Kungliga Tekniska Högskolan

# DD2424 Deep Learning in Data Science
# Assignment 2 Bonus

Quentin LEMAIRE

April 16, 2018

# 1 Optimizations

## 1.1 Data shuffling and record of the best epoch

The first optimization I did is to shuffle the dataset before each epoch. I don't think that it is a good idea for generalization that the network is seeing the images in the same order at every epochs. I also increased the number of epochs to 30 and keep into record the best epoch with the validation set.

The performances of the network increased a bit and the accuracy went up to 50.86 instead of the 47.23 I had at the end of the mandatory part.

## 1.2 He initialization

Then, I decided to put a better initialization of the parameters with the He initialization. I got a slightly improvement of the loss and the accuracy went up to 51.11.

## 1.3 More nodes and parameters adjustment

In order to get more information from the dataset, I put more nodes in the hidden layer. I also adjusted the parameters to put more regularization in order to not overfitting the dataset too much. My accuracy went up to 0.525 with 100 nodes.

One of problem I faced after and before the doubling of the number of node was that after a few epochs, the validation loss was starting to increase. The problem is caused by the fact that I chose my parameters according to a random search for 15 epochs. Therefore, I had a big overfitting and the learning rate was too high so increasing the number of epochs was useless.

To overcome this, I decreased the learning rate to 0.01 and increased the regularization to 10e-4. The loss curve was really better and I could take advantage of the 30 epochs. I got a final loss of 1.36 and an accuracy of 0.5343.

## 1.4 Data augmentation

The next optimization I put is the augmentation of the dataset. I doubled the dataset by horizontally flip every image of the dataset. Therefore, instead of a dataset of 49 000 images and used a dataset of 98 000. This optimization had a big impact over the validation loss, it went down to 1.29643801 and my accuracy went up to 0.5615 after 30 epochs.

# 2 New activation function: Leaky RELU

At this point the optimizations, I wanted to see if a different activation function could improve the network before going for a final fine search of the parameters.

With the data augmentation I did, one epoch is quite long (around 21 seconds), so I decided to try leaky RELU which still requires low computational power.

First, I tried the version I found on the slides with a parameter of 0.1 and I got a higher loss and a bigger accuracy on the test set (loss: 1.30524782, accuracy: 0.5587). The results are not so far from the previous one but it is still a bit less.

Then I decided to try a different parameter 0.01 which seems more commonly used. I got a loss of 1.29489779 and a final accuracy of 0.5625. It is both an improvement for the loss and for the accuracy compare to the simple RELU.

Therefore, I decided to keep the leaky RELU for the rest of the assignment.

## 3 Optimizations

### 3.1 Weight decay and fine search of the parameters

At the end of the last run, my validation loss was still going down so doing more epochs can provide better results. But before that I tried to find better parameters because the one I used because were not decided with a fine search.

Before this, I tried a different weight decay method. Instead of adding a 0.9 factor, I tried to divide eta by 10 every 10 epochs. I also increased the number of hidden nodes to 200 in order to get more information from the images. Therefore, I also increased the regularization of the network to get a better generalization.

Finally, I did a fine search of eta and lambda in order to have better parameters. My final parameters are eta = 0.015 and lambda = 0.00035 and I got an accuracy of 0.578.