Accelerating Dense LLMs via L0-regularized Mixture-of-Experts

Anonymous ACL submission

Abstract

Large language models (LLMs) achieve strong performance but suffer from slow and costly inference. Existing acceleration methods often lead to noticeable performance degradation, while Mixture-of-Experts (MoE) models require extensive computational resources. In this paper, we propose L0-MoE, a lightweight MoE approach using L0-regularization to accelerate dense LLMs nearly without performance loss. Our method introduces a cluster confusion matrix for domain-aware dataset curation and applies dynamic batching for efficient training. Experiments show that L0-MoE achieves up to 2.5x speedup over dense models while maintaining competitive performance, outperforming existing LLM acceleration baselines.

1 Introduction

Large language models (LLMs) have demonstrated remarkable intelligence across various tasks (OpenAI et al., 2024; Gemini-Team et al., 2024; Dubey et al., 2024; Jiang et al., 2023; DeepSeek-AI et al., 2025; Yang et al., 2024), including question answering, mathematics, coding, and content generation. A key insight into their success is the parameter scaling law (Kaplan et al., 2020), which suggests that increasing model size enhances performance across diverse tasks, potentially advancing artificial general intelligence (AGI) (Bubeck et al., 2023). However, larger LLMs incur high inference costs, leading to slower generation speeds and increased computational expenses. Thus, optimizing LLM inference efficiency has become a critical challenge for both academia and industry.

Various approaches have been proposed to accelerate LLM inference, which can be categorized into three main techniques: (1) **Quantization**, including GPTQ (Frantar et al., 2023), SmoothQuant (Xiao et al., 2023), and AWQ (Lin et al., 2024), reduces precision by converting weights and activations from floating-point to lower-bit integer formats, significantly improving efficiency. (2) **Model pruning**, such as LLM-Pruner (Ma et al., 2023) and LLM-Shearing (Xia et al., 2024), removes redundant parameters based on predefined criteria to compress models and accelerate inference. (3) **Knowledge distillation** (Gu et al., 2024; Feng et al., 2024), like reverse-KD (Gu et al., 2024) and Chain-of-Thought (CoT) Distillation (Feng et al., 2024), transfers knowledge from large LLMs to smaller ones using distillation techniques (Hinton et al., 2015), reducing computational demands. While these methods achieve substantial speedup, they often come at the cost of performance degradation, posing challenges for real-world deployment.

Recently, sparsely gated Mixture-of-Experts (MoE) models (Cai et al., 2024), particularly in transformer-based large language models, have significantly improved inference speed optimization. MoE operates on a simple yet effective principle: different model components, known as experts, specialize in distinct tasks or data aspects. For a given input, only relevant experts are activated, reducing computational costs while leveraging a vast pool of specialized knowledge. This scalable and flexible approach aligns with the scaling law, enabling larger model capacities without proportional computational overhead. However, current MoE training focuses on training from scratch or upcycling dense LLMs, both requiring vast computational resources and high-quality corpora. For instance, DeepSeek-V3 (DeepSeek-AI et al., 2024) and Qwen2.5-Max (Yang et al., 2025) were pretrained on 14.8T and 20T tokens, respectively, with additional fine-tuning, making them costly and less accessible. In contrast, *little research has explored leveraging MoE to accelerate inference using a small-scale training corpus (e.g., tens of billions of tokens) while maintaining performance comparable to dense LLMs*. This direction is particularly appealing for large-scale industrial applications with cost-sensitive deployment constraints.

To address this issue, we propose **L0-MoE**, a mixture-of-experts (MoE) model built via L0-regularization (Louizos et al., 2018) using a small, curated **30B**-token corpus. Our approach has two key components: (1) L0-regularization selects critical hidden dimensions in transformer MLPs to form experts. (2) A cluster confusion matrix (CCM)-based sampling method curates the training corpus and schedules dynamic batching. Using the BGE-M3 encoder (Chen et al., 2024) and K-means clustering (Jin and Han, 2010), we extract diverse semantic domains from RedPajama (Weber et al., 2024) to construct expert-relevant sub-datasets. A gating mechanism and dynamic batching optimize training. L0-MoE achieves **2.5×** inference speedup with no obvious performance loss across four benchmarks. Our contributions are as follows: 1) We introduce a novel MoE building method leveraging L0-regularization, enabling efficient LLM inference acceleration with minimal training cost. 2) We propose a CCM-based corpus curation and dynamic batching strategy for effective MoE training. 3) Extensive experiments validate the efficiency of our method in achieving inference speedup while maintaining performance.

2 Preliminary

2.1 L0-regularization

L0-regularization (Louizos et al., 2018) is a powerful technique for feature selection and parameter pruning in neural networks. It imposes a penalty on parameters that deviate from zero, without additional constraints. This approach enhances model efficiency by eliminating unnecessary computations and resources, as irrelevant parameters are pruned and thus not computed. For a given weight matrix $W \in R^{m \times n}$, a mask matrix $Z \in 0, 1^n$ is employed to derive a reduced weight $g(W, Z) \in R^{m \times n0}$, where g selects $n0 < n$ columns from W using Z. Due to the non-differentiable nature of Z, optimizing it is challenging. To address this, the binary hard concrete function is introduced for L0-regularization, as shown in Equation 1.

$$\begin{gathered} u \sim \mathcal{U} \\ s = Sigmoid((log(u) - log(1-u) + log\mathrm{a})/\mathrm{b}) \\ \bar{s} = s(\zeta - \gamma) + \gamma \\ z = min(1, max(0, \bar{s})) \end{gathered} \tag{1}$$

The uniform distribution $\mathcal{U}$ is defined over the interval [0,1]. We set the hyper-parameters as $b = 0.83$, $\zeta = 1.1$, and $\gamma = -0.1$ by following (Louizos et al., 2018). Using the learned z, we estimate the proportion of retained weights as $\hat{r} = \frac{sum(z)}{m*n}$. To effectively control the desired retention ratio r for a given weight matrix W, we employ a Lagrangian multiplier (Wang et al., 2019), as described in Equation 2.

$$\mathcal{L}_{l_0} = \lambda_1(\hat{r} - r) + \lambda_2(\hat{r} - r)^2 \tag{2}$$

We initialize the learnable parameters λ_1 and λ_2 to 0 in our experiments. In our approach, r represents the retention ratio of the feed-forward network (FFN) up-projection dimension.

2.2 Mixture of Expert

Mixture of Experts (MoE) (Cai et al., 2024) employs a modular architecture comprising a gating network and multiple expert networks to enhance efficiency and performance through parameter scaling. This architecture partitions the model into several experts, each specializing in specific subsets of input data. MoE utilizes a gating mechanism with a router to dynamically select the appropriate experts for processing incoming inputs, allowing the model to concentrate on relevant features while minimizing unnecessary computations. In our approach, the router is implemented as a linear projection layer $W_{router} \in R^{d \times N}$. MoE incorporates two auxiliary losses (Equation 3), such as the load balancing loss $\mathcal{L}_{balance}$ (Fedus et al., 2022) and the router Z-Loss $\mathcal{L}z$ (Zoph et al., 2022), to promote a balanced distribution of inputs among experts. These losses penalize high values in the logits produced by the gating network, encouraging a more even allocation of tokens to experts.

$$\begin{gathered} \mathcal{L}_{aux} = \mathcal{L}_{balance} + \lambda \mathcal{L}_z \\ \mathcal{L}_{balance} = \sum_{i=1}^{i=N} (\frac{c_i}{B} - \frac{1}{N})^2 \\ \mathcal{L}_z = \frac{1}{B} \sum_1^B (log(\sum_i^N e^{x_i^{(j)}})) \end{gathered} \tag{3}$$

Here c_i represents the tokens of the i^{th} expert, and N denotes the number of experts. The batch contains B tokens. The logit for the j^{th} token from the i^{th} expert, as determined by the router module, is denoted as $x_i^{(j)}$.

3 Approach

3.1 Cluster Confusion Matrix based Sampling

Given a pretraining corpus, we construct training datasets via the following steps: 1) Randomly sample a small subset without replacement and use the BGE-M3 encoder (Chen et al., 2024) to extract d_{sv}-dimensional semantic vectors for each sample. 2) Apply the K-means clustering algorithm (Jin and Han, 2010) to the semantic vectors to identify

K centers $C \in \mathbb{R}^{K \times d_{sv}}$. Divide the small subset into K folds and sample m instances from each fold to form a dataset $D_{sl} = \{D_{s_1}, \ldots, D_{s_K}\}$ for domain semantic learning, where $|D_{s_k}| = m$ for $1 \leq k \leq K$. 3) Repeat steps 1 and 2 for Q iterations to obtain $Q \times K$ centers and Q datasets. For the l^{th} iteration ($l = \{1, 2, \ldots, Q\}$), the cluster centers are $C^{(l)} \in \mathbb{R}^{K \times d_{sv}}$ and the constructed dataset is $D_{sl}^{(l)}$.

$$
\begin{aligned}
CCM[i,l] &= (f_1 + f_2) * f_3 \\
f_1(i,l) &= \tfrac{1}{K} \textstyle\sum_{k=1}^{k=K} e^{1-sim(C_i, C_k^{(l)})} \\
f_2(i,l) &= \tfrac{1}{K} \textstyle\sum_{k=1}^{k=K} e^{1-sim(C_k, C_i^{(l)})} \\
f_3(i,l) &= \delta \tfrac{\sum_{k=1}^{k=K} e^{sim(C_k^{(l)}, C_i^{(l)})}}{\sum_{k=1}^{k=K} e^{sim(C_k, C_i)}}
\end{aligned}
\tag{4}
$$

We define the clustering confusion matrix (CCM) as per Equation 4, where $\delta = 0.1$ is a hyperparameter, C_i represents the i^{th} center vector, and $CCM[i,l]$ denotes the clustering confusion value for the i^{th} center at iteration l. The hypothesis posits that the semantic domain distance for the i^{th} center between C_i and $C_i^{(l)}$ can be assessed using bidirectional inter-clustering (f_1 and f_2) and intra-clustering (f_3) cosine similarity. A larger semantic domain distance indicates that $D_{sl}^{(l)}$ from the l^{th} iteration divides domains more distinctly. We compute the domain semantic distance using Equation 5 and reorder the Q datasets based on $d_{ds}^{(l)}$. In addition to the initial $D_{sl}^{(0)}$, our datasets now include $Q-1$ ordered datasets $D_{sl}^{ord(l)}$, where $ord(l)$ is the order index.

$$
d_{ds}^{(l)} = max(CCM[:,l]) + \frac{\beta}{K} \sum_{i=1}^{i=K} CCM[i,l] \tag{5}
$$

3.2 Expert Construction via L0-regularization

We construct the experts using pretrained checkpoints of dense LLMs. The intermediate size of the feed forward network (FFN) layer is d_{int}, and we apply a mask $Z \in R^{d_{int}}$. For each domain subset D_{s_k} ($k \in \{1, 2, \ldots, K\}$) derived from the initial $D_{sl}^{(0)}$, we employ the LLM pretraining loss $\mathcal{L}_{llm}$ along with the L0-regularization loss, as specified in Equation 1, to select $r \in (0,1)$*100% of the dimensions from d_{int}, following Equation 6.

$$
\mathcal{L}_{exp} = \mathcal{L}_{llm} + \mathcal{L}_{l0} \tag{6}
$$

To ensure stable training, we gradually adjust r from 100% to the target ratio r^{target}. We freeze all non-MLP parameters of dense LLMs, and the L0-regularization-based training yields K experts, each specialized for distinct semantic domains.

3.3 Dynamic Batching for MoE Training

To train the MoE to effectively select appropriate experts based on inputs, we follow Equation 7, where $\mathcal{L}_{aux}$ is defined in Equation 3. The MoE is initialized with K pre-trained experts and a router for each MoE layer.

$$
\mathcal{L} = \mathcal{L}_{llm} + \alpha \mathcal{L}_{aux} \tag{7}
$$

We employ a two-loop batch construction strategy during training: 1) domain semantic distance scheduling, where we begin with $D_{sl}^{ord(l)}$ having a lower d_{ds}; 2) multi-domain gathering scheduling, where samples in $D_{sl}^{ord(l)}$ are arranged in a cyclic sequence order $x_i^1, x_i^2, \ldots, x_i^K$, and we select $p*K$ ($p = \{1, 2, 3, \ldots\}$) samples to form a batch. This scheduling offers two advantages: 1) In the initial iterations, the MoE rapidly learns to select appropriate experts since the domain samples in D_{sl} have been previously encountered by the experts. Consequently, the sequence-level selection capabilities of routers are effectively initialized. 2) As training progresses, the domains in $D_{sl}^{ord(l)}$ gradually transition to different semantic spaces, encouraging routers to select multiple experts for each input sample. This enhances the token-level selection capabilities of the routers.

4 Experiments

4.1 Experimental Setup

We train on the RedPajama dataset (Weber et al., 2024), a replicated pre-training corpus for LLaMA models, following prior work (Xia et al., 2024). Evaluation is conducted on four public benchmarks: MMLU (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), HumanEval (Chen et al., 2021), and BigBench Hard (BBH) (Suzgun et al., 2023). Each benchmark evaluates distinct aspects of model performance, offering insights into the strengths and limitations of LLMs. To assess effectiveness and versatility, we evaluate our method on three open-source LLMs: Llama-3-8B (Dubey et al., 2024), Mistral-7B (Jiang et al., 2023), and Qwen2-7B (Yang et al., 2024). Comparisons include L0-regularized MoEs, original LLMs, and inference optimization techniques such as GPTQ quantization (Frantar et al., 2023), LLM Shearing pruning (Xia et al., 2024), and RKD+CoT knowledge distillation (Gu et al., 2024; Feng et al., 2024). For CCM, we run 21 iterations, collecting 30B tokens. Experiments use a cluster/expert size of $K = 64$ with linear warmup, annealing, and a peak learning rate of 1e-4. Further details are in Appendix A.2.

Model	MMLU	GSM8K	HumanEval	BBH	Average	Speedup
Llama-3-8B	**66.6**	**56.0**	33.5	**57.7**	**53.5**	
Llama-3-8B w/ L0-MoE	66.3	55.9	**33.7**	57.2	53.3	2.0x
Mistral-7B	64.1	52.2	29.3	**56.1**	50.4	
Mistral-7B w/ L0-MoE	**64.8**	**53.6**	**31.1**	55.9	**51.4**	2.1x
Qwen2-7B	70.3	79.9	51.2	**62.6**	66.0	
Qwen2-7B w/ L0-MoE	**70.4**	**80.5**	**52.0**	61.5	**66.1**	**2.5x**

Table 1: Evaluation of different LLMs on MMLU, GSM8K, HumanEval and BBH benchmarks.

4.2 Main Results

Table 1 presents the model with the highest performance under our settings. The L0-MOE consistently achieves a 2-2.5x inference speedup across all base LLMs. Additionally, L0-MOE maintains performance comparable to the base LLMs across four benchmarks, with the L0-MOE variant of Mistral even demonstrating a 1% average performance improvement. Table 2 compares these results with other inference acceleration baselines, which, despite achieving some speedup, exhibit noticeable performance degradation.

Model	MMLU	GSM8K	Speedup
Qwen2-7B	70.3	79.9	-
L0-MoE	**70.4**	**80.5**	2.5x
GPTQ	67.8	73.8	1.8x
LLM Shearing	68.2	75.5	2.6x
RKD + CoT	61.2	60.2	**5.1x**

Table 2: Comparison with other inference acceleration baselines. We employ Qwen2-7B as the base LLM.

Model	MMLU	GSM8K
L0-MoE	**70.4**	**80.5**
CCM w/o K-means	68.2	78.1
w/ random order batching	68.2	75.5
w/ random batch batching	66.6	77.1
Random MoE	48.1	69.6
Magnitude	52.6	69.1
OBS	68.4	74.1
SVD	55.2	73.8

Table 3: Ablation study of CCM and L0-regularization. We conduct experiments on MMLU and GSM8K datasets with Qwen2-7B.

4.3 Ablation Study

Table 3 presents the ablation study on the CCM module, dynamic batching, and L0-regularization. Removing the K-means clustering from the CCM module results in a performance decline, underscoring the importance of effective sub-dataset curation. For dynamic batching, substituting it with random order or random batch scheduling also leads to degraded performance. In the context of MoE expert construction, we replace L0-regularization with four alternative methods: 1) **Random MoE**, which selects MLP dimensions randomly, 2) **Magnitude** (Sun et al., 2023), which selects the largest values in the weights matrix, 3) **OBS** (Frantar et al., 2021; Frantar and Alistarh, 2022), which identifies the most important dimensions using the OBS Hessian matrix, and 4) **SVD** (Wang et al., 2024), which decomposes the weights matrix using singular values to select the most significant columns. The results demonstrate the superiority of L0-MoE over them.

Model	MMLU	#Para.(B)	Speedup
L0-MoE	**70.4**	**23.3**	2.5x
CCM Iter. ($Q = 2$)	68.2	23.3	2.5x
CCM Iter. ($Q = 5$)	68.8	23.3	2.5x
CCM Iter. ($Q = 10$)	69.7	23.3	2.5x
Expert size ($K = 8$)	50.9	4.8	**4.6x**
Expert size ($K = 32$)	69.2	12.7	3.2x

Table 4: Hyper-parameter tuning of sampling iterations (Q) and cluster size (K), keeping 2.8B activated parameters for L0-MoE. Qwen2-7B is the base LLM. #Para.(B) is the number of model parameters.

4.4 Discussion

Our approach involves two critical hyperparameters: the sampling iterations Q in CCM curated datasets and the expert size K in MoE. Table 4 provides a detailed overview of hyperparameter tuning. Increasing the number of iterations for CCM enhances performance but also demands greater computational resources. We find that an initial iteration plus 20 additional iterations suffice to optimize model performance. While increasing the number of experts improves performance, it also reduces inference speed. Therefore, we select an appropriate expert size to balance performance enhancement and LLM acceleration.

5 Conclusion and Future Work

In this paper, we propose a novel Mixture-of-Experts based approach to accelerate LLM inference, leveraging clustering confusion matrix for dataset curation, L0-regularization for expert selection, and dynamic batching for efficient training with only 30B tokens. Our method achieves a 2.5× speedup over dense LLMs, outperforming strong baselines nearly without performance loss. Future work will explore scaling our approach to larger LLMs and expanding the corpus size to further enhance L0-MoE performance beyond dense LLMs.

Limitations

We did not compare our method with MoEs such as DeepSeek-MoE (Dai et al., 2024), Qwen-MoE (Team, 2024), and Mixtral (Jiang et al., 2024), which scale up model parameters in dense models with immense computational costs, processing trillions of tokens. In contrast, our approach utilizes only 30B tokens, making it more comparable to baseline post-training inference speedup methods.

Despite the promising results, several limitations remain: (1) The dataset for training each expert is selected via sequence-level semantic clustering, introducing exposure bias since MoE expert selection is performed at the token level. (2) The method does not explicitly measure inter-expert differences, potentially leading to redundant parameters that hinder L0-MoE's inference acceleration. Future work should explore token-level dataset partitioning to mitigate exposure bias. Additionally, novel learning paradigms are needed to reduce parameter redundancy and enhance expert routing efficiency.

References

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. 2023. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2024. A survey on mixture of experts. *arXiv preprint arXiv:2407.06204*.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems.

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.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, et al. 2024. Deepseek-v3 technical report.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Tao Feng, Yicheng Li, Li Chenglin, Hao Chen, Fei Yu, and Yin Zhang. 2024. Teaching small language models reasoning through counterfactual distillation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5831–5842.

Elias Frantar and Dan Alistarh. 2022. Optimal brain compression: A framework for accurate post-training quantization and pruning. *Advances in Neural Information Processing Systems*, 35:4475–4488.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. OPTQ: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*.

Elias Frantar, Eldar Kurtic, and Dan Alistarh. 2021. M-fac: Efficient matrix-free approximations of second-order information. *Advances in Neural Information Processing Systems*, 34:14873–14886.

Gemini-Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, et al. 2024. Gemini: A family of highly capable multimodal models.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. Minillm: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

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

Xin Jin and Jiawei Han. 2010. *K-Means Clustering*, pages 563–564. Springer US, Boston, MA.

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.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100.

Christos Louizos, Max Welling, and Diederik P. Kingma. 2018. Learning sparse neural networks through $l_0 regularization$. In *International Conference on Learning Representations*.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. LLM-pruner: On the structural pruning of large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, et al. 2024. Gpt-4 technical report.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2023. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. 2023. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, Toronto, Canada. Association for Computational Linguistics.

Qwen Team. 2024. Qwen1.5-moe: Matching 7b model performance with 1/3 activated parameters".

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. 2024. Svd-llm: Truncation-aware singular value decomposition for large language model compression. *arXiv preprint arXiv:2403.07378*.

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. 2019. Structured pruning of large language models. *arXiv preprint arXiv:1910.04732*.

Maurice Weber, Daniel Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, et al. 2024. Redpajama: an open dataset for training large language models. *arXiv preprint arXiv:2411.12372*.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. 2024. Sheared LLaMA: Accelerating language model pre-training via structured pruning. In *The Twelfth International Conference on Learning Representations*.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report.

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*.

A Appendix

This section provides further details on the model architecture, experimental setup (including evaluation tasks, baselines, and hyperparameter settings), and implementation details.

A.1 Model Architecture

Table 5 presents the detailed architecture of the baseline models and L0-MoE. All models incorporate group query attention (GQA) (Ainslie et al., 2023) within the self-attention layer. For the L0-MoE models, the bottom 4 layers (for Qwen2) and 8 layers (for Mistral and Llama3) are configured as dense layers, while the remaining layers are transformed into MoE layers. We select the top-2 experts for each input token.

A.2 Experimental Setups

Evaluation Tasks. We assess performance on four public benchmarks: MMLU (Hendrycks et al., 2021), GSM8K (Cobbe et al., 2021), HumanEval (Chen et al., 2021), and BigBench Hard (BBH) (Suzgun et al., 2023).

- **MMLU** (Massive Multitask Language Understanding) (Hendrycks et al., 2021) comprises 57 tasks spanning diverse subjects, including STEM (Science, Technology, Engineering, and Mathematics), humanities, social sciences, and specialized domains such as law and ethics.
- **GSM8K** (Grade School Math 8K) (Cobbe et al., 2021) is a benchmark designed to assess the mathematical reasoning capabilities of LLMs, containing 8,500 high-quality elementary math word problems.
- **HumanEval** (Chen et al., 2021) evaluates the code generation capabilities of LLMs through 164 programming tasks, each requiring the model to generate a function that satisfies a given set of test cases.
- **BBH** (BIG-Bench Hard) (Suzgun et al., 2023) is a subset of the larger BIG-Bench dataset, consisting of 23 highly challenging tasks designed to exceed the capabilities of current LLMs. These tasks demand creative problem-solving and deep domain expertise.

Baselines. We compare the L0-regularized MoEs with the original LLMs and other LLM inference optimization methods, including the quantization baseline GPTQ (Frantar et al., 2023), the model pruning baseline LLM Shearing (Xia et al., 2024), and the knowledge distillation baseline RKD + CoT (Gu et al., 2024; Feng et al., 2024).

- **GPTQ** (Frantar et al., 2023) is a block-wise quantization method that extends traditional power-of-two quantization by allowing non-uniform bin widths, enabling a better approximation of the original floating-point value distribution.
- **LLM-Shearing** (Xia et al., 2024) employs structured pruning to construct lightweight, structured LLMs from pretrained checkpoints. It jointly removes attention heads, layers, feed-forward networks (FFNs), and hidden dimensions in an end-to-end manner to optimize efficiency.
- **RKD + CoT**: We apply RKD (Gu et al., 2024) to distill the CoT (Feng et al., 2024) capabilities of Qwen2-7B into Qwen2-1.5B. RKD (Gu et al., 2024) aligns the student model with the teacher's distribution using reverse Kullback-Leibler divergence (KLD), encouraging the student to focus on the most probable outcomes. This helps preserve the quality of the student model's predictions by distilling Chain-of-Thought (CoT) reasoning from the teacher model.

Alternatives to L0-regularization. We provide a detailed description of the variants of L0-regularization in Table 3. Each of these variants leverages specific model compression principles to identify and retain the most important dimensions in expert construction.

- **Random MoE**: Selects MLP dimensions randomly, serving as a baseline to assess the necessity and effectiveness of dimension selection in expert construction.
- **Magnitude** (Sun et al., 2023)): Selects the most influential elements in the weight matrix, improving upon traditional magnitude pruning by considering both the weights and their corresponding input activations using the L2 norm.
- **OBS** (Frantar et al., 2021; Frantar and Alistarh, 2022)): Identifies the most critical dimensions using the OBS Hessian matrix, which

Model	Parameters(B)	Layer	Hidden	Q/KV	FFN	MoE FFN	Experts
Llama-3-8B	7.5	32	4096	32/8	14336		
Llama-3-8B w/ L0-MoE	25.1/3.2	32/24	4096	32/8	14336	1024	64:2
Mistral-7B	7.1	32	4096	32/8	14336		
Mistral-7B w/ L0-MoE	24.7/2.9	32/24	4096	32/8	14336	1024	64:2
Qwen2-7B	7	28	3584	28/4	18944		
Qwen2-7B w/ L0-MoE	23.3/2.8	28/24	3584	28/4	18944	1280	64:2

Table 5: Detailed model architecture parameters. We denote the total and activated parameters of the MoEs, as well as the total layers and MoE layers, using the format "32/24", etc. All models utilize GQA, and we present the query/key-value heads. "FFN" refers to the dense decoder MLP size, while "MoE FFN" indicates the intermediate size of the expert for the MoE layer. The total and activated experts are represented as "64:2", etc.

encapsulates second-order derivative information of the loss function with respect to model parameters. This approach is crucial for both pruning and quantization, as it helps retain the weights that most significantly impact model performance.

- **SVD** (Wang et al., 2024)): Decomposes the weight matrix using singular value decomposition (SVD) and selects the most significant columns. By retaining only the largest singular values, it reduces parameter count while preserving essential information. This truncation minimizes compression loss, and layer-wise updates further fine-tune the model to maintain accuracy.

Hyper-parameter Setting. The detailed hyper-parameter settings are presented in Table 6. This includes the hyper-parameters for the clustering confusion matrix (CCM) as well as those for MoE training.

CCM Hyper-parameters	
Q	21
K	64
d_{sv}	1024
$D_{sl}^{(0)}$	12B tokens; $\lvert D_{s_i}^{(0)}\rvert \approx 0.15B$
$D_{sl}^{(l)}, l \geq 1$	0.9B tokens; $\lvert D_{s_i}^{(0)}\rvert \approx 0.007B$
MoE Training Hyper-parameters	
Sequence length	4096
Learning rate	1e-4
Warmup ratio (expert)	0.2
Warmup ratio (MoE)	0.06
Warmup type	Linear
Annealing ratio	0.1
Annealing type	Cosine
Batch tokens	512K
α in Eq. 7	0.01
β in Equation 5	0.5
λ in Eq. 3	0.3
Training epoch	1

Table 6: Hyper-parameters for CCM and MoE training.

A.3 Implementation Details

We train our model using the FSDP framework[1], employing a layer-wise wrapping policy with the Zero-3 parameter sharding strategy, without CPU offloading. For inference during evaluation, we utilize the SGlang framework[2], which is highly optimized for the efficient execution of both dense LLMs and MoEs. To ensure a fair comparison, we strictly adhere to the original evaluation settings for each benchmark. To support future research, we will release our curated dataset and code to enhance the reproducibility of our work.

[1] https://pytorch.org/docs/stable/fsdp.html

[2] https://github.com/sgl-project/sglang