University of Illinois Fall 2024

ECE 364 Programming Methods for Machine Learning
Homework 1 finished by Jianchong Chen

Due on Thursday, September 12, 2024, 11:59pm on Gradescope

1. **[18 points]** PyTorch Storage

   (a) (3 points) Explain the difference between a PyTorch Tensor and a PyTorch View. Focus on memory consumption and creation of views and tensors. Explain your answers.

   Your answer:pytorch tensor allocates a new memory address for the data, while pytorch view just references to the existing memory address without allocating a new address. For the creation part, pytorch tensor has torch.tensor(), or torch.ones(), torch.zeros(), which all allocate a new memory space. And pytorch view has torch.Tensor.view(), which can create a new shape sharing the same underlying storage as the original, and the view() needs existing tensor to create.

   (b) (8 points) Suppose you have a dataset of 1 million RGB color images. Assume all images have a size of 256 × 256 pixels and we use 1 byte to store the intensity of each color channel. a) Provide the code to allocate a 4-dimensional PyTorch tensor of all zeros which stores this data in the most efficient way. b) How much memory does the data in this tensor consume? c) How much memory would this data consume if we had chosen to cast the images into a torch.double format?

   Your answer:(a):torch.zeros(1000000,3,256,256) (b):As for the memory, 1000000*3*256*256=196608000000 bytes which is about 183 GB (c): if we use double which is 8 bytes, the result will be 1000000*3*256*256*8= 1.572864e+12 bytes which is 1464.84375 GB

   (c) (3 points) Name three PyTorch operations which are guaranteed to create new tensors and three operations which are guaranteed to create views.

Your answer: For creating new tensor: torch.tensor(), torch.ones(), torch.zeros() For creating views: torch.Tensor.view(), torch.permute(), torch.t(),torch.Tensor.diagonal()

(d) (2 points) What is the printed output of the following code snippet? Explain your answer and reasoning carefully.

```python
import torch
a = torch.arange(100).view(10,10)
b = a[[2, 5, 6], ::3]
print(a.storage().data_ptr()==b.storage().data_ptr())
```

Your answer: The output is False. Because the b is new tensor copy of a instead of a view of a, a and b can't share the same address.

(e) (2 points) What is the printed output of the following code snippet? Explain your answer and reasoning carefully.

```python
import torch
a = torch.arange(100).view(10,10)
b = a.flatten()
print(a.storage().data_ptr()==b.storage().data_ptr())
```

Your answer:The output is True. Because a can be flatten, which lead to creating b as a view of a.

2. **[12 points]** Array Slicing and Truth Arrays

(a) (4 points) Suppose we would like to extract the columns at index 1 and 3 from tensor `a` and place them in tensor `b`. What should the arguments `x` and `y` be below to accomplish this array slice? You may state `x` and `y` or re-write the last line of code below.

```python
import torch
a = torch.rand(10, 10)
b = a[:, 1:4] # I have re-written the code
```

Your answer: I have already re-written the code

(b) (4 points) Suppose we would like to extract every value in tensor `a` that is greater than the mean value of `a` and place these values in tensor `b`. What should the argument `x` be below to accomplish this task? You may state `x` or re-write the last line of code below.

```
import torch
a = torch.rand(100)
b = a[a>torch.mean(a)] # finished
```

Your answer: I have re-written the code

(c) (4 points) Consider the below function where `n` is some given positive integer. What will be the resulting data stored in tensor `b`? In other words, how are tensors `a` and `b` related?

```
import torch
def mystery_function(n):
    a = torch.rand(1000, 1000)
    b = a[::n, ::n]
    return b
```

Your answer: the b stores every element of every nth row and nth column of tensor a

3. **[10 points]** PyTorch Functions

(a) (5 points) Recall from lecture that we defined the element-wise rectified linear unit (ReLU) and sigmoid functions as element-wise "activation functions" (more on them later in the course). Another popular element-wise operation is the soft thresholding function

$$\mathcal{S}_\tau(x) = \text{sgn}(x)\{0, |x| - \tau\},$$

where τ is the soft threshold value and the $\text{sgn}(x)$ function gives -1 for negative numbers, 1 for positive numbers, and 0 for 0. Below is the code to implement $\mathcal{S}_\tau(x)$.

```
    import torch
    def soft_thresholding(x, tau):
        x_abs = torch.abs(x)
        y = torch.zeros_like(x_abs)
        x_sign = torch.sign(x)
        result = torch.where(x_abs-tau==0,y,x_sign*(x_abs-tau))
    return result
```

Complete the above implementation of $\mathcal{S}_\tau(x)$ using the `torch.where` function by replacing the ??? line.

Your answer: I have modified it above

(b) (5 points) Consider the following alternative version of the soft thresholding function where we have two parameters $\tau_1 \leq 0$ and $\tau_2 \geq 0$.

```
    import torch
    def soft_thresholding_new(x, tau_1, tau_2):
        new_result = (x < tau_1)*(x-tau_1)+(x>tau_2)*(x-tau_2)
    return new_result
```

How is this new soft thresholding function different from the one in part (a)? Make sure to explain what the `tau_1` and `tau_2` parameters represent for this new function.

Your answer: if x > τ_2, the output is x - τ_2 ; if x < τ_1, the output is x - τ_1 ; if x $\geq \tau_1$ and x $\leq \tau_2$, the output is 0; The range of x is different when the output is 0. And the upper threshold is decided by τ_2 and lower threshold is τ_1 which is more flexible

4. **[10 points]** Derivatives Practice

(a) (6 points) Compute the derivative of the following function $f(x)$ ($x \in [-1, 1]$). Simplify as much as possible for full points.

$$f(x) = \log\left(\left(\cos(x^2 - 1)\right)^2\right)$$

Your answer: $-4 * x * tan(x^2 - 1)$

(b) (4 points) Evaluate the value of the derivative at $x = 0$, $x = 1$ and $x = -1$. What can we say about each of these three points?

Your answer: when $x = 0$ $f(0)' = 0$; when $x = 1$ $f(1)' = 0$; when $x = -1$ $f(-1)' = 0$; They are all critical points. When x=0 it is local min, when x=1,-1 they are global max.