SimCLR:

A Simple Framework for Contrastive Learning of Visual Representations

Ting Chen, Simon Kornblith, Mohammad Norouzi, Geoffrey Hinton

arXiv:2002.05709v1 Google Research, Brain Team.

Sungman, Cho.

Introduction

Contributions

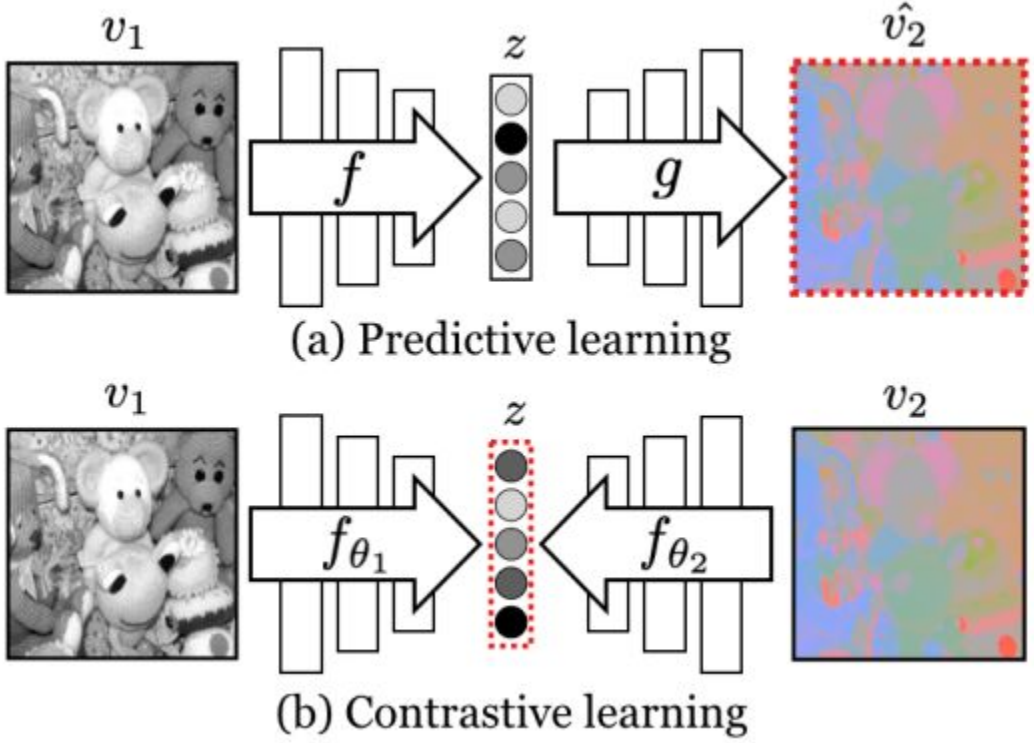
- Show that composition of **data augmentations plays a critical role** in defining effective predictive tasks.

- Introducing a **learnable nonlinear transformation** between the representation and the contrastive loss substantially improves the quality of the learned representations.

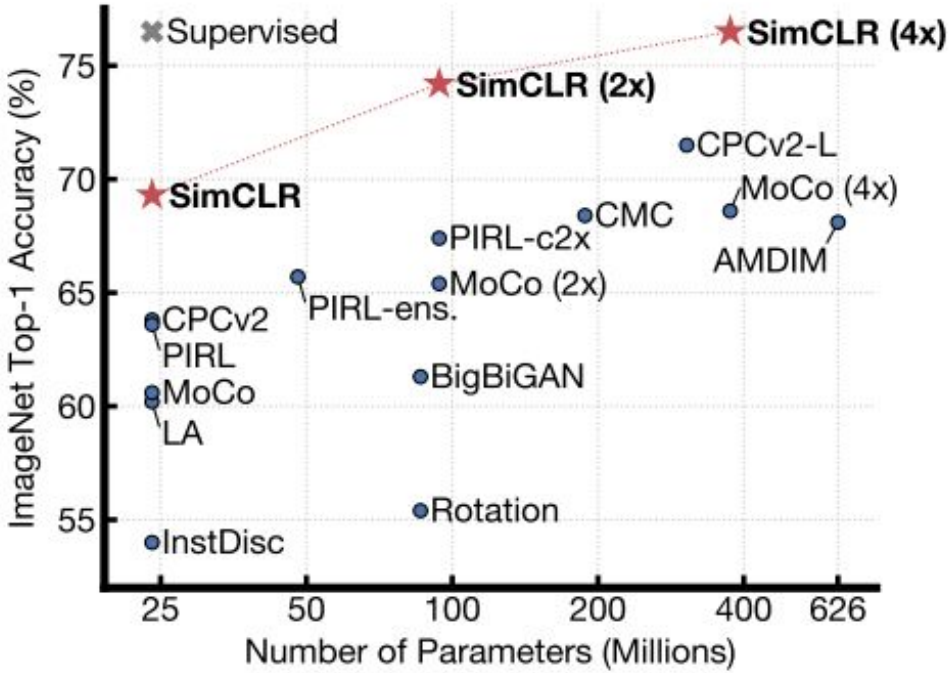
- Contrastive learning **benefits from larger batch sizes and more training steps** compared to supervised learning.

- Representation learning with **contrastive cross entropy loss benefits from normalized embeddings** and an appropriately adjusted temperature parameter.

Appendix : Contrastive Learning.



Contributions



Achieves 76.5% top-1 acc. , which is a 7% relative improvement over previous SOTA

Challenges

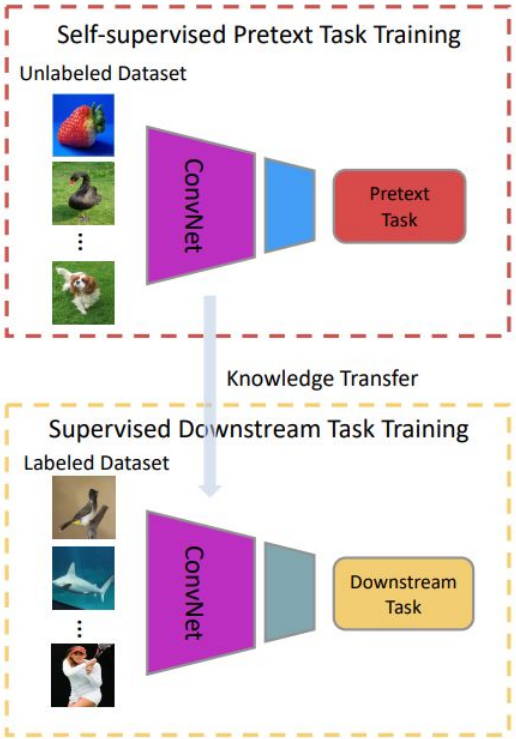
- **Two mainstream : Generative / Discriminative**

- Pixel-level generation is **computationally expensive** and **may not be necessary for representation learning**. (Generative)

- Many approaches have relied on heuristics to design pretext tasks, which could limit the generality of the learned representations. (Discriminative)

- Contrastive learning in the latent space have recently shown great promise.

Appendix : Pretext Task, Downstream Task



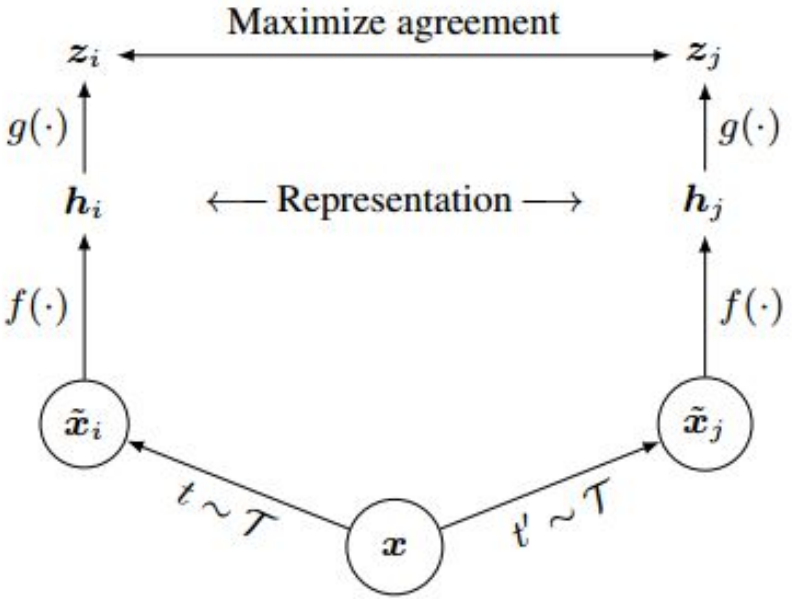
- **Pretext Task :** pre-designed tasks for networks to solve, and visual features are learned by learning objective functions of pretext tasks.

- **Downstream Task :** computer vision applications that are used to evaluate the quality of features learned by self-supervised learning.

2. Method

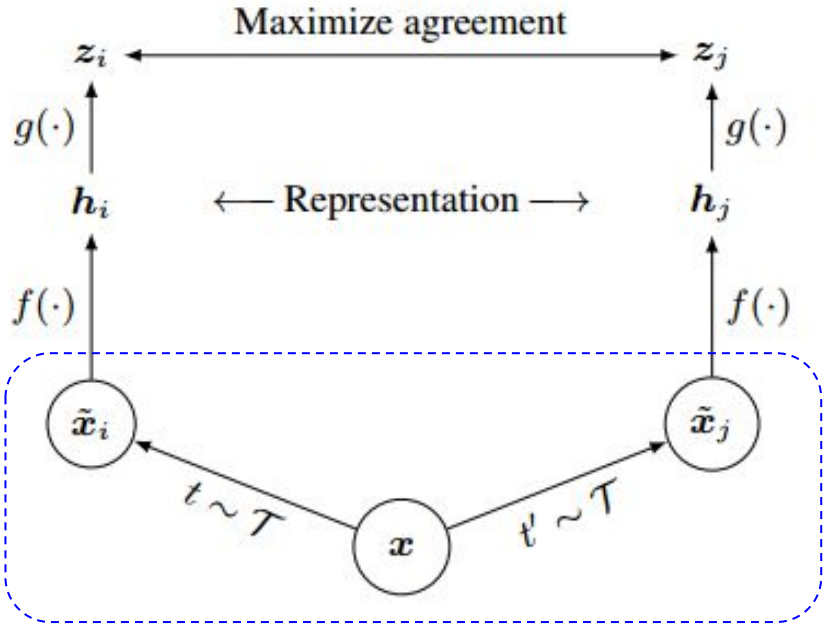
2.1. The Contrastive Learning Framework

SimCLR : Framework



SimCLR learns representations by maximizing agreement between differently augmented views of the same data example via a contrastive loss in the latent space

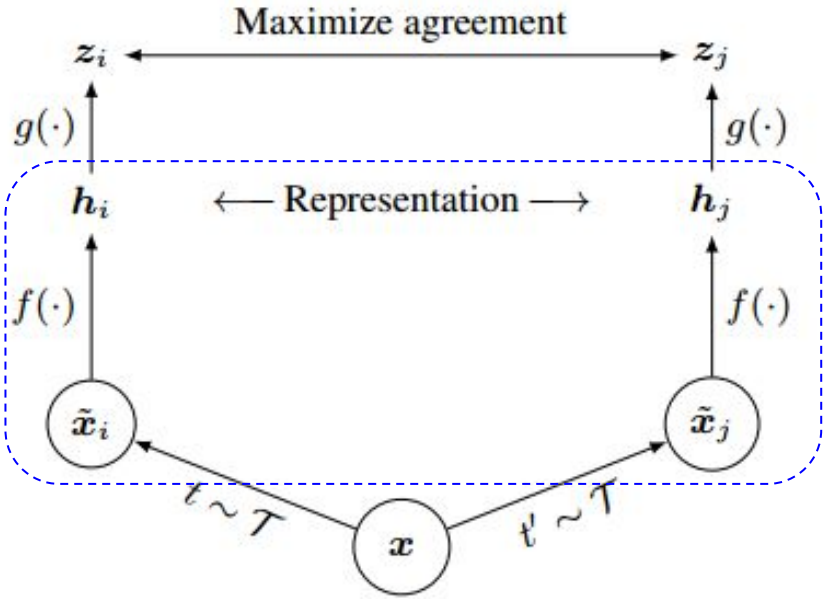
SimCLR : Data Augmentation



A stochastic data augmentation of the same data example (positive pair) :

random cropping, random color distortions, random Gaussian blur.

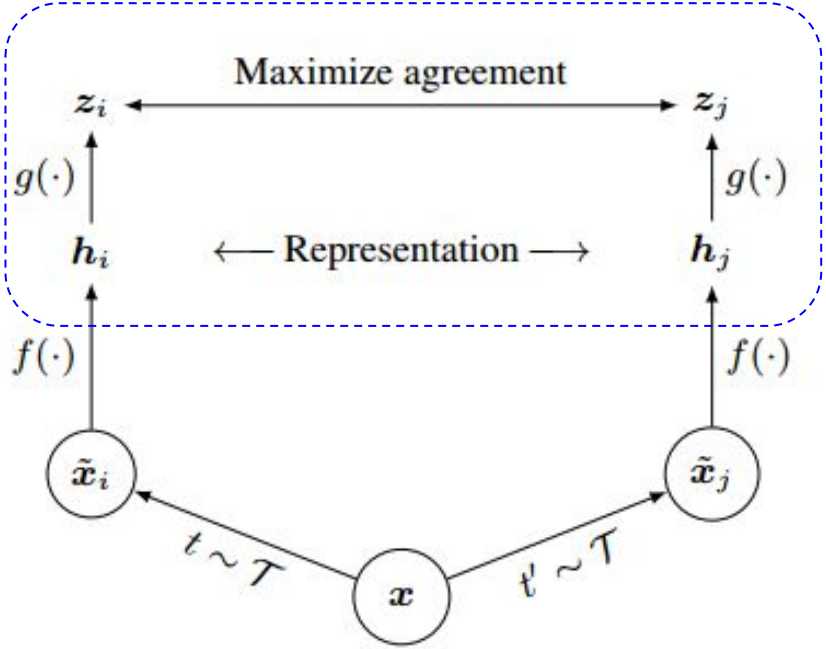
SimCLR : Base Encoder



Base encoder :

Extracts representation vectors from augmented data examples. (used ResNet)

SimCLR : Projection Head

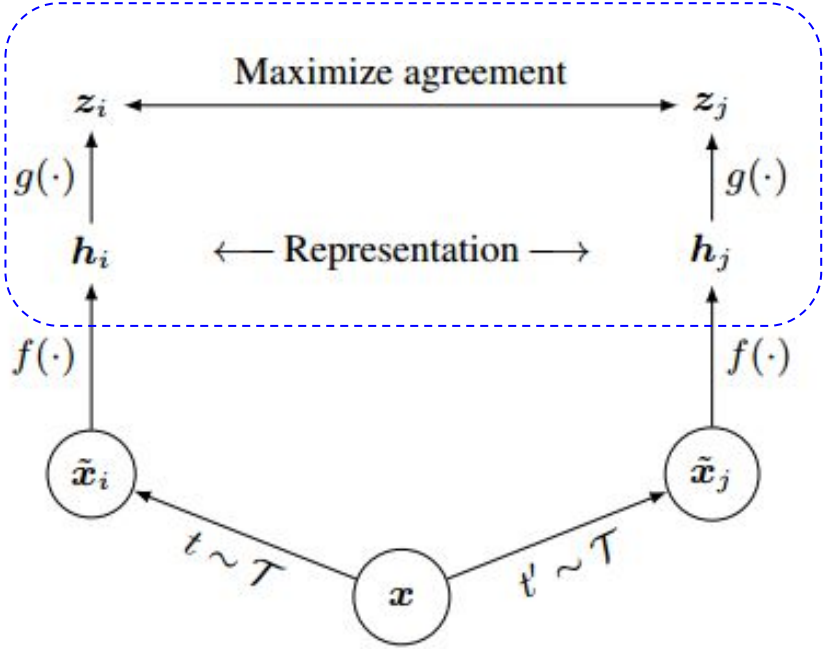


Projection head :

Maps representations to the space where contrastive loss is applied.

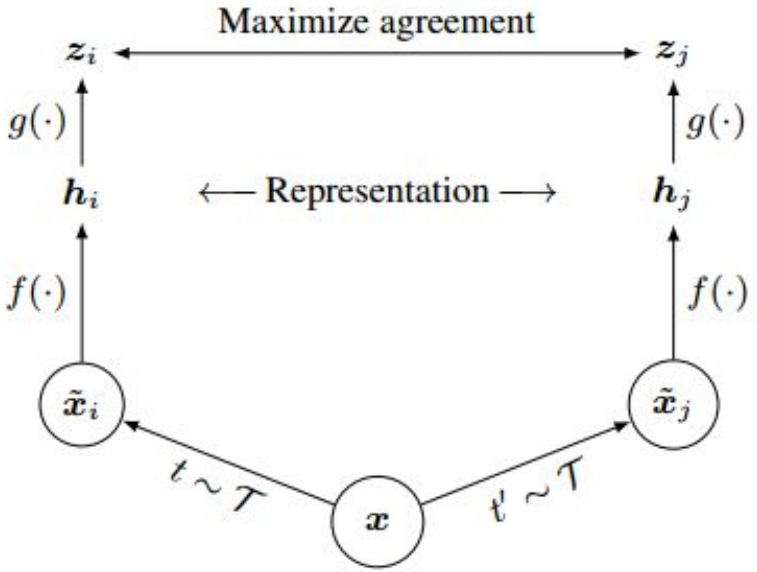
$$z_i = g(h_i) = W^{(2)}\sigma(W^{(1)}h_i)$$

SimCLR : Contrastive Learning



A *contrastive loss function* defined for a contrastive prediction task. Given a set $\{\tilde{\boldsymbol{x}}_k\}$ including a positive pair of examples $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_j$, the *contrastive prediction task* aims to identify $\tilde{\boldsymbol{x}}_j$ in $\{\tilde{\boldsymbol{x}}_k\}_{k \neq i}$ for a given $\tilde{\boldsymbol{x}}_i$.

SimCLR : Algorithm



Algorithm 1 SimCLR's main learning algorithm.

input: batch size N, temperature τ, structure of $f, g, \mathcal{T}$.
for sampled minibatch $\{\boldsymbol{x}_k\}_{k=1}^{N}$ **do**
  for all $k \in \{1, \ldots, N\}$ **do**
    draw two augmentation functions $t \sim \mathcal{T}$, $t' \sim \mathcal{T}$
    # the first augmentation
    $\tilde{\boldsymbol{x}}_{2k-1} = t(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k-1} = f(\tilde{\boldsymbol{x}}_{2k-1})$ # representation
    $\boldsymbol{z}_{2k-1} = g(\boldsymbol{h}_{2k-1})$ # projection
    # the second augmentation
    $\tilde{\boldsymbol{x}}_{2k} = t'(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k} = f(\tilde{\boldsymbol{x}}_{2k})$ # representation
    $\boldsymbol{z}_{2k} = g(\boldsymbol{h}_{2k})$ # projection
  end for
  for all $i \in \{1, \ldots, 2N\}$ and $j \in \{1, \ldots, 2N\}$ **do**
    $s_{i,j} = \boldsymbol{z}_i^{\top} \boldsymbol{z}_j / (\tau \|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|)$ # pairwise similarity
  end for
  define $\ell(i, j)$ **as** $\ell(i, j) = -\log \frac{\exp(s_{i,j})}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k})}$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network f

2. Method

2.2. Training with Large Batch Size

Training Batch Size

- We **do not train model with a memory bank.** Instead, we vary the training batch size N from 256 to 8192.

  (A batch size 8192 gives us 16382 negative examples per positive pair both augmentation views.)

- Training with **large batch size may be unstable** when using **standard SGD/Momentum with linear learning rate scaling.**

  → LARS optimizer

Training Batch Size

- We **do not train model with a memory bank.** Instead, we vary the training batch size N from 256 to 8192.

  (A batch size 8192 gives us 16382 negative examples per positive pair both augmentation views.)

- Training with **large batch size may be unstable** when using **standard SGD/Momentum with linear learning rate scaling.**

[1]With 128 TPU v3 cores, it takes ~1.5 hours to train our ResNet-50 with a batch size of 4096 for 100 epochs.

?????

Global BN

- In distributed training with data parallelism, the **BN mean and variance are typically aggregated locally per device**.

- In our contrastive learning, as **positive pairs are computed in the same device**, the model can exploit the local information leakage to improve prediction accuracy without improving representations.

- Address this issue by **aggregating BN mean and variance** over all devices.

2. Method

2.3. Evaluation Protocol

Dataset and Metrics

- Unsupervised pretraining : ImageNet ILSVRC-2012 dataset
- Additional pretraining : CIFAR-10
- Test the pretrained result on a wide range of datasets for transfer learning.
- To evaluate the learned representations, we follow the widely used linear evaluation protocol.

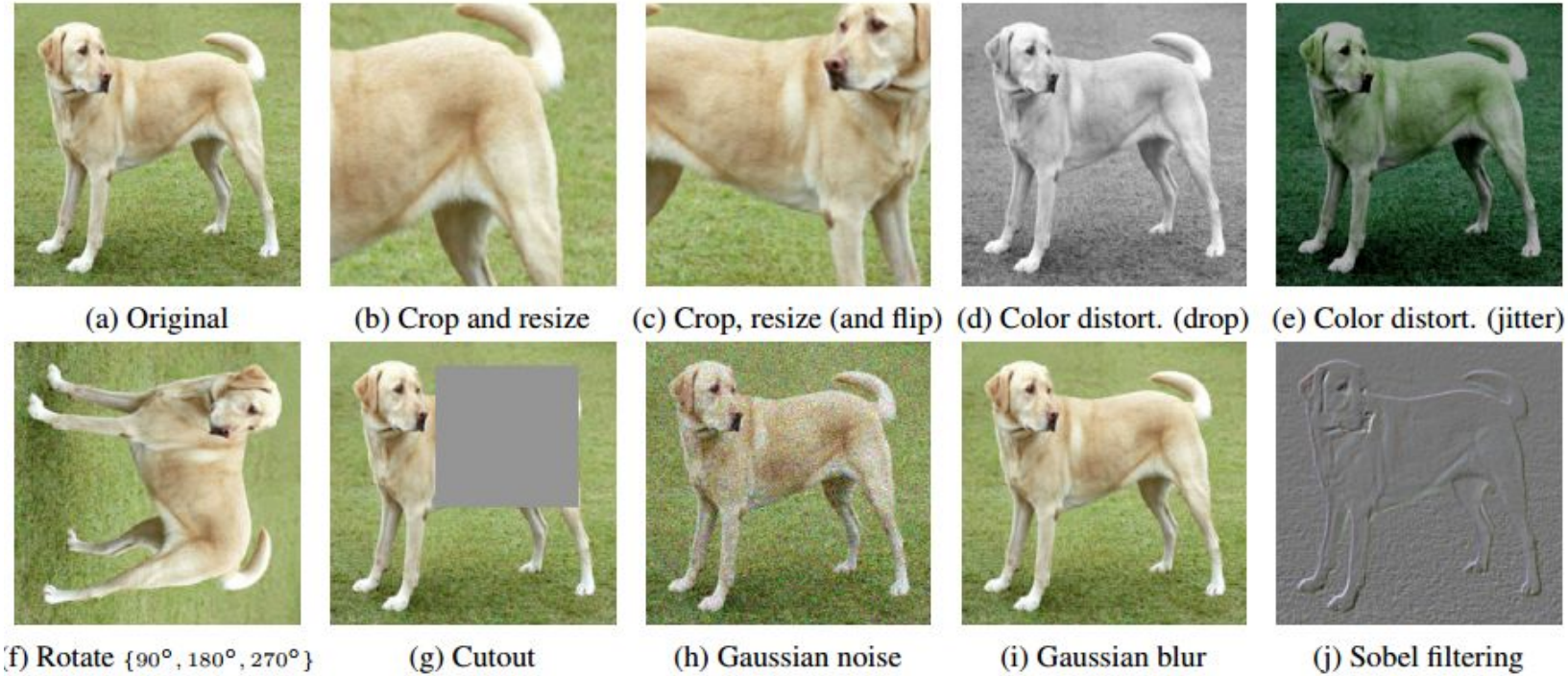
Default setting.

- Loss : NT-Xent
- Optimizer : LARS (lr= 0.3 x batch_size/256)
- Weight decay : 1e-06
- Batch size : 4096
- Epochs : 100
- Warmup for the first 10 epochs, decay the learning rate with the cosine decay schedule without restarts

3. DA for CRL

Data Augmentation for Contrastive Representation Learning

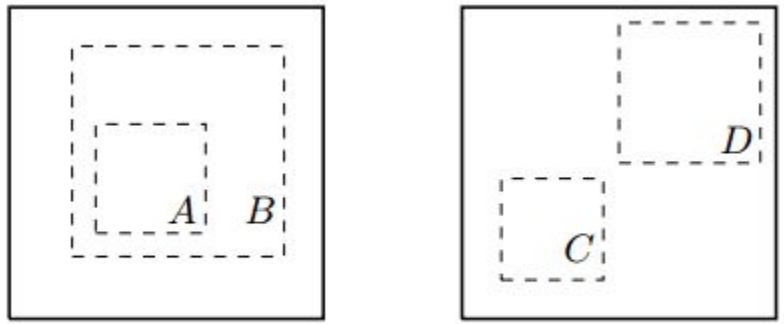
Data augmentation



The augmentation policy used to train : random crop, color distortion, Gaussian blur

Data augmentation : random crop

- Many existing approaches define contrastive prediction tasks by changing the architecture
    - global-to-local view prediction (Bachman et al. 2019)
    - Neighboring view prediction (Henaff et al. 2019)
- **By performing simple random cropping, we can avoid complexity.**



(a) Global and local views. (b) Adjacent views.

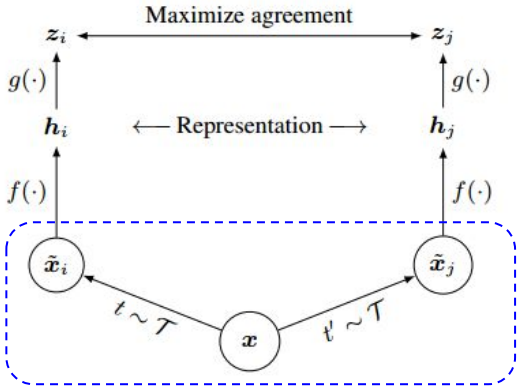
Composition of data augmentation

- Linear evaluation under individual or composition of data augmentations.
- Random cropping & random color distortion : Best

Composition of data augmentation

- Since **ImageNet images are of different sizes**, we **always apply crop and resize** images, which makes **it difficult to study other augmentations in the absence of cropping.**

- To eliminate this confound, we consider an asymmetric data transformation setting for this ablation.



If one branch is transformed,
the other branch must be identity

- Asymmetric data augmentation hurts the performance. Nonetheless, this setup should not substantively change.

CL need stronger DA than Supervised.

Methods	Color distortion strength					AutoAug
	1/8	1/4	1/2	1	1 (+Blur)	
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

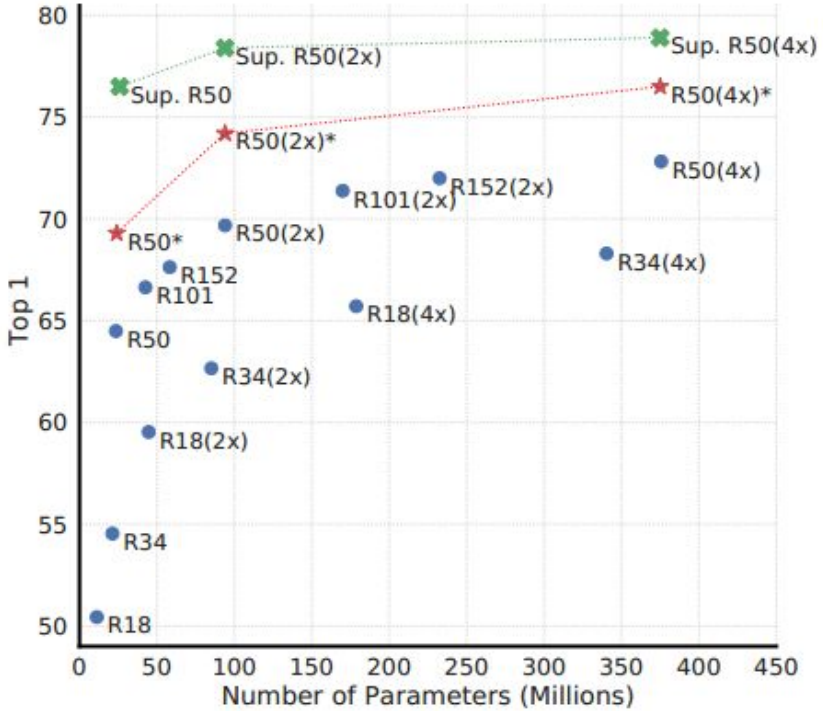
- **AutoAugment** (Cubuk et al.,2019) **does not work better than simple cropping + (stronger) color distortion in SimCLR.**
- When training **supervised models** with the same set of augmentations, we observe that **stronger color augmentation does not improve or even hurts their performance.**

CL : Contrastive Learning
DA : Data Augmentation

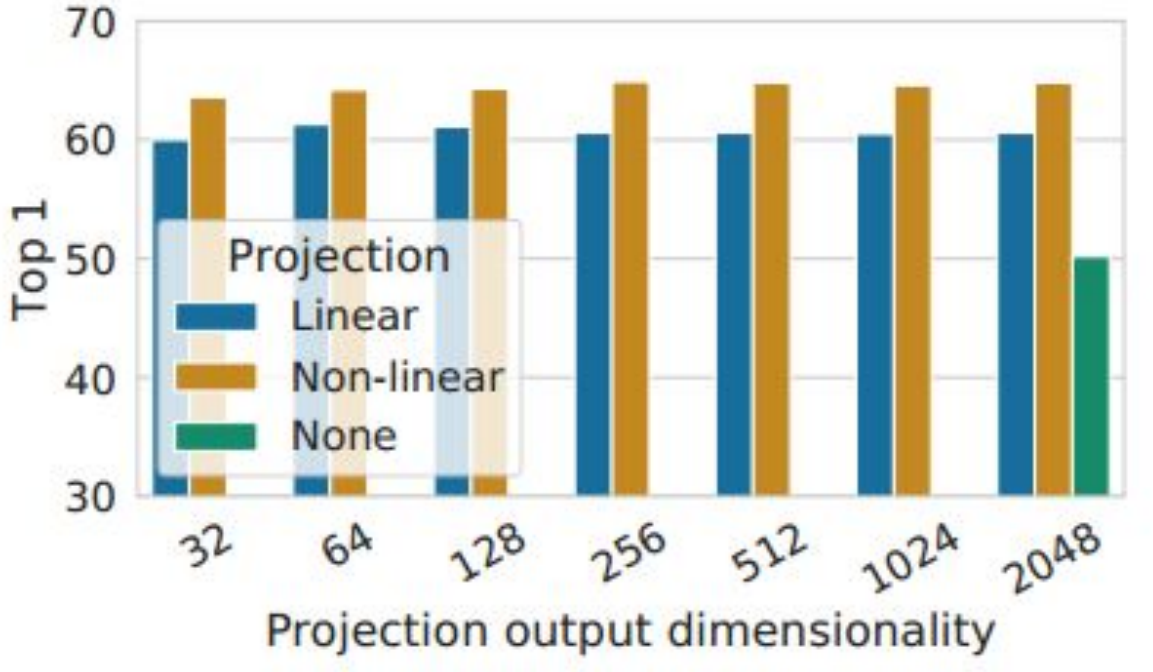
4. Architectures for Encoder and Head

UCL benefits (more) from bigger models.

- **Increasing depth and width both improve performance.**
- **Unsupervised learning benefits more** from bigger models than its supervised counterpart.



Nonlinear Projection Head



Nonlinear Projection Head : Information Loss

What to predict?	Random guess	Representation h	Representation $g(h)$
Color vs grayscale	80	99.3	97.4
Rotation	25	67.6	25.6
Orig. vs corrupted	50	99.5	59.6
Orig. vs Sobel filtered	50	96.6	56.3

- Even when nonlinear projection is used, the layer before the projection head, h, is still much better (>10%) than the layer after, z=g(h), which shows that the **hidden layer before the projection is a better representation than the layer after**

Nonlinear Projection Head : Information Loss

What to predict?	Random guess	Representation $\boldsymbol{h}$	Representation $g(\boldsymbol{h})$
Color vs grayscale	80	99.3	97.4
Rotation	25	67.6	25.6
Orig. vs corrupted	50	99.5	59.6
Orig. vs Sobel filtered	50	96.6	56.3

- In particular, ***z=g(h)*** **is trained to be invariant to data transformation**. Thus *g* can **remove information that may be useful for the downstream task**, such as the color or orientation of objects.

- By leveraging the nonlinear transformation *g(*)*, more information can be formed and maintained in *h.*

5. Loss Functions and Batch Size

NT-Xent : Hard Negative Mining

Name	Negative loss function	Gradient w.r.t. $\boldsymbol{u}$
NT-Xent	$\boldsymbol{u}^T\boldsymbol{v}^+/\tau - \log \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^+,\boldsymbol{v}^-\}} \exp(\boldsymbol{u}^T\boldsymbol{v}/\tau)$	$(1 - \frac{\exp(\boldsymbol{u}^T\boldsymbol{v}^+/\tau)}{Z(\boldsymbol{u})})/\tau\boldsymbol{v}^+ - \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^+,\boldsymbol{v}^-\}} \frac{\exp(\boldsymbol{u}^T\boldsymbol{v}/\tau)}{Z(\boldsymbol{u})}/\tau\boldsymbol{v}$
NT-Logistic	$\log \sigma(\boldsymbol{u}^T\boldsymbol{v}^+/\tau) + \log \sigma(-\boldsymbol{u}^T\boldsymbol{v}^-/\tau)$	$(\sigma(-\boldsymbol{u}^T\boldsymbol{v}^+/\tau))/\tau\boldsymbol{v}^+ - \sigma(\boldsymbol{u}^T\boldsymbol{v}^-/\tau)/\tau\boldsymbol{v}^-$
Margin Triplet	$-\max(\boldsymbol{u}^T\boldsymbol{v}^- - \boldsymbol{u}^T\boldsymbol{v}^+ + m, 0)$	$\boldsymbol{v}^+ - \boldsymbol{v}^-$ if $\boldsymbol{u}^T\boldsymbol{v}^+ - \boldsymbol{u}^T\boldsymbol{v}^- < m$ else $\mathbf{0}$

Table 2. Negative loss functions and their gradients. All input vectors, i.e. $\boldsymbol{u}, \boldsymbol{v}^+, \boldsymbol{v}^-$, are ℓ_2 normalized. NT-Xent is an abbreviation for "Normalized Temperature-scaled Cross Entropy". Different loss functions impose different weightings of positive and negative examples.

- **L2 normalization along with temperature effectively weights different examples**, and appropriate temperature can **help the model learn from hard negatives**
- Unlike cross-entropy, **other objective functions do not weigh the negatives** by their relative hardness.

NT-Xent : Hard Negative Mining

Margin	NT-Logi.	Margin (sh)	NT-Logi.(sh)	NT-Xent
50.9	51.6	57.5	57.9	63.9

Table 4. Linear evaluation (top-1) for models trained with different loss functions. "sh" means using semi-hard negative mining.

- **L2 normalization along with temperature effectively weights different examples**, and appropriate temperature can **help the model learn from hard negatives**
- Unlike cross-entropy, **other objective functions do not weigh the negatives** by their relative hardness.

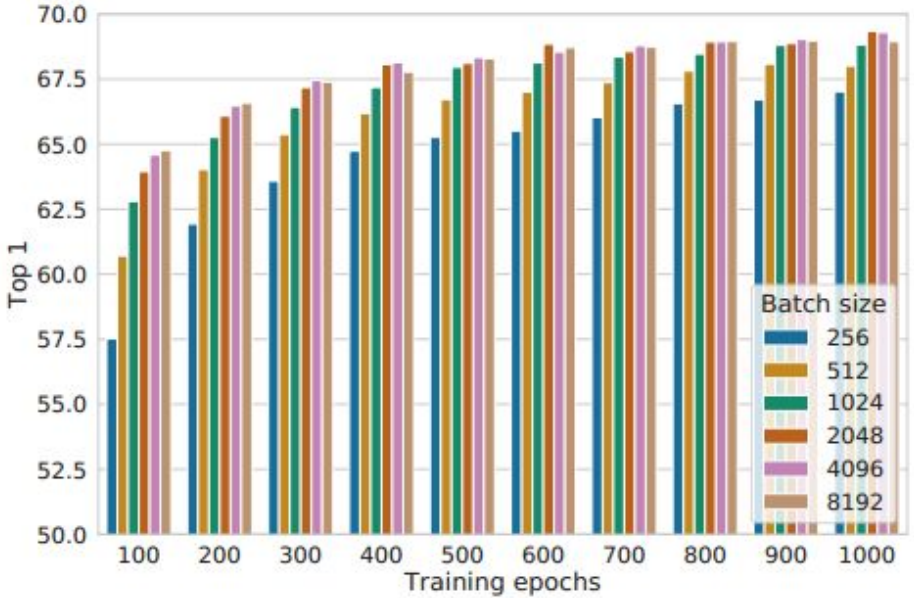
NT-Xent : L2 normalization

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.

- Without normalization and proper temperature scaling, performance is significantly worse.
- **Without L2 norm**alization, the **contrastive task accuracy is higher**, but the resulting **representation is worse** under linear evaluation.

CL benefits from larger batch, longer training



- Larger batch sizes provide more negative examples, facilitating convergence
- Training longer also provides more negative examples, improving the results.

6. Comparison with SOTA

ImageNet acc.

Method	Architecture	Param.	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.

Method	Architecture	Label fraction (Top 5) 1%	10%
Methods using other label-propagation:			
Pseudo-label	ResNet50	51.6	82.4
VAT+Entropy Min.	ResNet50	47.0	83.4
UDA (w. RandAug)	ResNet50	-	88.5
FixMatch (w. RandAug)	ResNet50	-	89.1
S4L (Rot+VAT+En. M.)	ResNet50 (4×)	-	91.2
Methods using representation learning only:			
InstDisc	ResNet50	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.

Transfer Learning Performance

	Food	CIFAR10	CIFAR100	Birdsnap	SUN397	Cars	Aircraft	VOC2007	DTD	Pets	Caltech-101	Flowers
Linear evaluation:												
SimCLR (ours)	**76.9**	**95.3**	80.2	48.4	**65.9**	60.0	61.2	**84.2**	**78.9**	89.2	**93.9**	**95.0**
Supervised	75.2	**95.7**	**81.2**	**56.4**	64.9	**68.8**	**63.8**	83.8	**78.7**	**92.3**	**94.1**	94.2
Fine-tuned:												
SimCLR (ours)	**89.4**	**98.6**	**89.0**	**78.2**	**68.1**	**92.1**	**87.0**	**86.6**	**77.8**	92.1	**94.1**	97.6
Supervised	88.7	98.3	**88.7**	**77.8**	67.0	91.4	**88.0**	86.5	**78.8**	**93.2**	**94.2**	**98.0**
Random init	88.3	96.0	81.9	**77.0**	53.7	91.3	84.8	69.4	64.1	82.7	72.5	92.5

Table 8. Comparison of transfer learning performance of our self-supervised approach with supervised baselines across 12 natural image classification datasets, for ResNet-50 $(4\times)$ models pretrained on ImageNet. Results not significantly worse than the best ($p > 0.05$, permutation test) are shown in bold. See Appendix B.6 for experimental details and results with standard ResNet-50.

6. Conclusion

Conclusion

- Present a **simple framework** and its instantiation for contrastive **visual representation learning.**
- **Complexity of some previous methods for self-supervised learning is not necessary** to achieve good performance.
- The strength of this simple framework suggests that, despite a recent surge in interest, Self-supervised learning remains undervalued.