Router Upcycling: Leveraging Mixture-of-Routers in Mixture-of-Experts Upcycling

Anonymous ACL submission

Abstract

The Mixture-of-Experts (MoE) models have gained significant attention in deep learning due to their dynamic resource allocation and superior performance across diverse tasks. However, efficiently training these models remains challenging. The MoE upcycling technique has been proposed to reuse and improve existing model components, thereby minimizing training overhead. Despite this, simple routers, such as linear routers, often struggle with complex routing tasks within MoE upcycling. In response, we propose a novel routing technique called Router Upcycling to enhance the performance of MoE upcycling models. Our approach initializes multiple routers from the attention heads of the preceding attention layer during upcycling. These routers collaboratively assign tokens to specialized experts in an attention-like manner. Each token is processed into diverse queries and aligned with the experts' features (serving as keys). Experimental results demonstrate that our method achieves state-of-the-art (SOTA) performance, outperforming other upcycling baselines. The code will be released on GitHub upon acceptance.

1 Introduction

The Mixture-of-Experts (MoE) model has emerged as a powerful paradigm in deep learning, enabling dynamic allocation of computational resources and achieving remarkable performance across a variety of tasks (Jacobs et al., 1991; Shazeer et al., 2017; Jiang et al., 2024). By combining sets of expert networks and gating routers where input data is assigned to the most appropriate experts, MoE models effectively capture diverse patterns within data, enhancing both efficiency and accuracy.

The convergence and generalization capabilities of MoE models heavily depend on the design of the routing strategy (Shazeer et al., 2017), as poor routers lead to the overtraining of some experts and the under-training of others. The commonly used dynamic routing method in MoE is to select the expert with the top-k highest scores based on a probability distribution output by an intermediate (mostly linear) layer with learnable parameters that act as the router. However, routing frameworks designed for vanilla MoE models (Shazeer et al., 2017) may fall short of newly emerged experts' evolution. For instance, upcycling (Komatsuzaki et al., 2023) is proposed as a popular approach that initializes experts from dense checkpoints and outperforms continued dense model training while reducing MoE training costs (He et al., 2024). Our preliminary experiments, as shown in Figure 1, present that leveraging inappropriate router structures in upcycling, such as upcycling with Switch Transformer (Fedus et al., 2022) routers, leads to poor token assignment to the trained upcycled experts, negatively impacting the performance of MoE upcycling models. Therefore, exploring appropriate router structures is a crucial field for MoE upcycling. However, building and initializing efficient routers remains yet to be explored in MoE upcy-

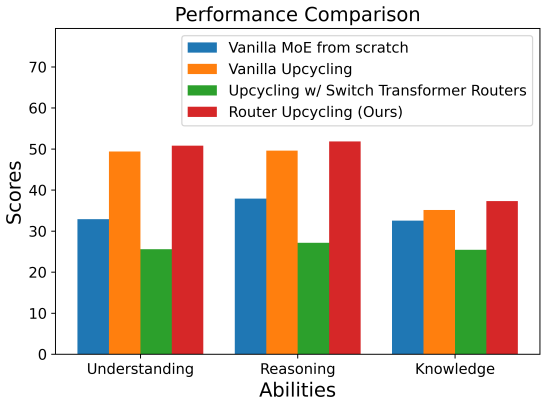

Figure 1: Performance Comparison between vanilla MoE from scratch and various upcycling models with different routers on several benchmarks in Section 4.2. Notably, Switch Transformer (Fedus et al., 2022) routers perform much poorly in MoE upcycling.

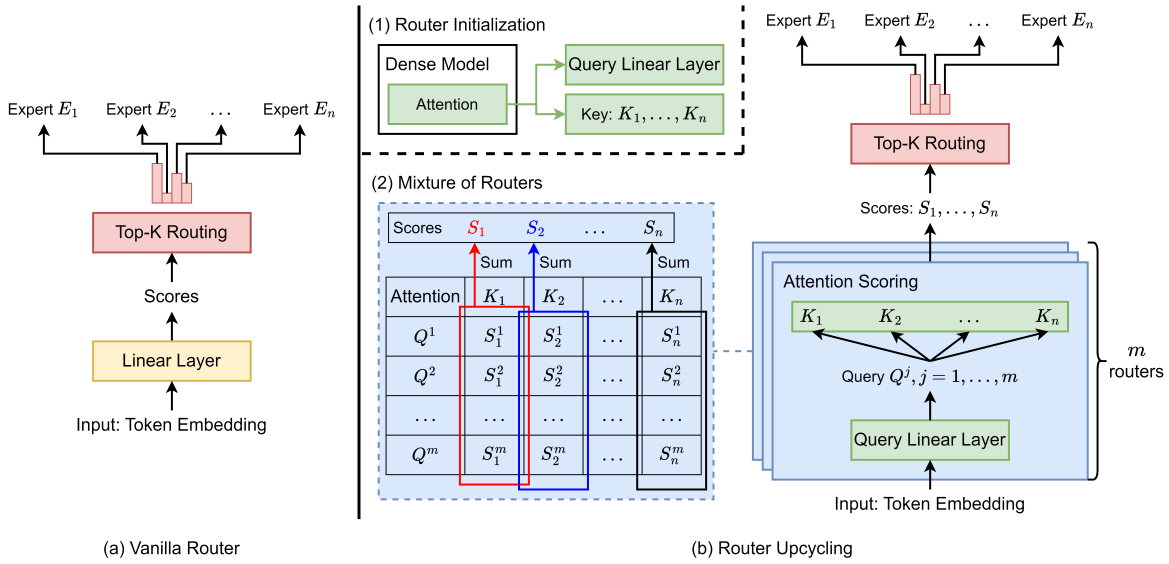

Figure 2: (a) Vanilla router (Shazeer et al., 2017). (b) Our proposed Router Upcycling method.

cling. In this paper, we comprehensively examine the performance of previous router designs in MoE upcycling and propose a novel routing method for upcycling. To our knowledge, we are the first to do router optimization in the Upcycling scenario.

Before we push further, we need to figure out why previous routers fail. Previous studies (Dai et al., 2022; Li et al., 2024; Roller et al., 2021) make various attempts to enhance the token-choice routing ability, many focusing on the ease of training instability that same or similar token representations are routed to different sets of experts during training (Dai et al., 2022). Nevertheless, most of them are fixed routing methods or simple in structure like the vanilla linear router as shown in Figure 2(a) and cannot inherit diverse token assignments, facing drawbacks such as the representation collapse issue (Chi et al., 2022). On the other hand, expert-choice routing (Zhou et al., 2022) flips the routing paradigm by allowing experts to select the top-k tokens, rather than tokens selecting the top-k experts but falls short in causal language modeling due to the reliance on future tokens. More importantly, while experts are the same at the beginning of the MoE upcycling, different experts always choose the same tokens as input in expert-choice routing, causing expert under-specialization, where assigned tokens are not diverse enough to ensure experts are specialized. To summarize, previous routers fail to implement diverse token assignments, leading to expert under-specialization in MoE upcycling.

We argue that, while experts are all initialized from a well-trained dense checkpoint in vanilla upcycling, a well-organized upcycling framework should also upcycle routers in alignment with upcycled experts. In response, we propose a novel routing method, Router Upcycling, as illustrated in Figure 2(b), that enhances MoE models by introducing a mixture of routers initialized from attention modules in a dense checkpoint. Specifically, each router leverages the query transformation from different attention heads in the preceding attention layer to process each token into diverse Queries to highlight token representations from multiple perspectives. Meanwhile, during an additional pre-training process, each expert's features are initialized as the average Key calculated by corresponding attention heads in the preceding attention layer. These experts' features serve as Keys in routers and are independent of the token representation to ensure training stability. In an attention-like manner, routing scores are calculated as the inner product of the diverse Queries and the experts' Keys, followed by a summation process to obtain the experts' scores, which are utilized for top-k routing afterward.

Intuitively, our proposed routers mutually align token representations and expert features to better route tokens to proper experts. Meanwhile, our proposed router utilizes more parameters than a standard linear router to store historical routing knowledge, enhancing stability and accuracy. Still, its size remains negligible compared to the entire model, making this improvement essentially a "free lunch" for existing models. Most importantly, our upcycling method initializes the MoE model with-

out a random router as before, greatly enhancing the stability of MoE upcycling.

Our method can be integrated into existing MoE models with minimal modifications. We demonstrate the versatility of our approach by implementing it on a Qwen (Qwen, 2024) 0.5B dense model. Despite negligible routing overhead increase, experimental evaluations indicate that our method achieves state-of-the-art performance, outperforming other upcycling baselines. Our model surpasses the upcycling baseline with more than 2% in general and an average of 4% more accuracy on the ARC datasets.

The contribution of the paper is as follows.

- We are the first to do router optimization in the upcycling scenario, recognizing the necessity to upcycle routers and leverage attention modules to initialize them.
- We further expand the idea of upcycling by using Router Upcycling, where the MoE model is initialized without a random router as before, greatly enhancing the stability of MoE upcycling.
- Experimental evaluations indicate that our "free lunch" method achieves state-of-the-art performance, outperforming other upcycling baselines with a more than 2% improvement.

2 Preliminaries

2.1 Mixture-of-Experts Layer

A Mixture-of-Experts (MoE) layer typically consists of a set of n experts $E_1, E_2, \ldots, E_n$ and a router or gating network G. The experts and router work collaboratively, functioning similarly to the Feed-Forward Network (FFN) in dense models. The router uses a gating function to assign tokens to the selected top-k experts $E_s, s \in \mathbb{S}$ based on scores from a probability distribution calculated by an intermediate (mostly linear) layer (Shazeer et al., 2017). Typically, the gating function uses the Softmax function over the product of the input token and a gating weight matrix, routing tokens to the most likely experts.

Let $x \in \mathbb{R}^d$ be the input token representation, W be the intermediate linear layer that outputs an n-dimensional vector, and $G(x)$ and $E_i(x)$ be the outputs of the router and each expert E_i, respectively. The output y of the MoE layer, without normalization, can be written as follows:

$$G(x) = \operatorname{Softmax} W(x). \tag{1}$$

$$y = \sum_{s \in \mathbb{S}} G(x)_s E_s(x). \tag{2}$$

2.2 Upcycling

The vanilla upcycling method (Komatsuzaki et al., 2023) initializes an MoE model from a dense model checkpoint. It converts a dense model into an MoE model by duplicating the FFN weights multiple times and initializing a randomized router:

$$E_1 = E_2 = \cdots = E_n = \text{Dense FFN}\,. \tag{3}$$

3 Router Upcycling

3.1 Overview

The widely-used vanilla linear router (Shazeer et al., 2017) is too simple to handle diverse token assignments, leading to representation collapse (Chi et al., 2022) and expert under-specialization in MoE upcycling. To address this, we introduce attention-like routers for better alignment between tokens and experts, as shown in Figure 2(b), where tokens act as queries and expert features act as keys. Each token is transformed into multiple queries with different representations. For example, one query may represent the syntax of a token, which is then matched with expert features using attention scoring. This approach allows each query from a token to have a different semantic expression in a specific router subspace. By considering multiple queries, our method constructs an equal number of routers, which work with a Mixture-of-Routers mechanism.

This section presents our novel Router Upcycling method for MoE models. Following (Komatsuzaki et al., 2023), our method initializes each expert as a copy of the original dense model's Feed-Forward Networks (FFN), as shown in Equation 3, while keeping the dense model's other parts unchanged. Our upcycled routers employ novel mechanisms:

1. **Multiple Routers Initialization from Attention Layers**: Initializing routers' query transformations and experts' keys from attention heads in the preceding attention layers, highlighting token representations from diverse perspectives.
2. **Mixture-of-Routers Attention Scoring**: Using an attention-like mechanism to compute matching scores between token queries and expert keys, effectively aligning tokens with

experts and incorporating scores from multiple collaborative routers to ensure specialized token routing.

3.2 Multiple Routers Initialization from Attention Layers

This section demonstrates how to initialize the routers using attention heads from the dense model. For each layer, we denote W_Q and K_A as the attention query transformations and average attention keys in the preceding attention layer, respectively. Our method freezes the dense model to collect average attention key representations K_A during certain iterations of an additional pre-training process, which will be elaborated in Section 4.1.

We then concatenate several attention heads containing W_Q and K_A, grouping them into m sets based on the highest similarity of K_A. This process constructs W_Q^C and K_A^C for m concatenated heads, enhancing their representation power with a larger hidden size. We will discuss the optimized settings in Section 5.2.1, where it is shown that the model performs best when the number of routers m equals the number of experts n, both being 8 in each layer.

Next, we use W_Q^C and K_A^C to initialize the token query transformations W and expert keys K in our proposed mixture of m routers:

$$\text{Expert}_i : K_i = (K_A^C)_i, \quad i = 1, \ldots, n, \quad (4)$$

$$\text{Router}^j : W^j = (W_Q^C)^j, \quad j = 1, \ldots, m, \quad (5)$$

The routers transform each token x by projecting it into m low-dimensional subspaces using linear transformations:

$$Q^j = W^j x, \quad j = 1, \ldots, m, \quad (6)$$

where $Q^j \in \mathbb{R}^{d'}$ is the j-th query for token x in the j-th router, and $W^j \in \mathbb{R}^{d' \times d}$ is the projection matrix. Unlike tokens, each expert E_i preserves its unique key embedding $K_i \in \mathbb{R}^{d'}$, independent of the token representation, to maintain its stable feature. Therefore, the number of keys equals the number of experts n in our approach. We demonstrate the decreased performance when each expert preserves multiple keys in Section 5.2.1.

3.3 Mixture-of-Routers Attention Scoring

With the token queries and expert keys built, our method routes tokens in an attention-like manner. Unlike the traditional attention mechanism (Vaswani et al., 2023), these queries are from the same token instead of a sequence of tokens, and the keys, held by experts, are independent of tokens. The attention scoring obtains attention-mapping scores by multiplying each token query Q^j in m routers by each expert key K_i for each Q-K pair, as shown on the right side of Figure 2(b):

$$S_i^j = \frac{Q^{j\top} K_i}{\sqrt{d'}}, i = 1, \ldots, n, j = 1, \ldots, m, \quad (7)$$

where S_i^j represents the matching score between the j-th query of the token and the i-th key of expert E_i. To maintain the diverse amplitude of each low-dimensional Q-K subspace, our method does not use any normalization technique such as the cosine router (Chi et al., 2022).

To incorporate attention-mapping scores from collaborative routers, we sum over the query dimension for each token:

$$S_i = \sum_j S_i^j, \quad i = 1, \ldots, n, j = 1, \ldots, m. \quad (8)$$

Finally, to obtain the top-k routing weights R for expert E_i, these scores are sent to a top-k router:

$$R = \text{Softmax}\, S. \quad (9)$$

By selecting the top-k experts with the highest routing weights R, we assign tokens to the most appropriate experts and use Equation 2 to get the output of the experts.

This mechanism effectively builds and capitalizes on the inter-expert relationships, ensured by the attention mechanism, guaranteeing that tokens are routed based on multiple facets of their representations. This leads to a more fine-grained and precise token allocation. Our method stabilizes MoE training by distributing tokens based on their multi-representation matching scores rather than a singular gating score. This approach also diminishes the likelihood of representation collapse (Chi et al., 2022), as experts specialize in processing tokens that align closely with their key representations across multiple subspaces.

4 Experiments

4.1 Experimental Setup

We verified our router upcycling method on a Qwen 0.5B model (Qwen, 2024) to obtain an 8x0.5B MoE model with approximately 2.1B total parameters and around 0.8B activated parameters. Due

Method	Vanilla MoE from scratch	Vanilla Upcycling	Switch Transformer Upcycling	LocMoE Upcycling	Upcycling w/ MLP routers	**Router Upcycling**
OpenbookQA	31.4	40.8	27.6	38.2	37.6	**42.6**
OpenbookQA-fact	34.4	58.0	23.6	51.4	55.0	**59.0**
ARC-C	27.12	39.66	14.58	29.15	34.92	**42.37**
ARC-E	30.51	52.73	20.63	45.68	51.15	**58.02**
Hellaswag	41.45	49.36	25.24	48.83	49.70	**50.12**
Winogrande	52.57	56.67	48.22	56.12	**58.88**	56.91
BoolQ	48.13	49.88	47.09	54.19	41.44	**55.84**
COPA	60	62	49	61	63	**64**
NQ	4.32	4.85	0.42	4.04	3.96	**5.15**
TriviaQA	17.74	23.82	5.24	22.16	24.17	**24.25**
Understanding Average	32.90	49.40	25.60	44.80	46.30	**50.80**
Reasoning Average	37.91	49.61	27.17	44.95	48.67	**51.86**
Knowledge Average	32.55	35.14	25.44	35.35	33.14	**37.31**
Average	34.76	43.78	26.17	41.08	41.98	**45.83**

Table 1: Performance comparison of different models on benchmark datasets, evaluated with zero-shot schema on every benchmark.

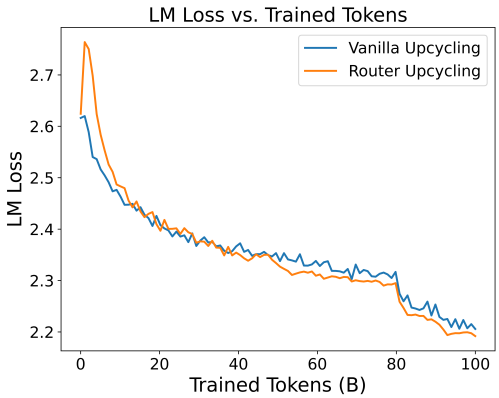

Figure 3: LM losses during training.

to computational budget limitations, we selected this small model as the backbone, one of the best dense models with less than 1B parameters. Our experiments were conducted under the Megatron (Shoeybi et al., 2020) framework on 64 NVIDIA H-100 GPUs. Since the model size is relatively small, each GPU has a copy of the whole model to ensure computational balance, and the micro-batch size is set to 4. Models are trained on 100B tokens, sampled from a large-scale multilingual corpus (Gao et al., 2020; Weber et al., 2024) designed for continued pretraining.

Before training, we upcycled the dense model by duplicating the FFN 8 times to form $n = 8$ experts and apply a top-2 selection, which is a classic setting (Jiang et al., 2024) and initializing $m = 8$ of our proposed routers in each layer to construct an 8x0.5B MoE model. Other parts of the dense model remained unchanged. The average key representations in the router were obtained by freezing the dense model and calculating the average key vectors over 10 iterations, with a batch size of 1024 and a sequence length of 4096. The original dense model employs 16 attention heads, each with a dimension of 64, resulting in a total feature dimension of $d = 1024$. We merge every 2 token query transformations and expert keys with the highest cosine similarity to form $m = 8$ routers to align with the expert number $n = 8$, and the intermediate dimension of each router is $d' = 128$. The additional router parameters are approximately $8 \times 1024 \times 128 = 1\text{M}$ for each layer, which is tiny compared to the 2.1B total MoE parameters.

During training, we employ the Adam optimizer (Kingma and Ba, 2017) with hyper-parameters set to $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$, gradient clipping norm $= 1.0$ and weight decay $= 0.1$. We do not use dropout due to the abundant training corpus. The learning rate is scheduled using a warmup-and-step-decay strategy (Dai et al., 2024). In the first 1% of warm-up steps, the learning rate increases from 0 to the maximum value, which is set to 5×10^{-4}. The learning rate stays at this constant value until the last 20% of training steps, where it is multiplied by 0.316 (approximately $1/\sqrt{10}$) at 80% and 90% of the training steps. Each training batch contains 4M tokens, with the batch size and sequence length set to 1024 and 4096, respectively. The total number of training steps is 25000 to match 100B training tokens.

For MoE settings, we do not drop any tokens

during training except for the Switch Transformer (Fedus et al., 2022). Our model leverages an auxiliary loss (Lewis et al., 2021) of 0.02 and a router z-loss (Zoph et al., 2022) of 0.001 to improve router stability.

4.2 Evaluation Benchmarks

We conduct experiments on several benchmark datasets commonly used in evaluating MoE models, grouped by the abilities needed:

- Understanding: OpenbookQA, OpenbookQA-fact (Mihaylov et al., 2018).
- Reasoning: ARC-C (Clark et al., 2018) and ARC-E, Hellaswag (Zellers et al., 2019), Winogrande (Sakaguchi et al., 2019).
- Knowledge: BoolQ (Clark et al., 2019), COPA (Gordon et al., 2011), TriviaQA (Joshi et al., 2017), NQ (Kwiatkowski et al., 2019)

We evaluate models in a zero-shot manner to showcase their generalization abilities without further instructions.

4.3 Baselines

We compare Router Upcycling with five other MoE model variants:

- **Vanilla MoE from scratch**: A vanilla MoE model (Shazeer et al., 2017) using traditional softmax gating. A normal initialization (mean $= 0.0$, std $= 0.02$) is applied to initialize all parameters.
- **Vanilla Upcycling**: Based on the vanilla MoE model, experts' parameters are upcycled (Komatsuzaki et al., 2023) as copies of the original dense model's FFN, and other parameters are also converted from the dense model, except routers.
- **Switch Transformer Upcycling**: Based on the Vanilla Upcycling model, the router is changed to the one proposed in Switch Transformer (Fedus et al., 2022).
- **LocMoE Upcycling**: Based on the Vanilla Upcycling model, the router is changed to the one proposed in LocMoE (Li et al., 2024).
- **Upcycling w/ MLP routers**: Based on the Vanilla Upcycling model, a two-layer MLP with a GELU (Hendrycks and Gimpel, 2023) activation function is used as the router, with the following dimension transformations: Layer 1: $1024 \rightarrow 1024$; Layer 2: $1024 \rightarrow 8$. Its router parameters are approximately the same as the proposed method.

5 Results and Analysis

5.1 Performance Comparison

Table 1 showcases the performance of our method compared to the baselines. Our findings can be summarized as follows:

- **Overall Performance**: Router Upcycling achieves the highest average performance across all benchmarks, with an average score of 45.83, improving by 2.05 points over the Vanilla Upcycling method. It consistently outperforms other upcycling baselines, demonstrating the robustness and generalization of the proposed routing method.
- **Understanding Tasks**: Router Upcycling excels in understanding tasks, achieving an average score of 50.80, compared to 49.40 by Vanilla Upcycling. The diverse token assignments facilitated by the collaborative routers prevent expert under-specialization and ensure exposure to varied token representations.
- **Reasoning Tasks**: In reasoning tasks, Router Upcycling achieves an average score of 51.86, surpassing the next-best score of 49.61 by Vanilla Upcycling. It shows a notable 4% improvement in accuracy on the ARC datasets, thanks to better alignment of token representations and expert features.
- **Knowledge Tasks**: Router Upcycling leads in knowledge tasks with an average score of 37.31, compared to 35.35 by LocMoE Upcycling. The initialization of expert features as the average key from the previous attention layer ensures training stability and maintains the integrity of knowledge representation.

Our Router Upcycling method achieves more diverse token assignments, leading to expert specialization in MoE upcycling. Additionally, our upcycling method accelerates the evolution of the MoE model more effectively than vanilla upcycling, as shown in Figure 3. Normally, the LM loss should rise during the warm-up stage in continued pretraining methods, as our method demonstrates. However, the vanilla upcycling method

Method	Vanilla Upcycling	m=2, n=8 key=8	m=4, n=8 key=8	**m=n=8 key=8 (Ours)**	m=16, n=8 key=16	m=32, n=8 key=32	m=n=16 key=16
OBQA	40.8	38.0	29.8	**42.6**	26.8	42.2	37.0
OBQA-fact	58.0	57.8	40.8	**59.0**	27.6	52.6	54.2
ARC-C	39.66	32.88	34.58	**42.37**	23.73	41.02	40.00
ARC-E	52.73	45.21	40.92	**58.02**	23.81	51.15	56.08
Hellaswag	49.36	47.24	39.56	**50.12**	30.67	49.26	50.00
Winogrande	56.67	57.62	55.56	56.91	52.49	**58.48**	57.62
BoolQ	49.88	51.65	47.22	**55.84**	44.43	53.49	41.01
COPA	62	**68**	63	64	59	64	61
NQ	4.85	3.32	2.52	**5.15**	1.05	3.96	3.66
TriviaQA	23.82	18.78	13.45	**24.25**	9.39	22.8	23.93
Average	43.78	42.05	36.74	**45.83**	29.90	43.90	42.45

Table 2: Ablation study on the number of routers, keys, and experts. m is the router number, key is the number of keys, and n is the expert number. Note that we have conducted all possible ablation studies on the numbers.

Method	Max Pooling	**Summation**
OpenbookQA	**44.6**	42.6
OpenbookQA-fact	56.4	**59.0**
ARC-C	37.63	**42.37**
ARC-E	54.14	**58.02**
Hellaswag	49.55	**50.12**
Winogrande	**58.17**	56.91
BoolQ	**56.54**	55.84
COPA	63	**64**
NQ	3.38	**5.15**
TriviaQA	22.62	**24.25**
Average	44.60	**45.83**

Table 3: Performance comparison between two models utilizing max pooling and summation as mixture methods.

fails to warm up and improves slowly in the later stages. Consequently, after the 50B training schedule, our Router Upcycling model has a smaller LM loss than the vanilla upcycling model, proving its utility with lower perplexity. We also verify the conclusions from (Komatsuzaki et al., 2023) that upcycling performs better than training an MoE model from scratch in small training regimes.

5.2 Ablation Study

In this section, we further conduct ablation studies to assess the impact of different components of our method.

5.2.1 Number of Routers, Keys, and Experts

We investigate the impact of varying the number of routers, keys, and experts on model performance, computational overhead, and expression power.

In Table 2, we set the number of experts to a fixed value and study the influence of the number of routers and keys. The router number m is a critical hyperparameter as it determines how attention heads are grouped and merged in the base model. We experiment with values of m that are powers of 2 and less than or equal to 16 (the number of attention heads in the base Qwen model). Two rules are applied to form proper models when the number of routers m and keys key changes compared to a fixed number of experts n: (1) If the router number $m < n$, attention heads are duplicated n/m times before merging to match the hidden size in the routers, and each expert holds one key, so $key = n$; (2) If the router number $m > n$, without any duplication, $m = key$, and each expert preserves more than one key as their feature representation, which would be selected based on the top-k score of any of its keys. Additionally, we create a new model variant "m=32, n=8, key=32" by splitting each attention head in the original dense model into two to explore further possibilities.

The results indicate no particular trend when the router number m changes, but the model performs best when m=n=8. Apart from $m = 8$, most other variants are outperformed by the vanilla upcycling method. When $m = 16$, the performance is the worst, possibly due to the limited expression power of a single attention head with a dimension of only 64 for the routing task. However, the model performance improves when $m = 32$ despite the smaller attention head dimension. Therefore, we conclude that the attention dimension in routers should not equal the dimension of each attention head in the attention module. Intuitively, aligning the number of routers m with the number of experts n appears to benefit the routing process.

In the last column of Table 2, we scale up the model by setting the expert number $n = 16$ to

create the variant "m=n=16, key=16". Its performance is worse than the optimized variant "m=n=8, key=8" with only 8 experts. This comparison suggests that scaling up the number of experts may not be effective for our proposed method in small regimes.

5.2.2 Router Mixture Methods

In this section, we explore alternative mixture methods for collaborative routers, comparing summation with max pooling to determine the optimal method for output aggregation. Max pooling involves using the top-k router score of all Query-Key pairs in all routers to route the token. As shown in Table 3, the max pooling method surpasses the summation method on minor benchmarks such as OpenbookQA, Winogrande, and BoolQ, with improvements of less than 2%. Although max pooling can reduce the negligible computational overhead in the summation process, the summation method generally performs better than max pooling.

6 Related Work

The prototype MoE models utilized naive routing strategies, where the gate network assigned tokens to experts based on their highest scores, typically employing a softmax over the product of the input token and a gating weight matrix.

Some works introduce noise and normalization to enhance the robustness of routers. Shazeer et al. (Shazeer et al., 2017) proposed improvements by introducing noise for load balance and retaining the top-k experts. Switch Transformers (Fedus et al., 2022) address overfitting in fine-tuning tasks with limited examples by simplifying the routing mechanism using a top-1 gating strategy, reducing computational overhead and communication costs.

Other works focus on adjusting routing mechanisms. StableMoE (Dai et al., 2022) proposes a two-stage routing strategy with a distilled router for stable decisions. Zuo et al. (Zuo et al., 2022) introduce stochastic experts to bypass the router, promoting consistency through regularization. LocMoE (Li et al., 2024) introduces a GrAP layer that divides the hidden state of tokens, computes gating values without learnable parameters, and adds a locality loss to ensure tokens are preferentially routed to local experts. Several studies have also explored dynamic routing mechanisms (Huang et al., 2024a; Zeng et al., 2024), allowing tokens to select a varying number of experts based on input difficulty, enhancing computational efficiency and model performance.

However, these routing mechanisms often encourage token clustering around expert centroids, leading to representation collapse (Chi et al., 2022). To tackle this, Chi et al. (Chi et al., 2022) leverage dimension reduction using linear projection to isolate interactions on a low-dimensional hypersphere. Other novel routing methods include expert choice routing (Zhou et al., 2022), where experts select tokens, and hashing-based routing (Roller et al., 2021), replacing traditional routers with hashing to address load imbalance.

Recent studies have attempted to build attention-like multi-head routers. Wu et al. (Wu et al., 2024) propose using smaller FFNs to process sub-tokens directly, but this approach is unsuitable for upcycling scenarios and fails to highlight diverse token representation. Another work (Huang et al., 2024b) tunes parameter settings for higher efficiency based on (Wu et al., 2024). Our method differs by initializing a mixture of collaborative routers from attention modules in a dense checkpoint, enhancing the model's ability to capture diverse patterns within the data and leading to improved performance and stability in MoE upcycling scenarios.

7 Conclusion

We introduce the first router specifically designed for upcycling in Mixture-of-Experts (MoE) models, utilizing a mixture of collaborative routers initialized from the attention module in the base dense model. Our method enhances the routing mechanism's precision, efficiency, and alignment by projecting tokens and experts into multiple low-dimensional representations and computing matching scores in an attention-like mechanism across these subspaces.

Experiment results on benchmark datasets demonstrate that our "free lunch" method achieves state-of-the-art performance, outperforming traditional upcycling methods by more than 2% in general and 4% on the ARC dataset. This framework pioneers router optimization in the upcycling scenario and extends upcycling from only upcycling the experts to upcycling the entire MoE structure. Our future work will optimize hyperparameters, extend the method to other models and tasks, and investigate the theoretical aspects of routing mechanisms in upcycling for deeper insights.

Limitations

While our Router Upcycling method demonstrates improvements in MoE models, there are some limitations to consider:

- **Limited Computational Budget**: Due to a limited computational budget, our experiments were conducted with only one model. This constraint is common among research groups in universities, including ours, which often lack the resources for extensive computational experiments. Despite this limitation, we conducted thorough research on the selected model to prove the effectiveness of our method. Future work could benefit from evaluating our proposed method on a broader range of models and larger datasets to validate its generalization and robustness.

- **Interpretability Dilemma**: Understanding how and why the router makes specific routing decisions is crucial for further improvements and trust in MoE models. However, the interpretability of the router's functionality remains an open question. Previous works have introduced concepts like gate importance (Shazeer et al., 2017) and analyzed expert specialization (Zhang et al., 2022; Zhu et al., 2024; Zoph et al., 2022). Nevertheless, many contradictions in this field remain and more research is needed to fully understand and explain the behavior of routers in MoE models. Therefore, in this study, we did not conduct interpretability experiments. Given the limited computational resources, our priority was to validate the performance improvements of the proposed method.

- **Hyperparameter Sensitivity and Generalization**: Another limitation to consider is the potential impact of hyperparameter settings on the performance of our method. While we have optimized certain hyperparameters, further tuning and exploration could yield even better results. Additionally, our method has primarily been tested on a specific architecture and set of tasks. Extending the evaluation to other architectures and diverse tasks would provide a more comprehensive understanding of its effectiveness and limitations.

In conclusion, while our Router Upcycling method shows promise, addressing these limitations will be crucial for its broader adoption and further improvement.

Impacts and Ethical Considerations

The development of the Router Upcycling method for MoE models has several potential impacts:

- **Enhanced Model Efficiency**: By improving the routing mechanism, our method enhances the efficiency and performance of MoE models. This can lead to more accurate and faster models, which are beneficial for various applications in natural language processing (NLP).

- **Resource Optimization**: The ability to upcycle existing dense models into more efficient MoE models can help optimize the use of computational resources. This is particularly important for research groups with limited budgets, as it allows them to leverage existing models without extensive retraining from scratch.

- **Broader Accessibility**: Improved efficiency and resource optimization can make advanced NLP models more accessible to a wider range of users, including smaller research groups and organizations with limited computational resources. This democratization of technology can foster innovation and collaboration across the field.

While the Router Upcycling method presents substantial advantages, it also raises important ethical considerations that must be addressed. The interpretability dilemma, as discussed in our limitations section, emphasizes the critical need for transparency in routing decisions to ensure accountability and trustworthiness. Enhancing the interpretability of Mixture-of-Experts (MoE) models is essential for fostering trust in models moving forward, as it allows researchers to understand and verify the decision-making processes.

References

Zewen Chi, Li Dong, Shaohan Huang, Damai Dai, Shuming Ma, Barun Patra, Saksham Singhal, Payal Bajaj, Xia Song, Xian-Ling Mao, Heyan Huang, and Furu Wei. 2022. On the Representation Collapse of Sparse Mixture of Experts.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge.

Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. 2024. DeepSeekMoE: Towards Ultimate Expert Specialization in Mixture-of-Experts Language Models.

Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. StableMoE: Stable Routing Strategy for Mixture of Experts.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The pile: An 800gb dataset of diverse text for language modeling.

Andrew S. Gordon, Zornitsa Kozareva, and Melissa Roemmele. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*.

Ethan He, Abhinav Khattar, Ryan Prenger, Vijay Korthikanti, Zijie Yan, Tong Liu, Shiqing Fan, Ashwath Aithal, Mohammad Shoeybi, and Bryan Catanzaro. 2024. Upcycling Large Language Models into Mixture of Experts.

Dan Hendrycks and Kevin Gimpel. 2023. Gaussian error linear units (gelus).

Quzhe Huang, Zhenwei An, Nan Zhuang, Mingxu Tao, Chen Zhang, Yang Jin, Kun Xu, Kun Xu, Liwei Chen, Songfang Huang, and Yansong Feng. 2024a. Harder Tasks Need More Experts: Dynamic Routing in MoE Models.

Shaohan Huang, Xun Wu, Shuming Ma, and Furu Wei. 2024b. MH-MoE: Multi-Head Mixture-of-Experts.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive mixtures of local experts. *Neural Computation*, 3:79–87.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of Experts.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Aran Komatsuzaki, Joan Puigcerver, James Lee-Thorp, Carlos Riquelme Ruiz, Basil Mustafa, Joshua Ainslie, Yi Tay, Mostafa Dehghani, and Neil Houlsby. 2023. Sparse Upcycling: Training Mixture-of-Experts from Dense Checkpoints.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. 2021. BASE Layers: Simplifying Training of Large, Sparse Models.

Jing Li, Zhijie Sun, Xuan He, Li Zeng, Yi Lin, Entong Li, Binfan Zheng, Rongqian Zhao, and Xin Chen. 2024. LocMoE: A Low-Overhead MoE for Large Language Model Training.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering.

Qwen. 2024. Introducing qwen1.5.

Stephen Roller, Sainbayar Sukhbaatar, Arthur Szlam, and Jason Weston. 2021. Hash Layers For Large Sparse Models.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Winogrande: An adversarial winograd schema challenge at scale.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2020. Megatron-lm: Training multi-billion parameter language models using model parallelism.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need.

Maurice Weber, Daniel Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams,

Ben Athiwaratkun, Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Ré, Irina Rish, and Ce Zhang. 2024. Redpajama: an open dataset for training large language models.

Xun Wu, Shaohan Huang, Wenhui Wang, and Furu Wei. 2024. Multi-Head Mixture-of-Experts.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence?

Zihao Zeng, Yibo Miao, Hongcheng Gao, Hao Zhang, and Zhijie Deng. 2024. AdaMoE: Token-Adaptive Routing with Null Experts for Mixture-of-Experts Language Models.

Xiaofeng Zhang, Yikang Shen, Zeyu Huang, Jie Zhou, Wenge Rong, and Zhang Xiong. 2022. Mixture of Attention Heads: Selecting Attention Heads Per Token.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew Dai, Zhifeng Chen, Quoc Le, and James Laudon. 2022. Mixture-of-Experts with Expert Choice Routing.

Tong Zhu, Xiaoye Qu, Daize Dong, Jiacheng Ruan, Jingqi Tong, Conghui He, and Yu Cheng. 2024. LLaMA-MoE: Building Mixture-of-Experts from LLaMA with Continual Pre-training.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. 2022. ST-MoE: Designing Stable and Transferable Sparse Expert Models.

Simiao Zuo, Xiaodong Liu, Jian Jiao, Young Jin Kim, Hany Hassan, Ruofei Zhang, Tuo Zhao, and Jianfeng Gao. 2022. Taming Sparsely Activated Transformer with Stochastic Experts.