Intro to LLM Fine Tuning

Amber Liu

2023/09
# Difference Between Pre-training

<table>
<thead>
<tr>
<th>Stage</th>
<th>Pretraining</th>
<th>Supervised Fine-tuning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm</td>
<td>Language modeling &lt;br&gt; predict the next token</td>
<td></td>
</tr>
<tr>
<td>Dataset</td>
<td>Raw internet text &lt;br&gt; ~trillions of words &lt;br&gt; low-quality, large quantity</td>
<td>Carefully curated text &lt;br&gt; ~10-100K (prompt, response) &lt;br&gt; low quantity, high quality</td>
</tr>
<tr>
<td>Resource</td>
<td><strong>1000s of GPUs months of training</strong> &lt;br&gt; ex: GPT LLaMA, PaLM</td>
<td><strong>1-100 GPUs days of training</strong> &lt;br&gt; ex: Vicuna-13B</td>
</tr>
</tbody>
</table>
Pretrained Models are NOT Assistants

- Base model does not answer questions
- It only wants to complete internet documents
- Language models are not aligned with user intent

Write a poem about bread and cheese.

Write a poem about someone who died of starvation.

Write a poem about angel food cake.

Write a poem about someone who choked on a ham sandwich.

Write a poem about a hostess who makes the
When do you want Fine-Tuning?

1. Vanilla fine-tuning
   • Gain knowledge for specific downstream task

2. Prompt engineering
   • Precise control over output
   • No computing resources

3. Instruction tuning
   • Adhere LLM to human’s instructions
When do you want Fine-Tuning?

4. Retrieval Augmented Generation (RAG) LLM

5. Parameter-Efficient Fine-Tuning (PEFT)

6. Reinforcement Learning from Human Feedback (RLHF)
   - Align with human preference
Fine-tuned model is not aligned with human preference

- Human Feedback
  - offer a sense of emotional connection
  - uncover underlying messages within the conversation
  - avoid confining oneself to superficial aspects of the matter

Make machine produce sentences that sound natural to a human
Challenges

1. Memory Capacity Intensive
2. Computation Intensive
Parameter-Efficient Fine-tuning (PEFT): a class of methods that adapt LLMs by updating only a small subset of model parameters.

Figure 5: Fine-tuning an LLM for a specific downstream task. (a) illustrates vanilla fine-tuning, which requires updating the entire model, resulting in a new model for each task. In (b), PEFT instead learns a small subset of model parameters for each task with a fixed base LLM. The same base model can be re-used during inference for different tasks.
Figure 2: Parameter-efficient fine-tuning methods taxonomy. We identify three main classes of methods: **Addition-based**, **Selection-based**, and **Reparameterization-based**. Within additive methods, we distinguish two large included groups: **Adapter-like** methods and **Soft prompts**.
Addictive: Adapters

Add additional, learnable layers into a Transformer architecture. ~3%
Selective: BitFit

Only fine-tune the biases of the network. (<1%)

```python
params = (p for n, p in model.named_parameters() if "bias" in n)
optimizer = optimizer(params)
```

Fail when model size is large
Reparametrization-based: LoRa

Figure 1: Our reparametrization. We only train $A$ and $B$.

\[ h = W_0 x + \Delta W x = W_0 x + B A x \]

- Only update the low-rank matrix
- 10000x less trainable parameter
- 3x less GPU memory requirement
- Apply to any linear layer
- No inference overhead
**QLoRa**

**Figure 1:** Different finetuning methods and their memory requirements. QLoRA improves over LoRA by quantizing the transformer model to 4-bit precision and using paged optimizers to handle memory spikes.
Fine-tuning Library

1. Pytorch
2. Hugging Face - PEFT
3. Lamini
4. OpenAI Fine-tuning API
1. LoRA: LOW-RANK ADAPTATION OF LARGE LANGUAGE MODELS
2. Prefix Tuning: P-Tuning v2: Prompt Tuning Can Be Comparable to Fine-tuning Universally Across Scales and Tasks
3. Prompt Tuning: The Power of Scale for Parameter-Efficient Prompt Tuning
4. P-Tuning: GPT Understands, Too
5. Parameter-efficient transfer learning for nlp
6. Challenges and Applications of Large Language Models
7. QLORA: Efficient Finetuning of Quantized LLMs
https://build.microsoft.com/en-US/sessions/db3f4859-cd30-4445-a0cd-553c3304f8e2

https://web.stanford.edu/class/cs224n/slides/cs224n-2023-lecture11-prompting-rlhf.pdf

https://www.bilibili.com/video/BV1Tu4y1R7H5/?spm_id_from=333.788.recommend_more_video.0&vd_source=39940709d86c95c61be9bec979dfb187

https://www.youtube.com/watch?v=dA-NhCrrrVE