Lab 8: Training on the linear dataset

5/9/2023

100 Possible Points

Attempt 1

5/4/2023
Next Up: Review Feedback

Attempt 1 Score: N/A

Add Comment

Unlimited Attempts Allowed

Details

Assignment

In this week's lab, you will write a client training loop to train and evaluate the accuracy a network on the linear dataset.

Incorporate your instructor's feedback

If you receive feedback from the instructor on the previous week's lab in a timely manner, incorporate that feedback into your derivations and implementation for this week's lab.

Copy the base client.py file

Copy the file into your working directory for this lab:

```
cp /data/cs3450/mlp_from_scratch/week6/client.py .
```

If you reuse code from previous labs, **be sure** you do not call `backward()` or any other torch autograd features anywhere in the code you reuse!

Implement the training loop

Write a client that sets up a network using your library, creating and

connecting the nodes to each other and telling your library what order the nodes should be

for the main training loop.

Implement stochastic gradient descent with batches. Add a loop to **client.py** to perform as many epochs over the training data as seems reasonable to you.

Train and test on a simple 2d dataset

Test your algorithm with the simple linear data produced by **create_linear_training_data.** You may find it helpful to test it with the data created by the **create_folded_training_data** as well. Two or three hidden nodes should be sufficient for these smaller datasets, which will also help with debugging.

In-lab checkoff

Demonstrate a working stochastic gradient descent loop for your library. The loop should train, weights should change, but the error doesn't need to go down yet.

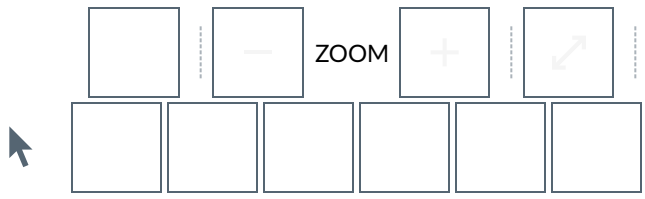
Submissions

In a comment at the end of the client.py file, copy-paste the output of your program's MSE and `M @ W` matrix (`@` is the matrix-multiplication operator.) The MSE should be near zero, and the `M @ W` matrix should be near the rotation matrix described in Lab 3.

Submit your client.py, along with all the .py files on which it depends and latest (and still passing) test_....py files. Submit the files individually to Canvas, NOT in a zip.

File Name	Size	
test_linear-2.py	1.39 KB	✓

File Name	Size	
test_relu-2.py	1.01 KB	✓
layers-2.py	15 KB	✓
network-2.py	3.24 KB	✓
test_network-2.py	2.09 KB	✓
test_mseloss-2.py	1.17 KB	✓
test_sum-2.py	1.43 KB	✓
client.py	10.7 KB	✓
test_softmax-2.py	1.46 KB	✓
test_regu...tion-2.py	1.18 KB	✓
test_layer-3.py	1.43 KB	✓
test_input-2.py	1.07 KB	✓



```
from unittest import TestCase
import layers
import numpy as np
import torch
import unittest


class TestLinear(TestCase):
    """
    Please note: I (Dr. Yoder) may have assumed different p
than you use.
    TODO: Update these tests to work with YOUR definitions
variables.
    """
    def setUp(self):
        self.h = layers.Input((2,2))
        self.M = layers.Input((2,2))
        self.b = layers.Bias((2,1)) # NOTE: constraint, can
biases!

        self.h.set(torch.eye(2, dtype=torch.float32))
        self.M.set(torch.tensor([[1,-1],[1,1]]).float())
        self.b.set(torch.tensor([[0],[1]]).float())

        self.linear = layers.Linear(self.M, self.b, self.h)

    def test_forward(self):
        self.linear.forward()

        np.testing.assert_allclose(self.linear.output.numpy
[2,2]]))

    def test_backward(self):
```

Previous (https://msoe.instructure.com/courses/13814/modules/items/564697)

Try Again

Next (https://msoe.instructure.com/courses/13814/modules/items/566340)