Course: DD2424 - Assignment 3
Optional for Bonus Points

Álvaro Orgaz Expósito

April 30, 2019

1 Optimize the performance of the network

I will study what is the best possible performance achievable by a k-layer fully connected network on CIFAR 10. There are some tricks/avenues I will explore to help bump up performance, and I will compare the performance results of them with the default parametrization of the assignment which, with *Batch Normalization* and 2 hidden layers of 50 nodes each one, achieves 1.4513 loss and 52.86% accuracy in the validation set. In the test data, it achieves a 53.64% accuracy. This corresponds to the parametrization case: batch size 100, 2 cycles, η_{min} 1e-5, η_{max} 1e-1, step size 5*45000/100=2250, λ 0.005. Note that it is equivalent to 20 epochs equal to 2 cycles $\cdot$ 2 $\cdot$ step size 2250 / (45000 images / batch size 100).

1.1 Do a more exhaustive random search for regularization

Firstly, I will do a more exhaustive random search to find good values for the amount of regularization using the previous coarse search in the assignment for defining the range of candidate values. Then, the list of lambdas used is: [0.001, 0.0035, 0.006, 0.0085, 0.011, 0.0135].

In figure 1 we can find the output of the fine search for λ regularization parameter. Here there is the performance of each λ value for the default parametrization case and the conclusion is that within the range of values searched, we find a region of optimal values for the validation accuracy and also for the loss since the shape of the curve of the points is not linear. Thus, choosing the optimal $\lambda = 0.0085$, we can see in figure 2 that the test accuracy achieved by this optimized 3-layer network with *Batch Normalization* is 53.98% which is not a big improvement with respect to the previous test accuracy with default $\lambda = 0.005$.

1.2 Explore more hidden nodes

For doing a more thorough search to find a good network architecture, I will train several configuration or architectures which are plotted in figure 3. Looking at this plot I can conclude that, at least with the given parametrization, making the network deeper does not improve the performance. As we can see, the loss curves do not converge to 0 and the best accuracies in validation and training sets are achieved by the networks with a lower number of layers. However, it seems that the 10-layer network improves exponentially in the last updates and increasing the number of training updates could let it learn deeper patterns in the data. It is important to note that I set each layer with 50 hidden nodes and it would be interesting to check also glass shape architectures, but the training using standard CPU it is very time-consuming.

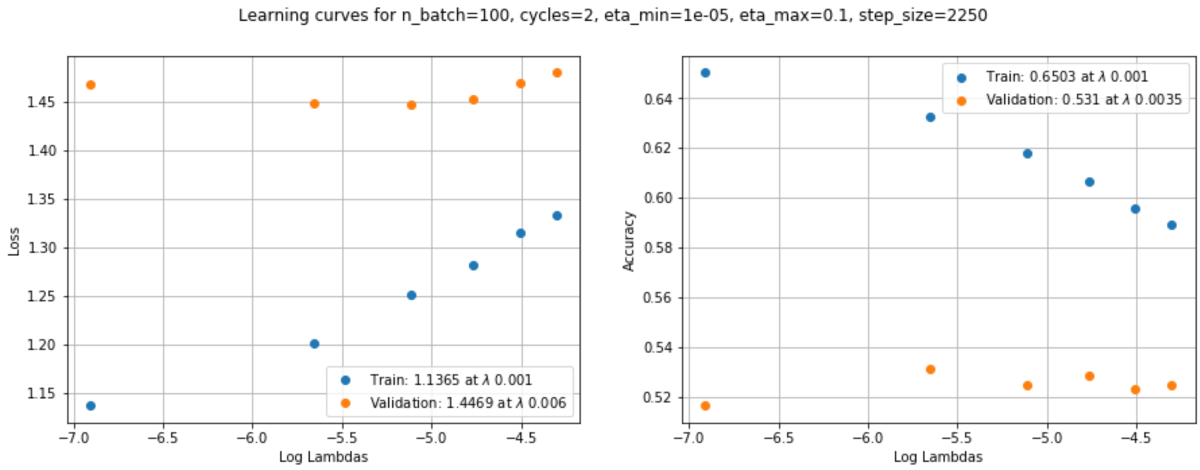

Figure 1: Fine search for λ regularization parameter.

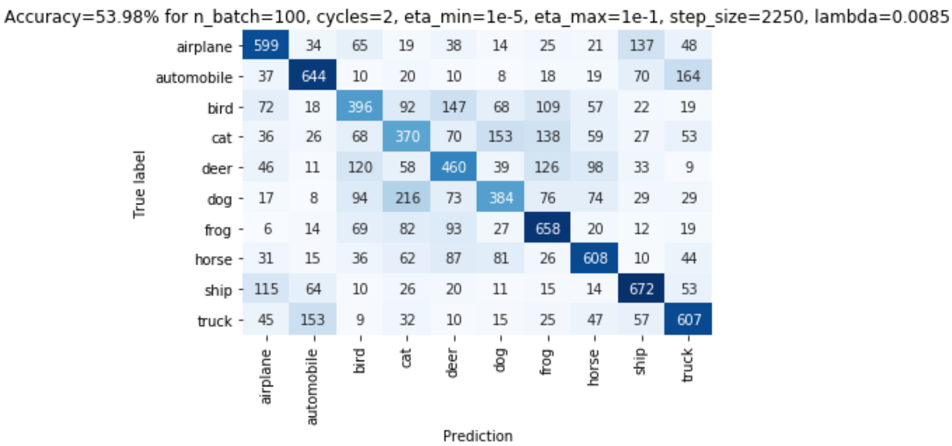

Figure 2: Test accuracy and confusion matrix for the default parametrization with *Batch Normalization* and optimal regularization.

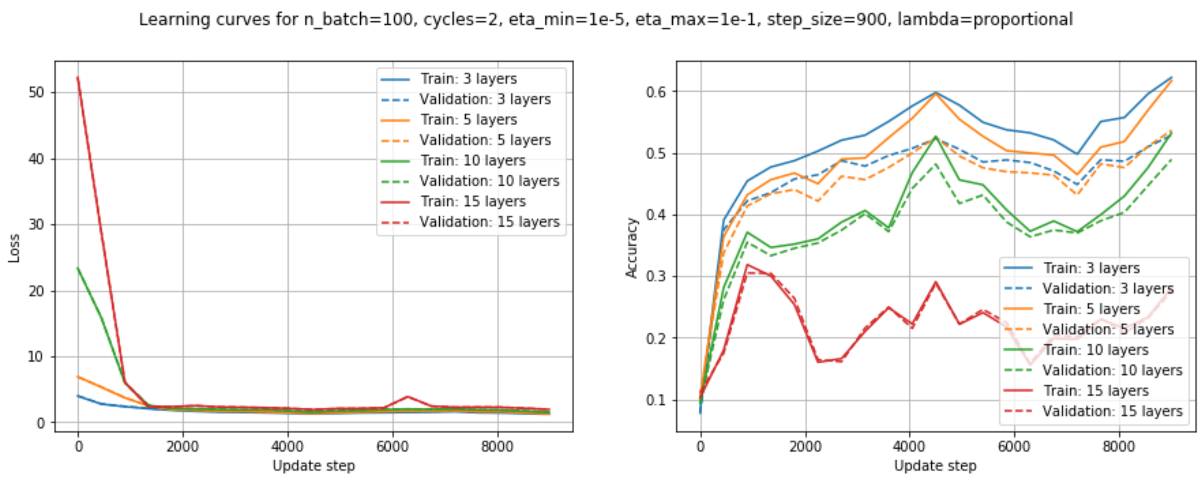

Figure 3: Learning curves for the default parametrization training by architectures.

1.3 Apply dropout

I will try with the regularization technique dropout in the training. Note that each training sample in the mini-batch has its own random dropout mask which means that the same activations (hidden nodes) in the hidden layer will be deactivated (turned to 0) for that sample, but since in this implementation the data is shuffled after each epoch, the masks are also changed by epochs. For selecting which nodes deactivate I will use random uniform values between 0 and 1 and if the value is above the given dropout probability of keeping the value then the hidden node is deactivated.

After applying this technique with a dropout rate of 0.75, my performance results are not improved as we can see in the learning curves in figure 4, and neither the test accuracy which is in figure 5. However, at least the results are not worsted which means that the implementation of the dropout is not worsening the network stability.

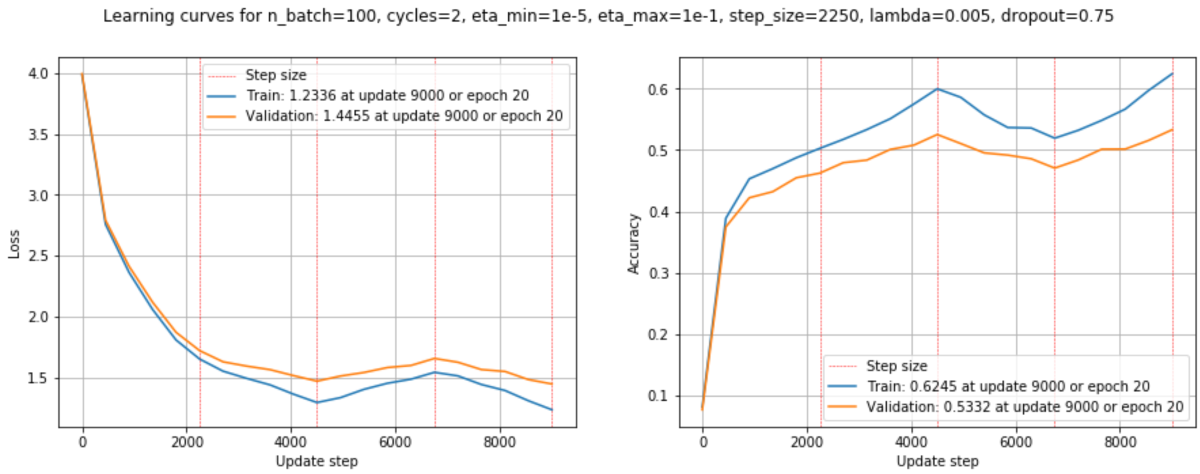

Figure 4: Learning curves for the default parametrization training with dropout 0.75.

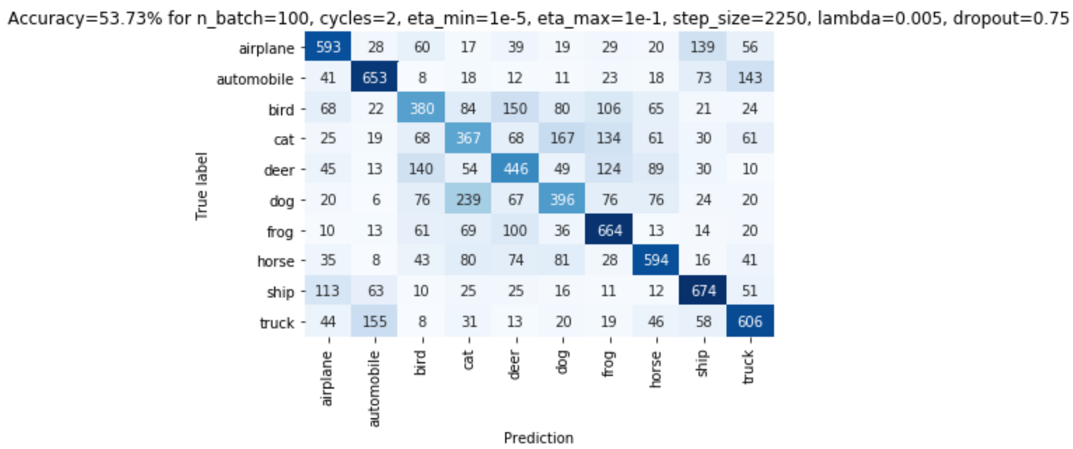

Figure 5: Test accuracy and confusion matrix for the default parametrization with *Batch Normalization* and dropout 0.75.

1.4 Data augmentation

I will try with the data augmentation technique during the training phase, concretely by applying small random geometric jitter to each original image in the mini-batch before doing the forward and backward pass.

After applying this technique with a noise level of 0.1 (basically the standard deviation of the normal distribution used for creating random noise), the performance results look like the default version without data augmentation or a noisy version of images at each batch. We can see it in the learning curves in figure 6 and 7.

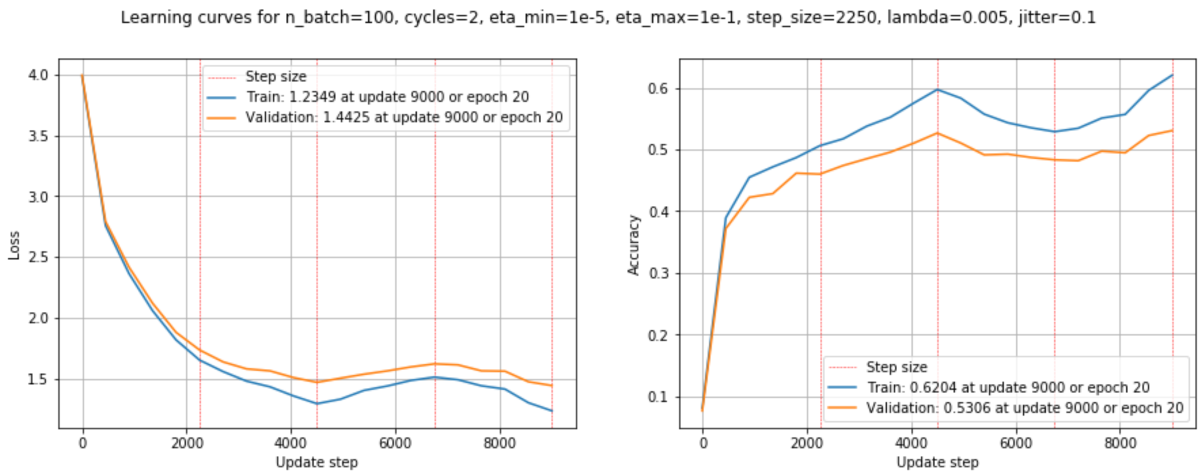

Figure 6: Learning curves for the default parametrization training with data augmentation with jitter noise level 0.1.

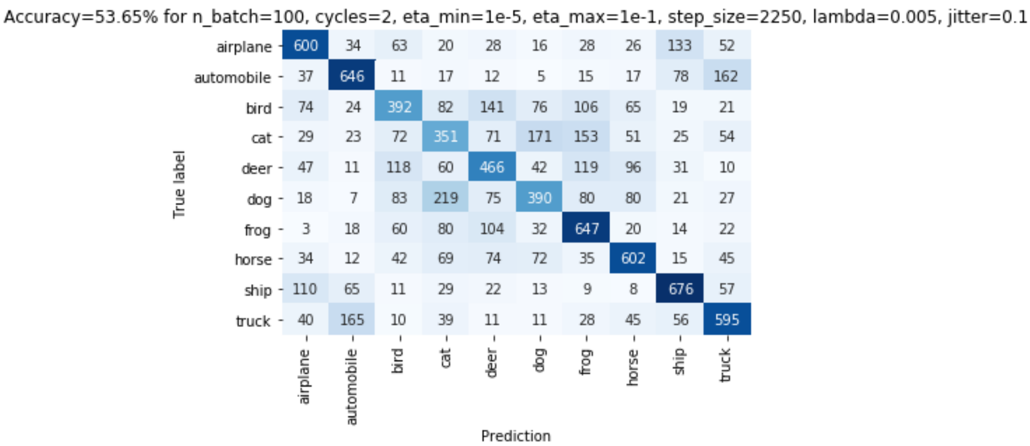

Figure 7: Test accuracy and confusion matrix for the default parametrization with *Batch Normalization* and data augmentation with jitter noise level 0.1.