Deferred Continuous Batching in Resource-Efficient Large Language Model Serving

Yongjun He (yongjun.he@inf.ethz.ch)
Yao Lu
Gustavo Alonso

ETH Zürich, National University of Singapore
The Rise of AI PCs
Resource-Efficient Fine-Tuning

Previous methods: Retrain all model parameters.

Low-Rank Adaptation (LoRA)

• Freezie the pre-trained model and update a small number of (additional) parameters

\[ h = (W + \Delta W)x = Wx + BAx \]

where \( W \in \mathbb{R}^{d \times k} \), \( B \in \mathbb{R}^{d \times r} \), \( A \in \mathbb{R}^{r \times k} \), and the rank \( r \ll (d,k) \).

• Reduce memory usage during fine-tuning by 60 - 70%.

• Reduce memory usage during inference by 100 - 10000x for each new fine-tuned model.
Resource-Efficient Inference

Previous methods: Newly arrived requests have to wait for the current batch to complete.

Continuous Batching

- Newly arrived requests only need to wait for the current token to complete.
- Enable 10 – 20x throughout since the decode stage is memory-bound

<table>
<thead>
<tr>
<th>Iteration</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>…</th>
</tr>
</thead>
<tbody>
<tr>
<td>Req1</td>
<td>Prompts ETH Zurich is located in Zurich</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Req2</td>
<td>Prompts Paris has the Eiffel Tower</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Req3</td>
<td>Prompts NUS was founded in 1905</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Heterogeneous Fine-Tuning and Inference

Example
A student wants to fine-tune an LLM to help improve her thesis for a herbology class. She would choose the biggest LLM that fits in her AI PC for better results.

Challenge
All other local LLM-based applications will not work until the completion of fine-tuning.
Prior Solutions

Spatial GPU sharing
• Execute fine-tuning and inference in parallel on smaller pre-trained LLMs
• Reduce model’s capabilities

Temporal GPU sharing
• Switch from fine-tuning to inference when new requests arrive
• Leave little time for fine-tuning tasks
• Incur high context switch overhead
Deferred Continuous Batching

A new task scheduling mechanism

- Schedule at the granularity of a single fine-tuning or inference iteration
- Slightly defer inference requests without violating service level agreements
FineInfer at a Glance

Designed for concurrent parameter-efficient Fine-tuning and Inference

• Deferred Continuous Batching
• Hybrid system architecture
Hybrid System Architecture

Minimize context switch overhead

- Base model multiplexing
- Iteration-level switching

<table>
<thead>
<tr>
<th>Stage</th>
<th>DeepSpeed</th>
<th>Colossal-AI</th>
<th>FineInfer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task initialization</td>
<td>1.153 / 0.015 s</td>
<td>0.28 / 0.045 s</td>
<td>0 / 0 s</td>
</tr>
<tr>
<td>Task cleanup</td>
<td>2.330 / 1.252 s</td>
<td>3.456 / 1.376 s</td>
<td>0 / 0 s</td>
</tr>
<tr>
<td>Data movement</td>
<td>5.882 s</td>
<td>5.918 s</td>
<td>0 - 0.052 s</td>
</tr>
</tbody>
</table>

The breakdown of switching overhead with Llama2-7B workloads on an Nvidia 4090 GPU.
Hybrid System Architecture

Amortize data movement overhead for larger-than-GPU LLMs

- Heterogenous batching
GPU-Resident Performance
Llama2-7B on a 24GB Nvidia 4090 GPU

Fine-tuning throughput (samples/m) vs. Deferral bound (s)

- FineInfer
- DeepSpeed (Optimized)
- Colossal-AI (Optimized)
Larger-than-GPU Performance
Llama2-13B on a 24GB Nvidia 4090 GPU

Fine-tuning throughput (samples/m)

- FinelInfer
- DeepSpeed (Optimized)
- FinelInfer (w/ hb)
- Colossal-AI (Optimized)

Deferral bound (s)

0  20  40  60  80  100  120

0  5  10  15  20  25  30
Summary

We need to evolve systems for LLMs for the new era of the AI PC.

FineInfer = **Fine-tuning** + **Infernece**

- Deferred continuous batching improves fine-tuning throughput by slightly deferring inference requests without violating SLAs.
- Hybrid system architecture minimizes context switch and data movement overhead.

Source code: [https://github.com/llm-db/FineInfer](https://github.com/llm-db/FineInfer)

Thank you!