Fast-Inference of Mixture-of-Experts Language Models with Offloading

Main Idea: this paper investigates a strategy to run large MoE in consumer hardware with limited accelerator memory. It succeeds in running Mixtral 8x7B in a free-tier Google Colab instance.

Method for MoE Generative Inference

1. Encoding the input prompt.
   a. Done in parallel (layer-by-layer).
2. Generate tokens conditioned on the input prompt.
   a. Done sequentially (token-by-token and layer-by-layer).

In other words, step 1 is easy to optimize since we can simply pass all tokens in parallel layer-by-layer. During token generation this is not possible since we need to pass one token at a time, making the offloading challenging to optimize for.

Improvements from this approach

- Caching experts
  - To exploit the fact that previous work shows that activated experts tend to be active for more than one token at a time (common for them to stay active for 2-4 tokens in a row), the experts activated from the previous token can simply be stored in a GPU cache.

- Prefetching
  - With dense models, offloading is simple due to the fixed order of layers to load. This is not true in MoE, so future layers cannot be pre-loaded since they are usually selected based on the previous layer's output. To help with this, a speculative loading technique is developed based on the heuristic that the previous layer's hidden state can be a good proxy for the next hidden state (since these hidden states are only updated and not recomputed from scratch). This allows us to predict the next layer's experts before knowing its hidden state (in case of wrong guesses, the gains are lost since we must load the experts while no computations are being done).
- In terms of quantization, HQQ (data-free) is used for convenience, however, other techniques such as GPTQ could also work. (QMoE was experimented with on Mixtral 8x7B, but loss in quality was too significant due to the 1-bit quantization).
  - Found that ideally experts can be quantized to 3 or even 2 bits and that attention layers should be kept at a larger bit width (16 or minimum 4 bits).
- For expert offloading, a cache of 2 experts per layer is used with 12 GB GPUs and of 4 for 16 GB GPUs. Additionally, 1 or 2 experts per layer are loaded speculatively as soon as the previous layer's hidden states are available.

<u>Results (on Mixtral 8x7B)</u>

- On the free Google Colab tier, inference speed is of around 2 tokens per second.

- In terms of cache hits, the accuracy to guess the next expert goes from around 0.2 with cache size of 1 to around 0.6-0.7 for cache size of 4.
- For speculative loading the results are even better and show that active experts can be estimated even when 10 layers ahead (that is, using the hidden state of the 10th hidden layer behind it).

My takeaways:

- Able to use this method to experiment with Mixtral in Google Colab.