On the Role of Discrete Representation in Sparse Mixture of Experts

Anonymous authors
Paper under double-blind review

Abstract

Sparse Mixture of Experts (SMoE) is an effective solution for scaling up model capacity without increasing the computational costs. A crucial component of SMoE is the router, responsible for directing the input to relevant experts; however, it also presents a major weakness, leading to routing inconsistencies and representation collapse issues. Instead of fixing the router like previous works, we propose an alternative that assigns experts to input via *indirection*, which employs the discrete representation of input that points to the expert. The discrete representations are learned via vector quantization, resulting in a new architecture dubbed Vector-Quantized Mixture of Experts (VQMoE). We provide theoretical support and empirical evidence demonstrating the VQMoE's ability to overcome the challenges present in traditional routers. Through extensive evaluations on both large language models and vision tasks for pre-training and fine-tuning, we show that VQMoE achieves a 28% improvement in robustness compared to other SMoE routing methods while maintaining strong performance in fine-tuning tasks.

1 Introduction

Scaling Transformer models with increasing data and computational resources has led to remarkable advances across a wide range of domains, including natural language processing (NLP) (Du et al., 2022; Fedus et al., 2022; Zhou et al., 2024) and visual representation learning (Riquelme et al., 2021a; Shen et al., 2023b). Despite these successes, training and deploying large-scale dense Transformer models often require substantial computational resources, frequently amounting to hundreds of thousands of GPU hours and incurring costs in the millions of dollars (Kaddour et al., 2023). To address this scalability bottleneck, Sparse Mixture of Experts (SMoE) architectures have emerged as a promising alternative (Shazeer et al., 2017; Zoph et al., 2022; Xue et al., 2024; Jiang et al., 2024). Inspired by classical mixture-of-experts formulations (Jacobs et al., 1991a), SMoE models consist of multiple expert subnetworks with shared architectures, where a routing mechanism dynamically selects a small subset of experts (often one or two) for each input token. This sparsity significantly reduces inference costs compared to dense counterparts of similar model capacity (Artetxe et al., 2022; Krajewski et al., 2024), making SMoEs attractive for efficient scaling.

Despite their efficiency benefits, SMoEs face critical training challenges, most notably, *representation collapse*. This phenomenon occurs when only a small subset of experts are frequently activated, or when all experts converge to similar representations, thereby negating the diversity and specialization that the architecture is intended to promote. Prior works have sought to mitigate this issue by improving the routing policy through regularization and auxiliary losses (Chi et al., 2022; Chen et al., 2023a; Do et al., 2023). However, these approaches focus on the routers improvement rather than questioning its necessity.

In this work, we explore a more fundamental question: *Is an explicit router necessary at all?* We argue that incorporating discrete representations offers a principled alternative. Discrete latent variables are inherently suited to capturing structured and interpretable patterns within data, aligning with the symbolic nature of human cognition, where concepts are often discretized as words, tokens, or categories. In the SMoE context, discrete representations can improve input routing by naturally clustering similar inputs, thereby enhancing expert specialization and utilization without relying solely on a learned gating mechanism.

Employing vector quantization (VQ) techniques to learn discrete representation, this paper proposes a novel mixture of expert framework, named VQMoE, which overcomes the representation collapse and inconsistency in training sparse mixture of experts. More specifically, we prove that the existing router methods are inconsistent and VQMoE suggests an optimal expert selection for training SMoE. Additionally, our method guarantees superior SMoE training strategies compared to the existing methods by solving the representation collapse by design.

We evaluate the proposed method by conducting pre-training of Large Language Models (LLMs) on several advanced SMoE architectures, such as SMoE (Jiang et al., 2024), StableMoE (Dai et al., 2022), or XMoE (Chi et al., 2022), followed by fine-tuning on downstream tasks on both Language and Vision domains.

In summary, the primary contributions of this paper are as follows:

- We theoretically demonstrate that learning discrete representations provides an effective mechanism for expert selection, and that VQMoE intrinsically mitigates the problem of representation collapse.
- We propose the use of vector quantization (VQ) to learn structured and interpretable expert clusters.
- We conduct extensive experiments on large language models as well as vision pre-training and fine-tuning tasks to validate the effectiveness of our method.
- We provide a comprehensive analysis of VQMoE's behavior, offering insights into its performance and robustness.

2 Related Work

Sparse Mixture of Experts (SMoE). Sparse Mixture of Experts (SMoE) builds on the Mixture of Experts (MoE) framework introduced by Jacobs et al. (1991b); Jordan & Jacobs (1994), with the core idea that only a subset of parameters is utilized to process each example. This approach was first popularized by Shazeer et al. (2017). SMoE's popularity surged when it was combined with large language models based on Transformers (Zhou et al., 2022; Li et al., 2022; Shen et al., 2023a), and its success in natural language processing led to its application across various fields, such as computer vision (Riquelme et al., 2021b; Hwang et al., 2023; Lin et al., 2024), speech recognition (Wang et al., 2023; Kwon & Chung, 2023), and multi-task learning (Ye & Xu, 2023; Chen et al., 2023b).

However, SMoE faces a major problem in training known as representation collapse, i.e., the experts converge to similar outputs. To address this, various methods have been introduced. XMoE (Chi et al., 2022) calculates routing scores between tokens and experts on a low-dimensional hypersphere. SMoE-dropout (Chen et al., 2023a) uses a fixed, randomly initialized router network to activate experts and gradually increase the number of experts involved to mitigate collapse. Similarly, HyperRouter (Do et al., 2023) utilizes HyperNetworks (Ha et al., 2016) to generate router weights, providing another pathway for training SMoE effectively. StableMoE (Dai et al., 2022) introduces a balanced routing approach where a lightweight router, decoupled from the backbone model, is distilled to manage token-to-expert assignments. The StableMoE strategy ensures stable routing by freezing the assignments during training, while SimSMoE Do et al. (2024) forces experts to learn dissimilar representations. Despite these extensive efforts, the representation collapse issue persists, as highlighted by Pham et al. (2024). While most solutions focus on improving routing algorithms, our approach takes a different path by learning a discrete representation of input that points to relevant experts.

Discrete Representation. Discrete representations align well with human thought processes; for example, language can be understood as a series of distinct symbols. Nevertheless, the use of discrete variables in deep learning has proven challenging, as evidenced by the widespread preference for continuous latent variables in most current research. VQVAE (van den Oord et al., 2017) implements discrete representation in Variational AutoEncoder (VAE) (Kingma & Welling, 2022) using vector quantisation (VQ). IMSAT (Hu et al., 2017) attains a discrete representation by maximizing the information-theoretic dependency between data and their predicted discrete representations. Recent works follow up the vector quantisation ideas and make some enhancements for VAE, for example: (Yu et al., 2022); (Mentzer et al., 2023); and (Yang et al., 2023).

Mao et al. (2022) utilize a discrete representation to strengthen Vision Transformer (ViT) (Dosovitskiy et al., 2021). To the best of our knowledge, our paper is the first to learn a discrete representation of Sparse Mixture of Experts.

3 Method

We propose a novel model, Vector-Quantized Mixture of Experts (VQMoE), which learns discrete representations for expert selection. As illustrated in Fig. 1a, our approach selects experts directly based on the input representation, eliminating the need for a trained router. To prevent information loss, we integrate discrete and continuous representations within the model.

3.1 Preliminaries

Sparse Mixture of Experts. Sparse Mixture of Experts (SMoE) is often a transformer architecture that replaces the MLP layers in standard transformers with Mixture of Experts (MoE) layers (Shazeer et al., 2017). Given $\mathbf{x} \in \mathbb{R}^{n \times d}$ as the output of the multi-head attention (MHA), the output of SMoE with N experts is a weighted sum of each expert's computation $E_i(x)$ by the router function $\mathcal{S}(x)$:

$$\begin{aligned} f_{\text{SMoE}}(\boldsymbol{x}) &= \sum_{i=1}^{N} \mathcal{S}(\boldsymbol{x})_i \cdot E_i(\boldsymbol{x}) \\ &= \sum_{i=1}^{N} \mathcal{S}(\boldsymbol{x})_i \cdot \boldsymbol{W}^2_{\text{FFN}_i} \phi \left(\boldsymbol{W}^1_{\text{FFN}_i} \boldsymbol{x} \right) \end{aligned} \tag{1}$$

Where $\mathcal{S}(x)$ is computed by $TopK$ function as equation (2) that determines the contribution of each expert to the SMoE output.

$$\begin{aligned} \mathcal{S}(\boldsymbol{x}) &= \text{TopK}(\text{softmax}(\mathcal{G}(\boldsymbol{x})), k), \\ \text{TopK}(\boldsymbol{v}, k) &= \begin{cases} \boldsymbol{v_i} & \text{if } \boldsymbol{v_i} \in \text{top } k \text{ largest of } \boldsymbol{v}, \\ -\infty & \text{otherwise.} \end{cases} \end{aligned} \tag{2}$$

Discrete Representation Learning. van den Oord et al. (2017) propose VQVAE, which uses Vector Quantisation (VQ) to learn a discrete representation. Given an input $x \in \mathbb{R}^{n \times d}$, VQVAE discretized the input into a codebook $V \in \mathbb{R}^{K \times d}$ where K is the codebook size and d is the dimension of the embedding. Let denote $z_v(x) \in \mathbb{R}^{n \times d}$ denotes the output of the VQVAE and $\mathbf{1}()$ is the indicator function. The discrete representation $z_q(x_i) = v_k, \quad \text{where} \quad k = \text{argmin}_j \left\| z_v(x_i) - v_j \right\|_2$ is achieved by vector quantizer q_θ that maps an integer z for each input x as:

$$q_\theta(z = k \mid x) = \mathbf{1} \left(k = \underset{j=1:K}{\arg\min} \left\| z_v(x) - \text{V}_j \right\|_2 \right) \tag{3}$$

3.2 Vector-Quantized Mixture of Experts (VQMoE)

Pre-training VQMoE.; Traditional Sparse Mixture of Experts (SMoE) models utilize continuous token representations and route them to experts based on learned token-expert affinity scores. We propose a novel architecture, VQMoE, that learns both continuous and discrete representations jointly during pre-training (see Figure 1a). The continuous component captures fine-grained data patterns, while the discrete component, learned via vector quantization, encodes robust latent structure useful for downstream transfer.

Let $\boldsymbol{x} \in \mathbb{R}^{n \times d}$ denote the input to the VQMoE layer (e.g., output from a multi-head attention block), and let f^{vq} denote the vector quantization operator. The VQMoE output during pre-training is defined as:

$$f^{\text{VQMoE}}(\boldsymbol{x}) = \underbrace{g_c(\boldsymbol{x}) \cdot f^{\text{SMoE}}(\boldsymbol{x})}_{\textit{(Continuous representation)}} + \underbrace{g_d(\boldsymbol{x}) \cdot \sum_{l=1}^{K} f_l^{\text{FFN}}(\tilde{\boldsymbol{x}}_l)}_{\textit{(Discrete representation)}} \tag{4}$$

where:

- $f^{\text{SMoE}}(\boldsymbol{x})$ denotes the output from a standard SMoE layer;
- f_l^{FFN} represents the l-th expert feedforward network (FFN) block;
- $\tilde{\boldsymbol{x}}_l = \boldsymbol{v}_k$ if $\boldsymbol{x}_l \in V_l$, i.e., the input $\boldsymbol{x}_l$ is assigned to the l-th codebook vector $\boldsymbol{v}_k$; otherwise, $\tilde{\boldsymbol{x}}_l = \mathbf{0}$;
- $g_c(\boldsymbol{x})$ and $g_d(\boldsymbol{x})$ are the gating weights for the continuous (SMoE) and discrete (VQ) paths, respectively, computed as:

$$\left[g_c(\boldsymbol{x}) \; g_d(\boldsymbol{x})\right] = \text{softmax}(W_g \boldsymbol{x}), \quad W_g \in \mathbb{R}^{2\times d} \tag{5}$$

Here, K is the number of vector quantization codebooks, and $\boldsymbol{v}_k$ is a learned codebook vector assigned by f^{vq}.

Fine-tuning VQMoE.; Based on insights from Geva et al. (2021), which note that feed-forward layers (FFNs) constitute a significant portion of a transformer's parameters, we adopt a lightweight fine-tuning strategy that retains only the discrete path of the VQMoE. This allows efficient adaptation while leveraging pre-trained latent representations (see Figure 1b). The fine-tuning output becomes:

$$f^{\text{VQMoE}}(\boldsymbol{x}) = \sum_{l=1}^{K} f_l^{\text{FFN}}(\tilde{\boldsymbol{x}}_l) \tag{6}$$

3.3 Training Procedure

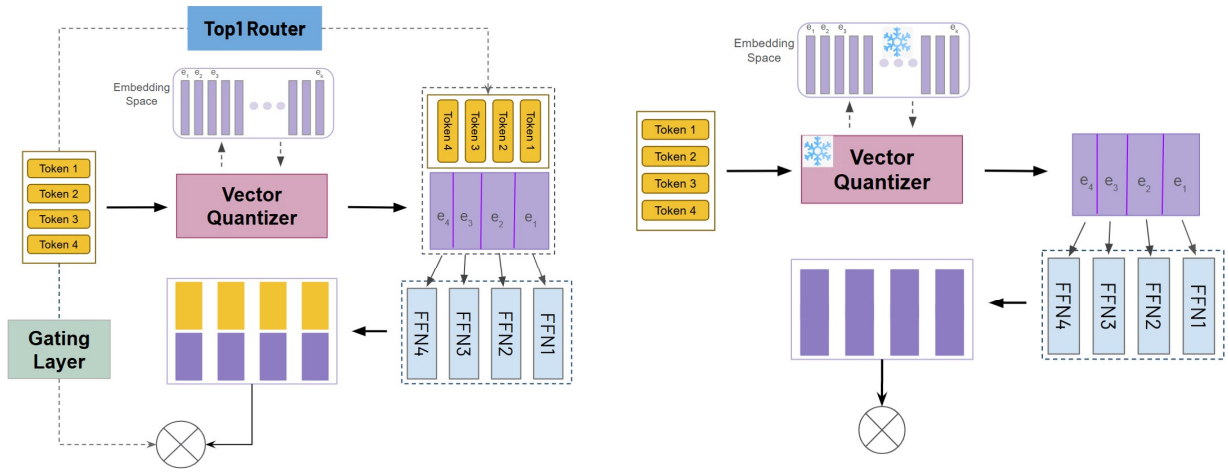
Pretraining. The training objective is jointly minimizing the loss of the target task and losses of the Vector Quantization module ($\mathcal{L}^{\text{l2}}$ and $\mathcal{L}^{\text{commitment}}$) as in (van den Oord et al., 2017). Equation 7 specifies the overall loss function for training VQMoE with three components: (1) task loss; (2) l_2 loss; (3) a commitment loss. While $\mathcal{L}^{\text{l2}}$ helps to move the embedding v_i towards the outputs $z_v(x)$, the commitment loss makes sure the output of the Vector Quantization module commits to the embedding and its output does not grow. The Vector Quantization algorithm does not vary with β, we follow $\beta = 0.25$ as van den Oord et al. (2017). We introduce a new parameter, α, to regulate the contribution of the Vector Quantization loss to the overall loss. A higher value of α favors a stronger adherence to the discrete representation, and vice versa.

$$L = \mathcal{L}_{\text{task}} \; + \alpha(\|\text{sg}\left[z_v(x)\right] - v\|_2^2 + \beta \left\|z_v(x) - \text{sg}[v]\right\|_2^2) \tag{7}$$

where $sg(.)$ is the stop gradient operator defined as follows:

$$\text{sg}(x) = \begin{cases} x & \text{forward pass} \\ 0 & \text{backward pass} \end{cases} \tag{8}$$

Fine-tuning. For downstream tasks, we fine-tune the pretraining model by utilizing the codebook learned from the Equation 7 by freezing all parameters at the Vector Quantization module. Thus, the training objective simply becomes: $L = \mathcal{L}_{\text{task}}$.



(a) VQMoE Pre-training (b) VQMoE Fine-tuning

Figure 1: Illustration of the proposed VQMoE architecture for Pre-training and fine-tuning. (a) At the Pre-training stage, VQMoE architecture learns simultaneously continuous and discrete representation at the Pre-training phase. The continuous representation is learned by the conventional SMoE, while the Vector Quantization block facilitates the learning of a discrete representation. The final output is then combined by a gate layer. (b) VQMoE learns a discrete representation that is capable of operating efficiently and robustly on downstream tasks. VQMoE computes the discrete representation only during the fine-tuning stage to achieve robustness and efficiency.

4 Theory Analysis

4.1 Optimal Experts Selection

Problem settings. We consider an MoE layer with each expert being an MLP layer which is trained by gradient descent and input data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ generated from a data distribution $\mathcal{D}$. Same as (Chen et al., 2022); (Dikkala et al., 2023), we assume that the MoE input exhibits cluster properties, meaning the data is generated from K distinct clusters $(C_1, C_2, ..., C_k)$.

Definition 4.1 (Consistent Router) *A sequence of points $x_1, x_2, \ldots, x_n$ and a corresponding sequence of clusters $C_1, C_2, \ldots, C_k$ are said to be **consistent** if, for every point $x_p \in C_i$, the condition*

$$dist(x_p, u_i) \leq \min_{j \neq i} dist(x_p, u_j)$$

is satisfied, where $dist(a, b)$ denotes the distance between a and b, and u_i is the center of cluster C_i.

Definition 4.2 (Inconsistent Router) *A sequence of points $x_1, x_2, \ldots, x_n$ and a corresponding sequence of clusters $C_1, C_2, \ldots, C_k$ are said to be **inconsistent** if there exists a point $x_p \in C_i$ such that*

$$dist(x_p, u_i) > \min_{j \neq i} dist(x_p, u_j),$$

where $dist(a, b)$ represents the distance between a and b, and u_i is the center of cluster C_i.

Inspired by (Dikkala et al., 2023), we conceptualize the router in Sparse Mixture of Experts as a clustering problem. This leads us to define a consistent router in Definition 4.1. Furthermore, we introduce a definition for an inconsistent router in SMoE as outlined in Definition 4.2, along with the concept of inconsistent expert selection presented in Theorem 4.3 during the training of SMoE.

Theorem 4.3 (Inconsistent Experts Selection) *Let f_{MHA} be a multi-head attention (MHA) function producing an output $x \in \mathbb{R}^{n \times d}$, and consider N experts with embeddings e_i for expert i where $i \in [1, N]$.*

Assume that f_{MHA} converges at step t_m, while the expert embeddings e converge at step t_e, with $t_m \gg t_e$. For each output x, the expert $K \in [1, N]$ is selected such that

$$K = \arg\min_{j \in [1,N]} dist(x, e_j).$$

Under these conditions, the expert embeddings e form an inconsistent routing mechanism.

The proof of Theorem 4.3 is given in Appendix A, and we have the following insights. Theorem 4.3 implies that an expert selection process by a router as the conventional SMoE leads to the inconsistent router. Indeed, the router layer is designed as a simple linear layer, x is the output of MHA function in practice. In practice, an SMoE router is significantly simpler than the MHA function. Consequently, this design leads to the router functioning as an inconsistent router, contributing to the representation collapse issue and instability during training.

Proposition 4.4 (Optimal Experts Selection) *Given input data partitioned into k clusters $(C_1, C_2, \ldots, C_k)$ and a mixture of experts (MoE) layer with k experts $(E_1, E_2, \ldots, E_k)$, the assignment of each cluster C_i to expert E_i for $i \in [1, k]$ constitutes an optimal expert selection solution.*

Proposition A.1 demonstrates that if we are given a clustering structure as input, assigning each part of the input to its corresponding expert results in an optimal expert selection. This implies that learning a discrete representation and directing each component to the appropriate expert yields an optimal solution. The proof of Proposition A.1 can be found in Appendix A.

4.2 Experts Representation Collapse

The *representation collapse* problem in Sparse Mixture-of-Experts (SMoE), where all experts converge to similar representations, was first highlighted by Chi et al. (2022). Following Chi et al. (2022) and Do et al. (2023), we analyze this issue using the Jacobian matrix of the model output with respect to the input $x \in \mathbb{R}^{n \times d}$. The Jacobian for SMoE is expressed as:

$$\begin{aligned} \boldsymbol{J}_{\text{SMoE}} &= \mathcal{S}(x)_k \boldsymbol{J}^{\text{FFN}} + \sum_{j=1}^{N} \mathcal{S}(x)_k (\delta_{kj} - \mathcal{S}(x)_j) \boldsymbol{E}(x)_i \boldsymbol{e}_j^\top \\ &= \mathcal{S}(x)_k \boldsymbol{J}^{\text{FFN}} + \sum_{j=1}^{N} \boldsymbol{c}_j \boldsymbol{e}_j^\top, \end{aligned} \tag{9}$$

where $\boldsymbol{c}_j = \mathcal{S}(x)_k (\delta_{kj} - \mathcal{S}(x)_j) \boldsymbol{E}(x)_i$, $\boldsymbol{J}^{\text{FFN}}$ is the Jacobian of the selected expert's feedforward network, and $\boldsymbol{e}_j$ are the expert embedding vectors. Equation 9 consists of two components:

- $\mathcal{S}(x)_k \boldsymbol{J}^{\text{FFN}}$: the main signal path from the input to the output through the selected expert.
- $\sum_{j=1}^{N} \boldsymbol{c}_j \boldsymbol{e}_j^\top$: the contribution from the gating function's gradient with respect to the expert embeddings.

Since the summation over expert embeddings lies in a subspace of dimension N, and typically $N \ll d$, this projection restricts the output space from $\mathbb{R}^d$ to $\mathbb{R}^N$, which effectively causes representation collapse.

Jacobian Analysis of VQMoE. To examine whether VQMoE mitigates this collapse, we derive the Jacobian of the VQMoE output with respect to the input $x \in \mathbb{R}^{n \times d}$. The detailed expression of the VQMoE Jacobian matrix is provided in Section A.1.1. Specifically, we have:

$$
\begin{aligned}
\boldsymbol{J}_{\text{VQMoE}} &= g_c(\boldsymbol{x}) \cdot \boldsymbol{J}_{\text{SMoE}} + \frac{\partial g_c(\boldsymbol{x})}{\partial \boldsymbol{x}} f^{\text{SMoE}}(\boldsymbol{x}) \\
&\quad + g_d(\boldsymbol{x}) \cdot \sum_{l=1}^{K} \boldsymbol{J}_l^{\text{FFN}} + \frac{\partial g_d(\boldsymbol{x})}{\partial \boldsymbol{x}} \sum_{l=1}^{K} f_l^{\text{FFN}}(\tilde{\boldsymbol{x}}_l) \\
&= J_1 + \sum_{j=1}^{N+K+2} o_j \boldsymbol{e}_j^{\top}.
\end{aligned} \tag{10}
$$

Same as the Jacobian matrix of SMoE, the Jacobian matrix of VQMoE consists two terms: (1) J_1 depends on input token and experts to the final output; (2) $\sum_{j=1}^{N+K+2} o_j \boldsymbol{e}_j^{\top}$ indicates to learn better gating function to minimize the task loss. We can see that $N + K + 2 >> N$, implying that VQMoE is better than SMoE in solving the representation collapse issue. In theory, we can choose the number of codes to be approximately $d - N - 2$ with a hashing index to experts to address the issue. However, this involves a trade-off with the computational resources required to learn the codebook.

5 Experiment

We conduct experiments to explore the following hypotheses: (i) VQMoE provides an effective SMoE training algorithm for LLMs; (ii) VQMoE delivers a robust and efficient solution during the fine-tuning phase; and (iii) VQMoE outperforms other routing methods in vision domain.

5.1 Experimental Settings

To answer the three above hypotheses, we conduct experiments on Vision, Language, and Time-series tasks. For **Pre-training language models**, we examine two common tasks in the training and evaluation of large language models: character-level language modeling using the enwik8 and text8 datasets (Mahoney, 2011), and word-level language modeling with the WikiText-103 (Merity et al., 2016) and One Billion Word datasets (Chelba et al., 2014). For **Parameter-efficient fine-tuning**, we consider pre-trained base models on enwik8 and efficient Fine-tuning it on a downstream task. We choose the SST-2 (Socher et al., 2013), SST-5 (Socher et al., 2013), IMDB (Maas et al., 2011), and BANKING77 (Casanueva et al., 2020) datasets. For **vision tasks**, we employ the Vision Transformer model (Dosovitskiy et al., 2021) with the state-of-the-art routing method and our method to train and evaluate the image classification task. Our experiments encompass five widely recognized image classification datasets: Cifar10, Cifar100 (Krizhevsky, 2009), STL-10 (Coates et al., 2011), SVHN (Netzer et al., 2011), ImageNet-1K(Deng et al., 2009).

5.2 Pre-training Language Models

Training tasks We explore two common tasks in the training and evaluation of LLMs. First, character-level language modeling on the enwik8 or text8 datasets (Mahoney, 2011), which are common datasets to evaluate the model's pre-training capabilities. We also consider the word-level language modeling task on WikiText-103 (Merity et al., 2016) and One Billion Word dataset (Chelba et al., 2014), a much larger and more challenging dataset, to test the models scaling capabilities. For all datasets, we follow the default splits of training-validation-testing. Second, we consider Fine-tuning the models on downstream applications to investigate the models' capabilities of adapting to different domains. To this end, we consider pre-trained medium models on enwik8 and Fine-tuning them on a downstream task. We choose the SST-2 (Socher et al., 2013), SST-5 (Socher et al., 2013), IMDB (Maas et al., 2011), and BANKING77 (Casanueva et al., 2020) datasets, which are common NLP tasks to evaluate pre-trained models. Following Chen et al. (2023a), we freeze the router and only optimize the experts' parameter in this experiment.

Models. For the language tasks, we follow the same settings as in SMoE-Dropout (Chen et al., 2023a). We consider two decoder-only architectures: (i) the standard Transformer (Vaswani et al., 2017); and (ii) and Transformer-XL (Dai et al., 2019a) with the same number of parameters as Transformer. We evaluate our method versus the state of art Sparse Mixture of Expert Layers such as StableMoE (Dai et al., 2022) and XMoE (Chi et al., 2022) with top$k = 2$ in the experiments. We consider two model configurations: (i) base:

Configuration		Enwik8 (BPC)		Text8 (BPC)		WikiText-103 (PPL)		lm1b (PPL)	
Architecture	Algorithm	Base	Large	Base	Large	Base	Large	Base	Large
Transformer	VQMoE	**1.48**	**1.41**	**1.47**	**1.40**	**38.74**	**31.98**	**59.48**	**49.30**
	SMoE	1.49	1.41	1.49	1.40	39.50	32.30	60.88	51.30
	SMoE-Dropout	1.82	2.22	1.70	1.89	72.62	107.18	97.45	159.09
	XMoE	1.51	1.42	1.49	1.42	39.56	32.65	61.17	51.84
	StableMoE	1.49	1.42	1.49	1.41	39.45	32.34	60.72	50.74
Transformer-XL	VQMoE	**1.19**	**1.08**	**1.28**	**1.17**	**29.48**	**23.85**	**56.85**	**48.70**
	SMoE	1.20	1.09	1.29	1.18	30.16	24.02	58.00	48.71
	SMoE-Dropout	1.56	2.24	1.56	1.86	58.37	40.02	93.17	68.65
	XMoE	1.21	1.09	1.28	1.17	30.34	24.22	58.33	50.64
	StableMoE	1.20	1.10	1.28	1.19	29.97	24.19	58.25	49.17
# Params		20M	210M	20M	210M	20M	210M	20M	210M

Table 1: BPC on the enwik-8 and text8 test sets; and perplexity on the Wikitext-103 and One Billion Word test sets. Lower is better, best results are in bold.

with four SMoE blocks and **20M** parameters; (ii) large: with twelve SMoE layers and **210M** parameters. We emphasize that we are not trying to achieve state-of-the-art results due to the limited resource constraints. Instead, we evaluate the small and large models on various datasets to demonstrate the scalability and efficacy of our algorithm. Lastly, we conduct extensive investigations using the tiny model to understand the algorithm behaviours and their robustness to different design choices.

Baselines. We compare our VQMoE with state-of-the-art SMoE training strategies for LLMs. **SMoE** (Jiang et al., 2024) employs a simple router trained end-to-end with the experts. **StableMoE** (Dai et al., 2022) proposes a two-phase training process where the first phase trains only the router, and then the router is fixed to train the experts in the second phase. **XMoE** (Chi et al., 2022) implements a deep router that comprises a down-projection and normalization layer and a gating network with learnable temperatures. Lastly, motivated by SMoE-Dropout (Chen et al., 2023a), we implement the **SMoE-Dropout** strategy that employs a randomly initialized router and freeze it throughout the training process.

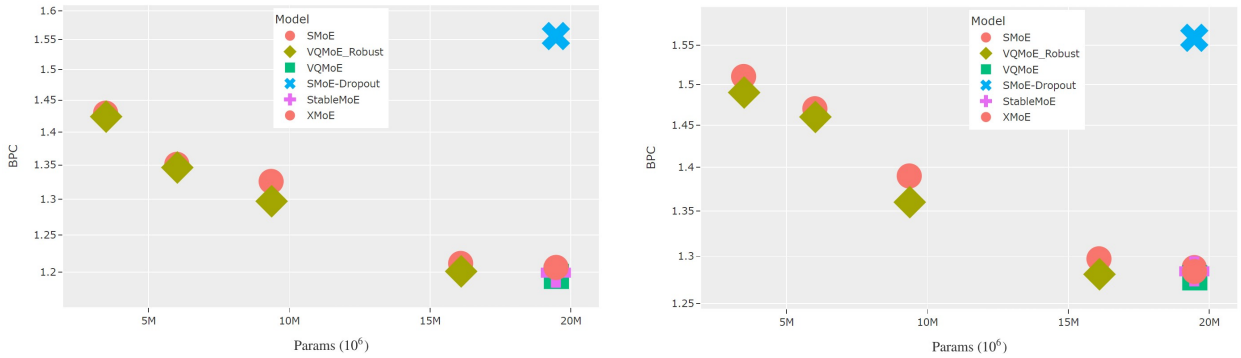
Training procedure. For the language modeling experiments, we optimize the base models and the large models for 100,000 steps. We use an Adam (Kingma & Ba, 2017) optimizer with a Cosine Annealing learning rate schedule (Loshchilov & Hutter, 2017). The lowest validation loss checkpoint is used to report the final performance on the test set.

Q1: Does VQMoE perform better on Pre-training tasks compared to routing methods? A1: Yes.

Table 1 presents the evaluation metrics comparing VQMoE with state-of-the-art approaches. We also show the performance progression of the base model on the validation set. Notably, across all methods, the Transformer-XL architecture consistently outperforms the standard Transformer on all datasets. While advanced strategies like XMoE and StableMoE tend to surpass vanilla SMoE when model complexity is increased (from small to medium) or more data is introduced (moving from enwik8 to WikiText-103 or One Billion Word), these improvements are often inconsistent or marginal. In contrast, VQMoE consistently outperforms all competitors across benchmarks (keeping in mind that the BPC metric is log-scaled), architectures, and also converges more quickly. This highlights VQMoE's effectiveness in learning an efficient routing policy for the language modeling pre-training task.

Q2: Does VQMoE keep outperforming the router method when scaling up? A2: Yes.

Table 1 also demonstrates that VQMoE maintains consistently strong performance when scaled up to 12-layer Transformer and Transformer-XL architectures. Across all four datasets, the performance gap between VQMoE and other routing methods widens as the dataset size increases, from enwik8 to the One Billion



(a) Robust VQMoE Benchmark (Enwik8)

(b) Robust VQMoE Benchmark (Text8)

Figure 2: Illustration of the proposed Robust VQMoE architecture for Pre-training on Enwik8 and Text8 dataset. (a) Robust VQMoE architecture achieves the same performance with the routing methods while only using 80% of the parameters on Enwik8 dataset. (b) Roubust VQMoE demonstrates robustness on the Text8 dataset. Bits-per-character (BPC) on the Enwik8 and Text8 datasets, and lower is better.

Architecture	FLOPs(x10^{10})	Transformer				Transformer-XL			
Dataset		SST-2	SST-5	IMDB	BANKING77	SST-2	SST-5	IMDB	BANKING77
VQMoE	**5.6145**	**82.6**	**41.1**	**89.5**	**84.8**	**83.3**	**42.0**	**89.1**	**85.3**
SMoE	7.7620	82.1	39.5	89.3	82.6	80.8	40.4	88.6	80.2
SMoE-Dropout	7.7620	81.3	39.6	88.9	77.9	81.8	40.0	89.1	77.3
XMoE	7.7620	82.4	39.9	89.0	83.1	81.3	40.3	88.7	82.7
StableMoE	7.7620	82.2	40.4	89.1	82.7	82.5	41.1	88.5	78.6

Table 2: Accuracy of the model after fine-tuned on various datasets. Higher is better, best results are in bold.

Word dataset. This suggests that our approach has the potential to scale effectively with larger language models and bigger datasets. An interesting observation is that SMoE-Dropout (Chen et al., 2023a) performs the worst among all methods, indicating that a random routing policy is insufficient and requires updating for effective training. This finding highlights that the success of SMoE-Dropout is largely due to its self-slimmable strategy, which linearly increases the number of activated experts (K) during training. However, this approach transforms the sparse network into a dense one, contradicting the original motivation behind using SMoE for large-scale models.

Q3: When does VQMoE outperform router methods in terms of robustness? A3: The lower hidden size of FFN.

Compared to the routing methods, VQMoE achieves competitive performance which only requires 80% number of parameters. Figure 2a and Figure 2b demonstrate the robustness of our method on the Enwik8 and Text8 datasets, respectively.

5.3 Parameter-Efficient Fine-Tuning

Q4: What is the biggest advantage of SMoE, compared to the conventional SMoE? A4: Parameter-Efficient Fine-Tuning.

We see that the discrete representation that VQMoE learns at the Pretraning stage 5.2 might consist of rich knowledge. To test this hypothesis, we use only the discrete representation for downstream tasks, allowing VQMoE to **save 28%** of computational resources compared to SMoE. Table 2 reports the accuracy of the models fine-tuned on the test sets of various datasets. Overall, we observe that VQMoE demonstrates strong transfer learning capabilities by achieving the highest accuracy on all datasets. Notably, on the more challenging datasets of SST-5 and BANKING77, which have fewer training samples or more classes,

we observe larger performance gains from VQMoE versus the remaining baselines (over 5% improvements compared to the second-best method). This result shows that VQMoE can learn a discrete representation that is not only good for pre-training but also exhibits strong transfer capabilities to various downstream tasks.

5.4 Vision

Q5: Can VQMoE compete with SMoE in the Vision domain? A5: Yes.

To make our performance comparison informative and comprehensive, we consider two kinds of baselines that are fairly comparable to VQMoE: (1) Dense Model (Vision Transformer) (Dosovitskiy et al., 2021); (2) SoftMoE (Puigcerver et al., 2024) - the most advanced MoE in Vision domain. We perform two configurations for training the Mixture of Experts: (1) small - *10 million parameters (10M)*; (2) *large - 110 million parameters (110M)*. The result at Table 3 shows that VQMoE outperforms both Vision Transformer Dense (Dosovitskiy et al., 2021), SoftMoE (Puigcerver et al., 2024), and other routing methods such as (Dai et al., 2022), (Chi et al., 2022) on six out of eight tasks across four image classification datasets. We conduct our experiments three times on four datasets (CIFAR-10, CIFAR-100, STL-10, and SVHN) using different seeds, reporting the average results along with the standard deviation. For the large-scale dataset ImageNet-1K, we perform a single run due to resource constraints. The average performance of our method surpasses other baselines and is more stable, as indicated by the low standard deviation.

Architecture	Vision Transformer (Small)					Vision Transformer (Large)					Average
# params	10M					110M		-			
Dataset	Cifar10	Cifar100	STL-10	SVHN	ImageNet-1K	Cifar10	Cifar100	STL-10	SVHN	ImageNet-1K	-
VQMoE	**89.7**$_{\pm 0.4}$	**67.3**$_{\pm 0.4}$	66.5$_{\pm 0.3}$	**95.6**$_{\pm 0.1}$	**54.8**	**92.8**$_{\pm 0.3}$	**67.0**$_{\pm 0.5}$	64.3$_{\pm 0.5}$	**96.0**$_{\pm 0.2}$	**71.3**	**76.5**$_{\pm 0.3}$
SMoE	88.7$_{\pm 0.2}$	65.4$_{\pm 0.5}$	66.4$_{\pm 0.1}$	95.4$_{\pm 0.1}$	52.8	85.7$_{\pm 8.5}$	55.5$_{\pm 2.8}$	64.4$_{\pm 0.2}$	94.5$_{\pm 0.1}$	71.0	74.0$_{\pm 1.6}$
XMoE	88.8$_{\pm 0.2}$	65.5$_{\pm 0.5}$	66.3$_{\pm 0.2}$	95.4$_{\pm 0.1}$	52.5	87.1$_{\pm 6.4}$	55.9$_{\pm 0.6}$	**64.6**$_{\pm 0.3}$	94.1$_{\pm 0.2}$	70.8	74.2$_{\pm 1.1}$
StableMoE	88.8$_{\pm 0.1}$	65.5$_{\pm 0.1}$	66.5$_{\pm 0.2}$	95.4$_{\pm 0.1}$	52.5	84.7$_{\pm 10.5}$	55.5$_{\pm 1.8}$	64.3$_{\pm 0.6}$	94.5$_{\pm 0.9}$	70.6	73.8$_{\pm 1.8}$
SoftMoE	85.6$_{\pm 0.3}$	61.4$_{\pm 0.3}$	65.4$_{\pm 0.2}$	94.8$_{\pm 0.1}$	41.6	80.3$_{\pm 9.7}$	42.9$_{\pm 1.4}$	63.2$_{\pm 0.5}$	93.5$_{\pm 0.1}$	68.2	69.7$_{\pm 1.6}$
ViT (Dense)	89.0$_{\pm 0.2}$	65.7$_{\pm 0.3}$	**66.6**$_{\pm 0.2}$	95.6$_{\pm 0.1}$	52.2	92.2$_{\pm 0.3}$	60.2$_{\pm 2.6}$	64.1$_{\pm 0.5}$	96.0$_{\pm 0.1}$	71.1	75.3$_{\pm 0.5}$

Table 3: Accuracy of models evaluated on vision datasets. Higher is better, the best results are in bold.

5.5 In-depth Analysis

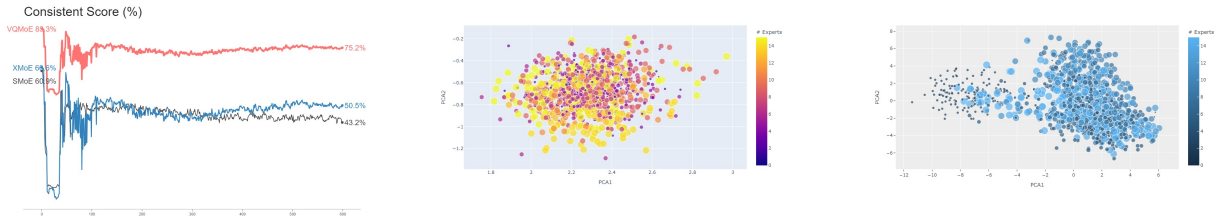
Consistent Score. Figure 3a illustrates that expert selections when training SMoE face inconsistent problems. As the Theorem 4.3, this inconsistency arises because the router's coverage rate significantly exceeds that of the Transformer representation. Figure 3a also shows that our method achieves the highest consistency score compared to the SMoE and XMoE models. However, the VQMoE model's consistency score is around 75%, as our method also requires learning a continuous representation during the Pre-training phase.

Representation Collapse issue. To visualize the Representation collapse problem in practice, we apply Principal Component Analysis (PCA) method to reduce from d dimension of the Transformer to 2D for plotting purposes, thanks to (Chi et al., 2022). Figures 3b and 3c show the expert representations from the pretrained VQMoE and SMoE models. The results suggest that VQMoE experiences less representation collapse in the expert space compared to SMoE. The analysis is in line with the theorem proof at Section 4. However, projecting the d-dimensional space onto 2D for visualization may lead to information loss.

5.6 Ablation Study

We examine the effectiveness of VQMoE across various hyper-parameter settings, with all experiments conducted using the base Transformer architecture on the WikiText-103 dataset.

Vector Quantization Method. To learn a discrete representation, we research various types of Vector Quantization methods, including VQVAE (van den Oord et al., 2017), VQGAN (Yu et al., 2022), LFQ (Yu et al., 2023), and ResidualVQ (Yang et al., 2023). We observe that VQGAN using cosine similarity for



(a) Consistent Score. (b) VQMoE Representation. (c) SMoE Representation.

Figure 3: Analysis Inconsistent Expert Selection and Representation Collapse issues when training SMoE. Figure 3a demonstrates consistent score movement from VQMoE, compared with SMoE and XMoE. Figure 3b and Figure 3c visualize the representation by experts in 2D dimension using Principal Component Analysis (PCA) method.

distance achieves good and stable results in practice as Figure 5a. Interestingly, VQGAN with lower dimensionality also delivers strong performance and exhibits robustness.

Number of codebook impact. The number of codebook entries is a crucial hyperparameter when training Vector Quantization techniques. As shown in Figure 5b, we can see the best performance when the number of codebook entries matches the number of experts. This aligns with the proof by (Dikkala et al., 2023), which demonstrates that in the optimal case, the number of clusters equals the number of experts.

Sensitiveness of VQ loss contribution α. Figure 5c illustrates the impact of α, which controls the contribution of the Vector Quantization loss to the overall loss. If α is too high, it leads to a better discrete representation but may negatively affect the final target. Conversely, if α is too low, it may result in a poor discrete representation. Therefore, α should be selected based on the data, typically within the range of $(0.05, 0.15)$.

6 Conclusion and Future Directions

This study illustrates Vector-Quantized Mixture of Experts (VQMoE), a novel and theoretically-grounded architecture, to overcome challenges in training SMoE such as representation collapse and inconsistency. We evaluate our method on various Pre-training and Fine-tuning tasks, for both language and vision domains. The results show that VQMoE outperforms the routing methods both theoretically and empirically. Furthermore, fine-tuning VQMoE with the discrete representation for downstream tasks could reduce computational resource usage by 28%. We believe that focusing on discrete representation learning will offer a promising strategy for training and testing sparse mixtures of experts (SMoE) at a large scale. Finally, we believe that our approach opens up new research avenues for effectively training SMoE, where cutting-edge techniques in discrete representation learning and vector quantization can be harnessed to enhance their performance.

References

Mikel Artetxe, Shruti Bhosale, Naman Goyal, Todor Mihaylov, Myle Ott, Sam Shleifer, Xi Victoria Lin, Jingfei Du, Srinivasan Iyer, Ramakanth Pasunuru, Giri Anantharaman, Xian Li, Shuohui Chen, Halil Akin, Mandeep Baines, Louis Martin, Xing Zhou, Punit Singh Koura, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Mona Diab, Zornitsa Kozareva, and Ves Stoyanov. Efficient large scale language modeling with mixtures of experts, 2022.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pp. 38–45, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.nlp4convai-1.5. URL `https://aclanthology.org/2020.nlp4convai-1.5`.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling, 2014. URL https://arxiv.org/abs/1312.3005.

Tianlong Chen, Zhenyu Zhang, Ajay Jaiswal, Shiwei Liu, and Zhangyang Wang. Sparse moe as the new dropout: Scaling dense and self-slimmable transformers, 2023a.

Zitian Chen, Yikang Shen, Mingyu Ding, Zhenfang Chen, Hengshuang Zhao, Erik G. Learned-Miller, and Chuang Gan. Mod-squad: Designing mixtures of experts as modular multi-task learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11828–11837, June 2023b.

Zixiang Chen, Yihe Deng, Yue Wu, Quanquan Gu, and Yuanzhi Li. Towards understanding the mixture-of-experts layer in deep learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 23049–23062. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/91edff07232fb1b55a505a9e9f6c0ff3-Paper-Conference.pdf.

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. On the representation collapse of sparse mixture of experts, 2022.

Adam Coates, Andrew Ng, and Honglak Lee. An Analysis of Single Layer Networks in Unsupervised Feature Learning. In *AISTATS*, 2011. https://cs.stanford.edu/~acoates/papers/coatesleeng_aistats_2011.pdf.

Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. Stablemoe: Stable routing strategy for mixture of experts, 2022.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988, Florence, Italy, July 2019a. Association for Computational Linguistics. doi: 10.18653/v1/P19-1285. URL https://aclanthology.org/P19-1285.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context, 2019b.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Nishanth Dikkala, Nikhil Ghosh, Raghu Meka, Rina Panigrahy, Nikhil Vyas, and Xin Wang. On the benefits of learning to route in mixture-of-experts models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9376–9396, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.583. URL https://aclanthology.org/2023.emnlp-main.583.

Giang Do, Khiem Le, Quang Pham, TrungTin Nguyen, Thanh-Nam Doan, Bint T. Nguyen, Chenghao Liu, Savitha Ramasamy, Xiaoli Li, and Steven Hoi. Hyperrouter: Towards efficient training and inference of sparse mixture of experts, 2023.

Giang Do, Hung Le, and Truyen Tran. Simsmoe: Solving representational collapse via similarity measure, 2024. URL https://arxiv.org/abs/2406.15883.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL https://arxiv.org/abs/2010.11929.

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 Bosma, Zongwei Zhou, Tao Wang, Yu Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc V Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. Glam: Efficient scaling of language models with mixture-of-experts, 2022.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity, 2022.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.446. URL https://aclanthology.org/2021.emnlp-main.446.

Yunhao Gou, Zhili Liu, Kai Chen, Lanqing Hong, Hang Xu, Aoxue Li, Dit-Yan Yeung, James T. Kwok, and Yu Zhang. Mixture of cluster-conditional lora experts for vision-language instruction tuning, 2024. URL https://arxiv.org/abs/2312.12379.

David Ha, Andrew Dai, and Quoc V. Le. Hypernetworks, 2016.

Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1558–1567. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/hu17b.html.

Changho Hwang, Wei Cui, Yifan Xiong, Ziyue Yang, Ze Liu, Han Hu, Zilong Wang, Rafael Salas, Jithin Jose, Prabhat Ram, Joe Chau, Peng Cheng, Fan Yang, Mao Yang, and Yongqiang Xiong. Tutel: Adaptive mixture-of-experts at scale, 2023.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991a. doi: 10.1162/neco.1991.3.1.79.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991b. doi: 10.1162/neco.1991.3.1.79.

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. Mixtral of experts, 2024.

Michael Jordan and Robert Jacobs. Hierarchical mixtures of experts and the. *Neural computation*, 6:181–, 01 1994.

Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. Challenges and applications of large language models, 2023.

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

Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022. URL https://arxiv.org/abs/1312.6114.

Jakub Krajewski, Jan Ludziejewski, Kamil Adamczewski, Maciej Pióro, Michał Krutul, Szymon Antoniak, Kamil Ciebiera, Krystian Król, Tomasz Odrzygóźdź, Piotr Sankowski, Marek Cygan, and Sebastian Jaszczur. Scaling laws for fine-grained mixture of experts, 2024.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, UoT, 2009.

Yoohwan Kwon and Soo-Whan Chung. Mole : Mixture of language experts for multi-lingual automatic speech recognition. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2023. doi: 10.1109/ICASSP49357.2023.10096227.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. Branch-train-merge: Embarrassingly parallel training of expert language models, 2022.

Bin Lin, Zhenyu Tang, Yang Ye, Jiaxi Cui, Bin Zhu, Peng Jin, Jinfa Huang, Junwu Zhang, Munan Ning, and Li Yuan. Moe-llava: Mixture of experts for large vision-language models, 2024.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2017. URL https://arxiv.org/abs/1608.03983.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1015.

Matt Mahoney. Large text compression benchmark, 2011. URL http://www.mattmahoney.net/dc/text.html.

Chengzhi Mao, Lu Jiang, Mostafa Dehghani, Carl Vondrick, Rahul Sukthankar, and Irfan Essa. Discrete representations strengthen vision transformer robustness, 2022. URL https://arxiv.org/abs/2111.10493.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: Vq-vae made simple, 2023. URL https://arxiv.org/abs/2309.15505.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016. URL https://arxiv.org/abs/1609.07843.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop*, 2011.

Quang Pham, Giang Do, Huy Nguyen, TrungTin Nguyen, Chenghao Liu, Mina Sartipi, Binh T. Nguyen, Savitha Ramasamy, Xiaoli Li, Steven Hoi, and Nhat Ho. Competesmoe – effective training of sparse mixture of experts via competition, 2024.

Joan Puigcerver, Carlos Riquelme, Basil Mustafa, and Neil Houlsby. From sparse to soft mixtures of experts, 2024. URL https://arxiv.org/abs/2308.00951.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts, 2021a. URL https://arxiv.org/abs/2106.05974.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 8583–8595. Curran Associates, Inc., 2021b. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/48237d9f2dea8c74c2a72126cf63d933-Paper.pdf.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, 2017.

Sheng Shen, Le Hou, Yanqi Zhou, Nan Du, Shayne Longpre, Jason Wei, Hyung Won Chung, Barret Zoph, William Fedus, Xinyun Chen, Tu Vu, Yuexin Wu, Wuyang Chen, Albert Webson, Yunxuan Li, Vincent Zhao, Hongkun Yu, Kurt Keutzer, Trevor Darrell, and Denny Zhou. Mixture-of-experts meets instruction tuning:a winning combination for large language models, 2023a.

Sheng Shen, Zhewei Yao, Chunyuan Li, Trevor Darrell, Kurt Keutzer, and Yuxiong He. Scaling vision-language models with sparse mixture of experts. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 11329–11344, Singapore, December 2023b. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.758. URL https://aclanthology.org/2023.findings-emnlp.758.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1170.

Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation, 2021. URL https://arxiv.org/abs/2105.05633.

Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. Neural discrete representation learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/7a98af17e63a0ac09ce2e96d03992fbc-Paper.pdf.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wenxuan Wang, Guodong Ma, Yuke Li, and Binbin Du. Language-routing mixture of experts for multilingual and code-switching speech recognition, 2023.

Fuzhao Xue, Zian Zheng, Yao Fu, Jinjie Ni, Zangwei Zheng, Wangchunshu Zhou, and Yang You. Openmoe: An early effort on open mixture-of-experts language models, 2024.

Dongchao Yang, Songxiang Liu, Rongjie Huang, Jinchuan Tian, Chao Weng, and Yuexian Zou. Hifi-codec: Group-residual vector quantization for high fidelity audio codec, 2023. URL https://arxiv.org/abs/2305.02765.

Hanrong Ye and Dan Xu. Taskexpert: Dynamically assembling multi-task representations with memorial mixture-of-experts. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 21828–21837, October 2023.

Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan, 2022. URL https://arxiv.org/abs/2110.04627.

Lijun Yu, Yong Cheng, Kihyuk Sohn, José Lezama, Han Zhang, Huiwen Chang, Alexander G. Hauptmann, Ming-Hsuan Yang, Yuan Hao, Irfan Essa, and Lu Jiang. Magvit: Masked generative video transformer, 2023. URL https://arxiv.org/abs/2212.05199.

Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset, 2018. URL https://arxiv.org/abs/1608.05442.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, zhifeng Chen, Quoc V Le, and James Laudon. Mixture-of-experts with expert choice routing. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 7103–7114. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/2f00ecd787b432c1d36f3de9800728eb-Paper-Conference.pdf.

Yanqi Zhou, Nan Du, Yanping Huang, Daiyi Peng, Chang Lan, Da Huang, Siamak Shakeri, David So, Andrew Dai, Yifeng Lu, Zhifeng Chen, Quoc Le, Claire Cui, James Laudon, and Jeff Dean. Brainformers: Trading simplicity for efficiency, 2024.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. St-moe: Designing stable and transferable sparse expert models, 2022.

A Appendix

Supplementary Material for "On the Role of Discrete Representation in Sparse Mixture of Experts"

This document is organized as follows. Appendix A.1 provides a detailed proof for Section 4. Appendix A.2 presents additional experimental results demonstrating the effectiveness of our method compared to the baselines. Finally, Appendix A.3 offers an in-depth analysis of representation collapse, while Appendix A.4 details the implementation aspects.

A.1 Proof for Results in Section 4

A.1.1 Jacobian Matrix of VQMoE

To investigate whether VQMoE alleviates this collapse, we derive the Jacobian of the VQMoE output with respect to the input $x \in \mathbb{R}^{n \times d}$:

$$\begin{aligned}
\boldsymbol{J}_{\text{VQMoE}} &= g_c(\boldsymbol{x}) \cdot \boldsymbol{J}_{\text{SMoE}} + \frac{\partial g_c(\boldsymbol{x})}{\partial \boldsymbol{x}} f^{\text{SMoE}}(\boldsymbol{x}) \\
&\quad + g_d(\boldsymbol{x}) \cdot \sum_{l=1}^{K} \boldsymbol{J}_l^{\text{FFN}} + \frac{\partial g_d(\boldsymbol{x})}{\partial \boldsymbol{x}} \sum_{l=1}^{K} f_l^{\text{FFN}}(\tilde{\boldsymbol{x}}_l) \\
&= g_c(\boldsymbol{x}) \cdot \left[J_1 + \sum_{j=1}^{N} \boldsymbol{c}_j \boldsymbol{e}_j^\top \right] + \sum_{m \in \{c,d\}} g_m \boldsymbol{e}_m^\top + \sum_{l=1}^{K} d_l \boldsymbol{e}_l^\top \\
&= J_1 + \sum_{j=1}^{N} c_j \boldsymbol{e}_j^\top + \sum_{l=1}^{K} d_l \boldsymbol{e}_l^\top + \sum_{m \in \{c,d\}} g_m \boldsymbol{e}_m^\top \\
&= J_1 + \sum_{j=1}^{N+K+2} o_j \boldsymbol{e}_j^\top .
\end{aligned} \tag{11}$$

where:

$$\begin{aligned}
\boldsymbol{e}_j &: \text{ Embedding of the } j\text{-th expert in the SMoE;} \\
J_1 = \mathcal{S}(x)_k \boldsymbol{J}^{\text{FFN}} &: \text{ Jacobian of the top-}k\text{ FFN block;}
\end{aligned}$$

As in SMoE, the Jacobian of VQMoE consists of two major components:

- J_1: the primary contribution from the input and selected expert.
- $\sum_{i=1}^{N+K+2} o_i \boldsymbol{e}_i^\top$: additional gradient contributions from both the continuous part and the discrete part.

Proposition A.1 (Optimal Experts Selection) *Given input data partitioned into k clusters $(C_1, C_2, \ldots, C_k)$ and a mixture of experts (MoE) layer with k experts $(E_1, E_2, \ldots, E_k)$, the assignment of each cluster C_i to expert E_i for $i \in [1, k]$ constitutes an optimal expert selection solution.*

Proposition A.1 demonstrates that if we are given a clustering structure as input, assigning each part of the input to its corresponding expert results in an optimal expert selection. This implies that learning a discrete representation and directing each component to the appropriate expert yields an optimal solution. The proof of Proposition A.1 can be found in Appendix A.

A.1.2 Proof of Theorem 4.3

In this proof, we use contradiction to establish the theorem. Assume that the expert embeddings e form a consistent router. By Definition 4.1, we have:

$$\text{dist}(x_p, u_i) \leq \min(\text{dist}(x_p, C_j)),$$

where u_i is the representation corresponding to the closest expert e_i.

According to (Chi et al., 2022), projecting information from a hidden representation space $\mathcal{R}^d$ to the expert dimension N leads to representation collapse. Now, consider three experts x, y, z whose embeddings e_x, e_y, e_z collapse. Without loss of generality, assume that e_y lies between e_x and e_z in the embedding space. Then, we have:

$$\begin{aligned} \text{dist}(y, u_y) &\leq \min(\text{dist}(x, e_x), \text{dist}(y, e_y), \text{dist}(z, e_z)) \\ &\leq \text{dist}(e_x, e_z). \end{aligned} \tag{12}$$

Let t_e denote the step at which the embeddings e_x and e_z converge, and t_m denote the step at which the Multi-Head Attention (MHA) module converges. From step t_e, it follows that:

$$\lim_{t_e \to t_m} \text{dist}(y, u_y) = \lim_{t_e \to t_m} \text{dist}(e_x, e_z) = 0.$$

Thus, y (the output of MHA) converges at step t_e.

This directly contradicts the assumption that the MHA converges at step t_m, where $t_e \ll t_m$.

A.1.3 Proof of Proposition A.1

We use contradiction to prove the proposition. Assume that, at training step t, there exists a set of pairs (C_i, E_j) such that $i \neq j$. Let $x_1, x_2, \ldots, x_k$ represent a sequence of inputs sampled from K clusters. From step t_0 to step t_{k-1}, each pair (x_j, E_j), where $j \in [1, k]$, is updated using the following gradient descent equation:

$$W_{E_j}^{l+1} = W_{E_j}^{l} - \eta \mathcal{J}(x_j),$$

where $W_{E_j}^{l}$ is the weight of expert E_j at iteration l, $\mathcal{J}(x_j)$ is the Jacobian matrix with respect to input x_j, and η is the learning rate.

Let $\mathcal{L}$ denote the loss function during the training process described by Equation 7. After t_k training steps, the following condition holds:

$$E_j(x_j) = \min_{c \in [1,k]} E_j(x_c).$$

Under the assumption of contradiction, there exists a set of pairs

$$\sum_{i,j=1; i \neq j}^{K} (C_i, E_j)$$

Scale	TopK	# Experts	SMoE	VQMoE (Discrete Only)
Base 20M-50K Steps	1	16	1.28	**1.25**
	2	16	1.26	-
	4	16	1.26	-
	8	16	1.27	-
	16	16	1.27	-
Base 20M-100K Steps	1	16	1.22	**1.18**
	2	16	1.20	-
	4	16	1.21	-
	8	16	1.21	-
	16	16	1.21	-
Large (210M)	1	64	**1.12**	1.14
	2	64	1.09	-
	4	64	1.09	-
	8	64	1.09	-
	16	64	1.10	-
	32	64	1.10	-
	64	64	1.12	-

Table 4: Performance comparison of SMoE and VQMoE (Discrete Only) on the *Enwik8* (BPC) dataset.

where the loss function $\mathcal{L}$ is minimized. However, by definition of the loss minimization process, the inequality

$$\sum_{i=1}^{K}(C_i, E_i) \leq \sum_{i,j=1;i\neq j}^{K}(C_i, E_j)$$

must hold.

This leads to a contradiction with our initial assumption.

A.2 Additional Experiment Results

Q6: Can VQMoE learn Discrete Representation Only from scratch? A6: Yes for small and medium scale, but no for large scale.

The answer is yes for small and medium-scale models. However, training a discrete representation-only approach is feasible primarily for small to medium-scale models with a moderately sized dataset. The results of the *Transformer-XL* model in Table 4 on the Enwik8 dataset support this observation. As the model scales up, relying solely on discrete representation reaches its limitations, leading to performance below the SMoE baselines.

Q7: Can VQMoE outperform the clustering-based approach such as KMean? A7: Yes.

We explored a clustering-based approach similar to MoCLEGou et al. (2024) but found it unsuitable for our method. Unlike MoCLE, Vector Quantization allows the model greater flexibility in learning cluster representations during training, making it more competitive in practical applications. The training results using the Transformer-XL model on the Enwik8 dataset are presented in Table 5.

Q8: Can VQMoE contribute to AI real-world applications? A8: Yes.

We found that VQMoE can directly benefit real-world AI applications, such as image segmentation, demonstrating its strong generalization capabilities. Specifically, our method outperforms both the baseline and dense models in terms of Mean Accuracy and mIoU metrics on the ADE20K dataset Zhou et al. (2018) using the Segmenter modelStrudel et al. (2021). Detailed results are provided in Table 6.

Scale	TopK	# Experts	SMoE	MoCLE	VQMoE
Base 20M-50K Steps	1	16	1.28	1.29	**1.25**
	2	16	1.26	1.28	-
	4	16	1.26	1.28	-
	8	16	1.27	1.28	-
	16	16	1.27	1.28	-

Table 5: Performance comparison of VQMoE and MoCLE (Clustering approach) on the *Enwik8* (BPC) dataset.

Model	ViT	SoftMoe	SMoE	StableMoE	XMoE	VQMoE	Metrics
Segmenter	20.8	19.0	23.1	22.4	22.3	**23.4**	Mean accuracy
	15.0	14.0	15.5	16.0	15.7	**16.6**	mIoU

Table 6: Comparison of VQMoE versus the baselines on the ADE20K dataset.

A.3 Representation Collapse Analysis

To illustrate Theorem 4.3, we perform a language model task as described in Section A.4.2, examining the movement of Expert Input Representation in Figure 4a and Expert Embedding (router) in Figure 4b. We analyze the dynamics of the expert input representations by tracking their changes across training iterations. The results indicate that the inputs to the experts become increasingly divergent over time. This divergence suggests that the model learns to represent the data in a more specialized and diverse manner, allowing each expert to focus on distinct features or patterns within the data. Similarly, we track the changes in expert embeddings (router) throughout the training process. However, the trend is the opposite: the expert embeddings appear to converge quickly, stabilizing around 10,000 iterations. The findings align with our assumption stated in Theorem 4.3, indicating that Expert Embedding converges more quickly than Expert Input Representation. These results provide further evidence supporting the Theorem 4.3.

A.4 Experiments implementation details

This section provides detailed parameters of our experiments in Section 5.

A.4.1 General Settings

The experiments are based on the publicly available SMoE-Dropout implementation(Chen et al., 2023a)[1]. However, the pre-training was conducted on two H100 GPUs, so results might differ when using parallel training on multiple GPUs.

A.4.2 Pre-training Experiments

Table 7 provides the detailed configurations for pre-training Transformer (Vaswani et al., 2017), Transformer-XL Dai et al. (2019b) on `Enwik8`, `Text8`, `WikiText-103`,and `One Billion Word`.

A.4.3 Fine-tuning Experiments

For fine-tuning experiments, we employ the identical model architecture as in pre-training. Table 8 presents the detailed configurations utilized for fine-tuning experiments on `SST-2`, `SST-5`, `IMDB`, and `BANKING77` datasets. We start with the pretrained checkpoint of the base model on enwik8, remove the final layer, and replace it with two randomly initialized fully connected layers to serve as the classifier for each fine-tuning dataset. All methods are fine-tuned for 5,000 steps with a uniform learning rate.

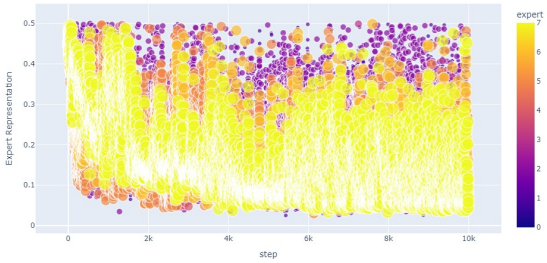
[1] `https://github.com/VITA-Group/Random-MoE-as-Dropout`

Dataset	Input length	Batch size	Optimizer	Lr	# Training Step	# Experts	TopK
`Enwik8`	512	48	Adam	3.5e-4	100k	16	2
`Text`	512	48	Adam	3.5e-4	100k	16	2
`WikiText-103`	512	22	Adam	3.5e-4	100k	16	2
`One Billion Word`	512	11	Adam	3.5e-4	100k	16	2

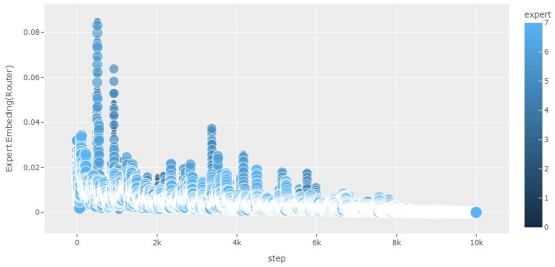
Table 7: Hyperparameter settings for pre-training experiments on `Enwik8`, `Text8` , `WikiText-103` , and `One Billion Word`.

Dataset	Input length	Batch size	Optimizer	Lr	# Epochs
`SST-2`	512	16	Adam	1e-4	5
`SST-5`	512	16	Adam	1e-4	5
`IMDB`	512	4	Adam	1e-4	5
`BANKING77`	512	16	Adam	1e-4	5

Table 8: Detail settings for fine-tuning experiments on the evaluation datasets.

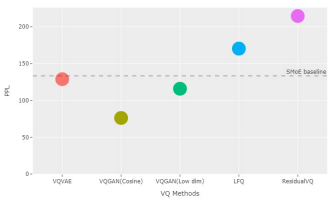


(a) Training Input Token Representations.

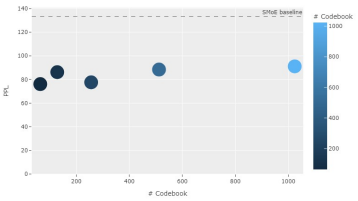


(b) Training Router Representation (Expert embedding).

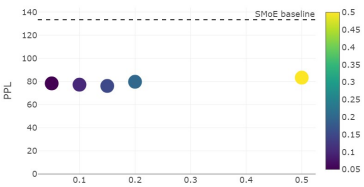
Figure 4: Comparison of Token Representation and Expert Representation across Training Iteration.



(a) Vector Quantization method.



(b) Number of codebook.



(c) Impact of α for VQMoE.

Figure 5: Pre-training small Transformer-XL on WikiText-103 across different hyperparameters.