Deep Learning Lab
Assignment 2

Pietro Miotti

November 17, 2021

1 Dataset

1.1 Ex 1)

Please find the code related to the import of the dataset from line 69 to 72 of the submitted file *main.py*. The train set contains 50000 images, the test set instead contains 10000.
Please find in the Fig:1 three random pictures taken from the train set. The code used to plot some pictures is commented from line 312 of the *main.py* file.

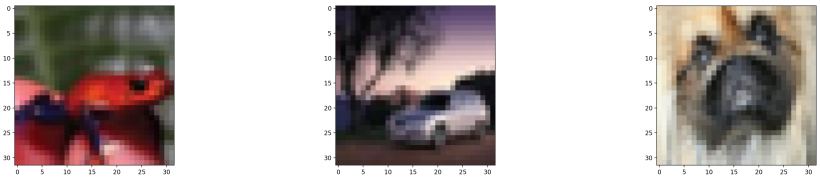

Figure 1: Three random pictures taken from the train set

1.2 Ex 2)

In order to normalize the data I created an ad hoc function *getMeanAndStd* in order to get the mean and the std from a set. I applied that function to the CIFAR10 train set and once I have obtained the mean and the variance (from line 76 on) I have normalized the dataset using the compose function as suggested. Please noticed that I have used the same normalization parameters for both train and test sets.

1.3 Ex 3)

In order to get the validation set, I used the method *SubsetRandomSampler*, as suggested. Please, you can find the relative code from line 103.

2 Model

2.1 Ex 1)

Please you can find the implementation of the model from line 125. To be more flexible for the following tasks of the assignment (especially for the implementation of the dropout), I decided to add an extra parameter to the model *apply_dropout* which is a boolean and is True if we want to use Dropout and False if we do not want to use dropout. Please notice that the softmax layer is not present in the model, since it is already implicitly defined when using the CrossEntropy Loss function.

3 Training

3.1 Ex 1)

Please find the training implementation within the function *train_and_validate* (from line 167) which trains and validate the model taken as input. Please noticed that also in this case I added an extra parameter *dropout* since for the training of a model which uses dropout we need more epochs to achieve convergence. I monitor the training loss and accuracy after 200 steps and after each epoch I perform the validation test. I monitor also the best validation accuracy and the corresponding epoch.

3.2 Ex 2)

Please consider the hyperparameters from line 26.

3.3 Ex 3)

Model without dropout:

- Best validation accuracy: 73.3 %
- Epoch Best validation accuracy: 16 (starting from 0)

3.4 Ex 4)

By looking at the plots reported below (comparison between training accuracy and validation accuracy in Fig: 2, and comparison between training loss and validation loss in Fig: 3), it is possible to see that the training accuracy is greater w.r.t to the one in validation and at the same time the error in validation starts to increase after the 13th epoch. This happens because the model starts overfitting and thus adjusting the parameter in order to minimize completely the error of the training while loosing in generalization (this is why the error in

validation starts to increase), for this purpose maybe early-stopping might be a good choice, or maybe use dropout.

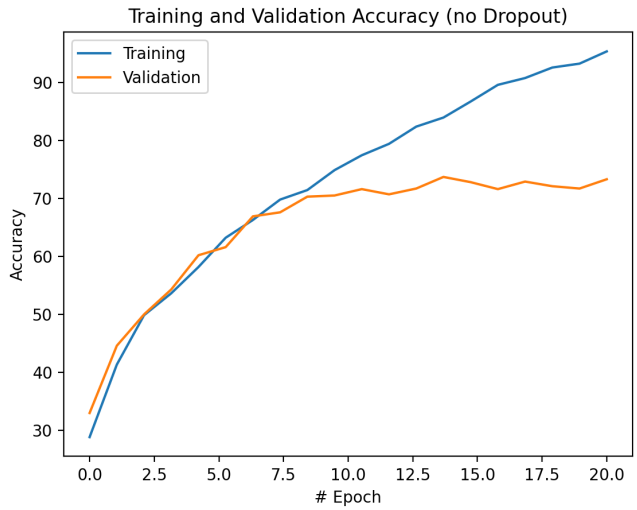

Figure 2: Training and Validation accuracy (model without Dropout)

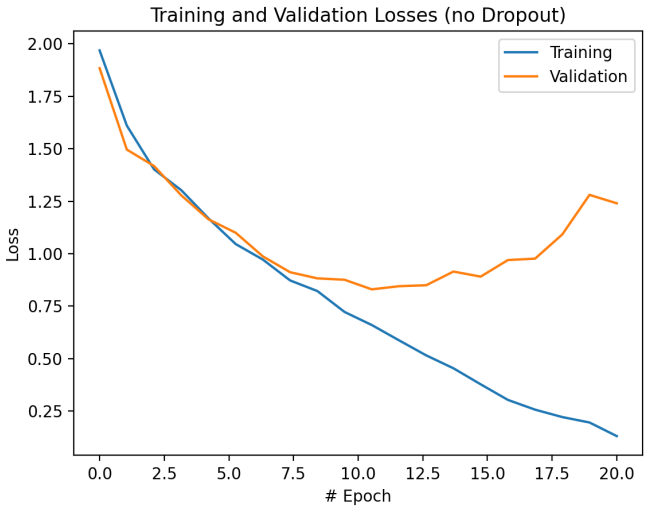

Figure 3: Training and Validation Losses (model without Dropout)

3.5 Ex 5)

Please find the comparison between the model Train and Validation accuracy in Fig: 4 and losses in fig 5 and the comparison between the accuracy of the model with and without dropout in Fig 6. We can clearly see that the model benefits from the use of dropout both in term of peak of accuracy and in term of regularization.

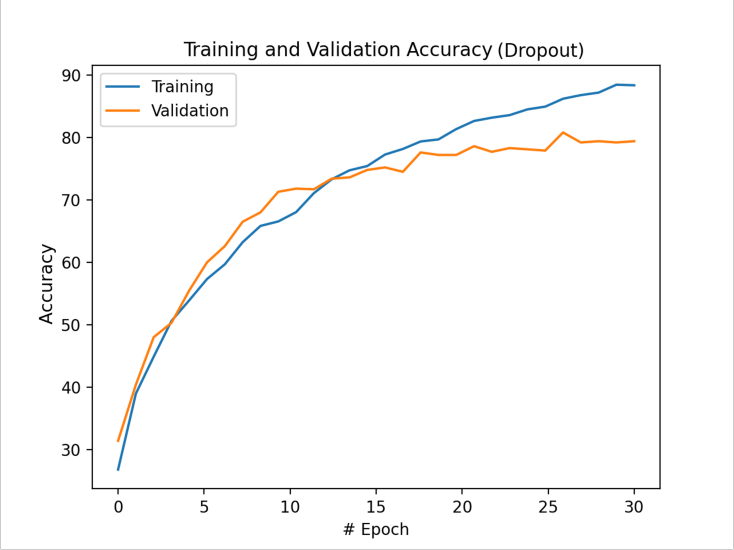

Figure 4: Training and Validation accuracy comparison (model with Dropout)

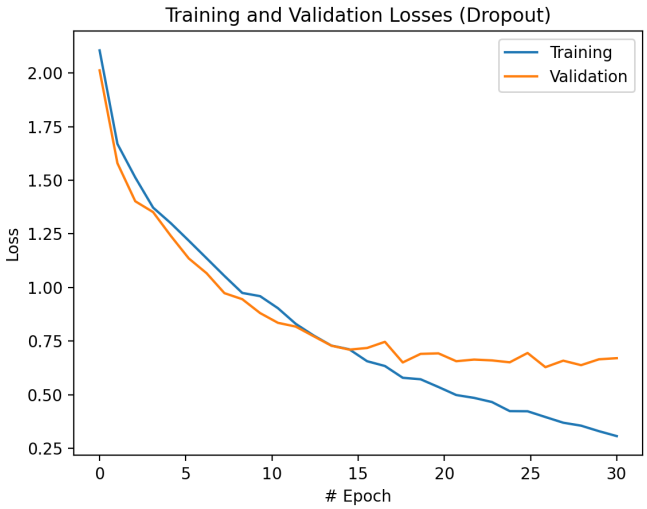

Figure 5: Training and Validation loss comparison (model with Dropout)

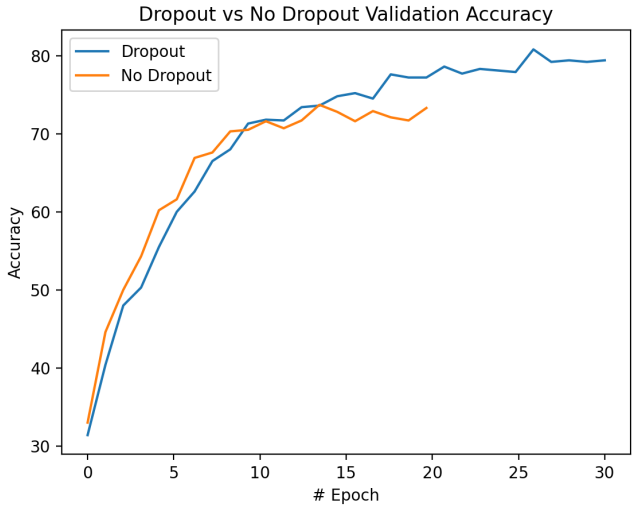

Figure 6: Dropout vs No Dropout model accuracy

3.6 Ex 6)

By implementing dropout I have managed to achieve the following results. Model without dropout:

- Best validation accuracy: 79.0 %
- Epoch Best validation accuracy: 29 (starting from 0)

3.7 Ex 7)

By testing the model with dropout I have managed to achieve an accuracy of 76.75 %

4 Questions

4.1 ex1

The softmax layer is the one that converts the output of a linear layer into probabilities (hence numbers between 0 and 1 such that the overall sum is equal to 1). It is the last layer of the network because for classification problems each digit of the output layer should represent the probability of belonging to the corresponding class and thus, in order to perform the optimization (using CrossEntropy), it is necessary to convert everything in probabilities.

4.2 ex2

The momentum parameter determines how quickly the contributions of previous gradients exponentially decay. It is a good idea to keep it non-zero because it leads to a faster convergence w.r.t to the stochastic gradient without momentum.

4.3 ex3

The main difference between convolutional layers and fully connected layers is the fact that convolutional layers try to learn from local information parameterizing kernels which are smaller than the input size, thus they do not use as many parameters as the input size, as instead fully connected layers do. This is a good idea when processing images because in images there exits an inductive bias given by the local proximity of the pixels which is exploited by the usage of kernels.

4.4 ex4

I might use 1D convolution for signals or text inputs.

4.5 ex5

Dropout, as we have seen from the plots reported above, is a technique used for regularization. Practically speaking dropout trains the ensemble consisting of all the subnetworks that can be created by removing units belonging to non-output layer from the underlying base network.