From Sparse to Soft Mixture of Experts

Joan Puigcerver, Carlos Riquelme, Basil Mustafa, Neil Houlsby
Google DeepMind

Presented to MLBBQ reading group by Riyasat Ohib

Preliminaries: Mixture of Experts (MoE)

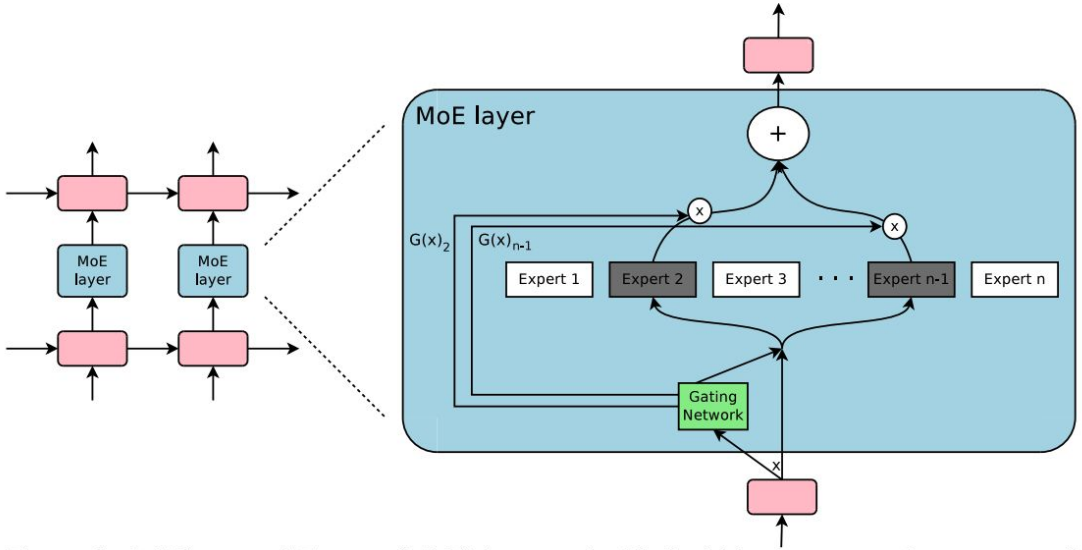


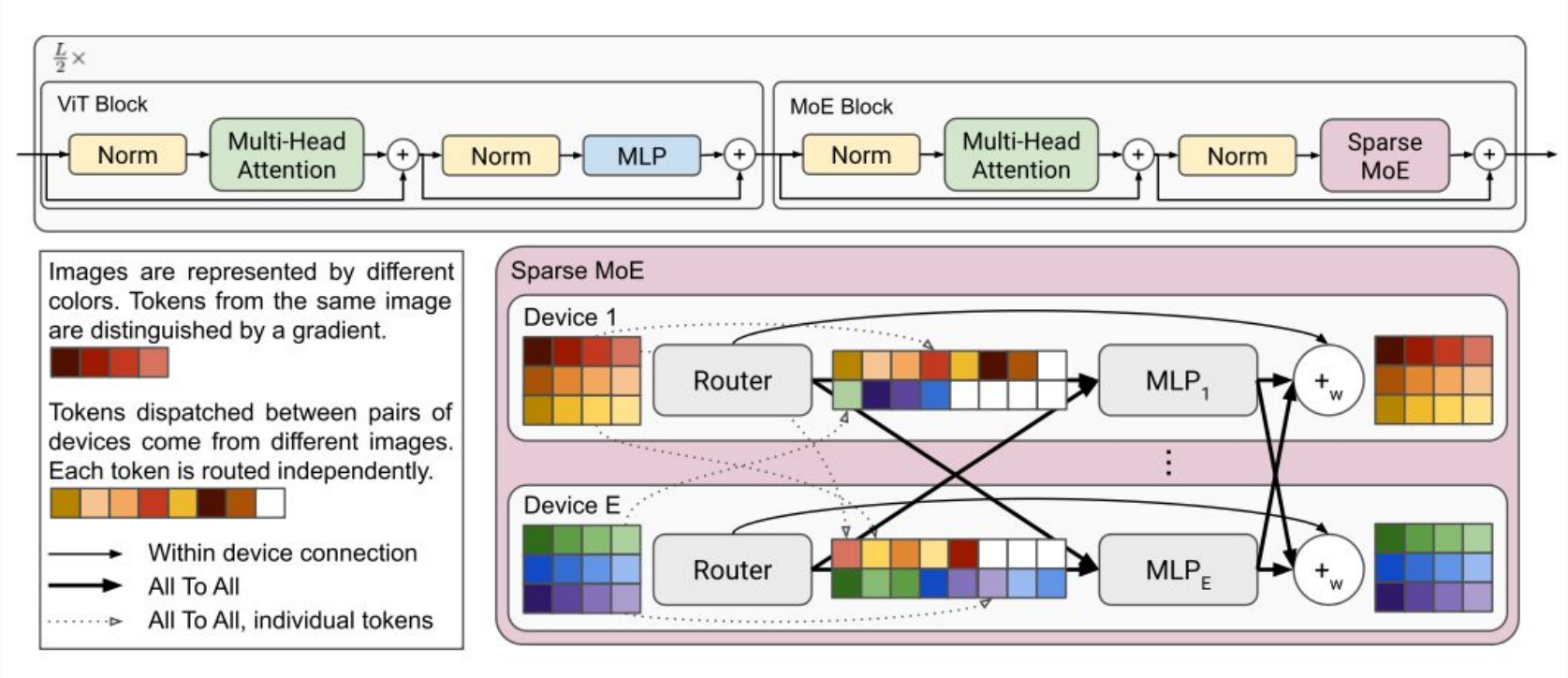
Figure 1: A Mixture of Experts (MoE) layer embedded within a recurrent language model. In this case, the sparse gating function selects two experts to perform computations. Their outputs are modulated by the outputs of the gating network.

The MoE layer consists of n "Expert networks" E1 to En and a gating network G whose output is a sparse n-dimensional vector.

$$y = \sum_{i=1}^{n} G(x)_i E_i(x)$$

Shazeer, Noam, et al. "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer." arXiv preprint arXiv:1701.06538 (2017).

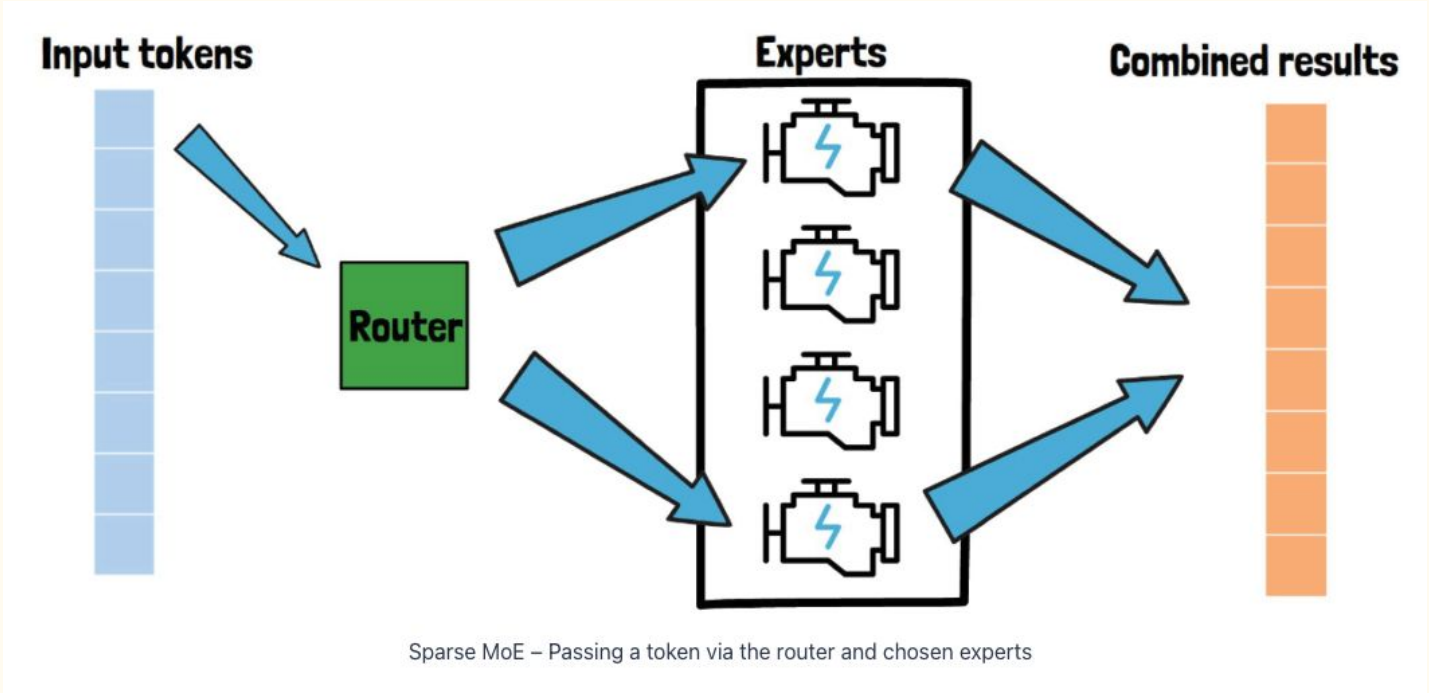
Sparse Mixture of Experts (MoE)



Sparse Mixture of Experts (MOE)

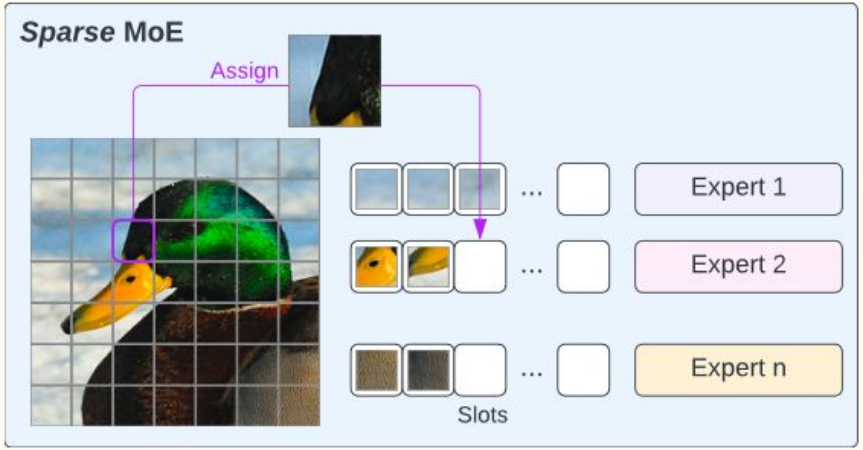
Vanilla vs Batch Prioritized Routing

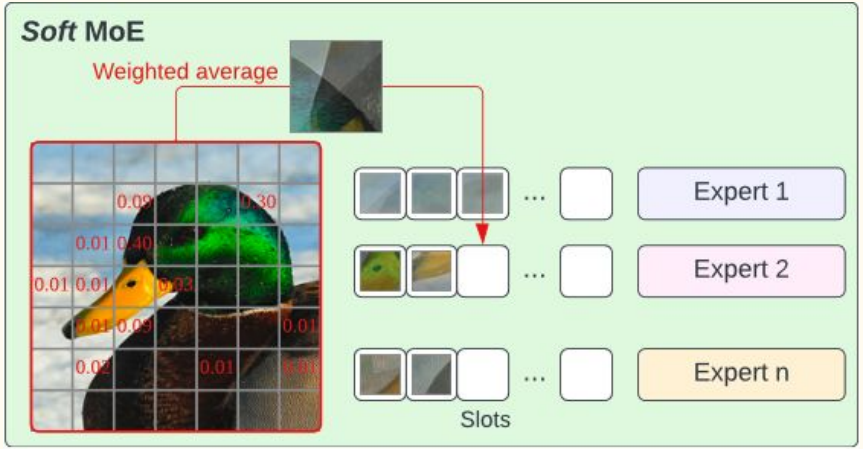
Sparse Mixture of Experts (MOE)



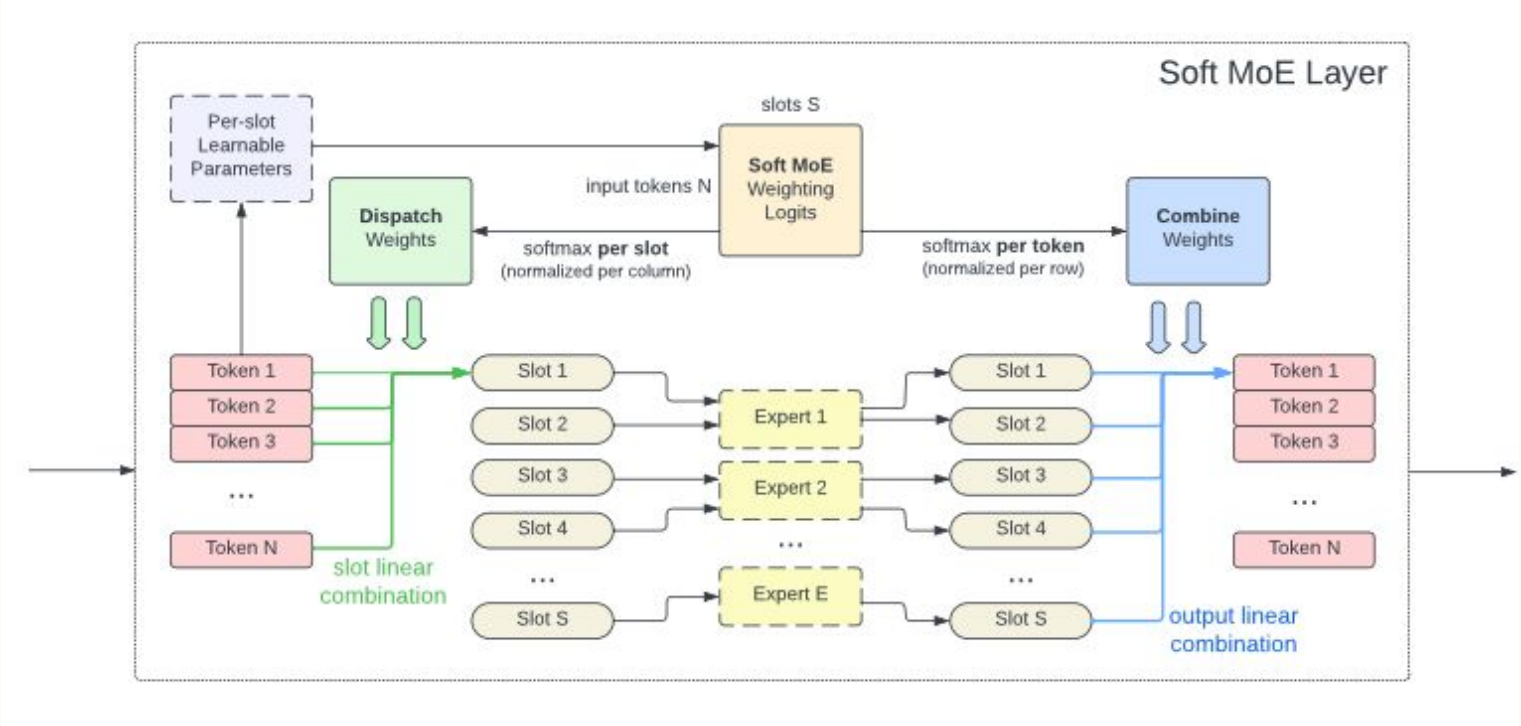
Sparse MoE – Passing a token via the router and chosen experts

From Sparse to Soft MoEs





Soft MOE Layer



Dispatch Weights

The Soft MoE routing algorithm is depicted in Figure 2. We denote the inputs tokens for one sequence by $\mathbf{X} \in \mathbb{R}^{m \times d}$, where m is the number of tokens and d is their dimension. Each MoE layer uses a set of n expert functions[1] applied on individual tokens, namely $\{f_i : \mathbb{R}^d \to \mathbb{R}^d\}_{1:n}$. Each expert will process p *slots*, and each slot has a corresponding d-dimensional vector of parameters. We denote these parameters by $\mathbf{\Phi} \in \mathbb{R}^{d \times (n \cdot p)}$.

In particular, the input slots $\tilde{\mathbf{X}} \in \mathbb{R}^{(n \cdot p) \times d}$ are the result of convex combinations of all the m input tokens, $\mathbf{X}$:

$$\mathbf{D}_{ij} = \frac{\exp((\mathbf{X}\mathbf{\Phi})_{ij})}{\sum_{i'=1}^{m} \exp((\mathbf{X}\mathbf{\Phi})_{i'j})} \tag{1}$$

$$\tilde{\mathbf{X}} = \mathbf{D}^\top \mathbf{X}.$$

Notice that $\mathbf{D}$, which we call the *dispatch* weights, is simply the result of applying a softmax over the *columns* of $\mathbf{X}\mathbf{\Phi}$. Then, as mentioned above, the corresponding expert function is applied on each slot (i.e. on rows of

1. Soft MoE first computes scores or logits for every pair of input token and slot, based on some learnable per-slot parameters.
2. These logits are then normalized per slot (columns) and every slot computes a linear combination of all the input tokens based on these weights (in green)

Combine Weights

$\tilde{\mathbf{X}}$) to obtain the output slots:

$$\tilde{\mathbf{Y}}_i = f_{\lfloor i/p \rfloor}(\tilde{\mathbf{X}}_i). \tag{2}$$

Finally, the output tokens $\mathbf{Y}$ are computed as a convex combination of all $(n \cdot p)$ output slots, $\tilde{\mathbf{Y}}$, whose weights are computed similarly as before:

$$\mathbf{C}_{ij} = \frac{\exp((\mathbf{X}\mathbf{\Phi})_{ij})}{\sum_{j'=1}^{n \cdot p} \exp((\mathbf{X}\mathbf{\Phi})_{ij'})} \tag{3}$$

$$\mathbf{Y} = \mathbf{C}\tilde{\mathbf{Y}}.$$

We refer to $\mathbf{C}$ as the *combine* weights, and it is the result of applying a softmax over the *rows* of $\mathbf{X}\mathbf{\Phi}$.

1. Each expert (an MLP in this work) then processes its slots.
2. the same original logits are normalized per token (i.e. by row) and used to combine all the slot outputs, for every input token (in blue).
3. For each token it is known how important it was for each slot and so we consider the expert slot result for each token

Properties of Soft MoEs: Differentiability

Fully Differentiable:

- **At the core of all Sparse MoE algorithms is an assignment problem between tokens and experts.**

Two Popular Choices that approximate the solution:

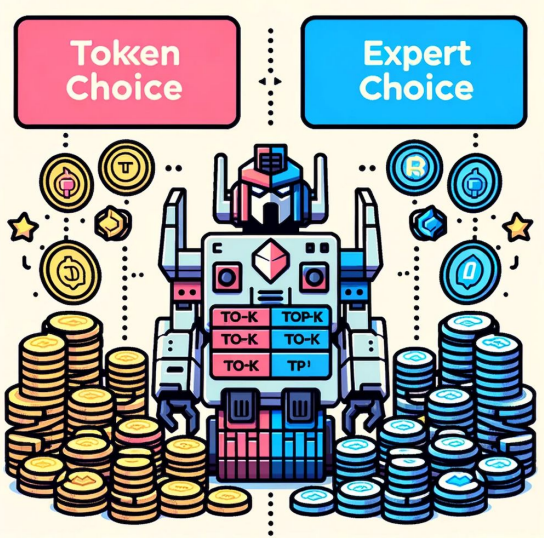
1. **Token Choice:** Top-k Tokens routed to each expert.
2. **Expert Choice:** Experts choose the top tokens based on it's set capacity.

Other Choices include more sophisticated but costly algorithmic solutions:

- **Linear Programming algorithms**
- **Optimal Transport**
- **Reinforcement Learning**

All of these approaches are discrete in nature and thus non-differentiable. In contrast Soft MoE layers are continuous and fully differentiable.

Properties: No token Dropping or Expert Unbalance



Two Popular Choices for Routing:

1. **Token Choice:** Top-k Tokens routed to each expert.
2. **Expert Choice:** Experts choose the top tokens based on it's capacity.

Issues:

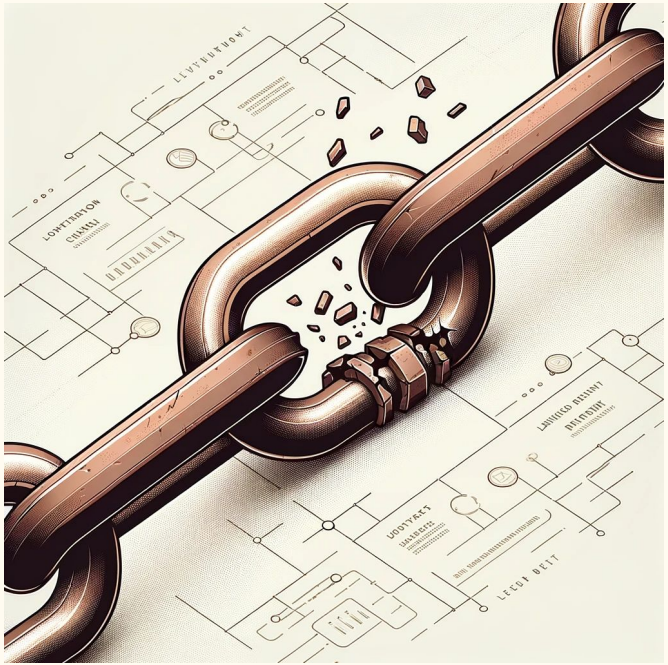
1. **Token Choice** suffers from "token dropping" (some tokens are not assigned to any expert), or "expert unbalance" (some experts receive far more tokens than others).
2. **Expert Choice** only suffers from "token dropping".

Both of these have detrimental effects on the model performance.

Soft MoE

- **Soft MoEs are basically immune to token dropping and expert unbalance** since every slot is filled with a weighted average of all tokens.

Limitations



Auto-regressive Decoding:

- One of the key aspects of Soft MoE consists in smartly merging all tokens in the input. This makes the use of Soft MoEs in auto-regressive decoders difficult, since causality between past and future tokens has to be preserved during training.
- Although causal masks used in attention layers could be used, one must be careful to not introduce any correlation between token and slot indices, since this would bias which token indices each expert is trained on.

Large Memory Footprint:

Consequently, Soft MoE tends to leverage a large number of experts and –while its cost is still similar to the dense backbone– the memory requirements of the model can grow large.

Visualization of Linear Combinations

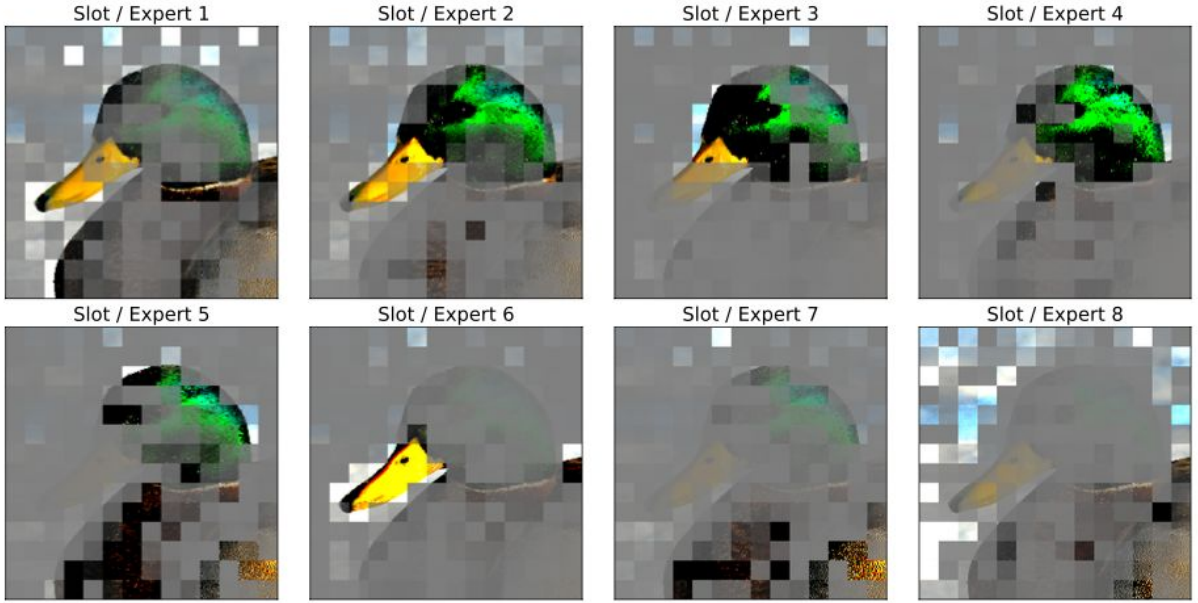
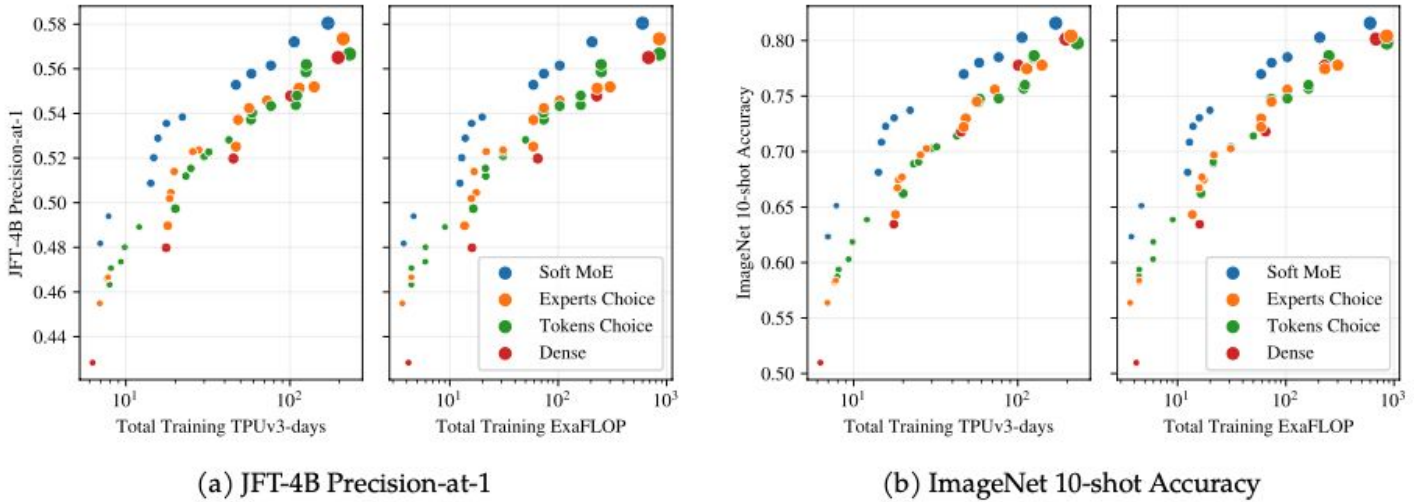


Figure 10: Linear combinations for 8 slots when using input image in Figure 1. Model is Soft MoE S/16 with 128 experts and one slot per expert, and it was finetuned on ImageNet. We show results for the first MoE layer (seventh block). The selected slots (among 128) are cherry-picked to highlight differences across slots.

Experimental Results: Pareto Models



(a) JFT-4B Precision-at-1

(b) ImageNet 10-shot Accuracy

Figure 3: **Pareto Models.** Soft MoE dominates both ViTs (dense) and popular MoEs (Experts Choice,Tokens Choice) on the training cost / performance Pareto frontier. Each point is a model trained for 300k steps, and larger marker sizes indicate larger models: S/32, S/16, B/32, B/16, L/16 and H/14. Cost is shown both in terms of FLOPS and realized TPU-v3 training time. MoE runs include different configuration; for clarity, only models on their respective Pareto frontier are displayed. Figure 22 in Appendix F shows all models.

Experimental Results

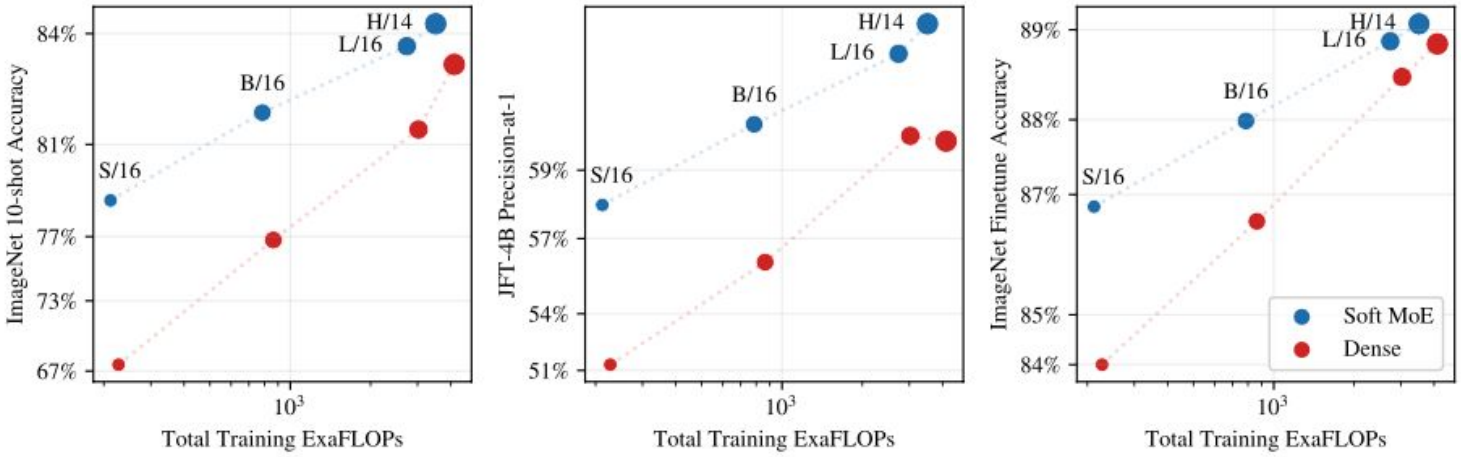
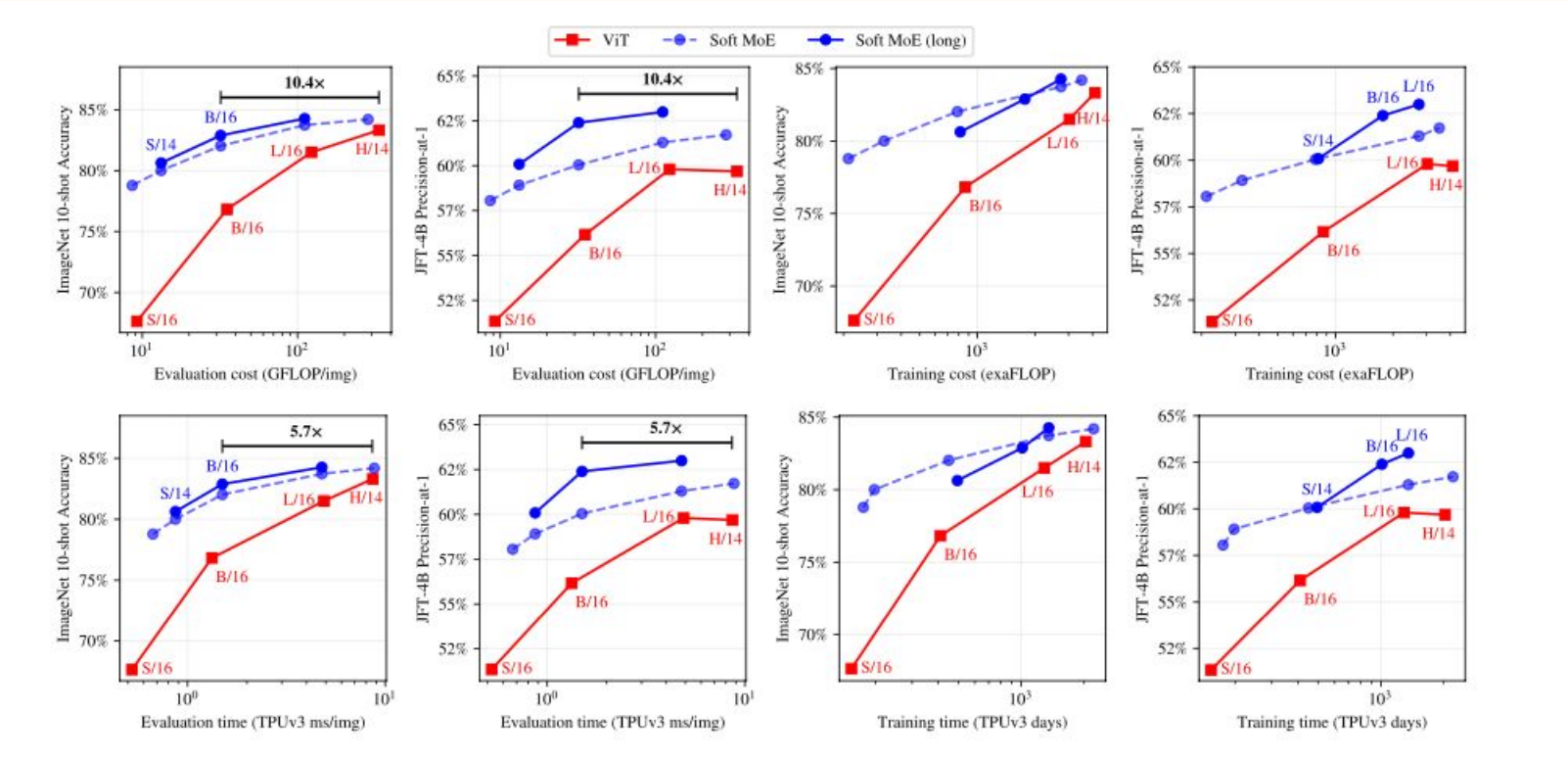


Figure 4: **Long runs.** Soft MoE and ViT models trained for 4 million steps with batch size 4096 (H/14 models trained for 2 million steps instead). Equivalent model classes (S/16, B/16, L/16, H/14) have similar training costs, but Soft MoE outperforms ViT on all metrics. We show ImageNet 10-shot (left), JFT precision at 1 (middle) and ImageNet accuracy after finetuning (right), versus total training FLOPs. See Table 2. We report training wall-clock time in Figure 19.

Experimental Results: ViT, Soft MoE and Soft MoE

Imagenet 10 Shot Accuracy



Experimental Results: Scaling Experts

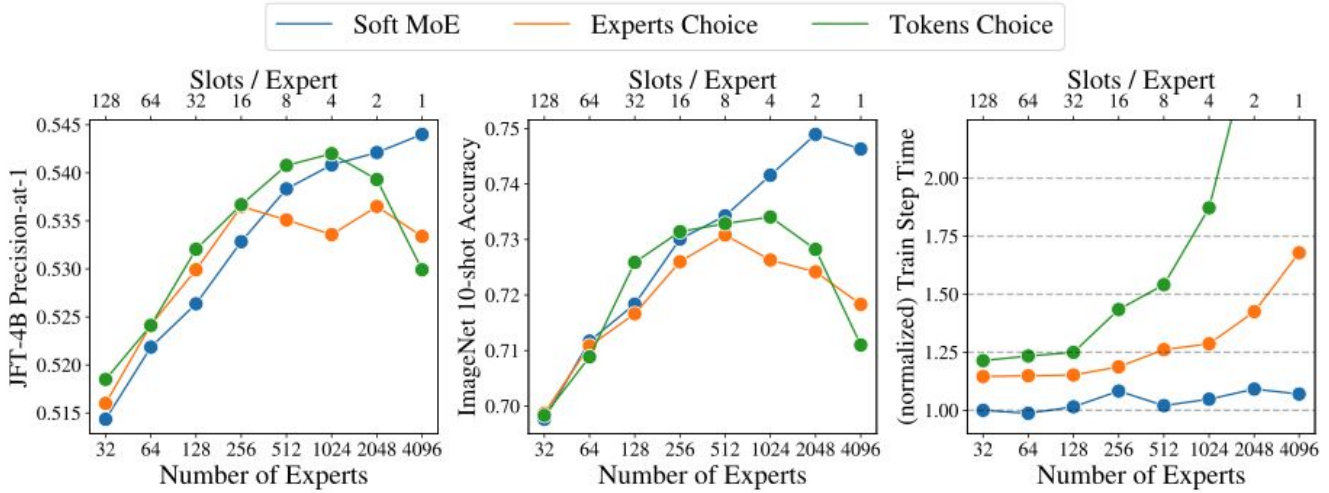


Figure 6: **Performance (left, center), and training step time (right) as a function of number of experts, for models with a fixed number of slots (Soft MoE) or expert buffer capacity (Sparse MoEs)** on a ViT-S/16 backbone with MoEs in the last two layers. Soft MoE achieves much better scaling with more experts, while cost is roughly constant. However, with Experts and Tokens Choice routers, having too many experts not only hurts performance but also significantly increases the cost (Tokens Choice reaches 3.9x with 4096 experts).

Questions and Discussion

Connection with other methods

- Although dispatch and combine weights are computed in a similar fashion methods that focus on reducing sequence lengths, the authors state **their goal is not to reduce the sequence length**
- They actually recover the original sequence length after weighting the experts' outputs with the combine weights, at the end of each Soft MoE layer.

Similarity to Multi-headed Attention: