An Introduction to Fine-Tuning LLMs

What’s Fine-Tuning?

- Adapting a large language model for a specific downstream task.

Benefits and Use-Cases of Fine-Tuning

- Ability to provide more examples than prompt engineering techniques like FSL.
- Reduce the number of tokens passed to the model for each input.
- Teaching model very complex instructions
- Customizing the structure or tone of responses.

How to Fine-Tune LLMs?

There are two major paradigms.

1. Full Fine-Tuning
2. Parameter Efficient Fine-Tuning

Full Fine-Tuning

It is the traditional approach in which all the parameters of the LLMs are trained.

For each downstream task a copy of the original LLM is required since full fine-tuning changes the parameters of the original model.

Full Fine-Tuning Disadvantages

1. It is a computationally very expensive process.
2. Risks over-writing existing knowledge.
3. Requires additional copy for each downstream task.

Parameter Efficient Fine-Tuning (PEFT)

The core idea of ‘PEFT’ is to fine-tune only a small subset of the LLM's parameters while keeping the majority frozen.

Advantages:

- Reduced Computational Cost
- Preservation of Knowledge
- Faster Training

PEFT Techniques

There are many PEFT techniques. Each has its own pros and cons.

- LoRA (Low Rank Adaptation)
- Prefix Tuning
- Prompt Tuning
- Adapter Layers

We will discuss only LoRA and Prefix Tuning.

Prefix Tuning

- Instead of manipulating the parameters of the LLM, Prefix-Tuning technique uses trainable prefix tokens to the beginning of every input sequence.
- During training, prefix tokens are guided to optimize the performance.
- The original LLM parameters remain frozen.

Low Rank Adaptation (LoRA)

LoRA largely freezes the original parameters of the LLM and uses low rank matrices to fine tune the LLM.

Prefix Tuning versus LoRA

LoRA

- Can be applied to a broader range of NLP tasks beyond generation. 👍
- Offers flexibility in where to inject the low-rank matrices for fine-grained control. 👍
- Offers better performance 👍
- Introduces slightly more parameters than prefix tuning. 👎
- Finding the optimal rank for matrices A and B can require more experimentation. 👎

Prefix Tuning

- Extremely parameter-efficient, often using the smallest number of additional parameters 👍
- Highly-tuned for generation tasks 👍
- Can be slightly more restrictive in terms of the range of tasks it suits best. 👎
- Tuning the length and initialization of the prefix is important. 👎