1

Self-Supervised Learning

Wei-Ta Chu

Supervised vs. Unsupervised Learning

	Supervised Learning	Unsupervised learning
Objective	To approximate a function that maps inputs to outputs based out example input-output pairs.	To build a concise representation of the data and generate imaginative content from it.
Accuracy	Highly accurate and reliable.	Less accurate and reliable.
Complexity	Simpler method.	Computationally complex.
Classes	Number of classes is *known.*	Number of classes is *unknown.*
Output	A desired output value (also called the supervisory signal).	No corresponding output values.

https://www.datacamp.com/blog/introduction-to-unsupervised-learning

Types of Unsupervised Learning

- Clustering
  - Tons of clustering algorithms
  - K-means, DBSCAN, affinity propagation, …
- Association Rule Mining
  - We typically see association rule mining used for market basket analysis.
  - the Apriori algorithm
- Dimensionality Reduction
  - Principle component analysis, singular value decomposition, …

SimCLR

Wei-Ta Chu

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton, “A Simple Framework for Contrastive Learning of Visual Representations,” ICML, 2020.

Introduction

- Goal: Learning effective visual representations without human supervision.
- Two categories of approaches: generative or discriminative
- Generative approaches learn to generate or otherwise model pixels in the input space.
- Discriminative approaches based on contrastive learning.

SimCLR

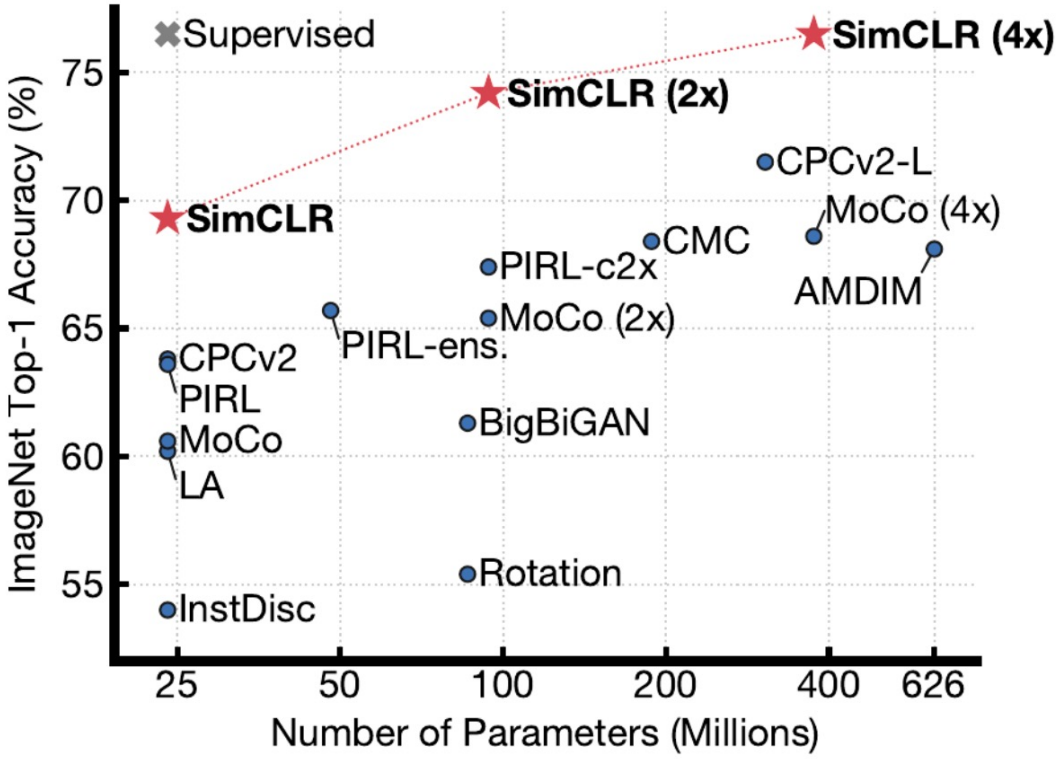


Figure 1. ImageNet Top-1 accuracy of linear classifiers trained on representations learned with different self-supervised methods (pretrained on ImageNet). Gray cross indicates supervised ResNet-50. Our method, SimCLR, is shown in bold.

The Contrastive Learning Framework

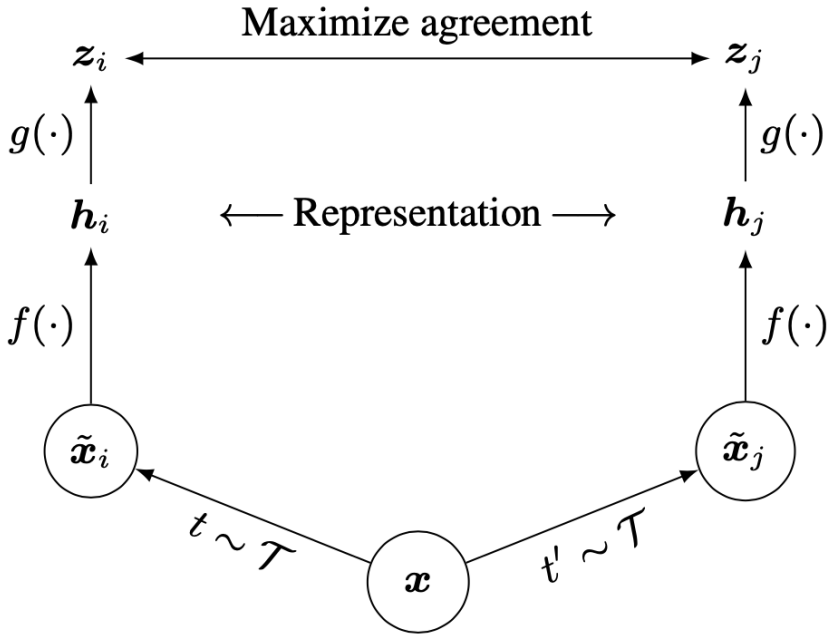


Figure 2. A simple framework for contrastive learning of visual representations. Two separate data augmentation operators are sampled from the same family of augmentations ($t \sim \mathcal{T}$ and $t' \sim \mathcal{T}$) and applied to each data example to obtain two correlated views. A base encoder network $f(\cdot)$ and a projection head $g(\cdot)$ are trained to maximize agreement using a contrastive loss. After training is completed, we throw away the projection head $g(\cdot)$ and use encoder $f(\cdot)$ and representation $\boldsymbol{h}$ for downstream tasks.

The Contrastive Learning Framework

- Stochastic data augmentation module
  - Random cropping followed by resize back to the original size
  - Random color distortions
  - Random Gaussian blur
- Base encoder
  - ResNet

$$\boldsymbol{h}_i = f(\tilde{\boldsymbol{x}}_i) = \text{ResNet}(\tilde{\boldsymbol{x}}_i)$$

- Projection head
  - Map representations to the space where contrastive loss is applied.
  - An MLP with one hidden layer

$$\boldsymbol{z}_i = g(\boldsymbol{h}_i) = W^{(2)}\sigma(W^{(1)}\boldsymbol{h}_i)$$

The Contrastive Learning Framework

- Contrastive loss
  - We randomly sample a minibatch of N examples and define the contrastive prediction task on pairs of augmented examples – $2N$ data points.
  - Given a positive pair, we treat the other $2(N-1)$ augmented examples within a minibatch as negative examples.
  - The loss function for a positive pair of examples (i, j) is defined as

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)}$$

$$\text{sim}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}^\top \boldsymbol{v} / \|\boldsymbol{u}\| \|\boldsymbol{v}\|$$

    τ denotes a temperature parameter
  - The final loss is computed across all positive pairs, both (i,j) and (j,i), in a mini-batch.

Training with Large Batch Size

- We vary the training batch size N from 256 to 8192. A batch size of 8192 gives us 16382 negative examples per positive pair from both augmentation views.
- Training with large batch size may be unstable when using standard SGD/Momentum with linear learning rate scaling. To stabilize the training, we use the LARS optimizer (You et al., 2017) for all batch sizes

Evaluation Protocol

- Dataset: ImageNet ILSVRC-2012 dataset
- A linear classifier is trained on top of the frozen base network, and test accuracy is used as a proxy for representation quality.

- For data augmentation we use random crop and resize (with random flip), color distortions, and Gaussian blur.
- ResNet-50 as the base encoder network, and a 2-layer MLP projection head to project the representation to a 128-dimensional latent space.
- Use linear warmup for the first 10 epochs, and decay the learning rate with the cosine decay schedule without restarts

Data Augmentation for Contrastive Representation Learning

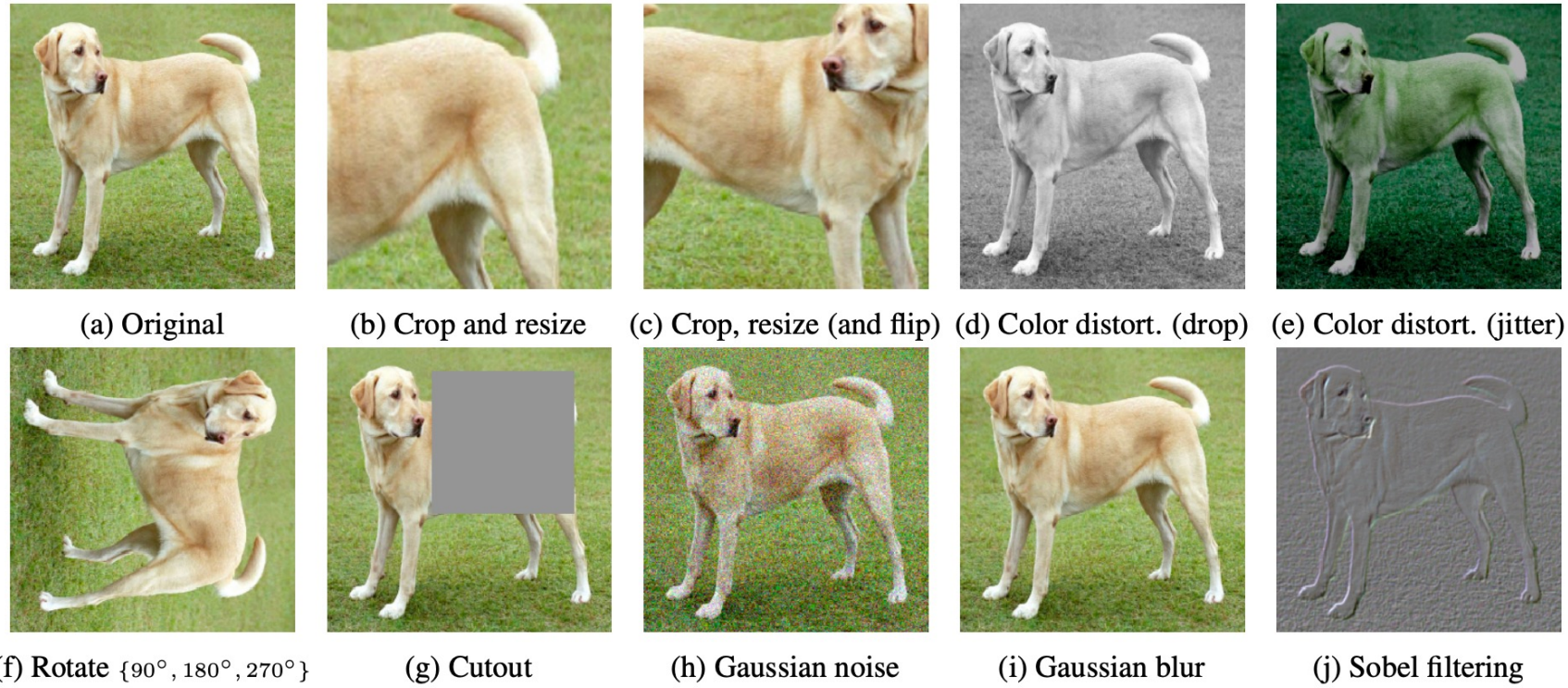


Figure 4. Illustrations of the studied data augmentation operators. Each augmentation can transform data stochastically with some internal parameters (e.g. rotation degree, noise level). Note that we *only* test these operators in ablation, the *augmentation policy used to train our models* only includes *random crop (with flip and resize)*, *color distortion*, and *Gaussian blur*. (Original image cc-by: Von.grzanka)

Composition of data augmentation operations is crucial

- We always first randomly crop images and resize them to the same resolution, and we then apply the targeted transformation(s) only to one branch of the framework in Figure 2, while leaving the other branch as the identity (i.e. $t(x_i) = x_i$).

- Figure 5 shows that *no single transformation suffices to learn good representations*.
- One composition of augmentations stands out: random cropping and random color distortion.

Composition of data augmentation operations is crucial

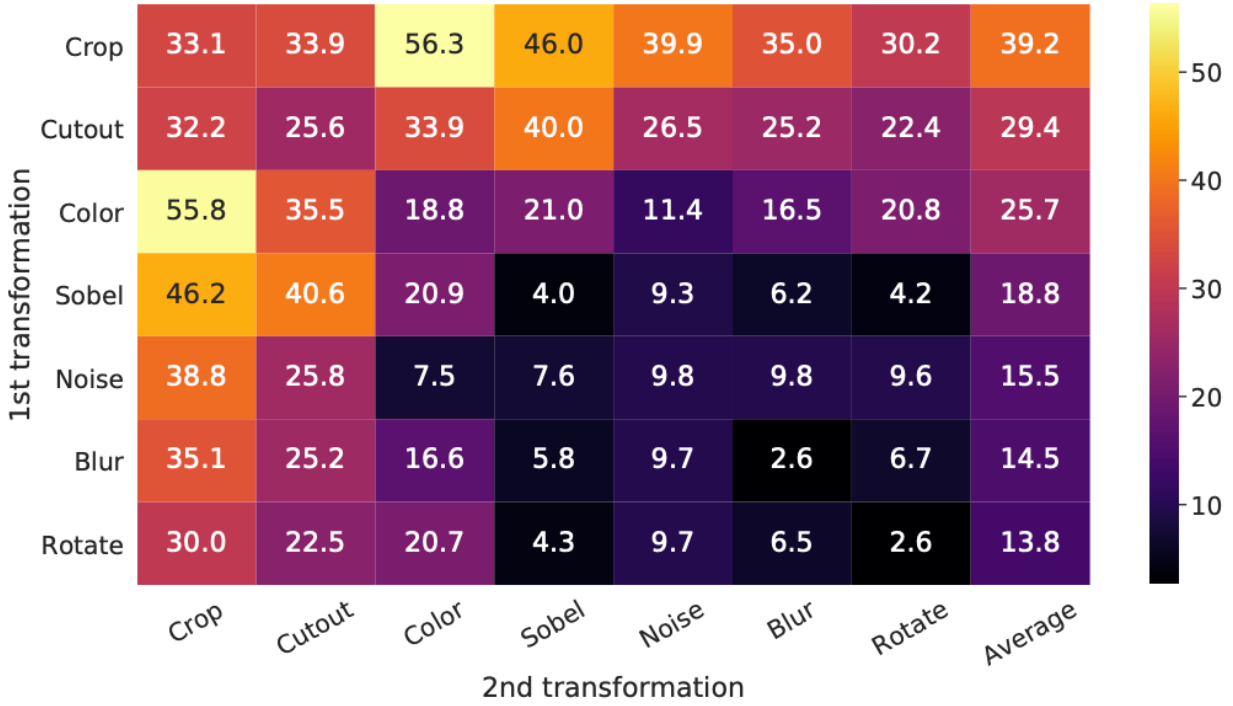


Figure 5. Linear evaluation (ImageNet top-1 accuracy) under individual or composition of data augmentations, applied only to one branch. For all columns but the last, diagonal entries correspond to single transformation, and off-diagonals correspond to composition of two transformations (applied sequentially). The last column reflects the average over the row.

Contrastive learning needs stronger data augmentation than supervised learning

- Stronger color augmentation substantially improves the linear evaluation.

Methods	Color distortion strength 1/8	1/4	1/2	1	1 (+Blur)	AutoAug
SimCLR	59.6	61.0	62.6	63.2	64.5	61.1
Supervised	77.0	76.7	76.5	75.7	75.4	77.1

Table 1. Top-1 accuracy of unsupervised ResNet-50 using linear evaluation and supervised ResNet-50[5], under varied color distortion strength (see Appendix A) and other data transformations. Strength 1 (+Blur) is our default data augmentation policy.

Architectures for Encoder and Head

- Unsupervised contrastive learning benefits (more) from bigger models.

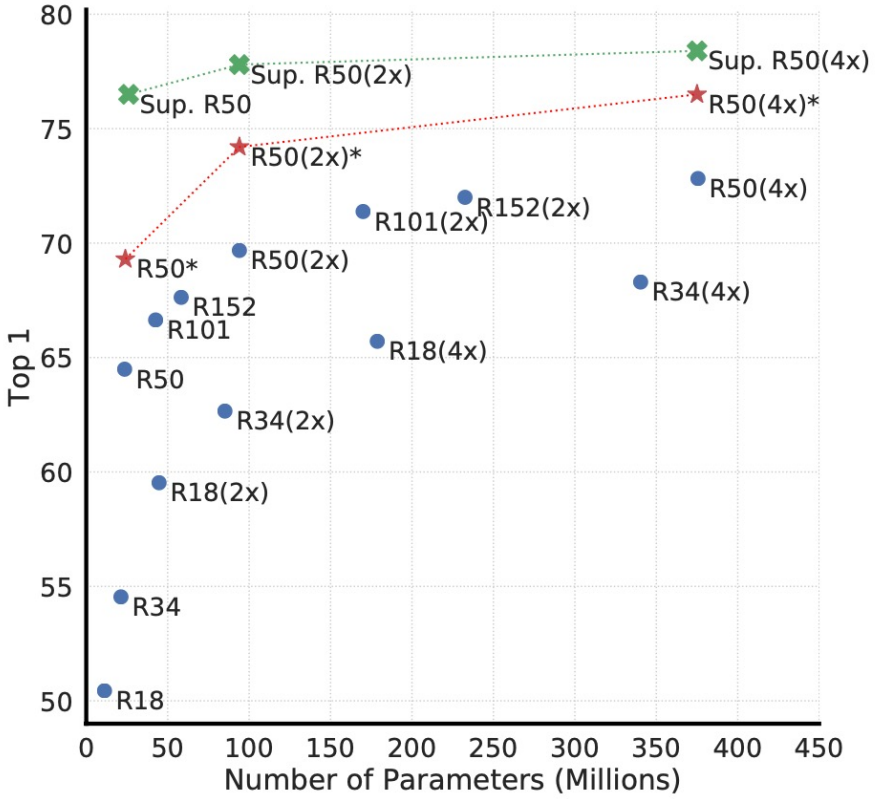


Figure 7. Linear evaluation of models with varied depth and width. Models in blue dots are ours trained for 100 epochs, models in red stars are ours trained for 1000 epochs, and models in green crosses are supervised ResNets trained for 90 epochs[7] (He et al., 2016).

Architectures for Encoder and Head

- A nonlinear projection head improves the representation quality of the layer before it

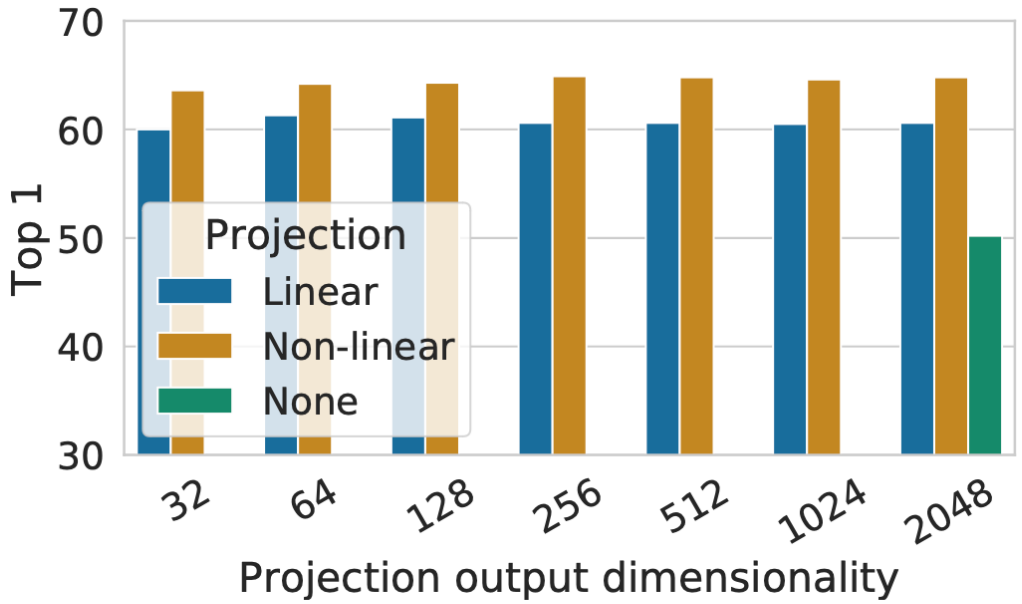


Figure 8. Linear evaluation of representations with different projection heads $g(\cdot)$ and various dimensions of $\boldsymbol{z} = g(\boldsymbol{h})$. The representation $\boldsymbol{h}$ (before projection) is 2048-dimensional here.

Loss Functions

- L2 normalization (i.e. cosine similarity) along with temperature effectively weights different examples, and an appropriate temperature can help the model learn from hard negatives.

ℓ_2 norm?	τ	Entropy	Contrastive acc.	Top 1
Yes	0.05	1.0	90.5	59.7
	0.1	4.5	87.8	64.4
	0.5	8.2	68.2	60.7
	1	8.3	59.1	58.0
No	10	0.5	91.7	57.2
	100	0.5	92.1	57.0

Table 5. Linear evaluation for models trained with different choices of ℓ_2 norm and temperature τ for NT-Xent loss. The contrastive distribution is over 4096 examples.

Contrastive learning benefits (more) from larger batch sizes and longer training

- When the number of training epochs is small (e.g. 100 epochs), larger batch sizes have a significant advantage.
- Training longer also provides more negative examples, improving the results.

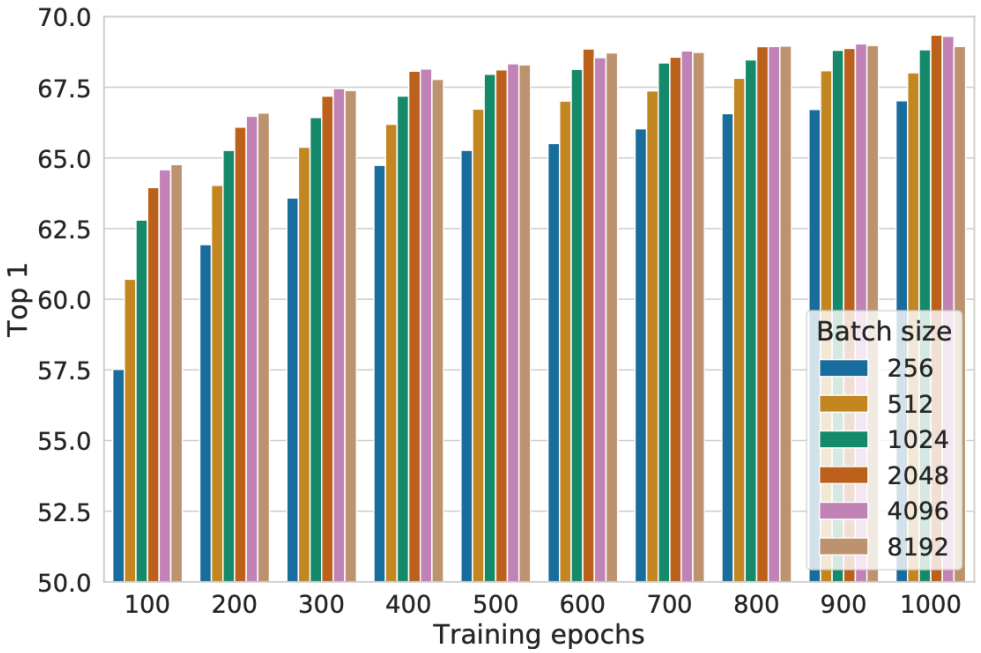


Figure 9. Linear evaluation models (ResNet-50) trained with different batch size and epochs. Each bar is a single run from scratch.[10]

Comparison with SOTAs

Method	Architecture	Param (M)	Top 1	Top 5
Methods using ResNet-50:				
Local Agg.	ResNet-50	24	60.2	-
MoCo	ResNet-50	24	60.6	-
PIRL	ResNet-50	24	63.6	-
CPC v2	ResNet-50	24	63.8	85.3
SimCLR (ours)	ResNet-50	24	**69.3**	**89.0**
Methods using other architectures:				
Rotation	RevNet-50 (4×)	86	55.4	-
BigBiGAN	RevNet-50 (4×)	86	61.3	81.9
AMDIM	Custom-ResNet	626	68.1	-
CMC	ResNet-50 (2×)	188	68.4	88.2
MoCo	ResNet-50 (4×)	375	68.6	-
CPC v2	ResNet-161 (∗)	305	71.5	90.1
SimCLR (ours)	ResNet-50 (2×)	94	74.2	92.0
SimCLR (ours)	ResNet-50 (4×)	375	**76.5**	**93.2**

Table 6. ImageNet accuracies of linear classifiers trained on representations learned with different self-supervised methods.

Comparison with SOTAs

- Semi-supervised learning

Method	Architecture	Label fraction 1%	10%
		Top 5	
Supervised baseline	ResNet-50	48.4	80.4
Methods using other label-propagation:			
Pseudo-label	ResNet-50	51.6	82.4
VAT+Entropy Min.	ResNet-50	47.0	83.4
UDA (w. RandAug)	ResNet-50	-	88.5
FixMatch (w. RandAug)	ResNet-50	-	89.1
S4L (Rot+VAT+En. M.)	ResNet-50 (4×)	-	91.2
Methods using representation learning only:			
InstDisc	ResNet-50	39.2	77.4
BigBiGAN	RevNet-50 (4×)	55.2	78.8
PIRL	ResNet-50	57.2	83.8
CPC v2	ResNet-161(∗)	77.9	91.2
SimCLR (ours)	ResNet-50	75.5	87.8
SimCLR (ours)	ResNet-50 (2×)	83.0	91.2
SimCLR (ours)	ResNet-50 (4×)	**85.8**	**92.6**

Table 7. ImageNet accuracy of models trained with few labels.

Conclusion

- We present a simple framework and its instantiation for contrastive visual representation learning.
- We carefully study its components, and show the effects of different design choices.

MoCo

Wei-Ta Chu

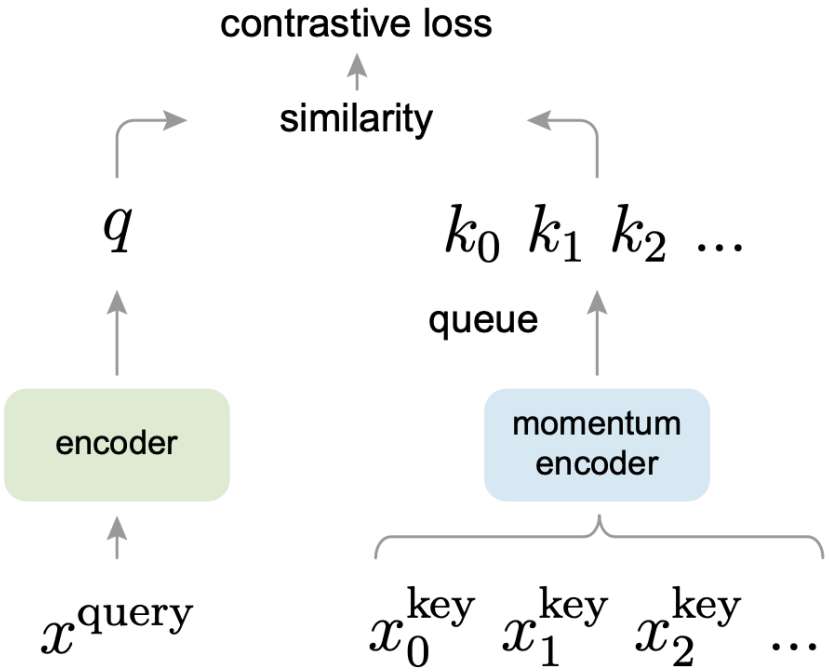
Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick, “Momentum Contrast for Unsupervised Visual Representation Learning,” CVPR, 2020.

Introduction

- Unsupervised learning methods can be thought of as building dynamic dictionaries.
- The “keys” (tokens) in the dictionary are sampled from data (e.g., images or patches) and are represented by an encoder network.
- An encoded “query” should be similar to its matching key and dissimilar to others.

Introduction

- We hypothesize that it is desirable to build dictionaries that are: (i) large and (ii) consistent as they evolve during training.
- We maintain the dictionary as a *queue* of data samples.



Contrastive Learning as Dictionary Look-up

- A contrastive loss is a function whose value is low when q is similar to its positive key k_+ and dissimilar to all other keys (considered negative keys for q).
- InfoNCE (Noise-Contrastive Estimation)

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)}$$

τ is a temperature hyper-parameter

Contrastive Learning as Dictionary Look-up

- The query representation is $q = f_q(x^q)$ where f_q is an encoder network and x^q is a query sample (likewise, $k = f_k(x^k)$).
- The input x^q and x^k can be images, patches, or context consisting a set of patches.
- The networks f_q and f_k can be identical, partially shared, or different.

Momentum Contrast

- Our hypothesis is that good features can be learned by a *large* dictionary that covers a rich set of negative samples, while the encoder for the dictionary keys is kept as *consistent* as possible despite its evolution.
- Maintaining the dictionary as a queue of data samples.
  - This allows us to reuse the encoded keys from the immediate preceding mini-batches.
  - Decouples the dictionary size from the mini-batch size. Our dictionary size can be much larger than a typical mini-batch size.

Momentum Contrast

- The samples in the dictionary are progressively replaced. The current mini-batch is enqueued to the dictionary, and the oldest mini-batch in the queue is removed.
- Using a queue can make the dictionary large, but it also makes it intractable to update the key encoder by back-propagation (the gradient should propagate to all samples in the queue).

Momentum Contrast

- Momentum update

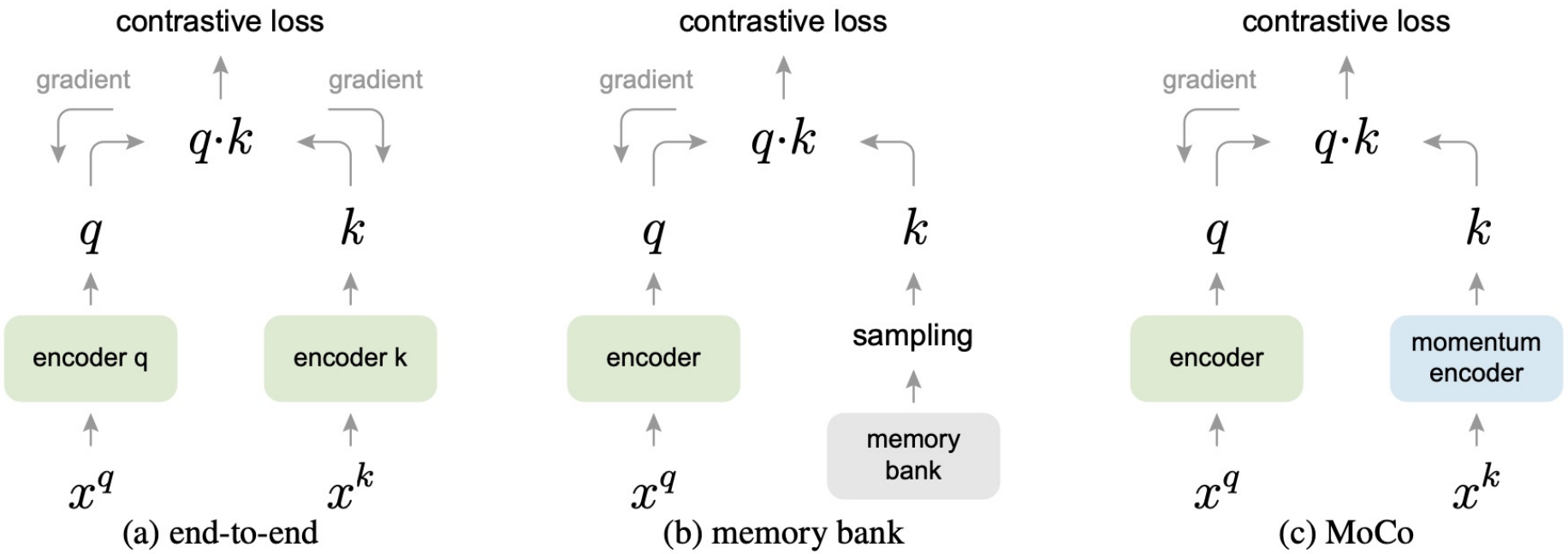
$$\theta_{\mathrm{k}} \leftarrow m\theta_{\mathrm{k}} + (1-m)\theta_{\mathrm{q}}$$

  m is a momentum coefficient.

- Only the parameters θ_{q} are updated by back-propagation. The momentum update makes θ_{k} evolve more smoothly than θ_{q}.
- As a result, though the keys in the queue are encoded by different encoders (in different mini-batches), the difference among these encoders can be made small.

Momentum Contrast

- Relations to previous mechanisms



Pretext Task

- We consider a query and a key as a positive pair if they originate from the same image, and otherwise as a negative sample pair.
- We adopt a ResNet as the encoder.
- The temperature τ is set as 0.07.
- Data augmentation
  - A 224×224-pixel crop is taken from a randomly resized image, and then undergoes random color jittering, random horizontal flip, and random grayscale conversion

Experiments

- Pre-training on IN-1M. Then we freeze the features and train a supervised linear classifier.

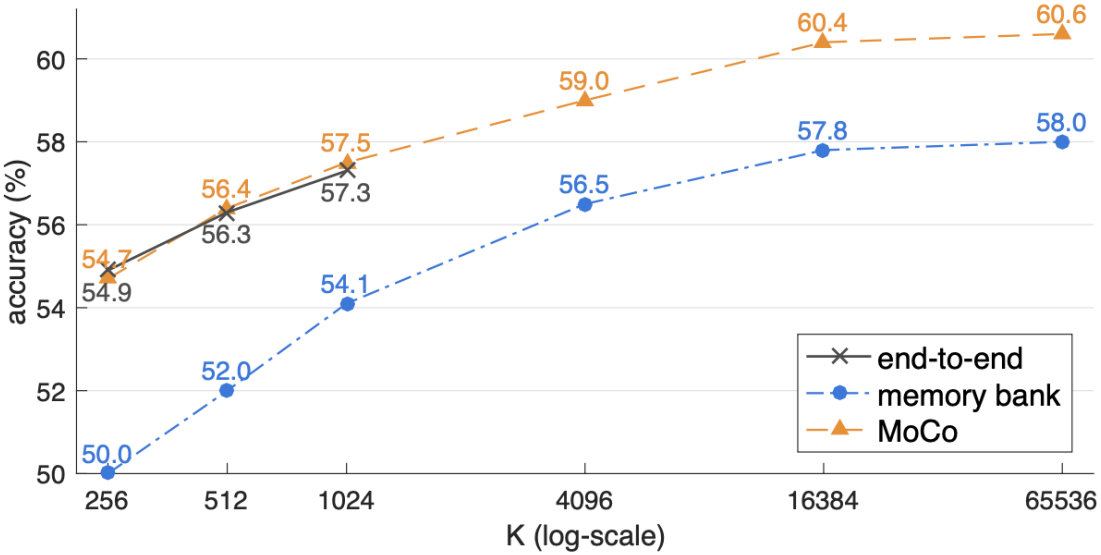


Figure 3. **Comparison of three contrastive loss mechanisms** under the ImageNet linear classification protocol. We adopt the same pretext task (Sec. 3.3) and only vary the contrastive loss mechanism (Figure 2). The number of negatives is K in memory bank and MoCo, and is $K-1$ in end-to-end (offset by one because the positive key is in the same mini-batch). The network is ResNet-50.

Experiments

- Ablation study of momentum

momentum m	0	0.9	0.99	0.999	0.9999
accuracy (%)	*fail*	55.2	57.8	59.0	58.9

- It performs reasonably well when m is in 0.99 ~ 0.9999, showing that a slowly progressing key encoder is beneficial.

Experiments

- Comparison with previous results

method	architecture	#params (M)	accuracy (%)
Exemplar [17]	R50w3×	211	46.0 [38]
RelativePosition [13]	R50w2×	94	51.4 [38]
Jigsaw [45]	R50w2×	94	44.6 [38]
Rotation [19]	Rv50w4×	86	55.4 [38]
Colorization [64]	R101*	28	39.6 [14]
DeepCluster [3]	VGG [53]	15	48.4 [4]
BigBiGAN [16]	R50	24	56.6
	Rv50w4×	86	61.3
methods based on contrastive learning follow:			
InstDisc [61]	R50	24	54.0
LocalAgg [66]	R50	24	58.8
CPC v1 [46]	R101*	28	48.7
CPC v2 [35]	$R170^{*}_{wider}$	303	65.9
CMC [56]	$R50_{L+ab}$	47	$64.1^{\dagger}$
	$R50w2\times_{L+ab}$	188	$68.4^{\dagger}$
AMDIM [2]	$AMDIM_{small}$	194	$63.5^{\dagger}$
	$AMDIM_{large}$	626	$68.1^{\dagger}$
MoCo	R50	24	60.6
	RX50	46	63.9
	R50w2×	94	65.4
	R50w4×	375	**68.6**

Transferring Features

pre-train	AP_{50}	AP	AP_{75}
random init.	64.4	37.9	38.6
super. IN-1M	81.4	54.0	59.1
MoCo IN-1M	81.1 (−0.3)	54.6 (+**0.6**)	59.9 (+**0.8**)
MoCo IG-1B	81.6 (+0.2)	55.5 (+**1.5**)	61.2 (+**2.1**)

(a) Faster R-CNN, R50-**dilated-C5**

pre-train	AP_{50}	AP	AP_{75}
random init.	60.2	33.8	33.1
super. IN-1M	81.3	53.5	58.8
MoCo IN-1M	81.5 (+0.2)	55.9 (+**2.4**)	62.6 (+**3.8**)
MoCo IG-1B	82.2 (+**0.9**)	57.2 (+**3.7**)	63.7 (+**4.9**)

(b) Faster R-CNN, R50-**C4**

Table 2. **Object detection fine-tuned on PASCAL VOC** `trainval07+12`. Evaluation is on `test2007`: AP_{50} (default VOC metric), AP (COCO-style), and AP_{75}, averaged over 5 trials. All are fine-tuned for 24k iterations (~23 epochs). In the brackets are the gaps to the ImageNet supervised pre-training counterpart. In green are the gaps of at least +**0.5** point.

Conclusion

- In sum, MoCo can outperform its ImageNet supervised pre-training counterpart in 7 detection or segmentation tasks.

MoCo v2

Wei-Ta Chu

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He, “Improved Baselines with Momentum Contrastive Learning,” arXiv:2003.04297, 2020.

Introduction

- Two improvements used in SimCLR, namely, an MLP projection head and stronger data augmentation, are orthogonal to the frameworks of MoCo and SimCLR.
- When used with MoCo they lead to better image classification and object detection transfer learning results.

Introduction

- SimCLR needs 4k~8k batches, which require TPU support. “MoCo v2” baselines can run on a typical 8-GPU machine and achieve better results than SimCLR.

Experiments

case	unsup. pre-train MLP	aug+	cos	epochs	ImageNet acc.	VOC detection AP_{50}	AP	AP_{75}
supervised					76.5	81.3	53.5	58.8
MoCo v1				200	60.6	81.5	55.9	62.6
(a)	✓			200	66.2	82.0	56.4	62.6
(b)		✓		200	63.4	82.2	56.8	63.2
(c)	✓	✓		200	67.3	**82.5**	57.2	63.9
(d)	✓	✓	✓	200	67.5	82.4	57.0	63.6
(e)	✓	✓	✓	**800**	**71.1**	**82.5**	**57.4**	**64.0**

Table 1. **Ablation of MoCo baselines**, evaluated by ResNet-50 for (i) ImageNet linear classification, and (ii) fine-tuning VOC object detection (mean of 5 trials). "**MLP**": with an MLP head; "**aug+**": with extra blur augmentation; "**cos**": cosine learning rate schedule.

Experiments

case	unsup. pre-train MLP	aug+	cos	epochs	batch	ImageNet acc.
MoCo v1 [6]				200	256	60.6
SimCLR [2]	✓	✓	✓	200	256	61.9
SimCLR [2]	✓	✓	✓	200	8192	66.6
MoCo v2	✓	✓	✓	200	256	**67.5**
*results of **longer** unsupervised training follow:*						
SimCLR [2]	✓	✓	✓	1000	4096	69.3
MoCo v2	✓	✓	✓	800	256	**71.1**

Table 2. **MoCo *vs*. SimCLR**: ImageNet linear classifier accuracy (**ResNet-50, 1-crop 224×224**), trained on features from unsupervised pre-training. "aug+" in SimCLR includes blur and stronger color distortion. SimCLR ablations are from Fig. 9 in [2] (we thank the authors for providing the numerical results).

Experiments

- Tables 2 and 3 suggest that large batches are not necessary for good accuracy, and state-of-the-art results can be made more accessible. The improvements we investigate require only a few lines of code changes to MoCo v1.

mechanism	batch	memory / GPU	time / 200-ep.
MoCo	256	**5.0G**	**53 hrs**
end-to-end	256	7.4G	65 hrs
end-to-end	4096	93.0G†	n/a

Table 3. **Memory and time cost** in 8 V100 16G GPUs, implemented in PyTorch. †: based on our estimation.

MoCo v3

Wei-Ta Chu

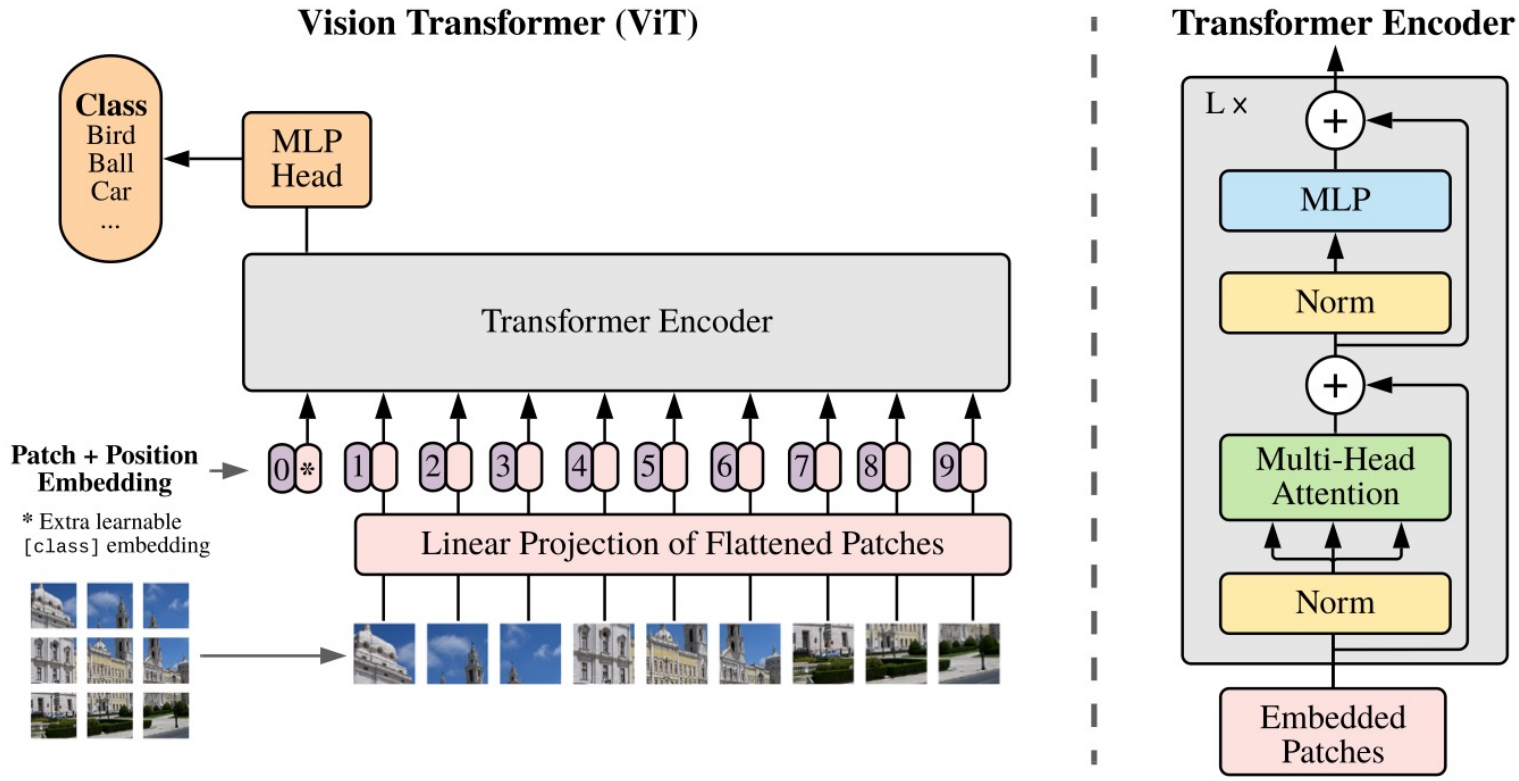
Xinlei Chen, Saining Xie, and Kaiming He, “An Empirical Study of Training Self-Supervised Vision Transformers,” ICCV, 2021.

Introduction

- Investigate the effects of several fundamental components for training self-supervised ViT: the batch size, learning rate, and optimizer.

framework	model	params	acc. (%)
linear probing:			
iGPT [9]	iGPT-L	1362M	69.0
iGPT [9]	iGPT-XL	6801M	72.0
MoCo v3	ViT-B	86M	76.7
MoCo v3	ViT-L	304M	77.6
MoCo v3	ViT-H	632M	78.1
MoCo v3	ViT-BN-H	632M	79.1
MoCo v3	ViT-BN-L/7	304M	**81.0**
end-to-end fine-tuning:			
masked patch pred. [16]	ViT-B	86M	79.9†
MoCo v3	ViT-B	86M	83.2
MoCo v3	ViT-L	304M	**84.1**

Vision Transformer (ViT)



Dosovitskiy et al., An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, ICLR 2021.

MoCo v3

- We take two crops for each image under random data augmentation. They are encoded by two encoders, f_q and f_k, with output vectors q (query) and k (key).
- InfoNCE loss

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k^+ / \tau)}{\exp(q \cdot k^+ / \tau) + \sum_{k^-} \exp(q \cdot k^- / \tau)}$$

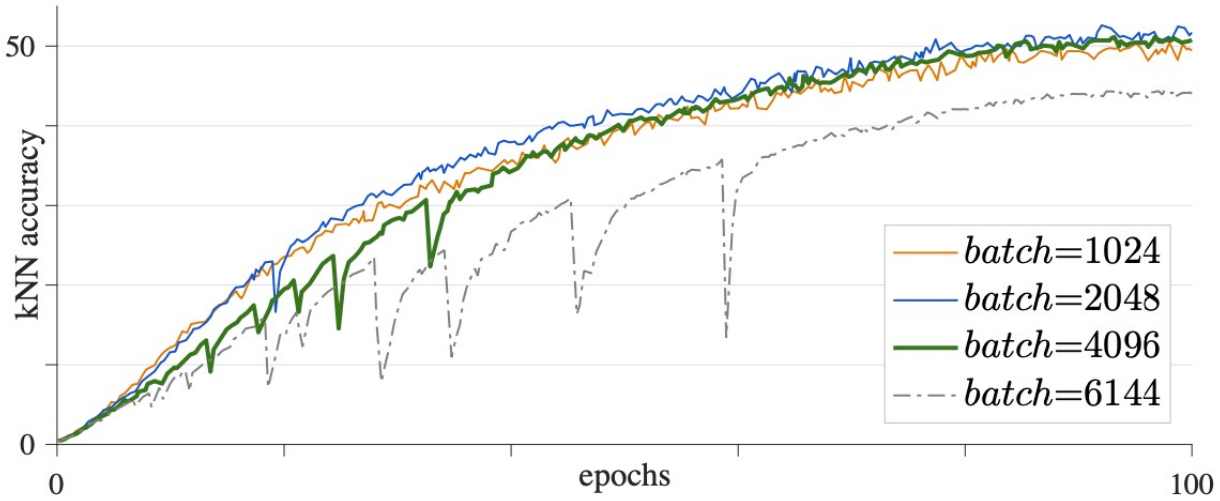
MoCo v3

R50, 800-ep	MoCo v2 [12]	MoCo v2+ [13]	MoCo v3
linear acc.	71.1	72.2	**73.8**

- We abandon the memory queue, which we find has diminishing gain if the batch is sufficiently large (e.g., 4096).
- With this simplification, the contrastive loss in (1) can be implemented by a few lines of code.
- The encoder f_q consists of a backbone (e.g., ResNet, ViT), a projection head, and an extra prediction head; the encoder f_k has the backbone and projection head, but not the prediction head. f_k is updated by the moving average of f_q, excluding the prediction head.

Stability of Self-Supervised ViT Training

- Batch size: The curve of a 4k batch becomes noticeably unstable.

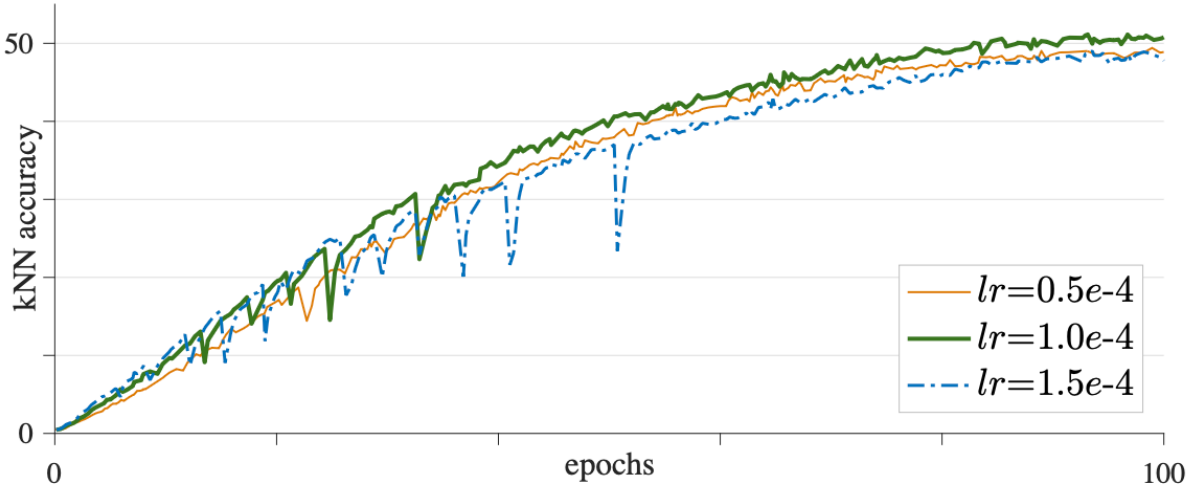


batch	1024	2048	4096	6144
linear acc.	71.5	72.6	72.2	69.7

Figure 1. **Training curves of different batch sizes** (MoCo v3, ViT-B/16, 100-epoch ImageNet, AdamW, lr=1.0e-4).

Stability of Self-Supervised ViT Training

- Learning rate. When lr is smaller, the training is more stable, but it is prone to under-fitting.

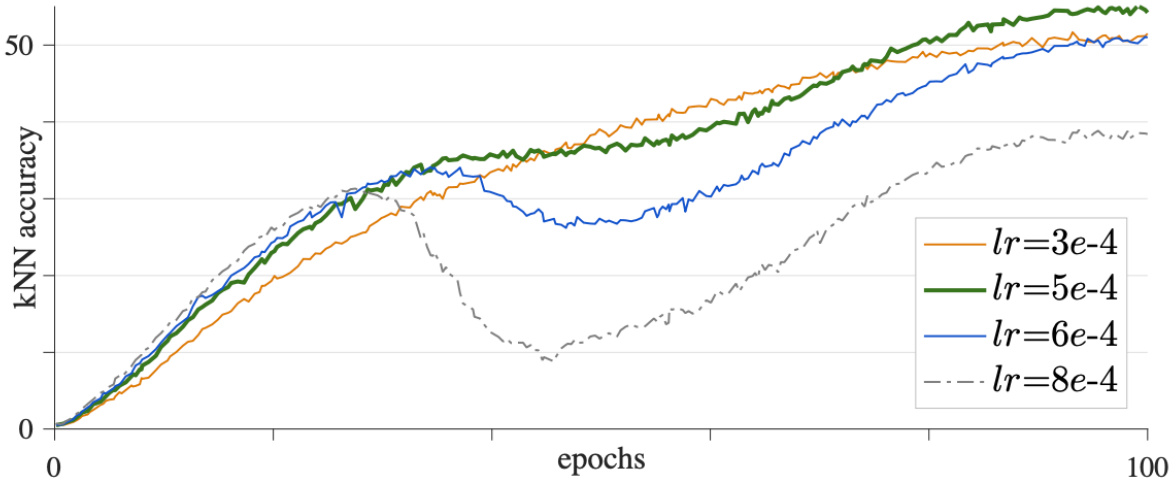


lr, $\times$1e-4	0.5	1.0	1.5
linear acc.	70.4	72.2	71.7

Figure 2. **Training curves of different learning rates** (MoCo v3, ViT-B/16, 100-epoch ImageNet, AdamW, batch 4096).

Stability of Self-Supervised ViT Training

- Optimizer. The accuracy drops rapidly when lr is larger than the optimal value.

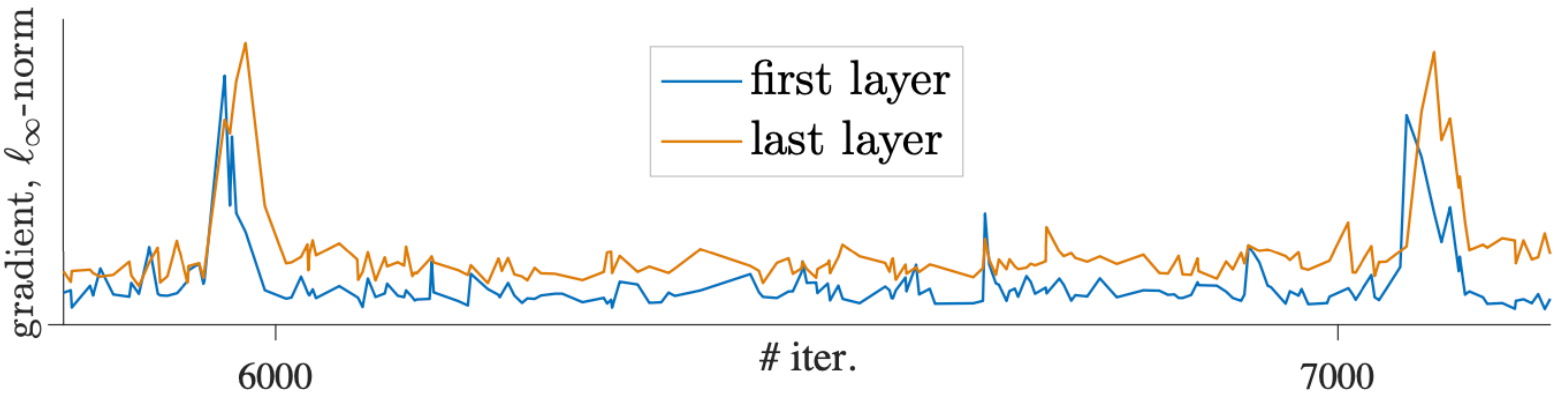


lr, ×1e-4	3.0	5.0	6.0	8.0
linear acc.	71.6	72.5	70.9	66.5

Figure 3. **Training curves of LAMB optimizer** (MoCo v3, ViT-B/16, 100-epoch ImageNet, wd=1e-3, batch 4096).

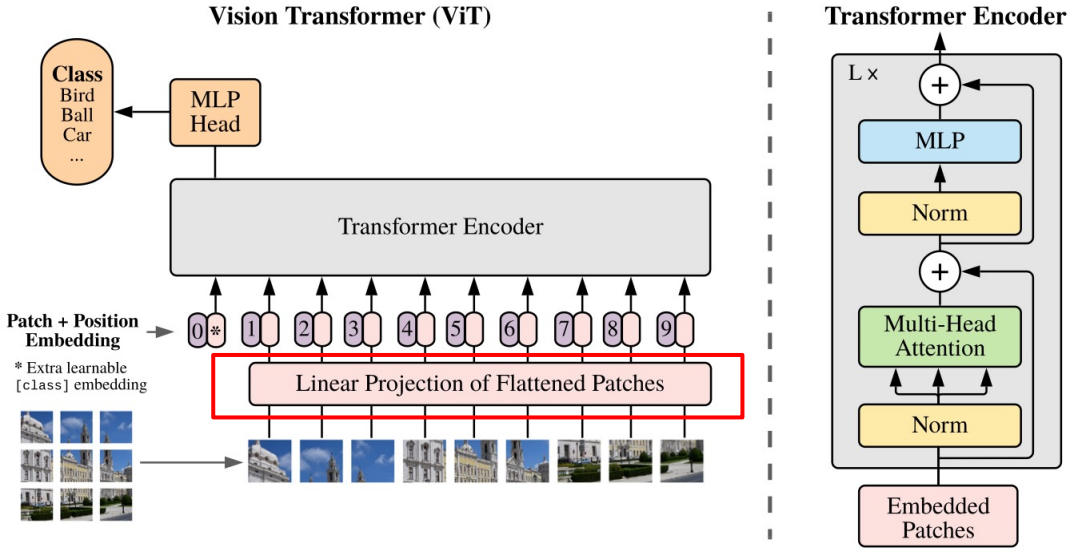
Trick for Improving Stability

- A sudden change of gradients causes a “dip” in the training curve.
- The gradient spikes happen earlier in the first layer, and are delayed by couples of iterations in the last layers.



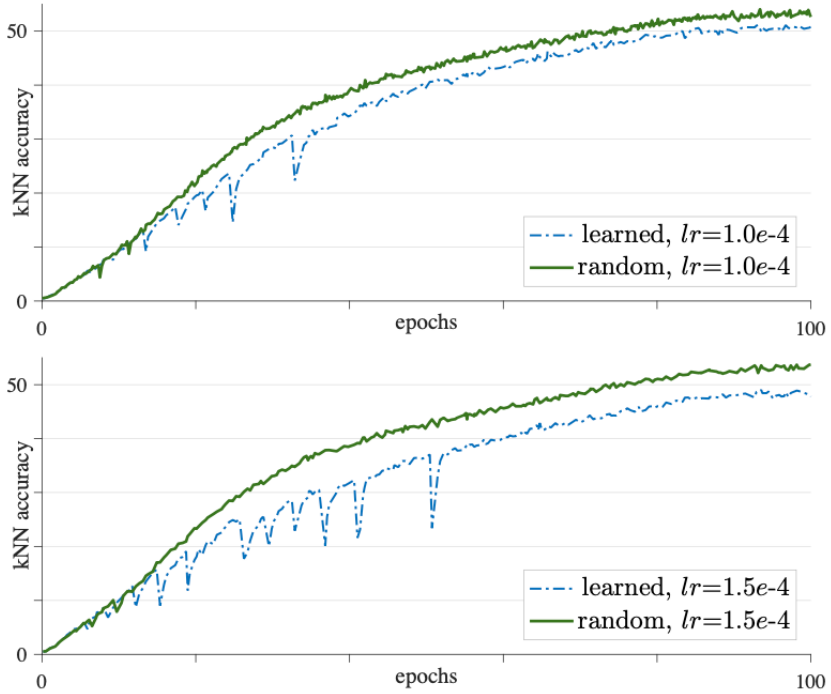
Trick for Improving Stability

- We hypothesize that the instability happens earlier in the shallower layers.
- We explore freezing the patch projection layer during training. We use a fixed ***random patch projection*** layer to embed the patches, which is not learned.



Trick for Improving Stability

- We use a fixed ***random patch projection*** layer to embed the patches, which is not learned.



lr, $\times 10^{-4}$	0.5	1.0	1.5
learned patch proj.	70.4	72.2	71.7
random patch proj.	**70.8**	**72.8**	**73.4**

Experimental Results

model	MoCo v3	SimCLR	BYOL	SwAV
R-50, 800-ep	73.8	70.4	**74.3**	71.8
ViT-S, 300-ep	**72.5**	69.0	71.0	67.1
ViT-B, 300-ep	**76.5**	73.9	73.9	71.6

Table 4. **ViT-S/16 and ViT-B/16 in different self-supervised learning frameworks** (ImageNet, linear probing). R-50 results of other frameworks are from the improved implementation in [13]. For fair comparisons, all are pre-trained with two 224×224 crops for each image (multi-crop training [7] could improve results, which is beyond the focus of this work).

Experimental Results

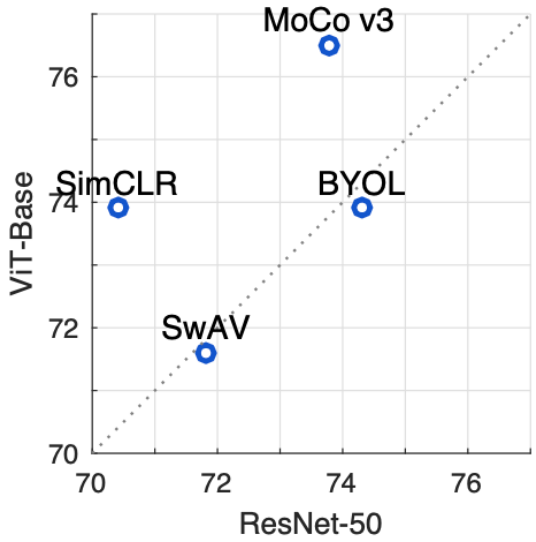


Figure 7. Different self-supervised learning frameworks perform differently between R-50 [21] (x-axis) and ViT-B [16] (y-axis). The numbers are ImageNet linear probing accuracy from Table 4.

Experimental Results