ECE-60146: Homework 3

Bilal Ahmed

February 6, 2023

1 Introduction

In this homework, the objective is to become familiar with the processes involved in loading images and making them available/ready to be fed to some neural network. We look at the transforms that can be applied to images to augment the available data and make our model more robust to variations in viewing angles, illumination and contrast levels, size, scale, and rotations of objects etc. In addition, customizing our dataset class and using pytorch dataloader to gain performance during data loading is also covered.

2 Description of SGD+ and Adam

2.1 SGD with Momentum (SGD+)

Stochastic Gradient Descent with momentum (SGD+) is a variant of stochastic gradient descent (SGD) where the step size is determined not only by the current gradient but by an exponentially decaying moving average of gradient.

$$v_{t+1} = \mu * v_t + g_{t+1}$$
$$p_{t+1} = p_t - lr * v_{t+1}$$

Here μ is the momentum coefficient, it controls the weight of the momentum (history of gradients) w.r.t. to newly computed gradient. So in effect past history of gradient also hold weights in making the step at present time. When we are in a region of steep surface, instead of taking small steps, momentum allows us to take bigger steps with confidence. This helps achieving faster convergence and also avoid being misled by noisy gradient. Also, with momentum getting stuck in local minima is less likely.

2.2 Adam

Adam is from the family of optimization algorithms with adaptive learning rate and it also has momentum. So in addition to benefits of momentum that we just discussed, it adjust learning rate for every parameter based on the second moment estimate. This allows it work efficiently in the situations like sparse gradients. If gradient of loss w.r.t. to some parameter is mostly zero, Adam adjust learning rate to take bigger steps whenever it encounters non-zero gradients. Simply stating, it provides bigger learning rates for small gradients and smaller learning rates for bigger gradients.

$$m_t = \beta_1 * m_{t-1} + (1 - \beta_1) * g_t$$
$$v_t = \beta_2 * v_{t-1} + (1 - \beta_2) * g_t^2$$

Here m_t is the first moment estimate and v_t is the second moment estimate, with $\beta 1$ and $\beta 2$ respective parameters. These moments are initialized to be zero, meaning that the estimates are biased towards zero, this is especially true for early iterations. So to get unbiased estimates, they are corrected as follows:

$$\hat{m_t} = \frac{m_t}{(1 - \beta_1^t)}$$
$$\hat{v_t} = \frac{v_t}{(1 - \beta_2^t)}$$

So the parameter update equation becomes:

$$p_t = p_{t-1} - \gamma \frac{\hat{m_t}}{\sqrt{\hat{v_t} + \epsilon}}$$

3 Plots

3.1 One-neuron classifier

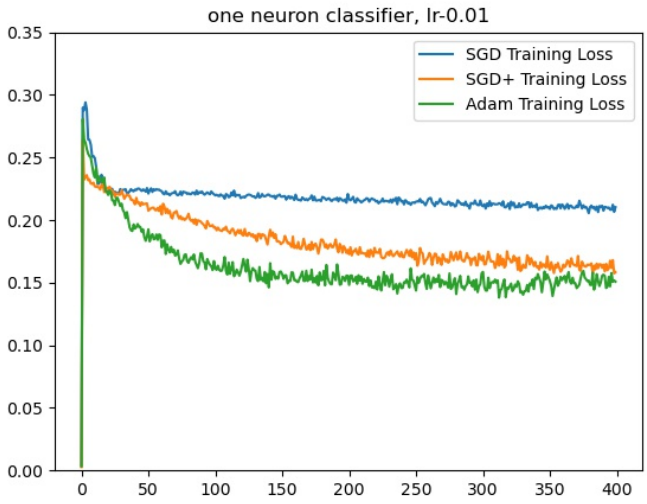

Figure 1: Training loss vs iterations: lr=0.01

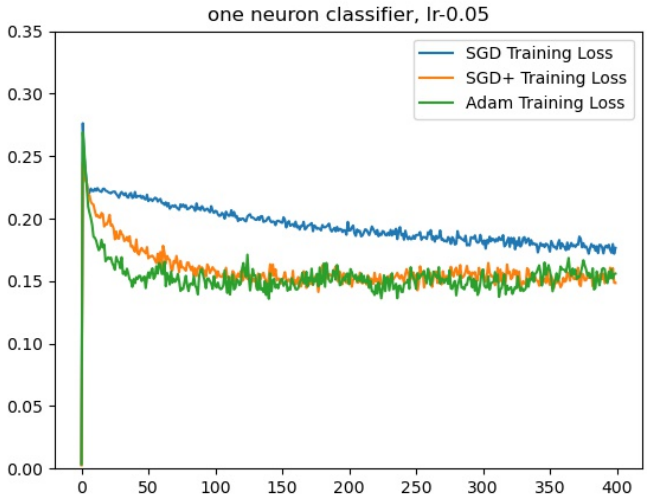

Figure 2: Training loss vs iterations: lr=0.05

3.2 Multi-neuron classifier

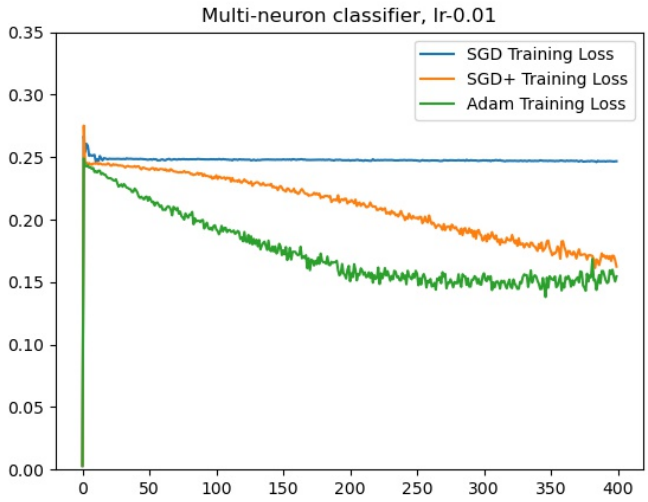

Figure 3: Training loss vs iterations: lr=0.01

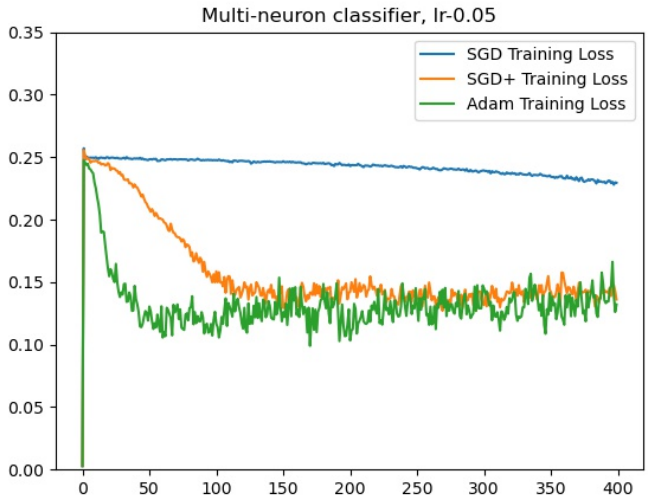

Figure 4: Training loss vs iterations: lr=0.05

4 Performance Comparison

In Figure 1. it looks like SGD is stuck in some local minima, its loss is not more decreasing. Whereas SGD+ keeps on decreasing and is approaching 0.16. Adam, on the other hand is fastest to converge. Similar observations are there in Figure 2. where learning rate is higher than first case so all of them are converging towards the global minima but at different rates. There are similar observations for multi-neuron case(in the figures 3 4). Here, SGD is stuck in local minima whereas SGD+ and Adam are converging towards global minima.

In summary, momentum helps avoid local minima and converges faster, as evident from the plots, than vanilla SGD. Adam converges fastest of all because it has both momentum and adaptive learning rate, which is especially helpful for very high dimensional parameter space. An important thing to note here is that for any given optimizer we still have to fine tune the learning rate. As the maximum step size at any given time is bounded by the learning rate, so a careful selection of learning rate, may be based on some intuition about loss surface or dimensions, is still important for all these choices of optimizers.