Deep Learning Report - Lab 3

Chirag Prakash Tubakad(cpt1u21@soton.ac.uk)

I. Task 1 - Exploring Optimisation of Analytic Functions

A. Rastrigin Function

The rastrigin function is used to test the performance of optimisation algorithms. Finding the minimum of this function is challenging due to its large search space and a large number of local minima. The defualt value of 'A' equal to 10 in the Rastrigin function produces a very bumpy loss surface. The general formula of the function is given by:

$$f(x) = An + \sum_{i=1}^{n}[x_i^2 - Acos(2\pi x_i)]$$

The rastrigin function of 2 independent variables with an 'A' value equal to 1 is depicted in figure 1.

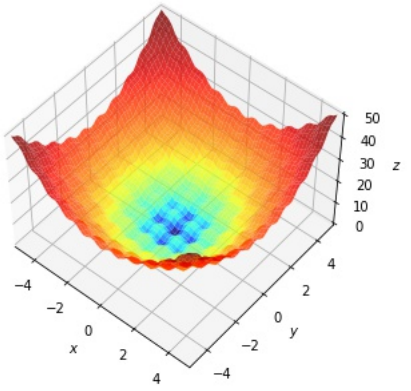

Fig. 1: Rastrigin function of 2 variables in 3D

B. Implementation of optimisers

Four different optimisers with different parameters were implemented. They are:

1) SGD (lr = 0.01)
2) SGD with momentum (lr = 0.01 , momentum = 0.9)
3) AdaGrad (lr = 0.01)
4) Adam (lr = 0.01)

All four optimisers were run with [5,5] as the starting point and for a fixed number of 100 iterations. The loss plot for each of these optimisers in shown in figure 2.

From the loss plot, it is clear that SGD with momentum works best to optimise the rastrigin function. It also reaches the minima in far fewer iterations than the other optimisers. Adagrad performs the worst in comparison to the other optimisers. It gets stuck at a local minima and fails to further optimise the function.

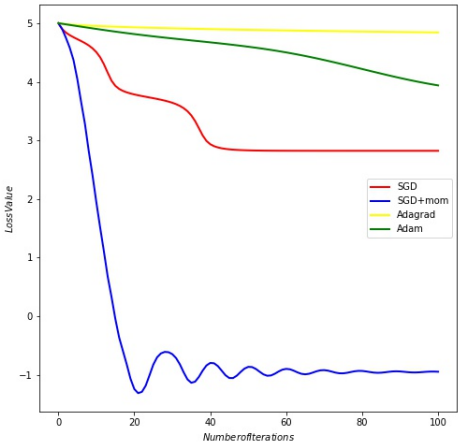

Fig. 2: Loss curves of the different optimisers

II. Task 2 - Optimisation of SVM on real data

For this task, two classes from the iris dataset were chosen and the class labels were changed to +1 and -1. The data was split in to training and validation set and the values were normalised. SGD with a learning rate of 0.01 and Adam with a learning rate of 0.01 were used to train soft margin linear SVM's with a weight decay of 0.0001 on the training data. The training batch size was set to 25 and the optimisers were run for a fixed number of 100 iterations. The weights were randomly initialised on every run and the scores for both the optimisers were averaged over 500 runs.

SVM with SGD obtained a validation score of 0.92 while SVM with Adam achieved a validation score of 0.86. On running the optimisers multiple times, adam showed a little variation in the scores obtained on individual runs whereas the scores obtained by SGD with momentum on successive runs was almost consistent. Even in the scenario of random initializations on successive runs, there was nothing counter intuitive that was predominantly observed. SGD with momentum consistently outperformed adam in almost every different parametric scenario.