CartesianMoE: Boosting Knowledge Sharing among Experts via Cartesian Product Routing in Mixture-of-Experts

Zhenpeng Su[1,2] **Xing Wu**[1,2] **Zijia Lin**[3*] **Yizhe Xiong**[3] **Minxuan Lv**[1,2]
Guangyuan Ma[1,2] **Hui Chen**[3*] **Songlin Hu**[1,2*] **Guiguang Ding**[3]
[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences [3]Tsinghua University
suzhenpeng13@163.com {wuxing,maguangyuan,lvminxuan,husonglin}@iie.ac.cn
linzijia07@tsinghua.org.cn dinggg@tsinghua.edu.cn, {xiongyizhe2001,huichen}@gmail.com

Abstract

Large language models (LLM) have been attracting much attention from the community recently, due to their remarkable performance in all kinds of downstream tasks. According to the well-known scaling law, scaling up a dense LLM enhances its capabilities, but also significantly increases the computational complexity. Mixture-of-Experts (MoE) models address that by allowing the model size to grow without substantially raising training or inference costs. Yet MoE models face challenges regarding knowledge sharing among experts, making their performance somehow sensitive to routing accuracy. To tackle that, previous works introduced shared experts and combined their outputs with those of the top K routed experts in an "addition" manner. In this paper, inspired by collective matrix factorization to learn shared knowledge among data, we propose CartesianMoE, which implements more effective knowledge sharing among experts in more like a "multiplication" manner. Extensive experimental results indicate that CartesianMoE outperforms previous MoE models for building LLMs, in terms of both perplexity and downstream task performance. And we also find that CartesianMoE achieves better expert routing robustness.

1 Introduction

Large language models (LLM) have demonstrated impressive performance across various downstream natural language tasks (Touvron et al., 2023; Dai et al., 2022; Brown et al., 2020; Anil et al., 2023; Chowdhery et al., 2022; Radford et al., 2019; Rae et al., 2021; Biderman et al., 2023). Moreover, the well-known scaling law suggests that, as the model size increases, the model capabilities will continue to improve (Kaplan et al., 2020; Hoffmann et al., 2022). However, for dense LLMs, the computational costs of scaling up their model sizes can become prohibitively high. To tackle that, sparse activation networks are proposed (Child et al., 2019; Du et al., 2022). They reduce computational costs by activating only a subset of parameters for each input. A prominent approach among them is the mixture-of-experts (MoE) (Lepikhin et al., 2021; Du et al., 2022; Dai et al., 2024; Fedus et al., 2022; Roller et al., 2021), which involves training multiple experts but using only a subset to process each input, with each expert generally being a feed-forward network (FFN). Compared to dense LLMs of equivalent sizes, MoE LLMs effectively reduces computational costs while delivering comparable results, in terms of both perplexity (PPL) and downstream task performance (Lepikhin et al., 2021; Du et al., 2022; Dai et al., 2024; Su et al., 2024a; Huang et al., 2024b; Yang et al., 2024; Zhao et al., 2024a).

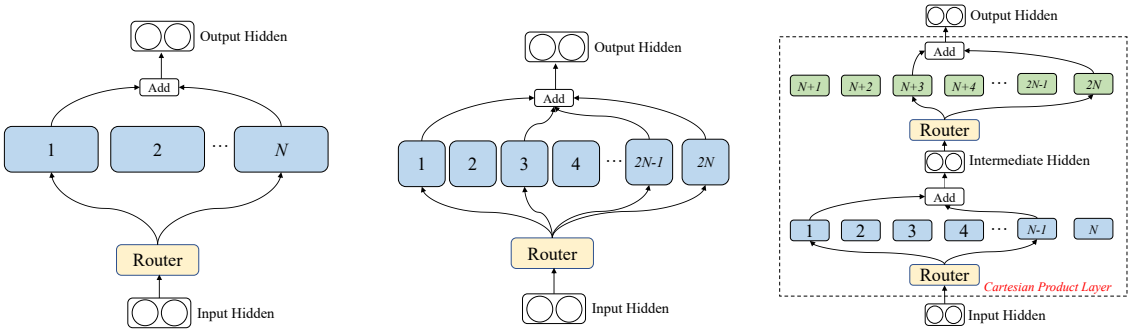
Conventional MoE models, like (Lepikhin et al., 2021; Fedus et al., 2022; Du et al., 2022), activate the top K routed experts among the total N experts. Due to the independent training of all experts, they rarely share learned knowledge, and thus routing fluctuations can affect the output substantially, making the performance of such MoE models somehow sensitive to the routing accuracy. To tackle that, (Dai et al., 2024; Rajbhandari et al., 2022) suggests using several fixed-activated shared experts to store shared knowledge, in addition to the top$K$ routed experts. And it has been well-validated to improve MoE model performance. With shared experts, (Dai et al., 2024; Yang et al., 2024) further split full-sized experts into more fine-grained experts to enhance representation specialization and gain additional performance improvement, where the N experts are split into mN smaller ones, and the top mK routed ones of them are activated.

The remarkable shared-expert method essentially merges the shared knowledge (i.e., outputs of shared experts) with the specific knowledge (i.e.,

*Corresponding authors.

This work was supported by Beijing Natural Science Foundation (L247026) and National Natural Science Foundation of China (No 62441235).

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 10040–10055



(a) Conventional Top-2 routing for full-sized experts (b) Top-4 routing for half-sized fine-grained experts (c) Top-4 Cartesian Product routing in CartesianMoE

Figure 1: Illustration of CartesianMoE. Subgraph a) represents the conventional top-2 routing for full-sized experts, subgraph b) illustrates the top-4 routing for half-sized fine-grained experts, and subgraph c) shows the top-4 Cartesian Product routing (i.e., top-2 routing for each sub-layer) in the proposed CartesianMoE. All subgraphs share the same numbers of model parameters and activated parameters.

outputs of the routed experts) in an "addition" manner. For instance, with a shared expert FFN_a and several routed experts FFN_b, FFN_c, and FFN_d, the knowledge sharing among experts can be represented as: $\text{FFN}_a + \text{FFN}_b$, $\text{FFN}_a + \text{FFN}_c$, and $\text{FFN}_a + \text{FFN}_d$. Inspired by collective matrix factorization to learn shared knowledge among data (Singh and Gordon, 2008), in this paper we propose to represent knowledge sharing among experts in an alternative "multiplication" manner, i.e., $\text{FFN}_a \cdot \text{FFN}_b$, $\text{FFN}_a \cdot \text{FFN}_c$, and $\text{FFN}_a \cdot \text{FFN}_d$. Specifically, by defining two sets of sub-experts $\{\text{FFN}_a^1, \text{FFN}_b^1, \ldots\}$ and $\{\text{FFN}_a^2, \text{FFN}_b^2, \ldots\}$, we derive each expert to be the combination of any two sub-experts from both sets respectively, like $\text{FFN}_{aa} = \text{FFN}_a^1 \cdot \text{FFN}_a^2$ or $\text{FFN}_{ab} = \text{FFN}_a^1 \cdot \text{FFN}_b^2$. In that sense, each expert share an identical sub-expert with many others. It can also be seen that, all the experts can be derived by the Cartesian product between both sub-expert sets, and thus we term our proposed method as CartesianMoE. Specifically, in our proposed CartesianMoE, we replace the conventional MoE layer as a *Cartesian Product Layer*, which consists of two sequential MoE sub-layers, each denoting a set of sub-experts, as illustrated in Fig. 1. Then the routing process to select routed experts is also divided into the two MoE sub-layers, termed as Cartesian Product routing.

Extensive experiments on building MoE LLMs show that CartesianMoE yields superior performance than previous counterparts, using the same number of model parameters and activated parameters. CartesianMoE also shows better routing robustness. We argue that the superiority of CartesianMoE comes from its more fine-grained knowledge sharing among experts. Specifically, compared to the shared-expert method that requests all routed experts to always share the same global knowledge held by the fixed shared experts, CartesianMoE allows to divide experts into groups with each sharing some group-wise knowledge. In that sense, CartesianMoE is supposed to be also equipped with shared experts, so as to form a "global shared knowledge + group-wise shared knowledge + expert-specific knowledge" system.

Our contributions are summarized as follows:

- Inspired by collective matrix factorization to learn shared knowledge among data, we analyze the feasibility of enabling knowledge sharing among experts in a "multiplication" manner, an alternative to the "addition" manner proposed by the shared-expert method.
- We propose CartesianMoE, which derives experts via the Cartesian Product of two sub-expert sets. CartesianMoE enables group-wise knowledge sharing among experts and helps to build a more complete knowledge sharing system with shared experts equipped.
- We validate the effectiveness of the proposed CartesianMoE with extensive experiments. Experimental results show that it consistently outperforms previous MoE models, and shows better routing robustness.

2 Related Work

The concept of MoE models was first introduced by (Jacobs et al., 1991). Then, (Eigen et al., 2013) extended the MoE model to multiple layers. Later,

(Shazeer et al., 2017) extended that idea to Long Short-Term Memory (LSTM) networks (Graves, 2013), training an LSTM model with up to 137 billion parameters. With the advent of the Transformer architecture (Vaswani et al., 2017; Devlin et al., 2019), the Gshard model (Lepikhin et al., 2021) applied MoE techniques to Transformers, paving the way for the development of more advanced MoE models like GLaM (Du et al., 2022) and Switch Transformer (Fedus et al., 2022).

In early works (Zoph et al., 2022; Fedus et al., 2022; Du et al., 2022; Lepikhin et al., 2021; Roller et al., 2021; Dai et al., 2022), when extending dense models to MoE models, the MoE layer of the Transformer consists of multiple FFNs that are of the same size as those in the dense models. Recent works (Muennighoff et al., 2024; Dai et al., 2024; Yang et al., 2024) show that splitting a fully-sized FFN into several smaller, fine-grained experts facilitates representation specialization. Additionally, shared experts are commonly adopted (Dai et al., 2024; Rajbhandari et al., 2022; Su et al., 2024a), to enhance knowledge sharing among experts for performance improvement.

The shared-expert method combines shared knowledge (i.e., the outputs of the shared experts) with specialized knowledge (i.e., the outputs of the routed experts) in an "addition" manner. Inspired by collective matrix factorization to learn shared knowledge among data, here we propose an alternative "multiplication" manner to share expert knowledge, which demonstrates superiority over previous MoE methods.

3 Background

3.1 Large Language Models

For simplicity, here we focus on the mainstream *generative LLM* with the Transformer backbone. Given a sequence of T tokens $\mathbf{x} = (x_1, x_2, \ldots, x_T)$, a generative LLM iteratively produces a probability distribution $\mathbf{p}$ over the vocabulary for each token, conditioning on its preceding tokens. Usually, the cross-entropy loss function is employed to optimize the predicted probability w.r.t the ground-truth token x_t. And thus in total, the training loss $\mathcal{L}_{lm}$ for the generative LLM can be expressed as:

$$\begin{aligned} \mathcal{L}_{lm} &= -\sum_{t=1}^{T-1} \log(\mathbf{P}_{x_{t+1},t}) \\ \text{s.t.,}\quad \mathbf{P}_{\cdot,t} &= \text{softmax}(W\mathbf{H}^L_{\cdot,t}) \\ \mathbf{H}^L &= \text{Transformer}(x_1, x_2, \ldots, x_{T-1}) \end{aligned} \tag{1}$$

Here, L is the number of blocks in the Transformer backbone. $\mathbf{P}_{\cdot,t}$ and $\mathbf{H}^L_{\cdot,t}$ represent the t-th column of the matrices $\mathbf{P}$ and $\mathbf{H}^L$, respectively, corresponding to x_t. $\mathbf{H}^L = [\mathbf{h}^L_1, \mathbf{h}^L_2, \ldots, \mathbf{h}^L_{T-1}]$ denotes the hidden states of the last layer, and $\mathbf{P}_{x_{t+1},t}$ denotes the predicted probability w.r.t the ground-truth token x_{t+1} in $\mathbf{P}_{\cdot,t}$. Here the linear projection layer W takes $\mathbf{H}^L_{\cdot,t}$ as input to compute the probability distribution $\mathbf{P}_{\cdot,t}$ across the vocabulary.

In the Transformers backbone, each layer features a multi-head self-attention (MHA) module and a feed-forward network (FFN), with the FFN typically comprising two fully connected layers. Formally,

$$\begin{aligned} \hat{\mathbf{h}}^l_t &= \text{MHA}([\mathbf{h}^{l-1}_1, \mathbf{h}^{l-1}_2, \ldots, \mathbf{h}^{l-1}_t]) \\ \mathbf{h}^l_t &= \text{FFN}(\hat{\mathbf{h}}^l_t) \end{aligned} \tag{2}$$

where l denotes the l-th block in the Transformer backbone.

3.2 Mixture-of-Experts

MoE methods typically replace the dense model's FFN module with an MoE module composed of multiple FFNs, each being an *expert*. The outputs of these FFNs are combined using a routing function, $\mathbf{r}(\cdot)$, referred to as the *router*. Formally,

$$\mathbf{h}^l_t = \sum_{i=1}^{N} \mathbf{r}_i(\hat{\mathbf{h}}^l_t) \cdot \text{FFN}_i(\hat{\mathbf{h}}^l_t) \quad \text{s.t. } |\mathbf{r}(\hat{\mathbf{h}}^l_t)|_0 = K \tag{3}$$

where N is the number of experts in a single MoE module, K is the number of activated experts, $\mathbf{r}_i$ represents the routing outcome for the i-th expert, and $|\cdot|_0$ denotes the L_0-norm, i.e., the number of non-zero elements. With $K \ll N$, only a small subset of experts is activated. And thus increasing the total number of experts in MoE models does not significantly increase computational time.

For fine-grained experts (Muennighoff et al., 2024; Dai et al., 2024; Yang et al., 2024), each of the original N experts is split into m equal parts, resulting in mN fine-grained experts in total. In that case, the intermediate size of the fine-grained

experts is $\frac{1}{m}$ of the original full-sized experts. To maintain a constant number of activated parameters, the number of activated experts is usually adjusted to mK as well. Formally,

$$\mathbf{h}_t^l = \sum_{i=1}^{mN} \mathbf{r}_i(\hat{\mathbf{h}}_t^l) \cdot \text{FFN}_i(\hat{\mathbf{h}}_t^l) \quad \text{s.t. } |\mathbf{r}(\hat{\mathbf{h}}_t^l)|_0 = mK \tag{4}$$

4 Method

4.1 Proposed CartesianMoE

As mentioned above, the current MoE models either rarely share learned knowledge among experts or only apply shared experts to share global knowledge. We propose CartesianMoE, as shown in Fig. 1, to facilitate more thorough expert sharing.

As shown in Figure 1(c), the proposed CartesianMoE introduces a *Cartesian Product Layer*, and also employs fine-grained experts in its two MoE sub-layers, denoted as A and B. Then CartesianMoE combines the fine-grained sub-experts across the two MoE sub-layers to derive real experts. Formally,

$$\begin{aligned} A \times B &= \{(a,b) \mid a \in A \text{ and } b \in B\} \\ \text{s.t.} \quad A &= \{\text{FFN}_1, \ldots, \text{FFN}_e\}, \\ B &= \{\text{FFN}_{e+1}, \ldots, \text{FFN}_{2e}\}. \end{aligned} \tag{5}$$

where e is the number of sub-experts in each MoE sub-layer. To maximize the diversity of $A \times B$, we set $e = mN/2$ experts, with mN being the number of all fine-grained sub-experts. Specifically, the computation of the *Cartesian Product Layer* is formulated as follows.

$$\tilde{\mathbf{h}}_t^l = \sum_{i=1}^{e} \mathbf{r}_i^1(\hat{\mathbf{h}}_t^l) \cdot \text{FFN}_i(\hat{\mathbf{h}}_t^l) \tag{6}$$

$$\bar{\mathbf{h}}_t^l = \tilde{\mathbf{h}}_t^l + \hat{\mathbf{h}}_t^l \tag{7}$$

$$\mathbf{h}_t^l = \sum_{i=e+1}^{2e} \mathbf{r}_i^2(\bar{\mathbf{h}}_t^l) \cdot \text{FFN}_i(\bar{\mathbf{h}}_t^l) \tag{8}$$

$$\text{s.t. } |\mathbf{r}^1(\hat{\mathbf{h}}_t^l)|_0 = |\mathbf{r}^2(\bar{\mathbf{h}}_t^l)|_0 = k \tag{9}$$

where $\mathbf{r}^1$ and $\mathbf{r}^2$ represent the *routers* corresponding to the 1st and 2nd MoE sub-layers of the *Cartesian Product Layer*, respectively. Note that we also add a residual connection between the two MoE sub-layers, to ensure that tokens exceeding the capacity of a sub-expert in the 1st MoE sub-layer can be directly passed to the 2nd MoE sub-layer, i.e., "token droppable" in (Fedus et al., 2022) to balance optimization among experts. In order to maintain a consistent total number of activated parameters as previous fine-grained MoE methods, the number of activated experts per MoE sub-layer, i.e., k in Eq. 9, is also reduced by half, i.e., $k = mK/2$. We term such a routing process as Cartesian Product routing. Through such a two-layer structural design, the Cartesian Product mechanism is natively implemented, and is supposed to facilitate knowledge sharing among experts.

Then following Transformer (Vaswani et al., 2017), we add $\bar{\mathbf{h}}_t^l$ to $\mathbf{h}_t^l$ to serve as the input for the next block with a skip connection. Formally,

$$\mathbf{h}_t^l \leftarrow \mathbf{h}_t^l + \bar{\mathbf{h}}_t^l \tag{10}$$

4.2 Load Balance Loss

LLMs are typically trained in a distributed manner, which can lead to load imbalances in MoE models (Lepikhin et al., 2021; Fedus et al., 2022; Dai et al., 2022), where a minority of experts handle the majority of tokens and meanwhile the majority of experts remain idle. Such imbalances can adversely affect the training efficiency. To address that issue, a load balancing loss is commonly introduced in the training of MoE models. We follow (Huang et al., 2024a; Fedus et al., 2022) and employ a balanced loss function by summing the routing losses of both MoE sub-layers within a *Cartesian Product Layer*:

$$\begin{aligned} \mathcal{L}_{bal} &= \sum_{i=1}^{e} w_i^1 R_i^1 + \sum_{i=e+1}^{2e} w_i^2 R_i^2 \\ \text{s.t.,} \quad w_i^k &= \frac{1}{B} \sum_{j=1}^{B} \mathbb{I}\left\{\text{argmax}(\mathbf{r}_{:,j}^k) = i\right\} \\ R_i^k &= \frac{1}{B} \sum_{j=1}^{B} \mathbf{r}_{i,j}^k \\ &\forall\, k \in \{1,2\} \end{aligned} \tag{11}$$

where B represents the number of tokens in a mini-batch, $k \in \{1,2\}$ denotes the sub-layer index within a *Cartesian Product Layer*, $\mathbf{r}_{:,j}^k$ denotes the routing output probability distribution for the j-th token in the 1st ($k=1$) or 2nd ($k=2$) MoE sub-layer, and $\mathbf{r}_{i,j}^k$ represents the specific probability value with respect to the i-th expert in either the 1st ($k=1$) or 2nd ($k=2$) MoE sub-layer.

Our final loss is a combination of the language model loss and the load-balance loss:

$$\mathcal{L} = \mathcal{L}_{lm} + \alpha \mathcal{L}_{bal} \tag{12}$$

where α is a hyperparameter.

4.3 Relations to Flattened Fine-grained Experts

As detailed above, the proposed CartesianMoE leverages two layers of fine-grained sub-experts to build a *Cartesian Product Layer*. Then it would be interesting to see its relations to the flattened fine-grained experts proposed in (Dai et al., 2024).

Suppose the number of fine-grained experts/sub-experts is $2e$ for both methods, same as before. The output of the MoE module in (Dai et al., 2024), together with that of the residual connection, is formulated as below:

$$\begin{aligned}\mathbf{h}_t^l = \hat{\mathbf{h}}_t^l &+ \sum_{i=1}^{e} \mathbf{r}_i^1(\hat{\mathbf{h}}_t^l) \cdot \text{FFN}_i(\hat{\mathbf{h}}_t^l) \\ &+ \sum_{i=e+1}^{2e} \mathbf{r}_i^1(\hat{\mathbf{h}}_t^l) \cdot \text{FFN}_i(\hat{\mathbf{h}}_t^l)\end{aligned} \tag{13}$$

where $\mathbf{r}^1$ denotes the single *router* in the MoE module. As for the proposed CartesianMoE, the output of the *Cartesian Product Layer*, can be derived as below, via integrating Eq. 6, Eq. 7 and Eq. 8 into Eq. 10.

$$\begin{aligned}\mathbf{h}_t^l = \hat{\mathbf{h}}_t^l &+ \sum_{i=1}^{e} \mathbf{r}_i^1(\hat{\mathbf{h}}_t^l) \cdot \text{FFN}_i(\hat{\mathbf{h}}_t^l) \\ &+ \sum_{i=e+1}^{2e} \mathbf{r}_i^2(\bar{\mathbf{h}}_t^l) \cdot \text{FFN}_i(\bar{\mathbf{h}}_t^l)\end{aligned} \tag{14}$$

Comparing Eq. 13 and Eq. 14, it can be seen that the proposed CartesianMoE and the flattened fine-grained experts mainly differ at the 3rd parts of both equations, i.e., $\sum_{i=e+1}^{2e} \mathbf{r}_i^1(\hat{\mathbf{h}}_t^l) \cdot \text{FFN}_i(\hat{\mathbf{h}}_t^l)$ versus $\sum_{i=e+1}^{2e} \mathbf{r}_i^2(\bar{\mathbf{h}}_t^l) \cdot \text{FFN}_i(\bar{\mathbf{h}}_t^l)$. Specifically, instead of sharing the same *router* $\mathbf{r}^1$ and the same input $\hat{\mathbf{h}}_t^l$ as the 2nd part, CartesianMoE leverages a separate *router* $\mathbf{r}^2$ and the output of the 1st sub-layer as input, i.e., $\bar{\mathbf{h}}_t^l$. Given $\bar{\mathbf{h}}_t^l = \tilde{\mathbf{h}}_t^l + \hat{\mathbf{h}}_t^l$ in Eq. 7, CartesianMoE can probably enjoy deeper representations of the input than the flattened counterpart, and the separate *router* offers more flexibility. Both can help CartesianMoE to achieve performance enhancement, as demonstrated in our experiments.

5 Experiments

5.1 Pre-training Dataset

Following previous works (Xie et al., 2023; Su et al., 2024b), we use the Pile dataset (Gao et al., 2021) as our pre-training data. The Pile is a large-scale, publicly available corpus comprising 22 domains and over 825 GB of English text. For tokenization, we utilize the widely adopted LLaMA tokenizer with a vocabulary size of 32k. We compute the sampling rate for each domain based on the number of tokens after tokenization, following the methodology described in (Xie et al., 2023; Su et al., 2024b). Due to our limited computational resources, unless otherwise specified, the models are pre-trained using 100B tokens, following (Dai et al., 2024; Su et al., 2024a; Xie et al., 2023; Su et al., 2024b; Huang et al., 2024a; Xiong et al., 2024; Lian et al., 2024).

5.2 Experimental Setup

Following (Yang et al., 2024; Huang et al., 2024a), we implement the LLaMA architecture for the *LARGE* models with 24 Transformer blocks and a hidden state dimensionality of 1024, and for the *BASE* models with 12 Transformer blocks and a hidden-state dimensionality of 768. We employ the AdamW (Loshchilov and Hutter, 2019) optimizer for all models with a cosine learning rate decay schedule. For the dense models, following (Touvron et al., 2023; Su et al., 2024b), we set the learning rate as $3e^{-4}$. For the MoE models, following (Lewis et al., 2021; Su et al., 2024b; Touvron et al., 2023), we reduce the learning rate to $1.5e^{-4}$ to ensure model convergence. By default, we set our maximum sequence length to 1024.

Following (Yang et al., 2024; Huang et al., 2024a; Su et al., 2024a), we conduct experiments on two different MoE model settings: *MoE-Base* and *MoE-Large*. The specific size configurations are shown in Table 1. We follow Gshard (Lepikhin et al., 2021), and replace the FFN layer with an MoE layer for every other Transformer block, resulting in a total of 12 MoE layers for *MoE-Large* and 6 MoE layers for *MoE-Base* in this setting. For the hyperparameter α w.r.t the load balanced loss (Eq. 12), we set it to 0.01. The expert capacity factor of tokens is set as 1 during training. Moreover, we adopt a dropless setup, ensuring that every token is retained during evaluation.

In the CartesianMoE, each *Cartesian Product Layer* contains 32 fine-grained sub-experts, with each sub-expert having a half-sized FFN. We assign 16 fine-grained sub-experts to each of the two MoE sub-layers, and use top-2 routing for each. In addition, each MoE sub-layer has a fixed-activated shared expert, so as to form a "global

Model	Configuration	SE	FGE	Params	Activated Params	Pile PPL (↓)
Base Model	d=768, D=3072	N/A	N/A	162M	162M	8.55
Large Model	d=1024, D=4096	N/A	N/A	468B	468M	6.95
MoE-Base						
SMoE-Share	d=768, D=3072, top$K$=2	True	False	842M	247M	7.37
SMoE-Top3	d=768, D=3264, top$K$=3	False	False	842M	258M	7.40
Hash Layer	d=768, D=3072, top$K$=2	True	False	842M	247M	7.47
Fine-grained Routing	d=768, D=1536, top$K$=4,	True	True	842M	247M	7.33
TopP Routing	d=768, D=3072, top$P$=0.4	True	False	842M	247M	7.41
CartesianMoE	d=768, D=1526, top$K$=(2+2)	True	True	842M	247M	**7.19**
MoE-Large						
SMoE-Share	d=1024, D=4096, top$K$=2	True	False	2.88B	770M	6.13
SMoE-Top3	d=1024, D=4352, top$K$=3	False	False	2.88B	808M	6.18
Hash Layer	d=1024, D=4096, top$K$=2	True	False	2.88B	770M	6.28
Fine-grained Routing	d=1024, D=2048, top$K$=4	True	True	2.88B	770M	6.16
TopP Routing	d=1024, D=4096, top$P$=0.4	True	False	2.88B	770M	6.14
CartesianMoE	d=1024, D=2048, top$K$=(2+2)	True	True	2.88B	770M	**6.08**

Table 1: Perplexity (PPL) results of language modeling. The best score is marked in **bold**. SE indicates whether to use shared experts, FGE indicates whether to use fine-grained experts, d represents the hidden state dimensionality, D represents the intermediate size of each FFN, and top$K$ refers to the number of experts activated for each token. For CartesianMoE, top$K$=(2+2) means that each of the two sub-layers activates two sub-experts. For TopP Routing, top$P$ is the threshold that controls how many experts should be activated to reach it.

shared knowledge + group-wise shared knowledge + expert-specific knowledge" system mentioned before. We compare the proposed CartesianMoE with 6 remarkable baselines (Touvron et al., 2023; Fedus et al., 2022; Roller et al., 2021; Huang et al., 2024a; Dai et al., 2024) in our experiments. The respective parameter settings for each compared model are provided in Appendix 9.1. Considering that shared experts are commonly included in MoE models (Dai et al., 2024; DeepSeek-AI et al., 2024; Zhao et al., 2024b; Rajbhandari et al., 2022; Su et al., 2024a), all compared baselines include shared experts to gain further performance improvement unless otherwise noted.

5.3 Main Results

We first present the model's perplexity (PPL) on the Pile validation set. Then, following (Touvron et al., 2023; Brown et al., 2020; Su et al., 2024b; Dai et al., 2024), we evaluate the model performance on various downstream benchmarks, including zero-shot tests for HellaSwag (Zellers et al., 2019), LAMBADA (Paperno et al., 2016), PIQA (Bisk et al., 2020), StoryCloze (Mostafazadeh et al., 2016), and Winogrande(Wino) (Sakaguchi et al., 2020), in terms of *accuracy*. In addition, following (Touvron et al., 2023; Su et al., 2024b), we conduct 5-shot evaluations on TriviaQA (Joshi et al., 2017), WebQuestions (WebQs) (Berant et al., 2013), and Natural Questions (NaturalQs) (Kwiatkowski et al., 2019) using the *exact match* metric.

5.3.1 Perplexity Results

Table 1 shows the perplexity (PPL) of language modeling on the Pile validation set. With the *same number of activated parameters*, the MoE models (*MoE-Base/MoE-Large*) consistently outperform the dense models (*Base/Large Model*) with significantly reduced PPL. Furthermore, CartesianMoE exhibits a substantial performance improvement over other models, in both *MoE-Base* and *MoE-Large* settings. The result presents the superiority of CartesianMoE, which is equipped with complete "global shared knowledge + group-wise shared knowledge + expert-specific knowledge".

Note that *Fine-grained Routing* with flattened fine-grained experts exhibits inconsistent improvements across different model sizes. In the *MoE-Base* setting, it significantly outperforms *SMoE-Share*, but in the *MoE-Large* setting, it performs slightly worse than *SMoE-Share*. In contrast, CartesianMoE demonstrates consistent performance improvements across different settings, highlighting its consistent superiority.

5.3.2 Benchmark Results

As shown in Table 2, we present the model's performance on downstream tasks. We can also observe that the MoE models achieve performance improvements over the dense counterpart in those benchmark tasks. More importantly, the proposed CartesianMoE stands out among all MoE models, in both *MoE-Base* or *MoE-Large* settings. Specifi-

Model	Hellaswag	LAMBADA	PIQA	StoryCloze	Wino	TriviaQA	WebQs	NatrualQs
Base Model	32.54	39.22	62.52	58.52	50.75	2.99	2.22	0.72
Large Model	40.73	52.55	67.62	63.55	53.75	7.44	4.97	1.78
MoE-Base								
SMoE-Share	37.87	47.35	**66.76**	61.52	51.14	6.11	**4.18**	1.39
SMoE-Top3	37.43	46.42	66.38	61.25	51.14	5.82	4.04	1.16
Hash Layer	33.39	43.08	61.59	57.88	50.20	2.85	1.57	0.61
Fine-grained Routing	37.50	45.10	66.49	61.20	51.22	6.12	3.79	1.30
TopP Routing	35.10	45.02	63.76	58.10	51.46	4.12	2.41	0.55
CartesianMoE	**38.17**	**48.17**	65.83	**62.53**	**51.62**	**7.23**	**4.18**	**1.58**
MoE-Large								
SMoE-Share	48.25	58.80	**70.51**	66.49	55.56	14.91	9.06	3.99
SMoE-Top3	47.59	57.60	70.40	65.95	53.75	14.46	7.92	4.24
Hash Layer	42.80	56.45	66.32	64.08	52.33	6.66	4.33	1.86
Fine-grained Routing	48.10	58.32	69.86	64.89	55.80	14.40	8.42	3.77
TopP Routing	44.31	**59.85**	67.46	63.92	54.54	11.02	6.15	2.41
CartesianMoE	**49.14**	59.17	70.24	**67.02**	**56.51**	**15.09**	**9.15**	**4.85**

Table 2: Performances of language models on downstream tasks. The best score is marked in **bold**.

cally, compared to other MoE models, CartesianMoE yields the best performance for 7 of 8 benchmarks in the *MoE-Base* setting and 6 of 8 benchmarks in the *MoE-Large* setting.

Particularly, against *Fine-grained Routing* with flattened fine-grained experts, CartesianMoE excels in 7 of 8 benchmarks with the *MoE-Base* setting and also excels in all benchmarks with the *MoE-Large* setting. That further verifies our analysis above that CartesianMoE can enjoy deeper representations of input and more flexible routing than the flattened counterpart. It also demonstrates the effectiveness of introducing the *Cartesian Product Layer* for group-wise knowledge sharing.

6 Analyses

6.1 Impact of Fixed-Activated Shared Expert

Under the *MoE-Large* setting, we remove the fixed-activated shared experts from CartesianMoE to investigate its impact on the model performance.

As shown in Table 3, after removing the fixed-activated shared experts (i.e., *w/o Shared Expert*), CartesianMoE yields slightly better performance than *Fine-grained Routing* equipped with shared experts. The result well reflects the effectiveness of group-wise knowledge sharing among experts proposed by CartesianMoE, which is equally important as global knowledge sharing introduced by shared experts. Moreover, when CartesianMoE is equipped with shared experts as by default, its performance is substantially enhanced, which further demonstrates the effectiveness of forming a "global shared knowledge + group-wise shared knowledge + expert-specific knowledge" system, as proposed by CartesianMoE.

Model	PPL
SMoE-Top3	6.18
Fine-grained Routing	6.16
CartesianMoE	**6.08**
w/o Shared Expert	6.15

Table 3: Impact of the fixed-activated shared expert.

6.2 Analysis on Expert Routing Robustness

To analyze the expert routing robustness of different MoE models, we disable the top-1 routed expert and then evaluate the PPL variance brought by such a routing change on the Pile validation set. Specifically, for each token, we mask the expert with the highest routing probability and then select the top K experts from the remaining ones. Since each *Cartesian Product Layer* in CartesianMoE has two MoE sub-layers, we randomly select one sub-layer each time and mask the corresponding top-1 expert.

As shown in Table 4, even with the top-1 routed expert disabled, CartesianMoE still yields the lowest PPL, and enjoys a much smaller PPL variance, compared to other MoE methods. That well indicates the superior routing robustness of CartesianMoE. And we attribute it to the more thorough knowledge sharing among experts in CartesianMoE, which includes both global and group-wise knowledge sharing.

6.3 Training with More Tokens

The previous experiments are conducted using 100B tokens. To investigate whether the superiority of the proposed CartesianMoE can be maintained after training with more tokens, here we continue to train CartesianMoE and the most competitive baseline *Fine-grained Routing* (Dai et al., 2024)

	PPL	PPL(disable top-1)
SMoE-Share	6.13	7568.85
SMoE-Top3	6.18	847.46
Fine-grained Routing	6.16	3095.44
TopP Routing	6.14	84.42
CartesianMoE	**6.08**	**27.72**

Table 4: PPL on the Pile validation set, with the top-1 routed expert disabled. The baseline *Hash Layer* is excluded here, as the experts for each input in it are fixedly assigned.

	MoE-Large		7.25B Params	
	CartesianMoE	Fine-grain	CartesianMoE	Fine-grain
Hellaswag	**54.28**	52.10	**56.96**	55.64
LAMBADA	**63.05**	61.55	**63.50**	62.83
PIQA	**71.71**	71.21	**73.99**	71.87
StoryCloze	67.93	**67.97**	**69.37**	69.11
Wino	**56.74**	55.88	**58.17**	58.16
TriviaQA	20.03	**21.40**	**24.87**	22.82
WebQs	**10.58**	9.01	**11.96**	11.32
NatrualQs	**5.87**	5.54	**6.57**	5.54
PPL(↓)	**5.69**	5.78	**4.92**	4.99

Table 5: The performance comparison after training 400B tokens with different model sizes, with *Fine-grain* being short for the baseline *Fine-grained Routing*. The best score in each setting is marked in **bold**.

until 400B tokens, and compare their performance in the *MoE-Large* setting.

As shown in the left part of Table 5, on the Pile validation set, the PPL of *Fine-grained Routing* converged to 5.78, while that of CartesianMoE further decreases to 5.69. And on downstream tasks, CartesianMoE also outperforms *Fine-grained Routing* in 6 out of 8 benchmarks. The full changing curves for PPL and benchmark performance of both MoE models are provided in Figure 2 and Figure 4 in the Appendix, respectively. It can be seen that even trained on more tokens, CartesianMoE consistently maintains superior performance, well demonstrating its effectiveness.

6.4 Scaling Up the Model Size

To investigate the performance of the proposed CartesianMoE with a larger model size, we follow the setting of (Muennighoff et al., 2024) to train CartesianMoE and the most competitive baseline *Fine-grained Routing*, with 7.25B parameters and 1.61B activated parameters. The specific parameter settings are provided in Appendix 9.2.

As shown in the right part of Table 5, on the Pile validation set, the PPL of *Fine-grained Routing* converged to 4.99, while that of CartesianMoE decreases to 4.92. And on all downstream tasks, CartesianMoE outperforms *Fine-grained Routing*.

D	K	m	Mode	PPL
3072	2	1	Full-sized	7.37
1536	4	2	Fine-grained Routing	7.33
768	8	4	Fine-grained Routing	7.34
1536	2+2	2	CartesianMoE	**7.19**
768	4+4	4	CartesianMoE	7.26

Table 6: PPL on the Pile validation set, with different expert granularity. D indicates the FFN intermediate size, K denotes the number of activated experts, and m denotes the splitting factor.

The full changing curves for PPL and downstream tasks of both MoE models are also provided in Figure 3 and Figure 5 in the Appendix. The experimental results further demonstrate the superiority and scalability of CartesianMoE.

6.5 Training in Different Expert Granularities

The experiments above use half-sized FFNs as fine-grained experts in CartesianMoE. It would be interesting to see whether CartesianMoE can maintain its superiority with more finer-grained experts. Suppose we have N full-sized experts. As mentioned before, to keep the numbers of total parameters and activated parameters unchanged, we equally split each full-sized expert into m fine-grained experts via splitting its FFN intermediate size into m equal parts, with m being the splitting factor, and the number of activated fine-grained experts would also be scaled up by m. It can be seen $m = 1$ for full-sized experts, and experiments above use $m = 2$ for CartesianMoE. Here we further conduct experiments with $m = 4$, for both CartesianMoE and the most competitive baseline *Fine-grained Routing*, to further validate CartesianMoE.

As is seen in Table 6, in both $m = 2$ and $m = 4$ settings, CartesianMoE consistently outperforms *Fine-grained Routing* in terms of PPL on the Pile validation set, which further demonstrates its superiority and robustness across different expert granularities. We also find that increasing m may not lead to better performance, as over-fine-grained experts can encounter underfitting.

7 Conclusions

Inspired by collective matrix factorization to capture shared knowledge within data, we introduce CartesianMoE, a "multiplication"-manner knowledge sharing method among experts in MoE models. CartesianMoE categorizes fine-grained sub-experts into two distinct sets, and uses their Cartesian product to build experts that facilitate group-

wise knowledge sharing. Equipped with shared experts as previous works, CartesianMoE builds a more thorough knowledge sharing system among experts, i.e., "global shared knowledge + group-wise shared knowledge + expert-specific knowledge". Extensive experiments well demonstrate that CartesianMoE outperforms previous MoE models across various settings, in terms of language modeling perplexity and downstream task performance. It also presents much better routing robustness due to enhanced knowledge sharing.

8 Limitations

We only perform Cartesian product computations between two MoE sub-layers. In fact, the Cartesian product can be extended to more than two sub-layers. However, (Dubey et al., 2024) has shown that increasing the number of model sub-layers requires a corresponding increase in hidden state dimensionality to ensure training effectiveness. And thus we leave the exploration of extending to more MoE sub-layers for future work.

References

Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Slav Petrov, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy P. Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul Ronald Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, and et al. 2023. Gemini: A family of highly capable multimodal models. CoRR, abs/2312.11805.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL, pages 1533–1544. ACL.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. In International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA, volume 202 of Proceedings of Machine Learning Research, pages 2397–2430. PMLR.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. PIQA: reasoning about physical commonsense in natural language. In The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, pages 7432–7439. AAAI Press.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. CoRR, abs/1904.10509.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways. CoRR, abs/2204.02311.

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. CoRR, abs/2401.06066.

Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Stablemoe: Stable routing strategy for mixture of experts. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022, pages 7085–7095. Association for Computational Linguistics.

DeepSeek-AI, Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, Hao Zhang, Hanwei Xu, Hao Yang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jin Chen, Jingyang Yuan, Junjie Qiu, Junxiao Song, Kai Dong, Kaige Gao, Kang Guan, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruizhe Pan, Runxin Xu, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Size Zheng, Tao Wang, Tian Pei, Tian Yuan, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaosha Chen, Xiaotao Nie, and Xiaowen Sun. 2024. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. CoRR, abs/2405.04434.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), pages 4171–4186. Association for Computational Linguistics.

Nan Du, Yanping Huang, Andrew M. Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten P. Bosma, Zongwei Zhou, Tao Wang, Yu Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen S. Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc V. Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. 2022. Glam: Efficient scaling of language models with mixture-of-experts. In International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA, volume 162 of Proceedings of Machine Learning Research, pages 5547–5569. PMLR.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The llama 3 herd of models. CoRR, abs/2407.21783.

David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. 2013. Learning factored representations in a deep mixture of experts. arXiv preprint arXiv:1312.4314.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. J. Mach. Learn. Res., 23:120:1–120:39.

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. 2021. The pile: An 800gb dataset of diverse text for language modeling. CoRR, abs/2101.00027.

Alex Graves. 2013. Generating sequences with recurrent neural networks. CoRR, abs/1308.0850.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan

Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training compute-optimal large language models. CoRR, abs/2203.15556.

Quzhe Huang, Zhenwei An, Nan Zhuang, Mingxu Tao, Chen Zhang, Yang Jin, Kun Xu, Liwei Chen, Songfang Huang, and Yansong Feng. 2024a. Harder tasks need more experts: Dynamic routing in moe models. CoRR, abs/2403.07652.

Quzhe Huang, Zhenwei An, Nan Zhuang, Mingxu Tao, Chen Zhang, Yang Jin, Kun Xu, Kun Xu, Liwei Chen, Songfang Huang, and Yansong Feng. 2024b. Harder task needs more experts: Dynamic routing in MoE models. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 12883–12895, Bangkok, Thailand. Association for Computational Linguistics.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. 1991. Adaptive mixtures of local experts. Neural Comput., 3(1):79–87.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers, pages 1601–1611. Association for Computational Linguistics.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. CoRR, abs/2001.08361.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. 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. Trans. Assoc. Comput. Linguistics, 7:452–466.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. Gshard: Scaling giant models with conditional computation and automatic sharding. In 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net.

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. 2021. BASE layers: Simplifying training of large, sparse models. In Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event, volume 139 of Proceedings of Machine Learning Research, pages 6265–6274. PMLR.

Haoran Lian, Yizhe Xiong, Jianwei Niu, Shasha Mo, Zhenpeng Su, Zijia Lin, Peng Liu, Hui Chen, and Guiguang Ding. 2024. Scaffold-bpe: Enhancing byte pair encoding with simple and effective scaffold token removal. arXiv preprint arXiv:2404.17808.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James F. Allen. 2016. A corpus and evaluation framework for deeper understanding of commonsense stories. CoRR, abs/1604.01696.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, et al. 2024. Olmoe: Open mixture-of-experts language models. arXiv preprint arXiv:2409.02060.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers. The Association for Computer Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. OpenAI blog, 1(8):9.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, H. Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant M. Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew J. Johnson, Blake A.

Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Edward Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. 2021. Scaling language models: Methods, analysis & insights from training gopher. CoRR, abs/2112.11446.

Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. 2022. Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation AI scale. In International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA, volume 162 of Proceedings of Machine Learning Research, pages 18332–18346. PMLR.

Stephen Roller, Sainbayar Sukhbaatar, Arthur Szlam, and Jason Weston. 2021. Hash layers for large sparse models. In Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual, pages 17555–17566.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, pages 8732–8740. AAAI Press.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net.

Ajit P. Singh and Geoffrey J. Gordon. 2008. Relational learning via collective matrix factorization. In Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08, page 650–658, New York, NY, USA. Association for Computing Machinery.

Zhenpeng Su, Zijia Lin, Xue Bai, Xing Wu, Yizhe Xiong, Haoran Lian, Guangyuan Ma, Hui Chen, Guiguang Ding, Wei Zhou, et al. 2024a. Maskmoe: Boosting token-level learning via routing mask in mixture-of-experts. arXiv preprint arXiv:2407.09816.

Zhenpeng Su, Zijia Lin, Baixue Baixue, Hui Chen, Songlin Hu, Wei Zhou, Guiguang Ding, and Xing W. 2024b. MiLe loss: a new loss for mitigating the bias of learning difficulties in generative language models. In Findings of the Association for Computational Linguistics: NAACL 2024, pages 250–262, Mexico City, Mexico. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. CoRR, abs/2302.13971.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 5998–6008.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. 2023. Doremi: Optimizing data mixtures speeds up language model pretraining. In Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023.

Yizhe Xiong, Xiansheng Chen, Xin Ye, Hui Chen, Zijia Lin, Haoran Lian, Jianwei Niu, and Guiguang Ding. 2024. Temporal scaling law for large language models. arXiv preprint arXiv:2404.17785.

Yuanhang Yang, Shiyi Qi, Wenchao Gu, Chaozheng Wang, Cuiyun Gao, and Zenglin Xu. 2024. XMoE: Sparse models with fine-grained and adaptive expert selection. In Findings of the Association for Computational Linguistics ACL 2024, pages 11664–11674, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers, pages 4791–4800. Association for Computational Linguistics.

Hao Zhao, Zihan Qiu, Huijia Wu, Zili Wang, Zhaofeng He, and Jie Fu. 2024a. HyperMoE: Towards better mixture of experts via transferring among experts. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 10605–10618, Bangkok, Thailand. Association for Computational Linguistics.

Hao Zhao, Zihan Qiu, Huijia Wu, Zili Wang, Zhaofeng He, and Jie Fu. 2024b. Hypermoe: Towards better

mixture of experts via transferring among experts. arXiv preprint arXiv:2402.12656.

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. arXiv preprint arXiv:2202.08906.

9 Appendix

9.1 Compared Models

The model settings we compare are as follows. For MoE models, following (Lepikhin et al., 2021; Yang et al., 2024), unless otherwise specified, each layer of the MoE has 16 experts, with the top-2 experts activated.

- **Dense** represents a standard Transformer language model.
- **SMoE-Share** denotes an MoE model similar to (Lepikhin et al., 2021; Fedus et al., 2022), without fine-grained splitting of experts. Additionally, each MoE layer in *SMoE-Share* includes 1 shared expert.
- **SMoE-Top3** denotes an MoE model with top-3 routing and no shared experts. To maintain the total number of parameters after removing the shared expert, *SMoE-Top3* increases the intermediate dimensionality of each expert's FFN, which results in slightly more activated parameters compared to other models and acts as a stronger baseline for comparison.
- **Hash Layer** (Roller et al., 2021) signifies a method without *router* parameters, where each token is fixedly assigned to two experts using a random hash. The model also has a shared expert for fair comparison with the other models.
- **Fine-grained Routing** denotes an MoE model that employs a *Fine-grained Routing* strategy (Dai et al., 2024). For both routing and shared experts, we split the fully-sized FFNs into 2 half-sized FFNs, resulting in 32 fine-grained experts per MoE layer. To maintain the total number of activated parameters consistent, the *Fine-grained Routing* strategy uses top-4 routing and includes 2 fixed-activated shared experts for each MoE layer.
- **TopP Routing** (Huang et al., 2024a) is a routing strategy that dynamically adjusts the number of activated experts based on the difficulty of tokens. It selects the top experts until their cumulative confidence exceeds the pre-set confidence threshold top$P$. Following (Huang et al., 2024a), we set top$P$ as 0.4. Similarly, each MoE layer includes one shared expert to enable fair comparison with other models.

9.2 Training Configuration

	CartesianMoE	Fine-grained Routing
Hidden Size	2,048	2,048
Activation	SwiGLU	SwiGLU
Intermediate Size	2,048	2,048
Attn heads	16	16
Num layers	16	16
Layer norm type	RMSNorm	RMSNorm
Pos emb.	RoPE	RoPE
MoE layers	Every	Every
Shared Experts	True	True
Fine-grained Experts	True	True
Max seq len	4096	4096
MoE sub-layers	2	N/A
# Experts	32	32
# Activated Expert	Top$K$=(2+2)	Top$K$=4
# Params	7.25B	7.25B
# Activated Params	1.61B	1.61B

Table 7: Configurations of CartesianMoE and *Fine-grained Routing* with 7.25B parameters.

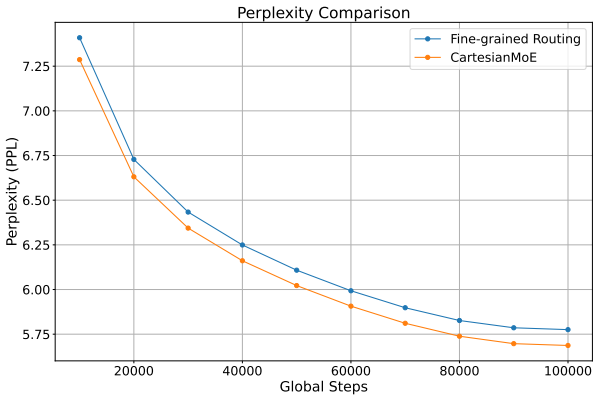


Figure 2: PPL changing curves during language model training with 400B tokens for CartesianMoE and *Fine-grained Routing* in *MoE-Large* setting.

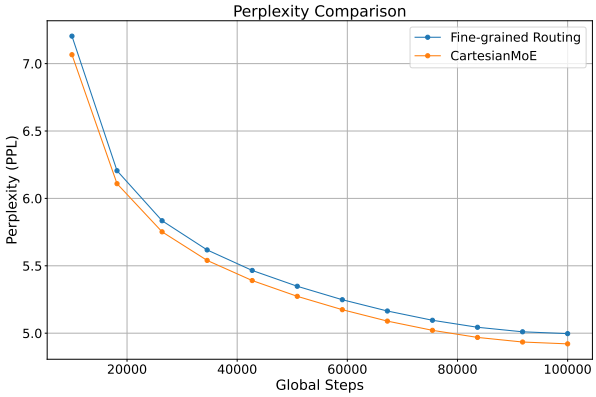


Figure 3: PPL changing curves during language model training with 400B tokens for CartesianMoE and *Fine-grained Routing* with 7.25B parameters.

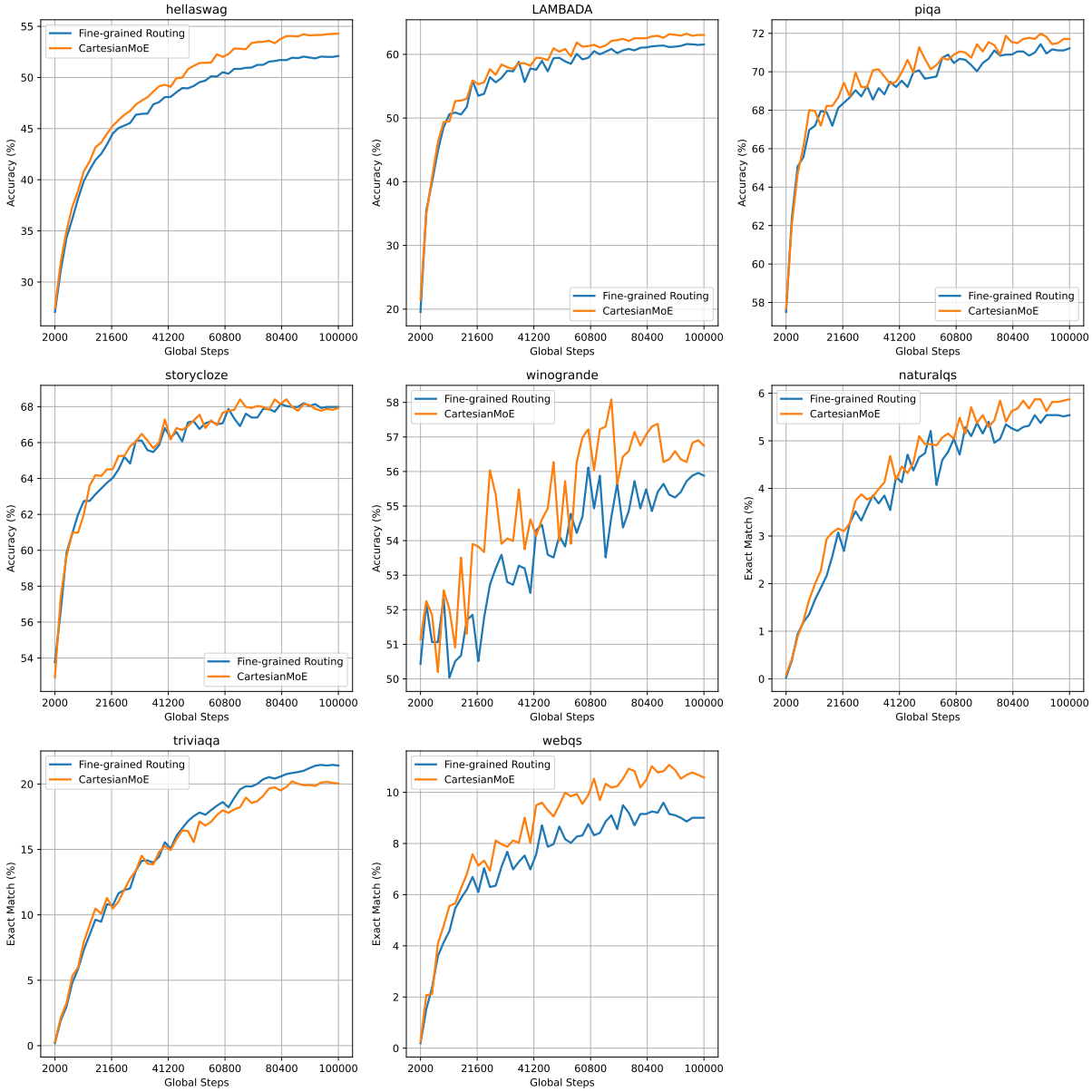


Figure 4: Changing curves of downstream task performance during language model training with 400B tokens for CartesianMoE and *Fine-grained Routing* in *MoE-Large* setting.

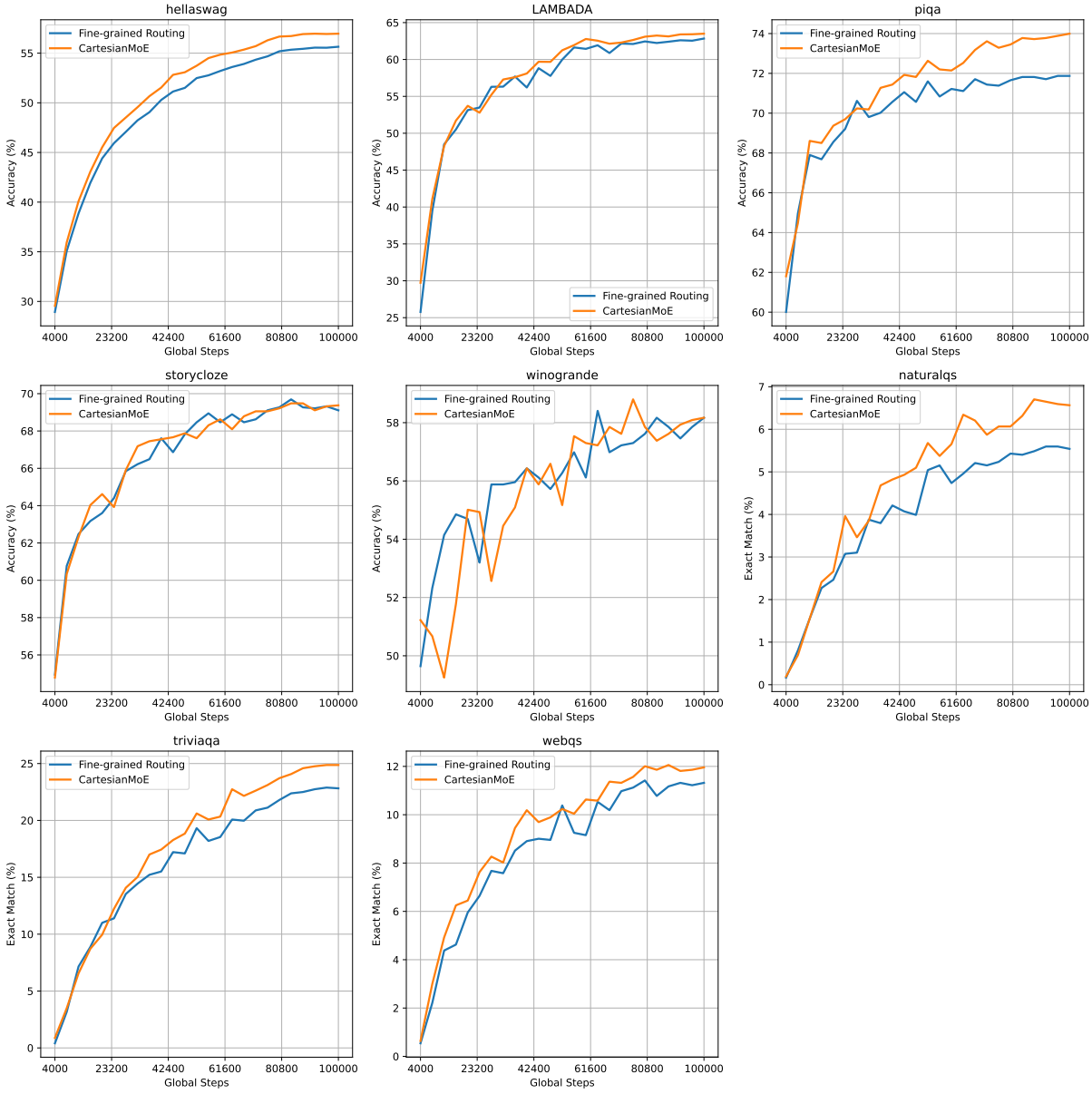


Figure 5: Changing curves of downstream task performance during language model training with 400B tokens for CartesianMoE and *Fine-grained Routing* with 7.25B parameters.