DON'T STOP ME NOW: EMBEDDING BASED SCHEDULING FOR LLMS

Rana Shahout Eran Malach Chunwei Liu Weifan Jiang
Minlan Yu Michael Mitzenmacher

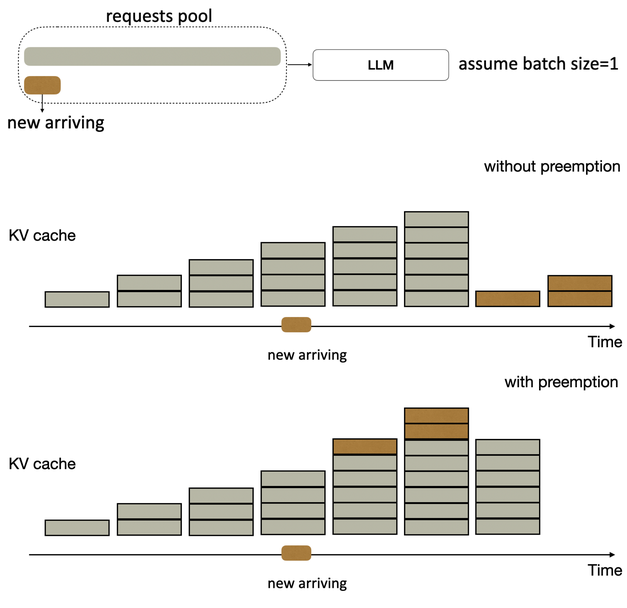
Harvard University, MIT

Fast LLM inference = smart scheduling

- But size-based scheduling (prioritizing short requests over long ones) requires knowing request sizes – a challenging task in LLM systems.
- **How can we predict request sizes accurately?**

Meet **TRAIL**! Our approach recycles LLM embeddings into a lightweight classifier to predict the remaining length for each running request. This enables efficient size-based scheduling like Shortest Remaining Processing Time (SRPT), *optimizing mean response time*.

Preemption enables dynamic scheduling by deciding whether to continue the current request or replace it with a shorter, newly arrived one, thereby reducing mean response time. SRPT is a classic preemptive policy. However, in LLMs, preemption introduces KV memory overhead—a challenge absent in traditional queueing systems.

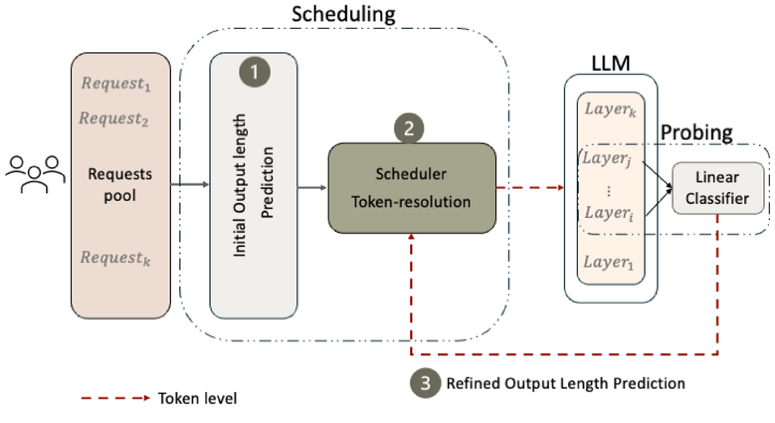


To tackle this, **TRAIL** allows preemption early in request execution when memory consumption is low but restricts preemption as requests approach completion to optimize resource utilization. Given a predicted request length r, preemption is only allowed during the first $\lfloor c \cdot r \rfloor$ iterations, for a fixed constant c.

Brief Summary

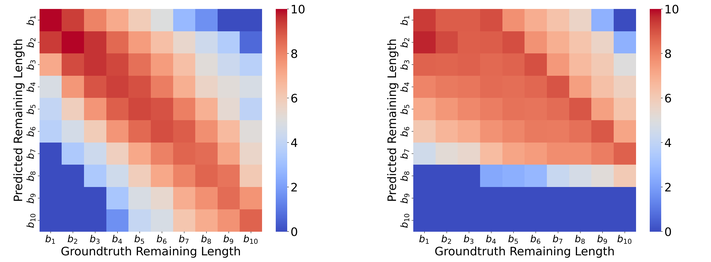
- After generating each output token, **TRAIL** recycles LLM embeddings into a lightweight classifier to predict the remaining length for each running request.
- **TRAIL** implements prediction-based SRPT variant with limited preemption designed to account for memory overhead in LLM systems.
- On the theoretical side, we derive a closed-form formula for this SRPT variant in an M/G/1 queue model, which demonstrates its potential value.

TRAIL Architecture



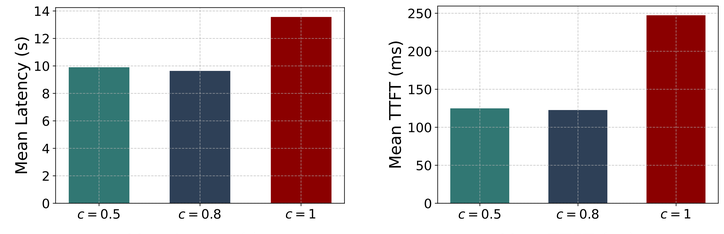
The system (1) initially orders requests using a BERT model, (2) schedules requests using a modified prediction-based SRPT with limited preemption, and (3) refines predictions during token generation using embeddings from the LLM's internal layers. At every token, steps 2 and 3 are repeated (represented as red dashed lines), which allows preemption at token-level granularity and refined predictions.

Empirical results



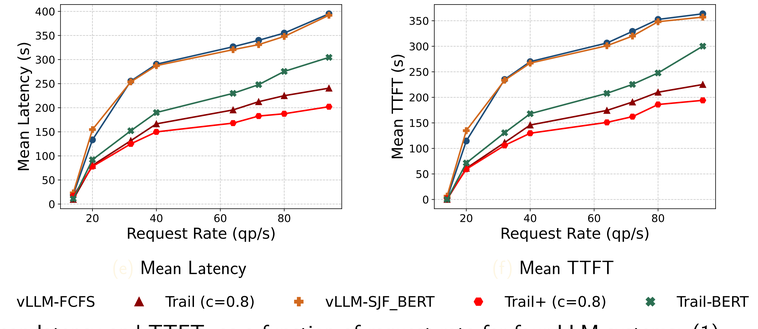
(a) embedding-based predictions (b) BERT prediction (prompt as an input)

Comparison of request size predictions: ground-truth vs. predicted length bins (log-scaled), using BERT-based predictions (prompt as input) and TRAIL's embedding-based predictions.



(c) Mean Latency (s) (d) Mean TTFT (ms)

Comparison of mean latency and Time to First Token (TTFT) across different preemption thresholds (c) at a request rate of 14.



(e) Mean Latency (f) Mean TTFT

Mean latency and TTFT, as a function of request rate for four LLM systems: (1) vLLM-FCFS, vLLM using FCFS; (2) vLLM-SJF_BERT, vLLM using SJF based on BERT; (3) with $c = 0.8$ and refined embedding predictions; and (4) -BERT with $c = 0.8$ using BERT predictions.