# COMP4901Y Homework4

## Question 1. Memory Efficiency Estimation for Fine-Tuning Methods (45 points).

Given a model based on stacking transformer layers, where

- $N_{\text{layer}}$ is the number of layers in the transformer layer;
- $B$ is the training batch size;
- $L$ is the training sequence length;
- $D$ is the model dimension;
- $n_H$ is the number of heads;
- $H$ is the head dimension. Note that we have $D = n_H \times H$.

Each layer of computation is summarized below:

| Computation | Input | Output |
|---|---|---|
| $Q = xW^Q$ | $x \in \mathbb{R}^{B\times L\times D}$, $W^Q \in \mathbb{R}^{D\times D}$ | $Q \in \mathbb{R}^{B\times L\times D}$ |
| $K = xW^K$ | $x \in \mathbb{R}^{B\times L\times D}$, $W^K \in \mathbb{R}^{D\times D}$ | $K \in \mathbb{R}^{B\times L\times D}$ |
| $V = xW^V$ | $x \in \mathbb{R}^{B\times L\times D}$, $W^V \in \mathbb{R}^{D\times D}$ | $V \in \mathbb{R}^{B\times L\times D}$ |
| $[Q_1, Q_2 \dots, Q_{n_h}] = \text{Partion}_{-1}(Q)$ | $Q \in \mathbb{R}^{B\times L\times D}$ | $Q_i \in \mathbb{R}^{B\times L\times H}, i = 1, \dots n_h$ |
| $[K_1, K_2 \dots, K_{n_h}] = \text{Partion}_{-1}(K)$ | $K \in \mathbb{R}^{B\times L\times D}$ | $K_i \in \mathbb{R}^{B\times L\times H}, i = 1, \dots n_h$ |
| $[V_1, V_2 \dots, V_{n_h}] = \text{Partion}_{-1}(V)$ | $V \in \mathbb{R}^{B\times L\times D}$ | $V_i \in \mathbb{R}^{B\times L\times H}, i = 1, \dots n_h$ |
| $\text{Score}_{\text{i}} = \text{softmax}(\frac{Q_i K_i^T}{\sqrt{D}}), i = 1, \dots n_h$ | $Q_i, K_i \in \mathbb{R}^{B\times L\times H}$ | $\text{score}_{\text{i}} \in \mathbb{R}^{B\times L\times L}$ |
| $Z_i = \text{score}_i V_i, i = 1, \dots n_h$ | $\text{score}_{\text{i}} \in \mathbb{R}^{B\times L\times L}$, $V_i \in \mathbb{R}^{B\times L\times H}$ | $Z_i \in \mathbb{R}^{B\times L\times H}$ |
| $Z = \text{Merge}_{-1}([Z_1, Z_2 \dots, Z_{n_h}])$ | $Z_i \in \mathbb{R}^{B\times L\times H}, i = 1, \dots n_h$ | $Z \in \mathbb{R}^{B\times L\times D}$ |
| $\text{Out} = ZW^O$ | $Z \in \mathbb{R}^{B\times L\times D}$, $W^O \in \mathbb{R}^{D\times D}$ | $\text{Out} \in \mathbb{R}^{B\times L\times D}$ |
| $A = \text{Out}\, W^1$ | $\text{Out} \in \mathbb{R}^{B\times L\times D}$, $W^1 \in \mathbb{R}^{D\times 4D}$ | $A \in \mathbb{R}^{B\times L\times 4D}$ |
| $A' = \text{relu}(A)$ | $A \in \mathbb{R}^{B\times L\times 4D}$ | $A' \in \mathbb{R}^{B\times L\times 4D}$ |
| $x' = A'W^2$ | $A' \in \mathbb{R}^{B\times L\times 4D}$, $W^2 \in \mathbb{R}^{4D\times D}$ | $x' \in \mathbb{R}^{B\times L\times D}$ |

Suppose we are always using SGD optimizer without momentum (that being said, there are no optimizer states ), and all the computation is in FP16.

1.1 Suppose we are using freezing layer fine-tuning. We only update the parameters in $N_{\text{top}}$ transformer layers of the above model. How many bytes must be allocated to store the gradients w.r.t parameters in this computation? [15 points]

1.2 Suppose we are using adapter fine-tuning. Suppose the intermediate dimension in each adapter layer is $R$. We are adding two adapter layers in each transformer layer defined above. How many bytes must be allocated to store i) the additional parameters in adapter layers and ii) the gradients w.r.t these additional parameters in this computation? [15 points]

1.3 Suppose we are using LoRA fine-tuning. Suppose the intermediate dimension in each LoRA adapter layer is $R$. We only apply LoRA to query, key, and value layers in each transformer layer defined above. How many bytes must be allocated to store i) the additional parameters in LoRA layers and ii) the gradients w.r.t these additional parameters in this computation? [15 points]

[Optional]: Think about the corresponding computation load in each fine-tuning workload.

**Submission**. This part should be submitted with:

- A pdf file named **question1.pdf** to include the computation.

## Question 2. Tree Attention (40 points).

2.1 Suppose we are using tree attention to improve memory efficiency in speculative decoding. Five candidates are provided by the assistant model(s):

Sequence 1: $t_1, t_2, t_3, t_4$;

Sequence 2: $t_1, t_2, t_5, t_6$;

Sequence 3: $t_1, t_7, t_8, t_9$;

Sequence 4: $t_1, t_7, t_8, t_{10}$;

Sequence 5: $t_1, t_7, t_{11}, t_{12}, t_{13}$;

Where $t_i$ indicates a unique token, please:

- Draw the corresponding prefix tree given the above sequences; [10 points]
- Write down the corresponding tree attention mask. (Suppose the attention mask is an upper triangular matrix. ) [10 points]

2.2 Suppose we are using tree attention to improve memory efficiency for Medusa parallel decoding. The original LM head generates $t_1$, and there are three Medusa heads, and we are using the top 2 predicated tokens.

Medusa head 1 generates the guess of $[t_2, t_3]$;

Medusa head 2 generates the guess of $[t_4, t_5]$;

Medusa head 3 generates the guess of $[t_6, t_7]$,

Where $t_i$ indicates a unique token, please:

- Draw the corresponding prefix tree given the above sequences; [10 points]
- Write down the corresponding tree attention mask. (Suppose the attention mask is an upper triangular matrix. ) [10 points]

**Submission**. This part should be submitted with:

- A pdf file named **question2.pdf** to include your answer.

## Question 3. Hugging Face Inference with Speculative Decoding (15 points).

We will use the hugging face API to learn how to easily apply speculative decoding for generative inference computation. Run the implementation provided in question3.py, and report the execution time for standard decoding and speculative decoding on your own machine.

**Submission**. This part should be submitted with:

- A PDF file named **question3.pdf** that includes your benchmark results.