CS587: Assignment 2

Linear Regression & Computational Graphs in PyTorch

Issued: Tuesday 02/04/2024
Deadline: Sunday 14/04/2024, 23:59

Description

The goal of this assignment is to make you familiar with computational graphs and the information flow (forward and backward operations). You will work with the PyTorch autograd package.

Part 1: Linear Regression

In this part, you are asked to define a linear regression model using PyTorch tensors and operations, using only PyTorch tensors and the autograd package (not Modules).

Specifically, you need to follow these steps:

1. Your training set will consist of 4 pairs (x_i, y_i), where $x = [1, 2, 3, 4]$ and $y = [0, -1, -2, -3]$. Define the inputs and outputs as tensors with 'float32' format.
2. Define the weight W and bias b as tensors with 'float32' format. Initialize the weight and the bias with the values 0.3 and -0.3 respectively.
3. Define the model operation as: $\hat{y} = W * x + b$
4. Define a quadratic loss between model output and expected value (MSE):

$$Loss(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

5. Train your model for 100 iterations using SGD with learning rate 0.01.
6. Visualize the behavior of the training loss and the resulting line $\hat{y}$. Include the figures in your report and comment on the performance of your model.

Questions

Answer briefly the following questions in your report:

1. In PyTorch, the computational graph is dynamic. Can you explain what this means in terms of how the graph is constructed and utilized?
2. How does a dynamic computational graph compare to static graphs? What are the advantages and disadvantages of each approach?
3. Why did we use *w.grad.zero_()* and *b.grad.zero_()* in our code? Are they necessary for proper program execution?
4. Why did we use *torch.no_grad* in our code?

Part 2: Computational graphs and custom gradients

In this part, you will work with the PyTorch autograd package. It provides classes and functions implementing automatic differentiation of commonly used arbitrary scalar valued functions, but also allows us to define custom autograd functions using the *torch.autograd.Function* class. We will use this mechanism to define our own functions and explore the computational graphs.

A) Theory

Describe briefly the concepts of **backpropagation**, **local gradients**, and **upstream gradient** in your report.

B) Custom Functions

1. One-variable function

Let

$$y(x) = \frac{\sin(x)}{x}$$

with derivative:

$$\frac{dy}{dx} = \frac{x\cos(x) - \sin(x)}{x^2}$$

1. Design the computational graph for this function and include it in your report.
2. Compute the derivative for the backpropagation process by hand and include the computations in your report.
3. Implement the forward and the corresponding backward operation using *torch.autograd.Function*.
4. Evaluate for $x = 10$, and print the results.

2. Two-variable function

Let $f(x, y) = ax^2 + bxy + cy^2$

1. Design the computational graph for this function and include it in your report.
2. Include the partial derivatives $\frac{\partial f}{\partial x}$, $\frac{\partial f}{\partial y}$ as derived by the backpropagation in your report.
3. Implement the forward and the corresponding backward operation using *torch.autograd.Function*.
4. Evaluate for $x = 2$ and $y = 3$, and print the results.

3. Second-order derivative

For the function $f(x) = \ln(1 + e^x)$

1. Design the computational graph for this function and include it in your report.
2. Define the forward pass and the operations that compute the 1st derivative $f'(x)$.
3. Define the operations that compute the 2nd derivative $f''(x)$ and include the calculations in your report.
4. Evaluate for $x = 10$, and print the results.

Notes

Submission info

- Create a .pdf or .doc file to report the resulting scores, images/figures and any other comments or description of your work you may need to submit. Do not forget to include your name, login, ID in the report. Save this file in your working folder.
- Use zip/rar/gz to compress your working folder and rename it to cs587_mylogin_assignment2.xxx in order to submit a single file.
- Upload your submission as a SINGLE zip/rar/gz file on e-learn.
- You can upload your submission as many times as you need, keeping the same filename.
- Late submissions will be accepted **within two days** of the original deadline. However, **a penalty of 25% per day** will be applied to the final grade for each late day. Assignments submitted more than two days late will not be accepted.

Academic integrity

The assignment is individual. You may discuss with each other in general terms, but the code and the report should be written individually. If you use existing material, be sure to cite the sources in your report. The use of AI language models to produce the report or your implementation is strictly prohibited, and submissions will be verified with an AI detection tool. You may be asked to take an additional short oral examination.

Troubleshooting & Contact

If you encounter any errors or bugs in the provided code, you can report them by sending an email to either kaziales@csd.uoc.gr or the course mailing list hy578-list@csd.uoc.gr. Please note that the course mailing list is the preferred channel for general questions about the assignment, as it allows everyone in the class to benefit from the answer.

For any questions of a more personal nature that are not relevant to the entire class, you can reach out to the teaching assistant directly at his email address kaziales@csd.uoc.gr.