Deep Learning Lab 2

Filippo Casari

November 18, 2021

Dataset

1.1

The number of images of the dataset is 50'000. I will show in the Fig. 1 the plot of soma images.

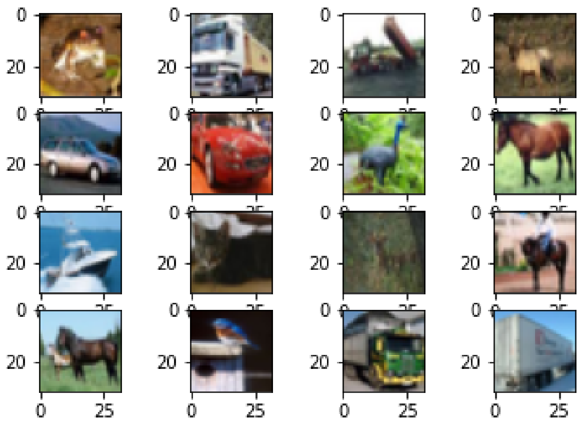

Figure 1: Some images of the Dataset

1.2

I computed the mean and the standard deviation of each channel of the training and test set, and I normalize them with the function Normalize provided by transforms of torchvision.

1.3

I split the original dataset of training set into a new training set and validation set by using SubsetRandomSampler method.

1.2 Model

1.1

I implemented the Network with the class "Net".

1.3 Training

1.

I monitor the the current training loss each 200 steps by printing it on console. I chose n=200 because we have about 1400 batch and so 200 can be a reasonable value.

3.

By training the model, I got a validation accuracy about 66 percent, closed to 70.

4.

Plot the evolution of the loss and accuracy for your training and validation set. Give a brief analysis and explanation for the discrepancies and similiarties between the training and validation metrics throughout training.

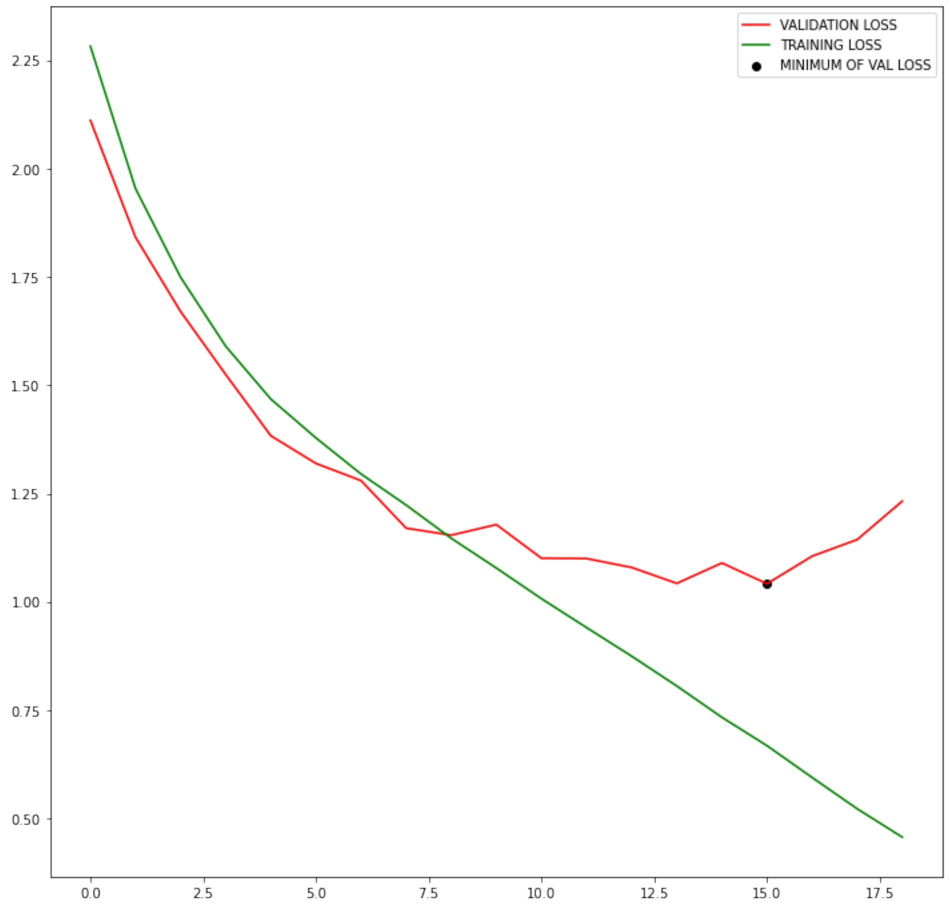

Figure 2: Validation and Training Loss

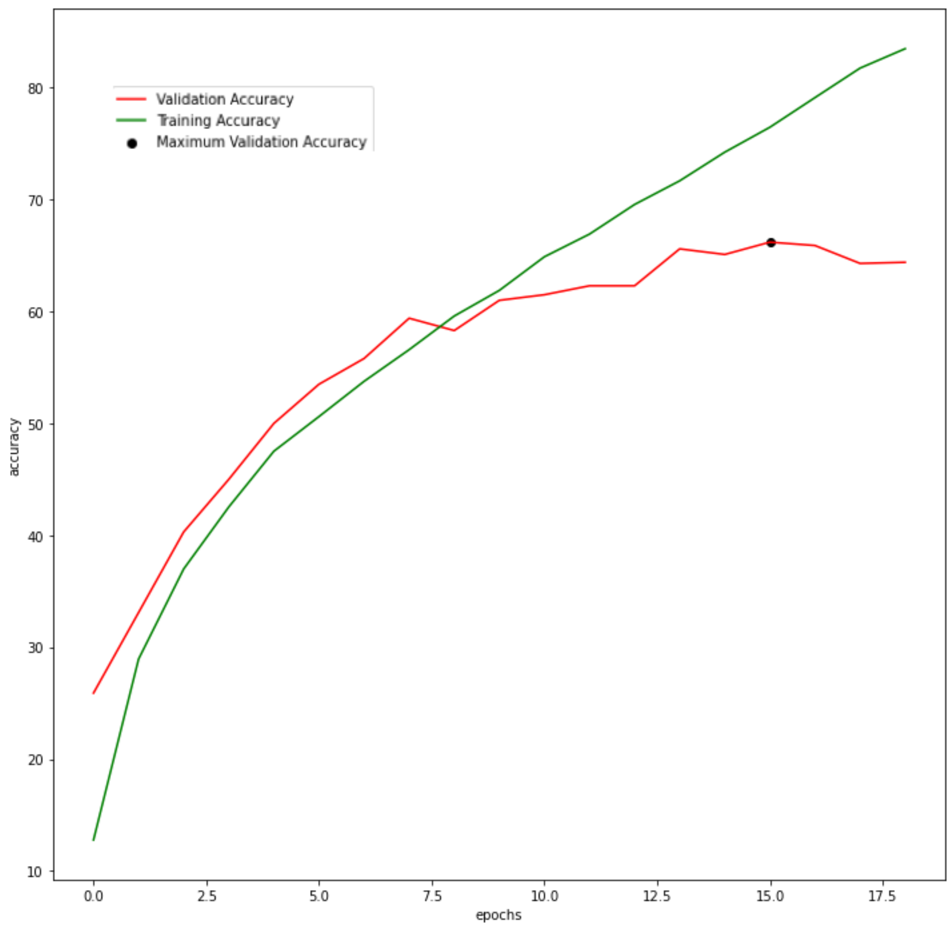

Figure 3: Validation and Training Accuracy

These two plots show that the model is **overfitting** because we got the training loss really low, decreasing quickly, and the validation loss at the epoch 15 that increases. This means that the model learned very well the data from the training set but it is not be able to generalize the problem. Moreover, the accuracy of validation grows up but at a specific point (15th epoch) it remains constant: the model does not improve his knowledge at all and performs worse than before.
The accuracy of final validation after 20 epochs is about 66 percent.

5.

I got a better results adding the drop out to avoid the overfitting as showed in the images below. I tried to set the probability of the drop out after the Convolutional layers to 0.15 and after the Fully-Connected NN set to 0.20. I set these values because for Convolutional it may be dangerous set them too high, network could forgot precious informations about the input and spatiality. For the Fully connected it is different because we have no informations about spatiality.
The fact that we got a validation loss below the training loss is caused maybe by the validation set itself. In fact, when we split the dataset we chose a small part of it, maybe with few outliers, and that could cause a really good prediction on validation set. The model does not overfit: as showed in the Fig. 4 the validation loss and the training loss are very low and they are closed. This means that the model does not overfit like before without dropout and the model generalized, performing well. I got the best validation accuracy about 74 percent.

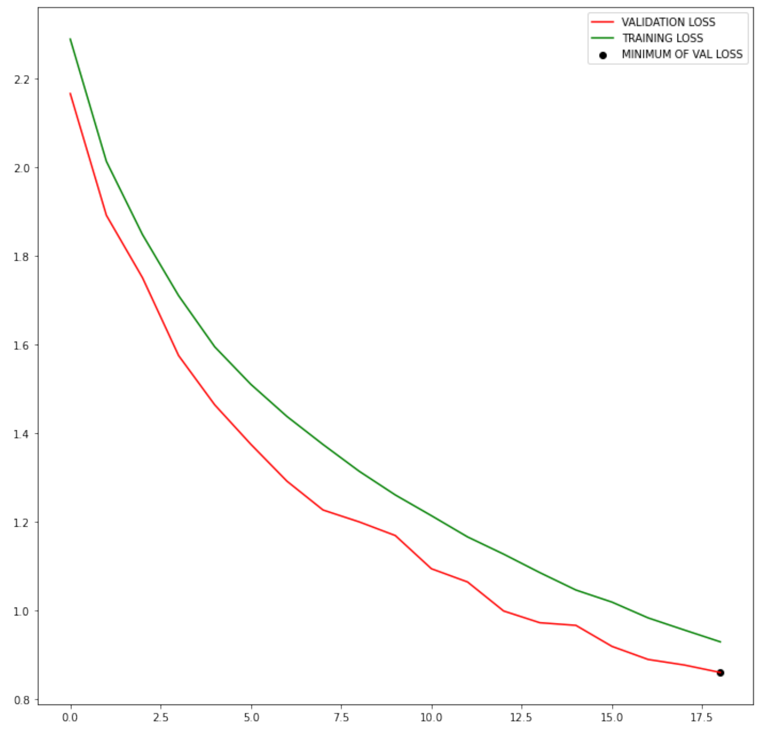

Figure 4: Validation and Training Loss with Dropout

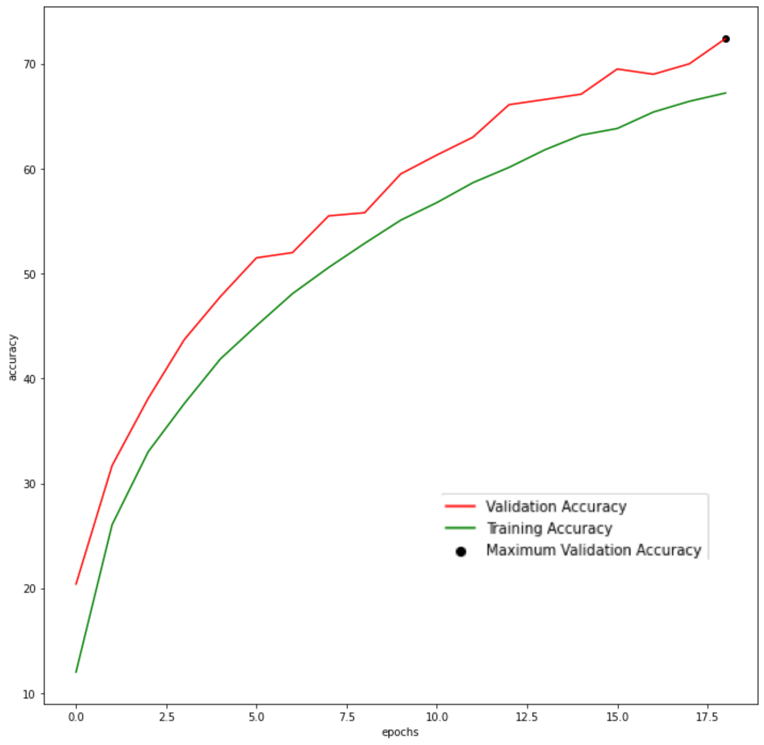

Figure 5: Validation and Training Accuracy with Dropout

6.

To improve my performance i tried to modify the number of epochs and the probability of the drop out. I increased the probability from 0.15 to 0.20 for the first dropout, from 0.20 to 0.30 for the second dropout. I wanted to see the performances keeping the same number of epochs and then increasing them (Fig. 6 and Fig. 7). We can see that there is not a really changing of the behaviour. Then, I tried to increase the the number of epochs from 20 to 35 to get better results. As showed in the Figures 8 and 9, the validation accuracy increases and the loss validation really goes down. Consequently, the model performs better at the epoch 32. With dropout we sacrificed the performance of training to generalize better and get better results of validation set.

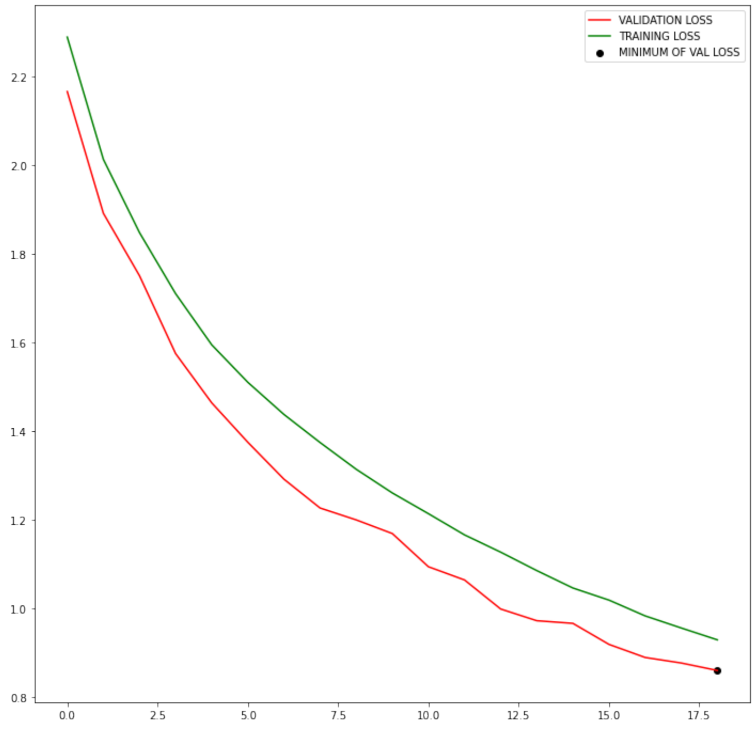

Figure 6: Validation and Training Loss with changed values of Dropout

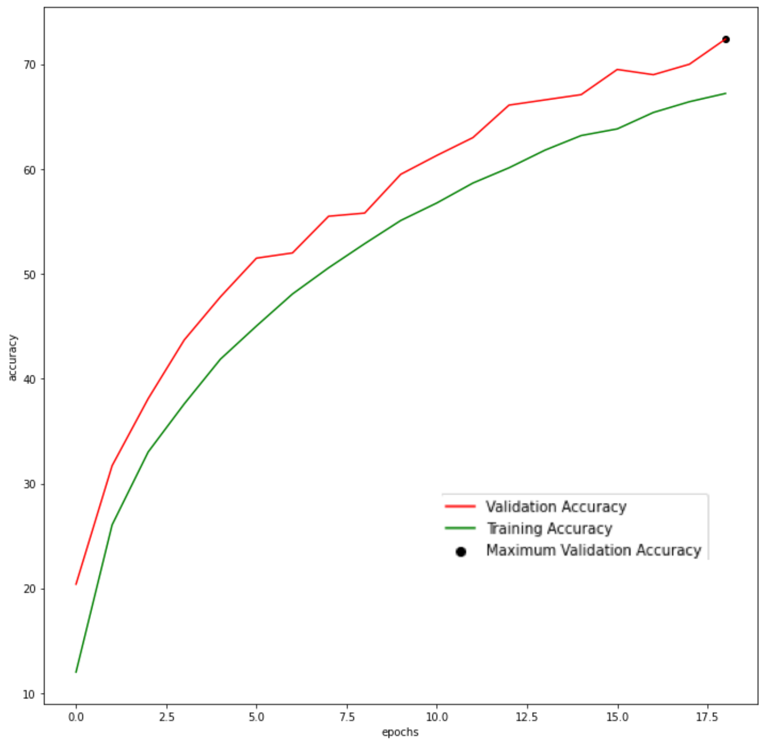

Figure 7: Validation and Training Accuracy with changed values of Dropout

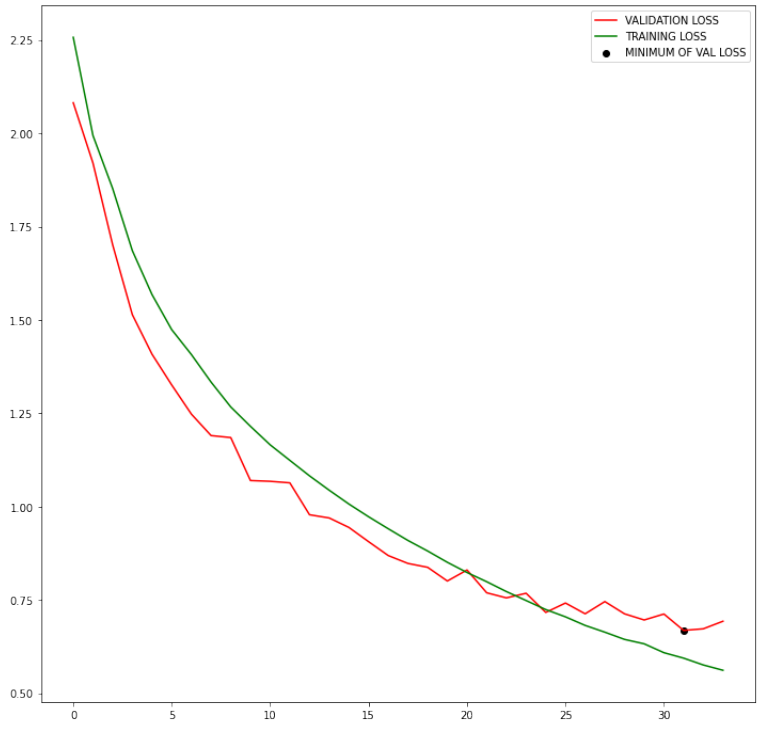

Figure 8: Validation and Training Loss with changed values of Dropout and number of epochs

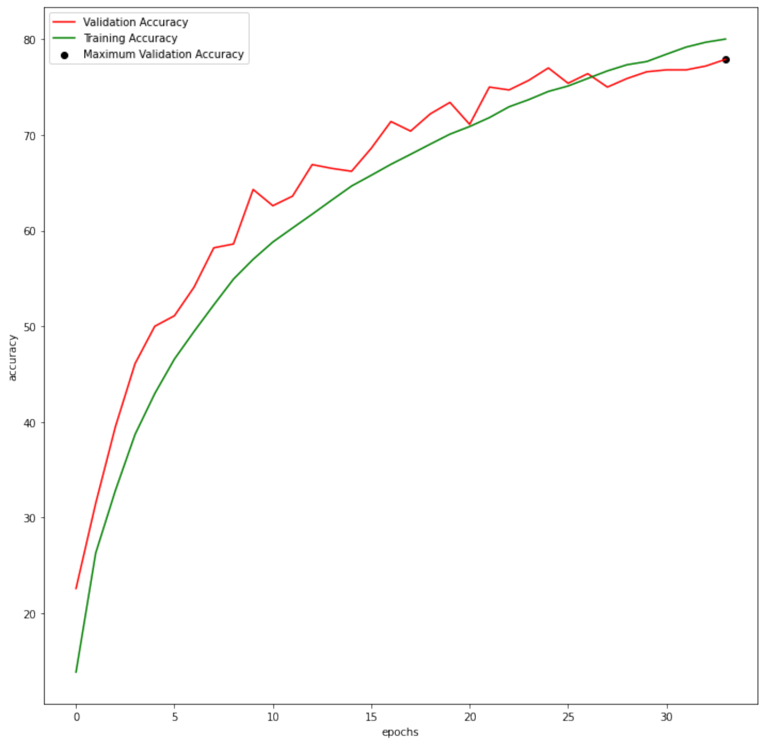

Figure 9: Validation and Training Accuracy with changed values of Dropout and number of epochs

I increased to 45 the number of epochs to have more accuracy with the validation set and indeed I got a better results. The best validation accuracy increased to 77.9 percent.(Fig. 11 and Fig. 10).

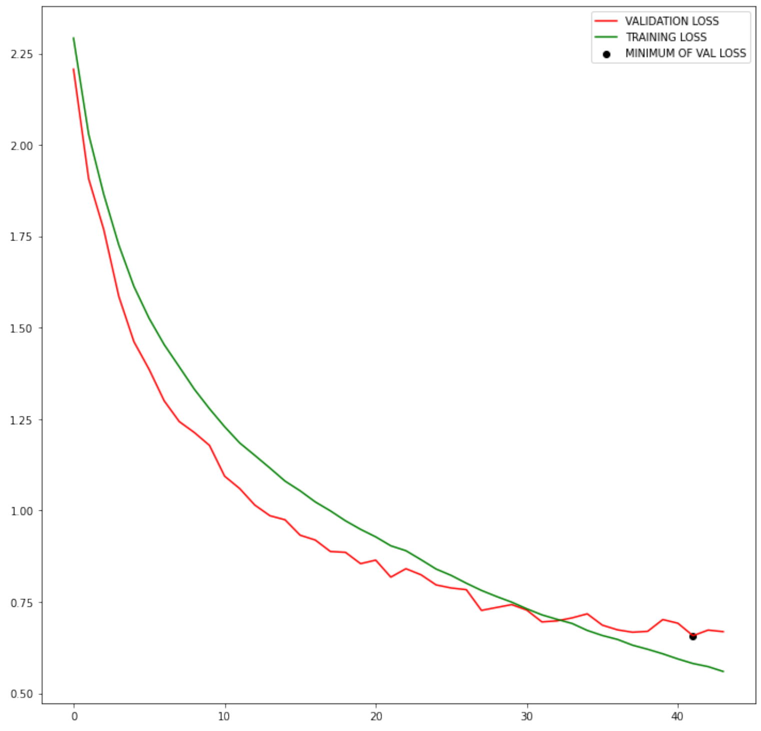

Figure 10: Validation and Training Loss with changed values of Dropout and 45 epochs

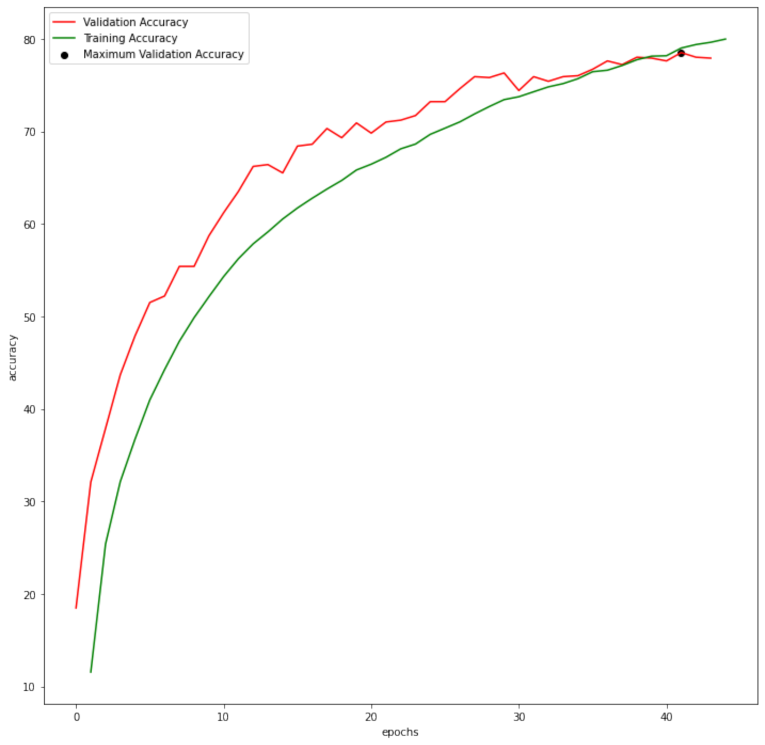

Figure 11: Validation and Training Accuracy with changed values of Dropout with 45 epochs

7.

With the test set I got a 76.27 percent of accuracy.

Questions 1.4

1. What is the Softmax layer and why is it the last layer of the network?

The Softmax layer is necessary if we want a probabilistic output because output will be a value between 0 and 1. As a consequence, it is useful to get the softmax layer at end of NN because we want to measure how much the probability of a class is.

2. What does the momentum parameter of SGD do? Why is it a good idea to set that hyper-parameter to a non-zero value?

Momentum is a value that is added to the Gradient Descent to find more quickly the local minimum and to speed up the learning process. Moreover, SGD with the momentum method remember delta of weights at each iteration.
It is a good idea to set this hyper-parameter to non zero value because if it was zero the model would remain the same without it.

3. What is the difference between a convolutional and a fully-connected layer in terms of how parameters are used? Explain why it is a good idea for processing images.

The difference between a convolutional and a FF layer is that the convolutional network implements ”weights sharing” while in FF we get nodes connect to all input node. Indeed, convolutional layer is better for images because uses less weights and furthermore it considers the spatiality of the input.

4. Here we used 2D convolution for images. Name two types of data for which you can use 1D convolution.

Two examples of types of data used with 1D convolution could be values of sound (the value of the sound wave during the time) and for text recognizing.

5. What does dropout do?

Dropout is a technique to zeros some input values randomly with a Bernoulli distribution probability set depending of the type of the network. This method could improves the performance of the model because forces the neurons to focus only on a subset of inputs. This is a regularization technique (that try to avoid overfitting for example).