Mixtral of Experts (+Mistral 7B)

Main Idea: Mixtral is a recent MoE model that is based on the Mistral architecture (Mistral is a dense model). The difference between these models is that each Mixtral layer consists of sparse FFNs, when these are dense in Mistral, with each sparse layer containing 8 experts each and being the equivalent of a 7B Mistral model.

Mistral 7B

Mistral uses grouped-query-attention (GQA) for accelerated inference speed and reduced memory requirements (allowing larger batch sizes) and sliding window attention (SWA) for handling longer sequences at a lower computational cost. The goal of Mistral is to provide an open-source model that beats other existing open-source models of similar size while improving on inference speed and memory/computational requirements, with a focus on practical use of the model and ease of fine-tuning.

Sliding Window Attention (SWA)

In regular attention, each token in a sequence attends to every other token, resulting in a complexity of O(n^2) with respect to the sequence length. In SWA, the tokens attended are limited by a sliding window, which masks tokens that are farther away from the current token than a pre-defined distance. This changes the complexity to O(n*w), where w is the maximum number of tokens to be attended (maximum window size).

- SWA reduces computational complexity and memory usage – the longer the sequences the bigger the improvement.
- SWA, due to the fixed window size, allows for a rolling buffer cache (this increases efficiency).
- SWA also allows for pre-fill and chunking for more efficient inference.

Results

- Mistral is compared to Llama 2 7B/13B, Llama 1 34B and Code-Llama 7B.
- Compared to Llama 2 7B/13B and Llama 1 34B, Mistral performs significantly better in complex reasoning areas (code, math, reasoning) and comes close to Code-Llama 7B in coding tasks.
- On knowledge tasks, Mistral also tended to perform better but the gap observed was not as significant as in complex reasoning tasks.
- Instruction fine-tuning was performed using publicly available data to show the straightforwardness of fine-tuning on Mistral 7B.
    - This resulted in comparable performance to 13B instruct models.

Mixtral

- Mixtral uses top-2 token-choice routing.
- Mixtral excels at math, code generation and multilingual benchmarks (consistent with Mistral).

- A Mixtral-Instruct model (performed SFT and DPO) is also provided and surpasses GPT 3.5-Turbo.
- The context length of Mixtral is 32k.
- The gating mechanism of Mixtral takes the SoftMax of the top-2 expert scores and weights the expert's outputs based on these weights.
    - The final output is then a weighted average of the sum of the two selected experts' outputs.
- Mixtral seems to be robust to long-range contexts.
    - Perhaps due to Mistral's SWA?
    - Experiments showed that up to a context length of 30k tokens, information can accurately be retrieved, and the perplexity of Mixtral decreases with an increase in context length.

The name Mixtral 8-7B might induce the thought of the architecture having 56B total parameters (8*7), but it consists of around 47B parameters due to shared parameters between experts across the embedding, attention and normalization layers (7B is the full size of each expert if converted to a dense model). Likewise, the inference cost is not the equivalent of running 14B parameters (7*2), but around 13B parameters due to these shared parameters.

In terms of routing analysis, it was shown that experts seem to be selected based on syntax rather than on specific domains – experts specialize in semantics and syntax, not on tasks. This is logical due to the token-choice routing. If routing is done on a token granularity, the experts are expected to specialize on token-level areas. With domain or task-routing (done at a sequence level), experts can be expected to specialize in domain/task-level areas.

My takeaways:

- The goal of Mistral 7B is to provide an open-source model with an optimal performance and efficiency balance.
  - Performance meaning quality, efficiency meaning inference speed and computational requirements.
- Sliding Window Attention seems to sacrifice the context length capacity in return of higher inference speed. The assumption taken for this not to hurt performance seems to be that the more you move away from a token, the lower the odds of it having meaningful dependencies to the current token.
  - Large context lengths are possible under SWA, but each individual token will not use the full context length for inference if the input is larger than the maximum window size.
- Perhaps the idea for Mixtral came after analyzing Mistral's results? Since Mistral performs significantly better on reasoning tasks but the improvement in knowledge tasks is not so big, it would make sense to try to apply a MoE architecture to this model, with the idea being to retain the reasoning abilities while improving knowledge abilities. This makes sense because other studies seem to show that MoE, due to additional model capacity added, tend to perform very well on knowledge tasks (weakness of Mistral) but the

performance on reasoning and fine-tuning tasks (strength of Mistral) leaves room for improvement (although MoE was shown to benefit from instruction-tuning in a more significant way than dense models).