Deep Learning - Homework Assignment 1

Lenart Rupnik, 63220472

1 Introduction

The main task of homework 1 was to familiarize ourselves with basic concept of neural networks and create a simple network without using any libraries.

2 Experiments

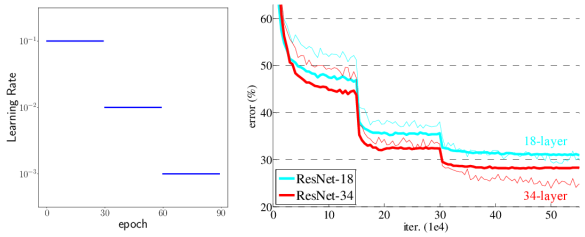
In the first task, we had to simulate both forward propagation and backpropagation by hand. For calculations, we used a simple neural network with one hidden layout, sigmoid activation functions in hidden layouts and quadratic loss function on last layout. To keep this report compact, calculations are shown in appendices.

Our next task was to manually implement a neural network without using any special libraries. To expedite our task, the base architecture of a network was already given by the assistant. After that we had to implement foward propagation and backpropagation and some additional improvements like regularization, learning rate decay, etc. To train and test our model, we used CIFAR-10 dataset which contains 60000 32 x 32 colour images categorized in 10 classes with 6000 images per class. Our goal was to achieve accuracy of at least 48 %. Results of different experiments are shown in Table 1.

To achieve the best results we could get with our model, we had to take into account many parameters and their behaviour. At first, we tried to implement all proposed improvements: L2 regularization, Adam optimizer and learning rate decay.

The main purpose of L2 regularization or Ridge Regression is to combat overfitting by forcing weights to be small while not making them exactly 0. In a way, we let less significant features have some influence over the final prediction. In practice, this is implemented by giving the loss function an additional part that consist of sum of squares of all feature weights multiplied by a constant called lambda. To implement this, we had to change our $backpropagation$ and $update\_network$ functions. Until we managed to set our lambda to some reasonable value, our results actually worsened. During the training we got the best results when setting lambda to 0.05. We also plotted training loss in consideration to validation loss and found out that after regularization, validation loss behaved more similar to training loss, meaning we stopped overfitting at early stage.

Our next improvement was implementing Adam optimizer. The main idea of Adam is to combine two gradient descent methodologies. One being momentum where we remember in which direction we were going in a previous interaction resulting in going over local minimums and root-mean-square propagation. When we implemented this method, we found out two things. First, our learning became much more time-consuming and second, that we need to change previously set learning rate. After some testing we found out that when using Adam we need to set leaning rate at least a tenth lower than when using SGD so around 0,001.



(a) Learning rate decay strategy (b) Figure taken from He et al. (2016)

Figure 1: Learning rate decay representation.

In the end, we tried to improve our model by implementing learning rate decay. The whole point of this improvement is to slowly reduce learning rate as training progresses through epochs. If we assume that during training we are closing to our gradient minimum, we know that it would be better to take smaller steps when we are near so that we don't overshoot with our correction. In the beginning learning rate decay didn't actually produce better results. After a while, we found out that we need to set decay parameter to some small value, otherwise the model stopped learning after 15 to 20 epochs. Alternatively, if we set our learning rate decay higher, we also have to reduce the number of epoch and increase our learning rate. That way we assume the model starts with big steps towards the result and then relatively soon starts to reduce corrections steps.

Table 1: Parameters used in different experiments.

	Exp. 1	Exp. 2	Exp. 3	Exp. 4
Layers	[512, 256, 128, 10]	[512, 256, 128, 10]	[512, 256, 128, 10]	[512, 256, 128, 10]
Optimizer	Adam	SGD	Adam	Adam
Learning Rate	0.0005	0,05	0.0005	0.0005
L2 reg.	Yes	Yes	No	Yes
LR Schedule	Yes	Yes	Yes	No
Epochs	30	30	30	30
Cls. Acc.	53.2%	52.87%	51.92%	52.94 %

After implementing all the above-mentioned improvements and successfully achieving minimal accuracy, we saved our network parameters and started comparing our network with and without different improvements. At first, we compared classification accuracy with Adam optimizer and with simple SGD. In both tests we used the same parameters with one exception - learning rate. Both results are seen in Table 1 annotated with Experiment 1 and Experiment 2. We can see that results with Adam optimizer were slightly better. Perhaps if we wanted to additionally increase performance with Adam, we should take more epochs. In next comparison, we trained a model with and without L2 regularization. A run without L2 regularization is labelled as Experiment 3. Again, we see a slight decrease in performance if we leave out the regularization. This makes sense, since regulation should improve our predictions.

At last we tested the performance of the model without learning rate schedule. Results are shown under Experiment 4. Even though the best of our presented models performed better with learning rate decay, this wasn't the case in all tests and we spent a long time founding a reasonable ratio between learning rate, learning rate decay factor and number of epochs.

3 Conclusion

Based on our tests, we can say that Adam optimizer works better than classical SGD. We also figured out that including L2 regularization gives better results. As for learning rate decay, we found out that it is highly depended on other parameters, so it's hard to find optimal solution which would greatly increase the performance. During testing, we also confirmed that choosing the right parameters for our neural network is quite hard since all parameters are intertwined and depend on one another. All the above mention improvements gave us slightly better performance but the biggest increase of our classification was when we increased the size and number of hidden layers. Our created model was more of a testing prototype, so it wasn't really time efficient and wouldn't be very useful in real life application. Still it gave us valuable insight into basic concepts of neural networks.