Supplementary Material: When Do We Not Need Larger Vision Models?

Baifeng Shi[1], Ziyang Wu[1], Maolin Mao[1], Xin Wang[2], and Trevor Darrell[1]

[1] UC Berkeley
[2] Microsoft Research

1 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 1, 2, and 3, respectively. We use ImageNet-21k pre-trained checkpoints for ViT[3,4,5], LVD-142M pre-trained checkpoints for DINOv2[6,7,8], and LAION-2B pre-trained checkpoints for OpenCLIP[9,10,11]. For each model type (ViT [6], DINOv2 [9], OpenCLIP [3]), 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 [2] following the configurations here[12]. For depth estimation, we train a VPD depth decoder [16] following the configurations here[13].

Table 4 and 5 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[14]. For evaluations on certain MLLM benchmarks such as MMMU [15], since it is not supported in the LLaVA-1.5 repo, we use VLMEvalKit [5] for evaluation[15].

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

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

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

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

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

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

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

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

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

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

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

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

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

Table 6 shows the model details and full results for the robotic manipulation task of cube picking. We use MVP [10] 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 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 1: 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

2 Derivation of Mutual Information

Denote the features from two models by $\mathbf{x} \in \mathbb{R}^{d_x}$ and $\mathbf{y} \in \mathbb{R}^{d_y}$ 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.*, $p(\mathbf{y}) \quad \mathcal{N}(\hat{}, \sigma^2 \mathbf{I})$ and $p(\mathbf{y}|\mathbf{x}) \quad \mathcal{N}(\hat{f}(\mathbf{x}), \sigma'^2 \mathbf{I})$, where $f(\,)$ 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 $I(\mathbf{x}; \mathbf{y}) = h(\mathbf{y}) \quad h(\mathbf{y}|\mathbf{x}) = d_y \log \sigma \quad d_y \log \sigma'$.

When reconstructing the features $\mathbf{y}$ from another model's features $\mathbf{x}$, the optimal MSE loss would be $l = \min_f \frac{1}{d_y} E||\mathbf{y} \quad f(\mathbf{x})||_2^2 = \frac{1}{d_y} E||\mathbf{y} \quad \hat{f}(\mathbf{x})||_2^2 = \sigma'^2$.

Table 2: 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

The optimal MSE loss of reconstructing y from a dummy constant vector would be $l_0 = \min_{\boldsymbol{\mu}} \frac{1}{d_y} E\|\mathsf{y} \quad \boldsymbol{\mu}\|_2^2 = \frac{1}{d_y} E\|\mathsf{y} \quad \hat{}\boldsymbol{\mu}\|_2^2 = \sigma^2$. Then we get the mutual information between x and y is $I(\mathsf{x};\mathsf{y}) = d_y \log \sigma \quad d_y \log \sigma' = \quad \frac{d_y}{2} \log \frac{\sigma^2}{\quad} \; / \; \log \frac{l}{l_0}$.

3 Results on ConvNeXt

To see if convolutional networks have similar behaviors as transformer-based models, we test ConvNeXt [8] models (per-trained on ImageNet-21k[16,17,18]) on three tasks: image classification, semantic segmentation, and depth estimation. We use ImageNet [11], ADE20k [17], and NYUv2 [12] datasets for each task. Similarly, we freeze the backbone and only train the task-specific head for all experiments, using a single linear layer, UPerNet [14], and VPD depth decoder [16] as the decoder heads for three tasks, respectively. For model size scaling, we test the base, large, and xlarge size performance of ConvNeXt [8] 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 1. 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 parameters. In the semantic

[16] https://dl.fbaipublicfiles.com/convnext/convnext_base_22k_224.pth
[17] https://dl.fbaipublicfiles.com/convnext/convnext_large_22k_224.pth
[18] https://dl.fbaipublicfiles.com/convnext/convnext_xlarge_22k_224.pth

Table 3: 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 4: 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

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 base and large model are not pre-trained with S^2 (see Section 4.3 of the paper).

4 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

Table 5: 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

Table 6: 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

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 7. 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 in OOM error when using (1x, 2x, 3x) scales.

Table 7: **Ablation of splitting large-scale images.** We compare splitting the large-scale image into regular-size 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

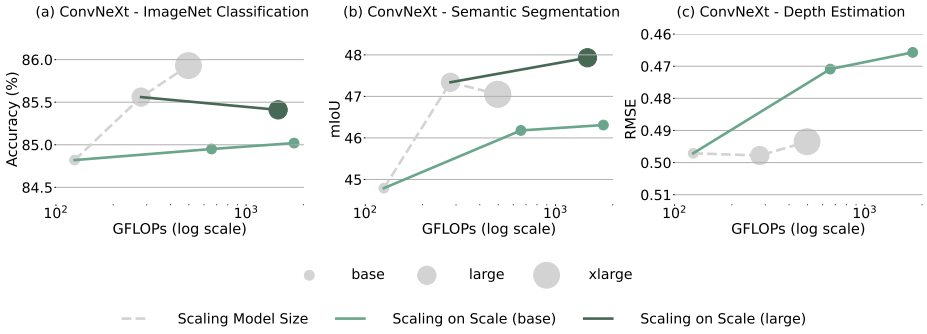

Fig. 1: Comparison of S^2 scaling and model size scaling on ConvNeXt. We evaluate three tasks: ImageNet classification, semantic segmentation, and depth estimation. For S^2 scaling (plotted in green curve), we test three sets of scales from single-scale (1x) to multi-scale (up to 3x), and we adjust each 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).

Results for (ii) are shown in Table 8. 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 8: 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

5 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 9 shows the results. We report the FLOPs and throughput put of DINOv2 model with base, large, and giant size, as well as base size with scales

of (1), (1 ,2), and (1 ,2 ,3). 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 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 ,2), and (1 ,2 ,3) have about 0.7 throughput as the large and giant models, respectively.

Table 9: 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

6 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 [7], and GPT-4V [1] on several examples in visual detail understanding (V* [13]). Similarly, for LLaVa-1.5 with S^2 scaling, we use Vicuna-7B [4] as LLM and OpenAI CLIP as the vision backbone and apply S^2 scaling on the vision backbone.

In Figure 2, 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.

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
2. Cheng, B., Misra, I., Schwing, A.G., Kirillov, A., Girdhar, R.: Masked-attention mask transformer for universal image segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1290–1299 (2022)
3. Cherti, M., Beaumont, R., Wightman, R., Wortsman, M., Ilharco, G., Gordon, C., Schuhmann, C., Schmidt, L., Jitsev, J.: Reproducible scaling laws for contrastive language-image learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2818–2829 (2023)
4. Chiang, W.L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J.E., et al.: Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. See https://vicuna. lmsys. org (accessed 14 April 2023) (2023)
5. Contributors, O.: Opencompass: A universal evaluation platform for foundation models. `https://github.com/open-compass/opencompass` (2023)
6. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
7. Liu, H., Li, C., Li, Y., Lee, Y.J.: Improved baselines with visual instruction tuning. arXiv preprint arXiv:2310.03744 (2023)
8. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11976–11986 (2022)
9. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al.: Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193 (2023)
10. Radosavovic, I., Xiao, T., James, S., Abbeel, P., Malik, J., Darrell, T.: Real-world robot learning with masked visual pre-training. In: Conference on Robot Learning. pp. 416–426. PMLR (2023)
11. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision 115, 211–252 (2015)
12. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: 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. pp. 746–760. Springer (2012)
13. Wu, P., Xie, S.: V*: Guided visual search as a core mechanism in multimodal llms. arXiv preprint arXiv:2312.14135 (2023)
14. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 418–434 (2018)
15. Yue, X., Ni, Y., Zhang, K., Zheng, T., Liu, R., Zhang, G., Stevens, S., Jiang, D., Ren, W., Sun, Y., et al.: Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. arXiv preprint arXiv:2311.16502 (2023)
16. Zhao, W., Rao, Y., Liu, Z., Liu, B., Zhou, J., Lu, J.: Unleashing text-to-image diffusion models for visual perception. arXiv preprint arXiv:2303.02153 (2023)
17. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 633–641 (2017)