ECS795P Deep Learning and Computer Vision, 2024

Coursework 1 Guideline: Building a Transformer from Scratch

Introduction

Aim:
The assignment is to implement a deep learning model Transformer using PyTorch for image classification. The objectives are:
(1) to obtain practical knowledge and hands-on understanding of the basic concepts in Transformer;
(2) to obtain practical experience on how to implement a basic Transformer using PyTorch.

Start: Download CW1_ECS795P.zip from the course website at:
http://www.eecs.qmul.ac.uk/~sgg/_ECS795P_/ .

Download and install PyTorch from its official website:
For Linux: https://pytorch.org/get-started/locally/#linux-installation .;
For Mac: https://pytorch.org/get-started/locally/#mac-installation ;
For Windows: https://pytorch.org/get-started/locally/#windows-installation .

How to use Google Colab: We recommend using Jupyter Hub to run your code. Alternatively, you can use Google Colab to implement the code with a time-limited free GPU source. Find more information about how to use Colab in this link:
https://colab.research.google.com/drive/16pBJQePbqkz3QFV54L4NIkOn1kwpuRrj

Tasks: three subtasks are involved:

1. Coding: to add your code blocks in the required sections, note code is in Python format and needs GPU to run, please log in to Jupyter Hub with your QM account, choose the ECS795P module and upload the *.ipynb file to run it; (40% of this CW)
2. Report: to complete the questions in the report; (30% of this CW)
3. Online assessment: to answer one question and to conduct one exercise, which is randomly selected from below. **It will be carried out during the lab demo session in WK10**; (30% of this CW)

Platform: Python + PyTorch

Basic material:
Some of the online materials for PyTorch-code may help you better complete this coursework (if you are not familiar with PyTorch, you can follow this step by step)
https://pytorch.org/tutorials/
https://github.com/yunjey/pytorch-tutorial

https://towardsdatascience.com/illustrated-guide-to-transformers-step-by-step-explanation-f74876522bc0

1. Understanding Self-Attention Mechanism

Objective: To get familiar with single-head and multi-head self-attentions.

Questions:

1. What is the concept of single-head self-attention?
2. What is the concept of multi-head self-attention?
3. What is the difference between single-head and multi-head self-attention?

Exercises:

1. Calculate the dimensions of W_q ,W_v , W_k in "Single-head Attention" section.

2. Calculate the dimensions of W_q ,W_v , W_k in "Multi-head Attention" section.

3. Show the weight of the causal attention mask in "Causal attention mask" section.

4. Show the figure of the causal attention mask in "Causal attention mask" section.

2. Understanding the structure of Transformer

Objective: To understand the implementation details of a transformer.

Questions:

1. What are the main compositional layers of a transformer?
2. What is the purpose of layer normalization in a transformer?
3. What is the difference between the training and testing stage of a transformer?
4. What is the skip connection in a transformer?
5. Why does a transformer usually use multi-head attention instead of single head one?

Exercises:

1. To show the projection weight (W^K) of **K** in the multi-head attention layer (Tip: *check the* PyTorch build-in class *nn.MultiheadAttention*).

2. To show the projection weight (W^Q) of **Q** in the multi-head attention layer

(Tip: *check the* PyTorch build-in class *nn.MultiheadAttention*).

3. To show the projection weight (W^V) of **V** in the multi-head attention layer (Tip: *check the* PyTorch build-in class *nn.MultiheadAttention*).

3. Image classification with Transformer

Objective: To perform image classification with a Transformer network and evaluate the performance.

Questions:

1. How to use a trained Transformer to perform image classification (testing stage)?
2. What are the input and the output of the Transformer?
3. What augmentations are used for training data?
4. What's the difference between the preprocess of the train set and the validation set?
5. Why augmentations for training data is necessary?

Exercises:

1. To load and preprocess the train set and validation set from the CIFAR10 dataset.
2. To transform training images to patch embedding.
3. To add a class toke to the patch embedding.
4. To feed the input image embedding into the Transformer.
5. To plot the training loss curve to show its variation with the epoch.
6. To plot the accuracy score curve to show its variation with the epoch.