CLIP-UP: A Simple and Efficient Mixture-of-Experts CLIP Training Recipe with Sparse Upcycling

Anonymous ACL submission

Abstract

Mixture-of-Experts (MoE) models are crucial for scaling model capacity while controlling inference costs. While integrating MoE into multimodal models like CLIP improves performance, training these models is notoriously challenging and expensive. We propose **CLIP-Upcycling** (**CLIP-UP**), an efficient alternative training strategy that converts a pre-trained dense CLIP model into a sparse MoE architecture. Through extensive experimentation with various settings and auxiliary losses, we demonstrate that CLIP-UP significantly reduces training complexity and cost. Remarkably, our sparse CLIP B/16 model, trained with CLIP-UP, outperforms its dense counterpart by 7.2% and 6.6% on COCO and Flickr30k text-to-image Recall@1 benchmarks respectively. It even surpasses the larger CLIP L/14 model on this task while using only 30% of the inference FLOPs. We further demonstrate the generalizability of our training recipe across different scales, establishing sparse upcycling as a practical and scalable approach for building efficient, high-performance CLIP models.

1 Introduction

CLIP (Radford et al., 2021; Jia et al., 2021) has become foundational across domains such as image classification, multimodal retrieval, and AI-driven multimodality content generation (Zhou et al., 2022; Rao et al., 2022; Gan et al., 2022; Ramesh et al., 2021; Liu et al., 2023). As applications grow, scaling CLIP becomes essential. Most efforts focus on enlarging dense models (Cherti et al., 2023), which improves performance but incurs high computational and inference costs.

An efficient alternative is sparse modeling with Mixture-of-Experts (MoE) (Mustafa et al., 2022; Shazeer et al., 2017). However, training MoE-based CLIP models like LIMOE (Mustafa et al., 2022) from scratch remains expensive and often requires auxiliary losses for stability. For instance, LIMOE outperforms dense CLIP but demands 1.35× more training FLOPs (Mustafa et al., 2022).

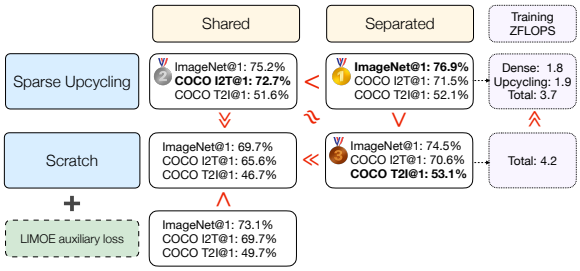

Figure 1: **Our proposed MoE CLIP pre-training recipe.** We highlight key factors for efficient training, including backbone sharing, training from scratch vs. sparse upcycling, and auxiliary losses. A detailed analysis is provided in Section 3.1 and Section 3.2.

To address this, we explore sparse upcycling (Komatsuzaki et al., 2023), which initializes MoE layers from a pre-trained dense model. As shown in Figure 1, our extensive experiments demonstrate that sparse upcycling with a separated backbone achieves the best performance while reducing training ZFLOPs from 4.2 to 3.7 compared to training from scratch. Although LIMOE's entropy losses (Mustafa et al., 2022) improve shared-backbone models trained from scratch, they still underperform other setups. Section 3.1 details these strategies and the effects of auxiliary losses.

In contrast, we propose **CLIP-UP**, a single-stage sparse upcycling method for CLIP. By leveraging pre-trained weights, CLIP-UP provides a warm start that boosts efficiency and surpasses both dense continued training and sparse-from-scratch methods across model scales.[1]

Our main contributions are:

1. We introduce CLIP-UP, a simple and effective training recipe for MoE CLIP models

[1]Concurrent work CLIP-MoE (Zhang et al., 2024) also explores MoE upcycling, using cluster-and-contrast learning to initialize experts. However, it requires additional training stages per expert, making it difficult to scale.

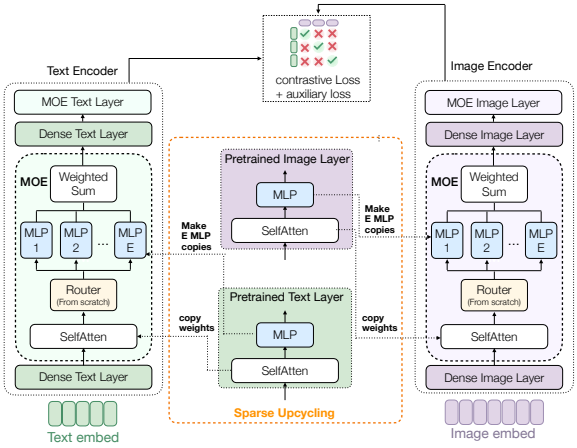

Figure 2: CLIP-UP overview with sparse upcycling initialization. Selected MLP layers are replaced with MoE layers, initialized from the dense checkpoint, while routers are randomly initialized.

via sparse upcycling, avoiding complex auxiliary losses and outperforming existing methods across shared and separated backbones.

2. CLIP-UP significantly improves performance on text-image retrieval, surpassing dense CLIP by 7.2% and 5.5% (recall@1) on COCO and Flickr30K, respectively, with a B/16 backbone.
3. We demonstrate CLIP-UP's scalability from B/32 to L/14 and provide insights into key factors and challenges to inform future design.

2 CLIP-UP

Figure 2 illustrates the CLIP-UP architecture and training strategy. We detail both in this section.

2.1 CLIP

Given n pairs of image and text captions $\{(\mathbf{I}_j, \mathbf{T}_j)\}_{j=1}^{n}$, CLIP (Radford et al., 2021) learns image and text embeddings $f(\mathbf{I}_j)$ and $g(\mathbf{T}_j)$ using a contrastive loss. With batch size B and temperature θ, the loss is

$$\mathcal{L}_{\text{Contrastive}} = -\frac{1}{2B}\sum_{j=1}^{B}\Big(\log\frac{e^{\text{sim}(f(\mathbf{I}_j),g(\mathbf{T}_j))/\theta}}{\sum_{k=1}^{B} e^{\text{sim}(f(\mathbf{I}_j),g(\mathbf{T}_k))/\theta}} + \log\frac{e^{\text{sim}(f(\mathbf{I}_j),g(\mathbf{T}_j))/\theta}}{\sum_{k=1}^{B} e^{\text{sim}(f(\mathbf{I}_k),g(\mathbf{T}_j))/\theta}}\Big) \tag{1}$$

2.2 CLIP with Mixture-of-Experts upcycling

Each MoE layer consists of E MLP experts and a router that activates the top-K experts per input token based on predicted gating logits. Let $\mathbf{X}_j \in \mathbb{R}^D$ be the input for the j-th token, $\mathbf{G}_{e,j} \in \mathbb{R}^D$ the gating logits, and $\mathbf{W}_e \in \mathbb{R}^D$ the router weights for expert e. The output $\text{MoE}(\mathbf{X}_j)$ is computed as:

$$\text{MoE}(\mathbf{X}_j) = \mathbf{X}_j + \sum_{e\in\text{Top-K}} G_{e,j}\text{MLP}_e(\mathbf{X}_j)$$

$$G_{e,j} = \begin{cases}\text{Softmax}(\mathbf{W}_e^T\mathbf{X}_j), & \text{if } e \in \text{Top-K},\\ 0, & \text{otherwise},\end{cases} \tag{2}$$

Each expert is assigned a fixed buffer capacity (Fedus et al., 2022), allowing it to process a limited number of tokens at a time. With capacity factor C, batch tokens B_t, the capacity per expert is $B_e = (B_t/E) \times C$. This ensures computational efficiency and effective resource management. Tokens are assigned to experts on a "first-come-first-serve" basis (Fedus et al., 2022). This simple mechanism avoids prioritization overhead while efficiently distributing tokens across experts.

Auxiliary loss. Simplified token assignment reduces overhead but risks imbalanced token distribution, leading to token dropping and performance degradation (Zeng et al., 2024). To mitigate this, we adopt an auxiliary loss (Zoph et al., 2022) combining load balance loss and router z-loss with scaling factors α and β. The load balance loss promotes uniform token allocation across experts. For a sequence of length S, it is defined as:

$$\mathcal{L}_{\text{Balance}} = \alpha \cdot \sum_{e=1}^{E} R_e \cdot P_e \tag{3}$$

where $R_e = \frac{E}{K\cdot S}\sum_{j=1}^{S}\mathbb{1}(\text{Token } j \to \text{Expert } e)$ and $P_e = \frac{1}{S}\sum_{j=1}^{S}\mathbf{G}_{e,j}$, denoting the token assignment ratio and average router probability for expert e respectively.

The router z-loss stabilizes gating by regularizing router logits to keep outputs within a reasonable range. It is defined as:

$$\mathcal{L}_{\text{Router}} = \beta \cdot \frac{1}{S}\sum_{j=1}^{S}\left(\log\sum_{e=1}^{E} e^{\mathbf{G}_{e,j}}\right)^2 \tag{4}$$

LIMOE auxiliary loss. We experimented with LIMOE's local and global entropy losses (Mustafa et al., 2022), tuning hyperparameters accordingly. While LIMOE auxiliary loss improves shared backbone trained from scratch, it underperforms in other settings. Therefore, we use load balance and router-z losses as our auxiliary loss.

Table 1: Ablation study comparing shared vs. separated backbones and training from scratch vs. sparse upcycling, evaluated on ImageNet (Accuracy@1 %) and COCO/Flickr30K text-to-image (T2I)/image-to-text (I2T) retrieval (Recall@1 %)

Backbone	Upcycle?	ImageNet	COCO		Flickr30K	
			T2I	I2T	T2I	I2T
		Acc@1	R@1	R@1	R@1	R@1
Shared	N	69.7	46.7	65.6	71.4	86.3
Shared	Y	75.2	51.6	**72.7**	78.0	92.0
Separated	N	74.5	**53.1**	70.6	78.3	88.2
Separated	Y	**76.9**	52.1	71.5	**80.9**	**92.3**

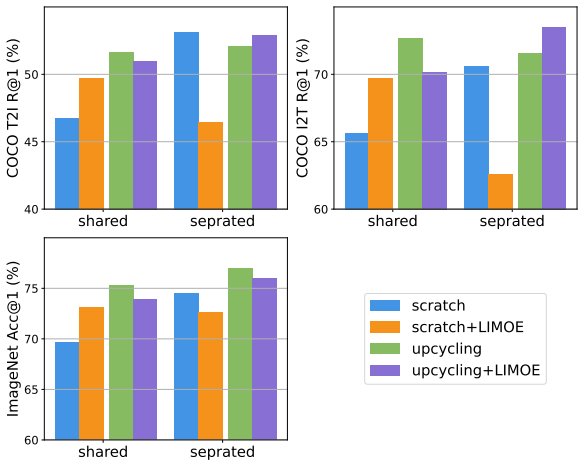

Figure 3: Impact of LIMOE auxiliary loss under different training setups. Adding LIMOE loss sometimes causes instability, especially with unshared backbones, while our upcycling recipe remains more robust.

2.3 Sparse Upcycling Training

Sparse upcycling begins with a pre-trained dense CLIP, replacing selected MLP layers with MoE layers—experts initialized from the dense weights and routers randomly initialized. All other layers remain unchanged. The model is then fine-tuned with slightly reduced learning rate and weight decay for improved stability, as shown in Figure 2.

3 Experiments

Datasets. We trained both the initial dense CLIP checkpoint, CLIP-UP, and the baseline model on the same paired image-text datasets—WIT-300M (Wu et al., 2024) and DFN-5B (Fang et al., 2023). Evaluation was performed on ImageNet (Deng et al., 2009; Shankar et al., 2020) for classification and on COCO (Lin et al., 2014) and Flickr30K (Plummer et al., 2017) for image-text retrieval, with additional benchmarks provided in the Appendix C.2. The input image resolution is 224 for all of the datasets.

Setup. We train a dense CLIP model for 440k steps, then upcycle it into an MoE version with 350k additional steps. Both use AdamW with a 32k batch size; the dense model uses a learning rate of 5×10^{-4} and weight decay of 0.2, reduced to 5×10^{-5} and 0.05 for upcycling. In the MoE model, half of the Transformer's MLP layers follow an alternating [dense, sparse] pattern (Zoph et al., 2022; Du et al., 2022), each sparse layer using 8 experts with top-2 routing. Router loss coefficients $\alpha = 0.01$ and $\beta = 0.001$ balance expert usage without dominating training (Zoph et al., 2022; Xue et al., 2024). For fair comparison, we also train a dense CLIP for 790k steps using the same settings.

3.1 Recipe Study

We compare shared vs. separated backbones and training from scratch vs. sparse upcycling using the CLIP-B/16 model, with the shared setup using 16 experts to match the separated configuration (8 per modality). As shown in Table 1, the separated backbone with sparse upcycling delivers the best overall performance due to dedicated parameters per modality, while the shared backbone sees greater relative gains from sparse upcycling. Overall, sparse upcycling consistently outperforms training from scratch, demonstrating CLIP-UP's versatility and efficiency across configurations.

3.2 Impact of LIMOE auxiliary loss.

We examine the LIMOE auxiliary loss by setting $\tau = 6$ in the global entropy loss (Mustafa et al., 2022), encouraging use of at least six experts per modality, and tuning the loss weight for our setup. As shown in Figure 3, it improves ImageNet and COCO performance with a shared backbone trained from scratch, consistent with prior work (Mustafa et al., 2022), but still underperforms compared to other configurations without it. Applying the loss to our best setup (Separated-Upcycle) slightly boosts text-image retrieval but falls short on ImageNet zero-shot classification.

These auxiliary losses also increase training complexity due to more hyperparameters. As LIMOE loss didn't work reliably across all setups, we use load balance and router-z losses to simplify tuning under resource constraints.

Table 2: Performance comparison of CLIP-UP, dense models, and LIMOE across model sizes. CLIP-UP is upcycled from a 440k-step CLIP checkpoint with 350k additional steps; a 790k-step dense CLIP is trained for fair comparison.

Model	Steps (K)	Inference GFLOPS	ImageNet classification Acc@1	ImageNet classification Acc@5	COCO retrieval T2I R@1	COCO retrieval T2I R@5	COCO retrieval I2T R@1	COCO retrieval I2T R@5	Flickr30K retrieval T2I R@1	Flickr30K retrieval T2I R@5	Flickr30K retrieval I2T R@1	Flickr30K retrieval I2T R@5
B/32												
OpenAI-CLIP	-	14.8	63.2	88.8	30.8	55.9	51.6	75.7	-	-	-	-
LIMOE	-	22.3	67.5	-	31.0	-	45.7	-	-	-	-	-
CLIP (ours)	440	14.8	72.4	92.8	41.7	68.2	62.3	84.3	68.0	90.0	**86.5**	**97.7**
CLIP (ours)	790	14.8	72.4	92.8	41.9	67.8	62.4	84.1	67.8	88.7	85.6	96.4
CLIP-UP	790	19.6	**73.2**	**93.3**	**47.3**	**74.0**	**66.6**	**86.7**	**72.9**	**91.9**	85.9	96.9
B/16												
OpenAI-CLIP	-	41.2	68.4	91.9	33.1	58.4	53.8	77.9	-	-	-	-
LIMOE	-	48.7	73.7	-	36.2	-	51.3	-	-	-	-	-
CLIP (ours)	440	41.2	76.0	94.7	44.4	70.3	65.7	87.0	73.6	92.1	88.0	97.8
CLIP (ours)	790	41.2	76.8	**95.1**	44.9	70.8	66.0	86.6	74.3	92.7	88.9	98.0
CLIP-UP	790	54.3	**76.9**	**95.1**	**52.1**	**77.6**	**71.5**	**89.2**	**80.9**	**95.6**	**92.3**	**99.2**
L/14												
OpenAI-CLIP	-	175.5	75.3	94.5	36.1	60.8	57.7	79.1	-	-	-	-
CLIP (ours)	440	175.5	81.1	96.4	49.6	74.4	70.9	89.6	78.4	94.7	91.9	**99.2**
CLIP (ours)	790	175.5	**81.6**	**96.6**	50.2	75.2	71.4	89.9	79.3	94.9	91.7	99.0
CLIP-UP	790	231.7	81.2	**96.6**	**53.9**	**79.4**	**73.8**	**92.0**	**82.0**	**96.1**	**92.4**	99.1

3.3 Final Model Evaluation and Baselines

Based on previous results, we adopt the separated backbone with sparse upcycling as the default setup and evaluate CLIP-UP across model sizes from B/32 to L/14. Table 2 compares zero-shot classification and retrieval performance against dense CLIP models trained for the same number of steps. While extending dense CLIP training from 440k to 790k steps yields minor gains, CLIP-UP shows consistent, significant improvements across scales, especially in retrieval. Notably, CLIP-UP B/32 uses only 47% of the inference GFLOPS yet outperforms dense CLIP B/16 in COCO T2I recall@1 by 2.4%, while CLIP-UP B/16 uses just 31% of the GFLOPS and surpasses dense CLIP L/14 by 1.9%. These results demonstrate the efficiency and effectiveness of sparse upcycling for scaling CLIP models.

3.4 Training Efficiency

To highlight the effectiveness of upcycle training, Figure 4 compares dense pretraining + upcycling with training from scratch. The pretrained dense model provides a strong starting point, while training from scratch requires significantly more compute to match CLIP-UP's performance—especially on COCO image-to-text and ImageNet. Although sparse upcycling initially causes a performance drop on ImageNet due to reconfiguration, CLIP-UP consistently outperforms the scratch baseline, demonstrating better efficiency and overall performance.

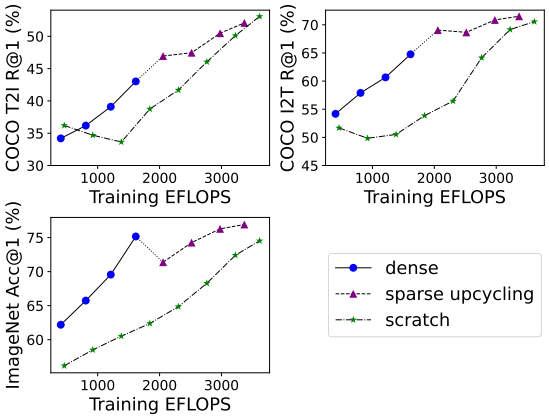

Figure 4: Performance vs. training EFLOPS for CLIP-UP and sparse-from-scratch model on CLIP B/16.

4 Conclusions

We present CLIP-UP, an efficient CLIP training strategy that combines MoE with sparse upcycling. Extensive experiments show it reduces training costs and inference FLOPs while improving performance across scales, even outperforming larger dense models. Ablation studies shown in Appendix B further validate key design choices, highlighting CLIP-UP's practicality and scalability.

5 Limitation

While our proposed method demonstrates strong performance improvements in retrieval tasks such as COCO and Flickr30K, it reveals a trade-off with classification performance on ImageNet and its variants. Specifically, we have not yet identified a training configuration that yields significant gains across both retrieval and classification simultaneously. Our current best setup prioritizes retrieval effectiveness, achieving notable improvements on COCO and Flickr30K, but leads to only marginal gains on ImageNet.

We discuss this trade-off in more detail in Appendix B.3, highlighting the role of the expert capacity factor in shaping task-specific performance. In particular, we provide examples showing how ImageNet and COCO respond differently to token dropping under varying expert capacities, which we believe contributes to the observed trade-off. While these insights offer a preliminary understanding, we are still exploring more effective strategies to better balance retrieval and classification performance.

References

Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. 2023. Reproducible scaling laws for contrastive language-image learning. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

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

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, and 8 others. 2022. Glam: Efficient scaling of language models with mixture-of-experts. *Preprint*, arXiv:2112.06905.

Alex Fang, Albin Madappally Jose, Amit Jain, Ludwig Schmidt, Alexander Toshev, and Vaishaal Shankar. 2023. Data filtering networks. *Preprint*, arXiv:2309.17425.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Preprint*, arXiv:2101.03961.

Zhe Gan, Linjie Li, Chunyuan Li, Lijuan Wang, Zicheng Liu, and Jianfeng Gao. 2022. Vision-language pre-training: Basics, recent advances, and future trends. *Preprint*, arXiv:2210.09263.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. 2021a. The many faces of robustness: A critical analysis of out-of-distribution generalization. *Preprint*, arXiv:2006.16241.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. 2021b. Natural adversarial examples. *Preprint*, arXiv:1907.07174.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*, pages 4904–4916. PMLR.

Aran Komatsuzaki, Joan Puigcerver, James Lee-Thorp, Carlos Riquelme Ruiz, Basil Mustafa, Joshua Ainslie, Yi Tay, Mostafa Dehghani, and Neil Houlsby. 2023. Sparse upcycling: Training mixture-of-experts from dense checkpoints. *Preprint*, arXiv:2212.05055.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *Preprint*, arXiv:2304.08485.

Basil Mustafa, Carlos Riquelme, Joan Puigcerver, Rodolphe Jenatton, and Neil Houlsby. 2022. Multimodal contrastive learning with limoe: the language-image mixture of experts. *Preprint*, arXiv:2206.02770.

Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. 2017. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. *Int. J. Comput. Vision*, 123(1):74–93.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. *Preprint*, arXiv:2103.00020.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. *Preprint*, arXiv:2102.12092.

Yongming Rao, Wenliang Zhao, Guangyi Chen, Yansong Tang, Zheng Zhu, Guan Huang, Jie Zhou, and Jiwen Lu. 2022. Denseclip: Language-guided dense prediction with context-aware prompting. *Preprint*, arXiv:2112.01518.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. 2019. Do imagenet classifiers generalize to imagenet? *Preprint*, arXiv:1902.10811.

Vaishaal Shankar, Rebecca Roelofs, Horia Mania, Alex Fang, Benjamin Recht, and Ludwig Schmidt. 2020. Evaluating machine accuracy on ImageNet. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8634–8644. PMLR.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *Preprint*, arXiv:1701.06538.

Wentao Wu, Aleksei Timofeev, Chen Chen, Bowen Zhang, Kun Duan, Shuangning Liu, Yantao Zheng, Jonathon Shlens, Xianzhi Du, Zhe Gan, and Yinfei Yang. 2024. Mofi: Learning image representations from noisy entity annotated images. *Preprint*, arXiv:2306.07952.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. 2020. On layer normalization in the transformer architecture. *Preprint*, arXiv:2002.04745.

Fuzhao Xue, Zian Zheng, Yao Fu, Jinjie Ni, Zangwei Zheng, Wangchunshu Zhou, and Yang You. 2024. Openmoe: An early effort on open mixture-of-experts language models. *Preprint*, arXiv:2402.01739.

Zhiyuan Zeng, Qipeng Guo, Zhaoye Fei, Zhangyue Yin, Yunhua Zhou, Linyang Li, Tianxiang Sun, Hang Yan, Dahua Lin, and Xipeng Qiu. 2024. Turn waste into worth: Rectifying top-k router of moe. *Preprint*, arXiv:2402.12399.

Jihai Zhang, Xiaoye Qu, Tong Zhu, and Yu Cheng. 2024. Clip-moe: Towards building mixture of experts for clip with diversified multiplet upcycling. *Preprint*, arXiv:2409.19291.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. 2022. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348.

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. *Preprint*, arXiv:2202.08906.

A Training Details

Below, we provide detailed training hyper-parameters and setups for dense CLIP (weights are used for sparse upcycling), sparse CLIP trained from scratch, and CLIP-UP.

A.1 Training hyper-parameters

We primarily follow (Radford et al., 2021) for hyper-parameter selection, using the WIT-3000M (Wu et al., 2024) and DFN-5B (Fang et al., 2023) training datasets. Table 3 summarizes the hyper-parameters for all experiments, including MoE-specific configurations and parameters for dense CLIP, sparse CLIP, and CLIP-UP.

Table 3: Training hyper-parameters and settings for dense CLIP used for sparse upcycling and CLIP-UP

General	
Batch size	32768
Image size	224×224
Text tokenizer	T5 (Raffel et al., 2023), lowercase
Text maximum length	77 tokens
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.98$)
LR schedule	cosine decays with linear warm-up (first 2k steps)
Dropout rate	0.0
MoE	
Inner structure	Pre-Layer Normalization (Xiong et al., 2020)
Router type	Top-2 routing
Expert capacity factor (C)	2.0 (both text and image)
MoE position	[dense, sparse] (half of MLP layers replaced by MoE layers)
Load balance loss weight	0.01
Router-z loss weight	0.0001
Dense Model	
Steps	439087 (*i.e.,* $\sim$ 14B examples seen)
Peak learning rate (LR)	$5e^{-4}$
Weight decay	0.2
CLIP-UP	
Steps	351269 (*i.e.,* $\sim$ 11B examples seen)
Peak learning rate (LR)	$5e^{-5}$
Weight decay	0.05
Expert count	8 (for text and image separately)
Sparse Model	
Steps	790356 (*i.e.,* $\sim$ 25B examples seen)
Peak learning rate (LR)	$5e^{-4}$
Weight decay	0.2
Expert count	8

B Ablation study

B.1 MoE added to single or multiple modalities

In the CLIP-UP model, the MoE setup is applied to both the text encoder and image encoder. We also explore the effect of adding MoE layers to only one modality while keeping the other modality fully dense. The COCO retrieval and ImageNet results for these configurations are shown in Figure 5, measured across different training steps.

We observe that the initial performance of newly upcycled models tends to decline compared to the starting dense model, regardless of where the MoE setup is applied. However, all configurations recover after approximately 5k training steps. Applying MoE layers to both modalities leads to a more significant initial drop on average. When MoE layers are applied to only one modality (either image or text), the final performance remains comparable. Notably, the initial performance drop is more pronounced when MoE layers are applied to the image modality, suggesting that the model is more sensitive to changes in image representations.

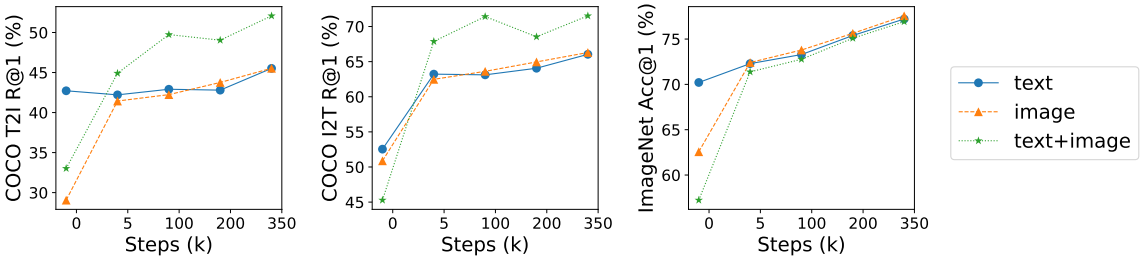

Figure 5: CLIP-UP with MoE upcycling for only the text encoder, image encoder, or both. We observe upcycling both the image and text encoders into MoE generally helps, especially for retrieval tasks.

B.2 Expert capacity factor

Intuitively, the number of tokens processed by each expert plays a crucial role in determining model quality. In CLIP-UP, this is controlled by the expert capacity factor, denoted as C. A higher C results in less token dropping, thereby reducing the initial quality drop. However, this doesn't necessarily guarantee a higher final model quality. By default, we set the capacity factor to 2.0 for both modalities. As shown in Figure 6, increasing C_{image} to 4.0 significantly boosts the ImageNet zero-shot metrics. However, this adjustment results in a noticeable drop in performance on COCO and Flickr30k retrieval tasks.

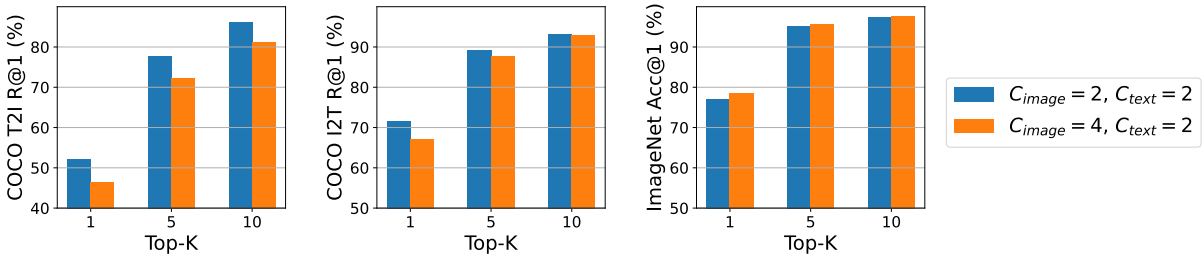

Figure 6: Effects of different expert capacity C.

Figure 7 visualizes the behavior of dropped image tokens under different capacity settings of COCO and Imagenet. The red squares represent the dropped image tokens. With a higher C_{image}, fewer image tokens are discarded. This benefits ImageNet performance since the dataset primarily consists of single-object images. Retaining more tokens allows the model to focus on the key features. In contrast, COCO images often depict complex scenes with multiple objects, but not all of which are relevant to the paired captions. Dropping less important image tokens helps the model concentrate on the most important objects, which explains the drop in COCO performance when fewer tokens are discarded. We leave the further study of capacity factor for different tasks to a future work.

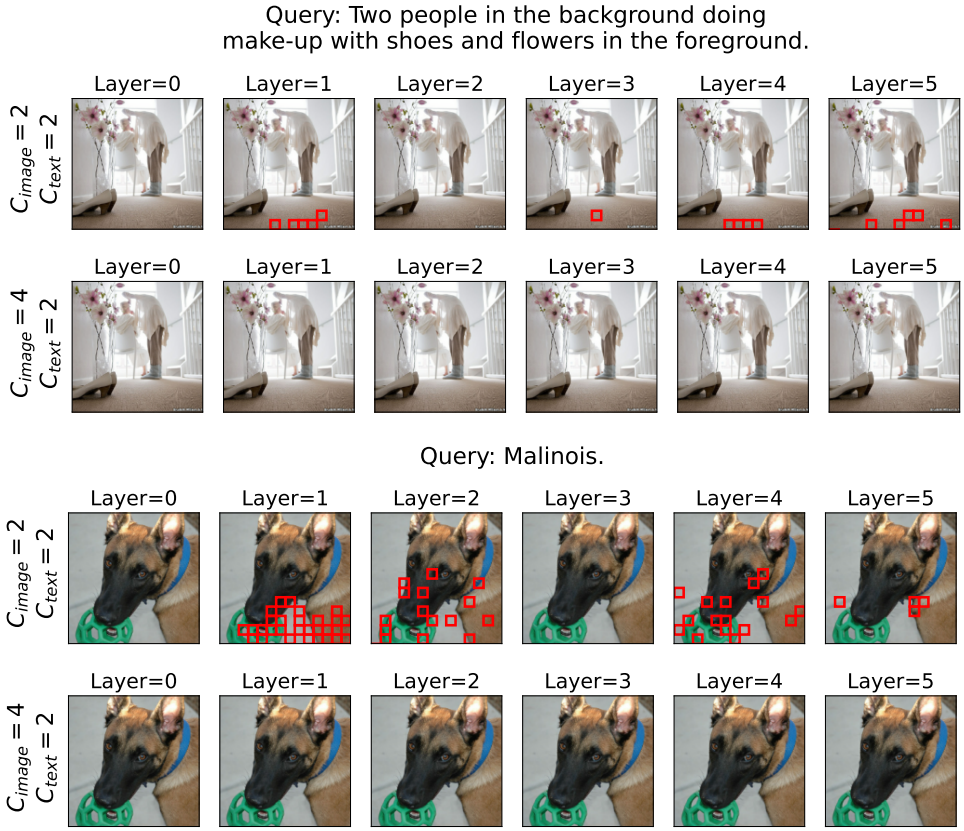

Figure 7: Visualization of image tokens dropped by the router (i.e., not being assigned to any expert due to capacity constraint) on COCO (Top) and ImageNet (Bottom).

B.3 Normalize gating weights before or after routing.

To mitigate the initial quality drop observed when applying sparse upcycling, we experimented with normalizing the router output logits after routing. This ensures the remaining gating weights are normalized to sum to 1, even when some tokens are dropped due to expert capacity constraints. The intuition behind this approach is that in the dense model, each token was previously processed by a single expert MLP.

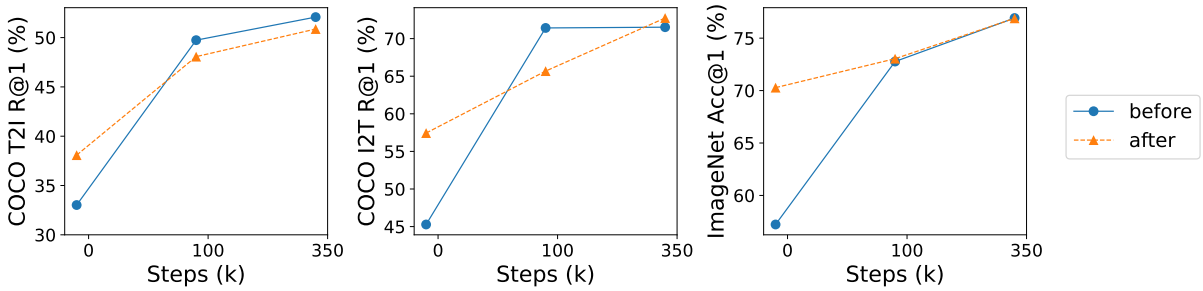

Figure 8: Model performance for gating normalization applied before or after routing

As shown in Figure 8, normalizing gating weights post-routing helps reduce the initial quality drop. However, in terms of final model performance, this approach shows improved results in image-to-text retrieval, but performs worse in text-to-image retrieval.

A possible explanation for this discrepancy is that post-routing normalization maintains the magnitude of all remaining tokens, which benefits the image encoder, as most image tokens are informative. In contrast, text encoder often deals with padding tokens, and reducing the magnitude of these tokens can enhance the text encoder's ability to focus on meaningful content. It also aligns with the finding that the initial quality drop is the biggest when adding MoE layers into image modality only.

C Tabular results

C.1 Comparison of model architectures and impact of LIMOE auxiliary loss

All results for different model architectures, with and without LIMOE auxiliary loss, as discussed in Section 3.1.

Table 4: All results from Table 1 and Figure 3 as discussed in Section 3.1

Model	ImageNet Acc@1	COCO T2I R@1	COCO I2T R@1	Flickr30K T2I R@1	Flickr30K I2T R@1
Shared	69.7	46.7	65.6	71.4	86.3
+LIMOE Aux. Loss	73.1	49.7	69.7	75.6	87.9
Δ	+3.4	+3.0	+4.1	+4.2	+1.6
Shared-UpCycle	75.2	51.6	**72.7**	78.0	92.0
+LIMOE Aux. Loss	73.9	50.9	70.1	73.8	85.5
Δ	-1.3	-0.7	-2.6	-4.2	-6.5
Separated	74.5	**53.1**	70.6	78.3	88.2
+LIMOE Aux. Loss	72.6	46.4	62.6	73.4	85.2
Δ	-2.0	-6.7	-8.0	-4.9	-3.0
Separated-UpCycle	**76.9**	52.1	71.5	**80.9**	**92.3**
+LIMOE Aux. Loss	75.9	52.9	73.5	81.3	92.5
Δ	-1.0	+0.8	+2.0	+0.4	+0.2

C.2 Performance on ImageNet variants

To complement the performance comparison in Table 2, we additionally evaluated our models on several ImageNet variants, including ImageNet-V2 (Recht et al., 2019), ImageNet-A (Hendrycks et al., 2021b), and ImageNet-R (Hendrycks et al., 2021a). As shown in Table 5, the performance trends on these datasets are consistent with those observed on the original ImageNet benchmark.

Table 5: Performance on ImageNet Variants of CLIP-UP, dense models across model sizes.

Model	Steps (K)	ImageNet-V2 classification Acc@1	ImageNet-V2 classification Acc@5	ImageNet-A classification Acc@1	ImageNet-A classification Acc@5	ImageNet-R classification Acc@1	ImageNet-R classification Acc@5
B/32							
CLIP (ours)	440	64.0	88.0	32.1	65.4	80.0	92.7
CLIP (ours)	790	64.0	87.7	32.6	64.8	**80.3**	92.7
CLIP-UP	790	**65.1**	**88.5**	**34.2**	**66.8**	79.9	**92.8**
B/16							
CLIP (ours)	440	68.2	90.7	47.7	78.5	84.6	95.3
CLIP (ours)	790	69.3	**91.2**	**50.5**	**80.2**	85.9	**95.8**
CLIP-UP	790	**69.6**	91.1	49.2	78.6	**85.9**	95.6
L/14							
CLIP (ours)	440	74.3	93.5	67.1	**89.0**	90.6	97.7
CLIP (ours)	790	**74.7**	93.6	**68.2**	88.9	**91.2**	**97.8**
CLIP-UP	790	73.8	**93.6**	66.1	88.1	89.7	97.3

C.3 Comparison of MoE added to single modality or both modalities

All results from Figure 5 to compare MoE added into different modalities.

Table 6: All results from Figure 5. MoE-text: MoE layers are added into text modality only. MoE-image: MoE layers are added into image modality only. MoE-both: MoE layers are added into both text and image modalities.

Model	Steps	ImageNet Acc@1	COCO T2I R@1	COCO I2T R@1	Flickr30K T2I R@1	Flickr30K I2T R@1
MoE-text	0	70.2	42.7	52.5	72.1	79.4
	5	72.3	42.2	63.2	70.0	86.9
	100	73.3	42.9	63.1	70.4	88.0
	200	75.4	42.8	64.1	72.7	88.6
	350	77.2	45.5	66.0	74.2	89.6
MoE-image	0	62.6	29.1	50.9	56.1	73.7
	5	72.4	41.4	62.5	70.0	84.8
	100	73.8	42.2	63.6	71.7	88.2
	200	75.6	43.8	65.0	72.2	88.4
	350	77.6	45.5	66.3	74.5	89.4
MoE-both	0	57.2	33.0	45.3	62.7	73.4
	5	71.4	44.9	67.9	71.6	87.8
	100	72.8	49.7	71.4	74.9	89.1
	200	75.1	49.0	68.5	73.1	85.1
	350	76.9	52.1	71.5	80.9	92.3

C.4 Comparison of capacity factor

As discussed in Section B.2, expert capacity factor C plays a crucial role in balancing classification performance on ImageNet and its variants with retrieval performance on Flickr and COCO. As shown in Table 7, increasing C from 2.0 to 4.0 consistently improves accuracy on ImageNet and its variants, but at the expense of COCO performance—highlighting an inherent trade-off.

Table 7: All results from Figure 6

Model	ImageNet Acc@1	ImageNet-V2 Acc@1	ImageNet-A Acc@1	ImageNet-R Acc@1	COCO T2I R@1	COCO I2T R@1	Flickr30K T2I R@1	Flickr30K I2T R@1
$C_{image} = 2, C_{text} = 2$	76.9	69.4	49.8	83.9	**52.1**	**71.5**	**80.9**	**92.3**
$C_{image} = 4, C_{text} = 2$	**78.4**	**71.0**	**53.7**	**86.9**	46.3	66.9	75.5	90.2

D Router Analysis

D.1 Routing distribution

The routing is balanced across all transformer layers. The average token ratios assigned to each expert for both text and image modalities on ImageNet and COCO are shown in Figure 9 and Figure 10, respectively.

Token assignment appears more balanced for images than for text, likely due to the higher number of tokens in images, many of which carry similar or redundant information. This redundancy facilitates a more uniform distribution of tokens across experts. In contrast, text is typically more discrete and information-dense, leading to a stronger preference for certain experts that specialize in specific linguistic patterns.

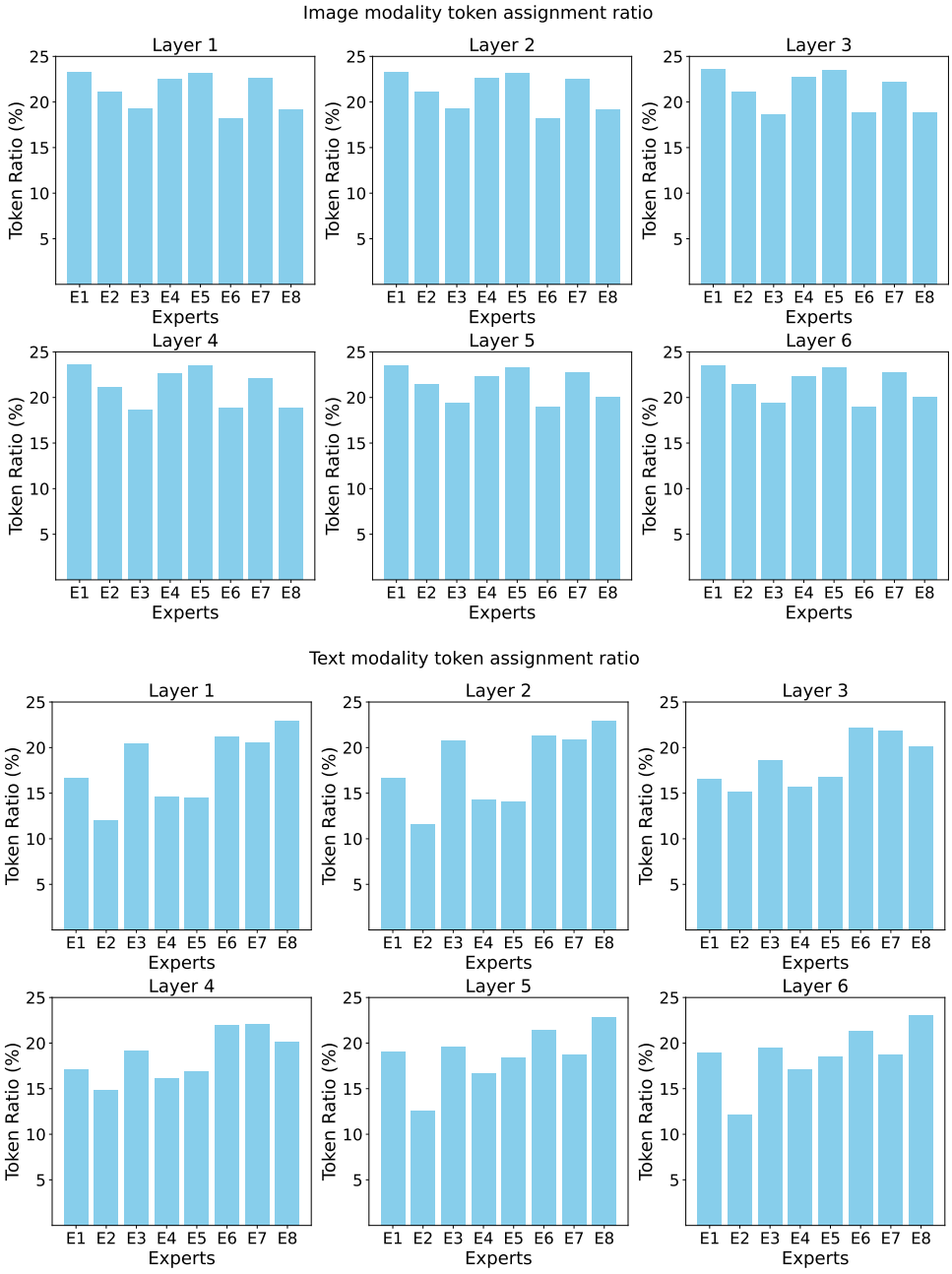

Figure 9: Visualization of the token assignment ratios to each expert in each layer by the router on ImageNet — image tokens (top) and text tokens (bottom).

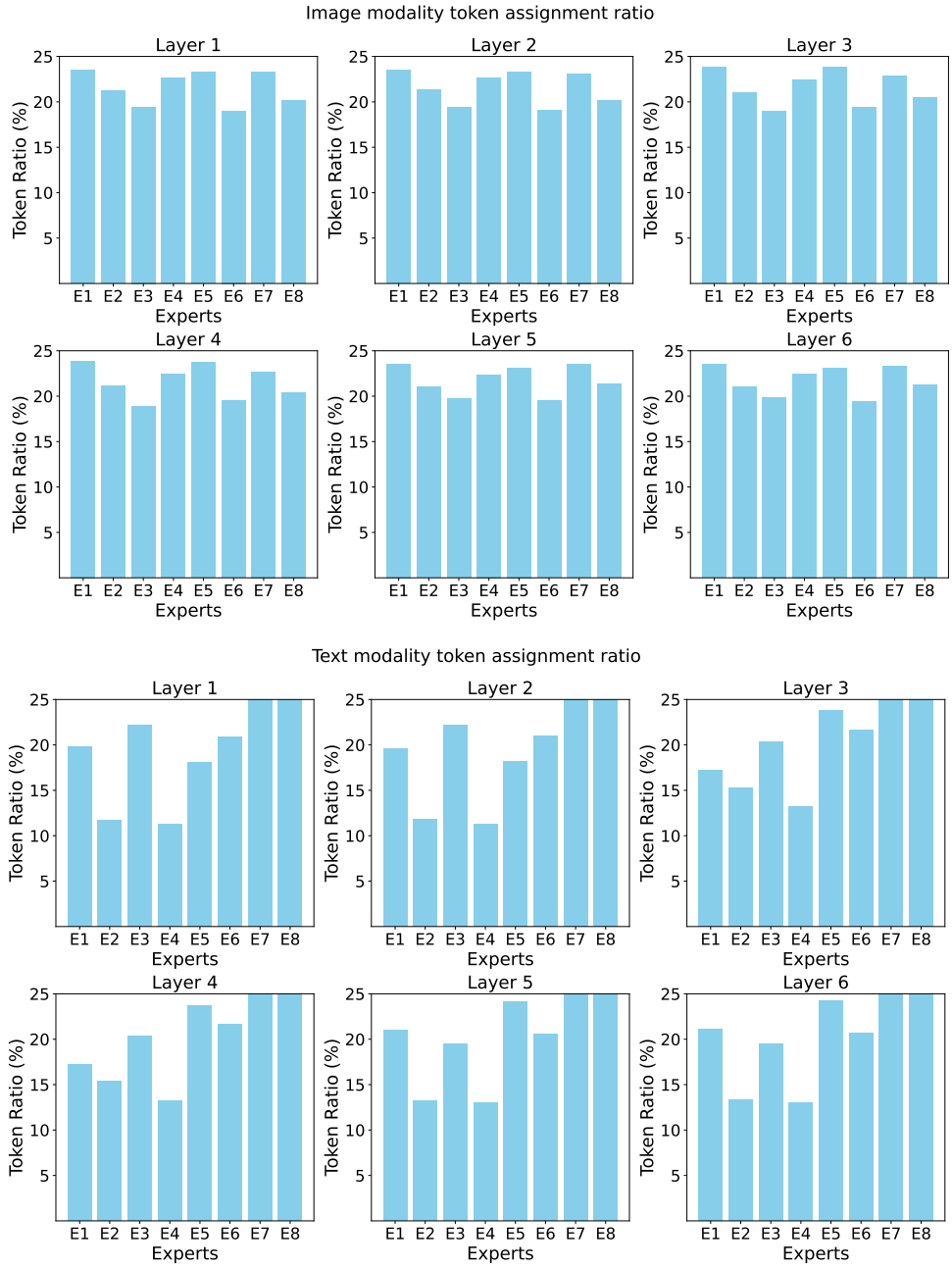

Figure 10: Visualization of the token assignment ratios to each expert in each layer by the router on COCO — image tokens (top) and text tokens (bottom).

D.2 Expert preference pattern

We observe distinct preference patterns among experts. As illustrated in Figure 11, one expert predominantly processes tokens related to "eyes," consistently attending to the eye regions of various animals and humans. Another expert demonstrates specialization in "symbols," handling text tokens from a wide range of contexts, including posters, machine interfaces, book titles, store signage, and instructional materials.

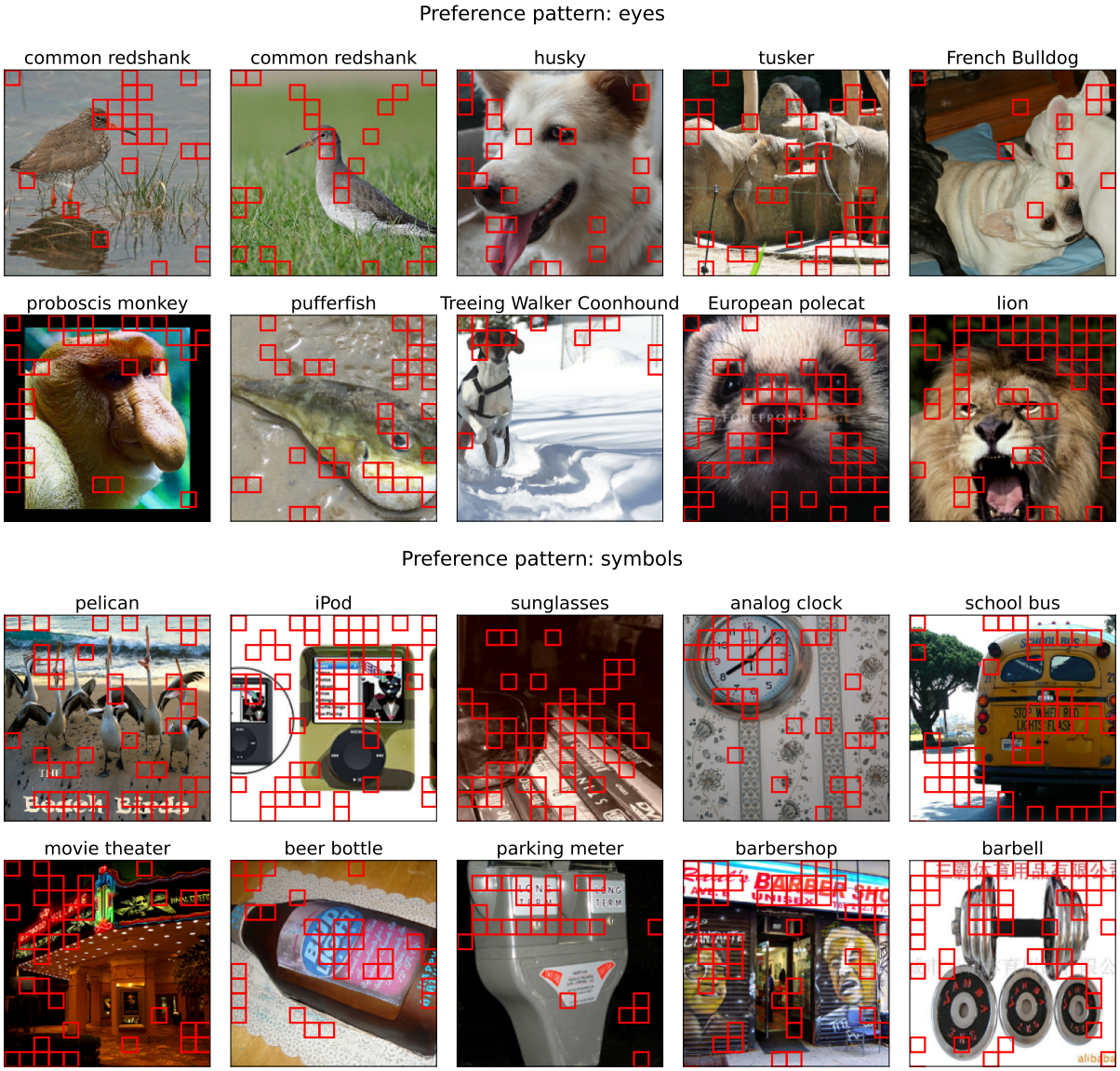

Figure 11: Visualization of expert preference pattern examples. Red bounding boxes highlight the image tokens assigned to the expert.