Deep Learning in Practice - TP1

Alain Riou Robin San Roman
Ecole Normale Supérieure de Cachan
{alain.riou,robin.san_roman}@ens-paris-saclay.fr

This document summarizes the results and main findings of the experiments done in the exercise 1 of practical session 1 of Deep Learning In Practice course.

1. Impact of the architecture of the model

It is impossible to achieve the classification task with a single layer neural network since the data is not linearly separable. It seems in this situation that a 3-layer network with 15-20 hidden dimension is enough. ReLU activation works way better than sigmoid function. With more parameters (layers or number of neurons) it would work but it would take longer and since the data distribution is simple it is not useful. On top of that a huge number of parameters makes training harder and might lead to overfitting.

2. Impact of the optimizer

Batch size first makes the training faster ($\approx$ 6x faster between 10 and 100 batch size). Very small batch size seems to lead to training inconsistency.

Increasing the learning rate improve a lot the training time in the first place but then the network struggle to go bellow a certain error. The training is not as consistent since the loss is not always decreasing. It would be an idea to decrease the learning rate at regular epochs interval or every time the loss is increasing compared to the previous epoch. It seems that increasing the learning rate only works if the batch size is sufficient otherwise it doesn't work.

Since the learning rate scales the gradient, its optimal value depends on the architecture of the value of those gradients, and thus on the architecture. Our experiments reveal that the behavior described above (inconsistent training with a too high learning rate, local minimum found with a too low one)

Increasing the duration of the training is a safe way to improve performance if the learning rate is small enough otherwise it will only lead to oscillations around the local minimum.

Adam optimizer is working much better than SGD in almost all our experiments. Even though SGD is faster to update weight at every epoch Adam perform such a better loss drop per epoch which makes it way more efficient anyway. RMSprop has poor results compared to Adam even though with a few more epochs it performs also 100% accuracy.

3. Impact of the loss function

Using BCEWithlogit loss makes the training quicker. It is hard to measure the improvement since the 100% accuracy is reached very quickly. With this loss, 100% accuracy is reached with a 10 epochs training.

Conclusion

These experiments illustrate the influence of the different hyperparameters on a deep learning classification task. They reveal some of the classical undesirable situations that can happen while training a neural network and indicate which hyperparameters should be tuned in order to prevent them.

However, the classification problem we studied during this practical session is "too easy" for a deep neural network, and even sub-optimal sets of hyperparameters achieve 100% accuracy within a few epochs. It is therefore hard to anticipate very precisely what could be the effect of some hyperparameters with a more exotic problem.