Deep Learning CEE690.06 – Assignment 2

Aaron Williams

Due Date: 9-28-18

Problem 1:

**a) Using the fundamental theorem of calculus, and the fact that the gradient is smooth with Lipschitz constant 𝐿, prove the form of the Lipschitz gradient upper bound on the function from the lectures.**

represents a function where

So that

Our function is monotone as it is descending, so for .

Set

Such that

A gradient is Lipschitz continuous with parameter

Cauchy Swarchz implies

Which is the monotonicity of so which is convex.

Therefore, we can say is a quadratic upper bound on

**b) Using the form of the upper bound for stochastic gradient descent given in the literature, under what regimes will the following step sizes be optimal:**

Our step size is determined from an input step size and the equation

is a determination of variance such that

is a term that references our gradient that gets smaller as our function approaches a global optimum.

is the maximum allowable step size that reduces to an approximated minimum. As an imput, the equation reduces to the following.

is optimal for a function where is low so that reduces to zero, particularly when reduces slowly. is also optimal if a quick, low iteration solution is required for a problem.

is half the previous step size.

As shown above, a step size of results in a smaller increment. This step size is optimal if more precision is required than the results from but a somewhat low iteration count is still desired. appropriately should be more than 3\* where further optimization is desired.

produces a much smaller step size and reduces the impact of the variance term to a very low value.

In the case that variance is a very large value and reduces rapidly at the optimum, is the correct size. is also optimal when precision is required and iteration count isn’t a major issue.

Problem 2b:

1. **How did each algorithm qualitatively work?**

Each model was submitted to the same conditions to directly compare their performance. This could potentially ignore the improvements of changing iteration count, layer sizes, and batch count which per step size and model could produce differing loss characteristics. In terms of time, most of the models performed similarly at around 2 minutes and 30 seconds per test (as I ran my tests on my laptop overnight). I tested a wide range of possible step sizes and momentum terms for each algorithm and considered the log scale as well as any default values that could exist.

For SGD step sizes, I tried a range of values between 0.99 and 0.01 centered at 0.5. The value step size 0.99 performed the best with performance decreasing as the step size was decreased. 50,000 iterations also took slightly longer (~30 seconds) then a step size of 0.99.

For SGD with momentum, I tried a range of both step and momentum sizes. I tried a shorter range of values but ranged them between 0.99 and 0.01 centered at 0.5. A high step size and momentum meant that the algorithm potentially didn’t converge. A low step size and momentum generally meant a poor validation by the time the iteration count was expended (higher iterations could lead to better performance). My best performance was near the center of these ranges. These tests were consistently at 2 minutes 30 seconds except for non-converging tests at high step and momentum sizes

For the Adam Optimizer I used a range centered roughly around 0.001, the default step size. The default performed alright, but I achieved higher accuracy with the value 0.01, potentially due to increment performance. Generally, the Adam optimizer took around 2 minutes 40 seconds across 50,000 iterations.

1. **How hard was it to tune your step sizes and settings?**

Quite difficult as I didn’t have a good starting inclination of step sizes, particularly for SGD with momentum. I also think to really optimize performance with a 2-layer neural network I’d need a more advanced algorithm that determined loss on the training and test set thus determining the correct number of iterations for a given algorithm. I feel like you could continue to optimize the layer size, iteration count, batch count, and the algorithm settings to ad nauseum, and you must consider if validation performance or time are more important and to what extent. I ran my tests under limited conditions with that in mind.

1. **How did each algorithm perform in validation performance estimates?**

Baseline Estimates for Validation Accuracy using Gradient Descent at step size 0.5 were estimated to be at ~0.95 with slightly higher results ~0.951 for larger iteration counts (100,000 vs 50,000).

The validation performance of simple SGD was the worst of the 3 models. It’s possible that a higher value for step size would have produced better results, but not likely that it would have been better than later models. However, varying the step size produced better results than the baseline.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Step Size | 0.99 | 0.9 | 0.75 | 0.6 | 0.5 | 0.4 | 0.1 | 0.01 |
| Validation Accuracy | 0.9527 | 0.9514 | 0.9508 | 0.9510 | 0.9495 | 0.9489 | 0.9367 | 0.8837 |

The validation accuracy of SGD with momentum was better than SGD but worse than Adam Optimizer. There seems to be a possible sweet spot derived from a function of both inputs.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Momentum\Step Size | 0.99 | 0.9 | 0.5 | 0.1 | 0.01 |
| 0.99 | 0.0980 | 0.0980 | 0.0980 | 0.6772 | 0.9524 |
| 0.9 | 0.0980 | 0.6790 | 0.9638 | 0.9541 | 0.9392 |
| 0.5 | 0.9555 | 0.9541 | 0.9522 | 0.9437 | 0.9059 |
| 0.1 | 0.9521 | 0.9521 | 0.9482 | 0.9379 | 0.8857 |
| 0.01 | 0.9528 | 0.9497 | 0.9489 | 0.9377 | 0.8810 |

Adam Optimizer produces the best results with validation accuracies as high as 0.97 over the step sizes I tested. Here are the estimates found from averages of 3 trials. Adam Optimizer also seems like it could be easy to maximize validation accuracy by changing the step size due to sharper drop offs in performance.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Step Size | 0.99 | 0.9 | 0.1 | 0.01 | 0.001 | 1e-4 | 1e-5 | 1e-6 |
| Validation Accuracy | 0.0980 | 0.0980 | 0.8663 | 0.9688 | 0.9599 | 0.9383 | 0.8193 | 0.2607 |

Problem 3:

**(a) How many hours did this assignment take you? (There is NO correct answer here, this is**

**just an information gathering exercise)**

First problem took about 4-5 hours. Second problem didn’t take much in the way of active work or thinking about the problem, maybe 2 hours and then an hour for the writeup. I ran my scripts overnight on my laptop and throughout the day when I was doing orthogonal descent on my layer sizes and iteration count. Overall, I’d say 8 hours.

**(b) Verify that you adhered to the Duke Community Standard in this assignment**

**(**[**https://studentaffairs.duke.edu/conduct/about-us/duke-community-standard**](https://studentaffairs.duke.edu/conduct/about-us/duke-community-standard)**).**

I adhered to the Duke Community Standard in the completion of this assignment.