Self-supervised learning overview

BYOL | DINOv1 | iBOT | DINOv2

Denis Shepelev, Jul 2023
https://github.com/denzist/presentations

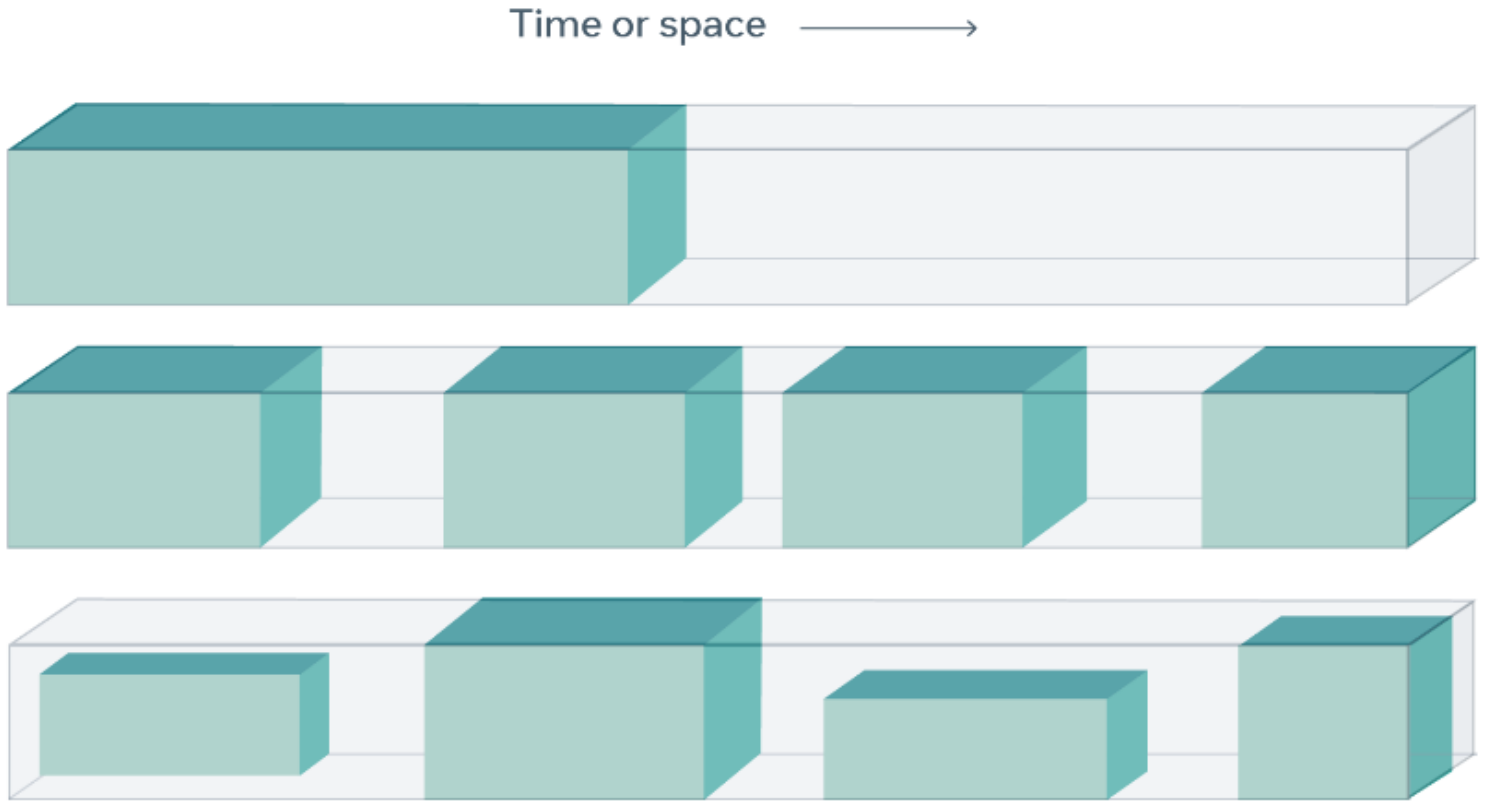
Plan

- Self supervised learning (SSL)
- BYOL
- DINO
- iBOT
- DINOv2

SSL

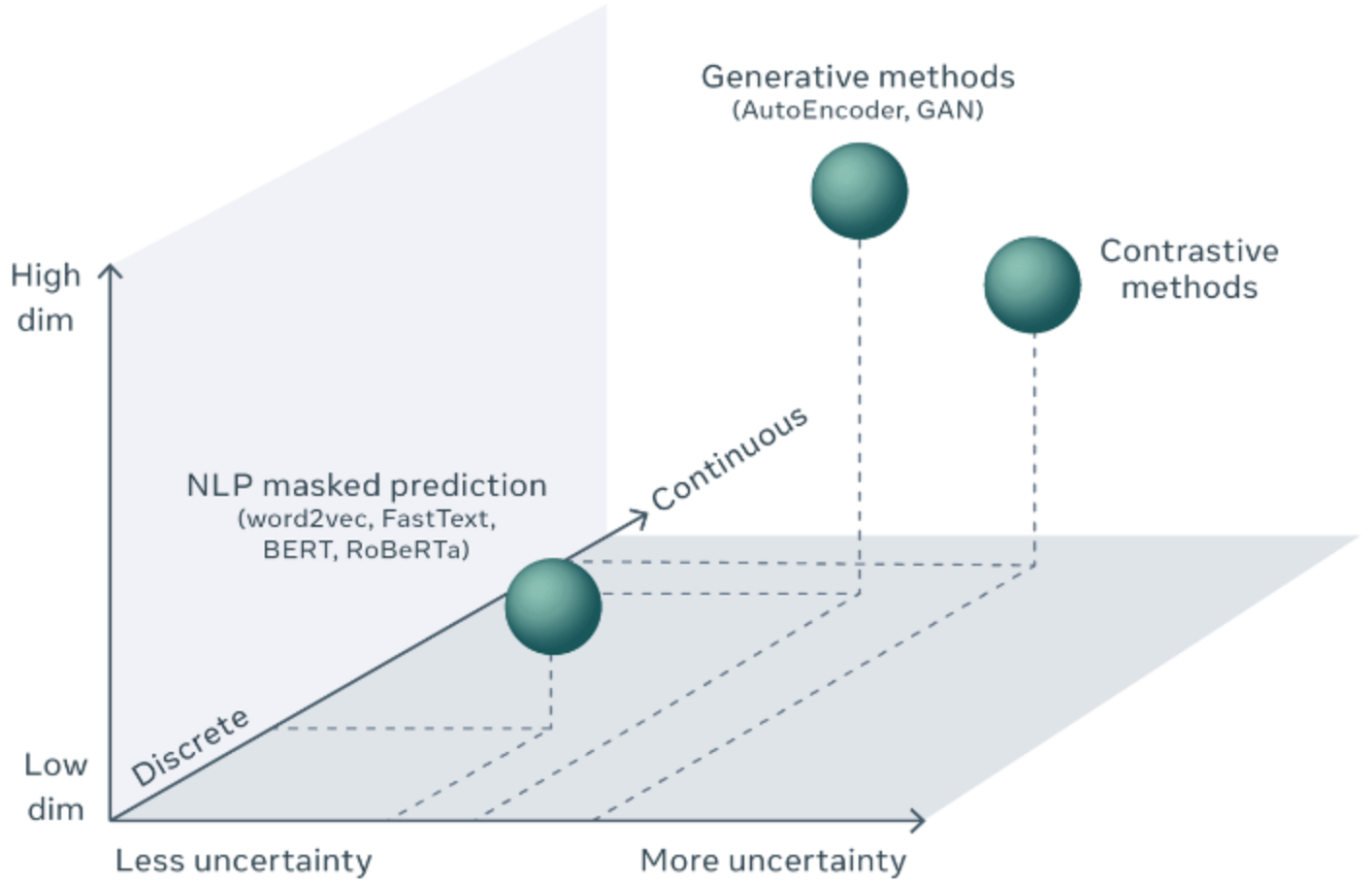
SSL: The dark matter of intelligence

- SSL leverages the underlying structure of the data **without relying on human labels**, obtaining supervisory signals from the data itself
- **The aim of SSL is to obtain generalized features** that can be used for other various downstream tasks



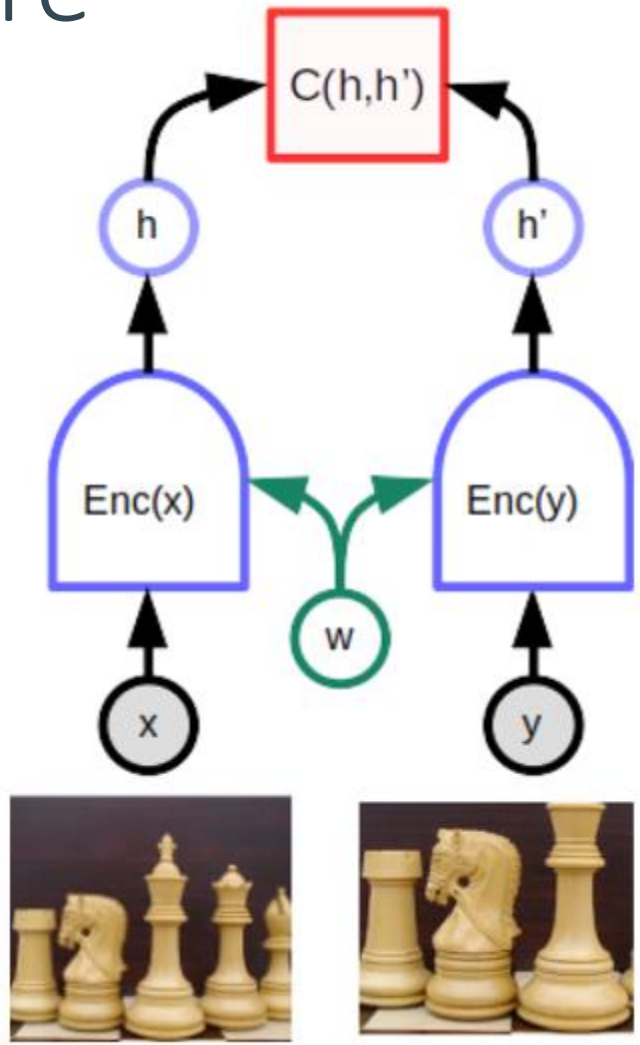
Uncertainty in prediction: LM vs VM

- These methods have a particularly profound impact on language models (LM), however, it is difficult to use them in vision models (VM)



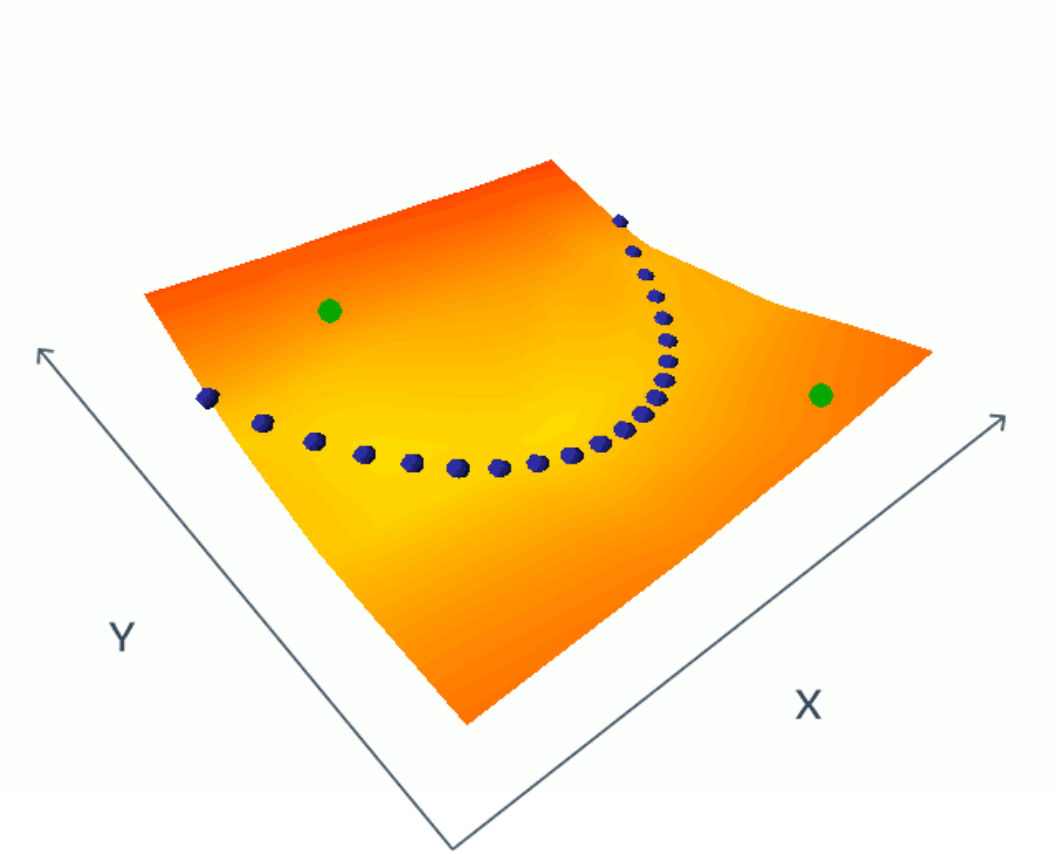
Joint embedding architecture

- The function C at the top produces a scalar loss that measures the distance between the representation vectors (embeddings) produced by two identical twin networks sharing the same parameters (w)
- When x and y are slightly different versions of the **same image**, the system is trained to produce **a low loss**, which forces the model to produce similar embedding vectors for the two images
- The difficult part is to train the model so that it produces **high loss** (i.e., different embeddings) **for images that are different**
- **Without negative samples these framework may return constant embedding – trivial solution**



Contrastive learning problem

- Contrastive methods simultaneously **push down on the loss of compatible (x,y) pairs**, indicated by the blue dots, and **push up on the loss of well chosen (x,y) pairs that are incompatible**, symbolized by the green dots
- In this simple example, x and y are both scalars, but in real situations, x and y could be an image or a video with millions of dimensions
- Producing incompatible pairs that will shape the energy in suitable ways is **challenging and expensive computationally**
- **Current SotA SSL develops negative-free approaches**



BYOL

Bootstrap Your Own Latent A New Approach to Self-Supervised Learning

(NIPS 2020, DeepMind)

BYOL

- BYOL achieves higher performance than SotA contrastive methods **without using negative pairs**
- Starting from an **augmented view** of an image, BYOL trains its **online network to predict the target network's representation** of another **augmented view** of the same image
- Solution does not collapse because:
  - the **addition of a predictor** to the online network
  - target network –a **slow-moving average** of the online parameters

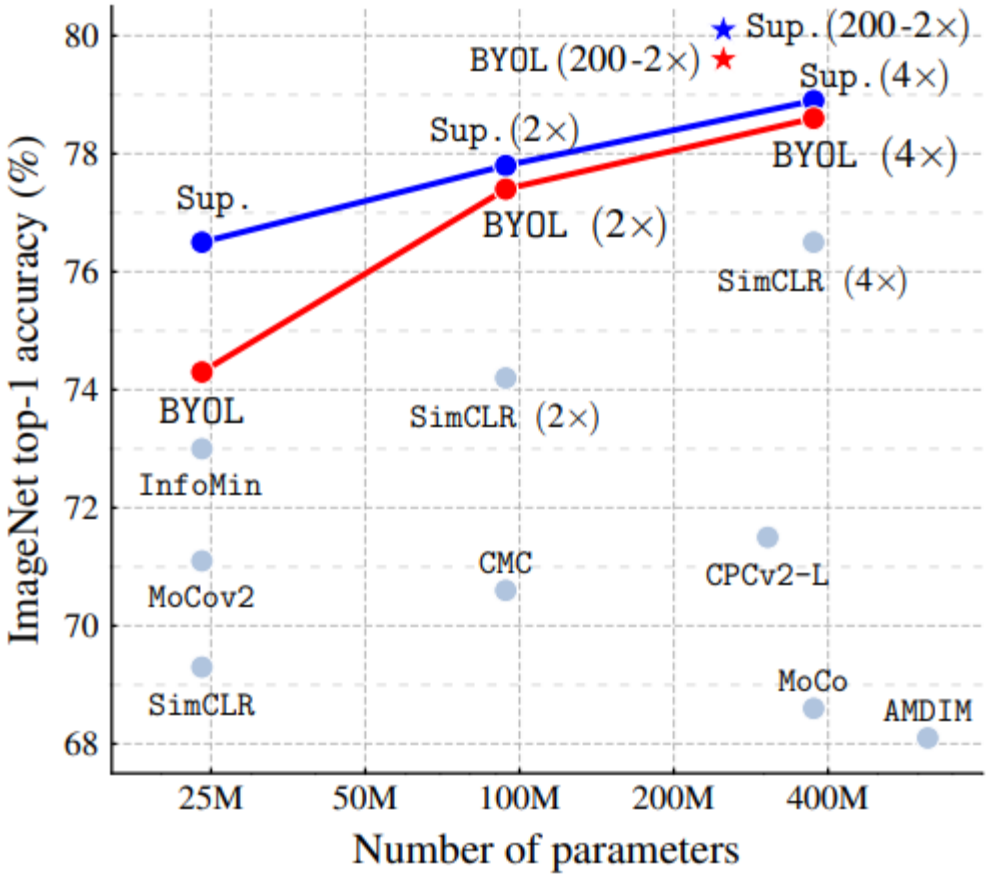


Figure 1: Performance of BYOL on ImageNet (linear evaluation) using ResNet-50 and our best architecture ResNet-200 (2×), compared to other unsupervised and supervised (Sup.) baselines [8].

Framework

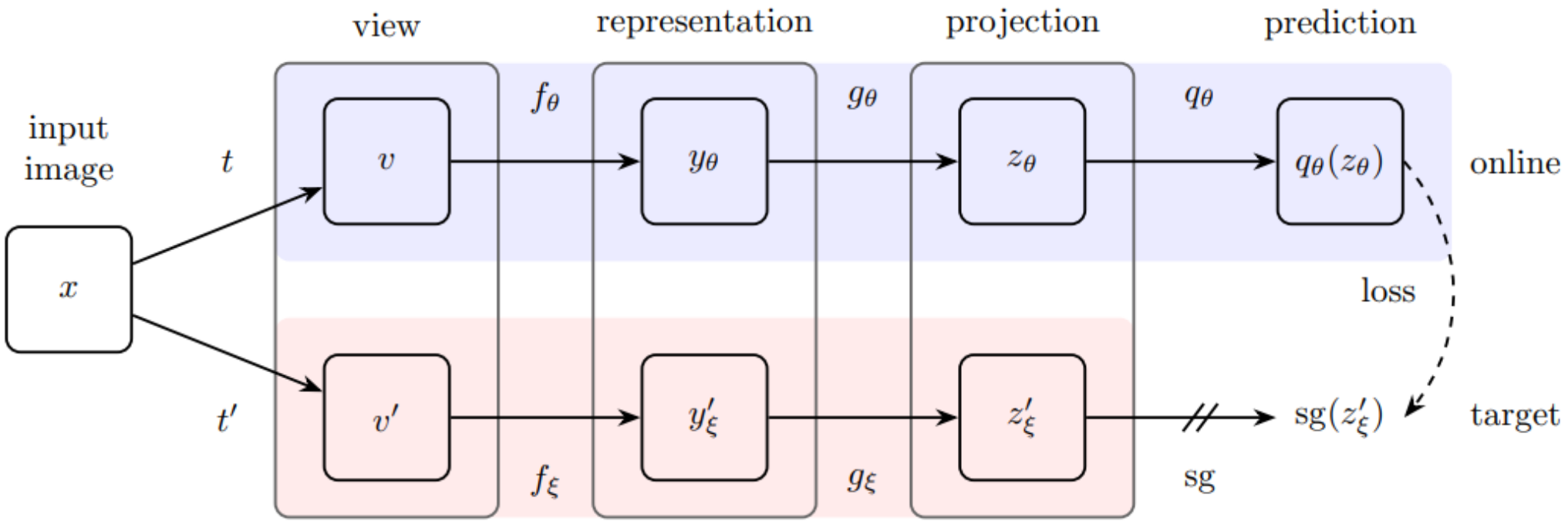


Figure 2: BYOL's architecture. BYOL minimizes a similarity loss between $q_\theta(z_\theta)$ and $\mathrm{sg}(z'_\xi)$, where θ are the trained weights, ξ are an exponential moving average of θ and sg means stop-gradient. At the end of training, everything but f_θ is discarded, and y_θ is used as the image representation.

Minimization problem

$$\theta \leftarrow \text{optimizer}\left(\theta, \nabla_\theta \mathcal{L}^{\texttt{BYOL}}_{\theta,\xi}, \eta\right)$$
$$\xi \leftarrow \tau\xi + (1-\tau)\theta,$$

$$\mathcal{L}^{\texttt{BYOL}}_{\theta,\xi} = \mathcal{L}_{\theta,\xi} + \widetilde{\mathcal{L}}_{\theta,\xi}$$

$$\mathcal{L}_{\theta,\xi} \triangleq \left\|\overline{q_\theta}(z_\theta) - \overline{z}'_\xi\right\|_2^2 = 2 - 2 \cdot \frac{\langle q_\theta(z_\theta), z'_\xi \rangle}{\left\|q_\theta(z_\theta)\right\|_2 \cdot \left\|z'_\xi\right\|_2}$$

Augmentation

During self-supervised training, BYOL uses the following image augmentations (which are a subset of the ones presented in [8]):

- random cropping: a random patch of the image is selected, with an area uniformly sampled between 8% and 100% of that of the original image, and an aspect ratio logarithmically sampled between $3/4$ and $4/3$. This patch is then resized to the target size of 224×224 using bicubic interpolation;
- optional left-right flip;
- color jittering: the brightness, contrast, saturation and hue of the image are shifted by a uniformly random offset applied on all the pixels of the same image. The order in which these shifts are performed is randomly selected for each patch;
- color dropping: an optional conversion to grayscale. When applied, output intensity for a pixel (r, g, b) corresponds to its luma component, computed as $0.2989r + 0.5870g + 0.1140b$;
- Gaussian blurring: for a 224×224 image, a square Gaussian kernel of size 23×23 is used, with a standard deviation uniformly sampled over $[0.1, 2.0]$;
- solarization: an optional color transformation $x \mapsto x \cdot \mathbf{1}_{\{x<0.5\}} + (1-x) \cdot \mathbf{1}_{\{x \geq 0.5\}}$ for pixels with values in $[0, 1]$.

Ablations: predictor and target network

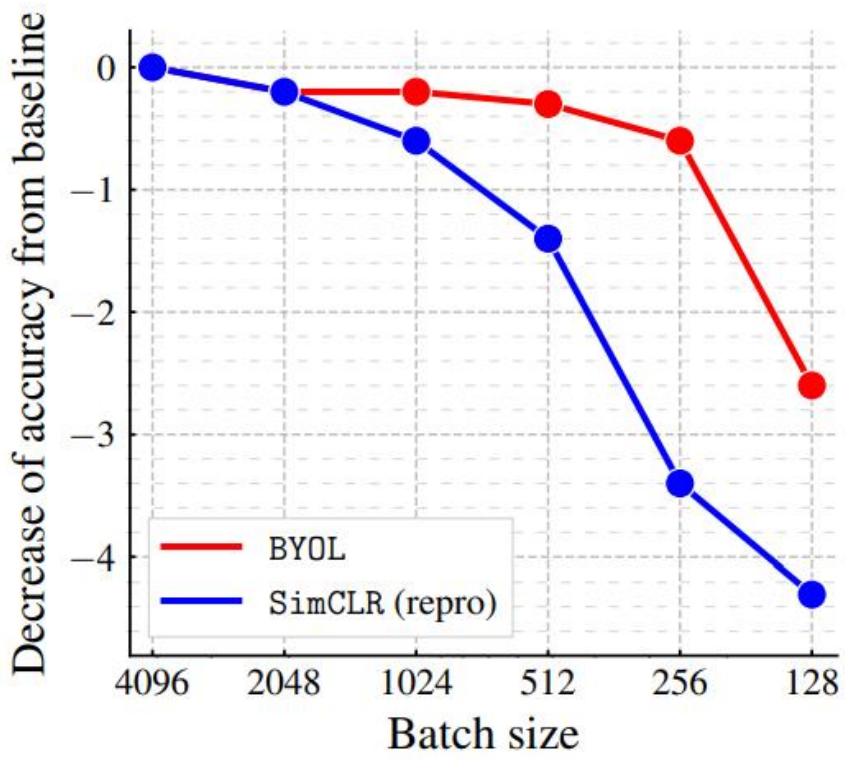
$$\text{InfoNCE}_\theta^{\alpha,\beta} \triangleq \frac{2}{B}\sum_{i=1}^{B} S_\theta(v_i, v_i') - \frac{2\alpha\cdot\beta}{B}\sum_{i=1}^{B}\ln\left(\sum_{j\neq i}\exp\frac{S_\theta(v_i,v_j)}{\alpha} + \sum_j \exp\frac{S_\theta(v_i,v_j')}{\alpha}\right),$$

$$S_\theta(u_1,u_2) \triangleq \frac{\langle\phi(u_1),\psi(u_2)\rangle}{\|\phi(u_1)\|_2\cdot\|\psi(u_2)\|_2}.$$

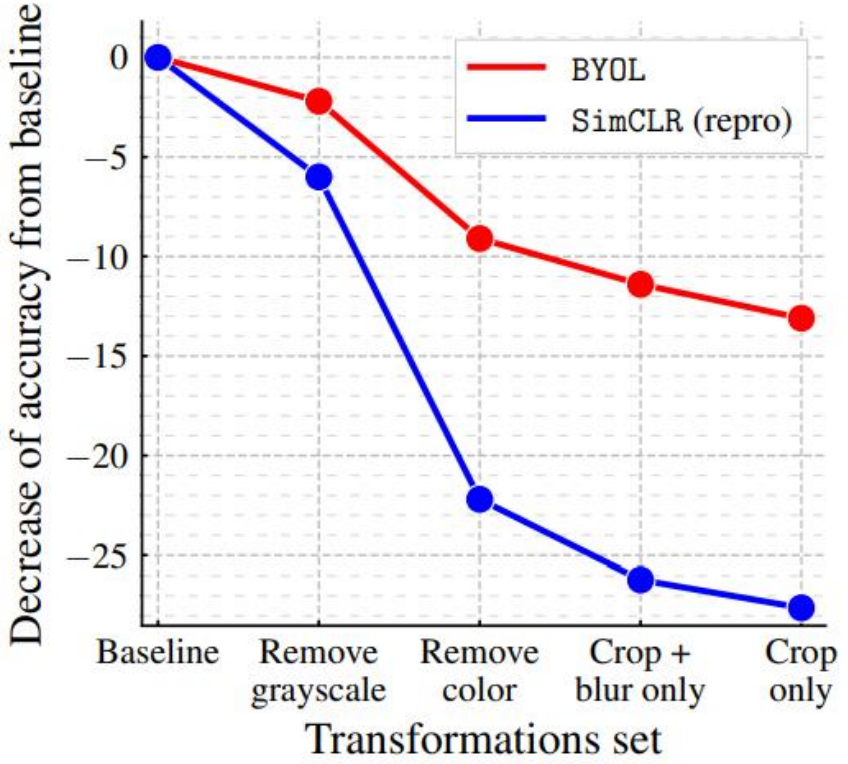
- Network learns embeddings **without negative examples only with predictor layers and slow-moving average** of parameters (1 row)

Method	Predictor	Target parameters	β	Top-1
BYOL	✓	ξ	0	72.5
	✓	ξ	1	70.9
		ξ	1	70.7
	✓	$\text{sg}(\theta)$	1	70.2
SimCLR		θ	1	69.4
	✓	$\text{sg}(\theta)$	1	70.1
		$\text{sg}(\theta)$	1	69.2
	✓	θ	1	69.0
	✓	$\text{sg}(\theta)$	0	5.5
	✓	θ	0	0.3
		ξ	0	0.2
		$\text{sg}(\theta)$	0	0.1
		θ	0	0.1

Ablations: batch size and augmentations



(a) Impact of batch size



(b) Impact of progressively removing transformations

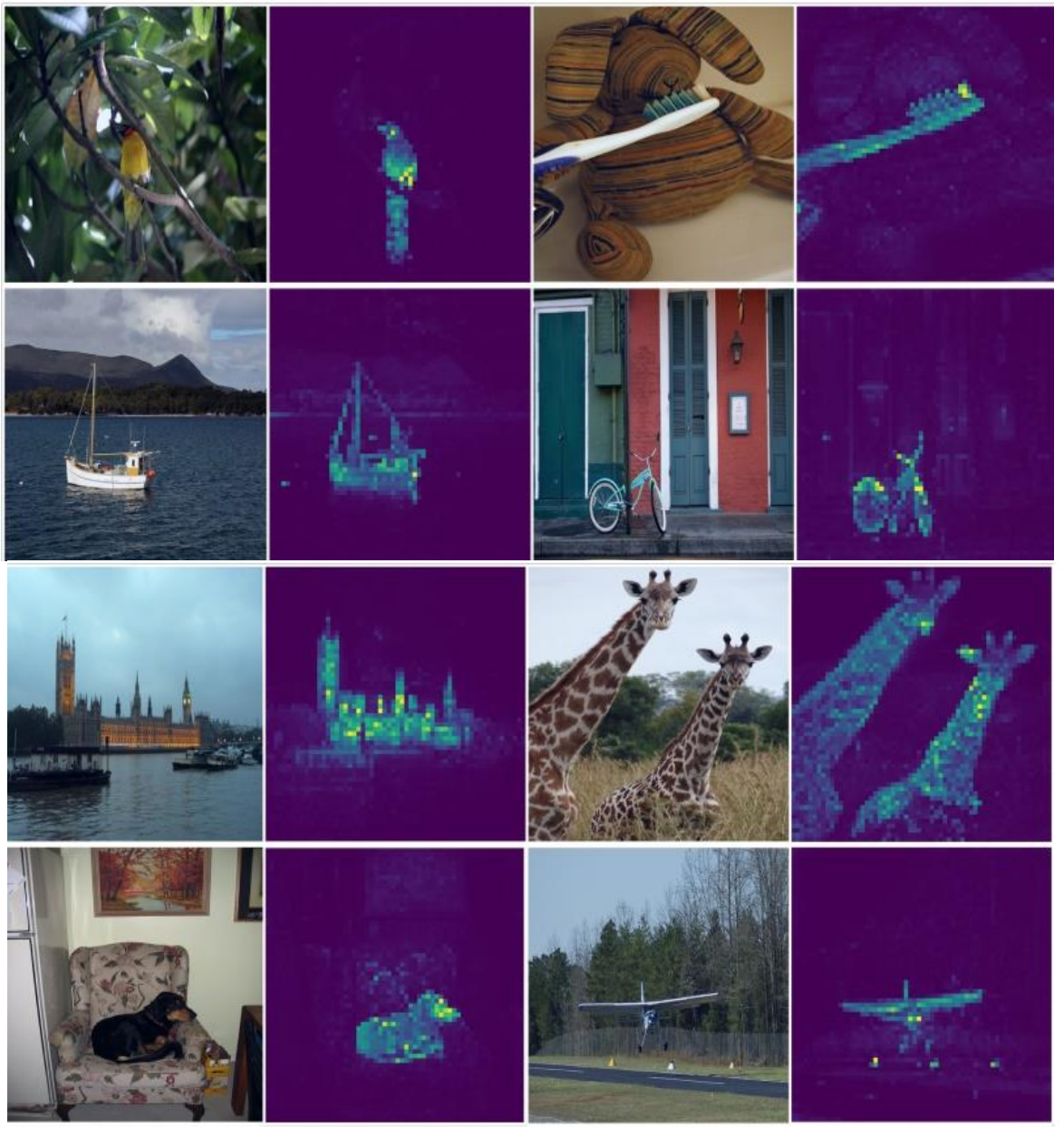
Figure 3: Decrease in top-1 accuracy (in % points) of BYOL and our own reproduction of SimCLR at 300 epochs, under linear evaluation on ImageNet.

DINO

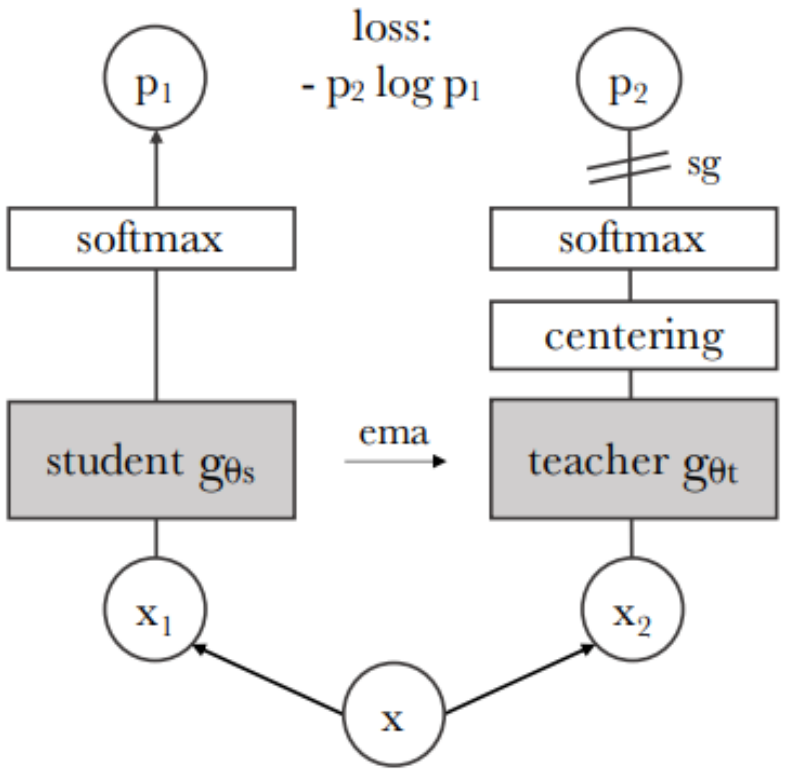
Emerging Properties in Self-Supervised Vision Transformers
(ICCV 2021, Facebook AI Research)

DINO

- DINO follows BYOL ideas and architecture setup, but **changes training objectives and augmentation** which provide better results
- DINO avoids collapsing by using **sharpening** (distillation) and batch **centering**, and do not use predictor layers
- Usage of **global-local augmentation** in target (teacher) and online (student) networks improves metrics
- Their ViT features perform particularly well with a basic nearest neighbors classifier (k-NN) without any finetuning, linear classifier nor data augmentation, achieving 78.3% top-1 accuracy on ImageNet



Framework



Algorithm 1 DINO PyTorch pseudocode w/o multi-crop.

```
# gs, gt: student and teacher networks
# C: center (K)
# tps, tpt: student and teacher temperatures
# l, m: network and center momentum rates
gt.params = gs.params
for x in loader: # load a minibatch x with n samples
    x1, x2 = augment(x), augment(x) # random views

    s1, s2 = gs(x1), gs(x2) # student output n-by-K
    t1, t2 = gt(x1), gt(x2) # teacher output n-by-K

    loss = H(t1, s2)/2 + H(t2, s1)/2
    loss.backward() # back-propagate

    # student, teacher and center updates
    update(gs) # SGD
    gt.params = l*gt.params + (1-l)*gs.params
    C = m*C + (1-m)*cat([t1, t2]).mean(dim=0)

def H(t, s):
    t = t.detach() # stop gradient
    s = softmax(s / tps, dim=1)
    t = softmax((t - C) / tpt, dim=1) # center + sharpen
    return - (t * log(s)).sum(dim=1).mean()
```

$$P_s(x)^{(i)} = \frac{\exp(g_{\theta_s}(x)^{(i)}/\tau_s)}{\sum_{k=1}^{K}\exp(g_{\theta_s}(x)^{(k)}/\tau_s)}, \qquad \min_{\theta_s} \sum_{x\in\{x_1^g,x_2^g\}} \sum_{\substack{x'\in V\\ x'\neq x}} H(P_t(x), P_s(x'))$$

Avoiding collapse

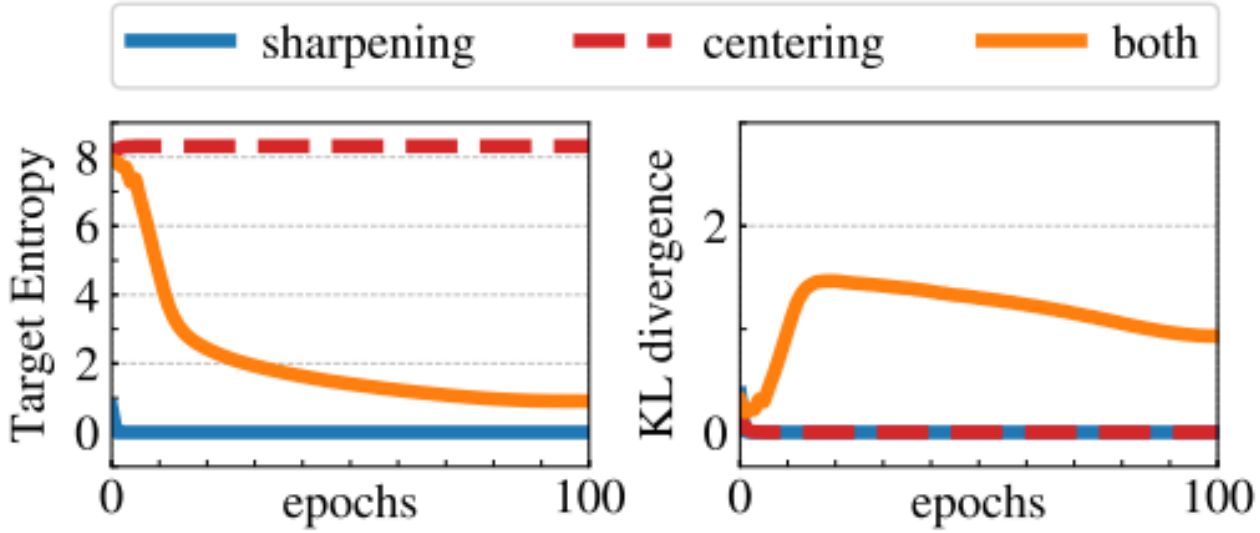


Figure 7: **Collapse study. (left)**: evolution of the teacher's target entropy along training epochs; **(right)**: evolution of KL divergence between teacher and student outputs.

$$H(P_t, P_s) = h(P_t) + D_{KL}(P_t|P_s).$$

```
s = softmax(s / tps, dim=1)
t = softmax((t - C) / tpt, dim=1)
```

- Sharpening **make one dimension dominance**

$$P_s(x)^{(i)} = \frac{\exp(g_{\theta_s}(x)^{(i)}/\tau_s)}{\sum_{k=1}^{K}\exp(g_{\theta_s}(x)^{(k)}/\tau_s)},$$

- Centering **prevents one dimension dominance**

$$c \leftarrow mc + (1-m)\frac{1}{B}\sum_{i=1}^{B} g_{\theta_t}(x_i).$$

Ablations

	Method	Mom.	SK	MC	Loss	Pred.	k-NN	Lin.
1	DINO	✓	✗	✓	CE	✗	72.8	76.1
2		✗	✗	✓	CE	✗	0.1	0.1
3		✓	✓	✓	CE	✗	72.2	76.0
4		✓	✗	✗	CE	✗	67.9	72.5
5		✓	✗	✓	MSE	✗	52.6	62.4
6		✓	✗	✓	CE	✓	71.8	75.6
7	BYOL	✓	✗	✗	MSE	✓	66.6	71.4
8	MoCov2	✓	✗	✗	INCE	✗	62.0	71.6
9	SwAV	✗	✓	✓	CE	✗	64.7	71.8

SK: Sinkhorn-Knopp, MC: Multi-Crop, Pred.: Predictor
CE: Cross-Entropy, MSE: Mean Square Error, INCE: InfoNCE

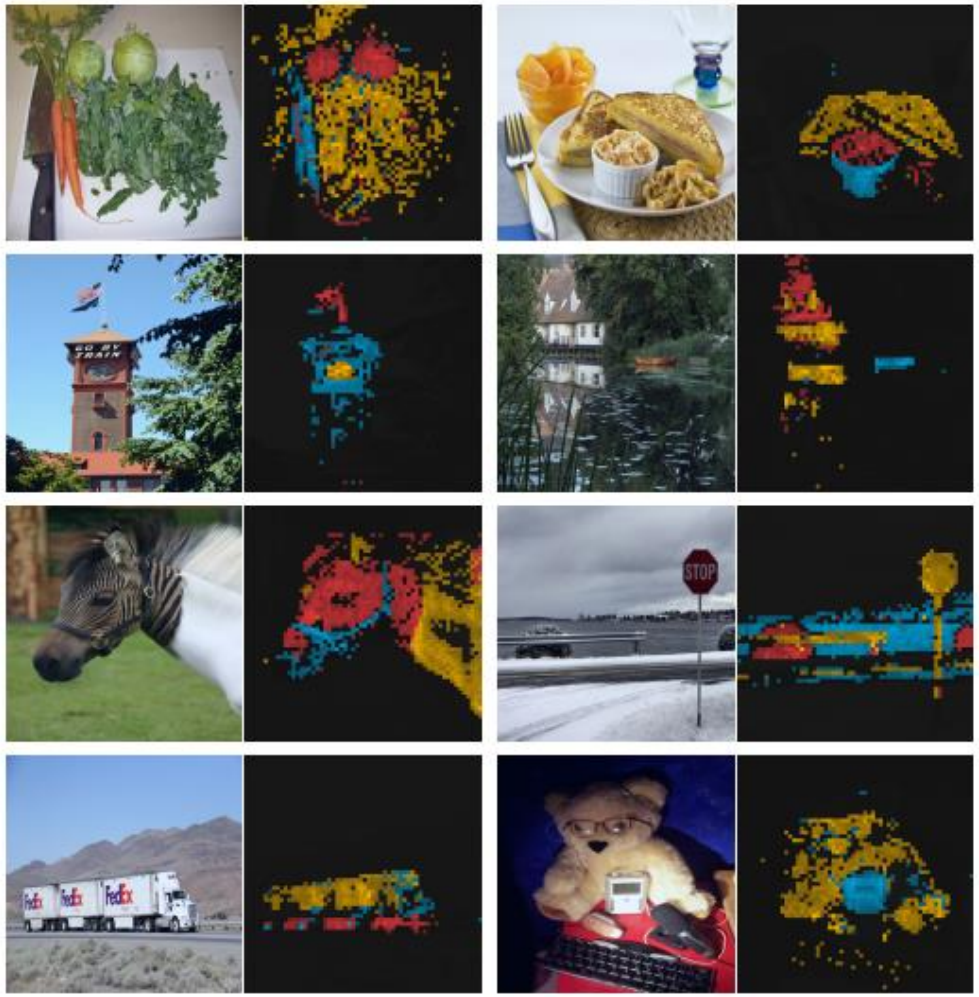
```
# x is n-by-K
# tau is Sinkhorn regularization param
x = exp(x / tau)
for _ in range(num_iters): # 1 iter of Sinkhorn
  # total weight per dimension (or cluster)
  c = sum(x, dim=0, keepdim=True)
  x /= c

  # total weight per sample
  n = sum(x, dim=1, keepdim=True)
  # x sums to 1 for each sample (assignment)
  x /= n
```

Table 7: **Important component for self-supervised ViT pre-training.** Models are trained for 300 epochs with ViT-S/16. We study the different components that matter for the k-NN and linear ("Lin.") evaluations. For the different variants, we highlight the differences from the default DINO setting. The best combination is the momentum encoder with the multicrop augmentation and the cross-entropy loss. We also report results with BYOL [30], MoCo-v2 [15] and SwAV [10].

Self-attention video visualization

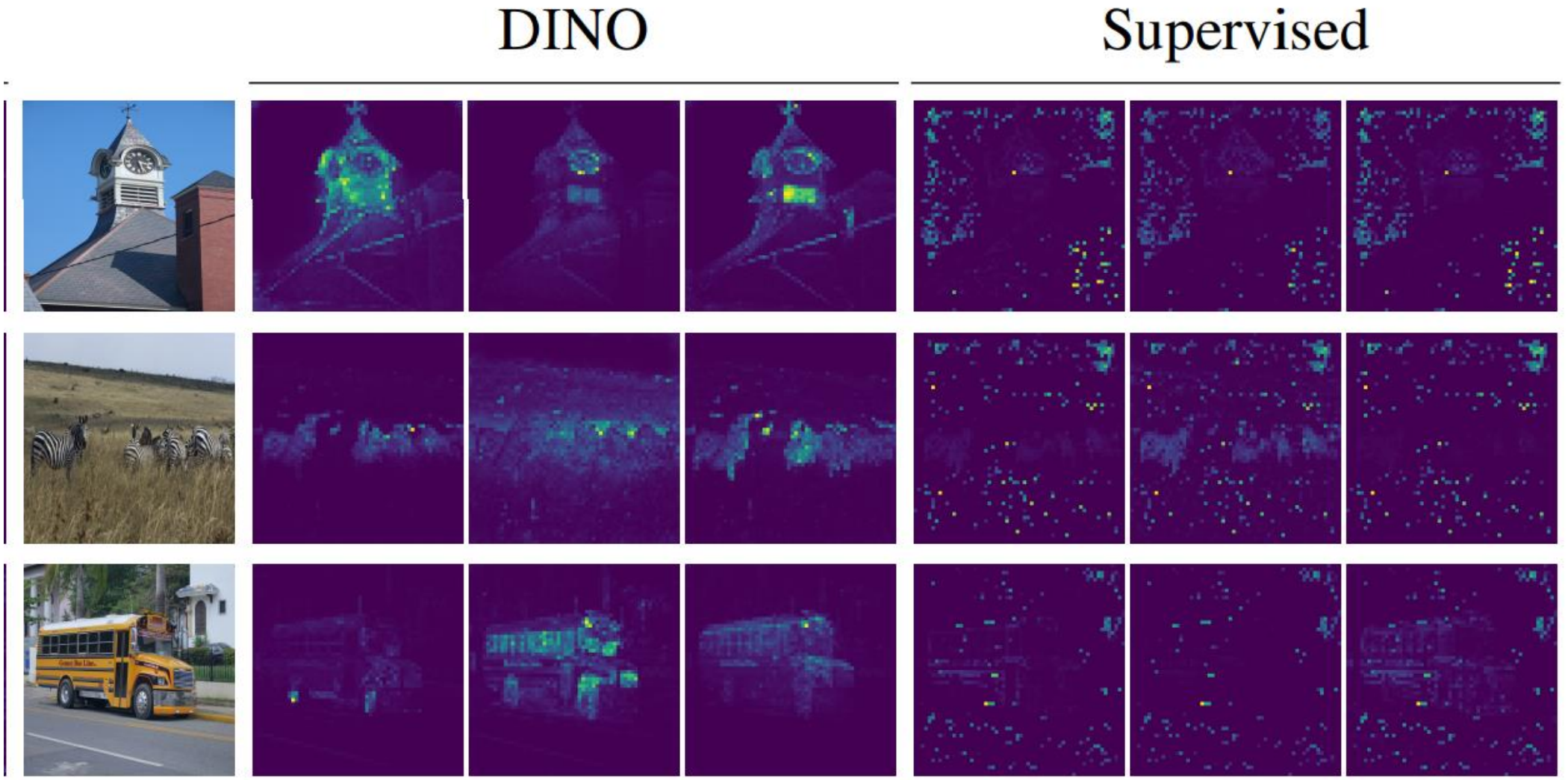
Comparison: segmentation on self-attention masks



	Random	Supervised	DINO
ViT-S/16	22.0	27.3	45.9
ViT-S/8	21.8	23.7	44.7

Figure 4: **Segmentations from supervised versus DINO.** We visualize masks obtained by thresholding the self-attention maps to keep 60% of the mass. On top, we show the resulting masks for a ViT-S/8 trained with supervision and DINO. We show the best head for both models. The table at the bottom compares the Jaccard similarity between the ground truth and these masks on the validation images of PASCAL VOC12 dataset.

Comparison: attention maps



Evaluation

ImageNet-1K classification

Method	Arch.	Param.	im/s	Linear	k-NN
Comparison across architectures					
SCLR [12]	RN50w4	375	117	76.8	69.3
SwAV [10]	RN50w2	93	384	77.3	67.3
BYOL [30]	RN50w2	93	384	77.4	–
DINO	ViT-B/16	85	312	78.2	76.1
SwAV [10]	RN50w5	586	76	78.5	67.1
BYOL [30]	RN50w4	375	117	78.6	–
BYOL [30]	RN200w2	250	123	79.6	73.9
DINO	ViT-S/8	21	180	79.7	**78.3**
SCLRv2 [13]	RN152w3+SK	794	46	79.8	73.1
DINO	ViT-B/8	85	63	**80.1**	77.4

Copy detection

Method	Arch.	Dim.	Resolution	mAP
Multigrain [5]	ResNet-50	2048	224^2	75.1
Multigrain [5]	ResNet-50	2048	largest side 800	82.5
Supervised [69]	ViT-B/16	1536	224^2	76.4
DINO	ViT-B/16	1536	224^2	81.7
DINO	ViT-B/8	1536	320^2	**85.5**

Image retrieval

			$\mathcal{R}$Ox		$\mathcal{R}$Par	
Pretrain	Arch.	Pretrain	M	H	M	H
Sup. [57]	RN101+R-MAC	ImNet	49.8	18.5	74.0	**52.1**
Sup.	ViT-S/16	ImNet	33.5	8.9	63.0	37.2
DINO	ResNet-50	ImNet	35.4	11.1	55.9	27.5
DINO	ViT-S/16	ImNet	41.8	13.7	63.1	34.4
DINO	ViT-S/16	GLDv2	**51.5**	**24.3**	**75.3**	51.6

Evaluation: low-shot learning

Method	Arch	Param.	Top 1 1%	Top 1 10%
Self-supervised pretraining with finetuning				
UDA [75]	RN50	23	–	68.1
SimCLRv2 [13]	RN50	23	57.9	68.4
BYOL [30]	RN50	23	53.2	68.8
SwAV [10]	RN50	23	53.9	70.2
SimCLRv2 [16]	RN50w4	375	63.0	74.4
BYOL [30]	RN200w2	250	71.2	77.7
Semi-supervised methods				
SimCLRv2+KD [13]	RN50	23	60.0	70.5
SwAV+CT [3]	RN50	23	–	70.8
FixMatch [64]	RN50	23	–	71.5
MPL [49]	RN50	23	–	73.9
SimCLRv2+KD [13]	RN152w3+SK	794	76.6	80.9
Frozen self-supervised features				
DINO -FROZEN	ViT-S/16	21	64.5	72.2

Table 12: **Low-shot learning on ImageNet with frozen ViT features.** We train a logistic regression on frozen features (FROZEN). Note that this FROZEN evaluation is performed *without any fine-tuning nor data augmentation*. We report top-1 accuracy. For reference, we show previously published results that uses finetuning and semi-supervised learning.

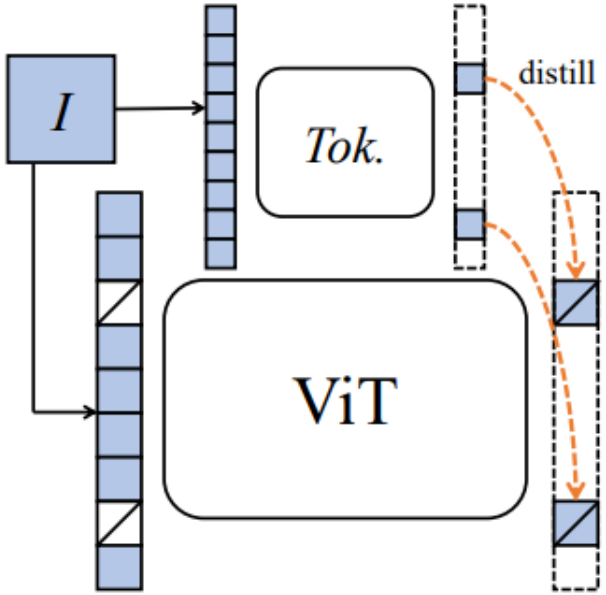
iBOT

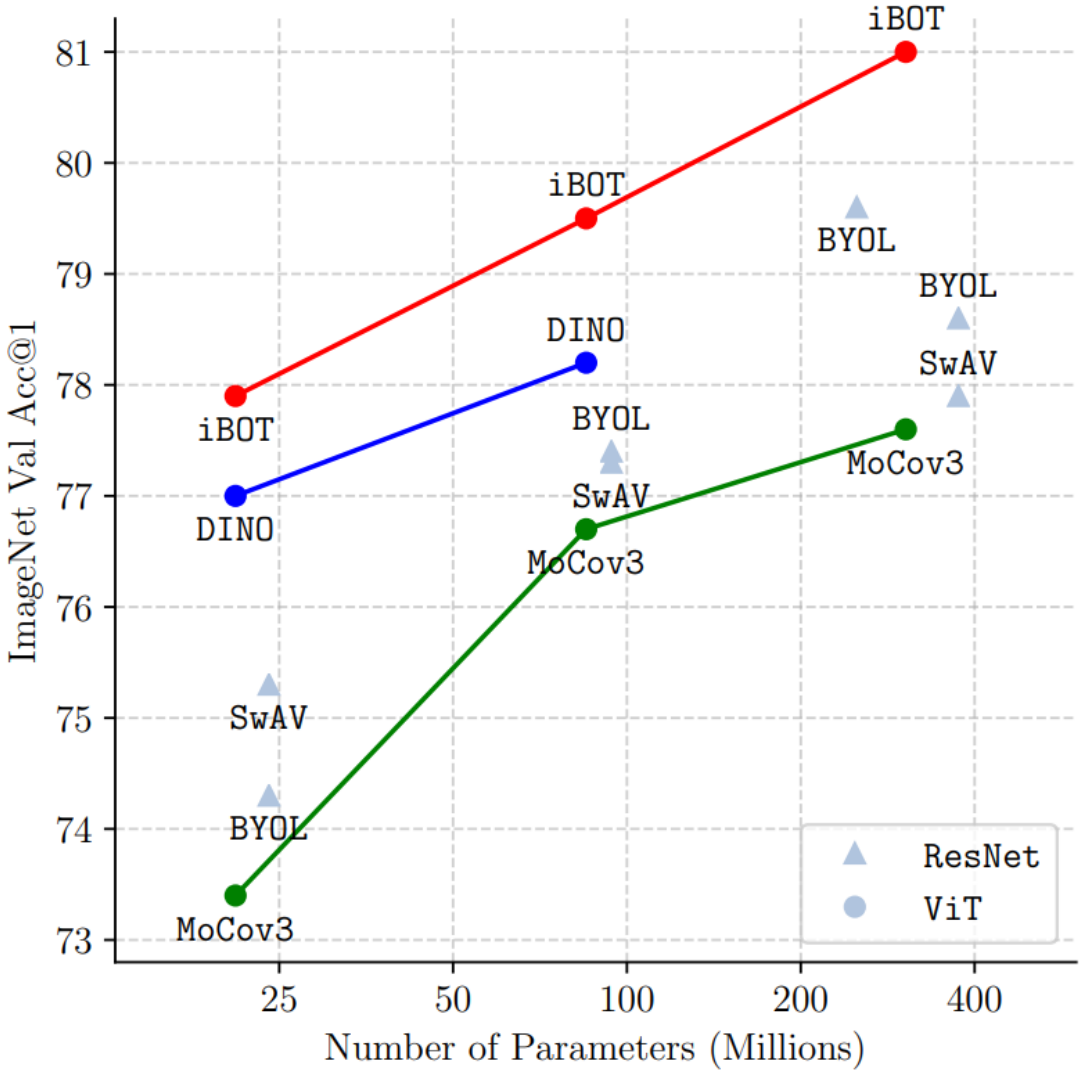
iBOT: Image BERT Pre-Training with Online Tokenizer

(ICLR 2022, ByteDance)

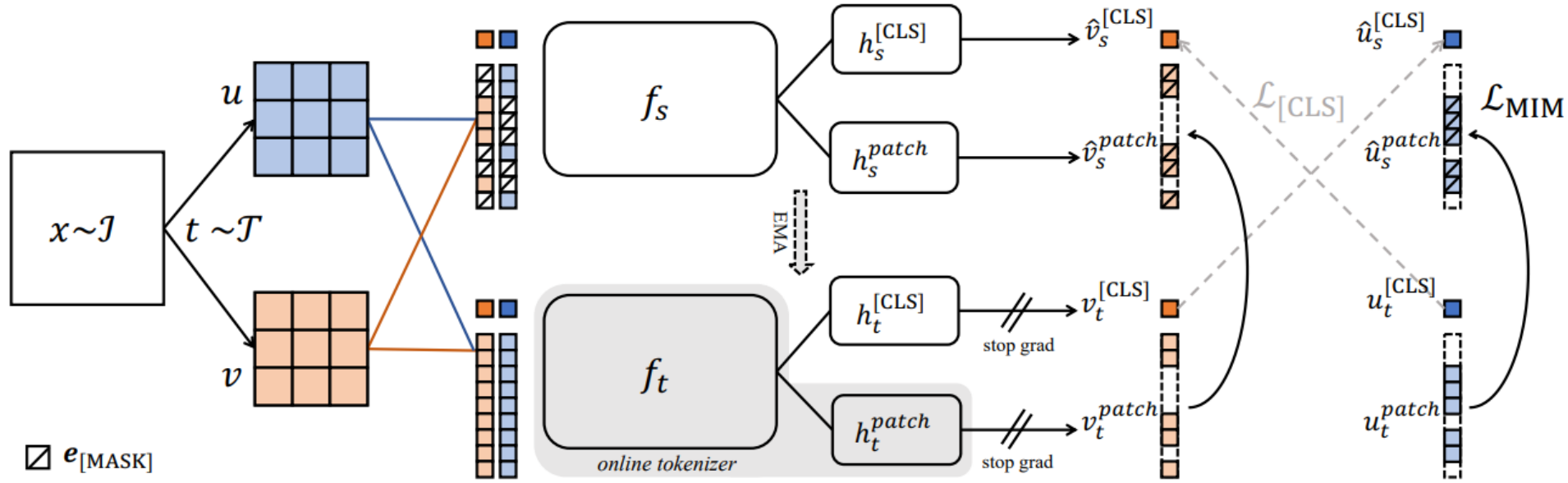
iBOT

- Training VM to obtain image tokenizers using Masked Image Modeling (MLM) objectives
- IBOT develop ideas of BEiT, but learns tokens from self-distillation (DINO), not from pretrained VAE distillation





Framework



$$\mathcal{L} = \mathcal{L}_{\text{CLS}} + \mathcal{L}_{\text{MIM}}, \quad \mathcal{L}_{\text{MIM}} = -\sum_{i=1}^{N} m_i \cdot P_{\boldsymbol{\theta}'}^{\text{patch}}(\boldsymbol{u}_i)^{\text{T}} \log P_{\boldsymbol{\theta}}^{\text{patch}}(\hat{\boldsymbol{u}}_i).$$

Multi-crop

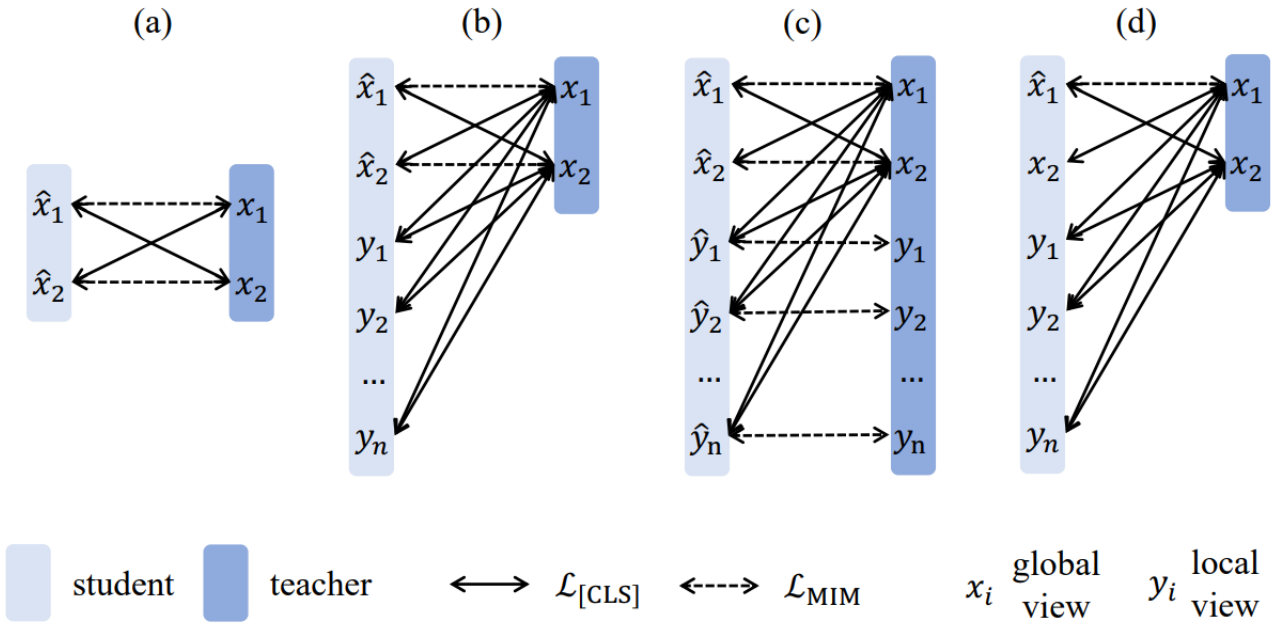


Figure 7: **Computation pipelines for iBOT with or without multi-crop augmentation.** (a) iBOT w/o multi-crop augmentation. (b), (c), and (d) are three pipelines w/ multi-crop augmentation. (b) does not perform MIM for local crops, whereas (c) performs MIM for all crops. (d) only performs MIM for one of the two global crops. iBOT uses (b) with random MIM.

ViT-S/16, 100 epochs	(a)	(b)	(c)	(d)	(e)	(b) w/ rand. MIM
k-NN	62.1	62.0	31.9	69.8	56.6	71.5

Learned patch patterns

MIM improves recognition?

CLS tokens lin. classification only 0.9% gap for ViT-S/16

Method	Arch.	Par.	im/s	Epo.[1]	k-NN	Lin.
SSL Transformers						
MoCov3	ViT-S/16	21	1007	1200	-	73.4
MoCov3	ViT-B/16	85	312	1200	-	76.7
SwAV	ViT-S/16	21	1007	2400	66.3	73.5
DINO	ViT-S/16	21	1007	3200	74.5	77.0
DINO	ViT-B/16	85	312	1600	76.1	78.2
EsViT	Swin-T/7	28	726	1200	75.7	78.1
EsViT	Swin-T/14	28	593	1200	77.0	78.7
iBOT	ViT-S/16	21	1007	3200	75.2	77.9
iBOT	Swin-T/7	28	726	1200	75.3	78.6
iBOT	Swin-T/14	28	593	1200	76.2	79.3
iBOT	ViT-B/16	85	312	1600	77.1	79.5
iBOT	ViT-L/16	307	102	1200	**78.0**	81.0
iBOT ‡	ViT-L/16	307	102	200	72.9	**82.3**

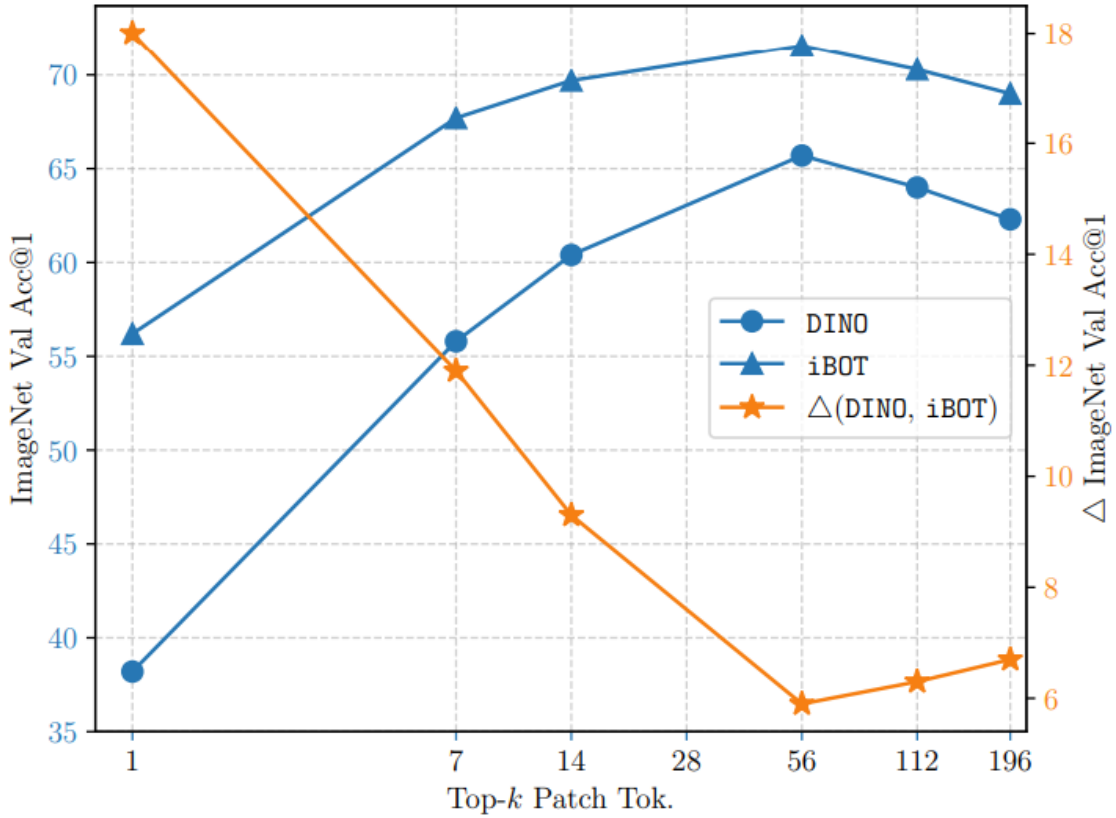
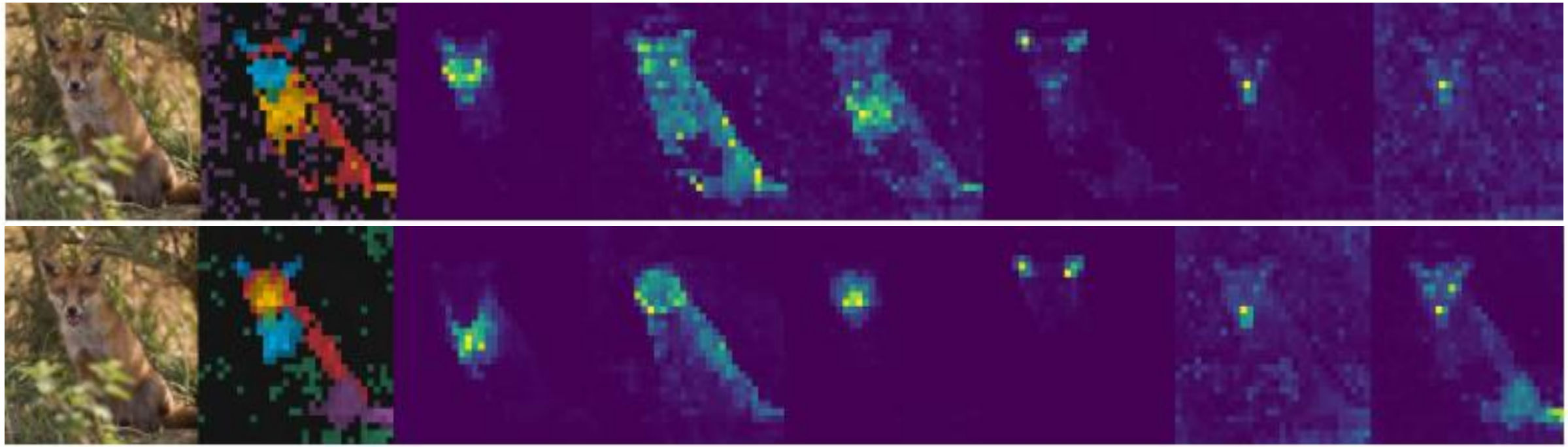


Figure 5: **Part-wise linear probing accuracy.** Top-k tokens with the highest attention scores are averaged for classification.

Comparison: attention maps

DINO

iBOT



Ablations (1)

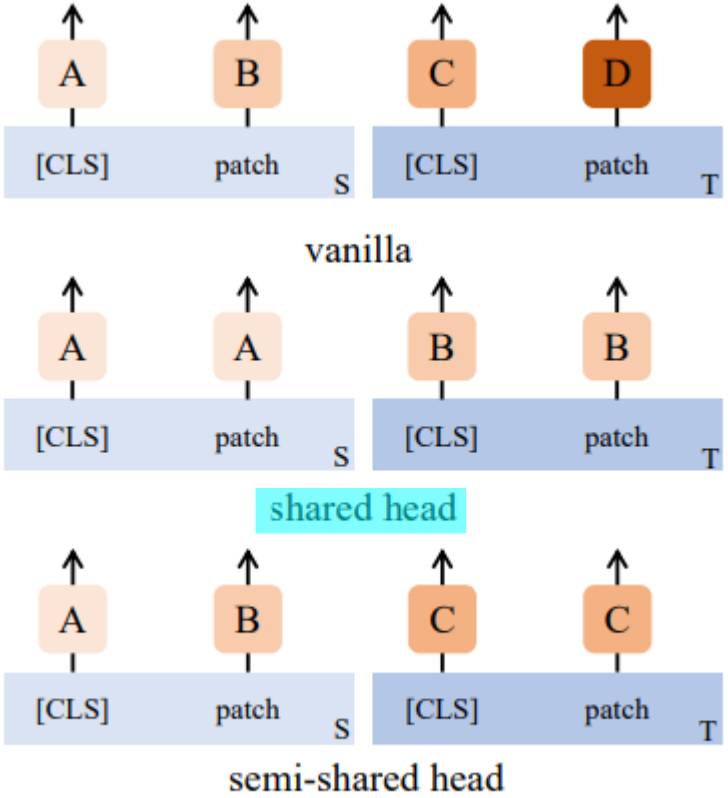
- Note that **network learns embeddings only with CLS loss**

Table 9: **Effect of design choices of semantically meaningful tokenization.**

Method	$\mathcal{L}_{\text{MIM}}$	$\mathcal{L}_{[\text{CLS}]}$	SH	k-NN	Lin.	Fin.
iBOT	✓	✓	✓	69.1	74.2	81.5
	✓	✓	✗	69.0	73.8	81.5
	✓	✗	-	9.5	29.8	79.4
	○	✗	-	44.3	60.0	81.7
BEiT	△	✗	-	6.9	23.5	81.4
DINO	✗	✓	-	67.9	72.5	80.6
BEiT + DINO	△	✓	-	48.0	62.7	81.2

○: standalone DINO (w/o mcrop, 300-epoch)
△: pre-trained DALL-E encoder

Ablations (2)



Arch.	vanilla	shared†	sm. shared	sm. shared†	shared
k-NN .	68.9	68.0	68.4	68.4	69.1
Lin.	73.9	73.7	73.7	73.8	74.2

Table 17: **Different head sharing strategy.**

Evaluation: ImageNet

Table 1: k-**NN and linear probing on ImageNet-1K.** † denotes using selective kernel. ‡ denotes pre-training on ImageNet-22K.

Method	Arch.	Par.	im/s	Epo.[1]	k-NN	Lin.
SSL big ResNets						
MoCov3	RN50	23	1237	1600	-	74.6
SwAV	RN50	23	1237	2400	65.7	75.3
DINO	RN50	23	1237	3200	67.5	75.3
BYOL	RN200w2	250	123	2000	73.9	79.6
SCLRv2	RN152w3†	794	46	2000	73.1	79.8
SSL Transformers						
MoCov3	ViT-S/16	21	1007	1200	-	73.4
MoCov3	ViT-B/16	85	312	1200	-	76.7
SwAV	ViT-S/16	21	1007	2400	66.3	73.5
DINO	ViT-S/16	21	1007	3200	74.5	77.0
DINO	ViT-B/16	85	312	1600	76.1	78.2
EsViT	Swin-T/7	28	726	1200	75.7	78.1
EsViT	Swin-T/14	28	593	1200	77.0	78.7
iBOT	ViT-S/16	21	1007	3200	75.2	77.9
iBOT	Swin-T/7	28	726	1200	75.3	78.6
iBOT	Swin-T/14	28	593	1200	76.2	79.3
iBOT	ViT-B/16	85	312	1600	77.1	79.5
iBOT	ViT-L/16	307	102	1200	**78.0**	81.0
iBOT ‡	ViT-L/16	307	102	200	72.9	**82.3**

Table 2: **Fine-tuning on ImageNet-1K.**

Method	Arch.	Epo.[1]	Acc.
Rand.	ViT-S/16	-	79.9
MoCov3	ViT-S/16	600	81.4
DINO	ViT-S/16	3200	82.0
iBOT	ViT-S/16	3200	**82.3**
Rand.	ViT-B/16	-	81.8
MoCov3	ViT-B/16	600	83.2
BEiT	ViT-B/16	800	83.4
DINO	ViT-B/16	1600	83.6
iBOT	ViT-B/16	1600	**84.0**
MoCov3	ViT-L/16	600	84.1
iBOT	ViT-L/16	1000	84.8
BEiT	ViT-L/16	800	**85.2**

Table 3: **Fine-tuning on ImageNet-1K.** Pre-training on ImageNet-22K.

Method	Arch.	Epo.[1]	Acc.
BEiT	ViT-B/16	150	83.7
iBOT	ViT-B/16	320	**84.4**
BEiT	ViT-L/16	150	86.0
iBOT	ViT-L/16	200	86.6
iBOT	ViT_{512}-L/16	200	**87.8**

Evaluation: low-shot learning and other tasks

Table 4: **Semi-supervised learning on ImageNet-1K.** 1% and 10% denotes label fraction. SD denotes self-distillation.

Method	Arch.	1%	10%
SimCLRv2	RN50	57.9	68.1
BYOL	RN50	53.2	68.8
SwAV	RN50	53.9	70.2
SimCLRv2+SD	RN50	60.0	70.5
DINO	ViT-S/16	60.3	74.3
iBOT	ViT-S/16	**61.9**	**75.1**

Table 5: **Unsupervised learning on ImageNet-1K.** † denotes k-means clustering on frozen features.

Method	Arch.	ACC	ARI	NMI	FMI
Self-label†	RN50	30.5	16.2	75.4	-
InfoMin†	RN50	33.2	14.7	68.8	-
SCAN	RN50	39.9	27.5	72.0	-
DINO	ViT-S/16	41.4	29.8	76.8	32.8
iBOT	ViT-S/16	**43.4**	**32.8**	**78.6**	**35.6**

Table 6: **Object detection (Det.) & instance segmentation (ISeg.) on COCO and Semantic segmentation (Seg.) on ADE20K.** We report the results of ViT-S/16 (left) and ViT-B/16 (right) Seg.† denotes using a linear head for semantic segmentation.

Method	Arch.	Param.	Det. AP^b	ISeg. AP^m	Seg. mIoU
Sup.	Swin-T	29	48.1	41.7	44.5
MoBY	Swin-T	29	48.1	41.5	44.1
Sup.	ViT-S/16	21	46.2	40.1	44.5
iBOT	ViT-S/16	21	**49.4**	**42.6**	**45.4**

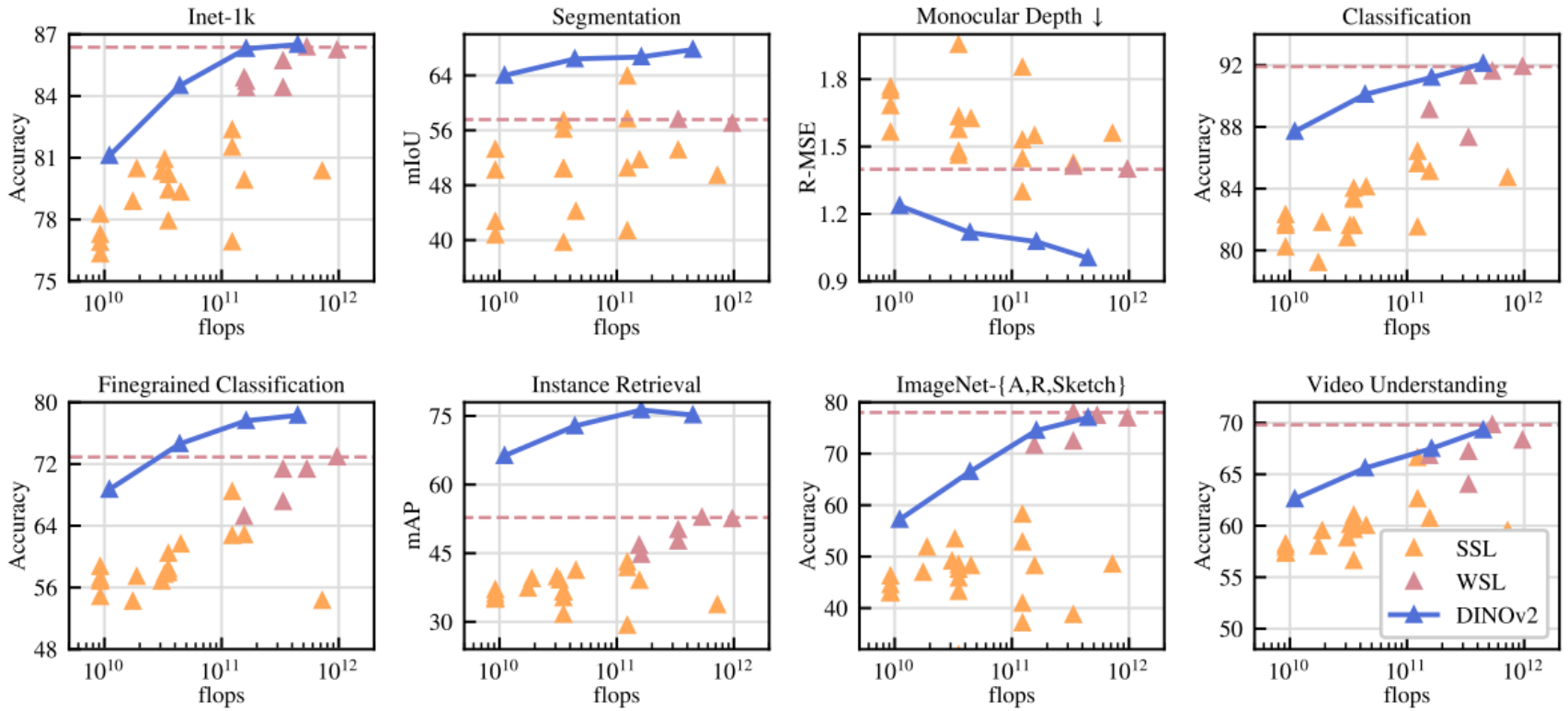
Method	Det. AP^b	ISeg. AP^m	Seg.† mIoU	Seg. mIoU
Sup.	49.8	43.2	35.4	46.6
BEiT	50.1	43.5	27.4	45.8
DINO	50.1	43.4	34.5	46.8
iBOT	**51.2**	**44.2**	**38.3**	**50.0**

DINOv2

DINOv2: Learning Robust Visual Features without Supervision
(2023, Meta AI Research)

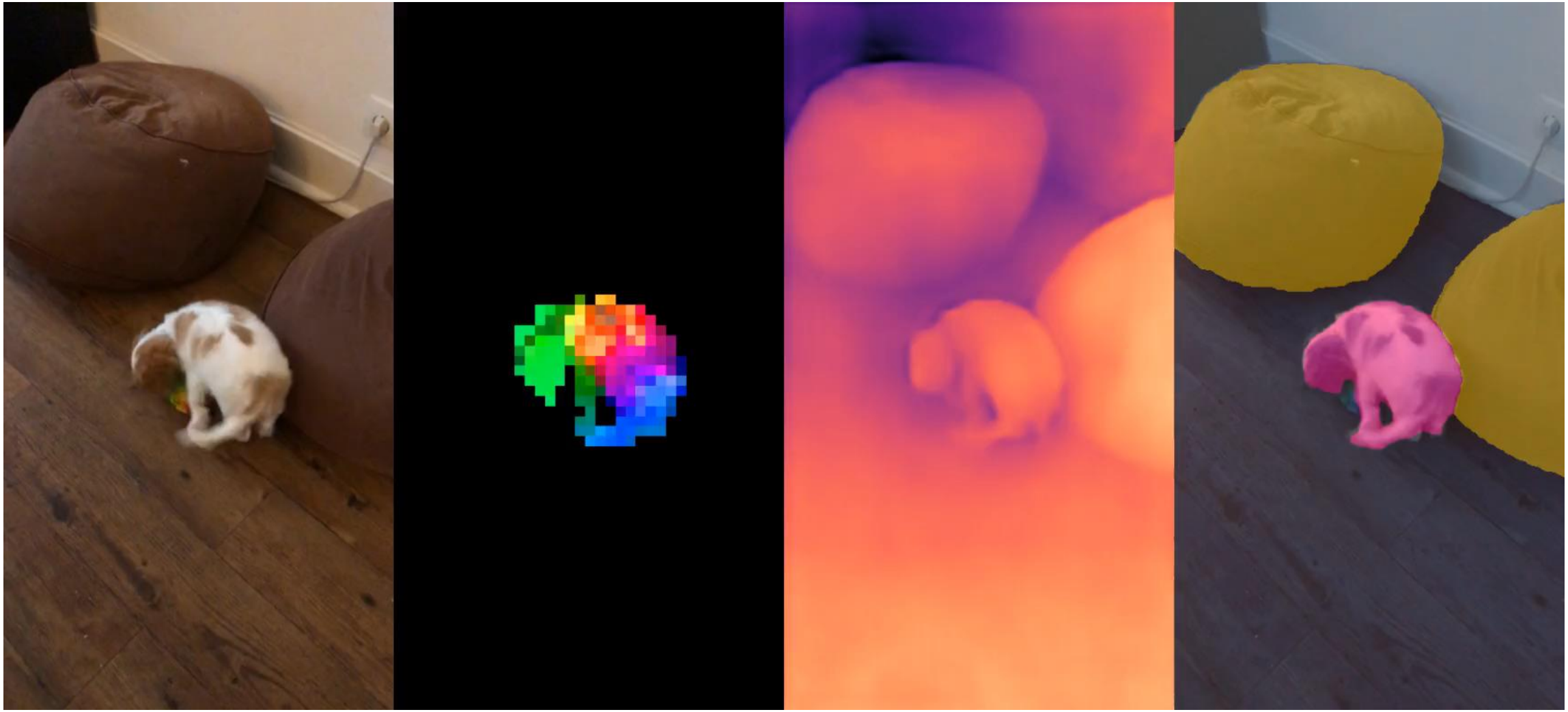
DINOv2

- **Efficient implementation -> batch size 3K**
- **Huge semi-curated dataset (142M)**
- Improves DINO+iBOT loss with **Sinkhorn-Knopp centering** and **KoLeo regularizer**
- **Distillation of smaller models** from huge model
- DINOv2 features **surpass the best available all-purpose features, OpenCLIP on most of the benchmarks at image and pixel levels**

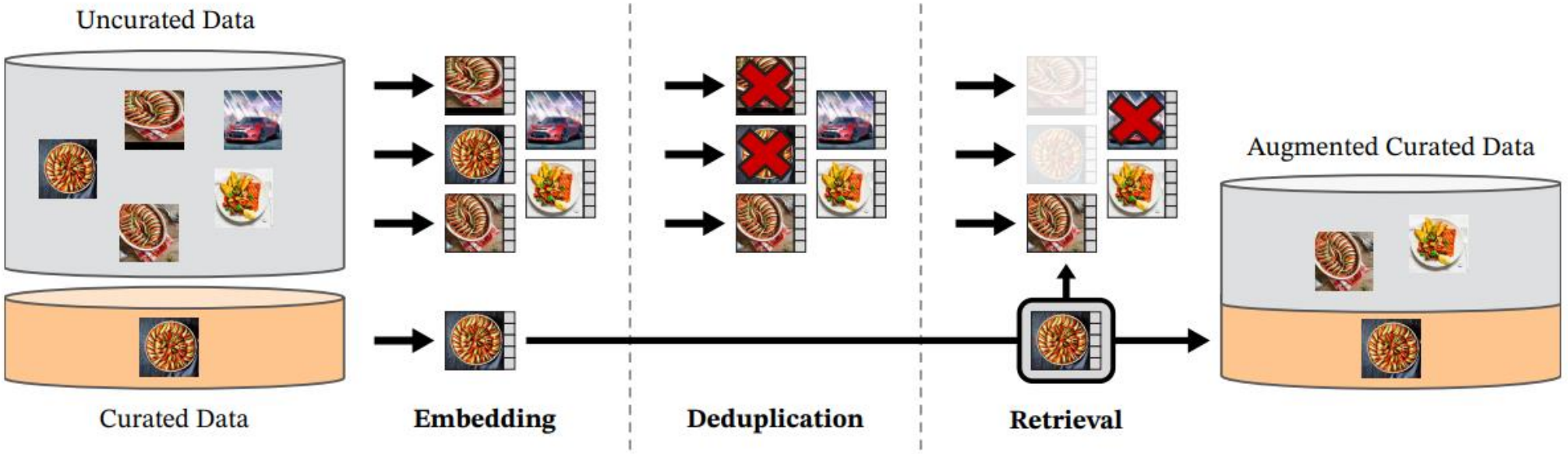


Dark blue – DINOv2; pale orange – other self-supervised methods; dark pink – weakly-supervised methods

Demo



LVD-142M dataset collection framework



- **Source curated datasets**: classif. (ImageNet-22k, ImageNet-1k); fine-grained classif. (Food-101, SUN397, etc), seg. (ADE20K, Cityscapes, Pascal VOC 2012); depth est. (Mapillary SLS, KITTY, NYU Depth V2, etc); retrieval (Google Landmarks, AmsterTime, Met, etc)
- Source of uncurated data: ?
- The whole processing is distributed on a compute cluster of **20 nodes equipped with 8 V100-32GB GPUs** and takes less than two days to produce the **LVD-142M dataset**

Loss changes

- **DINO image-level loss**, **including multi-crops**
- **iBOT MIM loss** is combined with the image-level loss
- Tying the weights associated with both objectives makes the model **underfit at the patch-level while overfitting at the image-level**. **Untying resolves this issue and improve the performances at both scales**
- **Sinkhorn-Knopp centering instead of** the teacher softmax-centering step. We run the Sinkhorn-Knopp algorithm steps for 3 iterations. For the student, the softmax normalization is applied
- **KoLeo regularizer** encourages a uniform span of the features within a batch

$$\mathcal{L}_{koleo} = \frac{1}{n}\sum_{i=1}^{n} \log\left(d_{n,i}\right), \qquad d_{n,i} = \min_{i \neq j} \left\| x_i - x_j \right\|$$

- **Increase resolution at the end of pretraining** improves performance on pixel-level downstream tasks such as segmentation or detection, where small objects disappear at low resolutions (from 224x224 to 518x518)

Efficient implementation

Improvements

- Code reproduction
- Fast and memory-efficient attention (FlashAttention)
- Nested tensors in self-attention (xFormers library)
- Efficient stochastic depth implementation
- Fully-Sharded Data Parallel (FSDP)

- Distill low-compute SSL models from giant SSL model

Implementation ablations

	INet-1k k-NN	INet-1k linear
iBOT	72.9	82.3
+(our reproduction)	74.5 ↑ 1.6	83.2 ↑ 0.9
+LayerScale, Stochastic Depth	75.4 ↑ 0.9	82.0 ↓ 1.2
+128k prototypes	76.6 ↑ 1.2	81.9 ↓ 0.1
+KoLeo	78.9 ↑ 2.3	82.5 ↑ 0.6
+SwiGLU FFN	78.7 ↓ 0.2	83.1 ↑ 0.6
+Patch size 14	78.9 ↑ 0.2	83.5 ↑ 0.4
+Teacher momentum 0.994	79.4 ↑ 0.5	83.6 ↑ 0.1
+Tweak warmup schedules	80.5 ↑ 1.1	83.8 ↑ 0.2
+Batch size 3k	81.7 ↑ 1.2	84.7 ↑ 0.9
+Sinkhorn-Knopp	81.7 =	84.7 =
+Untying heads = DINOv2	82.0 ↑ 0.3	84.5 ↓ 0.2

Table 1: **Ablation study of the training differences between iBOT and DINOv2.** We optimize for k-NN performance, as in our experience, the linear probe performance is lower-bounded by the k-NN performance. Some modifications, like LayerScale and a high Stochastic Depth (rate=0.4), incur a decrease in linear probe performance, but have the benefits of increasing the stability of training by avoiding NaN loss values during training. Overall, these modifications allowed for the next set of improvements to be added. Experiments are run using the ViT-Large architecture on ImageNet-22k.

Low-compute models distillation

A simple recipe for competitive low-compute self supervised vision models (2023, Meta AI)

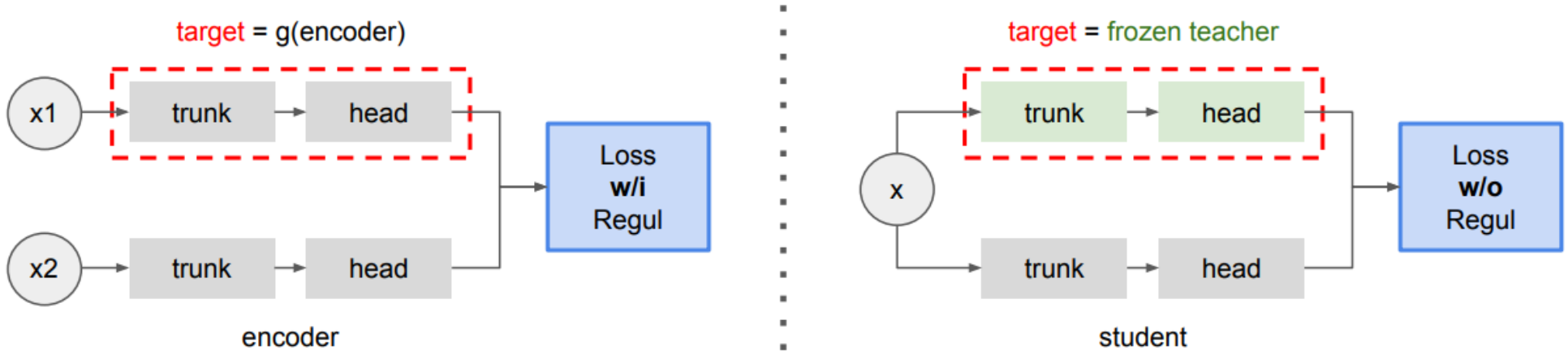


Figure 3. **Overview of our method.** RoB adapts existing joint-embedding approaches (left) by replacing the branch that produces the target with a frozen teacher pre-trained with the same SSL method (right). Importantly, RoB removes the regularisation terms that aim at preventing collapse from the loss, and use identical-view predictions instead of cross-view predictions in the loss.

$$\mathcal{L}_{\text{DINO}} = \frac{1}{N}\Big(\sum_{i=0}^{1} H(z_i^T, z_i^S) + \sum_{i=2}^{N-1}\sum_{j=0}^{1} H(z_j^T, z_i^S)\Big), \qquad \mathcal{L}_{\text{iBOT}} = \lambda_1 \mathcal{L}_{\text{DINO}} + \frac{\lambda_2}{2N_{\text{mask}}} \sum_{i=0}^{1}\sum_{p=1}^{N_{\text{mask}}} H(z_{i,p}^T, z_{i,p}^S),$$

DINOv2 distillation

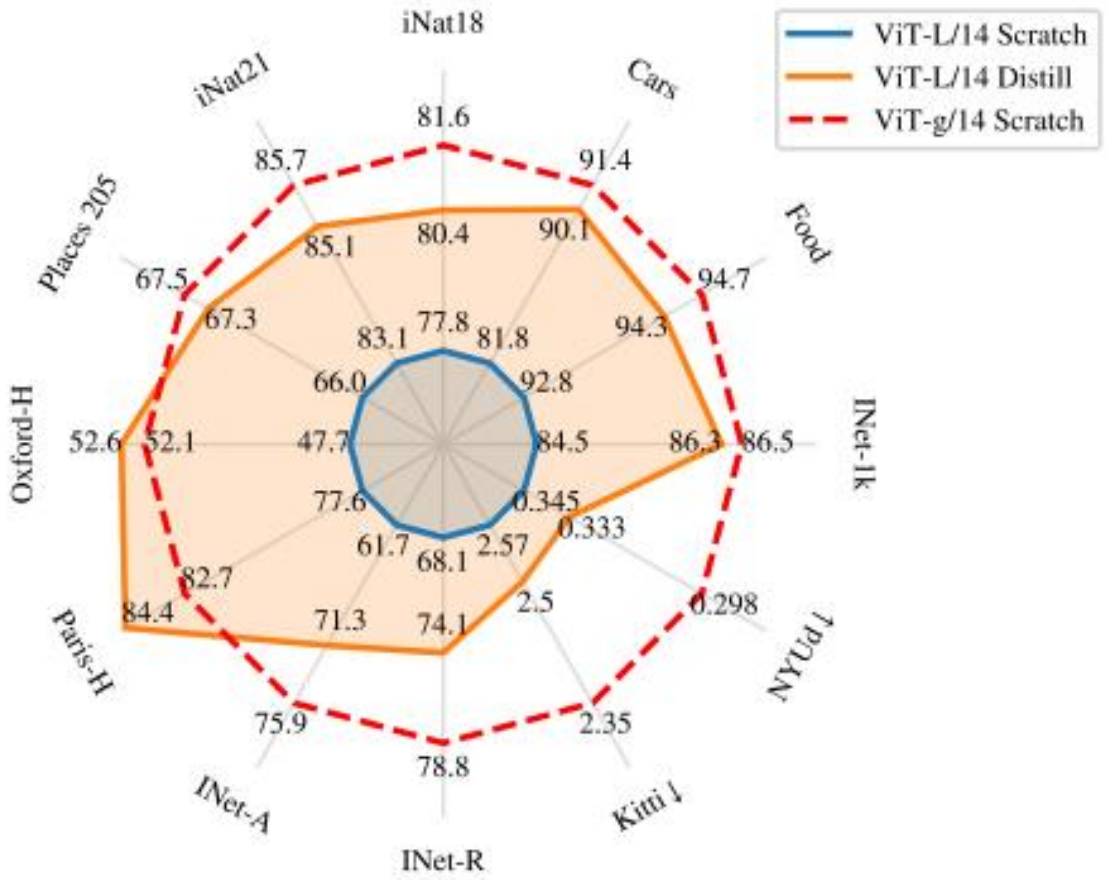
DINOv2 leverage the same training loop with a few exceptions:

- Use a **larger model as a frozen teacher**
- **Keep a spare EMA of the student** that we use as our **final model**
- Remove the masking (?) and stochastic depth
- Apply the **iBOT loss on the two global crops**

Arch	Method	INet-1k	Segm.	Depth↓	Classif.
ViT-g/14	Scratch	86.5	73.4	1.00	92.1
ViT-L/14	Scratch	84.5	72.2	1.10	90.2
ViT-L/14	Distill	**86.3**	**73.3**	**1.08**	**91.2**

Arch	Method	Finegr.	Retriev.	ARSketch	Video
ViT-g/14	Scratch	78.3	75.2	77.0	69.3
ViT-L/14	Scratch	75.8	71.3	69.5	67.3
ViT-L/14	Distill	**77.6**	**76.3**	**74.5**	**67.5**

(b) Averaged metrics on 8 vision tasks



(a) Comparison on individual metrics

Loss ablations

KoLeo	INet-1k	Im-A	ADE-20k	Oxford-M
✗	85.3	70.6	47.2	55.6
✓	85.8	72.8	47.1	63.9

(a) Koleo loss

MIM	INet-1k	Im-A	ADE-20k	Oxford-M
✗	85.3	72.0	44.2	64.3
✓	85.8	72.8	47.1	63.9

(b) MIM objective in iBOT

Table 3: **(a)** Effect of the KoLeo loss term. **(b)** Effect of the iBOT Masked Image Modeling (MIM) loss term. Evaluation performed on ImageNet-{1k,A} (classification with linear probe, accuracy %), ADE-20k (segmentation with linear layer, mIoU) and Oxford-M (image retrieval, mAP). Each model is trained on the same number of iterations, that is smaller than our final run. The KoLeo loss term improves nearest-neighbor search tasks (e.g. retrieval), and the MIM loss improves patch-level tasks (e.g. segmentation).

Datasets ablations: curated vs uncurated

Training Data	INet-1k	Im-A	ADE-20k	Oxford-M
INet-22k	85.9	73.5	46.6	62.5
INet-22k \ INet-1k	85.3	70.3	46.2	58.7
Uncurated data	83.3	59.4	48.5	54.3
LVD-142M	85.8	73.9	47.7	64.6

Table 2: **Ablation of the source of pretraining data.** We compare the INet-22k dataset that was used in iBOT to our dataset, LVD-142M. Each model is trained for the same number of iterations, that is smaller than in our final run. Pretraining on LVD-142M maintains the performance over INet-1k while leading to models that perform better in other domains.

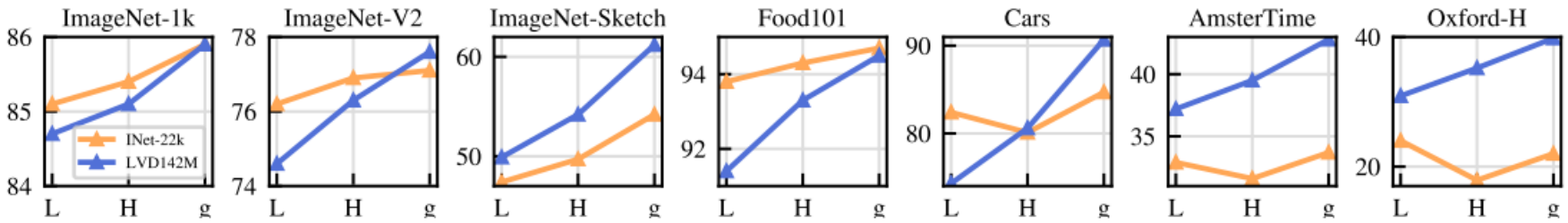


Figure 4: **Model scale versus data scale.** Evolution of performance as a function of model size for two different pretraining datasets: ImageNet-22k (14M images) and LVD-142M (142M images). The ViT-g trained on LVD-142M surpasses the ViT-g trained on ImageNet-22k on most benchmarks.

Resolution ablations

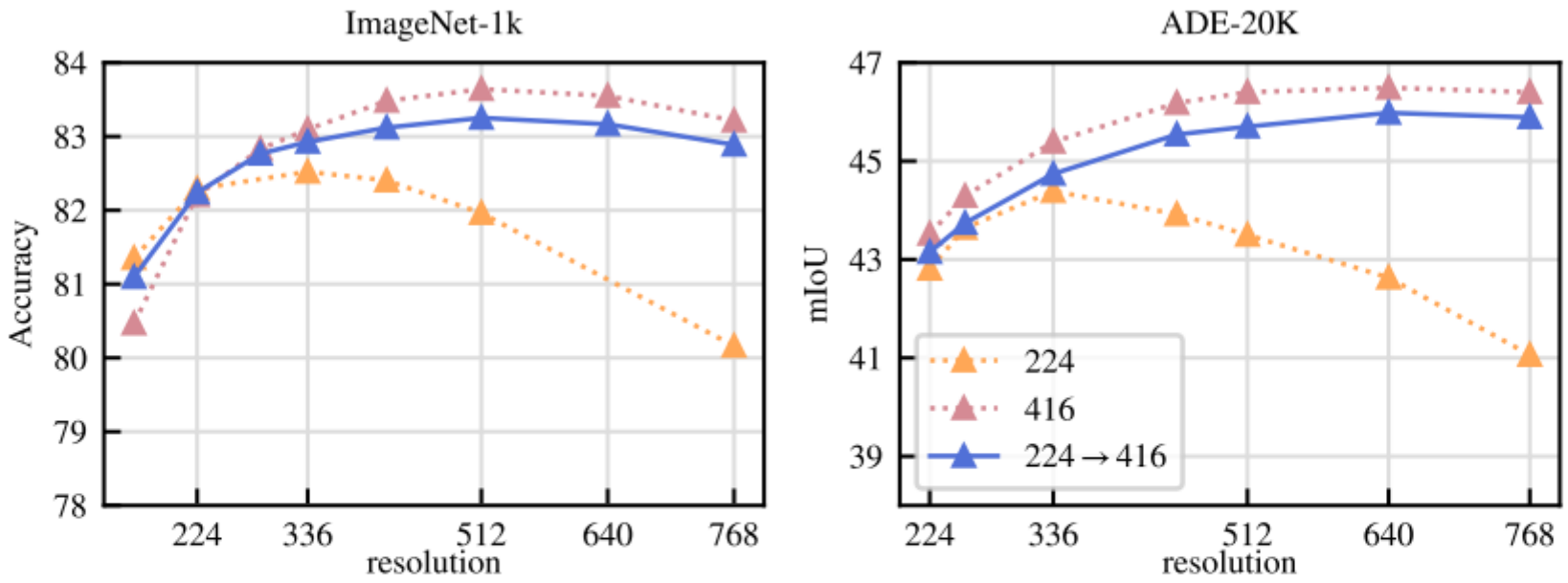
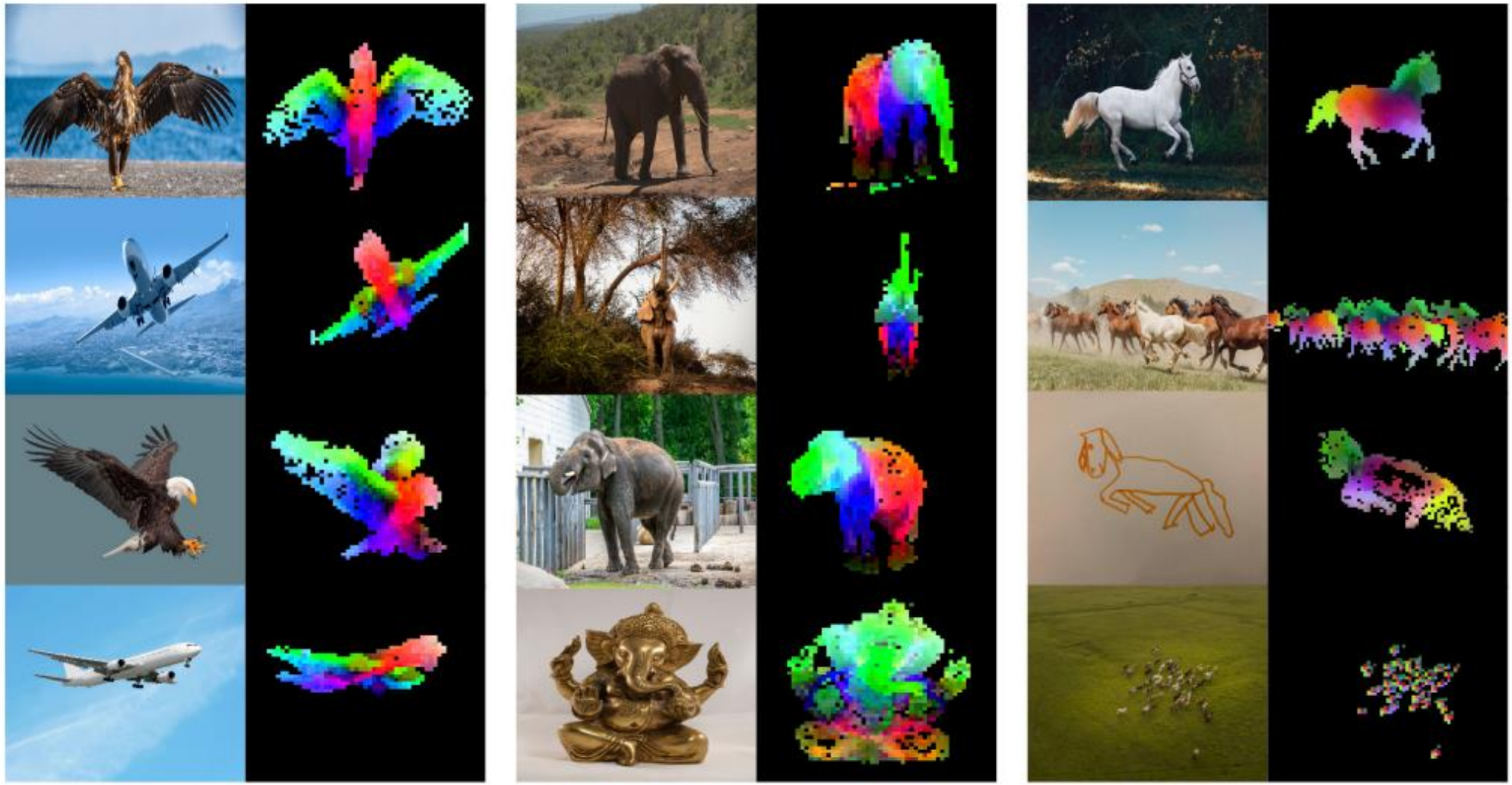


Figure 6: **Role of resolution.** Performance of ViT-L/16 trained on ImageNet-1k at fixed resolution ("224" and "416") or trained at 224 then 416 for a short duration ("224→416"). We train linear classifiers on top of frozen features at different resolutions and report Top-1 accuracy on ImageNet and mIoU on ADE-20k. We observe that performing SSL training at high resolution for a short duration achieve behavior and results close to training at the same high resolution for the full training, at a fraction of the cost.

Self-attention visualization



Evaluation: ImageNet

Method	Arch.	Data	Text sup.	kNN	linear		
				val	val	ReaL	V2
Weakly supervised							
CLIP	ViT-L/14	WIT-400M	✓	79.8	84.3	88.1	75.3
CLIP	ViT-L/14_{336}	WIT-400M	✓	80.5	85.3	88.8	75.8
SWAG	ViT-H/14	IG3.6B	✓	82.6	85.7	88.7	77.6
OpenCLIP	ViT-H/14	LAION	✓	81.7	84.4	88.4	75.5
OpenCLIP	ViT-G/14	LAION	✓	83.2	86.2	89.4	77.2
EVA-CLIP	ViT-g/14	custom*	✓	**83.5**	86.4	89.3	77.4
Self-supervised							
MAE	ViT-H/14	INet-1k	✗	49.4	76.6	83.3	64.8
DINO	ViT-S/8	INet-1k	✗	78.6	79.2	85.5	68.2
SEERv2	RG10B	IG2B	✗	–	79.8	–	–
MSN	ViT-L/7	INet-1k	✗	79.2	80.7	86.0	69.7
EsViT	Swin-B/W=14	INet-1k	✗	79.4	81.3	87.0	70.4
Mugs	ViT-L/16	INet-1k	✗	80.2	82.1	86.9	70.8
iBOT	ViT-L/16	INet-22k	✗	72.9	82.3	87.5	72.4
DINOv2	ViT-S/14	LVD-142M	✗	79.0	81.1	86.6	70.9
	ViT-B/14	LVD-142M	✗	82.1	84.5	88.3	75.1
	ViT-L/14	LVD-142M	✗	**83.5**	86.3	89.5	78.0
	ViT-g/14	LVD-142M	✗	**83.5**	**86.5**	**89.6**	**78.4**

Evaluation: domain generalization tests

Method	Arch	Data	Im-A	Im-R	Im-C↓	Sketch
OpenCLIP	ViT-G/14	LAION	63.8	**87.8**	45.3	**66.4**
MAE	ViT-H/14	INet-1k	10.2	34.4	61.4	21.9
DINO	ViT-B/8	INet-1k	23.9	37.0	56.6	25.5
iBOT	ViT-L/16	INet-22k	41.5	51.0	43.9	38.5
DINOv2	ViT-S/14	LVD-142M	33.5	53.7	54.4	41.2
	ViT-B/14	LVD-142M	55.1	63.3	42.7	50.6
	ViT-L/14	LVD-142M	71.3	74.4	31.5	59.3
	ViT-g/14	LVD-142M	**75.9**	78.8	**28.2**	62.5

Table 6: **Domain Generalization with a linear probe** on top of frozen features at a resolution of 224. Higher numbers are better for all benchmarks except Im-C.

- Here best models with linear classification heads trained on ImageNet-1k are evaluated on domain generalization benchmarks

Evaluation: semantic segmentation

Method	Arch.	ADE20k (62.9)		CityScapes (86.9)		Pascal VOC (89.0)	
		lin.	+ms	lin.	+ms	lin.	+ms
OpenCLIP	ViT-G/14	39.3	46.0	60.3	70.3	71.4	79.2
MAE	ViT-H/14	33.3	30.7	58.4	61.0	67.6	63.3
DINO	ViT-B/8	31.8	35.2	56.9	66.2	66.4	75.6
iBOT	ViT-L/16	44.6	47.5	64.8	74.5	82.3	84.3
DINOv2	ViT-S/14	44.3	47.2	66.6	77.1	81.1	82.6
	ViT-B/14	47.3	51.3	69.4	80.0	82.5	84.9
	ViT-L/14	47.7	**53.1**	70.3	80.9	82.1	86.0
	ViT-g/14	**49.0**	53.0	**71.3**	**81.0**	**83.0**	**86.2**

Table 10: **Semantic segmentation on ADE20K, CityScapes and Pascal VOC with frozen features** and a linear classifier (lin.) and with multiscale (+ms). The absolute state of the art – from Wang et al. (2022), Liu et al. (2021) and Chen et al. (2018) respectively – are mentioned at the top of the Table. For reference, using the Mask2Former pipeline (Steiner et al., 2021) with a ViT-Adapter (Chen et al., 2022) on top of our frozen ViT-g/14 backbone gives 60.2 mIoU on ADE-20k.

Evaluation: depth estimation

		NYUd (0.330)			KITTI (2.10)			NYUd → SUN RGB-D (0.421)		
Method	Arch.	lin. 1	lin. 4	DPT	lin. 1	lin. 4	DPT	lin. 1	lin. 4	DPT
OpenCLIP	ViT-G/14	0.541	0.510	0.414	3.57	3.21	2.56	0.537	0.476	0.408
MAE	ViT-H/14	0.517	0.483	0.415	3.66	3.26	2.59	0.545	0.523	0.506
DINO	ViT-B/8	0.555	0.539	0.492	3.81	3.56	2.74	0.553	0.541	0.520
iBOT	ViT-L/16	0.417	0.387	0.358	3.31	3.07	2.55	0.447	0.435	0.426
DINOv2	ViT-S/14	0.449	0.417	0.356	3.10	2.86	2.34	0.477	0.431	0.409
	ViT-B/14	0.399	0.362	0.317	2.90	2.59	2.23	0.448	0.400	0.377
	ViT-L/14	0.384	0.333	0.293	2.78	2.50	2.14	0.429	0.396	0.360
	ViT-g/14	**0.344**	**0.298**	**0.279**	**2.62**	**2.35**	**2.11**	**0.402**	**0.362**	**0.338**

Table 11: **Depth estimation with frozen features**. We report performance when training a linear classifier on top of one (lin. 1) or four (lin. 4) transformer layers, as well, as the DPT decoder (DPT) of Ranftl et al. (2021). We report the RMSE metric on the 3 datasets. Lower is better. For reference, we report state-of-the-art results taken from Li et al. (2022b) on each benchmark on top of the Table.

Examples (1)

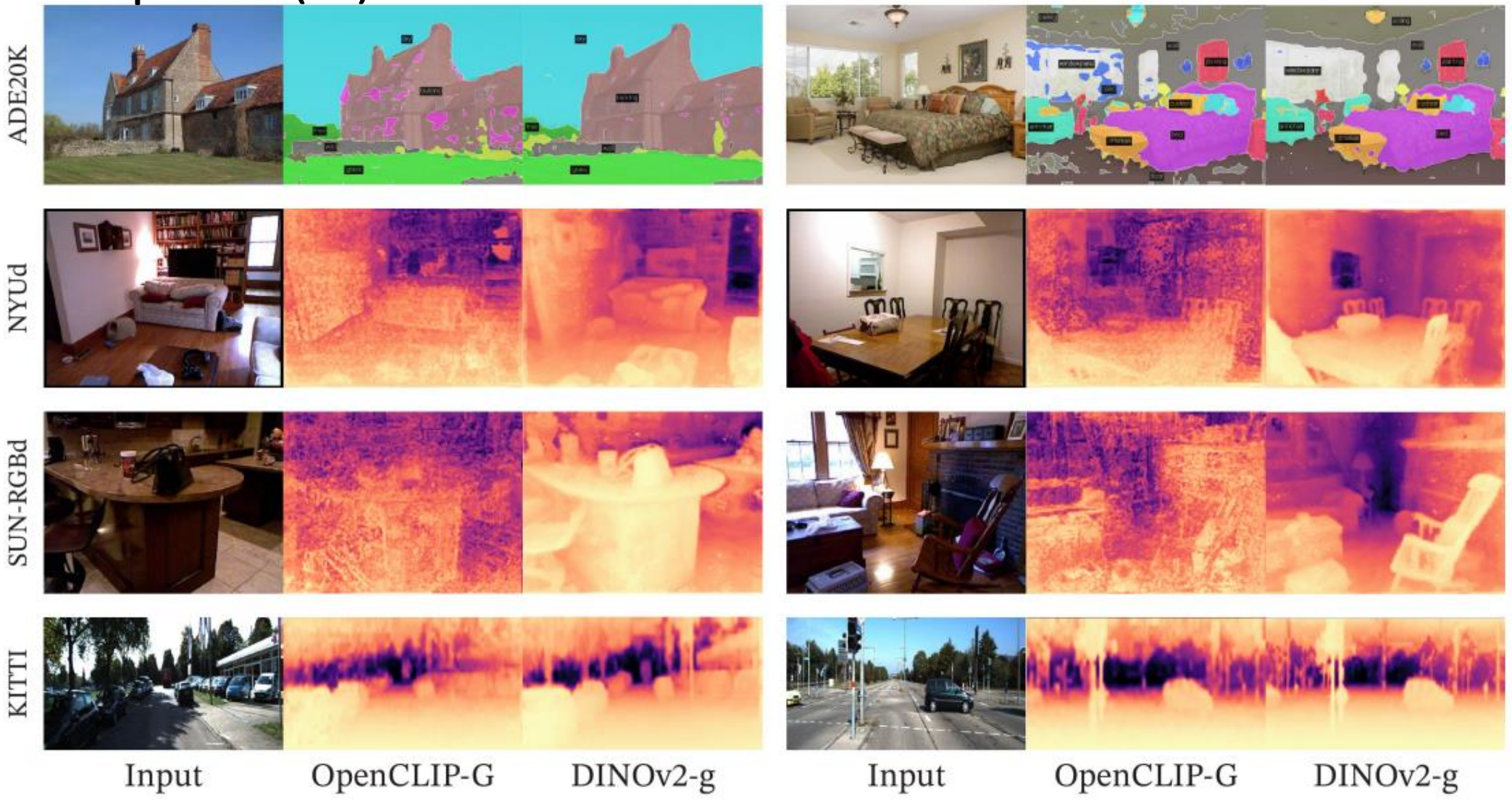


Figure 7: **Segmentation and depth estimation with linear classifiers.** Examples from ADE20K, NYUd, SUN RGB-D and KITTI with a linear probe on frozen OpenCLIP-G and DINOv2-g features.

Examples (2)

Figure 8: **Examples of out-of-distribution examples** with frozen DINOv2-g features and a linear probe.

Conclusions

- Contrastive approach is computationally expensive and challenging
- SotA SSL models avoid this issue by using **self-distillation:**
  - **Augmentations**
  - **Big batch sizes (3k)**
  - **Regularizations (sharpening-centering, forcing batch uniform distribution)**
  - **Multi-crops**
  - **Patch-masking**
  - **Big semi-curated datasets**
- Obtained SSL features shows **strong generalization abilities** and can be used on downstream tasks **frozen** or **as pretrain**
- **Small models should be obtained from distillation**, not from scratch training

Literature

- SSL: The dark matter of intelligence
- BYOL
- DINOv1
- iBOT
- DINOv2
- A Simple Recipe for Competitive Low-compute Self supervised Vision Models