Assignment 02

Image and Video Processing with Deep Learning

PUBLISHED: WEDNESDAY, 5 FEBRUARY 2025

SUBMISSION DEADLINE:
Sunday 9 FEB 2025 16:59:59 (5 PM)

Instructions

The following instructions should be followed strictly. Please read carefully as there are changes from previous assignment even though the basic problem is the same.

Your assignment is going to be checked and graded using a python script which depends on these naming and format conventions to be able to run your program successfully. So any deviation from the following format conventions will result in 0 points.

File Name format:

asgn2_student_full_name.py
where student_full_name should be the name as it appears in the document *student_name_format.txt* that I have posted on google classroom. Please copy paste your name from that file.

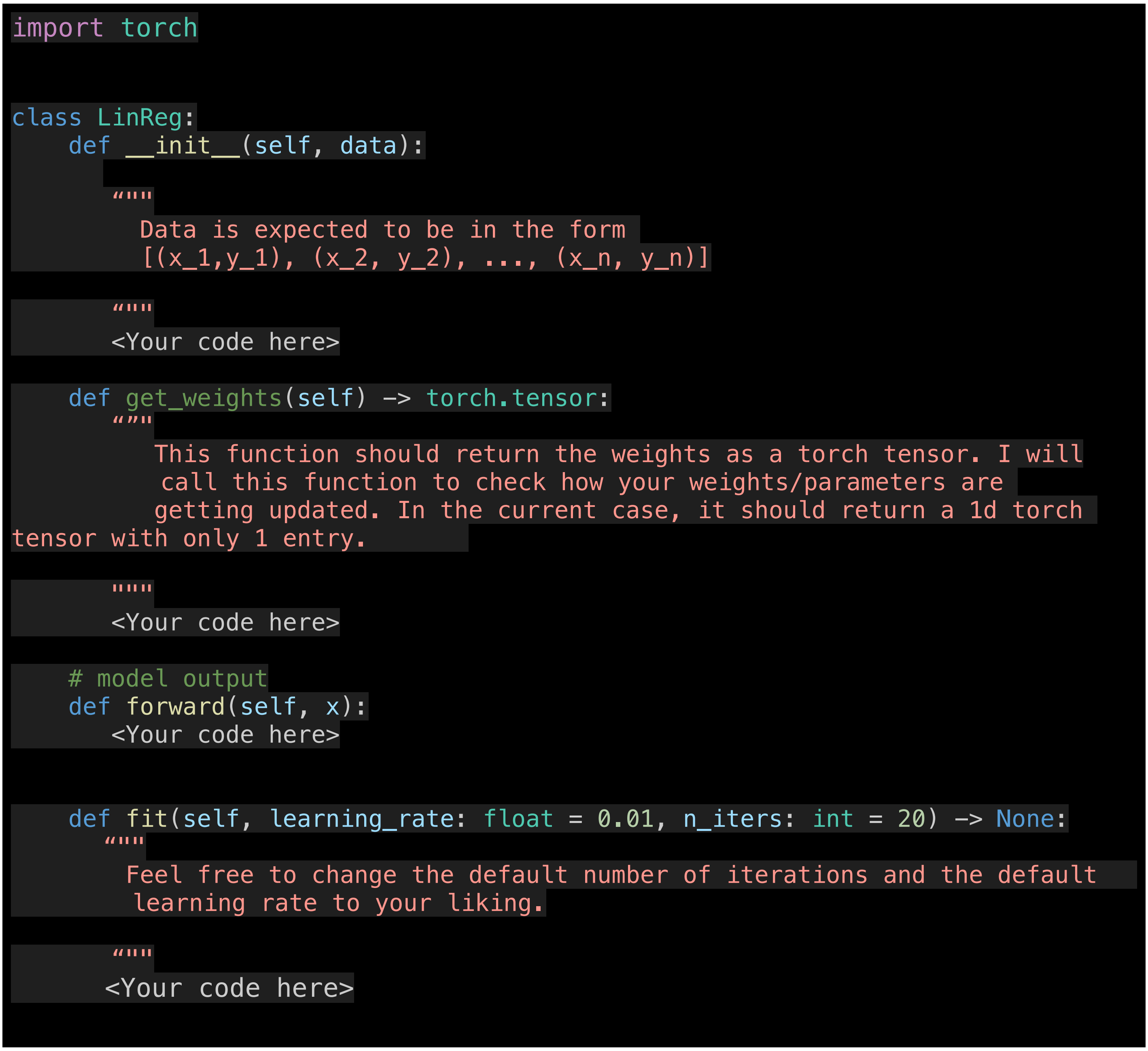
For example, if my name appears as
"chaitanya_Guttikar" in that file, then the filename should read
asgn2_chaitanya_Guttikar.py

Program format:

Your program should have the following class implemented. Make sure all the methods mentioned below are implemented with exact same names (including capital/small case) and their input and output types are as mentioned here.

We are building a learning pipeline using pytorch to fit the model
y = wx

to the data
[(x_1, y_1) , … , (x_n, y_n)]



```
import torch


class LinReg:
    def __init__(self, data):
       “""
         Data is expected to be in the form
         [(x_1,y_1), (x_2, y_2), ..., (x_n, y_n)]

       “""
       <Your code here>

    def get_weights(self) -> torch.tensor:
       “”"
          This function should return the weights as a torch tensor. I will
           call this function to check how your weights/parameters are
          getting updated. In the current case, it should return a 1d torch
tensor with only 1 entry.

       """
       <Your code here>

    # model output
    def forward(self, x):
       <Your code here>


    def fit(self, learning_rate: float = 0.01, n_iters: int = 20) -> None:
      “""
        Feel free to change the default number of iterations and the default
         learning rate to your liking.

       “""
      <Your code here>
```

DO NOT USE any packages other than torch and the built in python packages.

Use **torch.nn.MSELoss** as the loss function.

Use **torch.optim.SGD** as the optimizer.

If needed, please read PyTorch documentation to understand how to use these things correctly. (Hint: You will need to understand these - loss.backward(), optimizer.step(), optimizer.zero_grad())

What my program will do:

1. It will have some code that generates *my_data* which will be in the form mentioned above ([(x_1, y_1), …, (x_n, y_n)]. Since this is synthetically generated data, I will already know the best ‘w’ for the model y = wx. Let’s call it *w_best*
2. It will import the LinReg class from your file and instantiate a model *model = LinReg(my_data)*
3. It will run *model.fit()* with your default *learning_rate* and *n_iters* as well as some other values.
4. It will periodically run model.get_weights() to see how the weights are changing.
5. It will check the final value of weights with the best w that I am expecting by doing *weight = model.get_weights().item(),* and comparing *w_best* with *weight* - Note that *get_weights* method returns a torch tensor of weights. For a torch tensor t containing only 1 value, t.item() returns that value.
6. It will also run model.forward to check implementation and it will check that you are using torch.optim.SGD as the optimizer and torch.nn.MSELoss as the loss function.

If there are any questions, please post them on the google classroom.