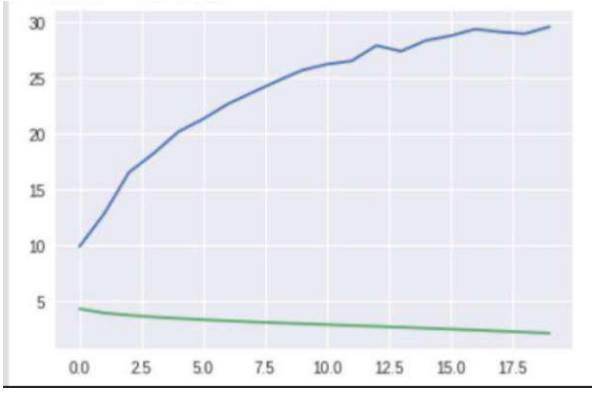
```
[1,   195] loss: 4.224
Accuracy of the network on the test set: 10 %
[2,   195] loss: 3.802
Accuracy of the network on the test set: 13 %
[3,   195] loss: 3.636
Accuracy of the network on the test set: 16 %
[4,   195] loss: 3.532
Accuracy of the network on the test set: 16 %
[5,   195] loss: 3.447
Accuracy of the network on the test set: 18 %
[6,   195] loss: 3.372
Accuracy of the network on the test set: 19 %
[7,   195] loss: 3.300
Accuracy of the network on the test set: 20 %
[8,   195] loss: 3.234
Accuracy of the network on the test set: 20 %
[9,   195] loss: 3.175
Accuracy of the network on the test set: 21 %
[10,   195] loss: 3.118
Accuracy of the network on the test set: 22 %
[11,   195] loss: 3.067
Accuracy of the network on the test set: 22 %
[12,   195] loss: 3.016
Accuracy of the network on the test set: 23 %
[13,   195] loss: 2.972
Accuracy of the network on the test set: 24 %
[14,   195] loss: 2.925
Accuracy of the network on the test set: 24 %
[15,   195] loss: 2.884
Accuracy of the network on the test set: 24 %
[16,   195] loss: 2.839
Accuracy of the network on the test set: 25 %
[17,   195] loss: 2.799
Accuracy of the network on the test set: 25 %
[18,   195] loss: 2.758
Accuracy of the network on the test set: 25 %
[19,   195] loss: 2.717
Accuracy of the network on the test set: 26 %
[20,   195] loss: 2.676
Accuracy of the network on the test set: 26 %
```



POINT 2: CONVOLUTIONAL NEURAL NETWORK

In this case it is used a Convolutional neural network with 4 convolutional layers, a pooling layer applied on last convolutional layer, a 2 fc layer with 4096 neurons.
At 20th epoch this network has an accuracy of 29% and a loss of 2.0 in 3 minutes.
There are 32 feature maps for first 3 layer and 64 feature maps for the last layer.
Comparing the net to the previous one, performance are better in less time because this net is deeper reducing the number of units in the network (since they are *many-to-one mappings*).
This means, there are fewer parameters to learn which reduces the chance of overfitting as the model would be less complex than a fully connected network.
More over convolution layers allows a better extraction of features in the training samples.
There isn't a strong overfitting, because loss value isn't low and increasing number of epochs accuracy value still grows.

```
[1,   195] loss: 4.240
Accuracy of the network on the test set: 9 %
[2,   195] loss: 3.864
Accuracy of the network on the test set: 12 %
[3,   195] loss: 3.666
Accuracy of the network on the test set: 16 %
[4,   195] loss: 3.508
Accuracy of the network on the test set: 18 %
[5,   195] loss: 3.375
Accuracy of the network on the test set: 20 %
[6,   195] loss: 3.261
Accuracy of the network on the test set: 21 %
[7,   195] loss: 3.162
Accuracy of the network on the test set: 22 %
[8,   195] loss: 3.066
Accuracy of the network on the test set: 23 %
[9,   195] loss: 2.982
Accuracy of the network on the test set: 24 %
[10,   195] loss: 2.904
Accuracy of the network on the test set: 25 %
[11,   195] loss: 2.826
Accuracy of the network on the test set: 26 %
[12,   195] loss: 2.745
Accuracy of the network on the test set: 26 %
[13,   195] loss: 2.667
Accuracy of the network on the test set: 27 %
[14,   195] loss: 2.586
Accuracy of the network on the test set: 27 %
[15,   195] loss: 2.501
Accuracy of the network on the test set: 28 %
[16,   195] loss: 2.418
Accuracy of the network on the test set: 28 %
[17,   195] loss: 2.334
Accuracy of the network on the test set: 29 %
[18,   195] loss: 2.238
Accuracy of the network on the test set: 29 %
[19,   195] loss: 2.147
Accuracy of the network on the test set: 28 %
[20,   195] loss: 2.055
Accuracy of the network on the test set: 29 %
Finished Training
```



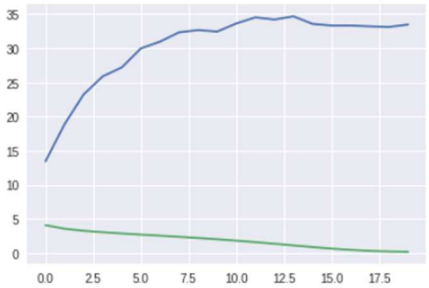
POINT 3: CONVOLUTIONAL NEURAL NETWORK CHANGING CONVOLUTIONAL FILTERS

In this case it is used the same Convolutional Neural Network but I increase progressively the number convolutional filters for each cnn layers.
I found the follow values:

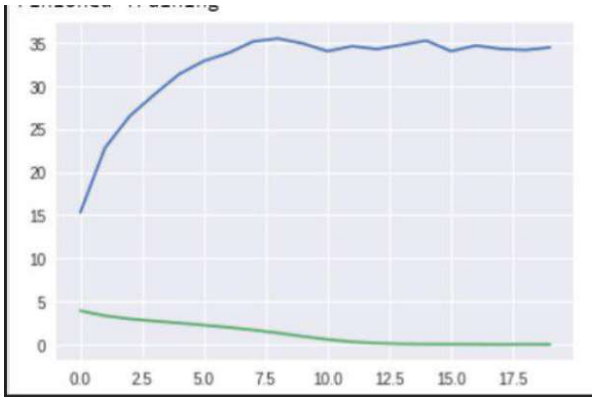
a. 128/128/128/256 33% accuracy, 0.168 loss in 6 minutes
b. 256/256/256/512 34% accuracy, 0.071 loss in 14 minutes
c. 512/512/512/1024 36% accuracy, 0.053 loss in 42 minutes

It's possible to notice from the collected data between the three different cases that accuracy doesn't increase too much increasing the number of convolutional filters. This could be due to resolution of the images that is 32x32 that don't own so much feature to extract.
It's possible to notice that the model converges to max accuracy value faster. In fact in third case, from 5th epoch the accuracy value is stable and only the loss decrease, probably the model has adapted too much its self to the dataset, leading to overfitting.

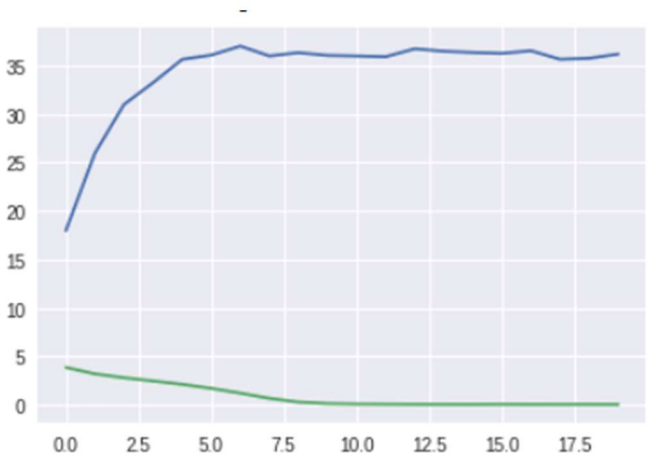
```
[1,   195] loss: 4.062
Accuracy of the network on the test set: 13 %
[2,   195] loss: 3.551
Accuracy of the network on the test set: 18 %
[3,   195] loss: 3.247
Accuracy of the network on the test set: 23 %
[4,   195] loss: 3.035
Accuracy of the network on the test set: 25 %
[5,   195] loss: 2.852
Accuracy of the network on the test set: 27 %
[6,   195] loss: 2.686
Accuracy of the network on the test set: 29 %
[7,   195] loss: 2.531
Accuracy of the network on the test set: 30 %
[8,   195] loss: 2.361
Accuracy of the network on the test set: 32 %
[9,   195] loss: 2.193
Accuracy of the network on the test set: 32 %
[10,   195] loss: 2.004
Accuracy of the network on the test set: 32 %
[11,   195] loss: 1.802
Accuracy of the network on the test set: 33 %
[12,   195] loss: 1.584
Accuracy of the network on the test set: 34 %
[13,   195] loss: 1.349
Accuracy of the network on the test set: 34 %
[14,   195] loss: 1.101
Accuracy of the network on the test set: 34 %
[15,   195] loss: 0.865
Accuracy of the network on the test set: 33 %
[16,   195] loss: 0.633
Accuracy of the network on the test set: 33 %
[17,   195] loss: 0.451
Accuracy of the network on the test set: 33 %
[18,   195] loss: 0.305
Accuracy of the network on the test set: 33 %
[19,   195] loss: 0.216
Accuracy of the network on the test set: 33 %
[20,   195] loss: 0.168
Accuracy of the network on the test set: 33 %
```



```
[1,   195] loss: 3.961
Accuracy of the network on the test set: 15 %
[2,   195] loss: 3.379
Accuracy of the network on the test set: 22 %
[3,   195] loss: 3.016
Accuracy of the network on the test set: 26 %
[4,   195] loss: 2.766
Accuracy of the network on the test set: 28 %
[5,   195] loss: 2.537
Accuracy of the network on the test set: 31 %
[6,   195] loss: 2.295
Accuracy of the network on the test set: 32 %
[7,   195] loss: 2.037
Accuracy of the network on the test set: 33 %
[8,   195] loss: 1.736
Accuracy of the network on the test set: 35 %
[9,   195] loss: 1.390
Accuracy of the network on the test set: 35 %
[10,   195] loss: 0.995
Accuracy of the network on the test set: 34 %
[11,   195] loss: 0.642
Accuracy of the network on the test set: 33 %
[12,   195] loss: 0.364
Accuracy of the network on the test set: 34 %
[13,   195] loss: 0.208
Accuracy of the network on the test set: 34 %
[14,   195] loss: 0.124
Accuracy of the network on the test set: 34 %
[15,   195] loss: 0.099
Accuracy of the network on the test set: 35 %
[16,   195] loss: 0.092
Accuracy of the network on the test set: 33 %
[17,   195] loss: 0.084
Accuracy of the network on the test set: 34 %
[18,   195] loss: 0.072
Accuracy of the network on the test set: 34 %
[19,   195] loss: 0.086
Accuracy of the network on the test set: 34 %
[20,   195] loss: 0.071
Accuracy of the network on the test set: 34 %
Finished Training
```




```
[1,   195] loss: 3.869
Accuracy of the network on the test set: 17 %
[2,   195] loss: 3.206
Accuracy of the network on the test set: 25 %
[3,   195] loss: 2.804
Accuracy of the network on the test set: 30 %
[4,   195] loss: 2.469
Accuracy of the network on the test set: 33 %
[5,   195] loss: 2.127
Accuracy of the network on the test set: 35 %
[6,   195] loss: 1.709
Accuracy of the network on the test set: 36 %
[7,   195] loss: 1.220
Accuracy of the network on the test set: 37 %
[8,   195] loss: 0.678
Accuracy of the network on the test set: 35 %
[9,   195] loss: 0.297
Accuracy of the network on the test set: 36 %
[10,   195] loss: 0.145
Accuracy of the network on the test set: 36 %
[11,   195] loss: 0.097
Accuracy of the network on the test set: 35 %
[12,   195] loss: 0.082
Accuracy of the network on the test set: 35 %
[13,   195] loss: 0.065
Accuracy of the network on the test set: 36 %
[14,   195] loss: 0.062
Accuracy of the network on the test set: 36 %
[15,   195] loss: 0.056
Accuracy of the network on the test set: 36 %
[16,   195] loss: 0.070
Accuracy of the network on the test set: 36 %
[17,   195] loss: 0.060
Accuracy of the network on the test set: 36 %
[18,   195] loss: 0.062
Accuracy of the network on the test set: 35 %
[19,   195] loss: 0.066
Accuracy of the network on the test set: 35 %
[20,   195] loss: 0.053
Accuracy of the network on the test set: 36 %
```

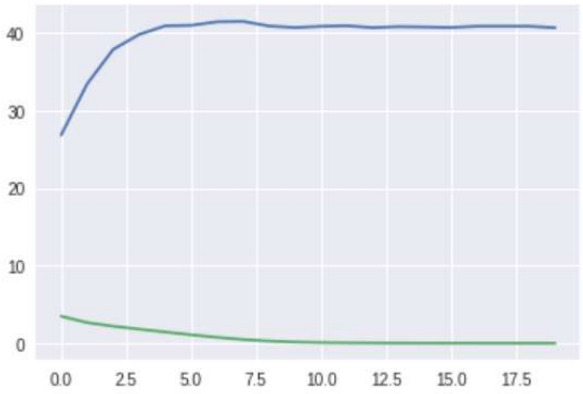


Point 4: BATCH NORMALIZATION E DROPOUT

In this point the use of batch normalization and dropout is required. Both techniques are regularization technique and are used in order to improve performance in the presence of overfitting.

4a) Is used only batch normalization after each Convolutional layer. In the code the layers are define in init pass in this way `self.conv2_batchn =nn.BatchNorm2d(128)` than used in forwad pass in this way `x = self.conv1_batchn(F.relu(self.conv1(x)))`; The normalization is $x' = \frac{x - E(x)}{\sqrt{Var(x)}}$. The transformation is executed over batches. This normalization prevents the activations to become too high or too low. This is the result with the same network configuration of 3a) point with add bn layers.

```
[1,   195] loss: 3.484
Accuracy of the network on the test set: 26 %
[2,   195] loss: 2.660
Accuracy of the network on the test set: 33 %
[3,   195] loss: 2.201
Accuracy of the network on the test set: 37 %
[4,   195] loss: 1.818
Accuracy of the network on the test set: 39 %
[5,   195] loss: 1.452
Accuracy of the network on the test set: 40 %
[6,   195] loss: 1.094
Accuracy of the network on the test set: 40 %
[7,   195] loss: 0.760
Accuracy of the network on the test set: 41 %
[8,   195] loss: 0.482
Accuracy of the network on the test set: 41 %
[9,   195] loss: 0.280
Accuracy of the network on the test set: 40 %
[10,   195] loss: 0.160
Accuracy of the network on the test set: 40 %
[11,   195] loss: 0.093
Accuracy of the network on the test set: 40 %
[12,   195] loss: 0.059
Accuracy of the network on the test set: 40 %
[13,   195] loss: 0.040
Accuracy of the network on the test set: 40 %
[14,   195] loss: 0.030
Accuracy of the network on the test set: 40 %
[15,   195] loss: 0.023
Accuracy of the network on the test set: 40 %
[16,   195] loss: 0.020
Accuracy of the network on the test set: 40 %
[17,   195] loss: 0.017
Accuracy of the network on the test set: 40 %
[18,   195] loss: 0.015
Accuracy of the network on the test set: 40 %
[19,   195] loss: 0.013
Accuracy of the network on the test set: 40 %
[20,   195] loss: 0.012
Accuracy of the network on the test set: 40 %
Finished Training
```

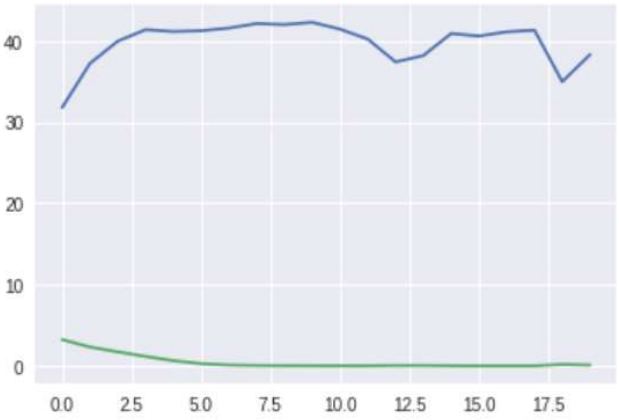


It's possible to see a strong increase in performance from 33% in 6 minutes to 40% in only 7 minutes. So BatchNorm is introduced to reduce overfitting but the method has other advantages

- It allows an higher learning rate so it should accelerate the training time
- It decreases the dependence of weights initialization
- It can render Dropout useless

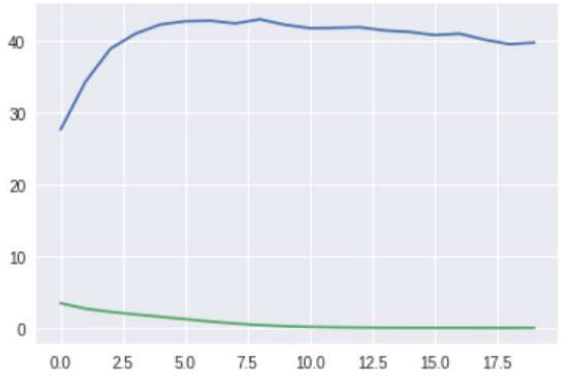
4b) In this point is used the same network but with a bigger number of neurons only in the last fully connected network from 4096 to 8192. In this case It's possible to see that it's not said that increasing the number of neurons the performance increases. There is no perfect number of neurons, but having too high increases the calculation time and the possibility of generating noise, too low reduces the capacity of network.

```
[1,   390] loss: 3.238
Accuracy of the network on the test set: 31 %
[2,   390] loss: 2.323
Accuracy of the network on the test set: 37 %
[3,   390] loss: 1.726
Accuracy of the network on the test set: 39 %
[4,   390] loss: 1.163
Accuracy of the network on the test set: 41 %
[5,   390] loss: 0.639
Accuracy of the network on the test set: 41 %
[6,   390] loss: 0.277
Accuracy of the network on the test set: 41 %
[7,   390] loss: 0.110
Accuracy of the network on the test set: 41 %
[8,   390] loss: 0.054
Accuracy of the network on the test set: 42 %
[9,   390] loss: 0.032
Accuracy of the network on the test set: 41 %
[10,   390] loss: 0.023
Accuracy of the network on the test set: 42 %
[11,   390] loss: 0.020
Accuracy of the network on the test set: 41 %
[12,   390] loss: 0.020
Accuracy of the network on the test set: 40 %
[13,   390] loss: 0.050
Accuracy of the network on the test set: 37 %
[14,   390] loss: 0.055
Accuracy of the network on the test set: 38 %
[15,   390] loss: 0.020
Accuracy of the network on the test set: 40 %
[16,   390] loss: 0.009
Accuracy of the network on the test set: 40 %
[17,   390] loss: 0.010
Accuracy of the network on the test set: 41 %
[18,   390] loss: 0.009
Accuracy of the network on the test set: 41 %
[19,   390] loss: 0.182
Accuracy of the network on the test set: 34 %
[20,   390] loss: 0.109
Accuracy of the network on the test set: 38 %
Finished Training
```



4c) In this point are used together BatchNormalization and Dropout. Compared to point 4a), the code is the same but is add a dropout layer in cnn init `self.dropfc = nn.Dropout(0.5)` and in the forward `x = self.dropfc(x)` just before the last fully connected layer. Dropout is a technique where randomly selected neurons are ignored during training. They are “dropped-out” randomly. This means that their contribution to the activation of downstream neurons is temporally removed on the forward pass and any weight updates are not applied to the neuron on the backward pass.
The result is the following:

```
[1,   390] loss: 3.488
Accuracy of the network on the test set: 27 %
[2,   390] loss: 2.709
Accuracy of the network on the test set: 34 %
[3,   390] loss: 2.272
Accuracy of the network on the test set: 38 %
[4,   390] loss: 1.917
Accuracy of the network on the test set: 41 %
[5,   390] loss: 1.586
Accuracy of the network on the test set: 42 %
[6,   390] loss: 1.252
Accuracy of the network on the test set: 42 %
[7,   390] loss: 0.928
Accuracy of the network on the test set: 42 %
[8,   390] loss: 0.650
Accuracy of the network on the test set: 42 %
[9,   390] loss: 0.427
Accuracy of the network on the test set: 43 %
[10,   390] loss: 0.274
Accuracy of the network on the test set: 42 %
[11,   390] loss: 0.176
Accuracy of the network on the test set: 41 %
[12,   390] loss: 0.119
Accuracy of the network on the test set: 41 %
[13,   390] loss: 0.087
Accuracy of the network on the test set: 41 %
[14,   390] loss: 0.067
Accuracy of the network on the test set: 41 %
[15,   390] loss: 0.052
Accuracy of the network on the test set: 41 %
[16,   390] loss: 0.048
Accuracy of the network on the test set: 40 %
[17,   390] loss: 0.049
Accuracy of the network on the test set: 41 %
[18,   390] loss: 0.047
Accuracy of the network on the test set: 40 %
[19,   390] loss: 0.043
Accuracy of the network on the test set: 39 %
[20,   390] loss: 0.052
Accuracy of the network on the test set: 39 %
Finished Training
```



It's possible to see that dropout does not add any value to the increase in accuracy, on the contrary it seems not working. Probably the work of the dropout is rendered useless by the batch normalization, in fact trying to perform a train removing the layers of the batch normalization and adding only the dropout you can see results similar to those of point 4a. So probably BatchNormalization and DropOut come to the same conclusions and in this case it does not make much sense to use them together.

Point 5: Convolutional Neural Network with data augmentation

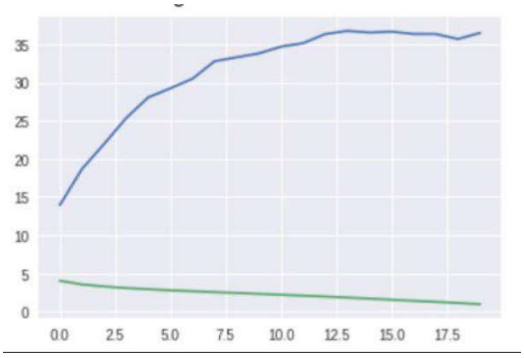
In this case is used 3a configuration, using different data augmentation techniques. Adding more data in the training set is one of the best ways to improve the performance of a Deep Learning model.
There are many ways to augment existing datasets and produce more robust models. In this case are used two techniques, Flip Horizontal Flipping, that reverses the active layer horizontally, that is, from left to right. It leaves the dimensions of the layer and the pixel information unchanged.
Random Crop, performs training on causal image clippings to try to vary the position and orientation of the subject in the image.

The following values were found:

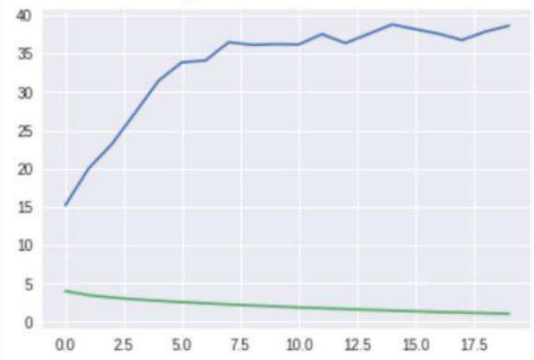
a. Flip horizontal flipping applied with the following function `transforms.RandomHorizontalFlip(),` in the transform train, it produced accuracy value of 36% and a loss value of 0.982 in 7 minutes
b. Random Crop applied with
```
transforms.Resize((40,40)),
transforms.RandomCrop([32, 32]),
```
in the same place, produced accuracy value of 38 % and a loss value of 1.065 in 8 minutes

Using these techniques is a good way to increase the width of the dataset and accordingly the performance of the net. In fact an appreciable improvement is visible, considering that the network does not use nor normalization nor drop out.

```
[1,   195] loss: 4.036
Accuracy of the network on the test set: 13 %
[2,   195] loss: 3.565
Accuracy of the network on the test set: 18 %
[3,   195] loss: 3.289
Accuracy of the network on the test set: 21 %
[4,   195] loss: 3.085
Accuracy of the network on the test set: 25 %
[5,   195] loss: 2.931
Accuracy of the network on the test set: 28 %
[6,   195] loss: 2.792
Accuracy of the network on the test set: 29 %
[7,   195] loss: 2.671
Accuracy of the network on the test set: 30 %
[8,   195] loss: 2.549
Accuracy of the network on the test set: 32 %
[9,   195] loss: 2.441
Accuracy of the network on the test set: 33 %
[10,   195] loss: 2.323
Accuracy of the network on the test set: 33 %
[11,   195] loss: 2.210
Accuracy of the network on the test set: 34 %
[12,   195] loss: 2.086
Accuracy of the network on the test set: 35 %
[13,   195] loss: 1.967
Accuracy of the network on the test set: 36 %
[14,   195] loss: 1.838
Accuracy of the network on the test set: 36 %
[15,   195] loss: 1.698
Accuracy of the network on the test set: 36 %
[16,   195] loss: 1.555
Accuracy of the network on the test set: 36 %
[17,   195] loss: 1.409
Accuracy of the network on the test set: 36 %
[18,   195] loss: 1.275
Accuracy of the network on the test set: 36 %
[19,   195] loss: 1.120
Accuracy of the network on the test set: 35 %
[20,   195] loss: 0.982
Accuracy of the network on the test set: 36 %
Finished Training
```



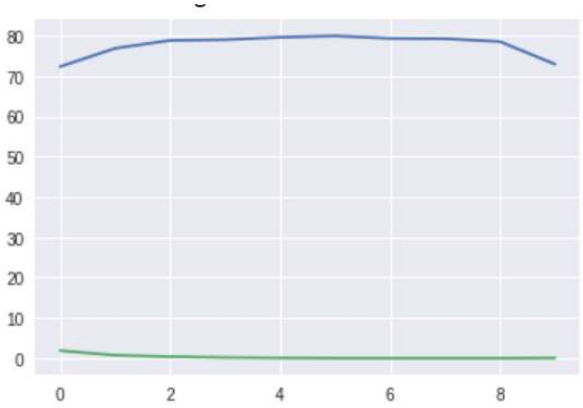

```
[1,   195] loss: 4.009
Accuracy of the network on the test set: 15 %
[2,   195] loss: 3.467
Accuracy of the network on the test set: 20 %
[3,   195] loss: 3.163
Accuracy of the network on the test set: 23 %
[4,   195] loss: 2.921
Accuracy of the network on the test set: 27 %
[5,   195] loss: 2.732
Accuracy of the network on the test set: 31 %
[6,   195] loss: 2.561
Accuracy of the network on the test set: 33 %
[7,   195] loss: 2.411
Accuracy of the network on the test set: 34 %
[8,   195] loss: 2.256
Accuracy of the network on the test set: 36 %
[9,   195] loss: 2.130
Accuracy of the network on the test set: 36 %
[10,   195] loss: 1.996
Accuracy of the network on the test set: 36 %
[11,   195] loss: 1.875
Accuracy of the network on the test set: 36 %
[12,   195] loss: 1.763
Accuracy of the network on the test set: 37 %
[13,   195] loss: 1.656
Accuracy of the network on the test set: 36 %
[14,   195] loss: 1.551
Accuracy of the network on the test set: 37 %
[15,   195] loss: 1.454
Accuracy of the network on the test set: 38 %
[16,   195] loss: 1.365
Accuracy of the network on the test set: 38 %
[17,   195] loss: 1.267
Accuracy of the network on the test set: 37 %
[18,   195] loss: 1.207
Accuracy of the network on the test set: 36 %
[19,   195] loss: 1.125
Accuracy of the network on the test set: 37 %
[20,   195] loss: 1.065
Accuracy of the network on the test set: 38 %
Finished Training
```



Point 6: ResNet

In this case, instead of training the network from scratch it's loaded a pretrained network called Resnet trained on a large dataset named Imagenet, that has more than 14 million images and about 20 miles classes. This network uses a residual network architecture and it solves the convergence and degradation of performance due to a strong gradient vanish problem for deeper networks.

```
[1,   390] loss: 1.883
Accuracy of the network on the test set: 72 %
[2,   390] loss: 0.746
Accuracy of the network on the test set: 76 %
[3,   390] loss: 0.394
Accuracy of the network on the test set: 78 %
[4,   390] loss: 0.181
Accuracy of the network on the test set: 78 %
[5,   390] loss: 0.075
Accuracy of the network on the test set: 79 %
[6,   390] loss: 0.031
Accuracy of the network on the test set: 79 %
[7,   390] loss: 0.018
Accuracy of the network on the test set: 79 %
[8,   390] loss: 0.021
Accuracy of the network on the test set: 79 %
[9,   390] loss: 0.016
Accuracy of the network on the test set: 78 %
[10,   390] loss: 0.067
Accuracy of the network on the test set: 72 %
```



Output values are really good and are expected, thank to a better thanks to better training on a huge dataset and a better network architecture.

Better CNN found from points:

The most optimized CNN found from the points of homework is 4a) because has the highest accuracy value and required less time for training.

Better CNN found freely:

The Better net found has the following setting: Adam solver with learning rate 0.001, batch size at 12, batch normalization after each Convolutional Neural Network and a DropOut at 0.8 before the last fully connected layer. Data augmentation is used with random horizontal flip. Result values is 52% of accuracy

```
[1,   390] loss: 3.992
Accuracy of the network on the test set: 23 %
[2,   390] loss: 3.279
Accuracy of the network on the test set: 30 %
[3,   390] loss: 2.940
Accuracy of the network on the test set: 35 %
[4,   390] loss: 2.653
Accuracy of the network on the test set: 39 %
[5,   390] loss: 2.440
Accuracy of the network on the test set: 42 %
[6,   390] loss: 2.242
Accuracy of the network on the test set: 44 %
[7,   390] loss: 2.084
Accuracy of the network on the test set: 46 %
[8,   390] loss: 1.945
Accuracy of the network on the test set: 48 %
[9,   390] loss: 1.830
Accuracy of the network on the test set: 49 %
[10,   390] loss: 1.716
Accuracy of the network on the test set: 49 %
[11,   390] loss: 1.619
Accuracy of the network on the test set: 50 %
[12,   390] loss: 1.532
Accuracy of the network on the test set: 50 %
[13,   390] loss: 1.435
Accuracy of the network on the test set: 50 %
[14,   390] loss: 1.352
Accuracy of the network on the test set: 51 %
[15,   390] loss: 1.272
Accuracy of the network on the test set: 51 %
[16,   390] loss: 1.200
Accuracy of the network on the test set: 52 %
[17,   390] loss: 1.131
Accuracy of the network on the test set: 52 %
[18,   390] loss: 1.075
Accuracy of the network on the test set: 52 %
[19,   390] loss: 1.014
Accuracy of the network on the test set: 52 %
[20,   390] loss: 0.968
Accuracy of the network on the test set: 52 %
Finished Training
```