Multi-Headed Lory: Fine-Grained Differentiable Mixture-of-Experts

Michał Maszkowski, Antoni Janowski, Miriam Lipniacka
m.maszkowsk2, m.lipniacka@student.uw.edu.pl, aj421072@students.mimuw.edu.pl
Supervisor: Jan Ludziejewski

June 19, 2024

1 Introduction

Transformer architecture (Vaswani et al., 2017) introduced parallelizable way to combine information from distant parts of sequences allowing effective language modeling. It was improved by Mixture-of-Experts [MoE] (Shazeer et al., 2017) which allowed to train models with significantly more parameters without much increase in training or inference costs. But due to non-differentiable discrete routing, such as top-k expert-activation routing, MoE has a problem with learning effective routing strategies (Muqeeth et al., 2023). It also suffers from low expert activation and lack of fine-grained capabilities to analyze multiple semantic concepts within individual token (Wu et al., 2024).

Recently multiple works have tried to address non-differentiability of expert choice in the MoE architecture (Puigcerver et al., 2023, Muqeeth et al., 2023). In (Muqeeth et al., 2023) each token is processed by a "merged expert" which has parameters that are a sum of experts' parameters weighted uniquely for every token according to the router. This operation is differentiable, but increases computational complexity, scaling linearly with the number of experts, which makes it impractical for auto-regressive tasks. This problem is answered by Lory (Zhong et al., 2024) which recomputes the "merged expert" only once for a segment (comprising about few hundred tokens) instead of doing so for each token, while avoiding information leakage.

A related problem of low expert activation is solved by Multi-Head Mixture-of-Experts (MH-MoE)(Wu et al., 2024). At the same time it claims to answer the lack of fine-grained capabilities to analyze multiple semantic concepts within individual token. It does so by splitting tokens into sub-tokens and then assigning and processing those in parallel with experts before reintegrating them.

2 Related work

2.1 Lory

Assume that we have E feed-forward expert networks, each parameterised by weights θ_i. To process a sequence of L tokens, with hidden representations $h_1, h_2, \dots h_L$ we divide it into N segments of length $T = \frac{L}{N}$. Each segment will be processed by a single "merged expert" that is created based on the information from the previous segment. Let's suppose that we currently want to start to process segment $k > 1$ (so we want to process tokens $h_k, h_{k+1} \dots, h_{k+T-1}$, and previous segment consisted of tokens $h_{k-T}, h_{k-T+1} \dots, h_{k-1}$). We process it like this:

$$\text{output}(h_{k+i}) = \text{FFN}(h_{k+i}, \sum_{j=1}^{E} \theta_j w_j), \text{ where } w_j = \text{Softmax}(R(\frac{1}{T}\sum_{q=0}^{T-1} h_{k-T+q}))_j$$

where $\text{FFN}(a, b)$ means feed-forward networks with weights b on input a and R is our router network.

This approach allows us to compute weighted sum of experts only once per segment making it suitable for auto-regressive tasks. The authors also used similarity-based data batching technique introduced by (Shi et al., 2023).

It prevents information leakage and keep the causal nature of the model.

The authors also used similarity-based data batching technique introduced by (Shi et al., 2023). It ensures that batches contain similar documents, so even when one document ends and other start inside one segment, routing decision made based on the previous document can still be relevant to the next document, which enhances expert specialization in later layers.

2.2 MH-MoE

MH-MoE uses a multi-head mechanism to split each input token into sub-tokens, then processes them in parallel by diverse experts, and then reintegrates them into the original token form. Let us denote number of heads as H and token of hidden size $h \in \mathbb{R}^d$ split into H sub-tokens as $h_1, h_2, ..., h_H \in \mathbb{R}^{\frac{d}{H}}$. Then output of the p-th sub-token from j-th token MH-MoE layer would be computed as:

$$output(h_p^j) = h_p^j + \sum_{i=1}^{i=k} e_i(h_p^j) \cdot g(h_p^j, e_i)$$

This way the average volume of data routed to a specific expert is increased by a factor of H. Additionally, while current tokenization patterns limit a model's ability to capture multiple semantic interpretations of tokens, MH-MoE enables capturing information from diverse feature spaces across these experts.

3 Project proposal

Both Lory and MH-MoE report promising gains in performance, while using techniques that we believe can be beneficially combined into MH-Lorry. Our main goal is to compare 2 versions of our model to Lory, MH-MoE and baseline MoE in terms of perplexity on a language modelling task with the equal number of active parameters. We will also compare the overall number of parameters that each has, and estimate inference and training efficiency.

Constrained by limited computational resources we plan to train and validate the aforementioned models on the Wikipedia dataset (Foundation, n.d.). We leave it for a future research to test how these models scale. We hope to pursue this question in case of promising experiment results.

Combining Lory and MH-MoE architectures admittedly poses a challenge as their strategies aren't orthogonal but diverge slightly. Lory achieves full activation of experts by merging them in pursue of full differentiability. Keeping this feature in the MH-Lory makes one of two advantages of MH-MoE irrelevant as introducing heads can't increase expert activation further. What's more, in Lory whole segments are processed by the same "merged expert" while MH-MoE goes in different direction, routing mere parts of tokens to different experts. Thus Lory decreases the variability of assigned experts between tokens compared to traditional MoE while MH-MoE does the opposite.

The most natural ways to combine Lory and MH-MoE is to use the Lory model as a baseline and modify it by adding heads. More precisely MH-Lory will calculate the distribution over experts used for making "merged expert" once per segment, separately for each head. We hope that we can still make an improvement on Lory by reaping the rewards of allowing more fine-grained understanding of multiple semantic concepts within a token as happens in MH-MoE. As an additional benefit by introducing expert heads to Lory architecture we decrease the number of parameters per expert allowing to increase the number of experts.

We will experiment with and compare two versions of MH-Lory depending on whether all heads share the set of experts from which a "merged expert" is computed (v1) or whether each head has a separate set of experts (v2). Intuitively MH-Lory-v1 has advantage of increasing the number of examples (sub-tokens) per expert and having less parameters. But on the other hand MH-Lory-v2 allows for greater specialization of heads, potentially leading each head to focus on a particular aspect of token meaning. There are N experts in MH-Lory-v1 and $N \cdot H$ experts in MH-Lory-v2.

References

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., & Dean, J. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

Muqeeth, M., Liu, H., & Raffel, C. (2023). Soft merging of experts with adaptive routing. *arXiv preprint arXiv:2306.03745*.

Puigcerver, J., Riquelme, C., Mustafa, B., & Houlsby, N. (2023). From sparse to soft mixtures of experts. *ArXiv*, *abs/2308.00951*. https://api.semanticscholar.org/CorpusID: 260378993

Shi, W., Min, S., Lomeli, M., Zhou, C., Li, M., Lin, V., Smith, N. A., Zettlemoyer, L., Yih, S., & Lewis, M. (2023). In-context pretraining: Language modeling beyond document boundaries. *arXiv preprint arXiv:2310.10638*.

Wu, X., Huang, S., Wang, W., & Wei, F. (2024). Multi-head mixture-of-experts.

Zhong, Z., Xia, M., Chen, D., & Lewis, M. (2024). Lory: Fully differentiable mixture-of-experts for autoregressive language model pre-training.

Foundation, W. (n.d.). *Wikimedia downloads*. https://dumps.wikimedia.org