A Simple Framework for Contrastive Learning of Visual Representations

University of Freiburg

Seminar on Deep Learning for Bio-Medical Data Analysis

October 5, 2020

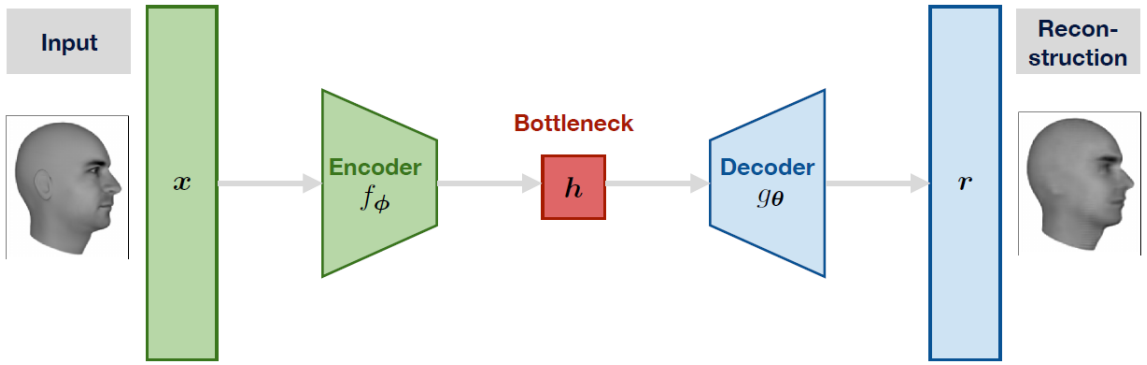
Julia Mertesdorf

Motivation: Self-supervised Learning

- DL requires large-scale data
- Avoid expensive & time-consuming data annotations
- Exploit large amounts of available unlabeled data
- **Goal**: Learn good & generic visual feature representations from unlabeled data
- Self-supervised learning: automatically generate labels from data

Self-Supervised Learning

- Generative approaches: The Autoencoder
- Goal: Reconstruct the original input (label = input image)
- Bottleneck prevents simply copying the input



Encode input into lower-dimensional representation h

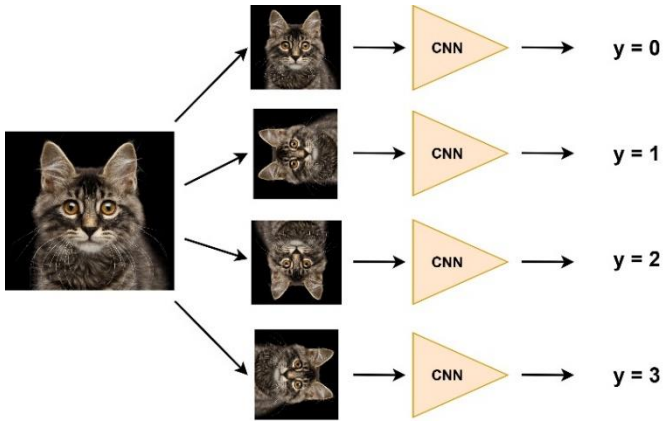
Reconstruct original image from representation h

DL Lecture Freiburg (WS 19/20)

- Drawbacks:
  - Computationally expensive
  - Too much focus on details instead of high-level semantic features

Self-Supervised Learning

- Discriminative approaches: Manually designed pretext tasks
- Common tasks
  - Rotation angle prediction: Classification with 4 classes



  - Colorize greyscale image: Predict color channels
  - Jigsaw puzzle: Predict relative ordering of image patches

- Pretext tasks should ensure that semantic features are learned
- Drawback: Limited generality of the learned representations

Contrastive Learning: Idea

- Learn features that make 2 images similar / dissimilar
- High-level features instead of pixel-level details
- Formally: Learn an encoder f such that:

$$\text{score}(f(x), f(x^+)) >> \text{score}(f(x), f(x^-))$$

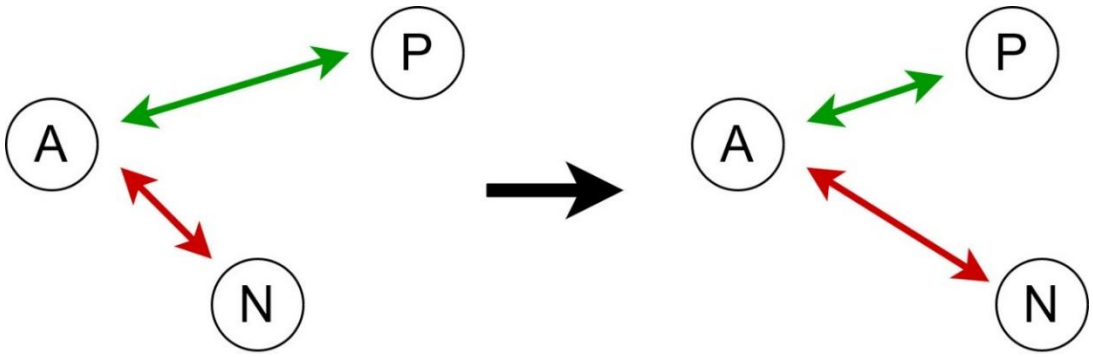
- Representations are learned by contrasting positive & negative data samples

→ Learn more diverse & generic features

Margin triplet loss

- Triplets of training data samples:
  Anchor sample (A), Positive sample (P) & Negative sample (N)
- Idea: pull A & P closer together. Push A & N away from each other

$$L_{triplet} = max(0,\ \ d(A,P) - d(A,N) + m)$$

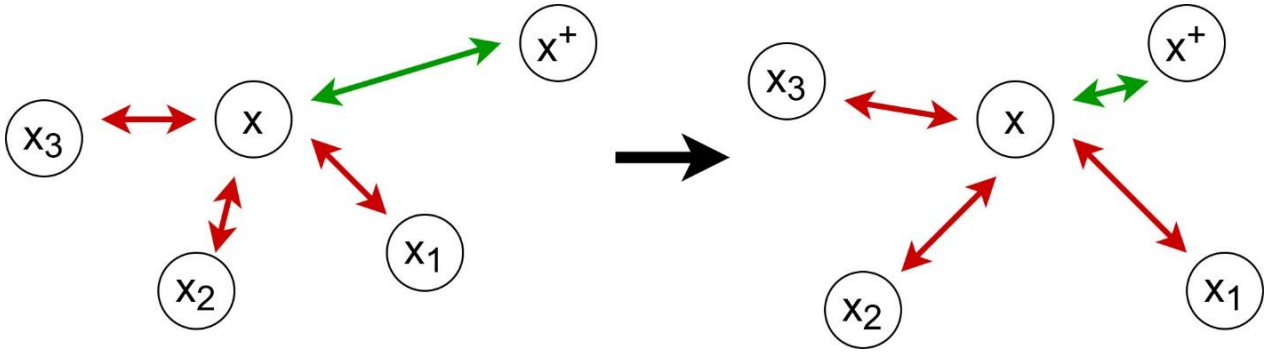


- Points with distance > *m* do not contribute to loss
- Problem: requires good triplet sampling strategy
  → High impact on performance

Noise contrastive estimation loss

- Generalization of the triplet loss
- Pull positive pair closer together in representation space
- Push all negative samples away simultaneously

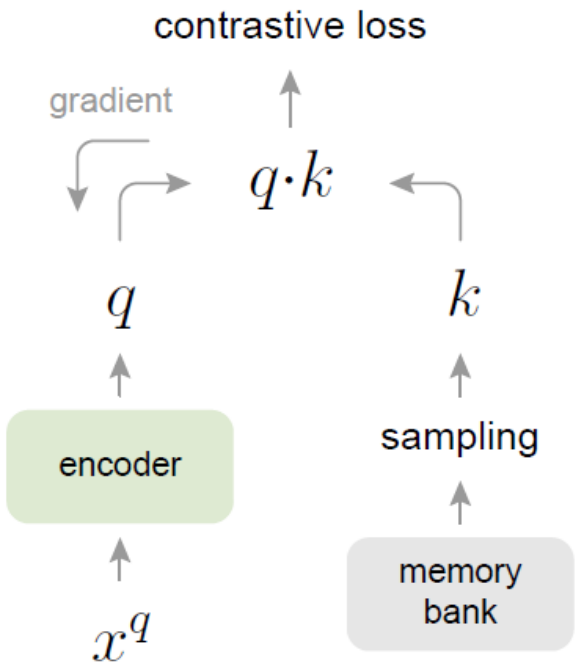
$$\mathcal{L}_N = -\mathbb{E}_X \left[\log \frac{\exp\left(f(x)^T f\left(x^+\right)\right)}{\exp\left(f(x)^T f\left(x^+\right)\right) + \sum_{j=1}^{N-1} \exp\left(f(x)^T f\left(x_j\right)\right)} \right]$$



- Contrastive power increases with more negatives
- Advantage: resolves problem of negative mining

Background: Instance Discrimination

- Idea: Dictionary lookup problem
- Each data sample is its own class
- Encode features for each image
- Store representations in a memory bank
- Find best matching representation („key“) to a given encoded image („query“)
- Drawbacks:
  - Outdated representations
  - High memory costs



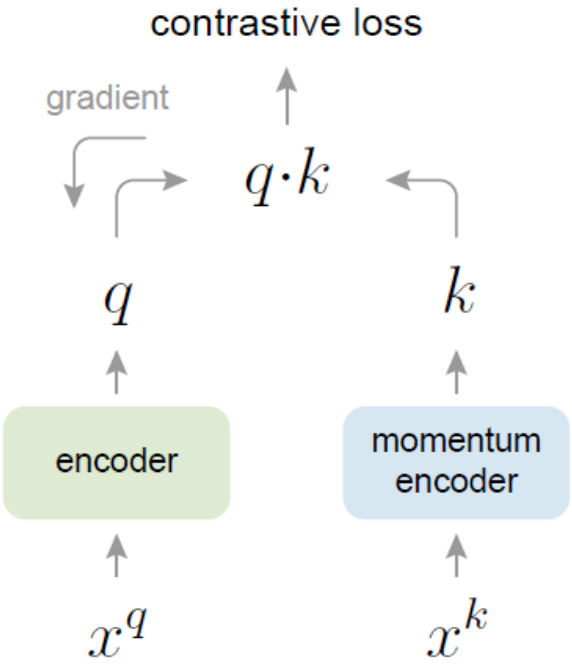
He, K. et al. (2020)

Background: Momentum Contrast (MoCo)

- Positive keys: data augmentation of current query
- Uses a queue instead of a memory bank
- Only query encoder is trained
- Key encoder parameters are updated by a momentum update:

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

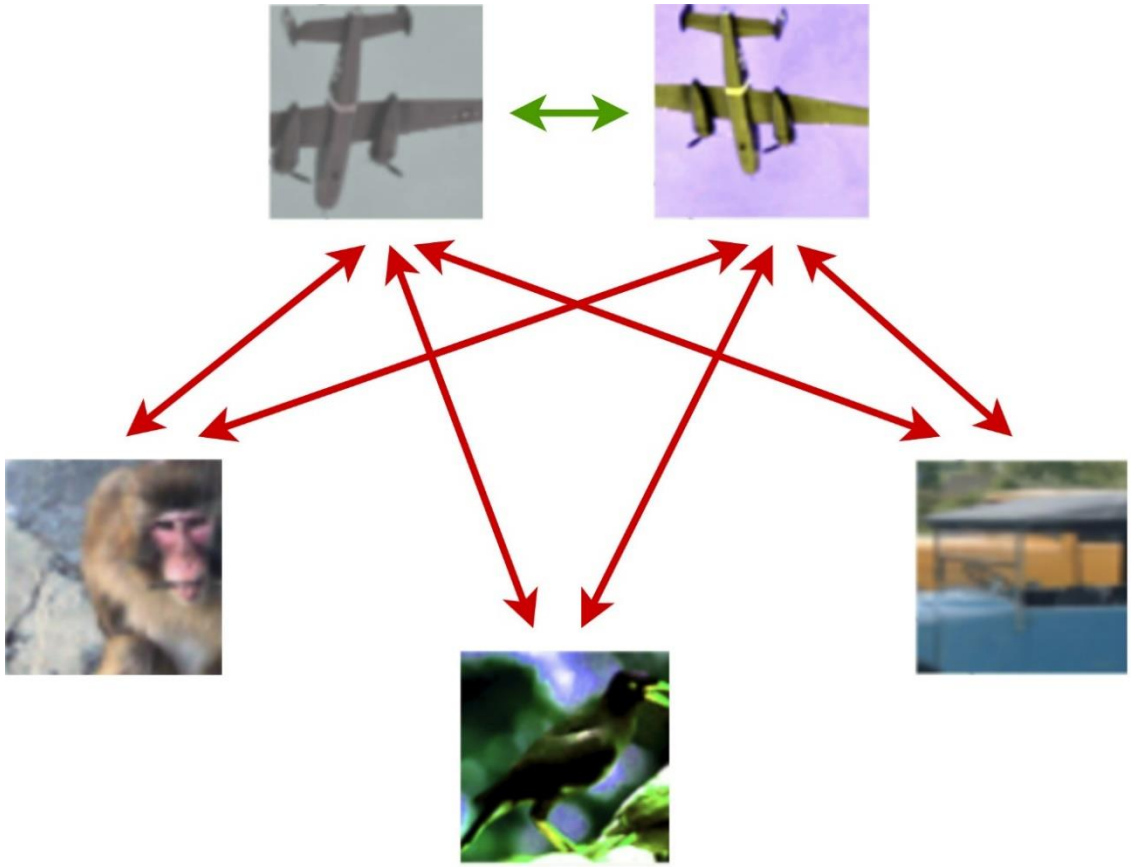
- Advantages:
  - Memory-efficient
  - Consistent representations



He, K. et al. (2020)

SimCLR framework: Idea

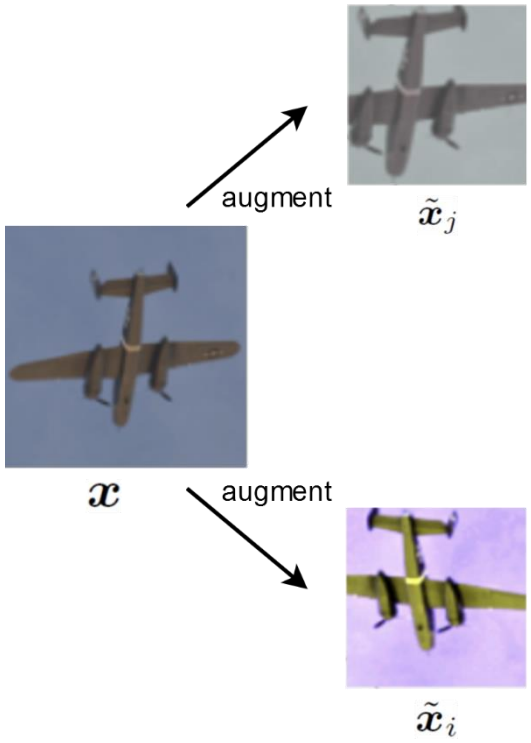
Maximize agreement between differently augmented views of the same data sample via a contrastive loss



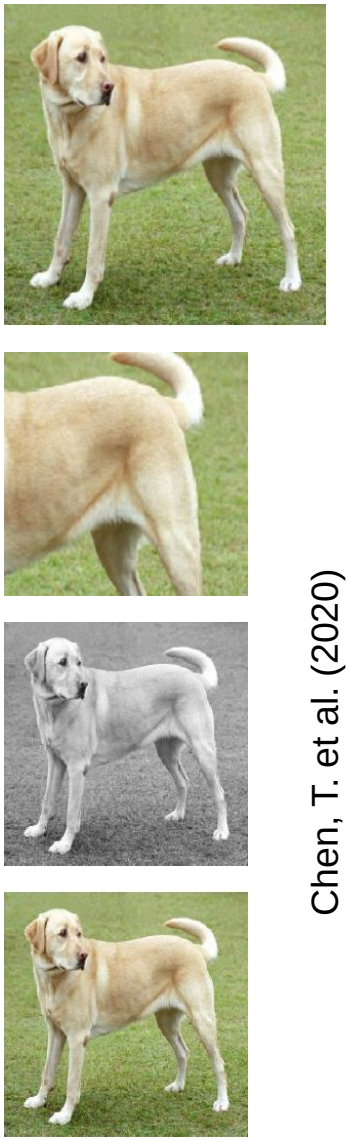
SimCLR framework components

1. Stochastic data augmentation module

→ Two randomly transformed views $\tilde{\boldsymbol{x}}_i$ & $\tilde{\boldsymbol{x}}_j$



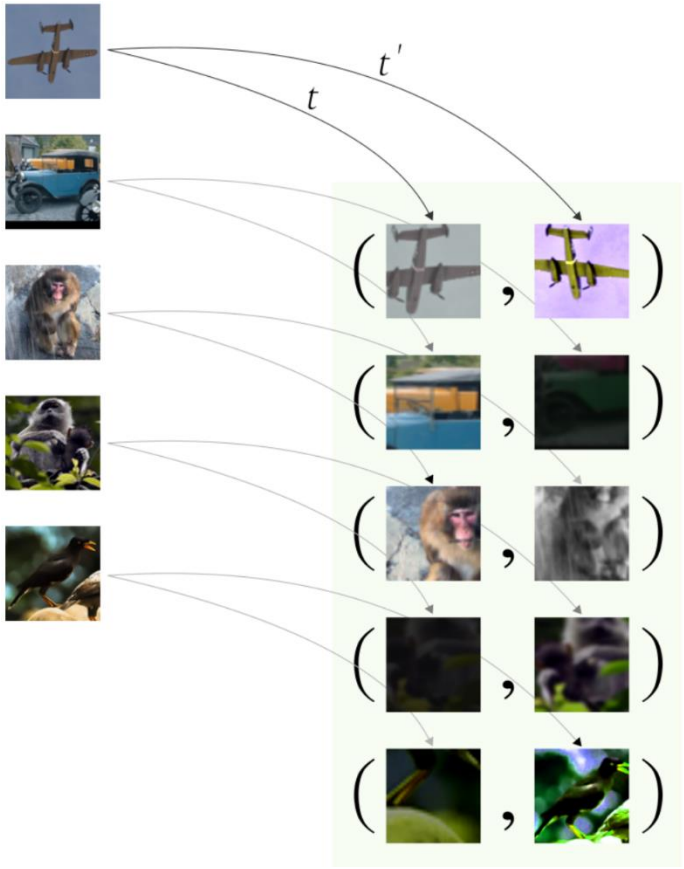
1. Random Cropping & Resize
2. Random Color Distortions
3. Random Gaussian Blur



Chen, T. et al. (2020)

Positive & negative samples

Positive pairs



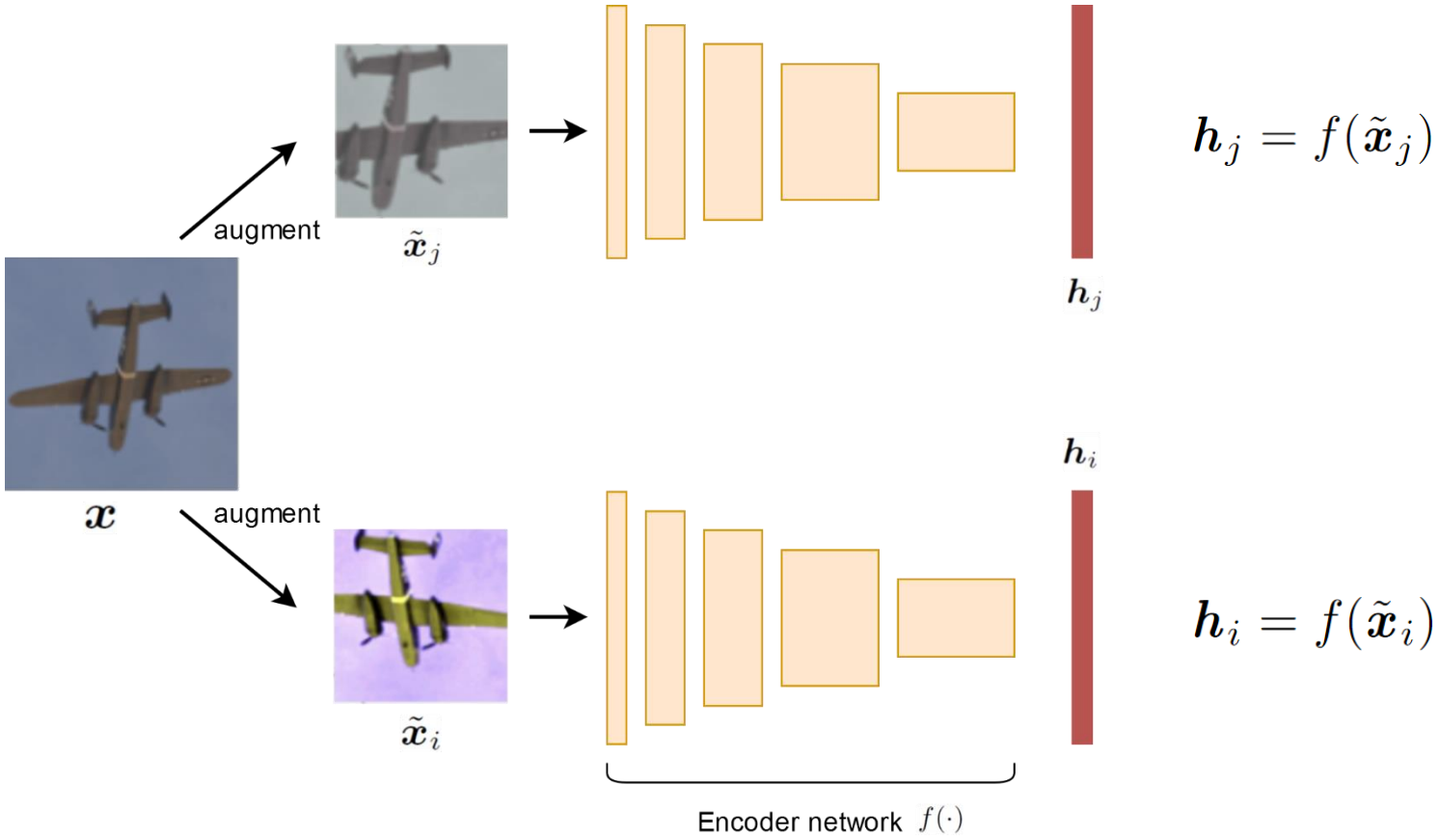
Negative pairs

Silva, T. (2020)

SimCLR framework components

2. Feature encoder network $f(\cdot)$ (default: ResNet-50)

→ Representation vectors $\boldsymbol{h}_i$ & $\boldsymbol{h}_j$



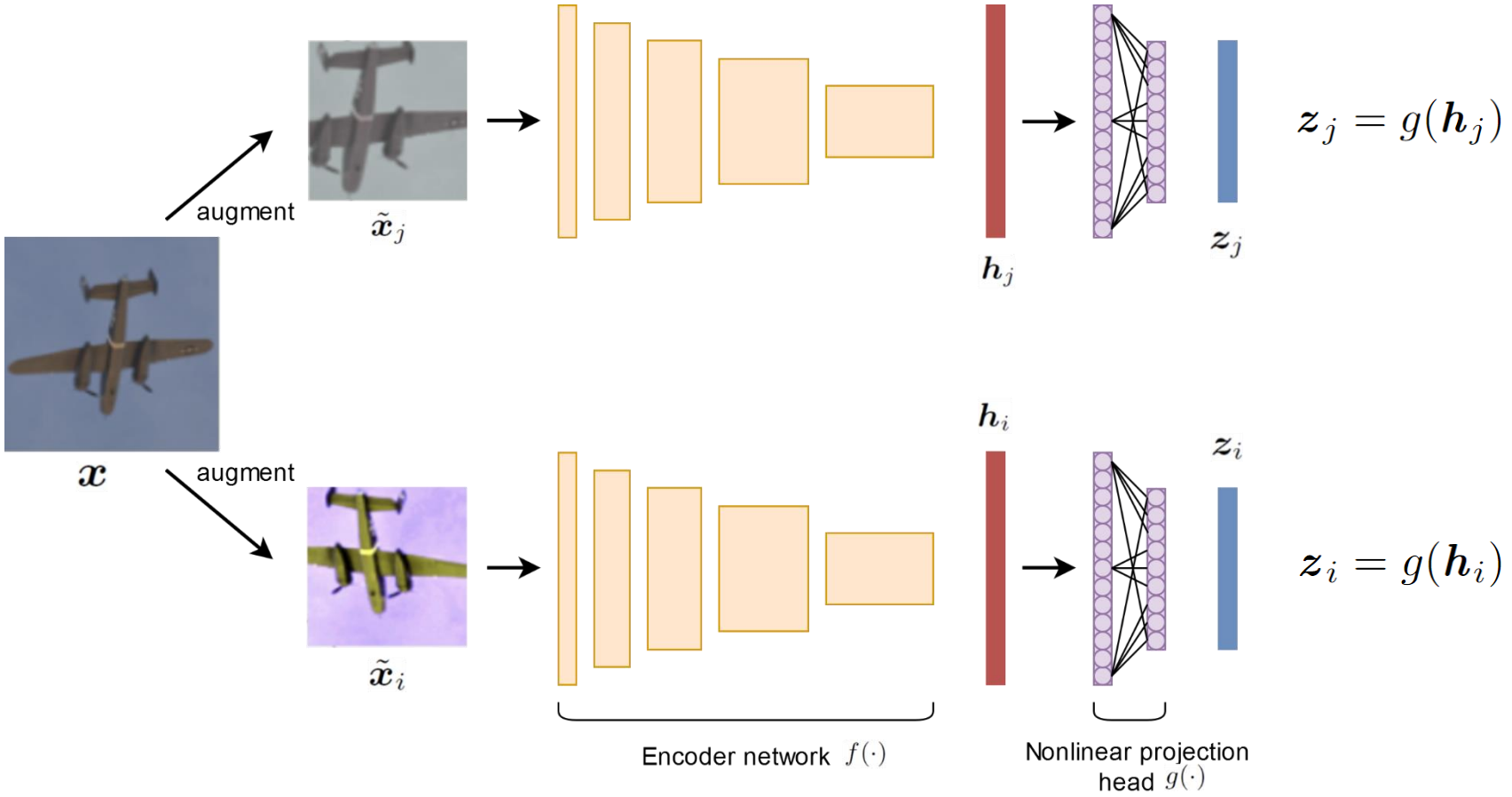
$$\boldsymbol{h}_j = f(\tilde{\boldsymbol{x}}_j)$$

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

SimCLR framework components

3. Small nonlinear projection head $g(\cdot)$ (2-layer MLP)

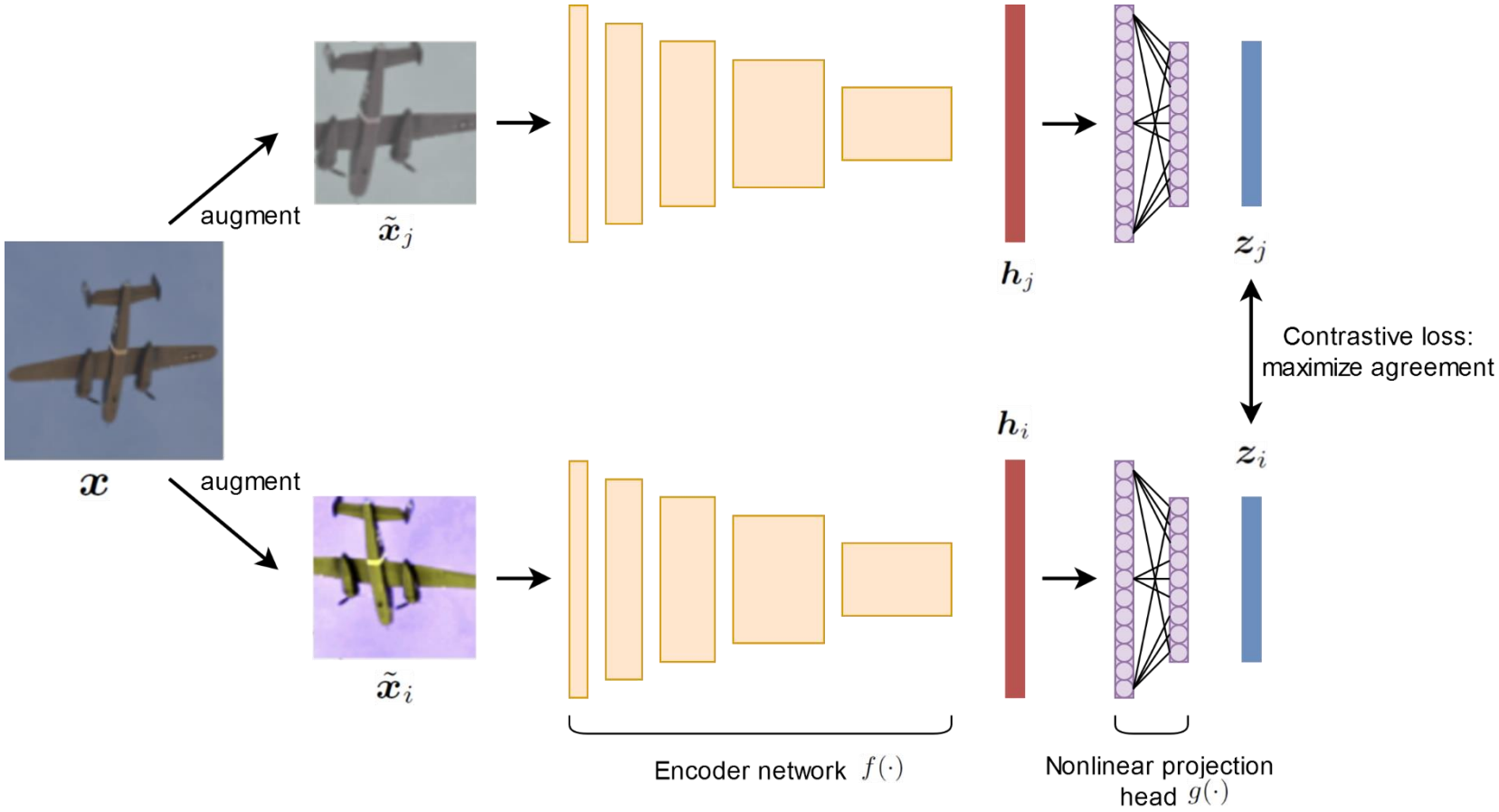
→ Representation vectors $\boldsymbol{z}_i$ & $\boldsymbol{z}_j$



SimCLR framework components

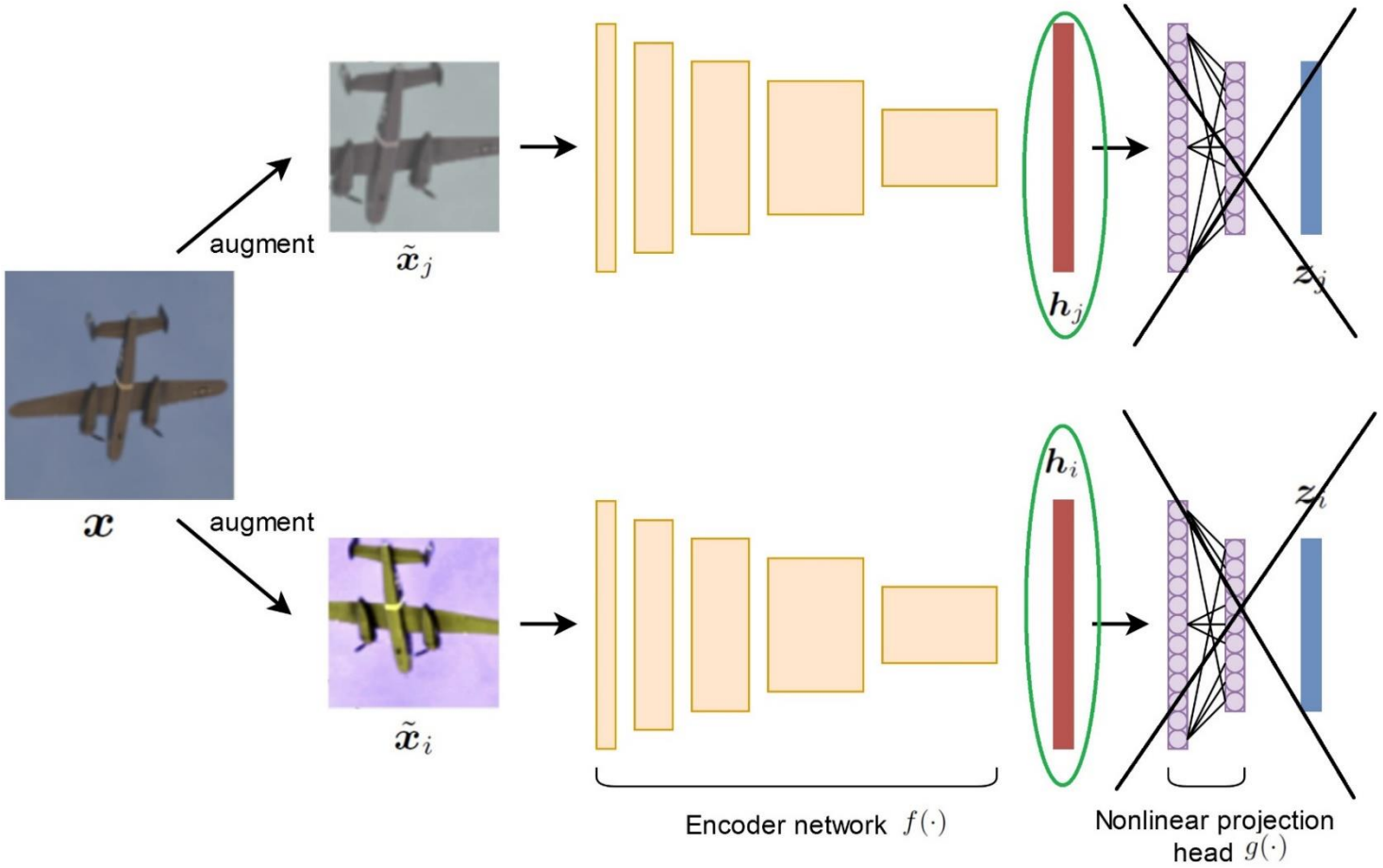
4. Contrastive loss function & prediction task

→ For a given $\tilde{\boldsymbol{x}}_i$, identify $\tilde{\boldsymbol{x}}_j$ in the set $\{\tilde{\boldsymbol{x}}_k\}_{k \neq i}$



Inference in SimCLR

- Remove nonlinear projection head
- Use representation **h** for downstream tasks

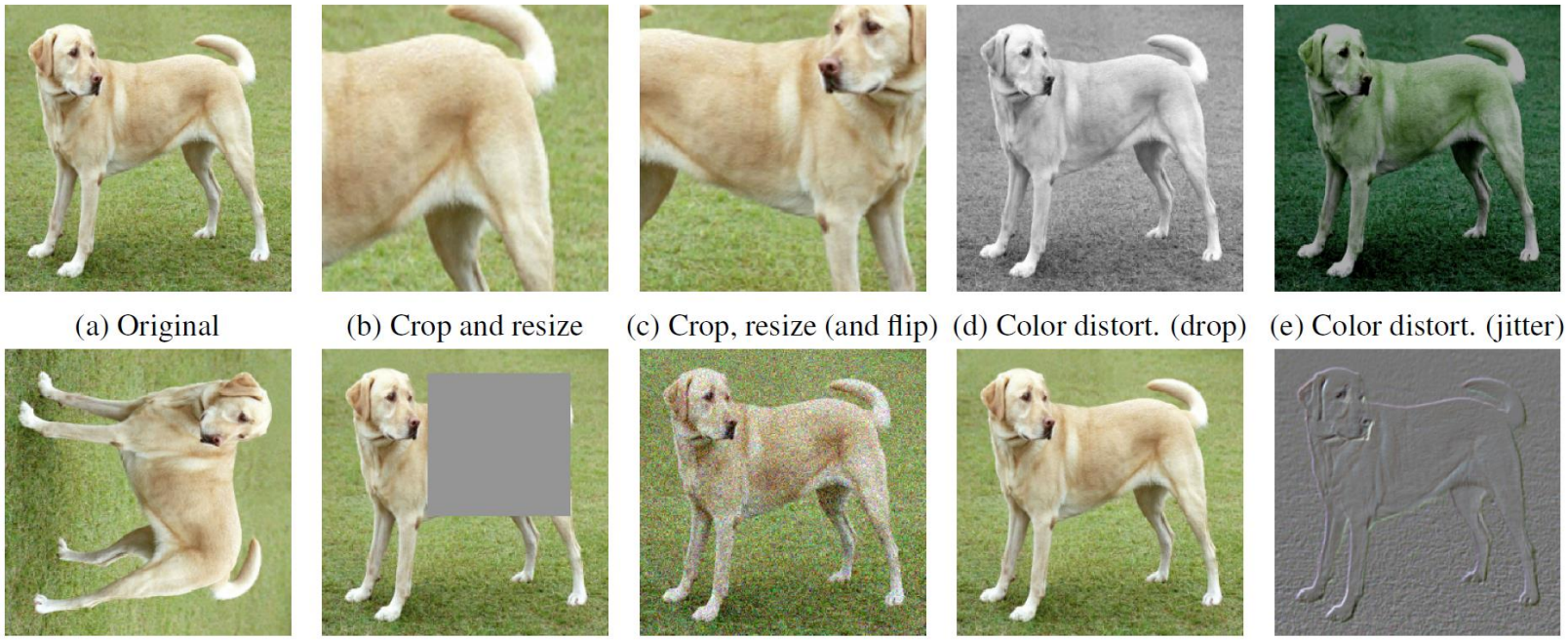


What enables good contrastive representation learning?

- Data augmentations & their compositions
- Learnable nonlinear projection head
- Scaling up architecture, training data & duration

Composition of augmentations is crucial

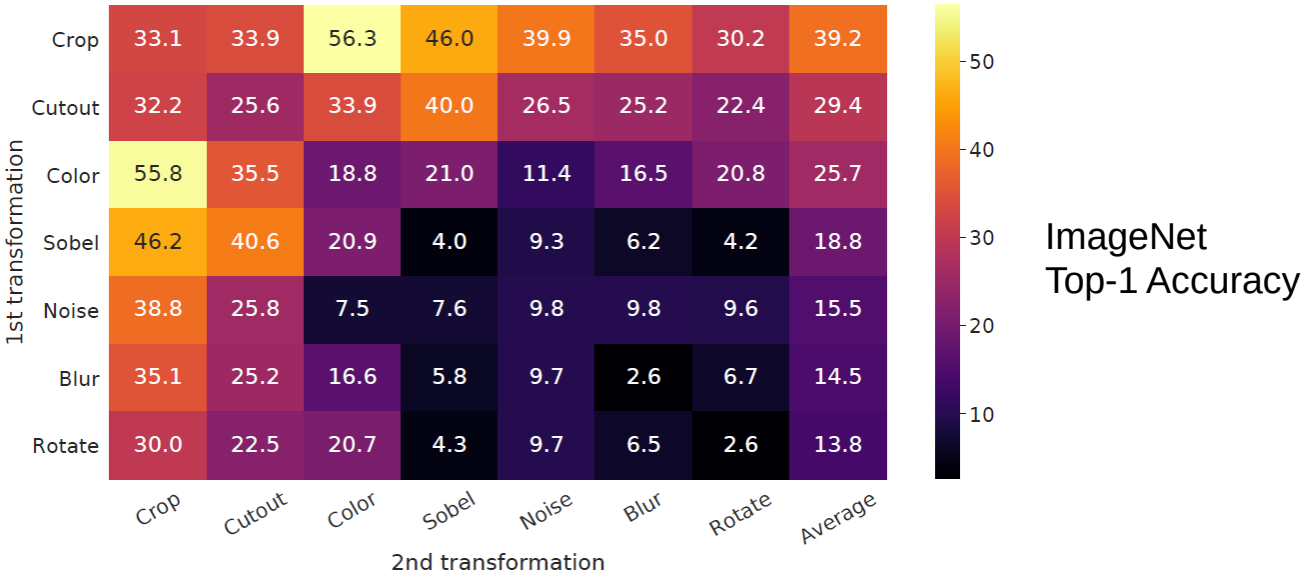
- Systematic study of different data augmentations & compositions
- Considered transformations:



Chen, T. et al. (2020)

Composition of augmentations is crucial

- Target transformation is applied to only one branch
- Augmentations applied individually (on diagonal) & in pairs



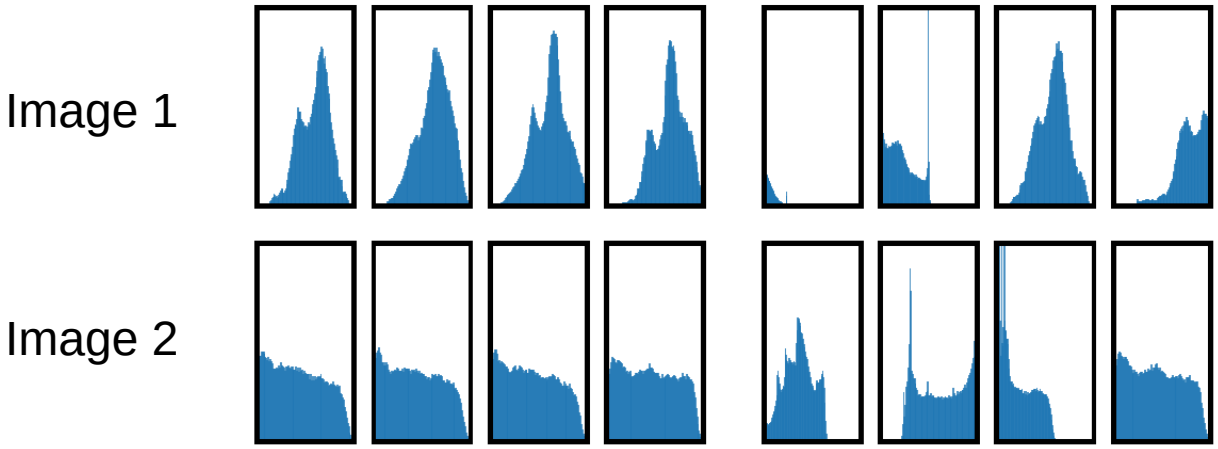
Chen, T. et al. (2020)

→ No single transformation is sufficient for good representations

→ Best composition: Random crop & color distortion

Effect of color distortion

- Problem: most random crops from an image share a similar color distribution



(a) Without color distortion. (b) With color distortion.

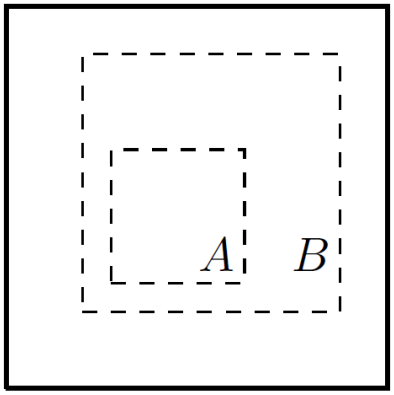
Chen, T. et al. (2020)

→ Color histograms suffice to distinguish images

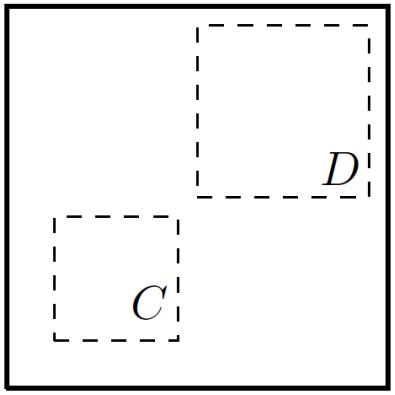
→ Potential shortcut for neural networks to solve task

Data augmentation defines predictive tasks

- Previous approaches define prediction tasks by changing the architecture
- SimCLR: Simple random cropping creates predictive tasks:



Global-to-local view prediction



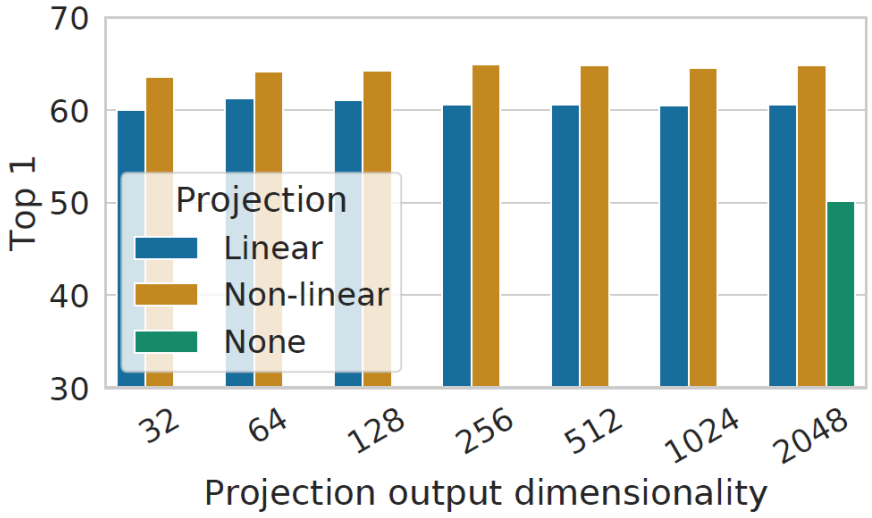
Neighboring view prediction

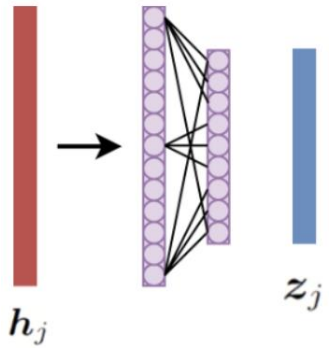
Chen, T. et al. (2020)

→ Decouples predictive task from other components (e.g. architecture)

Nonlinear projection head improves representation

- Linear evaluation results for 3 different projection heads:





Chen, T. et al. (2020)

No projection	Linear projection	Nonlinear projection
50%	60% (+ 10%)	63 (+ 13%)

→ z is trained to be invariant to data transformations

→ By using a nonlinear projection, more information can be maintained in h

Information loss induced by contrastive loss

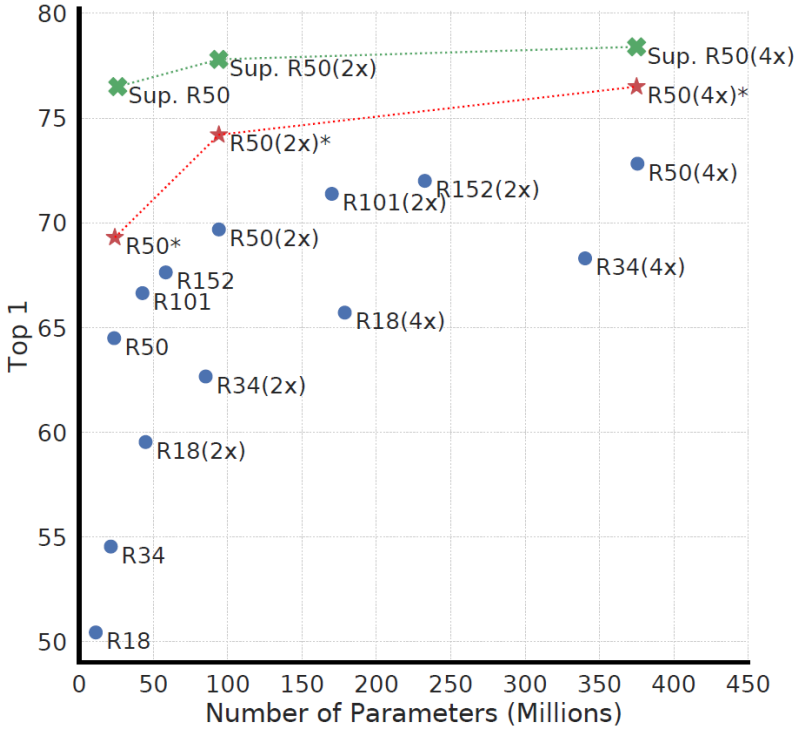
- Assumption: nonlinear projection ***g*** removes information about transformations
- Experiment: Learn to predict the transformation applied during pretraining, using either ***h*** or ***g(h)***

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

Chen, T. et al. (2020)

→ ***h*** contains more information while ***g(h)*** loses information

Self-supervised learning benefits more from scaling up



Chen, T. et al. (2020)

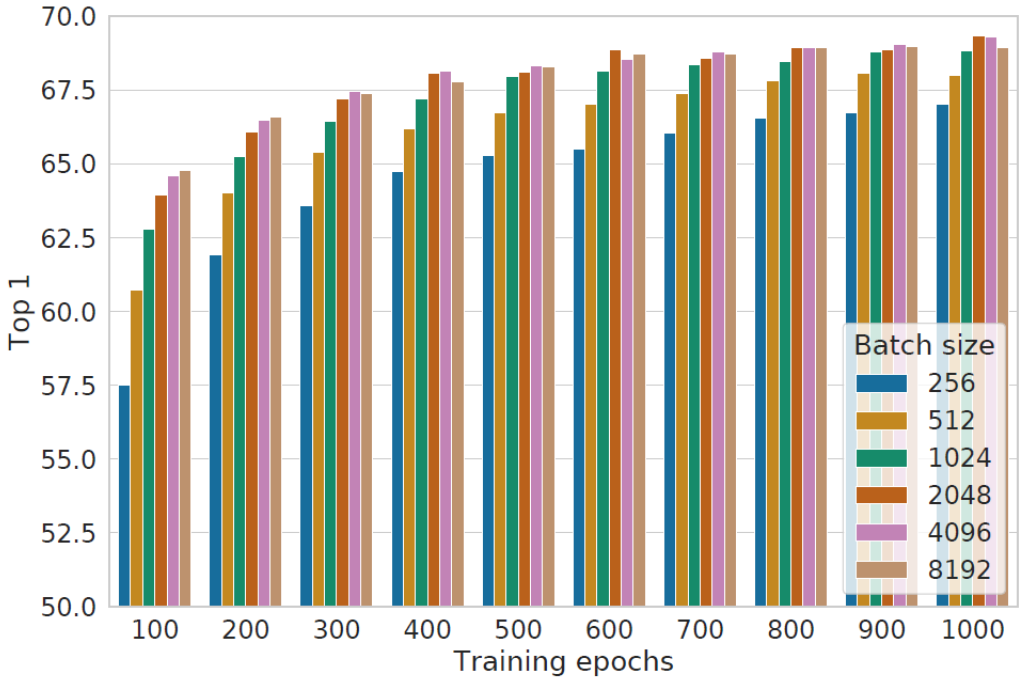
Green: 90 epochs
Blue: 100 epochs
Red: 1000 epochs

→ Increasing encoder depth & width improves performance

→ Accuracy gap between supervised & self-supervised models shrinks with increasing model size

Self-supervised learning benefits more from scaling up

- Impact of batch size for different #training epochs:



Chen, T. et al. (2020)

→ The shorter the training, the more beneficial are larger batch sizes

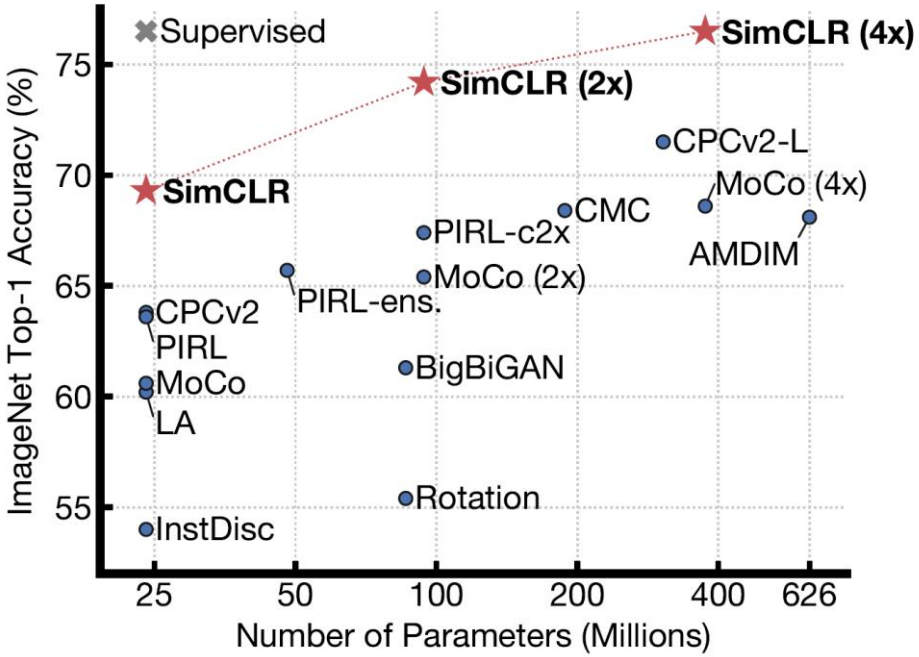
→ Larger batch size = more negative samples → Faster progress

Summing up: Advantages of SimCLR

- Very simple design
- No specialized architecture or memory bank required
- Exchangeable encoder architecture
- Large benefits from scaling up

Results: Linear evaluation performance

- Linear classifiers trained on top of frozen pretrained models



Chen, T. et al. (2020)

→ SimCLR outperforms previous self-supervised models

→ Best SimCLR model reaches accuracy of supervised pretrained ResNet

Results: Transfer learning performance

- Evaluation across 12 image datasets
- Two settings: Linear evaluation (frozen network) & Fine-tuning
- Results for ResNet-50 (4x):

	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

Chen, T. et al. (2020)

→ Fine-tuning: SimCLR outperforms supervised on 5 datasets

→ No clear advantage of supervised over self-supervised learning

Limitations of SimCLR

- Needs much larger encoder architectures to compete with supervised models
- Large batch size requires TPU support (default size: 4096)
  → not memory-efficient
- Very sensitive to hyperparameters
- Global semantic structure of dataset is ignored (Negative pairs with similar semantics are pushed apart)

Outlook

- MoCo v2: integrates concepts of SimCLR
  - Nonlinear projection head
  - Stronger data augmentation
  - → Improves SimCLR accuracy & runs on a regular GPU machine
- SimCLR v2
  - Encoder network: Deeper ResNet model
  - Deeper nonlinear projection head
  - Incorporated memory mechanism from MoCo
- Many new state of the art self-supervised methods

→ Fast progress & advances!

Take home messages

- Simple often works better
- Composition of data augmentations is crucial
- Introducing a nonlinear projection before the contrastive loss improves the representation quality
- Contrastive learning benefits more from scaling-up than supervised learning
- Closing gap between supervised & self-supervised learning in computer vision

Thank you for your attention!

Bibliography

- Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*
- Wu, Z., Xiong, Y., Yu, S. X., & Lin, D. (2018). Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3733-3742)
- He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 9729-9738)
- Silva, T. (2020). Exploring SimCLR: A Simple Framework for Contrastive Learning of Visual Representations. https://towardsdatascience.com/exploring-simclr-a-simple-framework-for-contrastive-learning-of-visual-representations-158c30601e7e. Visited on 26th of October, 2020
- Sohn, K. (2016). Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems* (pp. 1857-1865)
- Image source for slide 2 https://raw.githubusercontent.com/bagdonas/beautystack/master/docs/images/example1.jpg

Appendix

NT-Xent loss

- Normalized temperature-scaled cross entropy loss
- Modification of the InfoNCE loss (similarity-score is scaled by a temperature parameter, cosine similarity used as score)
- Loss function for a positive pair of samples (i, j):

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

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

- Final loss is computed across all positive pairs in a minibatch
- Loss is applied on normalized embeddings (z has unit length)
- Temperature τ normalizes similarity score (improves stability); with lower value, the importance of the positive pair is increased

Algorithm pseudo code

Algorithm 1 SimCLR's main learning algorithm.

input: batch size N, constant 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**
    $s_{i,j} = \boldsymbol{z}_i^\top \boldsymbol{z}_j / (\|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|)$ # pairwise similarity
  end for
  define $\ell(i, j)$ **as** $\ell(i, j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network $f(\cdot)$, and throw away $g(\cdot)$

Chen, T. et al. (2020)

Setup & Hyperparameters for ablation study

Contrastive loss	NT-Xent loss (adaption of NCE loss)
Encoder network	ResNet-50
Nonlinear projection head	2-layer MLP (output dim.: 128)
Optimizer	LARS with linear learning rate scaling
Learning rate	0.3 * batch_size / 256
Learning rate scheduler	Linear warmup for the first 10 epochs, Cosine decay schedule without restarts
Weight decay	10^{-6}
Batch size	4096
Training epochs	100
Augmentations	Random crop & resize, Color distortion, Gaussian Blur

Hyperparameters for performance comparison

Contrastive loss	NT-Xent loss (adaption of NCE loss)
Encoder network	ResNet-50 in 3 different widths (1x, 2x, 4x)
Nonlinear projection head	2-layer MLP (output dim.: 128)
Optimizer	LARS with linear learning rate scaling
Learning rate	0.3 * batch_size / 256
Learning rate scheduler	Linear warmup for the first 10 epochs, Cosine decay schedule without restarts
Weight decay	10^{-6}
Batch size	4096
Training epochs	1000
Augmentations	Random crop & resize, Color distortion, Gaussian Blur

Augmentation procedure details

1. Random crop & resize to 224 x 224

- Crop of random size (uniform from 0.08 to 1.0) & random aspect ratio (default: 3/4 to 4/3)
- Crop is resized to the original size
- With probability p = 0.5, perform horizontal / left-to-right flip

2. Color distortion

- Composed of color jittering & color dropping with a strength parameter
- Color jittering: random brightness, contrast, saturation, hue (p = 0.8)
- Color drop: convert rgb image to grayscale image (p = 0.2)

3. Gaussian blur

- Blur image with probability p = 0.5 using a Gaussian kernel
- Randomly sample $\sigma \in [0.1, 2.0]$, kernel size = 10% of image height / width

Contrastive learning benefits from stronger augmentation

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

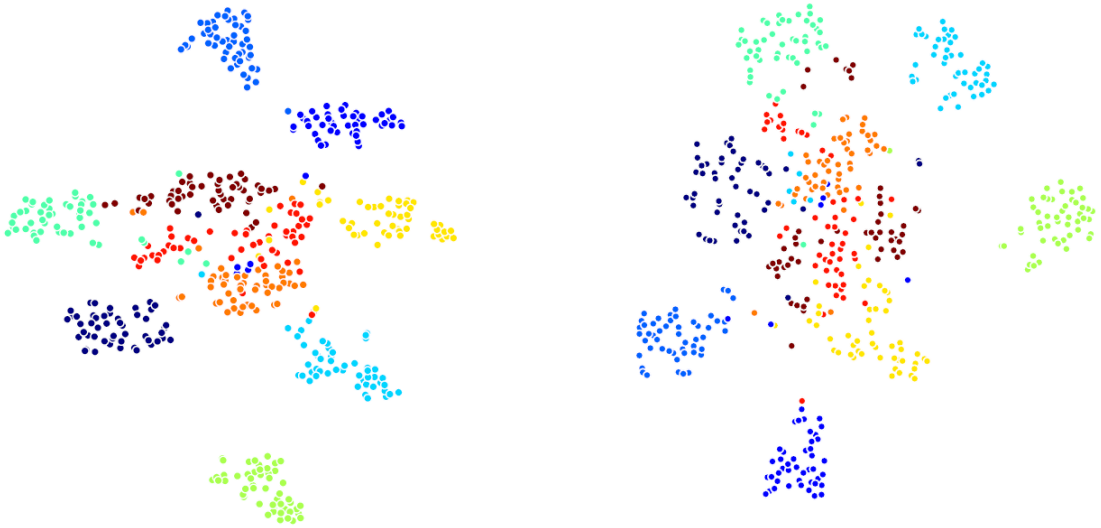
Chen, T. et al. (2020)

→ **SimCLR**: stronger color augmentation improves linear evaluation accuracy

→ **Supervised**: extreme color distortion does not improve & can hurt performance

Nonlinear head: Better representation separability

Visualizations of hidden vectors of images from randomly selected 10 classes



(a) $\boldsymbol{h}$ (b) $\boldsymbol{z} = g(\boldsymbol{h})$

Chen, T. et al. (2020)

→ Classes represented by ***h*** are better separated compared to ***z***

Performance comparison of different contrastive losses

- NT-Logistic & Margin Triplet loss only regard 1 positive & 1 negative sample → need negative mining for good performance
- Linear evaluation for models trained with different contrastive losses (sh = using negative mining)

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

Chen, T. et al. (2020)

→ Negative mining helps, but NT-Xent loss works still much better

Linear evaluation performance

- Linear classifiers trained on top of frozen pretrained models
- ImageNet Top-1 & Top-5 accuracy

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**

Chen, T. et al. (2020)

Design choices comparison: SimCLR & previous approaches

- **CPC**: defines context prediction task by splitting images into patches. Uses a context aggregation network
- **AMDIM**: performs global-to-local / local-to-neighbor prediction. Uses modified ResNet with constraints on the receptive fields
- **CMC**: uses a separated network for each view & a different loss
- **MoCo** & **PIRL**: use an explicit memory bank

Model	Data Augmentation	Base Encoder	Projection Head	Loss	Batch Size	Train Epochs
CPC v2	Custom	ResNet-161 (modified)	PixelCNN	Xent	$512^{\#}$	$\sim$200
AMDIM	Fast AutoAug.	Custom ResNet	Non-linear MLP	Xent w/ clip,reg	$1008^{\#}$	150
CMC	Fast AutoAug.	ResNet-50 (2×, L+ab)	Linear layer	Xent w/ ℓ_2, τ	156^{*}	280
MoCo	Crop+color	ResNet-50 (4×)	Linear layer	Xent w/ ℓ_2, τ	256^{*}	200
PIRL	Crop+color	ResNet-50 (2×)	Linear layer	Xent w/ ℓ_2, τ	1024^{*}	800
SimCLR	Crop+color+blur	ResNet-50 (4×)	Non-linear MLP	Xent w/ ℓ_2, τ	4096	1000

Chen, T. et al. (2020)

* = a memory bank is used

= images are split into multiple patches

→ Combination of design choices in SimCLR is crucial. Design choices in SimCLR are generally simpler

Transfer learning performance: ResNet vs. ResNet 4x

SimCLR vs. Supervised: **ResNet-50 (4x)**

	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

→ SimCLR outperforms on 5 datasets

Chen, T. et al. (2020)

SimCLR vs. Supervised: **ResNet-50**

	Food	CIFAR10	CIFAR100	Birdsnap	SUN397	Cars	Aircraft	VOC2007	DTD	Pets	Caltech-101	Flowers
Linear evaluation:												
SimCLR (ours)	68.4	90.6	71.6	37.4	58.8	50.3	50.3	80.5	**74.5**	83.6	90.3	91.2
Supervised	**72.3**	**93.6**	**78.3**	**53.7**	**61.9**	**66.7**	**61.0**	**82.8**	74.9	**91.5**	**94.5**	**94.7**
Fine-tuned:												
SimCLR (ours)	**88.2**	**97.7**	**85.9**	**75.9**	63.5	91.3	**88.1**	84.1	**73.2**	89.2	92.1	97.0
Supervised	**88.3**	**97.5**	**86.4**	**75.8**	**64.3**	**92.1**	86.0	**85.0**	**74.6**	**92.1**	**93.3**	**97.6**
Random init	86.9	95.9	80.2	**76.1**	53.6	91.4	85.9	67.3	64.8	81.5	72.6	92.0

Chen, T. et al. (2020)

→ SimCLR outperforms on only 1 dataset

→ With a narrower architecture, supervised learning has a clear advantage over self-supervised learning. Accuracy gap decreases for bigger models