When Do We Not Need Larger Vision Models?

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Scaling up the size of vision models has been the *de facto* standard to obtain more powerful visual representations. In this work, we discuss the point beyond which larger vision models are *not* necessary. We demonstrate the power of **S**caling on **S**cales ($\mathbf{S^2}$), whereby a pre-trained and frozen smaller vision model (*e.g.*, ViT-B or ViT-L), run over multiple image scales, can outperform larger models (*e.g.*, ViT-H or ViT-G) on classification, segmentation, depth estimation, Multimodal LLM (MLLM) benchmarks, and robotic manipulation. We further show that features of larger vision models can be well approximated by those of multi-scale smaller models through a linear transform, which suggests a multi-scale smaller model has comparable learning capacity to a larger model.

1 Introduction

Scaling up model size has been one of the key drivers of recent progress in various domains of artificial intelligence, including language modeling [4, 27, 40], image and video generation [45, 31, 17, 3], *etc*. Similarly, for visual understanding, larger models have consistently shown improvements across a wide range of downstream tasks given sufficient pre-training data [37, 48, 6, 26]. This trend has led to the pursuit of gigantic models with up to tens of billions of parameters as a default strategy for achieving more powerful visual representations and enhanced performance on downstream tasks [6, 9, 36, 12].

In this work, we revisit the question: *Is a larger model always necessary for better visual understanding?* Instead of scaling up model size, we consider scaling on the dimension of image scales—which we call **S**caling on **S**cales ($\mathbf{S^2}$). With S^2, a pre-trained and frozen smaller vision model (*e.g.*, ViT-B or ViT-L) is run on multiple image scales to generate a multi-scale representation. We take a model pre-trained on single image scale (*e.g.*, 224^2), interpolate the image to multiple scales (*e.g.*, 224^2, 448^2, 672^2), extract features on each scale by splitting larger images into sub-images of the regular size (224^2) and processing each separately before pooling them and concatenating with features from the original representation (Figure 1).

From evaluations on visual representations of various pre-trained models (*e.g.*, ViT [10], DINOv2 [26], OpenCLIP [6], MVP [30]), we show that smaller models with S^2 scaling consistently outperform larger models on classification, semantic segmentation, depth estimation, MLLM benchmarks, and robotic manipulation, with significantly fewer parameters (*e.g.*, $0.07\times$) and comparable GFLOPS.

While these results suggest larger models are not necessary for better downstream performance, it is still not clear if they are irreplaceable in terms of representation learning, *i.e.*, is there any representation that larger models can learn but smaller models cannot? Surprisingly, we find that the features of larger models can be well approximated by multi-scale smaller models through a single linear transform, which means smaller models should have at least a similar learning capacity of their larger counterparts.

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

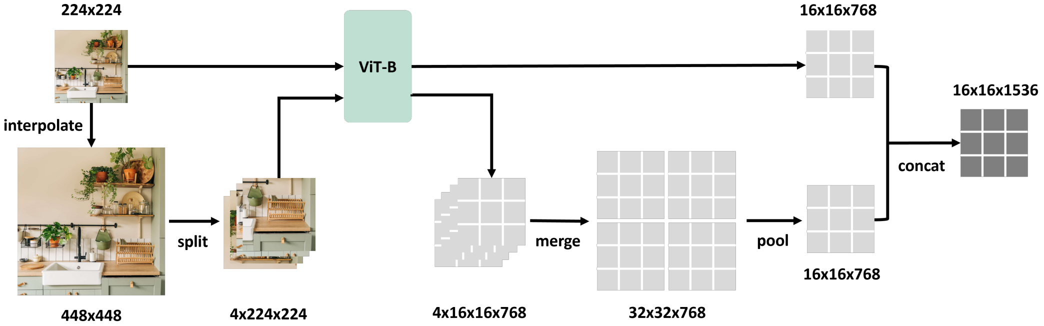


Figure 1: **S^2-Wrapper is a simple mechanism that extends any pre-trained vision model to multiple image scales in a parameter-free manner.** Taking ViT-B as an example, S^2-Wrapper first interpolates the input image to different scales (*e.g.*, 224^2 and 448^2) and splits each into several sub-images of the same size as the default input size ($448^2 \rightarrow 4 \times 224^2$). For each scale, all sub-images are fed into the same model and the outputs (*e.g.*, 4×16^2) are merged into feature map of the whole image (32^2). Feature maps of different scales are average-pooled to the original spatial size (16^2) and concatenated together. The final multi-scale feature has the same spatial shape as single-scale feature while having higher channel dimension (*e.g.*, 1536 *vs*. 768).

2 The Power of Scaling on Scales

2.1 Scaling Pre-Trained Vision Models to Multiple Image Scales

We first introduce S^2-Wrapper, a parameter-free mechanism to enable multi-scale feature extraction on any pre-trained vision model. Regular vision models are normally pre-trained at a single image scale (*e.g.*, 224^2). S^2-Wrapper extends a pre-trained model to multiple image scales (*e.g.*, 224^2, 448^2) by splitting different scales of images to the same size as seen in pre-training. Specifically, given the image at 224^2 and 448^2 scales, S^2-Wrapper first divides the 448^2 image into four 224^2 sub-images, which along with the original 224^2 image are fed to the same pre-trained model. The features of four sub-images are merged back to the large feature map of the 448^2 image, which is then average-pooled to the same size as the feature map of 224^2 image. Output is the concatenation of feature maps across scales. The whole process is illustrated in Figure 1. Note that instead of directly using the 448^2 resolution image, we obtain the 448^2 image by interpolating the 224^2 image. This is to make sure no additional high-resolution information is introduced so we can make a fair comparison with model size scaling which never sees the high-resolution image. On the other hand, we interpolate the large feature map into the regular size to make sure the number of output tokens stays the same, making it a fair comparison to larger models which give the same number of tokens for downstream applications such as MLLMs. Note that we do not claim the novelty of extracting multi-scale features since concurrent work (*e.g.*, [21]) also use similar methods. Instead, we only choose the simplest algorithm design and study its scaling property.

2.2 Scaling on Image Scales Can Beat Scaling on Model Size

S^2-Wrapper enables S^2 scaling, *i.e.*, keeping the same size of a pre-trained model while getting more and more powerful features by running on more and more image scales. Here we compare the scaling curve of S^2 to the regular approach of scaling up model size and show that S^2 scaling is a competitive, and in some cases, preferred scaling approach. To get a holistic analysis of two scaling approaches, we test their scaling curves on three representative tasks (image classification, semantic segmentation, and depth estimation) which correspond to the three dimensions of vision model capability [25], as well as on MLLMs and robotic manipulation which reflect the comprehensive ability of visual understanding. Please find the results on MLLMs below and other results in Appendix A.

Case study: Multimodal LLMs. We compare S^2 scaling and model size scaling on MLLMs. We use a LLaVA [20]-style model where LLM is a Vicuna-7B [7] and the vision backbone is OpenCLIP. We keep the same LLM and only change the vision backbone. For model size scaling,

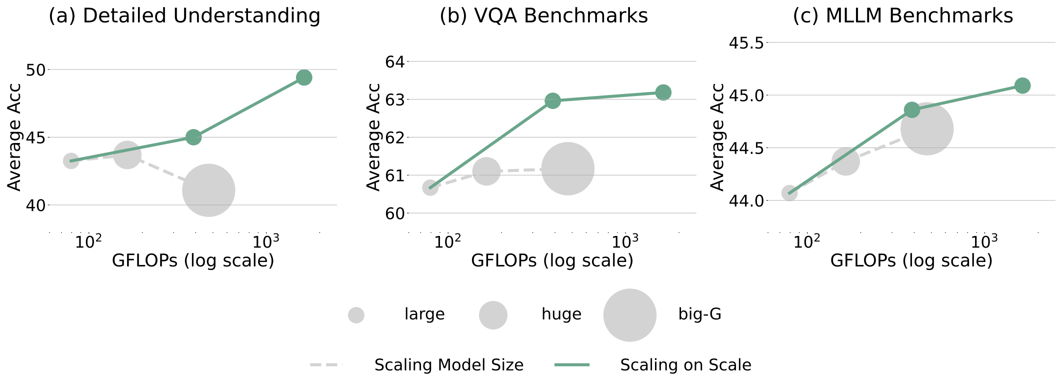


Figure 2: **Comparison of S^2 scaling and model size scaling on MLLM.** For each type of tasks, we test large, huge, and big-G models for model size scaling (plotted in gray curve). For S^2 scaling (plotted in green curve), we test three sets of scales including (1x), (1x, 2x), (1x, 2x, 4x). S^2 scaling has comparable or better scaling curve than model size scaling on all three types of benchmarks. Using large image scales consistently gives better performance while using larger model can degrade model performance in certain cases.

we test vision model sizes of large, huge, and big-G. For S^2 scaling, we keep the large-size model and test scales of (224^2), (224^2, 448^2), and (224^2, 448^2, 896^2). For all experiments, we keep the vision backbone frozen and only train a projector layer between the vision feature and LLM input space as well as a LoRA [16] on LLM. We follow the same training recipe as in LLaVA-1.5 [19]. We evaluate three types of benchmarks: (i) visual detail understanding (V^* [42]), (ii) VQA benchmarks (VQAv2 [13], TextVQA [34], VizWiz [14]), and (iii) MLLM benchmarks (MMMU [47], MathVista [24], MMBench [22], SEED-Bench [18], MM-Vet [46]).

A comparison of the two scaling approaches is shown in Figure 2. We report the average accuracy on each type of benchmarks. We can see that on all three types of benchmarks, S^2 scaling on large-size models performs better than larger models, using similar GFLOPs and much smaller model sizes. Especially, scaling to 896^2 improves the accuracy of detailed understanding by about 6%. On all benchmarks, larger image scales consistently improve performance while bigger models sometimes fail to improve or even hurt performance. These results suggest S^2 is a preferable scaling approach for vision understanding in MLLMs. Please see the complete results on MLLMs in Appendix B.

2.3 Can Smaller Models Learn What Larger Models Learn?

Despite the superior performance, can multi-scale smaller models replace larger models for representation learning as well? We design experiments to study how much of the representation of larger models is also learned by multi-scale smaller models. Surprisingly, our results suggest that *most, if not all, of the representation of larger models is also learned by multi-scale smaller models.*

To quantify how much of the representation of a larger model (*e.g.*, ViT-L) is also learned by a multi-scale smaller model (*e.g.*, ViT-B-S^2), we adopt a reconstruction-based evaluation, *i.e.*, we train a linear transform to reconstruct the representation of a larger model from that of a multi-scale smaller model. Intuitively, low reconstruction loss means the representation of larger model can be equivalently learned by the multi-scale smaller model (through a linear transform) to a large extent. More formally, the reconstruction loss reflects the mutual information between two sets of representations. If we use MSE loss for reconstruction, the mutual information equals $I = -\log(l/l_0)$, where l is the reconstruction loss and l_0 is the loss of vanilla reconstruction where the large model representation is reconstructed by a dummy vector (See Appendix D). This quantifies how much information in the larger model representation is also contained in the multi-scale smaller model. We use a linear transform for reconstruction to measure the information that is useful for downstream tasks considering the task decoders are usually light-weight modules such as a single linear layer [44].

Moreover, in practice we find the reconstruction loss is usually nowhere near zero. We hypothesize this is because part of the feature is *non-reconstructable* by nature, *i.e.*, feature that is not relevant to the pre-training task and is learned due to randomness in weight initialization, optimization dynamics,

Table 1: **Reconstructing representation of larger models from representation of regular or multi-scale smaller models.** We test three classes of models (ViT, OpenCLIP, and MAE), and for each class we test base, multi-scale base (Base-S^2), and huge or giant model. We report the reconstruction loss, the amount of information reconstructed, and the percentage of information reconstructed compared to huge or giant model on train and test set of ImageNet.

Model Class	Target	Source	Train Set			Test Set		
			Loss	Info	Ratio (%)	Loss	Info	Ratio (%)
ViT	Large	Base	0.1100	0.440	82.9%	0.0994	0.524	87.6%
		Base-S^2	0.1040	0.521	**98.1%**	0.0942	0.601	**100.5%**
		Huge	0.1033	0.531	100%	0.0944	0.598	100%
MAE	Large	Base	0.0013	7.460	97.3%	0.0010	7.840	96.0%
		Base-S^2	0.0011	7.694	**100.3%**	0.0009	7.972	**97.6%**
		Huge	0.001	7.669	100%	0.0008	8.169	100%
OpenCLIP	Large	Base	0.3693	1.495	92.7%	0.3413	1.723	90.7%
		Base-S^2	0.3408	1.611	**99.9%**	0.3170	1.830	**96.3%**
		Giant	0.3402	1.613	100%	0.3022	1.900	100%
OpenCLIP	Huge	Base	0.3926	1.407	83.2%	0.4231	1.413	80.8%
		Base-S^2	0.3670	1.504	**88.9%**	0.3970	1.505	**86.0%**
		Giant	0.3221	1.692	100%	0.3354	1.749	100%

etc., thus cannot be reconstructed from another model's feature. To this end, we use an even larger (*e.g.*, ViT-G) model to reconstruct the large model features as a comparison. Its reconstruction loss and corresponding mutual information are denoted by l^* and $I^* = -\log(l^*/l_0)$. If we assume that, when pre-trained on the same task and the same dataset, any task-relevant feature learned by a smaller model can also be learned by a larger model, then all the useful features in a large-size model should be reconstructable by a huge or giant model as well. This means I^*, the amount of information reconstructed from a huge or giant model, should serve as an *upper bound* of I. We empirically find this is indeed the case (see below). Therefore, we use the reconstruction ratio I/I^* to measure how much representation in a larger model is also learned by a multi-scale smaller model.

We evaluate three classes of models: (i) ViT [10] pre-trained on ImageNet-21k, (ii) OpenCLIP [6] pre-trained on LAION-2B, and (iii) MAE [15] pre-trained on ImageNet-1k. Reconstruction loss is averaged over all output tokens and is evaluated on ImageNet-1k. Results are shown in Table 1. Compared to base models, we observe that multi-scale base models consistently have lower loss and reconstructs more information of the large model representation (*e.g.*, 0.521 *vs*. 0.440 for ViT). More interestingly, we find that the amount of information reconstructed from a multi-scale base model is usually close to that of a huge or giant model, although sometimes slightly lower but never exceeding by a large margin. For example, while OpenCLIP-Base reconstructs 92.7% of the information, the multi-scale base model can reconstruct 99.9%. For other models, the reconstruction ratio of Base-S^2 model is usually close to 100% while never exceeding by more than 0.5%. This implies (i) huge/giant models are indeed a valid upper bound of feature reconstruction, and (ii) most part of the feature of larger models is also learned by multi-scale smaller models. The only exception is when we reconstruct OpenCLIP-Huge feature, the reconstruction ratio is 88.9%. Although it's not near 100%, it is still significantly better than the base-size model which means at least a large part of the huge model feature is still multi-scale feature. These results imply smaller models with S^2 scaling should have at least a similar level of capacity to learn what larger models learn. On the other hand, we also notice that the reconstruction ratio on test set can be lower than train set (*e.g.* 96.3% *vs*. 99.9% on OpenCLIP-L). We hypothesize this is because we only apply multi-scale after pre-training and the base model feature pre-trained on single image scale only has weaker generalizability.

3 Conclusion

In this work, we ask the question *is a larger model always necessary for better visual understanding?* We find that scaling on the dimension of image scales—which we call Scaling on Scales (S^2)—instead of model size usually obtains better performance on a wide range of downstream tasks. We further show that smaller models with S^2 can learn most of representation that larger models learn.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 2023.

[3] Tim Brooks, Bill Peebles, Connor Homes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. URL `https://openai.com/research/video-generation-models-as-world-simulators`.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[5] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022.

[6] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2818–2829, 2023.

[7] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2023.

[8] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning, 2023.

[9] Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *International Conference on Machine Learning*, pages 7480–7512. PMLR, 2023.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[11] Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, et al. Vlmevalkit: An open-source toolkit for evaluating large multi-modality models. *arXiv preprint arXiv:2407.11691*, 2024.

[12] Alaaeldin El-Nouby, Michal Klein, Shuangfei Zhai, Miguel Angel Bautista, Alexander Toshev, Vaishaal Shankar, Joshua M Susskind, and Armand Joulin. Scalable pre-training of large autoregressive image models. *arXiv preprint arXiv:2401.08541*, 2024.

[13] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913, 2017.

[14] Danna Gurari, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3608–3617, 2018.

[15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[16] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[17] Dan Kondratyuk, Lijun Yu, Xiuye Gu, José Lezama, Jonathan Huang, Rachel Hornung, Hartwig Adam, Hassan Akbari, Yair Alon, Vighnesh Birodkar, et al. Videopoet: A large language model for zero-shot video generation. *arXiv preprint arXiv:2312.14125*, 2023.

[18] Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. Seed-bench: Benchmarking multimodal llms with generative comprehension. *arXiv preprint arXiv:2307.16125*, 2023.

[19] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. *arXiv preprint arXiv:2310.03744*, 2023.

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

[21] Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024. URL `https://llava-vl.github.io/blog/2024-01-30-llava-next/`.

[22] Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. Mmbench: Is your multi-modal model an all-around player? *arXiv preprint arXiv:2307.06281*, 2023.

[23] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022.

[24] Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*, 2023.

[25] Jitendra Malik, Pablo Arbeláez, Joao Carreira, Katerina Fragkiadaki, Ross Girshick, Georgia Gkioxari, Saurabh Gupta, Bharath Hariharan, Abhishek Kar, and Shubham Tulsiani. The three r's of computer vision: Recognition, reconstruction and reorganization. *Pattern Recognition Letters*, 72:4–14, 2016.

[26] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

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

[28] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[29] Ilija Radosavovic, Baifeng Shi, Letian Fu, Ken Goldberg, Trevor Darrell, and Jitendra Malik. Robot learning with sensorimotor pre-training. *arXiv preprint arXiv:2306.10007*, 2023.

[30] Ilija Radosavovic, Tete Xiao, Stephen James, Pieter Abbeel, Jitendra Malik, and Trevor Darrell. Real-world robot learning with masked visual pre-training. In *Conference on Robot Learning*, pages 416–426. PMLR, 2023.

[31] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

[32] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[33] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12*, pages 746–760. Springer, 2012.

[34] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8317–8326, 2019.

[35] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv preprint arXiv:2106.10270*, 2021.

[36] Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. Eva-clip: Improved training techniques for clip at scale. *arXiv preprint arXiv:2303.15389*, 2023.

[37] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[38] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[39] Qwen Team. Introducing qwen-vl, Jan 2024. URL `https://qwenlm.github.io/blog/qwen-vl/`.

[40] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[41] Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Xixuan Song, et al. Cogvlm: Visual expert for pretrained language models. *arXiv preprint arXiv:2311.03079*, 2023.

[42] Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal llms. *arXiv preprint arXiv:2312.14135*, 2023.

[43] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 418–434, 2018.

[44] Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. A theory of usable information under computational constraints. *arXiv preprint arXiv:2002.10689*, 2020.

[45] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2(3):5, 2022.

[46] Weihao Yu, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. Mm-vet: Evaluating large multimodal models for integrated capabilities. *arXiv preprint arXiv:2308.02490*, 2023.

[47] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. *arXiv preprint arXiv:2311.16502*, 2023.

[48] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12104–12113, 2022.

[49] Wenliang Zhao, Yongming Rao, Zuyan Liu, Benlin Liu, Jie Zhou, and Jiwen Lu. Unleashing text-to-image diffusion models for visual perception. *arXiv preprint arXiv:2303.02153*, 2023.

[50] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017.

A Additional Comparison of S^2 and Model Size Scaling

Case study: image classification, semantic segmentation, and depth estimation. We use ImageNet [32], ADE20k [50], and NYUv2 [33] datasets for each task, respectively. We test on three families of pre-trained models (ViT [10], DINOv2 [26], and OpenCLIP [6]), spanning pre-training with different datasets (ImageNet-21k, LVD-142M, LAION-2B) and different pre-training objectives (supervised, unsupervised, and weakly-supervised). To see if the same observation holds for convolutional networks, we also test on ConvNeXt [23] (See Appendix E). To fairly evaluate the representation learned from pre-training, we freeze the backbone and only train the task-specific head for all experiments. We use a single linear layer, Mask2former [5], and VPD depth decoder [49] as decoder heads for three tasks, respectively. For model size scaling, we test the performance of base, large, and huge or giant size of each model on each task. For S^2 scaling, we test three sets of scales including (1x), (1x, 2x), (1x, 2x, 3x). For example, for ViT on ImageNet classification, we use three sets of scales: (224^2), (224^2, 448^2), and (224^2, 448^2, 672^2), which have the comparable GFLOPs as ViT-B, ViT-L, and ViT-H, respectively. Note that the scales for specific models and tasks are adjusted to match the GFLOPS of respective model sizes. The detailed configurations for each experiment can be found in Appendix C.

The scaling curves are shown in Figure 4. We can see that in six out of nine cases ((a), (d), (e), (f), (g), (i)), S^2 scaling from base models gives a better scaling curve than model size scaling, outperforming large or giant models with similar GFLOPs and much fewer parameters. In two cases ((b) and (h)), S^2 scaling from base models has less competitive results than large models, but S^2 scaling from large models performs comparatively with giant models. The only failure case is (c) where both base and large models with S^2 scaling fail to compete with the giant model. Note that ViT-H is worse than ViT-L on all three tasks possibly due to the sub-optimal pre-training recipe [35]. We observe that S^2 scaling has more advantages on dense prediction tasks such as segmentation and depth estimation, which matches the intuition that multi-scale features can offer better detailed understanding which is especially required by these tasks. For image classification, S^2 scaling is sometimes worse than model size scaling (*e.g.*, multi-scale DINOv2-B *vs*. DINOv2-L). We hypothesize this is due to the weak generalizability of the base model feature because we observe that the multi-scale base model has a lower training loss than the large model despite the worse performance, which indicates overfitting.

Case study: robotic manipulation. We compare S^2 and model size scaling on a robotic manipulation task of cube picking. The task requires controlling a robot arm to pick up a cube on the table. We train a vision-based end-to-end policy on 120 demos using behavior cloning, and evaluate the success rate of picking on 16 randomly chosen cube positions, following the setting in [29]. We use MVP [30] as the pre-trained vision encoder to extract visual features which are fed to the policy. Please refer to Appendix C for the detailed setting. To compare S^2 and model size scaling, we evaluate base and large models with single scale of (224^2), as well as a multi-scale base model with scales of (224^2, 448^2). Results are shown in Figure 3. Scaling from base to large model improves the success rate by about 6%, while scaling to larger image scales improves the success rate by about 20%. This demonstrates the advantage of S^2 over model size scaling on robotic manipulation tasks as well.

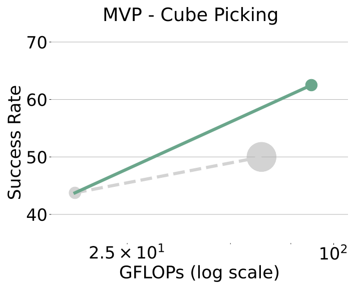


Figure 3: **S^2 *vs*. model size scaling on cube picking task.** S^2 scaling on base-size model improves the success rate by about 20%.

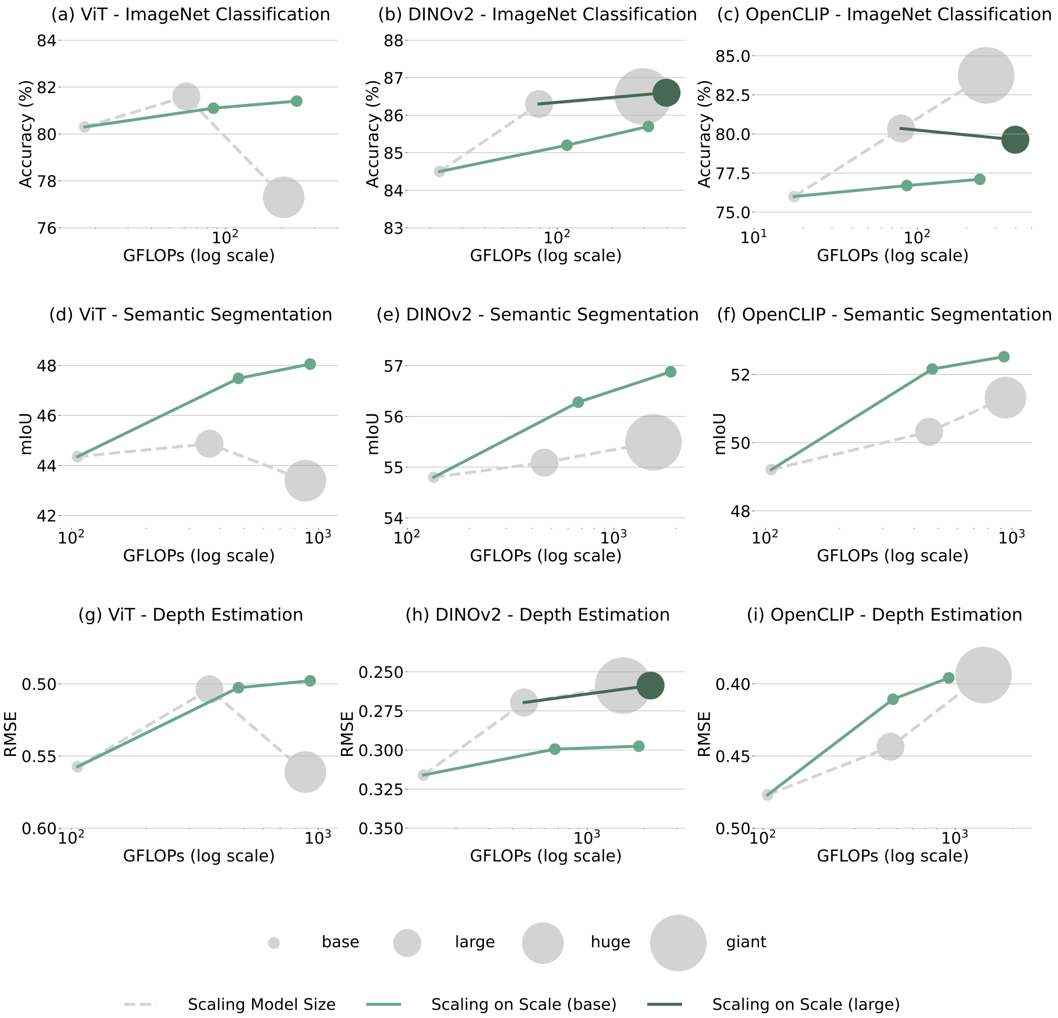


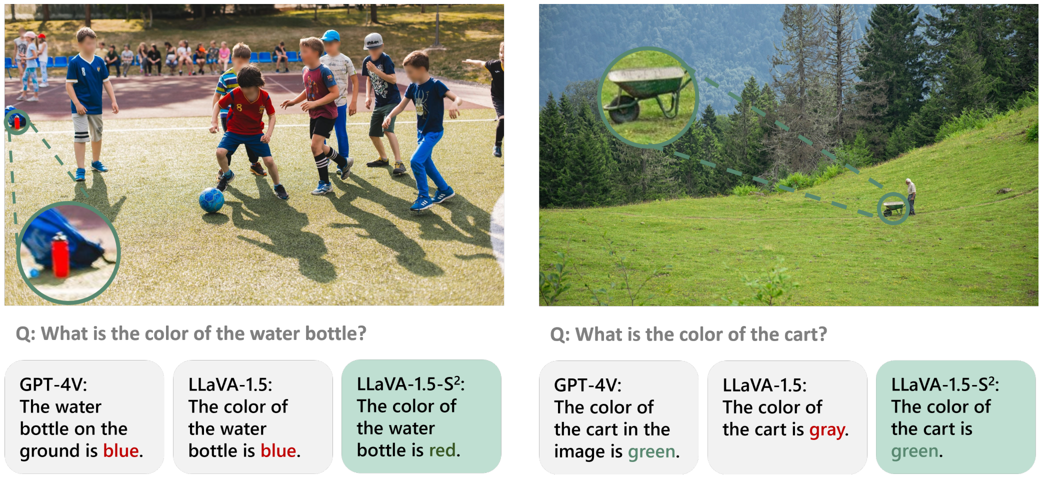
Figure 4: **Comparison of S^2 scaling and model size scaling** on three models (ViT, DINOv2, and OpenCLIP) and three tasks (ImageNet classification, semantic segmentation, and depth estimation). For each model and each task, we test base, large, and huge/giant models for model size scaling (plotted in gray curve). For S^2 scaling (plotted in green curve), we test three sets of scales from single-scale (1x) to multi-scale (up to 3x), and we adjust each set of scale so that it matches the GFLOPs of the respective model size. Note that for specific models and tasks, we test S^2 scaling on both base and large models (plotted in light green and dark green curves separately). We can see that in (a), (d), (e), (f), (g), and (i), the base model with S^2 scaling already achieves comparable or better performances than larger models with similar GFLOPs and much smaller model size. For (b), (h), S^2 scaling from the large model is comparable with the giant model, again with similar GFLOPs and fewer parameters. The only failure case is (c), where S^2 scaling on either base or large models does not compete with model size scaling.

B Complete results of MLLM

We observe that LLaVA-1.5, when equipped with S^2 scaling, is already competitive or better than state-of-the-art open-source and even commercial MLLMs. Results are shown in Table 2. Here we use OpenAI CLIP [28] as the vision model for fair comparison. On visual detail understanding, LLaVA-1.5 with S^2 scaling outperforms all other open-source MLLMs as well as commercial models such as Gemini Pro and GPT-4V. This is credited to the highly fine-grained features we are able to extract by scaling image resolution to 1008^2. A qualitative example is shown in Figure 5. We can see that LLaVA-1.5 with S^2 is able to recognize an extremely small object that only takes 23×64 pixels in a 2250×1500 image and correctly answer the question about it. In the meantime, both GPT-4V and LLaVA-1.5 fail to give the correct answer. More qualitative examples are shown in Appendix H. On VQA and MLLM benchmarks, S^2 consistently improves the model performance as well, especially on benchmarks such as TextVQA which requires understanding of the fine details. Note that the improvement on certain MLLM benchmarks such as MathVista is not as significant as others, which is probably because these benchmarks require strong mathematical or reasoning capabilities which are not achievable by only improving vision but require stronger LLMs as well. In contrast to previous experiments, here we directly use the high-resolution image instead of interpolating from the low-resolution image in order to compare with the state of the arts. Note that despite the large image scale, we keep the same number of image tokens as baseline LLaVA-1.5 since we interpolate the feature map of the large-scale images to the same size as that of the original image (see Section 2.1). This makes sure the context length (and thus the computational cost) of LLM does not increase when using larger image scales, allowing us to use much higher resolution than the baselines.

Table 2: **Results on MLLM.** We evaluate three types of benchmarks: visual detail understanding (V^* [42]), VQA benchmarks (VQAv2 [13], TextVQA [34], VizWiz [14]), and MLLM benchmarks (MMMU [47], MathVista [24], MMBench [22], SEED-Bench [18], MM-Vet [46]). Notably, S^2 significantly improves the detailed understanding capability on V^* benchmark, outperforming commercial models such as GPT-4V.

			Visual Detail		**VQA Benchmarks**			**MLLM Benchmarks**				
Model	Res.	#Token	V^*_{Att}	V^*_{Spa}	VQA^{v2}	VQA^{T}	Viz	MMMU	Math	MMB	SEED	MMVet
Commercial or proprietary models												
GPT-4V [1]	-	-	**51.3**	60.5	77.2	78.0	-	**56.8**	**49.9**	75.8	71.6	**67.6**
Gemini Pro [38]	-	-	40.9	59.2	71.2	74.6	-	47.9	45.2	73.6	70.7	64.3
Qwen-VL-Plus [39]	-	-	-	-	-	**78.9**	-	45.2	43.3	-	-	-
Open-source models												
InstructBLIP-7B [8]	224	-	25.2	47.4	-	50.1	34.5	-	-	36.0	-	26.2
QwenVL-7B [2]	448	1024	-	-	78.8	63.8	35.2	-	-	38.2	-	-
QwenVL-Chat-7B [2]	448	1024	-	-	78.2	61.5	38.9	-	-	60.6	-	-
CogVLM-Chat [41]	490	1225	-	-	**82.3**	70.4	-	41.1	34.5	**77.6**	**72.5**	51.1
LLaVA-1.5-7B [19]	336	576	43.5	56.6	78.5	58.2	50.0	36.2	25.2	64.3	65.7	30.5
LLaVA-1.5-7B-S^2	1008	576	**51.3**	61.8	80.0	61.0	50.1	37.7	25.3	66.2	67.9	32.4
LLaVA-1.5-13B [19]	336	576	41.7	55.3	80.0	61.3	53.6	36.4	27.6	67.8	68.2	35.4
LLaVA-1.5-13B-S^2	1008	576	50.4	**63.2**	80.9	63.1	**56.0**	37.4	27.8	67.9	68.9	36.4



Q: What is the color of the water bottle?

GPT-4V: The water bottle on the ground is blue.

LLaVA-1.5: The color of the water bottle is blue.

LLaVA-1.5-S^2: The color of the water bottle is red.

Q: What is the color of the cart?

GPT-4V: The color of the cart in the image is green.

LLaVA-1.5: The color of the cart is gray.

LLaVA-1.5-S^2: The color of the cart is green.

Figure 5: **LLaVA-1.5 with S^2 scaling is able to recognize extremely fine-grained details in an image**, *e.g.*, the color of a water bottle which lives in only 23×64 pixels of a 2250×1500 image.

C Detailed Experimental Settings and Full Results

The details of the models and the corresponding results on image classification, semantic segmentation, and depth estimation are listed in Table 3, 4, and 5, respectively. We use ImageNet-21k pre-trained checkpoints for ViT[1,2,3], LVD-142M pre-trained checkpoints for DINOv2[4,5,6], and LAION-2B pre-trained checkpoints for OpenCLIP[7,8,9]. For each model type (ViT [10], DINOv2 [26], OpenCLIP [6]), we choose the scales so that the models with S^2 have comparable number of FLOPs with corresponding larger models. For image classification, we train a linear classifier for 30 epochs with learning rate of 0.0005 and batch size of 512. For semantic segmentation, we train a Mask2Former decoder [5] following the configurations here[10]. For depth estimation, we train a VPD depth decoder [49] following the configurations here[11].

Table 6 and 7 show the model details and full results for V*, VQA tasks, and MLLM benchmarks. We use OpenCLIP with large, huge, and big-G sizes, and also large-size model with (224^2), $(224^2, 448^2)$, $(224^2, 448^2, 672^2)$ scales. We follow the training and testing configurations in LLaVA-1.5[12]. For evaluations on certain MLLM benchmarks such as MMMU [47], since it is not supported in the LLaVA-1.5 repo, we use VLMEvalKit [11] for evaluation[13].

Table 8 shows the model details and full results for the robotic manipulation task of cube picking. We use MVP [30] as the vision backbone and use base and large size as well as base size with $(224^2, 448^2)$ scales. The vision backbone is frozen and extracts the visual feature for the visual observation at each time step. We train a transformer that takes in the visual features, proprioception and actions for the last 16 steps and outputs the actions for the next 16 steps. We train the model

---

[1] https://huggingface.co/google/vit-base-patch16-224-in21k

[2] https://huggingface.co/google/vit-large-patch16-224-in21k

[3] https://huggingface.co/google/vit-huge-patch14-224-in21k

[4] https://dl.fbaipublicfiles.com/dinov2/dinov2_vitb14/dinov2_vitb14_pretrain.pth

[5] https://dl.fbaipublicfiles.com/dinov2/dinov2_vitl14/dinov2_vitl14_pretrain.pth

[6] https://dl.fbaipublicfiles.com/dinov2/dinov2_vitg14/dinov2_vitg14_pretrain.pth

[7] https://huggingface.co/laion/CLIP-ViT-B-16-laion2B-s34B-b88K

[8] https://huggingface.co/laion/CLIP-ViT-L-14-laion2B-s32B-b82K

[9] https://huggingface.co/laion/CLIP-ViT-g-14-laion2B-s34B-b88K

[10] https://github.com/open-mmlab/mmsegmentation/blob/main/configs/mask2former/mask2former_r50_8xb2-160k_ade20k-512x512.py

[11] https://github.com/open-mmlab/mmsegmentation/blob/main/configs/vpd/vpd_sd_4xb8-25k_nyu-512x512.py

[12] https://github.com/haotian-liu/LLaVA

[13] https://github.com/open-compass/VLMEvalKit

Table 3: Configurations of models and corresponding results on ImageNet classification.

	Model Size	Scales	#Params	#FLOPs	Acc.
ViT	Base	(224^2)	86M	17.6G	80.3
	Base	(224^2, 448^2)	86M	88.1G	81.1
	Base	(224^2, 448^2, 672^2)	86M	246.0G	81.4
	Large	(224^2)	307M	61.6G	81.6
	Huge	(224^2)	632M	204.9G	77.3
DINOv2	Base	(224^2)	86M	22.6G	84.5
	Base	(224^2, 448^2)	86M	112.8G	85.2
	Base	(224^2, 448^2, 672^2)	86M	315.9G	85.7
	Large	(224^2)	303M	79.4G	86.3
	Large	(224^2, 448^2)	303M	397.1G	86.6
	Giant	(224^2)	632M	295.4G	86.5
OpenCLIP	Base	(224^2)	86M	17.6G	76.0
	Base	(224^2, 448^2)	86M	86.1G	76.7
	Base	(224^2, 448^2, 672^2)	86M	241.0G	77.1
	Large	(224^2)	303M	79.4G	80.4
	Large	(224^2, 448^2)	303M	397.1G	79.6
	Giant	(224^2)	1012M	263.4G	83.8

with behavior cloning on 120 self-collected demos. We test the model on 16 randomly selected cube positions and report the rate of successfully picking up the cube at these positions.

Table 4: Configurations of models and corresponding results on ADE20k semantic segmentation.

	Model Size	Scales	#Params	#FLOPs	mIoU
ViT	Base	(512^2)	86M	105.7G	44.4
	Base	(256^2, 512^2, 1024^2)	86M	474.7G	47.8
	Base	(256^2, 512^2, 1536^2)	86M	926.7G	48.0
	Large	(512^2)	307M	362.1G	44.9
	Huge	(512^2)	632M	886.2G	43.4
DINOv2	Base	(518^2)	86M	134.4G	54.8
	Base	(518^2, 1036^2)	86M	671.8G	56.3
	Base	(518^2, 1036^2, 1554^2)	86M	1881G	56.9
	Large	(518^2)	303M	460.9G	55.1
	Giant	(518^2)	632M	1553G	55.5
OpenCLIP	Base	(512^2)	86M	105.7G	49.2
	Base	(256^2, 512^2, 1024^2)	86M	474.7G	52.2
	Base	(256^2, 512^2, 1536^2)	86M	926.7G	52.6
	Large	(518^2)	303M	460.9G	50.3
	Huge	(518^2)	632M	940.2G	51.3

D Derivation of Mutual Information

Denote the features from two models by $\boldsymbol{x} \in \mathbb{R}^{d_x}$ and $\boldsymbol{y} \in \mathbb{R}^{d_y}$ which follow the distribution $p(\mathbf{x})$ and $p(\mathbf{y})$, respectively. We make the simplest assumption that both the distribution and the conditional distribution of the features are isotropic gaussian distributions, *i.e.*, $p(\mathbf{y}) \sim \mathcal{N}(\hat{\boldsymbol{\mu}}, \sigma^2 \boldsymbol{I})$ and $p(\mathbf{y}|\mathbf{x}) \sim \mathcal{N}(\hat{f}(\mathbf{x}), \sigma'^2 \boldsymbol{I})$, where $f(\cdot)$ is a linear transform. The differential entropy and conditional differential entropy of $\mathbf{y}$ is $h(\mathbf{y}) = d_y \log \sigma + C$ and $h(\mathbf{y}|\mathbf{x}) = d_y \log \sigma' + C$, where C is a constant. The mutual information between features of two models is $I(\mathbf{x}; \mathbf{y}) = h(\mathbf{y}) - h(\mathbf{y}|\mathbf{x}) = d_y \log \sigma - d_y \log \sigma'$.

Table 5: Configurations of models and corresponding results on NYUv2 depth estimation.

	Model Size	Scales	#Params	#FLOPs	RMSE
ViT	Base	(512^2)	86M	105.7G	0.5575
	Base	$(256^2, 512^2, 1024^2)$	86M	474.7G	0.5127
	Base	$(256^2, 512^2, 1536^2)$	86M	926.7G	0.5079
	Large	(512^2)	307M	362.1G	0.5084
	Huge	(512^2)	632M	886.2G	0.5611
DINOv2	Base	(504^2)	86M	134.4G	0.3160
	Base	$(504^2, 1008^2)$	86M	671.8G	0.2995
	Base	$(504^2, 1008^2, 1512^2)$	86M	1881G	0.2976
	Large	(504^2)	303M	460.9G	0.2696
	Large	$(504^2, 1008^2)$	303M	2170G	0.2584
	Giant	(504^2)	632M	1553G	0.2588
OpenCLIP	Base	(512^2)	86M	105.7G	0.4769
	Base	$(256^2, 512^2, 1024^2)$	86M	474.7G	0.4107
	Base	$(256^2, 512^2, 1536^2)$	86M	926.7G	0.3959
	Large	(504^2)	303M	460.9G	0.4436
	Huge	(504^2)	632M	940.2G	0.3939

Table 6: Configurations of models and corresponding results on V* and VQA tasks.

	Model Size	Scales	#Params	#FLOPs	V^*_{Att}	V^*_{Spa}	VQA^{v2}	VQA^{T}	Viz
OpenCLIP	Large	(224^2)	304M	79.4G	36.5	50.0	76.6	53.8	51.6
	Large	$(224^2, 448^2)$	304M	389.1G	40.0	50.0	77.8	55.9	55.2
	Large	$(224^2, 448^2, 672^2)$	304M	1634G	35.7	63.2	77.9	56.5	55.3
	Huge	(224^2)	632M	164.6G	37.4	50.0	76.0	54.0	53.3
	big-G	(224^2)	1012M	473.4G	32.2	48.7	76.2	54.0	53.5

Table 7: Configurations of models and corresponding results on MLLM benchmarks.

	Model Size	Scales	#Params	#FLOPs	MMMU	Math	MMB	SEED	MMVet
OpenCLIP	Large	(224^2)	304M	79.4G	35.4	24.0	64.2	65.5	31.6
	Large	$(224^2, 448^2)$	304M	389.1G	37.6	24.2	64.5	66.0	33.0
	Large	$(224^2, 448^2, 672^2)$	304M	1634G	37.8	24.5	64.0	66.3	32.8
	Huge	(224^2)	632M	164.6G	36.1	25.2	64.2	65.6	30.7
	big-G	(224^2)	1012M	473.4G	35.6	25.2	64.8	65.1	32.8

When reconstructing the features $\mathbf{y}$ from another model's features $\mathbf{x}$, the optimal MSE loss would be $l = \min_f \frac{1}{d_y} E||\mathbf{y} - f(\mathbf{x})||_2^2 = \frac{1}{d_y} E||\mathbf{y} - \hat{f}(\mathbf{x})||_2^2 = \sigma'^2$. The optimal MSE loss of reconstructing $\mathbf{y}$ from a dummy constant vector would be $l_0 = \min_{\boldsymbol{\mu}} \frac{1}{d_y} E||\mathbf{y} - \boldsymbol{\mu}||_2^2 = \frac{1}{d_y} E||\mathbf{y} - \hat{\boldsymbol{\mu}}||_2^2 = \sigma^2$. Then we get the mutual information between $\mathbf{x}$ and $\mathbf{y}$ is $I(\mathbf{x}; \mathbf{y}) = d_y \log \sigma - d_y \log \sigma' = -\frac{d_y}{2} \log \frac{\sigma'^2}{\sigma^2} \propto -\log \frac{l}{l_0}$.

E Results on ConvNeXt

To see if convolutional networks have similar behaviors as transformer-based models, we test ConvNeXt [23] models (per-trained on ImageNet-21k[14],[15],[16]) on three tasks: image classification, semantic segmentation, and depth estimation. We use ImageNet [32], ADE20k [50], and NYUv2 [33] datasets for each task. Similarly, we freeze the backbone and only train the task-specific head for all

---

[14] https://dl.fbaipublicfiles.com/convnext/convnext_base_22k_224.pth

[15] https://dl.fbaipublicfiles.com/convnext/convnext_large_22k_224.pth

[16] https://dl.fbaipublicfiles.com/convnext/convnext_xlarge_22k_224.pth

Table 8: Configurations of models and corresponding results on robotic manipulation.

	Model Size	Scales	#Params	#FLOPs	Success Rate
MVP	Base	(224^2)	86M	17.5G	43.8
	Base	(224^2, 448^2)	86M	87.9G	62.5
	Large	(224^2)	307M	61.6G	50.0

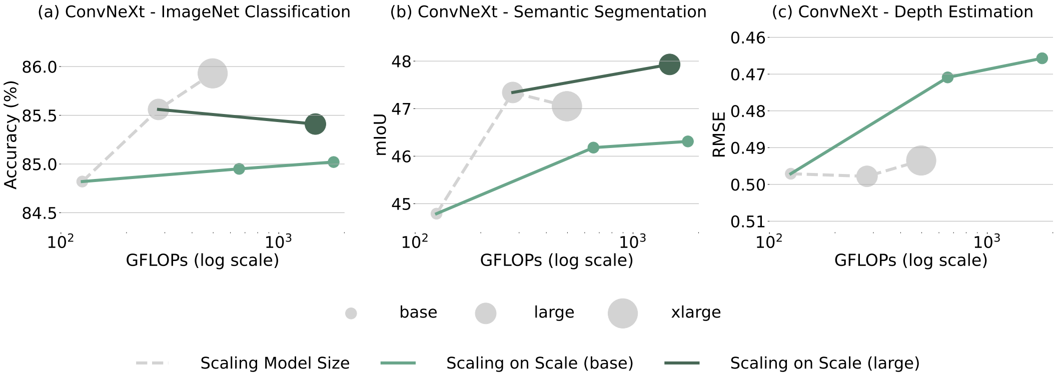


Figure 6: **Comparison of S^2 scaling and model size scaling on ConvNeXt.** We evaluate three tasks: ImageNet classification, semantic segmentation, and depth estimation. For S^2 scaling (plotted in green curve), we test three sets of scales from single-scale (1x) to multi-scale (up to 3x), and we adjust each set of scale so that it matches the GFLOPs of the respective model size. Note that for specific models and tasks, we test S^2 scaling on both base and large models (plotted in light green and dark green curves separately).

experiments, using a single linear layer, UPerNet [43], and VPD depth decoder [49] as the decoder heads for three tasks, respectively. For model size scaling, we test the base, large, and xlarge size performance of ConvNeXt [23] model on each task. For S^2 scaling, we test three sets of scales including (1x), (0.5x, 1x, 2x), and (0.5x, 1x, 2x, 3x).

The detailed curves are shown in Figure 6. We can see that in the depth estimation task (case (c)), S^2 scaling from base model significantly outperforms xlarge model with similar GFLOPs and only $0.25\times$ parameters. In the semantic segmentation task (case (b)), S^2 scaling from base model has less competitive result than larger models, while S^2 scaling from the large model outperforms the xlarge model with more GFLOPs but a smaller number of parameters. The ImageNet classification task (case (a)) is a failure case where S^2 scaling from both base and large model fail to compete with the xlarge model. From the observation above, we see that the convolutional networks show similar properties as transformer-based models: S^2 scaling has more advantages than model size scaling on dense prediction tasks such as segmentation and depth estimation while S^2 scaling is sometimes worse in image classification. This is possibly due to the fact that base and large model are not pre-trained with S^2 (see Section **??**).

F Ablations of Model Design

We conduct the ablations on several designs of S^2-Wrapper. Specifically, (i) we first compare running vision model on sub-images split from the large-scale image with running on the large-scale image directly, and then (ii) we compare concatenating feature maps from different scales with directly adding them together.

Results for (i) are shown in Table 9. We evaluate S^2-Wrapper with or without image splitting on ADE20k semantic segmentation. We test base and large baselines, as well as multi-scale base model with (1x, 2x) and (1x, 2x, 3x) scales separately. We can see that for (1x, 2x) scales, image splitting has better results than no splitting, which is due to image splitting makes sure the input to the model has the same size as in pre-training, and avoids performance degradation caused by positional embedding interpolation when directly running on large images. However, note that even running directly on large images, multi-scale base model still has better results than base and large models, which indicates the effectiveness of S^2 scaling. Furthermore, image splitting enjoys higher computational efficiency because it avoids the quadratic complexity of self-attention. Notice that without image splitting, the training will run into OOM error when using (1x, 2x, 3x) scales.

Table 9: **Ablation of splitting large-scale images.** We compare splitting the large-scale image into regular-sized sub-images *vs*. running the model directly on the large image. We evaluate on ADE20k semantic segmentation. We can see that S^2 scaling with image splitting consistently outperforms directly running on the large image while being more compute-efficient.

Model	Scales	Splitting	mIoU
Base	518^2		54.8
Large	518^2		55.1
Base-S^2	518^2, 1036^2	✗	55.7
Base-S^2	518^2, 1036^2	✓	56.3
Base-S^2	518^2, 1036^2, 1554^2	✗	OOM
Base-S^2	518^2, 1036^2, 1554^2	✓	56.9

Results for (ii) are shown in Table 10. We compare S^2-Wrapper with concatenating features from different scales with directly adding the features. We evaluate on ADE20k semantic segmentation with DINOv2 and OpenCLIP. On both models, concatenating, as done by default in S^2-Wrapper, has consistently better performance than adding the features.

Table 10: **Ablation of how to merge features from different scales.** We compare concatenating features with adding features from different scales. Concatenating has consistently better performance.

Model	Scales	Merging	mIoU
DINOv2-Base-S^2	518^2, 1036^2, 1536^2	add	55.7
DINOv2-Base-S^2	518^2, 1036^2, 1536^2	concat	56.9
OpenCLIP-Base-S^2	256^2, 512^2, 1024^2	add	51.4
OpenCLIP-Base-S^2	256^2, 512^2, 1024^2	concat	52.5

G Throughput of Models with S^2

Previously we use FLOPs to measure the computational cost of different models. Since FLOPs is only a surrogate metric for the actual throughput of the models, here we compare the throughput of different models and verify if it aligns with FLOPs. Table 11 shows the results. We report the FLOPs and throughput of DINOv2 model with base, large, and giant size, as well as base size with scales of $(1\times)$, $(1\times, 2\times)$, and $(1\times, 2\times, 3\times)$. We test on base scales of 224^2 and 518^2. We can see that in general, the throughput follows the similar trends as FLOPs. For example, the base model with scales of (224^2, 448^2, 672^2) has the similar throughput as the giant model with scale of (224^2). The base model with scales of (224^2, 448^2) has the about $0.8\times$ throughput as the large model with scale of (224^2). On base scale of 518^2, the multi-scale base models with scales of $(1\times, 2\times)$, and $(1\times, 2\times, 3\times)$ have about $0.7\times$ throughput as the large and giant models, respectively.

H Additional Qualitative Results on V^*

We show more qualitative results on the V^* benchmark. We compare the performances of LLaVA-1.5 with S^2 scaling, original LLaVA-1.5 [19], and GPT-4V [1] on several examples in visual detail understanding (V^* [42]). Similarly, for LLaVa-1.5 with S^2 scaling, we use Vicuna-7B [7] as LLM and OpenAI CLIP as the vision backbone and apply S^2 scaling on the vision backbone.

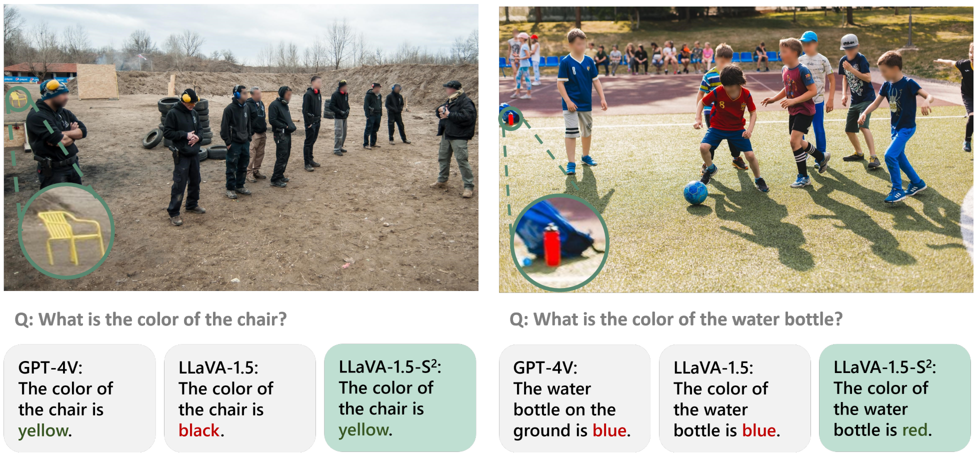
Table 11: Comparison of FLOPs and Throughput.

Model Size	Scales	#FLOPs	Throughput (image/s)
Base	(224^2)	17.6G	138.5
Base	$(224^2, 448^2)$	88.1G	39.5
Base	$(224^2, 448^2, 672^2)$	246.0G	16.5
Large	(224^2)	61.6G	54.5
Giant	(224^2)	204.9G	17.2
Base	(518^2)	134.4G	34.9
Base	$(518^2, 1036^2)$	671.8G	7.7
Base	$(518^2, 1036^2, 1554^2)$	1881G	2.7
Large	(518^2)	460.9G	11.8
Giant	(518^2)	1553G	3.8

In Figure 7, we see various examples that demonstrate the capabilities of different MLLMs. For instance, in example (f), the query is about the color of the flowers, which only occupy around 670 pixels in the 2550×1500 image. Here, LLaVA-1.5-S^2 correctly identifies the color as 'white'. However, LLaVa-1.5 fails to capture the correct color and recognizes it as 'red', which is actually the color of the flowerpot. On the other hand, GPT-4V recognizes the color as 'a mix of red and white', indicating that it cannot distinguish the subtle differences between the flowerpot and flowers.

In another example (c), the query is about the color of the woman's shirt. Here, the size of the woman's figure is small, and the purple color of the shirt is very similar to the dark background color. In this case, LLaVA-1.5-S^2 correctly identifies the color of the shirt as 'purple', while both LLaVA-1.5 and GPT-4V mistakenly identify the color of the shirt as 'black' or 'blue', which is the color of the background.

The above examples highlight the difference in performance between LLaVA-1.5-S^2, LLaVA-1.5 and GPT-4V. LLaVA-1.5-S^2 distinguishes itself through its heightened sensitivity and enhanced precision in visual detail understanding. This advanced level of detail recognition can be attributed to the S^2 scaling applied to its vision backbone, which significantly augments its ability to analyze and interpret subtle visual cues within complex images.



Q: What is the color of the chair?

GPT-4V: The color of the chair is yellow.

LLaVA-1.5: The color of the chair is black.

LLaVA-1.5-S^2: The color of the chair is yellow.

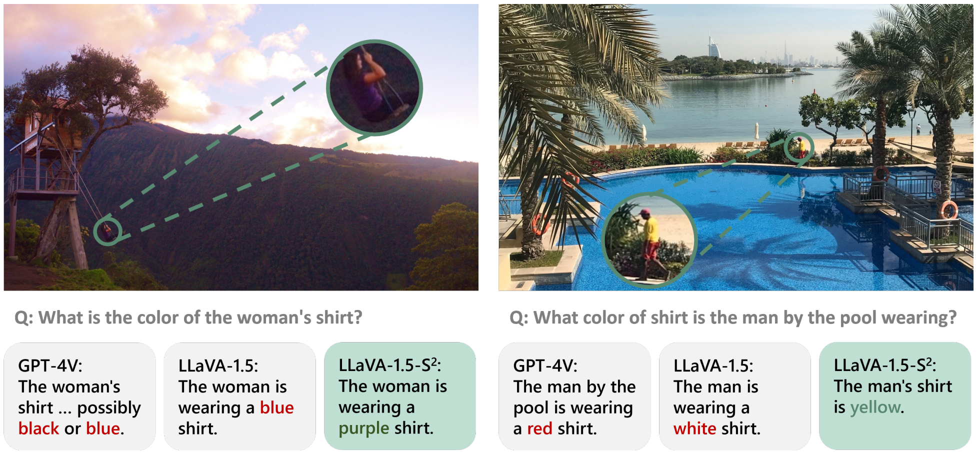
Q: What is the color of the water bottle?

GPT-4V: The water bottle on the ground is blue.

LLaVA-1.5: The color of the water bottle is blue.

LLaVA-1.5-S^2: The color of the water bottle is red.

(a) What is the color of the chair? (b) What is the color of the water bottle?



Q: What is the color of the woman's shirt?

GPT-4V: The woman's shirt ... possibly black or blue.

LLaVA-1.5: The woman is wearing a blue shirt.

LLaVA-1.5-S^2: The woman is wearing a purple shirt.

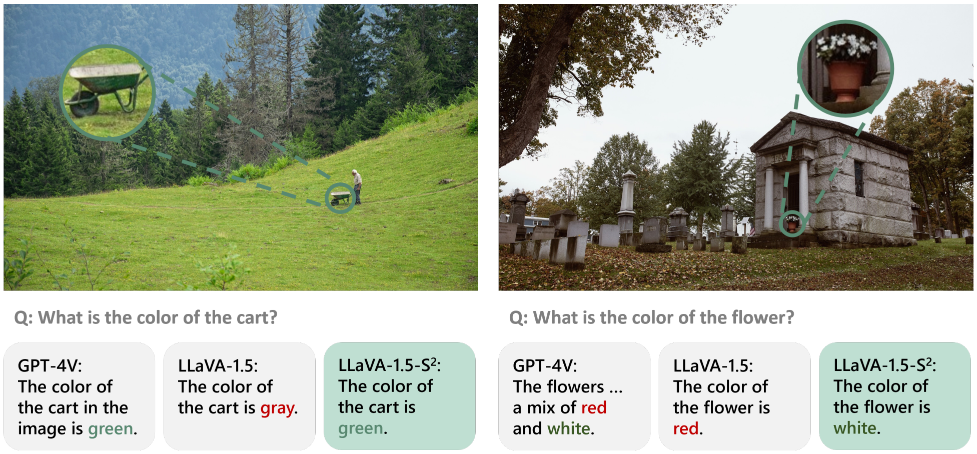
Q: What color of shirt is the man by the pool wearing?

GPT-4V: The man by the pool is wearing a red shirt.

LLaVA-1.5: The man is wearing a white shirt.

LLaVA-1.5-S^2: The man's shirt is yellow.

(c) What is the color of the woman's shirt? (d) What color of shirt is the man by the pool wearing?



Q: What is the color of the cart?

GPT-4V: The color of the cart in the image is green.

LLaVA-1.5: The color of the cart is gray.

LLaVA-1.5-S^2: The color of the cart is green.

Q: What is the color of the flower?

GPT-4V: The flowers ... a mix of red and white.

LLaVA-1.5: The color of the flower is red.

LLaVA-1.5-S^2: The color of the flower is white.

(e) What is the color of the cart? (f) What is the color of the flower?

Figure 7: **Examples of LLaVA-1.5 with S^2 scaling on the V* benchmark,** demonstrating its extreme ability in recognizing fine-grained details of an image.