# Enhancing Generalization in Sparse Mixture of Experts Models: The Case for Increased Expert Activation in Compositional Tasks

**Jinze Zhao**
University of Texas at Austin
jz24694@utexas.edu

**Junjie Yang**
The Ohio State University
yang.4972@osu.edu

**Peihao Wang**
University of Texas at Austin
peihaowang@utexas.edu

**Zhangyang Wang**
University of Texas at Austin
atlaswang@utexas.edu

**Yingbin Liang**
The Ohio State University
liang.889@osu.edu

## Abstract

As Transformer models grow in complexity, their ability to generalize to novel, compositional tasks becomes crucial. This study challenges conventional wisdom about sparse activation in Sparse Mixture of Experts (SMoE) models when faced with increasingly complex compositional tasks. Through experiments on the SRAVEN symbolic reasoning task and SKILL-MIX benchmark, we demonstrate that activating more experts improves performance on difficult tasks, with the optimal number of activated experts scaling with task complexity. Our findings reveal that pretrained SMoE-based Large Language Models achieve better results by increasing experts-per-token on challenging compositional tasks.

## 1 Introduction

The Sparse Mixture of Experts (SMoE) model, introduced by [26], has shown exceptional promise in the field of neural networks, especially for expanding model size without significantly increasing computational demands. SMoE's innovative approach involves dividing the traditional feed-forward network into several homogeneous expert networks. These expert networks are then dynamically and sparsely combined by a data-driven neural network called the router. This strategy enables a considerable increase in the overall parameter count while maintaining relatively constant computational costs, as only a small group of experts is activated for each input. Additionally, SMoE shows potential for enhancing model generalization due to its special modularity [15], particularly when working with diverse data domains. It facilitates collaborative learning among expert groups, allowing for generalization to novel domains. More recently, SMoE has been chosen to be the standard architecture of many Large Language Models (LLMs) due to its performance superiority [10, 28, 4, 3, 19, 22, 14]. However, **compositional generalization** requires the model to solve tasks composed by arbitrary number of base skills, and such novel combination is not encountered during training and the difficulty of such tasks can be built up exponentially. It is counterintuitive to believe that having limited number of experts (e.g., Top-1 or Top-2 activation out of 8 experts as SMoE-based LLMs suggested) can lead to good compositional generalization when the task becomes harder. Therefore, we are trying to empirically study the following questions in this work: **Is sparse activation always an optimal strategy as we increase the difficulty of compositional tasks?**

To address this question, our work delivers two sets of experiments as following, with a summary of their contributions:

NeurIPS 2024 Workshop on Compositional Learning: Perspectives, Methods, and Paths Forward.

- We trained standard SMoE-based Transformers from scratch on SRAVEN [24], a symbolic synthetic compositional task, with varying difficulty levels of the task and varying number of activated experts. Our results show that activating more experts can result higher Out-of-Distribution (OOD) accuracy and test accuracy when we train on more difficult tasks, and the optimal number of activated experts scales roughly with the number of features of the synthetic task.
- We tested (inferenced) on pretrained SMoE-based Large Language Models such as Mixtral-8$\times$7B [10] and DBRX 132B Instruct [4] on SKILL-MIX benchmark test [29], a high level compositional task that requires the model to construct a short paragraph fitting $k$ skills from linguistic studies. Our results show that these pretrained SMoE-based Large Language Models can obtain better performance for free by activating more experts-per-token when we test them with more challenging SKILL-MIX tasks.

## 2 Experiments

We postponed the related works section to Section A.1 due to page limit. We delivers two sets of experiments to study if sparse experts activation is always optimal when handling compositional tasks with varying difficulties during both training and inferencing phase.

### 2.1 Training SMoE-based Transformers on SRAVEN task

We trained standard decoder-only SMoE-based transformers on SRAVEN synthetic task. Each transformer block consists of Multi-Head attention (e.g., softmax attention or hypernetwork linear attention proposed by [24]) with relative positional encoding [20] and feedforward layer (FFN). The feedforward layer in each block is a SMoE structure with 8 parallel homogeneous experts, where each expert is a 2-layer multi-layer perceptron (MLP). The router is a simple 1-layer dense layer with Top-K softmax gating mechanism. More implementation details can be found in Section A.2.1. For the SRAVEN hyperparameters, we fixed the grid size of the problem to be $3 \times 3$, meaning we fix the number of in-context examples for the model. We have $R = 8$ possible rules to be sampled from. We can adjust the task difficulty by sampling $M \in \{1, 2, \cdots, R\}$ different rules to compose the task. We keep $25\%$ of the whole SRAVEN examples as OOD samples to evaluate if the model can generalize compositionally in OOD fashion.

#### 2.1.1 Sparse activation is not optimal over all difficulty settings

We trained SMoE transformers with different Top-K routing mechanisms until the same number of iterations over varying difficulty levels of the SRAVEN task, while setting all other model/training hyperparameters the same. Surprisingly, Top-1 routing obtains the lowest Out-of-Distribution (OOD) accuracy over all difficulty settings. Though Top-2 routing performs roughly the same as other more expensive routing mechanisms when the task is easy, it also starts to fall behind as the compositional task becomes harder, as shown in Figure 1. Similar trends can be found in the Test Accuracy evaluations, as shown in Figure 2 in Section A.2.2. Therefore, we claim that the most commonly-used sparse activation mechanisms are not optimal in learning compositional tasks, which challenges previous studies [26, 10].

#### 2.1.2 The optimal number of activated experts roughly scales with task difficulty

We also observe that as we increase the number of sampled rules $M$, more activated experts are required to obtain the optimal performance correspondingly. Starting from the setting where $M = 4$, the optimal number of activated experts $K$ roughly scales up with $M$, and the performance gap between each activation mechanisms also widens, as shown in Figure 1 and Figure 2. Therefore, We conjecture that each expert can specialize on specific rules when we train the model on compositional tasks, though such phenomenon is not observed when training on traditional language tasks [8].

#### 2.1.3 Ablation study: Switching Softmax attention to HYLA attention

Proposed by [24], Hypernetwork Linear Attention (HYLA) encourages compositional generalization by reinforcing the hypernetwork perspective on standard Transformers. We repeat the same set

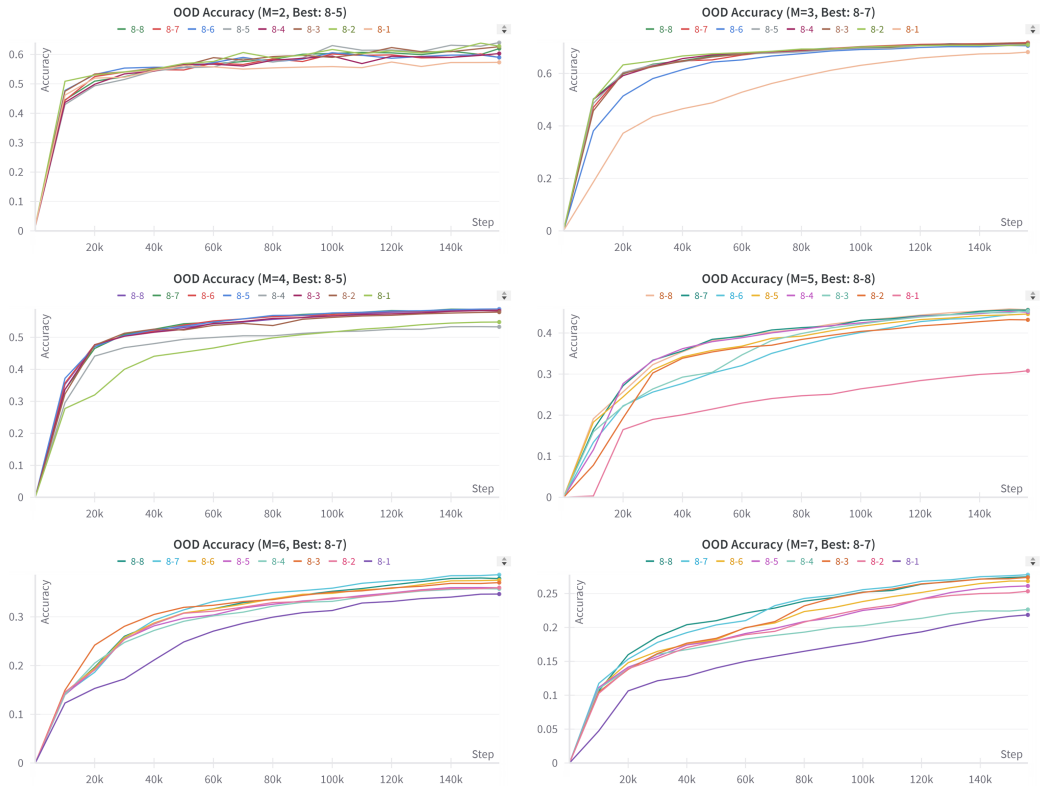


Figure 1: OOD Accuracy of training SMoE Transformer with softmax attention, where the difficulty level of the task (i.e., the number of sampled rules $M$) increases. "8-$k$" means activating $k$ out of 8 experts on every FFN layer. The best performing activation mechanism is labeled on the caption of each figure.

of experiments by switching from softmax attention to HYLA attentions. We have the following interesting findings:

- By comparing Figure 5 and Figure 1, we can see that HYLA does not improve or even degrade the compositional generalization (especially the OOD accuracy) on SMoE models when training on easy compositional tasks (i.e., $M = 2$) comparing to the case when we use softmax attention. However, as the trained compositional task becomes more challenging, HYLA will significantly outperform softmax attention and improve the compositional generalization of the model over all activation mechanisms.

- As shown in Figure 4 and Figure 5, Sparse activation mechanisms like Top-1 and Top-2 are still the worst-performing routing mechanisms over almost all task difficulty levels after opting HYLA attention, echoing our main findings in Section 2.1.1.

- As Figure 2 and Figure 1 suggested, under softmax attention setting, increasing the number of activated experts brings improvement on both Test and OOD accuracy when the compositional task becomes harder, though very marginal. However, the same change in increasing the number of activated experts will bring dramatic improvement for more challenging compositional tasks under HYLA attention setting, as shown in the last row of Figure 4 and Figure 5. The optimal number of activated experts also roughly scales with task difficulty $M$, and the best routing mechanism can outperform the second best one significantly. We conjecture that HYLA attention can furtherly improve the expert specialization on compositional tasks.

### 2.2 Evaluating SMoE-based Large Language Models on Skill-Mix

We evaluate two instruction-tuned SMoE-based Large Language Models (Mixtral-8×7B Instruct-v0.1 [10] and DBRX 132B Instruct [4]) on Skill-Mix. We choose GPT-4 [18] as the grading model. We use the released $10\%$ of the skill list and the topic list from Section A in [29]. We use the same optimized generation prompt for querying both Mixtral and DBRX. We use the same grading prompt and the evaluation metrics as [29] to query GPT-4 as our grading model.

#### 2.2.1 More Experts-per-token helps compositional generalization on harder tasks

We run Skill-Mix evaluation on Mixtral-8×7B Instruct-v0.1 with varying task difficulty $k$ (i.e., the number of skills that the model need to compose in its generated answer) and varying number of experts-per-token (ept). We observe similar patterns from testing both Mixtral and DBRX where the results are shown in Table 1 and Figure 6, and summarize our main findings as following:

- As we increase the difficulty of Skill-Mix task, more experts per token are required to maintain better performance, as shown in the highlighted part of Table 1. For example, the generated answer produced by the default Top-2 routing gets all zeros on all grading metrics when $k = 4$.
- The optimal number of experts per token roughly scales with the difficulty of Skill-Mix $k$. For example, setting experts per token to be 4 or 5 obtains the best performance when $k = 4$.
- Different from the SRAVEN training experiment which showcased that more activated experts cannot give worse performance on simpler compositional tasks, the Skill-Mix evaluation shows that activating more experts during inferencing can do harm to the compositional generalization. For example, while the model with default Top-2 routing setting can obtain perfect scores on $k = 1$ and $k = 2$ Skill-Mix tasks, the model's performance can substantially decline if we set experts per token to be 7 or 8.

| Graded by GPT-4 | Skill-Mix ($k$=1) | Skill-Mix ($k$=2) | Skill-Mix ($k$=3) | Skill-Mix ($k$=4) | Skill-Mix ($k$=5) |
|---|---|---|---|---|---|
| ept=1 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.20 ± 0.082 | 0.35 ± 0.100 | 0.00 ± 0.000 |
| ept=2 (default) | 1.00 ± 0.000 | 1.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| | 1.00 ± 0.000 | 1.00 ± 0.000 | 1.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| | 1.00 ± 0.000 | 1.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| ept=3 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| | 0.40 ± 0.245 | 0.20 ± 0.200 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| | 0.00 ± 0.000 | 0.10 ± 0.100 | 0.47 ± 0.082 | 0.60 ± 0.061 | 0.00 ± 0.000 |
| ept=4 | 0.20 ± 0.200 | 0.20 ± 0.200 | 0.20 ± 0.200 | 0.20 ± 0.200 | 0.00 ± 0.000 |
| | 0.80 ± 0.200 | 0.40 ± 0.245 | 0.20 ± 0.200 | 0.20 ± 0.200 | 0.20 ± 0.200 |
| | 0.20 ± 0.200 | 0.40 ± 0.187 | 0.67 ± 0.105 | 0.75 ± 0.079 | 0.00 ± 0.000 |
| ept=5 | 0.20 ± 0.200 | 0.20 ± 0.200 | 0.20 ± 0.200 | 0.20 ± 0.200 | 0.00 ± 0.000 |
| | 0.40 ± 0.245 | 0.20 ± 0.200 | 0.20 ± 0.200 | 0.20 ± 0.200 | 0.20 ± 0.200 |
| | 0.20 ± 0.200 | 0.30 ± 0.200 | 0.53 ± 0.133 | 0.65 ± 0.100 | 0.00 ± 0.000 |
| ept=6 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| | 0.20 ± 0.200 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| | 0.00 ± 0.000 | 0.20 ± 0.122 | 0.47 ± 0.082 | 0.60 ± 0.061 | 0.00 ± 0.000 |
| ept=7 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| | 0.00 ± 0.000 | 0.10 ± 0.100 | 0.40 ± 0.067 | 0.55 ± 0.050 | 0.00 ± 0.000 |
| ept=8 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| | 0.40 ± 0.245 | 0.20 ± 0.200 | 0.00 ± 0.000 | 0.00 ± 0.000 | 0.00 ± 0.000 |
| | 0.00 ± 0.000 | 0.10 ± 0.100 | 0.47 ± 0.082 | 0.60 ± 0.061 | 0.00 ± 0.000 |

Table 1: Skill-Mix Evaluation Results on Mixtral-8×7B Instruct-v0.1 [11]. The grading metrics are Ratio of Full Marks/Ratio of All Skills/Skill Fraction as defined in Section A.3.1.

## 3 Conclusion

In this paper, we pioneered to empirically investigate if the commonly-used sparse activation on SMoE models always optimal upon compositional tasks. We delivered two sets of experiment from both training and inferencing perspective on compositional tasks, showing that more activated experts are required in order to generalize compositionally on more challenging tasks. We hope that our work can shed light on building more compositional generalizable model architectures.

## References


[1] S. Arora and A. Goyal. A theory for emergence of complex skills in language models. *arXiv preprint arXiv:2307.15936*, 2023.

[2] N. Chomsky. *Aspects of the Theory of Syntax*. Number 11. MIT press, 2014.

[3] D. Dai, C. Deng, C. Zhao, R. X. Xu, H. Gao, D. Chen, J. Li, W. Zeng, X. Yu, Y. Wu, Z. Xie, Y. K. Li, P. Huang, F. Luo, C. Ruan, Z. Sui, and W. Liang. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models, 2024.

[4] Databricks. Introducing dbrx: A new state-of-the-art open llm. `https://www.databricks.com/blog/introducing-dbrx-new-state-art-open-llm`.

[5] Y. Du, C. Durkan, R. Strudel, J. B. Tenenbaum, S. Dieleman, R. Fergus, J. Sohl-Dickstein, A. Doucet, and W. S. Grathwohl. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc. In *International conference on machine learning*, pages 8489–8510. PMLR, 2023.

[6] N. Dziri, X. Lu, M. Sclar, X. L. Li, L. Jiang, B. Y. Lin, S. Welleck, P. West, C. Bhagavatula, R. Le Bras, et al. Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems*, 36, 2024.

[7] B. Esmaeili, H. Wu, S. Jain, A. Bozkurt, N. Siddharth, B. Paige, D. H. Brooks, J. Dy, and J.-W. van de Meent. Structured disentangled representations, 2018.

[8] D. Fan, B. Messmer, and M. Jaggi. Towards an empirical understanding of moe design choices. *arXiv preprint arXiv:2402.13089*, 2024.

[9] J. A. Fodor and Z. W. Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71, 1988.

[10] A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. de las Casas, E. B. Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L. R. Lavaud, L. Saulnier, M.-A. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T. L. Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mixtral of experts, 2024.

[11] Y. Jiang, B. Neyshabur, H. Mobahi, D. Krishnan, and S. Bengio. Fantastic generalization measures and where to find them, 2019.

[12] N. Lee, K. Sreenivasan, J. D. Lee, K. Lee, and D. Papailiopoulos. Teaching arithmetic to small transformers, 2023.

[13] E. Leivada, E. Murphy, and G. Marcus. Dall· e 2 fails to reliably capture common syntactic processes. *Social Sciences & Humanities Open*, 8(1):100648, 2023.

[14] O. Lieber, B. Lenz, H. Bata, G. Cohen, J. Osin, I. Dalmedigos, E. Safahi, S. Meirom, Y. Belinkov, S. Shalev-Shwartz, O. Abend, R. Alon, T. Asida, A. Bergman, R. Glozman, M. Gokhman, A. Manevich, N. Ratner, N. Rozen, E. Shwartz, M. Zusman, and Y. Shoham. Jamba: A hybrid transformer-mamba language model, 2024.

[15] S. Mittal, Y. Bengio, and G. Lajoie. Is a modular architecture enough? *Advances in Neural Information Processing Systems*, 35:28747–28760, 2022.

[16] M. L. Montero, C. J. Ludwig, R. P. Costa, G. Malhotra, and J. Bowers. The role of disentanglement in generalisation. In *International Conference on Learning Representations*, 2020.

[17] M. Okawa, E. S. Lubana, R. Dick, and H. Tanaka. Compositional abilities emerge multiplicatively: Exploring diffusion models on a synthetic task. *Advances in Neural Information Processing Systems*, 36, 2024.

[18] OpenAI, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, R. Avila, I. Babuschkin, S. Balaji, V. Balcom, P. Baltescu, H. Bao, M. Bavarian, J. Belgum, I. Bello, J. Berdine, G. Bernadett-Shapiro, C. Berner, L. Bogdonoff, O. Boiko, M. Boyd, A.-L. Brakman, G. Brockman, T. Brooks, M. Brundage, K. Button, T. Cai, R. Campbell, A. Cann, B. Carey, C. Carlson, R. Carmichael, B. Chan, C. Chang, F. Chantzis, D. Chen, S. Chen, R. Chen, J. Chen, M. Chen, B. Chess, C. Cho, C. Chu, H. W. Chung, D. Cummings, J. Currier, Y. Dai, C. Decareaux, T. Degry, N. Deutsch, D. Deville, A. Dhar, D. Dohan, S. Dowling, S. Dunning, A. Ecoffet, A. Eleti, T. Eloundou, D. Farhi, L. Fedus, N. Felix, S. P. Fishman, J. Forte, I. Fulford, L. Gao, E. Georges, C. Gibson,

V. Goel, T. Gogineni, G. Goh, R. Gontijo-Lopes, J. Gordon, M. Grafstein, S. Gray, R. Greene, J. Gross, S. S. Gu, Y. Guo, C. Hallacy, J. Han, J. Harris, Y. He, M. Heaton, J. Heidecke, C. Hesse, A. Hickey, W. Hickey, P. Hoeschele, B. Houghton, K. Hsu, S. Hu, X. Hu, J. Huizinga, S. Jain, S. Jain, J. Jang, A. Jiang, R. Jiang, H. Jin, D. Jin, S. Jomoto, B. Jonn, H. Jun, T. Kaftan, ÅĄukasz Kaiser, A. Kamali, I. Kanitscheider, N. S. Keskar, T. Khan, L. Kilpatrick, J. W. Kim, C. Kim, Y. Kim, J. H. Kirchner, J. Kiros, M. Knight, D. Kokotajlo, ÅĄukasz Kondraciuk, A. Kondrich, A. Konstantinidis, K. Kosic, G. Krueger, V. Kuo, M. Lampe, I. Lan, T. Lee, J. Leike, J. Leung, D. Levy, C. M. Li, R. Lim, M. Lin, S. Lin, M. Litwin, T. Lopez, R. Lowe, P. Lue, A. Makanju, K. Malfacini, S. Manning, T. Markov, Y. Markovski, B. Martin, K. Mayer, A. Mayne, B. McGrew, S. M. McKinney, C. McLeavey, P. McMillan, J. McNeil, D. Medina, A. Mehta, J. Menick, L. Metz, A. Mishchenko, P. Mishkin, V. Monaco, E. Morikawa, D. Mossing, T. Mu, M. Murati, O. Murk, D. MÃľly, A. Nair, R. Nakano, R. Nayak, A. Neelakantan, R. Ngo, H. Noh, L. Ouyang, C. O'Keefe, J. Pachocki, A. Paino, J. Palermo, A. Pantuliano, G. Parascandolo, J. Parish, E. Parparita, A. Passos, M. Pavlov, A. Peng, A. Perelman, F. de Avila Belbute Peres, M. Petrov, H. P. de Oliveira Pinto, Michael, Pokorny, M. Pokrass, V. H. Pong, T. Powell, A. Power, B. Power, E. Proehl, R. Puri, A. Radford, J. Rae, A. Ramesh, C. Raymond, F. Real, K. Rimbach, C. Ross, B. Rotsted, H. Roussez, N. Ryder, M. Saltarelli, T. Sanders, S. Santurkar, G. Sastry, H. Schmidt, D. Schnurr, J. Schulman, D. Selsam, K. Sheppard, T. Sherbakov, J. Shieh, S. Shoker, P. Shyam, S. Sidor, E. Sigler, M. Simens, J. Sitkin, K. Slama, I. Sohl, B. Sokolowsky, Y. Song, N. Staudacher, F. P. Such, N. Summers, I. Sutskever, J. Tang, N. Tezak, M. B. Thompson, P. Tillet, A. Tootoonchian, E. Tseng, P. Tuggle, N. Turley, J. Tworek, J. F. C. Uribe, A. Vallone, A. Vijayvergiya, C. Voss, C. Wainwright, J. J. Wang, A. Wang, B. Wang, J. Ward, J. Wei, C. Weinmann, A. Welihinda, P. Welinder, J. Weng, L. Weng, M. Wiethoff, D. Willner, C. Winter, S. Wolrich, H. Wong, L. Workman, S. Wu, J. Wu, M. Wu, K. Xiao, T. Xu, S. Yoo, K. Yu, Q. Yuan, W. Zaremba, R. Zellers, C. Zhang, M. Zhang, S. Zhao, T. Zheng, J. Zhuang, W. Zhuk, and B. Zoph. Gpt-4 technical report, 2024.

[19] Qwen. Qwen1.5-moe: Matching 7b model performance with 1/3 activated parameters", February 2024.

[20] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2023.

[21] J. Raven and a. Raven. *Uses and Abuses of Intelligence: Studies Advancing Spearman and RavenâĂŹs Quest for Non-Arbitrary Metrics.* 04 2008.

[22] SambaNova. Samba-coe v0.3: The power of routing ml models at scale. `https://sambanova.ai/blog/samba-coe-the-power-of-routing-ml-models-at-scale`.

[23] R. Schaeffer, B. Miranda, and S. Koyejo. Are emergent abilities of large language models a mirage?, 2023.

[24] S. Schug, S. Kobayashi, Y. Akram, J. Sacramento, and R. Pascanu. Attention as a hypernetwork, 2024.

[25] S. Schug, S. Kobayashi, Y. Akram, M. Wołczyk, A. Proca, J. Von Oswald, R. Pascanu, J. Sacramento, and A. Steger. Discovering modular solutions that generalize compositionally. *arXiv preprint arXiv:2312.15001*, 2023.

[26] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[27] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[28] xAI. Open release of grok-1. `https://x.ai/blog/grok-os`.

[29] D. Yu, S. Kaur, A. Gupta, J. Brown-Cohen, A. Goyal, and S. Arora. Skill-mix: A flexible and expandable family of evaluations for ai models. *arXiv preprint arXiv:2310.17567*, 2023.

[30] S. Zhao, H. Ren, A. Yuan, J. Song, N. Goodman, and S. Ermon. Bias and generalization in deep generative models: An empirical study. *Advances in Neural Information Processing Systems*, 31, 2018.

# A Appendix / supplemental material

## A.1 Related Works

**Compositional Generalization** Compositional generalization refers to the ability of a system to understand and produce novel combinations of finite number of familiar elements [9, 2]. This capability is crucial for efficient learning and robust generalization across various domains. In the field of computer vision, studies have focused on generating images from novel concept combinations, often using disentangled representation learning [7]. Researchers have evaluated VAE-based generative models on compositional tasks [30, 16], exploring the relationship between disentanglement and generalization performance in image reconstruction and generation. Recent studies have made significant strides by conducting a controlled investigation of compositional generalization in conditional diffusion models [17, 5, 13], revealing insights into the emergence of compositional abilities and the factors influencing out-of-distribution generation. For large language models (LLMs), recent works have observed emergent compositional capabilities on natural languages [27, 29, 23]. A bunch of evaluation methods have been proposed to quantify the compositional generalization, include imposing generation constraints [29], multi-hop question answering [6], and elementary math operations [12]. Theoretical work has also made strides in understanding the conditions necessary for achieving compositional generalization in neural networks [24, 25, 1]. However, despite the breadth of research in this area, there remains a notable gap in studying compositional generalization within the context of modular models, particularly Sparse Mixture of Experts models. Our work pioneered on studying this relationship, especially on empirically understanding if sparse activation is still optimal in the context of compositional tasks with higher and higher difficulty.

**SRAVEN symbolic reasoning test** Inspired by Raven Progressive Matrices (RAVEN) Test [21], a human Intelligence Quotient test on abstract reasoning, [24] proposed the symbolic compositional SRAVEN test that requires a model to learn the composition of arbitrarily sampled rules by searching through a large number of possible hypotheses. Similar to RAVEN, each SRAVEN task is a $3 \times 3$ grid and the model is asked to query the final panel on the grid given the information from the first 8 panels. Each panel is a vector of length $M$, where each entry on the vector corresponds to a different rule sampled from $R = 8$ possible rules. Therefore, each task is composed by a finite set of rule combinations, and the prediction will be marked as correct only if the model predict all the entries of the final vector. More detailed description can be found in Section 4 in [24]. In this paper, we trained SMoE-based transformers on SRAVEN task due to its compositional nature and the flexibility on tuning the difficulty of the task.

**Skill-Mix** Skill-Mix [29] is a novel evaluation method for assessing language models' compositional abilities. It challenges models to generate short text pieces combining random sets of $k$ skills out of $N$ number of linguistic skills within a given topic. Therefore, the test's difficulty increases with $k$. A Grader model (e.g., GPT-4 [18]) is used to evaluate the generated outputs based on skill application, topic relevance, length, and coherence. In this paper, we tested SMoE-based LLMs on Skill-Mix with varying number of $k$. More experimental and grading details can be found in Section C in [29].

## A.2 Training SMoE-based Transformers on SRAVEN task

### A.2.1 Experiment Implentation Details

```
class MlpBlock(nn.Module):
    hidden_dim: int
    dropout_rate: float

    @nn.compact
    def __call__(self, inputs, deterministic):
        x = nn.Dense(self.hidden_dim)(inputs)
        x = nn.gelu(x)
        x = nn.Dropout(rate=self.dropout_rate)(x, deterministic=
            deterministic)
        x = nn.Dense(inputs.shape[-1])(x)
        x = nn.Dropout(rate=self.dropout_rate)(x, deterministic=
            deterministic)
        return x

```

```python
class MoEBlock(nn.Module):
    num_experts: int
    num_experts_per_tok: int
    hidden_dim: int
    dropout_rate: float

    def setup(self):
        self.experts = [MlpBlock(self.hidden_dim, self.dropout_rate)
            for _ in range(self.num_experts)]
        self.gate = nn.Dense(self.num_experts, use_bias=False)

    def __call__(self, x, deterministic=False):
        orig_shape = x.shape
        x = x.reshape(-1, x.shape[-1])
        scores = self.gate(x)
        expert_weights, expert_indices = jax.lax.top_k(scores, self.
            num_experts_per_tok)
        expert_weights = jax.nn.softmax(expert_weights, axis=-1)
        # Prepare inputs for each expert
        x_expanded = jnp.repeat(x[:, None, :], self.
            num_experts_per_tok, axis=1)
        print('x_expanded shape:', x_expanded.shape)
        # Define a function to apply experts
        def apply_experts(inputs, indices):
            outputs = jnp.zeros_like(inputs)
            for i in range(self.num_experts):
                mask = indices == i
                expert_input = jnp.where(mask[:, :, None], inputs,
                    0.0)
                expert_output = self.experts[i](expert_input,
                    deterministic=deterministic)
                outputs = jnp.where(mask[:, :, None], expert_output,
                    outputs)
            return outputs
        # Apply experts
        expert_outputs = apply_experts(x_expanded, expert_indices)
        print('Expert outputs shape:', expert_outputs.shape)
        # Weight and sum expert outputs
        y = jnp.sum(expert_outputs * expert_weights[..., None], axis
            =1)
        return y.reshape(orig_shape)
```

Code Listing 1: Implementation of SMoE block.

### A.2.2 Experimental Results Details

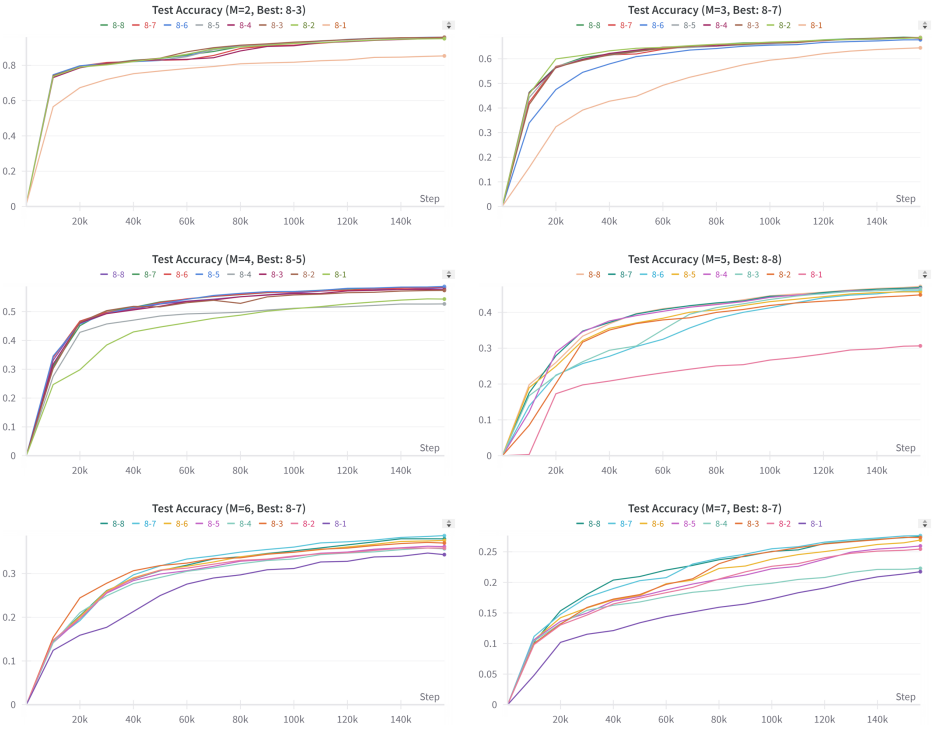


Figure 2: Test Accuracy of training SMoE Transformer with softmax attention.

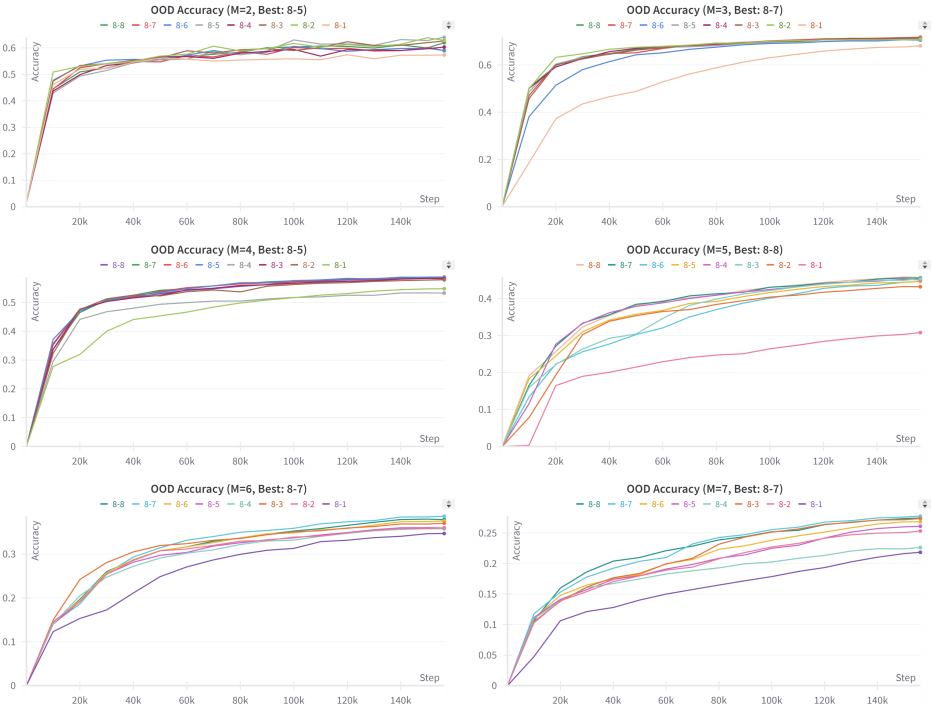


Figure 3: OOD Accuracy of training SMoE Transformer with softmax attention.

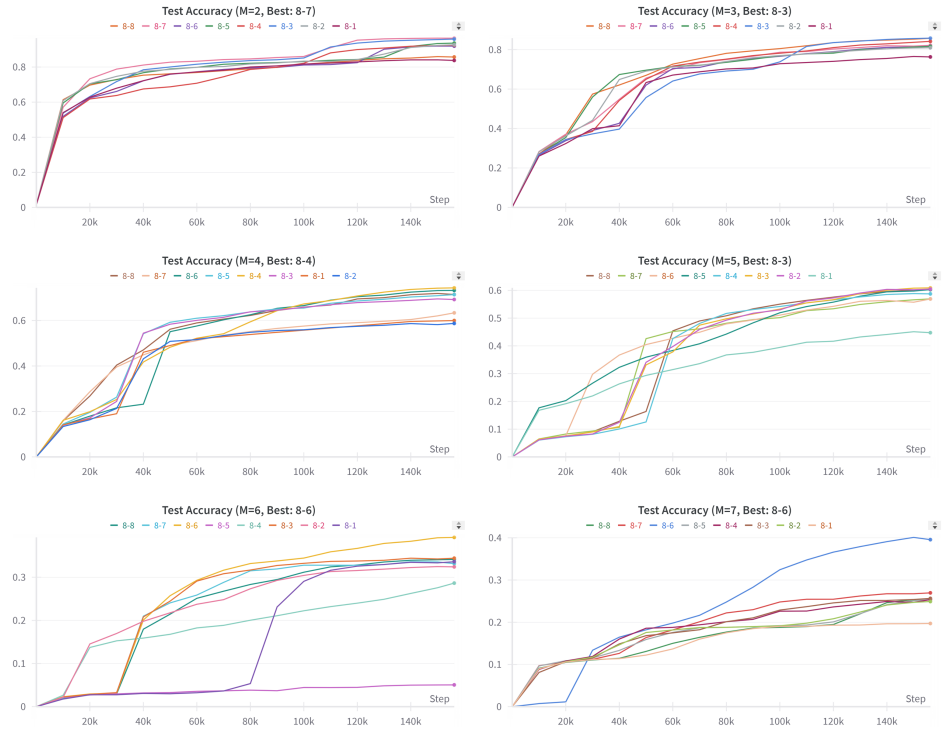


Figure 4: Test Accuracy of training SMoE Transformer with hypernetwork linear attention.

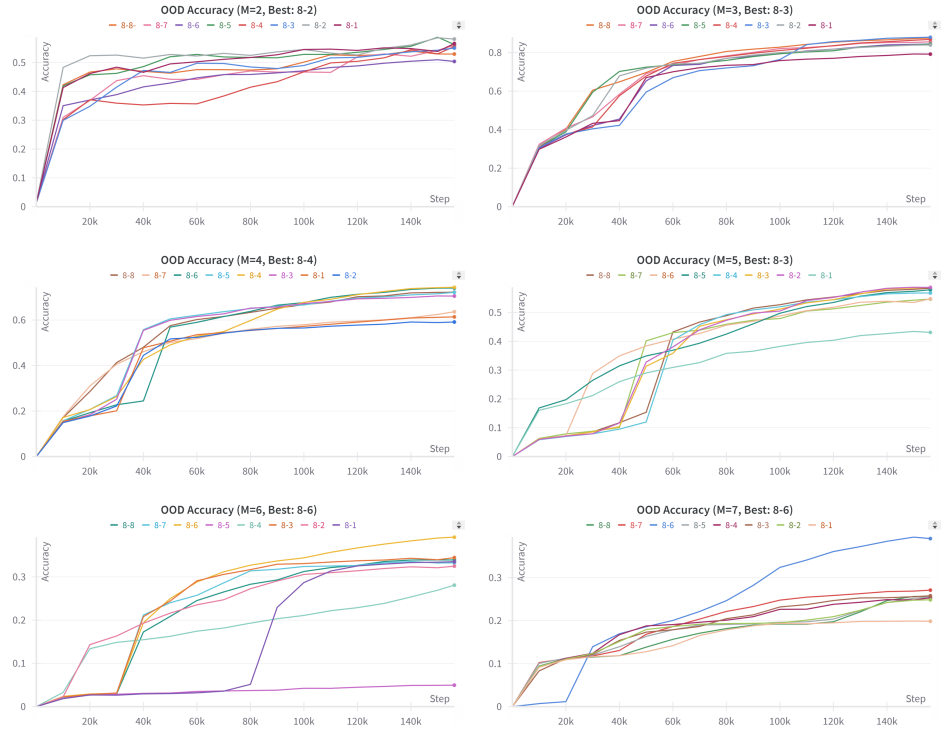


Figure 5: OOD Accuracy of training SMoE Transformer with hypernetwork linear attention.

### A.3 Evaluating SMoE-based Large Language Models on Skill-Mix

We copied the grading metrics definitions from [29] as a reference for the readers. Note that the first three grading metrics are very tough and most models will earn very few points when $k$ increases, as these metrics are conditioned on a specific event.

#### A.3.1 Grading Metrics Definition

Each generated text can receive up to $k+3$ points: 1 point for each correctly illustrated skill, 1 point for sticking to the topic, 1 point for coherence / making sense, and 1 point for having at most $k-1$ sentence. Recall that we grade each generated text three times. In each round of grading, we parse each of the criteria individually from the Grader model's output. For each criterion, we then collect the majority vote among the three grading rounds. The grading metrics are the following:

- *Ratio of Full Marks*: 1 if all $k+3$ points are earned, and 0 otherwise
- *Ratio of All Skills*: 1 if $k$ points are awarded for the $k$ skills and at least 2 points are awarded for the remaining criteria, and 0 otherwise
- *Skill Fraction*: the fraction of points awarded for the $k$ skills if all 3 points are awarded for the remaining criteria, and 0 otherwise
- *Total Score:* sum of the individual points awarded
- *Total Skill Score:* sum of the points awarded for the $k$ skills
- *Rescaled Score:* $\left(\frac{c}{k+3}\right)^{k+3}$ where $c$ is the total score

We then take the maximum value of the metrics among the 3 generations for a given ($k$ skill, 1 topic) combination, and average the maximum value across all the combinations.

### A.4 Full Skill-Mix Results

| Graded by GPT-4 | Skill-Mix ($k$=1) | Skill-Mix ($k$=2) | Skill-Mix ($k$=3) | Skill-Mix ($k$=4) | Skill-Mix ($k$=5) |
|---|---|---|---|---|---|
| ept=1 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.00 ± 0.000 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.00 ± 0.000 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.20 ± 0.082 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.35 ± 0.100 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.00 ± 0.000 |
| ept=2 (default) | 1.00 ± 0.000<br>1.00 ± 0.000<br>1.00 ± 0.000 | 1.00 ± 0.000<br>1.00 ± 0.000<br>1.00 ± 0.000 | 0.00 ± 0.000<br>1.00 ± 0.000<br>0.00 ± 0.000 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.00 ± 0.000 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.00 ± 0.000 |
| ept=3 | 0.00 ± 0.000<br>0.40 ± 0.245<br>0.00 ± 0.000 | 0.00 ± 0.000<br>0.20 ± 0.200<br>0.10 ± 0.100 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.47 ± 0.082 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.60 ± 0.061 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.00 ± 0.000 |
| ept=4 | 0.20 ± 0.200<br>0.80 ± 0.200<br>0.20 ± 0.200 | 0.20 ± 0.200<br>0.40 ± 0.245<br>0.40 ± 0.187 | 0.20 ± 0.200<br>0.20 ± 0.200<br>0.67 ± 0.105 | 0.20 ± 0.200<br>0.20 ± 0.200<br>0.75 ± 0.079 | 0.00 ± 0.000<br>0.20 ± 0.200<br>0.00 ± 0.000 |
| ept=5 | 0.20 ± 0.200<br>0.40 ± 0.245<br>0.20 ± 0.200 | 0.20 ± 0.200<br>0.20 ± 0.200<br>0.30 ± 0.200 | 0.20 ± 0.200<br>0.20 ± 0.200<br>0.53 ± 0.133 | 0.20 ± 0.200<br>0.20 ± 0.200<br>0.65 ± 0.100 | 0.00 ± 0.000<br>0.20 ± 0.200<br>0.00 ± 0.000 |
| ept=6 | 0.00 ± 0.000<br>0.20 ± 0.200<br>0.00 ± 0.000 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.20 ± 0.122 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.47 ± 0.082 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.60 ± 0.061 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.00 ± 0.000 |
| ept=7 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.00 ± 0.000 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.10 ± 0.100 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.40 ± 0.067 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.55 ± 0.050 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.00 ± 0.000 |
| ept=8 | 0.00 ± 0.000<br>0.40 ± 0.245<br>0.00 ± 0.000 | 0.00 ± 0.000<br>0.20 ± 0.200<br>0.10 ± 0.100 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.47 ± 0.082 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.60 ± 0.061 | 0.00 ± 0.000<br>0.00 ± 0.000<br>0.00 ± 0.000 |

Table 2: Skill-Mix Evaluation Results on Mixtral-8×7B Instruct-v0.1 [11]. Note that we only evaluate using Ratio of Full Marks/Ratio of All Skills/Skill Fraction metrics.

| DBRX in 4-bit, graded by GPT4 | k=1 | k=2 | k=3 | k=4 |
|---|---|---|---|---|
| ept=1 | 0.00 ± 0.000 0.00 ± 0.000 0.00 ± 0.000 0.20 ± 0.200 0.00 ± 0.000 0.00 ± 0.001 | 0.00 ± 0.000 0.00 ± 0.000 0.00 ± 0.000 0.40 ± 0.400 0.00 ± 0.000 0.00 ± 0.002 | 0.00 ± 0.000 0.00 ± 0.000 0.00 ± 0.000 1.00 ± 0.316 0.00 ± 0.000 0.00 ± 0.000 | 0.00 ± 0.000 0.00 ± 0.000 0.00 ± 0.000 0.40 ± 0.245 0.00 ± 0.000 0.00 ± 0.000 |
| ept=2 | 0.60 ± 0.245 1.00 ± 0.000 0.60 ± 0.245 3.60 ± 0.245 1.00 ± 0.000 0.73 ± 0.167 | 0.20 ± 0.200 0.40 ± 0.245 0.40 ± 0.187 4.00 ± 0.316 1.40 ± 0.245 0.41 ± 0.155 | 0.00 ± 0.000 0.00 ± 0.000 0.33 ± 0.000 4.00 ± 0.000 1.20 ± 0.200 0.09 ± 0.000 | 0.00 ± 0.000 0.00 ± 0.000 0.15 ± 0.100 4.00 ± 0.447 1.60 ± 0.510 0.04 ± 0.021 |
| ept=3 | 0.80 ± 0.200 1.00 ± 0.000 0.80 ± 0.200 3.80 ± 0.200 1.00 ± 0.000 0.86 ± 0.137 | 0.40 ± 0.245 0.60 ± 0.245 0.70 ± 0.122 4.40 ± 0.245 1.60 ± 0.245 0.60 ± 0.165 | 0.00 ± 0.000 0.00 ± 0.000 0.60 ± 0.067 4.80 ± 0.200 1.80 ± 0.200 0.29 ± 0.049 | 0.00 ± 0.000 0.00 ± 0.000 0.35 ± 0.061 4.40 ± 0.245 1.40 ± 0.245 0.05 ± 0.018 |
| ept=4 (default setting) | 1.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 4.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 | 0.40 ± 0.245 0.40 ± 0.245 0.70 ± 0.122 4.40 ± 0.245 1.40 ± 0.245 0.60 ± 0.165 | 0.00 ± 0.000 0.00 ± 0.000 0.60 ± 0.067 4.80 ± 0.200 1.80 ± 0.200 0.29 ± 0.049 | 0.00 ± 0.000 0.00 ± 0.000 0.40 ± 0.061 4.60 ± 0.245 1.60 ± 0.245 0.06 ± 0.018 |
| ept=5 | 1.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 4.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 | 0.60 ± 0.245 0.80 ± 0.200 0.80 ± 0.122 4.60 ± 0.245 1.80 ± 0.200 0.73 ± 0.165 | 0.20 ± 0.200 0.20 ± 0.200 0.67 ± 0.105 5.00 ± 0.316 2.00 ± 0.316 0.42 ± 0.153 | 0.00 ± 0.000 0.00 ± 0.000 0.45 ± 0.094 4.80 ± 0.374 1.80 ± 0.374 0.11 ± 0.059 |
| ept=6 | 0.80 ± 0.200 0.80 ± 0.200 0.80 ± 0.200 3.60 ± 0.400 0.80 ± 0.200 0.81 ± 0.187 | 0.40 ± 0.245 0.40 ± 0.245 0.70 ± 0.122 4.40 ± 0.245 1.40 ± 0.245 0.60 ± 0.165 | 0.20 ± 0.200 0.20 ± 0.200 0.60 ± 0.163 5.00 ± 0.316 2.20 ± 0.200 0.42 ± 0.153 | 0.00 ± 0.000 0.00 ± 0.000 0.50 ± 0.000 5.00 ± 0.000 2.00 ± 0.000 0.09 ± 0.000 |
| ept=7 | 1.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 4.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 | 0.40 ± 0.245 0.40 ± 0.245 0.70 ± 0.122 4.40 ± 0.245 1.60 ± 0.245 0.60 ± 0.165 | 0.20 ± 0.200 0.20 ± 0.200 0.67 ± 0.105 5.00 ± 0.316 2.00 ± 0.316 0.42 ± 0.153 | 0.00 ± 0.000 0.00 ± 0.000 0.65 ± 0.061 5.60 ± 0.245 2.60 ± 0.245 0.24 ± 0.060 |
| ept=8 | 0.80 ± 0.200 1.00 ± 0.000 0.80 ± 0.200 3.80 ± 0.200 1.00 ± 0.000 0.86 ± 0.137 | 0.20 ± 0.200 0.20 ± 0.200 0.60 ± 0.100 4.20 ± 0.200 1.20 ± 0.200 0.46 ± 0.134 | 0.20 ± 0.200 0.20 ± 0.200 0.67 ± 0.105 5.00 ± 0.316 2.00 ± 0.316 0.42 ± 0.153 | 0.00 ± 0.000 0.00 ± 0.000 0.50 ± 0.112 5.00 ± 0.447 2.00 ± 0.447 0.16 ± 0.074 |
| ept=9 | 0.80 ± 0.200 1.00 ± 0.000 0.80 ± 0.200 3.80 ± 0.200 1.00 ± 0.000 0.86 ± 0.137 | 0.40 ± 0.245 0.80 ± 0.200 0.60 ± 0.187 4.20 ± 0.374 1.80 ± 0.200 0.55 ± 0.191 | 0.20 ± 0.200 0.40 ± 0.245 0.67 ± 0.105 5.00 ± 0.316 2.20 ± 0.374 0.42 ± 0.153 | 0.00 ± 0.000 0.00 ± 0.000 0.55 ± 0.050 5.20 ± 0.200 2.40 ± 0.245 0.14 ± 0.049 |
| ept=10 | 1.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 4.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 | 0.20 ± 0.200 0.60 ± 0.245 0.60 ± 0.100 4.20 ± 0.200 1.60 ± 0.245 0.46 ± 0.134 | 0.00 ± 0.000 0.00 ± 0.000 0.53 ± 0.082 4.60 ± 0.245 1.60 ± 0.245 0.24 ± 0.061 | 0.00 ± 0.000 0.00 ± 0.000 0.40 ± 0.061 4.60 ± 0.245 1.60 ± 0.245 0.06 ± 0.018 |
| ept=11 | 1.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 4.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 | 0.60 ± 0.245 0.60 ± 0.245 0.80 ± 0.122 4.60 ± 0.245 1.60 ± 0.245 0.73 ± 0.165 | 0.20 ± 0.200 0.20 ± 0.200 0.47 ± 0.170 4.60 ± 0.400 2.00 ± 0.316 0.32 ± 0.177 | 0.00 ± 0.000 0.00 ± 0.000 0.55 ± 0.094 5.20 ± 0.374 2.40 ± 0.400 0.18 ± 0.068 |
| ept=12 | 0.80 ± 0.200 1.00 ± 0.000 0.80 ± 0.200 3.80 ± 0.200 1.00 ± 0.000 0.86 ± 0.137 | 0.60 ± 0.245 0.60 ± 0.245 0.80 ± 0.122 4.60 ± 0.245 1.60 ± 0.245 0.73 ± 0.165 | 0.00 ± 0.000 0.20 ± 0.200 0.40 ± 0.163 4.40 ± 0.400 1.80 ± 0.490 0.22 ± 0.070 | 0.20 ± 0.200 0.20 ± 0.200 0.70 ± 0.122 5.80 ± 0.490 2.80 ± 0.490 0.41 ± 0.160 |
| ept=13 | 1.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 4.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 | 0.40 ± 0.245 0.40 ± 0.245 0.60 ± 0.187 4.20 ± 0.374 1.20 ± 0.374 0.55 ± 0.191 | 0.40 ± 0.245 0.60 ± 0.245 0.67 ± 0.149 5.20 ± 0.374 2.40 ± 0.400 0.55 ± 0.189 | 0.00 ± 0.000 0.00 ± 0.000 0.50 ± 0.079 5.00 ± 0.316 2.40 ± 0.510 0.13 ± 0.055 |
| ept=14 | 1.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 4.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 | 0.60 ± 0.245 0.60 ± 0.245 0.60 ± 0.245 4.20 ± 0.490 1.80 ± 0.200 0.63 ± 0.226 | 0.00 ± 0.000 0.00 ± 0.000 0.53 ± 0.082 4.60 ± 0.245 1.60 ± 0.245 0.24 ± 0.061 | 0.00 ± 0.000 0.00 ± 0.000 0.50 ± 0.079 5.20 ± 0.200 2.40 ± 0.245 0.14 ± 0.049 |
| ept=15 | 1.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 4.00 ± 0.000 1.00 ± 0.000 1.00 ± 0.000 | 0.20 ± 0.200 0.60 ± 0.245 0.40 ± 0.187 4.20 ± 0.200 1.60 ± 0.245 0.46 ± 0.134 | 0.00 ± 0.000 0.00 ± 0.000 0.53 ± 0.082 4.60 ± 0.245 1.80 ± 0.200 0.24 ± 0.061 | 0.00 ± 0.000 0.00 ± 0.000 0.40 ± 0.061 4.60 ± 0.245 1.60 ± 0.245 0.06 ± 0.018 |
| ept=16 | 0.80 ± 0.200 0.80 ± 0.200 0.80 ± 0.200 3.80 ± 0.200 1.00 ± 0.000 0.86 ± 0.137 | 0.20 ± 0.200 0.40 ± 0.245 0.50 ± 0.158 4.00 ± 0.316 1.40 ± 0.245 0.41 ± 0.155 | 0.20 ± 0.200 0.40 ± 0.245 0.47 ± 0.170 4.60 ± 0.510 2.00 ± 0.447 0.35 ± 0.174 | 0.00 ± 0.000 0.00 ± 0.000 0.25 ± 0.112 4.00 ± 0.447 1.40 ± 0.510 0.04 ± 0.021 |

Figure 6: Skill-Mix Evaluation Results on DBRX-instruct [4]. Note that we use all available grading metrics.