COMP4901Y Homework 1

Question 1. Einstein Notation in PyTorch (15 points).

Implement the following operations according to PyTorch Einstein notations:

- 4D tensor element-wise multiplication given input tensors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{d_1 \times d_2 \times d_3 \times d_4}$, output should be $\mathbf{c} \in \mathbb{R}^{d_1 \times d_2 \times d_3 \times d_4}$;
- Batch transposed matrix multiplication given input tensor $\mathbf{a} \in \mathbb{R}^{d_1 \times d_2 \times d_3 \times d_4}$, $\mathbf{b} \in \mathbb{R}^{d_5 \times d_4}$, output should be $\mathbf{c} \in \mathbb{R}^{d_1 \times d_2 \times d_3 \times d_5}$;
- Aggregate a 4D tensor through the 3rd dimension given input tensor $\mathbf{a} \in \mathbb{R}^{d_1 \times d_2 \times d_3 \times d_4}$, output should be $\mathbf{a'} \in \mathbb{R}^{d_1 \times d_2 \times d_4}$.

Submission. This part should be submitted in a python file named **question1.py** by fill the missing part in the corresponding sample code.

Question 2. Training a two-layer MLP ***without*** PyTorch Autograd (25 points).

Implement an SGD training algorithm to train a simple 2-layer MLP regression model based on the L2 loss defined below ***without*** using pytorch autograd.

- Input: $\mathbf{X} \in \mathbb{R}^{N \times D}, \mathbf{Y} \in \mathbb{R}^{N \times 1}$
- Model parameters: $\mathbf{W}_1 \in \mathbb{R}^{D \times H}, \mathbf{W}_2 \in \mathbb{R}^{H \times 1}$
- Forward computation (each iteration samples a single data point $\left(\mathbf{x}_t \in \mathbb{R}^{1 \times D}, y_t \in \mathbb{R}\right)$):
  - $\mathbf{a} = \mathbf{x}_t \mathbf{W}_1$
  - $\mathbf{a}' = \mathrm{relu}\left(\mathbf{a}\right)$
  - $y' = \mathbf{A'} W_2$
  - $l = \left(y' - y_t\right)^2$
- Backward computation:
  - Construct the backward computation for the linear layer as we discussed in class;
  - The derivative of the relu function can be computed by:

$$\frac{df_{\mathrm{relu}}(a)}{da} = \begin{cases} 1 & a \geq 0 \\ 0 & a < 0 \end{cases}$$

- SGD update:
  - Use a fixed learning rate of 0.03 (tunable hyper-parameter), no momentum;
  - Implement sampling with replacement;
- Run the 10^5 iteration of the SGD iteration. Record the loss for every 100 iterations. Plot the results as we introduced in class.

Submission. This question should be submitted by:

- A Python file named **question2.py** fills in the missing part in the corresponding sample code.
- A pdf file named **question2_result.pdf** to visualize the convergence result.

Question 3. Transfer Learning by PyTorch Autograd (20 points).

PyTorch provides a set of pre-trained models; for example, you can load a resnet18 model by the following statement:

```
resnet18_model = torchvision.models.resnet18(weights='IMAGENET1K_V1')
```

We want to use the intermediate convolutional layers defined in this resnet18_model but change the fc layer to fit the FashionMNIST dataset, as we demoed in our class. Here are some additional notes:

- You need to change the output dimensions of the last fc layer;
- You need to change the input dimensions of the first input layer;
- You need to tune the SGD hyper-parameters in the following sets and use the optimal setting in your submitted script.
    - Batch size: $16, 64$
    - Learning rate: $0.01, 0.03$
    - Momentum : $0.0, 0.9$
- Run the 2 epochs of the training. Record the loss and plot the results as we introduced in class.

Submission. This part should be submitted with:

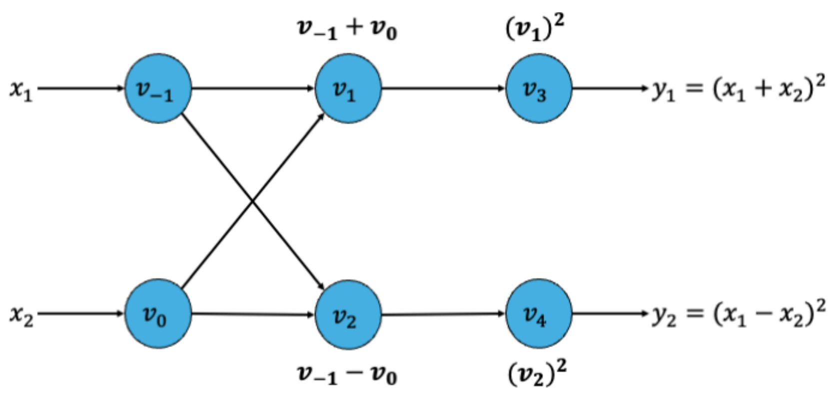
- A Python file named **question3.py** fills in the missing part in the corresponding sample code.
- A pdf file named **question3_result.pdf** of convergence result.

Question 4. Auto-Diff Example (30 points).

Given the following function $\mathbf{y} = f(\mathbf{x}) : \mathbb{R}^2 \to \mathbb{R}^2$ defined by:

$$y_1 = (x_1 + x_2)^2$$
$$y_2 = (x_1 - x_2)^2$$

One way to construct the computation graph is as follows:



Please fill the following tables to compute the Jacobin matrix at the point $x_1 = 2, x_2 = 4$:

- Table 1: forward mode to compute $\frac{\partial y_1}{\partial x_1}, \frac{\partial y_2}{\partial x_1}$:

Forward Primal	Value	Forward Tangent	Value
$v_{-1} = x_1$	2	$\dot{v}_{-1} = \dot{x}_1$	1
$v_0 = x_2$	4	$\dot{v}_0 = \dot{x}_2$	0
$v_1 = v_{-1} + v_0$		$\dot{v}_1$	
$v_2 = v_{-1} - v_0$		$\dot{v}_2$	
$v_3 = (v_1)^2$		$\dot{v}_3$	
$v_4 = (v_2)^2$		$\dot{v}_4$	
$y_1 = v_3$		$\dot{y}_1 = \dot{v}_3$	
$y_2 = v_4$		$\dot{y}_2 = \dot{v}_4$	

- Table 2: forward mode to compute $\frac{\partial y_1}{\partial x_2}, \frac{\partial y_2}{\partial x_2}$:

Forward Primal	Value	Forward Tangent	Value
$v_{-1} = x_1$	2	$\dot{v}_{-1} = \dot{x}_1$	0
$v_0 = x_2$	4	$\dot{v}_0 = \dot{x}_2$	1
$v_1 = v_{-1} + v_0$		$\dot{v}_1$	
$v_2 = v_{-1} - v_0$		$\dot{v}_2$	
$v_3 = (v_1)^2$		$\dot{v}_3$	
$v_4 = (v_2)^2$		$\dot{v}_4$	
$y_1 = v_3$		$\dot{y}_1 = \dot{v}_3$	
$y_2 = v_4$		$\dot{y}_2 = \dot{v}_4$	

- Table 3: reverse mode to compute $\frac{\partial y_1}{\partial x_1}, \frac{\partial y_1}{\partial x_2}$:

Forward Primal	Value	Reverse Adjoint	Value
$v_{-1} = x_1$	2	$\bar{x}_1 = \bar{v}_{-1}$	
$v_0 = x_2$	4	$\bar{x}_2 = \bar{v}_0$	
$v_1 = v_{-1} + v_0$		$\bar{v}_{-1}$	
		$\bar{v}_0$	
$v_2 = v_{-1} - v_0$		$\bar{v}_{-1}$	
		$\bar{v}_0$	
$v_3 = (v_1)^2$		$\bar{v}_1$	
$v_4 = (v_2)^2$		$\bar{v}_2$	
$y_1 = v_3$		$\bar{y}_1 = \bar{v}_3$	1
$y_2 = v_4$		$\bar{y}_2 = \bar{v}_4$	0

- Table 4: reverse mode to compute $\frac{\partial y_2}{\partial x_1}, \frac{\partial y_2}{\partial x_2}$:

Forward Primal	Value	Reverse Adjoint	Value
$v_{-1} = x_1$	2	$\bar{x}_1 = \bar{v}_{-1}$	
$v_0 = x_2$	4	$\bar{x}_2 = \bar{v}_0$	
$v_1 = v_{-1} + v_0$		$\bar{v}_{-1}$	
		$\bar{v}_0$	
$v_2 = v_{-1} - v_0$		$\bar{v}_{-1}$	
		$\bar{v}_0$	
$v_3 = (v_1)^2$		$\bar{v}_1$	
$v_4 = (v_2)^2$		$\bar{v}_2$	
$y_1 = v_3$		$\bar{y}_1 = \bar{v}_3$	0
$y_2 = v_4$		$\bar{y}_2 = \bar{v}_4$	1

- [Optional]: Think about whether the Jacobin matrix will change when we reformulate $\mathbf{y} = f(\mathbf{x}) : \mathbb{R}^2 \to \mathbb{R}^2$ as:

$$y_1 = x_1^2 + 2x_1x_2 + x_2^2$$
$$y_2 = x_1^2 - 2x_1x_2 + x_2^2$$

Submission. This part should be submitted with:

- A pdf file named **question4.pdf** to include Table 1, Table 2, Table 3, and Table 4.

Question 5. Gradient Computation in FC Layer (10 points).

Following the computation introduced in Slides 29-31 of Lecture 4, verify that $\frac{\partial L}{\partial \mathbf{W}} = \mathbf{X}^T \frac{\partial L}{\partial \mathbf{Y}}$.

Submission. This part should be submitted with:

- A pdf file named **question5.pdf** to include the computation.
- Latex is recommended for equation editing.

Submission Checklist.

You should submit a zip file including the following components:

- **`question1.py`**
- **`question2.py`**
- **`question2_result.pdf`**
- **`question3.py`**
- **`question3_result.pdf`**
- **`question4.pdf`**
- **`question5.pdf`**