a) Compute the time and space complexities of applying QLoRA and LoRA to LFFN and MHA seperately.

LoRA :

LFFN time complexity for LoRA

$$LFFN(X) = XW + XA(B)^T$$

- XW : $nxd \times dxd$ is $O(nd^2)$
- XA : $nxd \times dxr$ is $O(ndr)$
- $XA(B)^T$: $nxr \times rxd$ is $O(ndr)$
- $XW + XA(B)^T$: $nxd + nxd$ is $O(nd)$

Total time complexity is $O(nd^2) + 2 \cdot O(ndr) + O(nd)$, which is $O(nd^2)$.

LFFN space complexity LoRA

The trainable parameters are

$A, B \in R^{dxr}$

This is $O(2dr)$, which is $O(d)$

since $d >> r$

MHA time complexity LoRA

For queries, keys and values we do

HW: $nxd \times dxd$ is $O(nd^2)$

HA: $nxd \times dxr$ is $O(ndr)$

$HA(B)^T$: $nxr \times rxd$ is $O(ndr)$

$HW + HA(B)^T$: $nxd + nxd$ is $O(nd)$

This total time complexity is $O(hn^2d)$

because we have h heads and $d >> r$

Note, is $X \in \mathbb{R}^{a \times b}$ then

softmax(X) is $O(a \cdot b)$

time complexity of head(H):

QK^T : $n \times d \times d \times r$ is $O(n^2 d)$

$\frac{QK^T}{\sqrt{d}}$: $n \times n$ is $O(n^2)$

$softmax\left(\frac{QK^T}{\sqrt{d}}\right)$: $n \times n$ is $O(n^2)$

$softmax\left(\frac{QK^T}{\sqrt{d}}\right) * V$: $n \times n \times n \times d$ is $O(n^2 d)$

total time complexity is

$O(h n^2 d)$ for all h heads

Last step is to compute time complexity of MHA(H), which is $O(hnd^2)$

Final time complexity:

$$O(hnd^2 + hn^2d)$$

Space complexity of MHA LoRA

have matrices $A^{\cdot,l_i}, B^{\cdot,l_i} \in \mathbb{R}^{d \times r}$

for each Q, K, V and h heads.

This is: $O(3h(2d \cdot r)) = O(hd)$,

since $d >> r$.

QLoRA:

Time complexity for LFFN:

Same steps as LoRA but need to apply a dequantization step to $W \in \mathbb{R}^{d \times d}$

$$O(nd^2) + O(d^2) = O(nd^2)$$

Space complexity for LFFN:

Same as LoRa: $O(d)$

Time complexity for MHA:

Same steps as LoRA but with dequantization steps.

dequant(Δ, W) is $O(a \cdot b)$,

given $W \in R^{a \times b}$

dequantization is done once for Q, K, V for each head h. and the final step with $W^{o,l}$.

This is $O(hd^2)$.

Using time complexity from LoRA, total complexity is

$$O(hnd^2 + hn^2d) + O(hd^2) = O(hn^2d + hnd^2)$$

Space complexity MHA QLORa :

Same as LoRa, $O(hd)$

b)

	MHA	LFFN
Memory Limit	QLoRA	QLoRA
Faster Inference	LoRA	LoRA
Both	QLoRA	LoRA

With a memory limit, QLoRA is better because dequantizing saves memory.

For faster inference, LoRA is better because dequantizing takes time.

If you want both lower memory and faster inference, it will depend. MHA has many more parameters so maybe in that

case the inference penalty you pay with QLoRA is worth it.

By similar logic, LoRA might be worth the memory penalty since the are less parameters.