ExeGPT: Constraint-Aware Resource Scheduling for LLM Inference

Summary

The paper proposes two different scheduling strategies - Round-Robin Allocation: - Workload-Aware Allocation: both of these are suitable for different NLP workloads. These two strategies "decouple" the encoding and decoding process and allow them to happen more efficiently. They use probability to decide between the two scheduling strategies. There are four variables that the strategies use to determine a fast execution plan. The work is split in to batches but problems arise the batch have different amount of work loads.

LLMs works by encoding the data then decoding the result in autoregressive iterations. Tokens are transformed into <query, key, value> It then does some linear algebra. Then some other transformations are done. And the output is sent to the next layer of the nueral net. Some early work proposes a dynamic progamming like model for remembering key value pairs to do less computation.

XProfiler - measures the execution time of the batch processes. XSimulator - takes the results from the xProfiler and constructs a execution timeline. XScheduler - This looks for the best configuration parameter. XRunner - enforces the schedule during runtime.

ExeGPT tries to find a balance in between the encoding and decoding operations. RRA tries to empnasis large decoding batches. WAA has smaller decoding batches on average then the other scheduling stratigy. Increasing the batch size increase pawallelism and therefore throught put but it also increase latency because there is more data to process.

Pros

- maintains large input batch while minimizing pipeline bubbles.
- it challenges specific assumptions in the LLM world

Cons

Further Developments

Other Comments

The Trade-off section of the paper is very helpful for understanding what they accomplished.