Nexus: Specialization meets Adaptability for Efficiently Training Mixture of Experts

Nikolas Gritsch[1,2*], Qizhen Zhang[3], Acyr Locatelli[2], Sara Hooker[1], Ahmet Üstün[1]

[1]Cohere For AI [2]Cohere [3]University of Oxford

*Work done as part of the Research Scholar Program

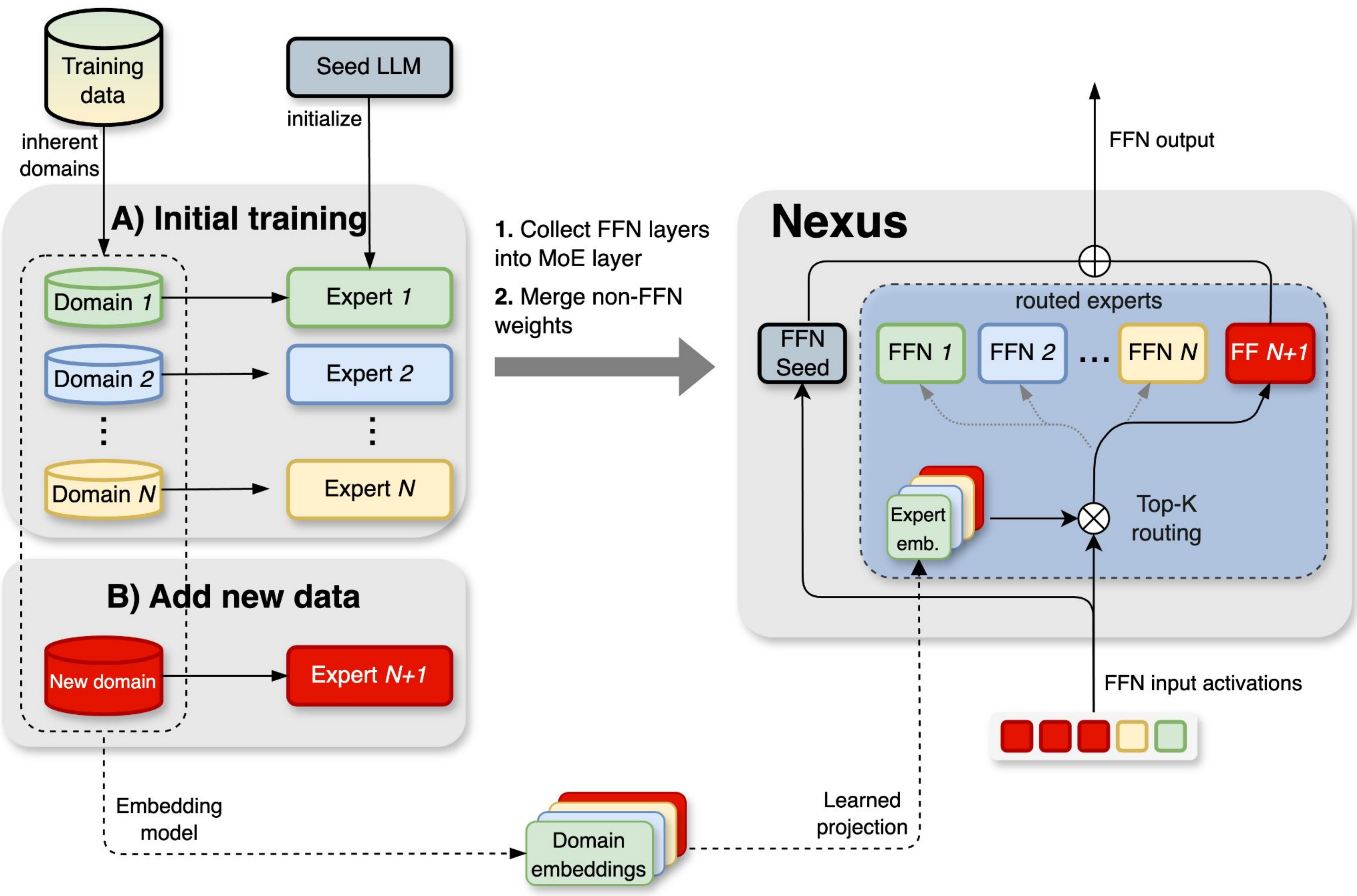
arxiv.org/abs/2408.15901

Cohere For AI

Mixture of Experts from Specialized LMs

? How can we best upcycle specialized dense models into an MoE model?

? How can an MoE router adapt to new experts after the initial training?



Current MoEs are limited in different ways:

	Nexus	BTM[1]	BTX[2]	MoE
Experts trained independently	✅	✅	✅	❌
Experts are specialized	✅	✅	✅	❌
Learned routing	✅	❌	✅	✅
New experts can be added	✅	❌	❌	❌

Contributions of Nexus

💪 **Efficient** parallel, asynchronous expert training

💪 Experts truly **specialized** on individual domains

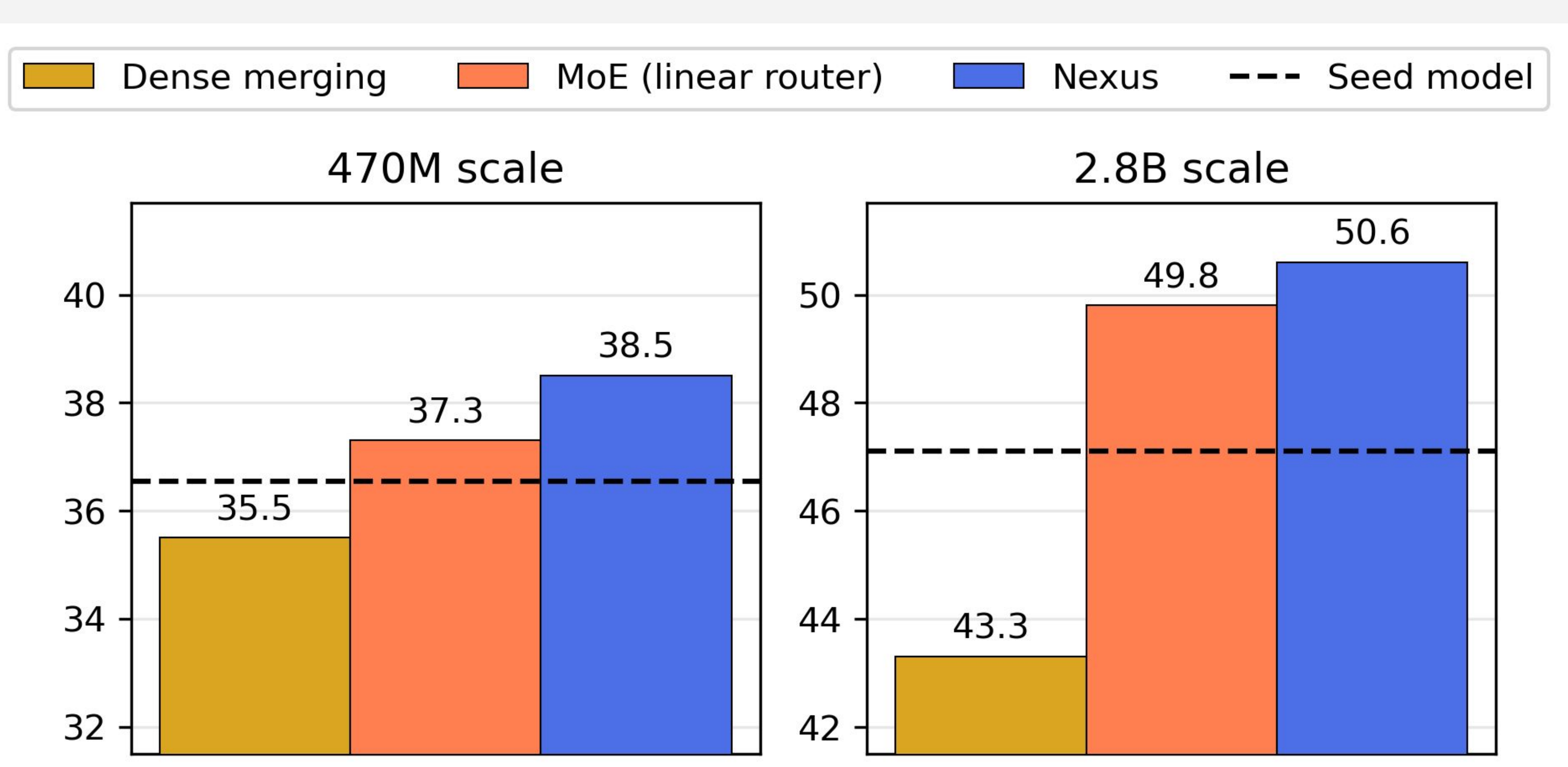
💪 **Adapt** to new experts after initial training without catastrophic forgetting

Our vision: In an ecosystem with many open-source finetunes of the same base model (e.g. Llama 3), use Nexus to quickly assemble your personalized MoE, and extend it anytime with new domains!

Methodology

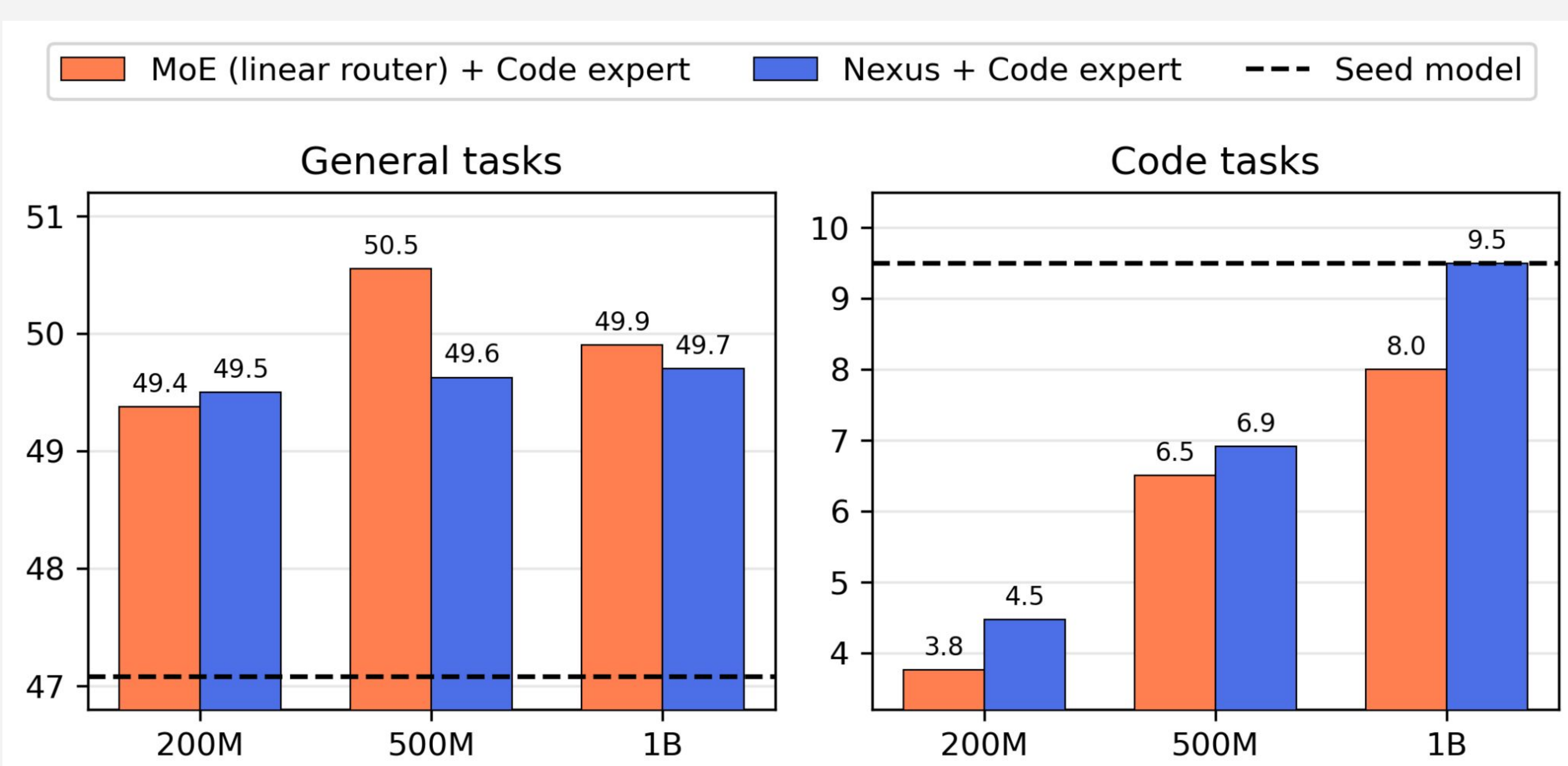
- Train *n* dense experts (initialized from same pretrained model) on different datasets/domains (e.g. ArXiv, Books, C4, Wikipedia)
- Convert the dense models to a single Nexus model by **stacking** the dense model MLPs into an MoE layer and **averaging** all other params
- How do we know when to route to each expert?
  1. Baseline (BTX): train a router (linear proj.) for 40B tokens
  2. Nexus: use **expert training data embeddings** as informative prior! They capture the "knowledge" each expert has. Train for 40B tokens to learn a projection from data embedding space to model latent space, then **route by choosing the most similar embedding** to a token's latent representation

Nexus beats BTM and BTX for pre-training:



- 4 experts are initialized from a pretrained model and trained for 40B tokens on ArXiv, Books, C4, and Wikipedia
- Upcycling with Nexus outperforms both BTX and a full model averaging baseline on Knowledge, Science, Reasoning and MMLU downstream tasks (all compute and data matched)
- Nexus also beats training a dense model with the data/compute of all experts!

Nexus adapts better to new experts:



- For the new expert, a dense model is trained for 40B tokens on a new domain (Code), appended to the Nexus expert layer, and fine-tuned on all data for budgets of 200M/500M/1B tokens
- Nexus **outperforms** BTX on the new domain, and the gap increases with more finetuning

Q: Does Nexus add overhead for training/inference?

A: No, same complexity as vanilla MoE!

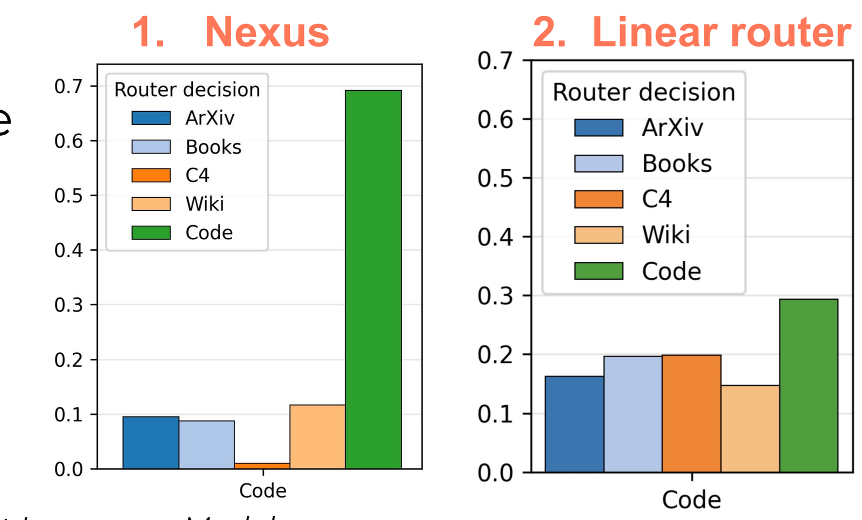
- Training: less than 1% additional parameters
- Inference: 0% overhead as expert embeddings can be precomputed!
  - Intuition: learned projection is a **hypernetwork** that computes the router weights, using the dataset embeddings as input. Only need to recompute when set of expert changes

Q: Is the routing in Nexus truly specialized?

A: Yes!

- Nexus assigns tokens more often to the expert specialized on that domain
- Comparison of **routing distributions for code tokens**:

1. Nexus 2. Linear router



References

[1] Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, & Luke Zettlemoyer. (2022). *Branch-Train-Merge: Embarrassingly Parallel Training of Expert Language Models.*

[2] Sainbayar Sukhbaatar, Olga Golovneva, Vasu Sharma, Hu Xu, Xi Victoria Lin, Baptiste Rozière, Jacob Kahn, Daniel Li, Wen-tau Yih, Jason Weston, & Xian Li. (2024). *Branch-Train-MiX: Mixing Expert LLMs into a Mixture-of-Experts LLM.*

[3] Suchin Gururangan, Margaret Li, Mike Lewis, Weijia Shi, Tim Althoff, Noah A. Smith, & Luke Zettlemoyer. (2023). *Scaling Expert Language Models with Unsupervised Domain Discovery.*

*Special thanks to **John Lin**, **Tim Chung, Sylvie Shi, Arkady Arkhangorodsky, David Cairuz, Felipe Cruz Salinas, Milad Alizadeh, James Owers-Bardsley, and Viraat Aryabumi** for their support!*

nikolasgritsch@cohere.com