Pretext tasks
3. SimCLR

1 2	SELF-PREDICTION	INNATE RELATIONSHIP (Context-based)	1. ROTATION 2. RELATIVE POSITION	IMAGE
3	CONTRASTIVE LEARNING	INTER-SAMPLE CLASSIFICATION	1. Instance Discrimination 2. SimCLR [Contrastive Loss] 3. Theory – Guarantees / Bounds	IMAGE
4	CONTRASTIVE LEARNING	INTER-SAMPLE CLASSIFICATION	Contrastive Predictive Coding (CPC), [NCE, InfoNCE Loss]	AUDIO/ SPEECH
5	SELF-PREDICTION	GENERATIVE (VAE)	1. AE – Variational Bayes 2. VQ-VAE + AR	IMAGE AUDIO/ SPEECH
6	SELF-PREDICTION	GENERATIVE (AR)	1. AR-LM – GPT 2. Masked-LM – BERT	LANGUAGE
7	SELF-PREDICTION	MASKED-GEN (Masked LM for ASR)	1. Wav2Vec / 2.0 2. HuBERT	AUDIO/ SPEECH

A Simple Framework for Contrastive Learning of Visual Representations

Ting Chen, Simon Kornblith, Mohammad Norouzi, Geoffrey Hinton

A Simple Framework for Contrastive Learning of Visual Representations

Ting Chen [1] Simon Kornblith [1] Mohammad Norouzi [1] Geoffrey Hinton [1]

[1]Google Research, Brain Team. Correspondence to: Ting Chen <iamtingchen@google.com>.

Proceedings of the 37th International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

[1]Code available at https://github.com/google-research/simclr.

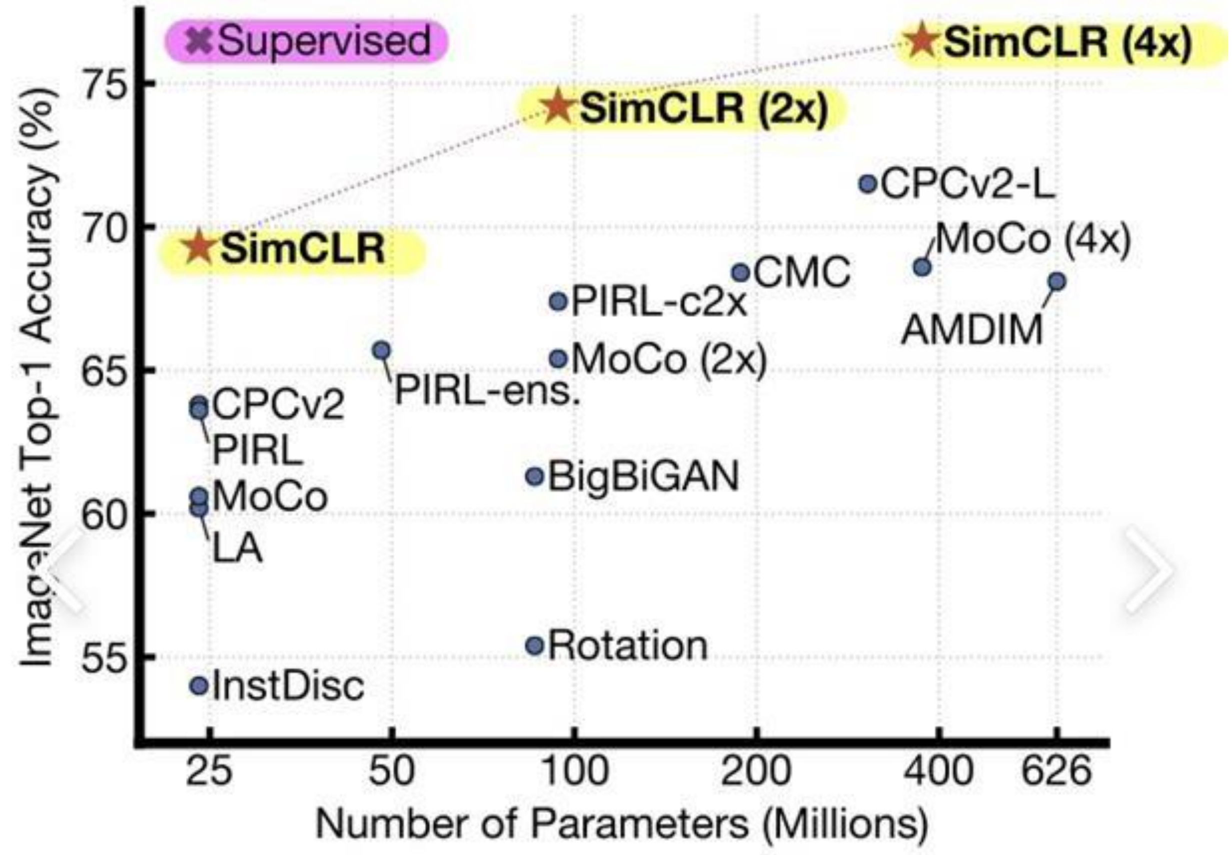
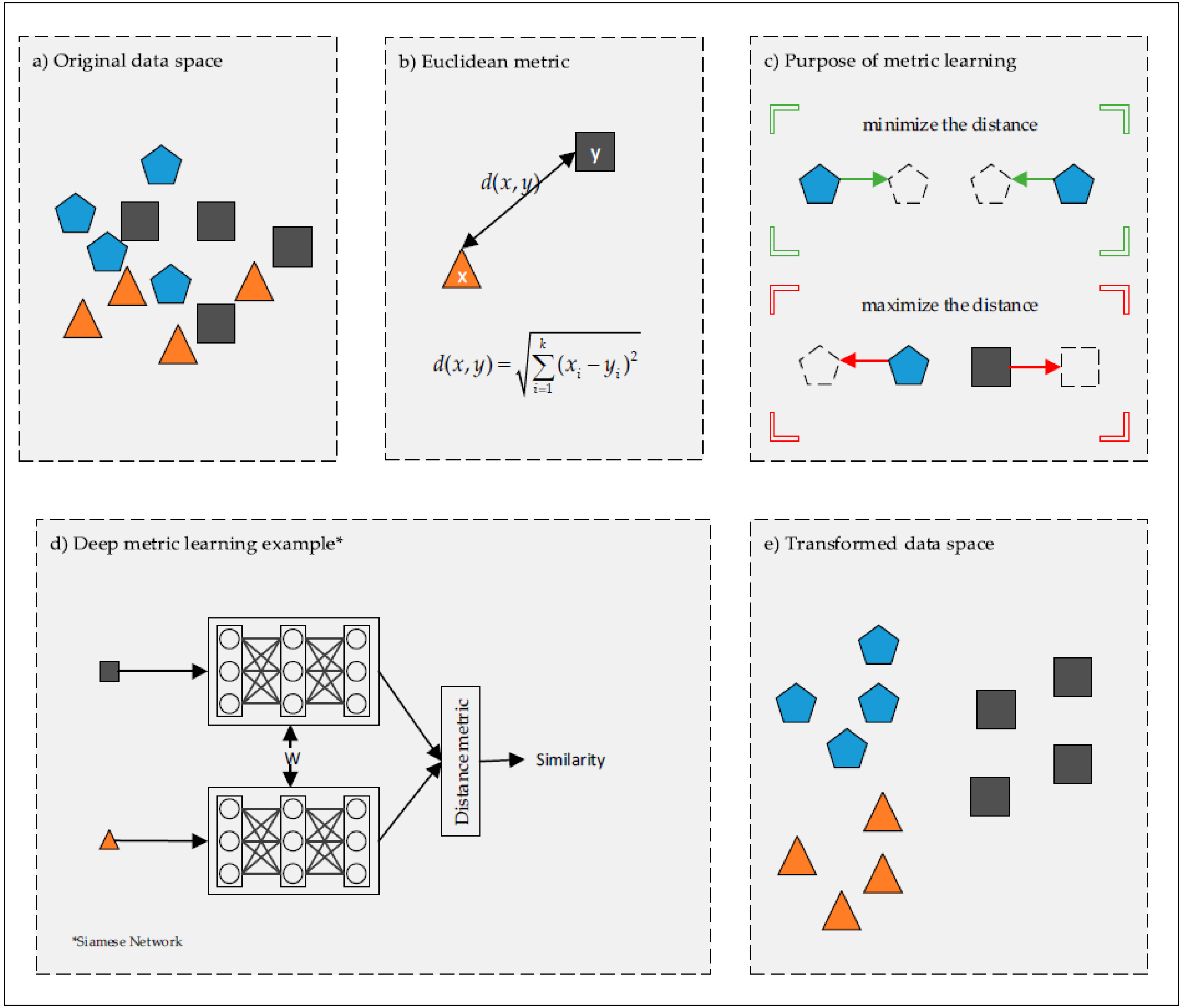
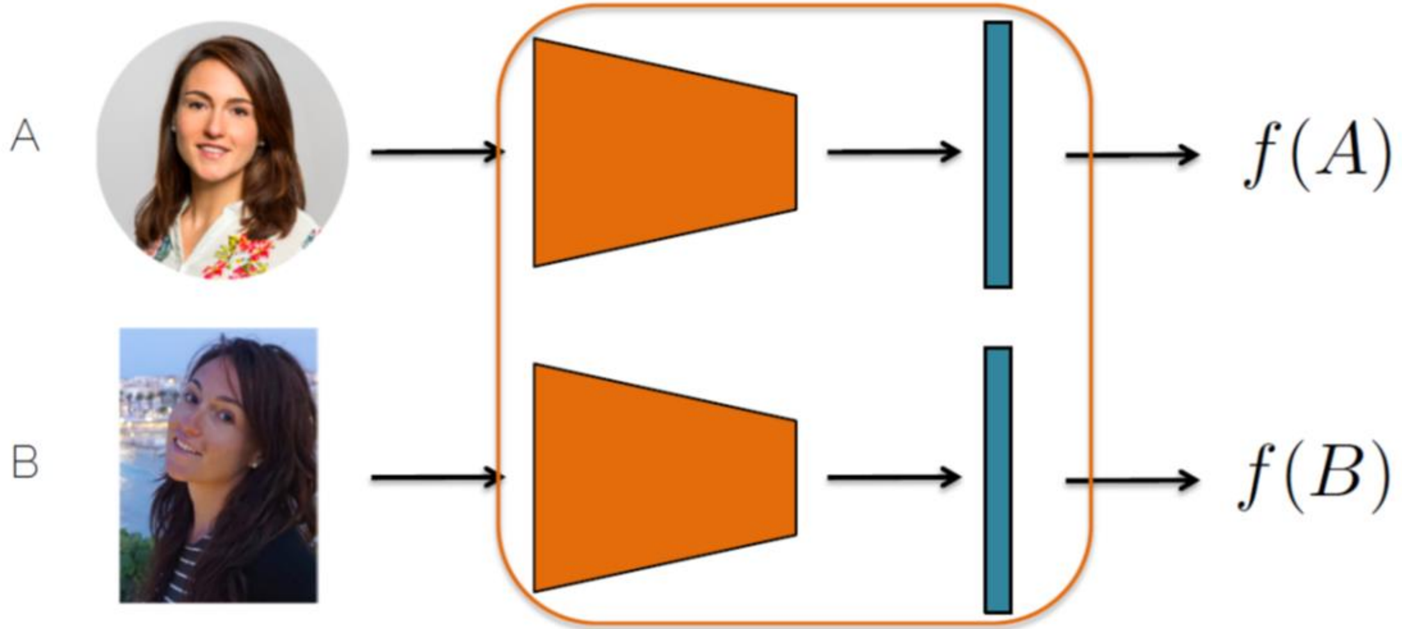


Figure 1. ImageNet Top-1 accuracy of linear classifiers trained on representations learned with different self-supervised methods (pretrained on ImageNet). Gray cross indicates supervised ResNet-50. Our method, SimCLR, is shown in bold.

Deep Metric Learning



- Siamese network = shared weights



- Distance function $d(A, B) = ||f(A) - f(B)||^2$

- Training: learn the parameter such that

  - If A and B depict the same person, $d(A, B)$ is small

  - If A and B depict a different person, $d(A, B)$ is large

- Contrastive loss:

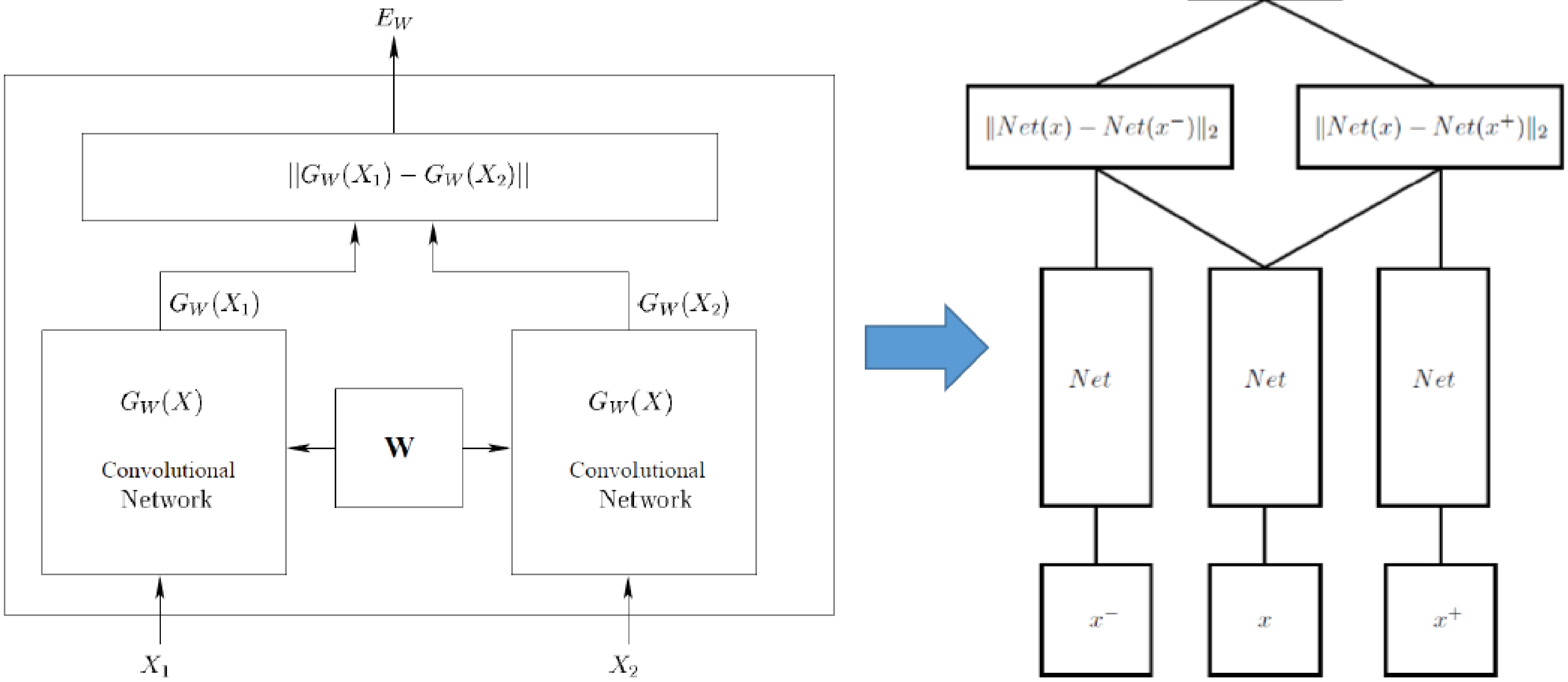
$$\mathcal{L}(A,B) = y^*||f(A)-f(B)||^2 + (1-y^*)max(0, m^2 - ||f(A)-f(B)||^2)$$

Positive pair, reduce the distance between the elements

Negative pair, brings the elements further apart up to a margin

- Training the siamese networks
  - You can update the weights for each channel independently and then average them
- This loss function allows us to learn to bring positive pairs together and negative pairs apart

Triplet Network



From Siamese to Triplet Network

Triplet loss

- Triplet loss allows us to learn a ranking

Anchor (A)

Positive (P)

Negative (N)

We want: $||f(A) - f(P)||^2 < ||f(A) - f(N)||^2$

- Triplet loss allows us to learn a ranking

$$||f(A) - f(P)||^2 < ||f(A) - f(N)||^2$$

$$||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 < 0$$

$$||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + m < 0$$

margin

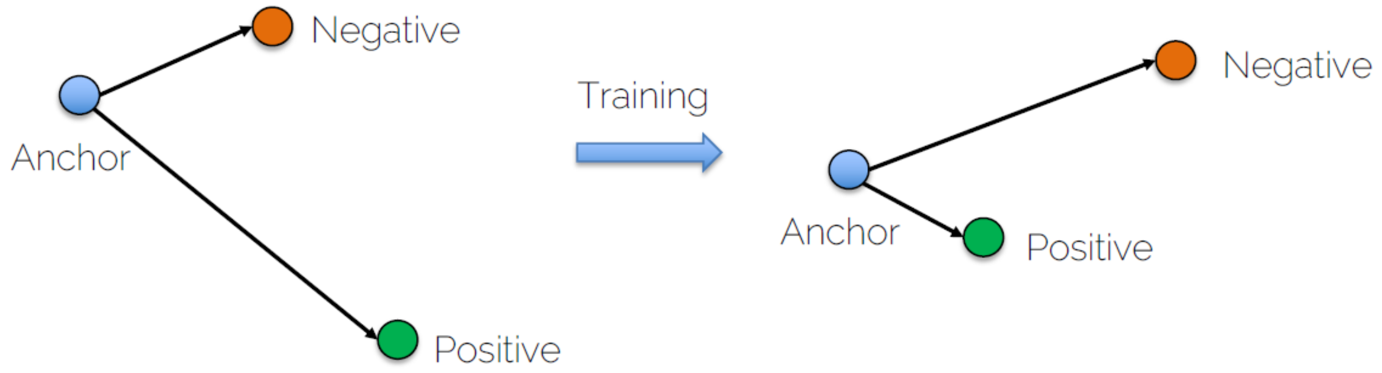
- Triplet loss allows us to learn a ranking

$$||f(A) - f(P)||^2 < ||f(A) - f(N)||^2$$

$$||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 < 0$$

$$||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + m < 0$$

$$\mathcal{L}(A, P, N) = max(0, ||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + m)$$



The proposed SimCLR framework

A simple idea: maximizing the agreement of representations under data transformation, using a contrastive loss in the latent/feature space.

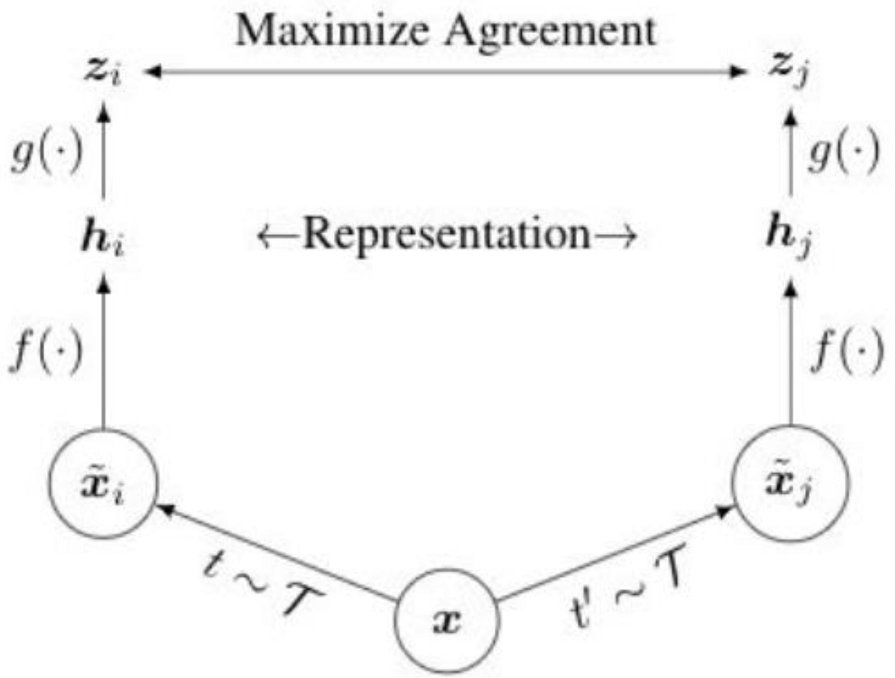
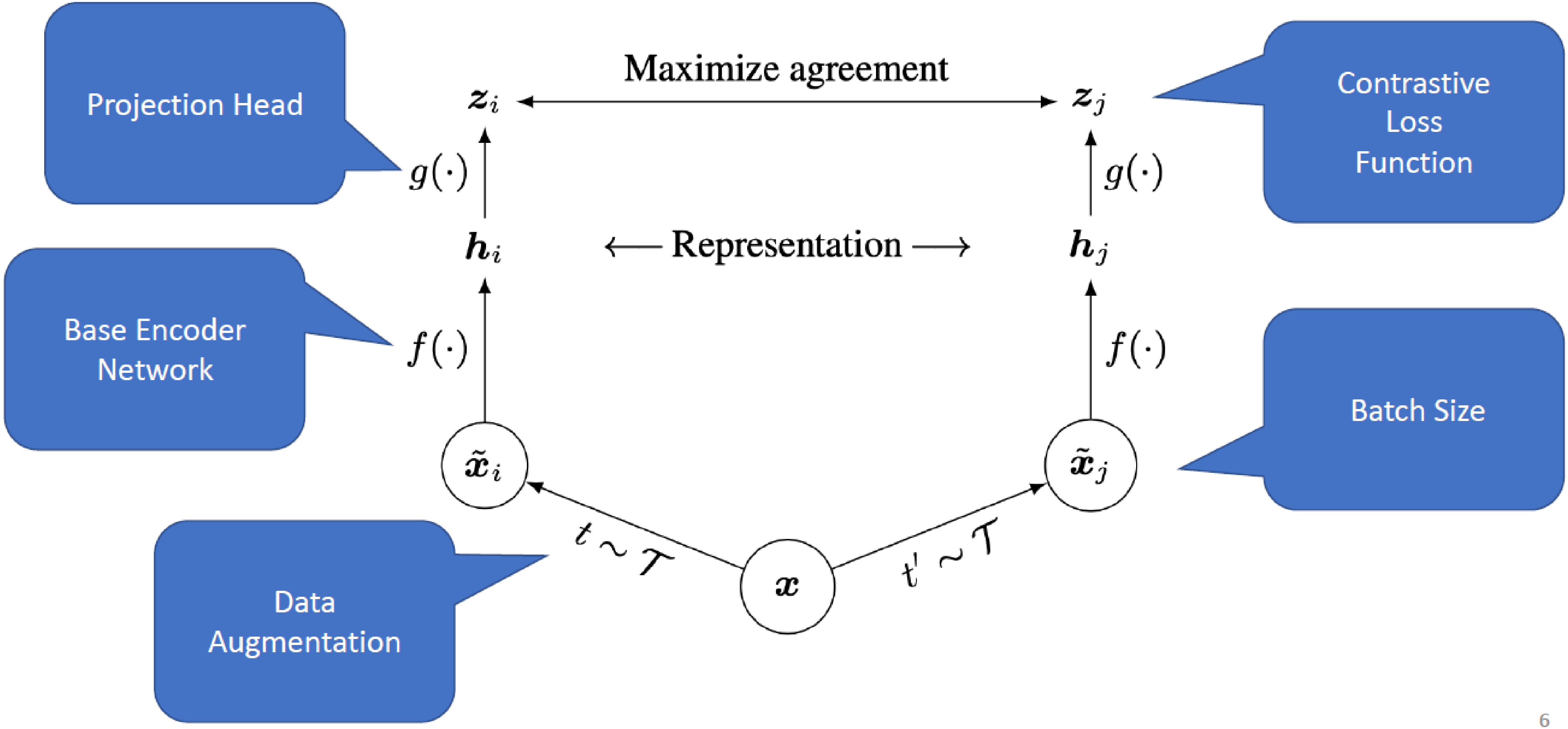


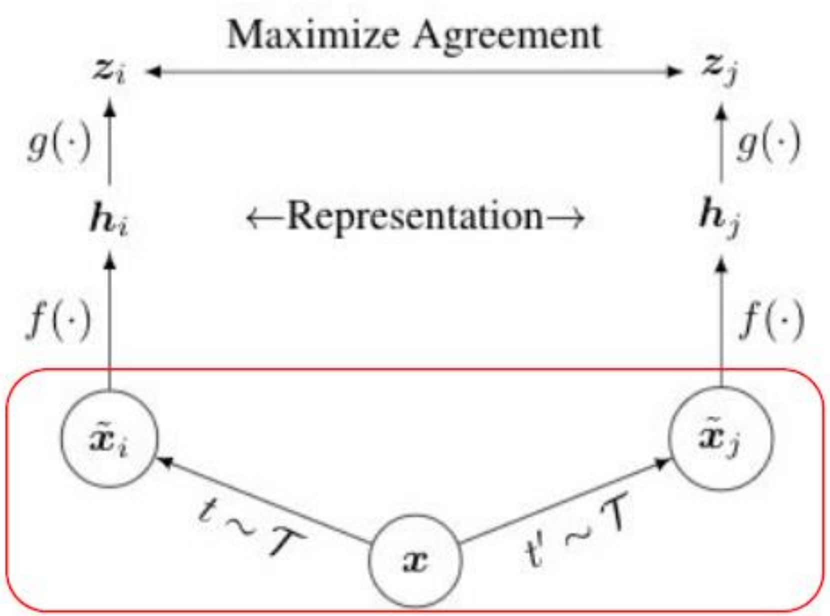
Figure 2. A framework for contrastive representation learning. Two separate stochastic data augmentations $t, t' \sim \mathcal{T}$ are applied to each example to obtain two correlated views. A base encoder network $f(\cdot)$ with a projection head $g(\cdot)$ is trained to maximize agreement in *latent representations* via a contrastive loss.

Framework



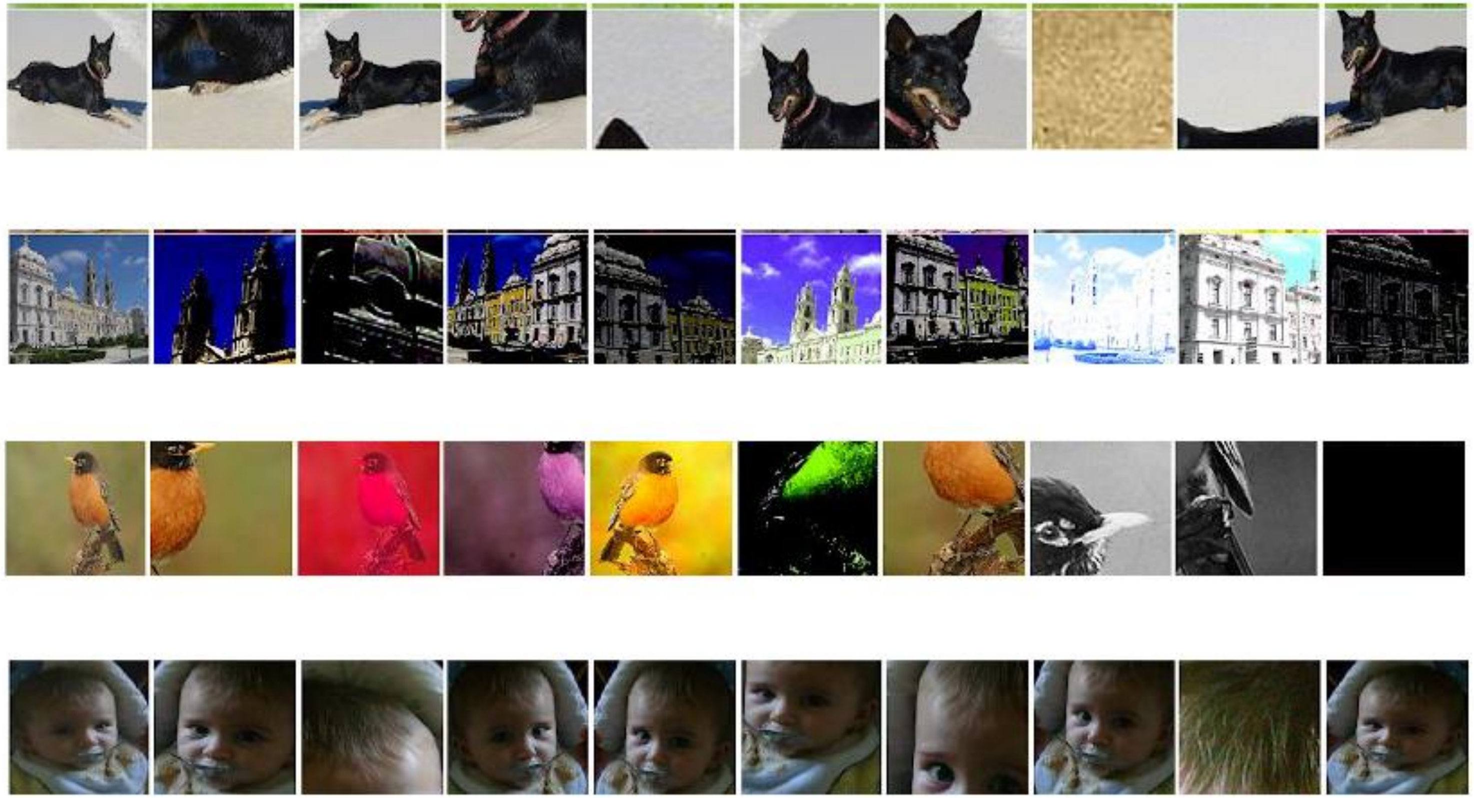
Projection Head
Base Encoder Network
Data Augmentation
Contrastive Loss Function
Batch Size
Maximize agreement
z_i
z_j
$g(\cdot)$
$g(\cdot)$
h_i
h_j
← Representation →
$f(\cdot)$
$f(\cdot)$
$\tilde{x}_i$
$\tilde{x}_j$
$t \sim \mathcal{T}$
$t' \sim \mathcal{T}$
x

6



We use random crop and color distortion for augmentation.

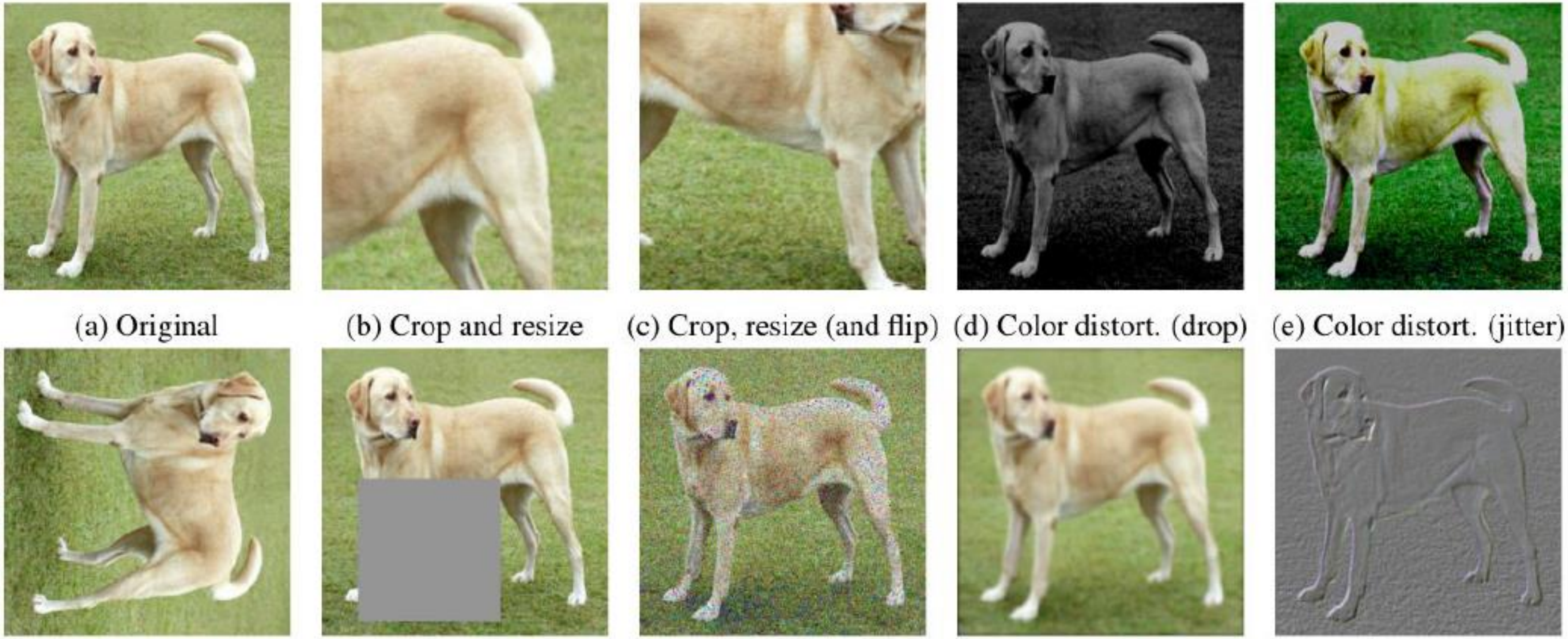
Examples of augmentation applied to the left most images:



Three augmentations applied sequentially

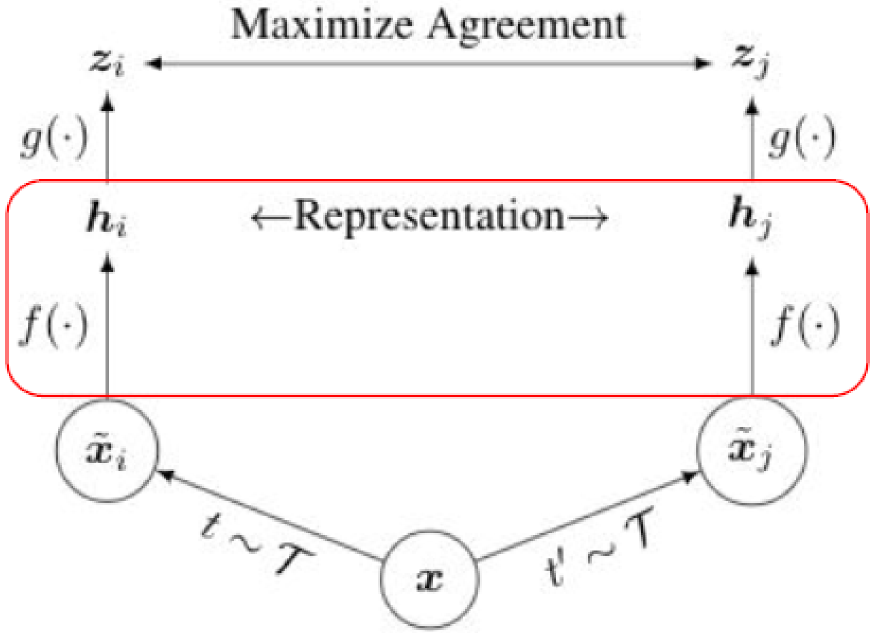
- Random cropping
- Random color distortions
- Random Gaussian blur

Systematically study a set of augmentation



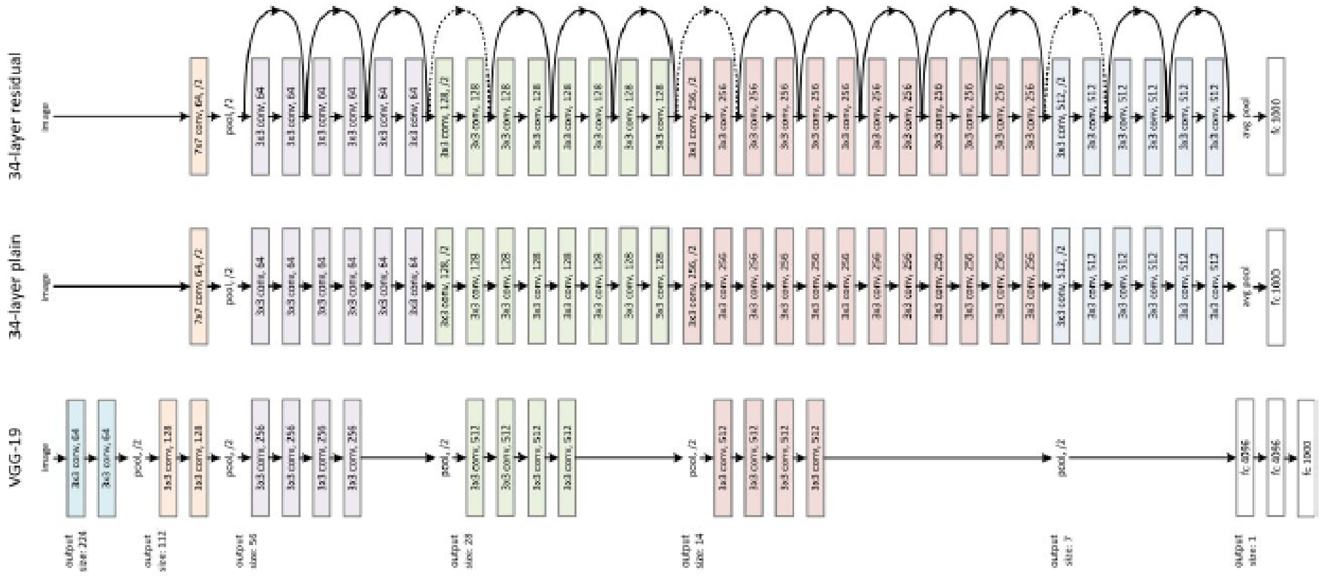
(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)

(f) Rotate $\{90°, 180°, 270°\}$ (g) Cutout (h) Gaussian noise (i) Gaussian blur (j) Sobel filtering



f(x) is the base network that computes internal representation.

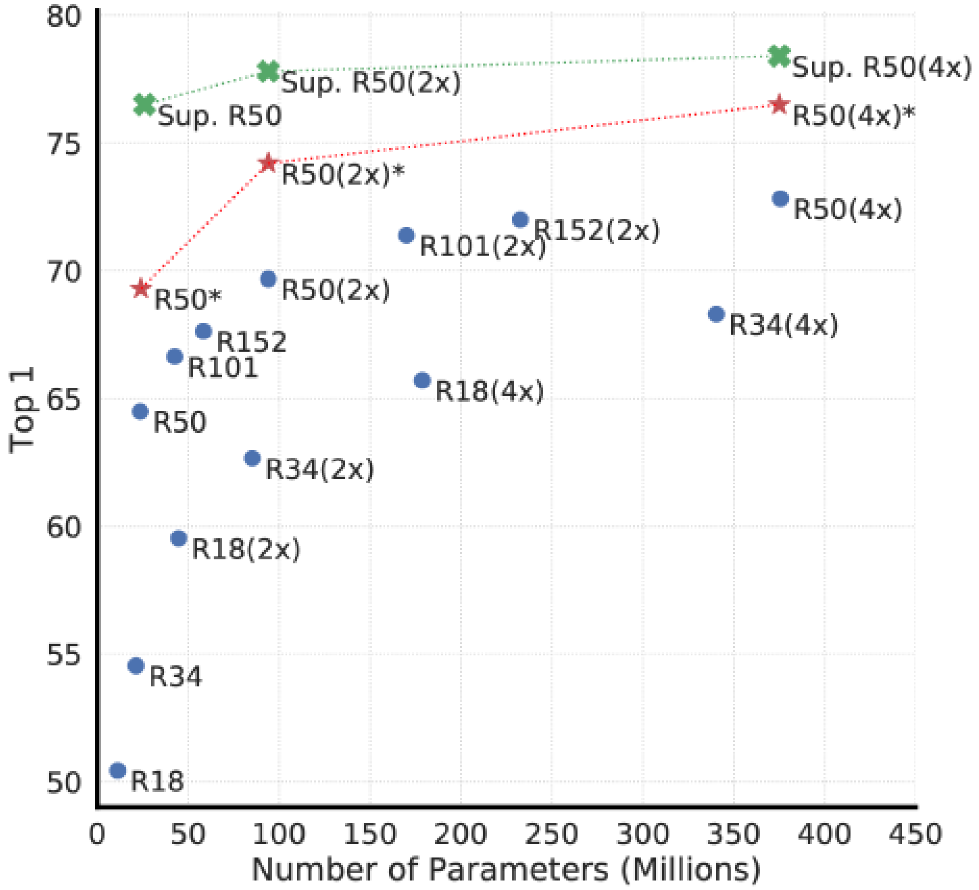
We use (unconstrained) ResNet in this work. However, it can be other networks.



SimCLR chooses ResNet

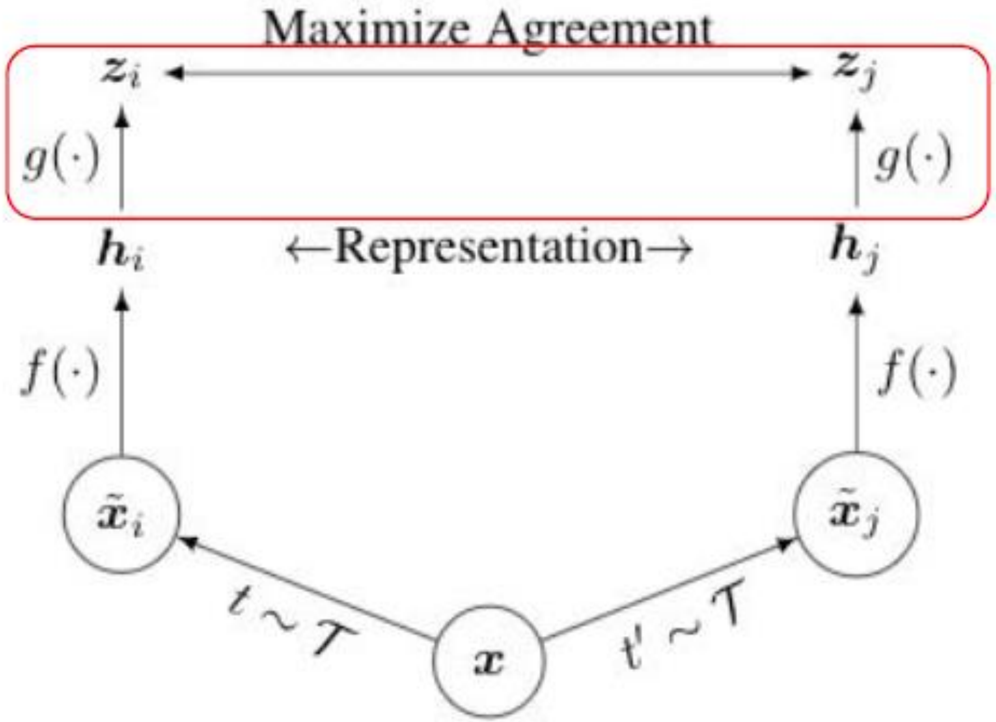
$$\boldsymbol{h}_i = f(\tilde{\boldsymbol{x}}_i) = ResNet(\tilde{\boldsymbol{x}}_i)$$

Base Encoder



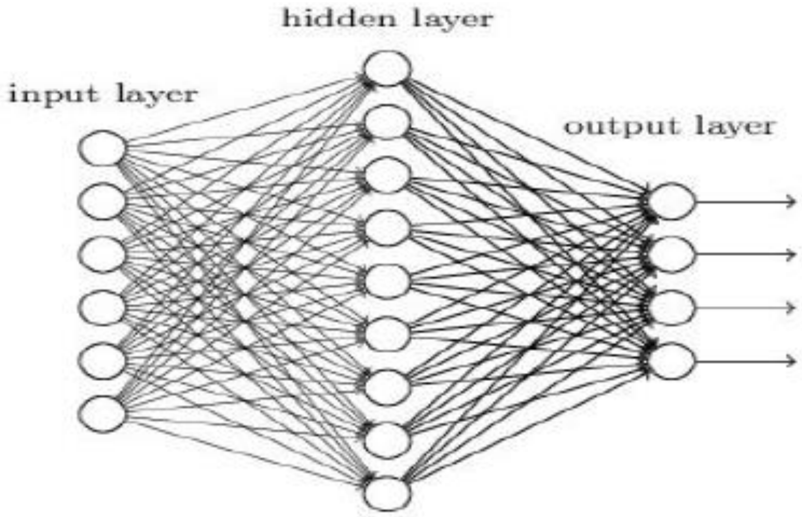
Performance gap shrinks as model size increases

Unsupervised learning benefits more from bigger models



g(h) is a projection network that project representation to a latent space.

We use a 2-layer non-linear MLP (fully connected net).



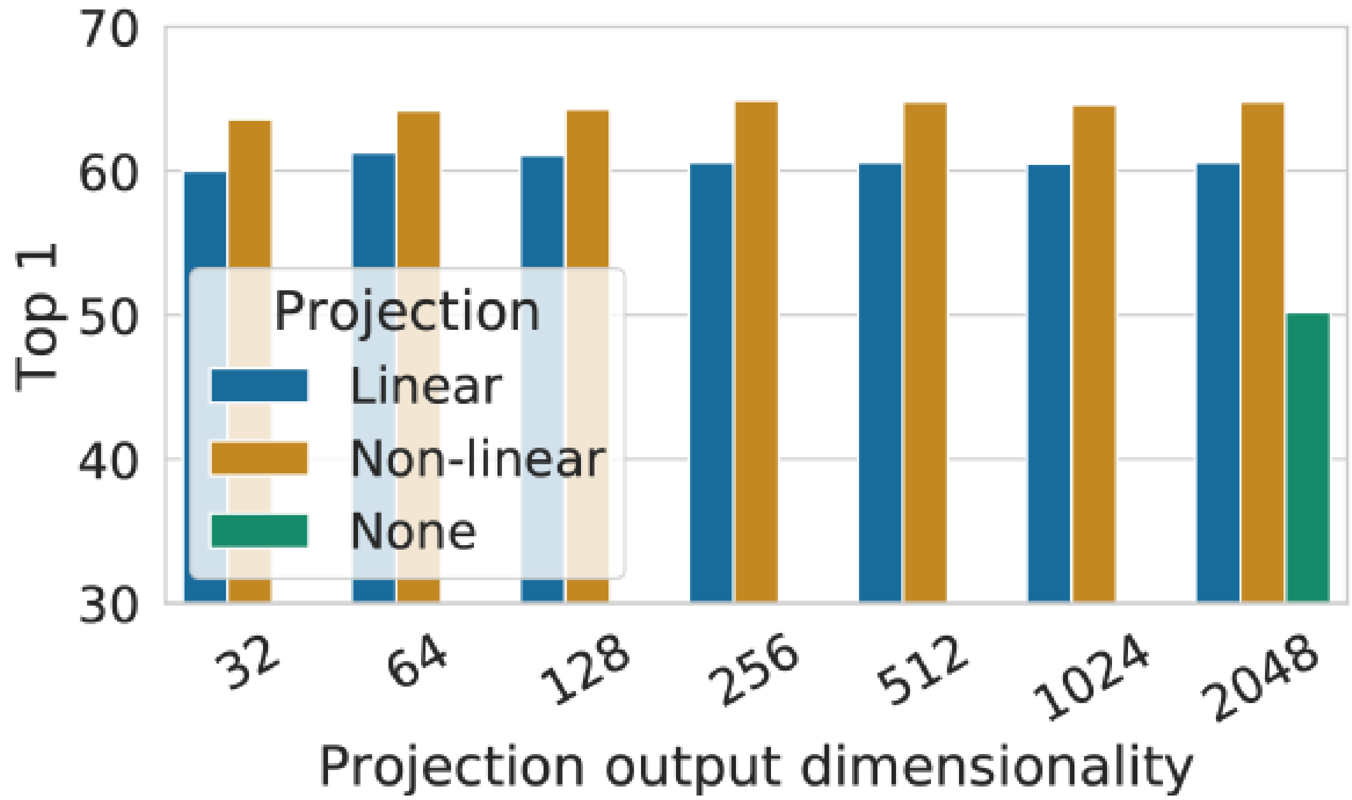
A small neural network

Multilayer Perceptron (MLP)

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

σ is ReLU (non-linearity)

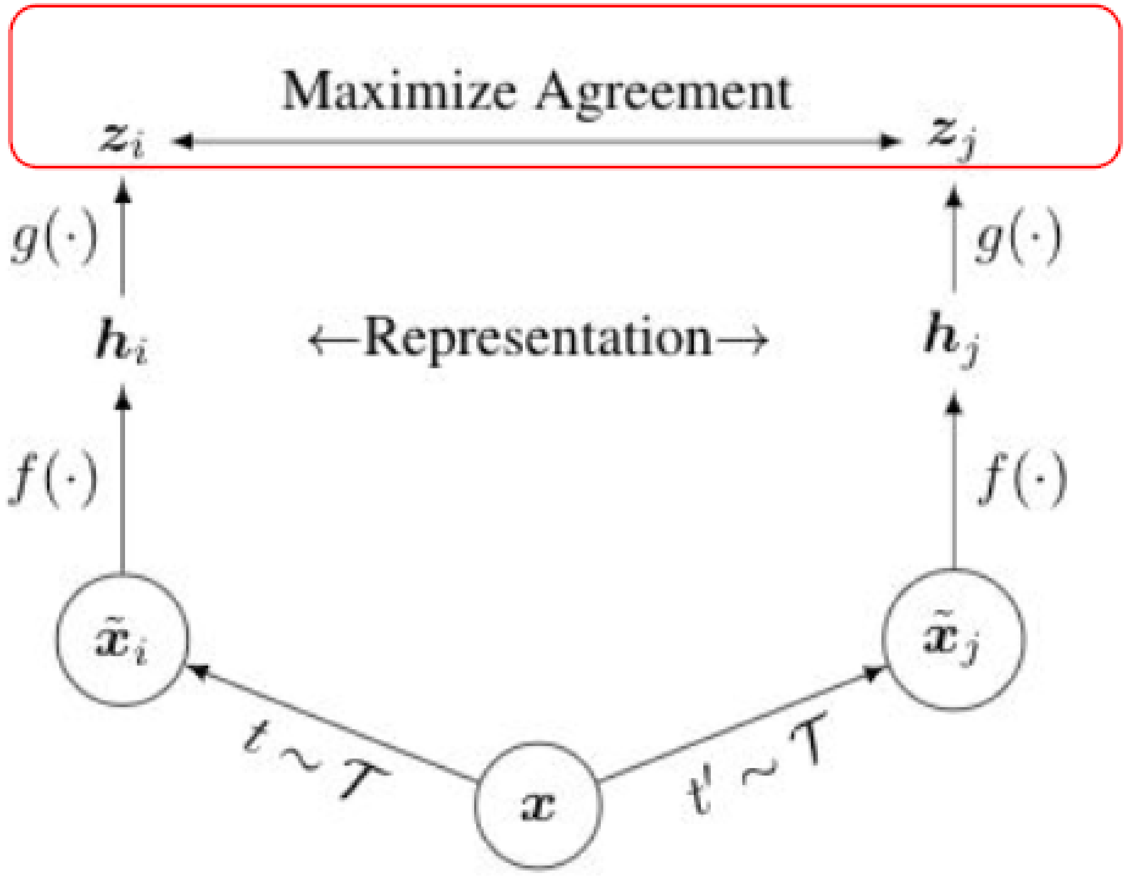
Projection Head



Non-Linear > Linear >> None

Maximize agreement using a contrastive task:

Given {x_k} where two different examples x_i and x_j are a positive pair, identify x_j in {x_k}_{k!=i} for x_i.



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

Loss function:

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

SimCLR pseudo code

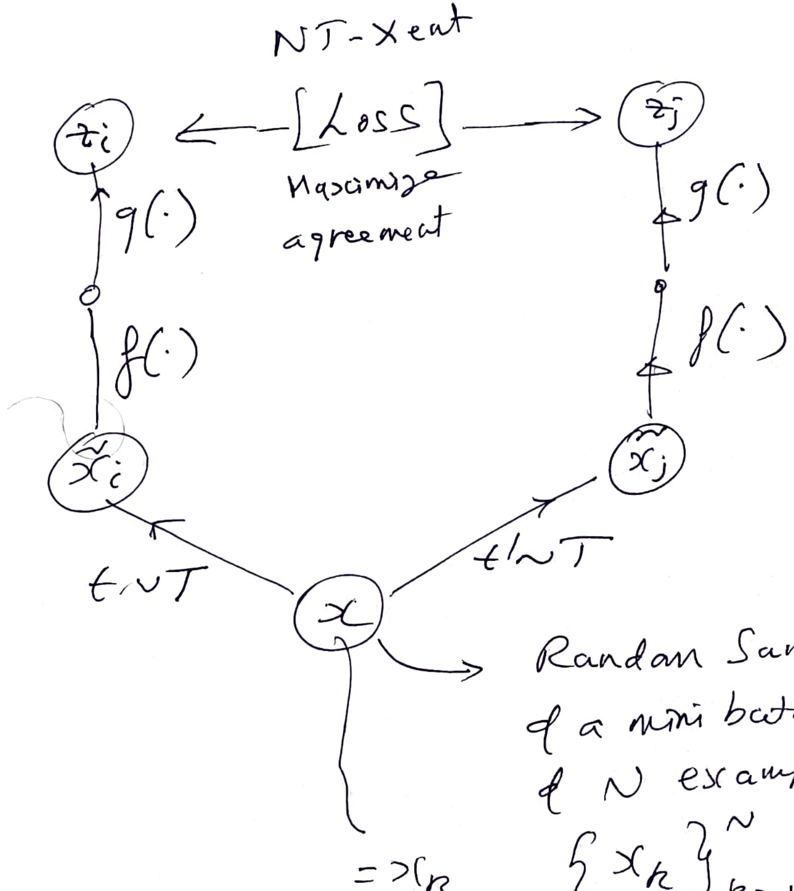
Algorithm 1 SimCLR's main learning algorithm.

```
input: batch size N, temperature τ, form of f, g, T.
for sampled mini-batch {x_k}_{k=1}^N do
    for all k ∈ {1, ..., N} do
        draw two augmentation functions t∼T, t'∼T
        # the first augmentation
        x̃_{2k−1} = t(x_k)
        h_{2k−1} = f(x̃_{2k−1})                      # representation
        z_{2k−1} = g(h_{2k−1})                      # projection
        # the second augmentation
        x̃_{2k} = t'(x_k)
        h_{2k} = f(x̃_{2k})                          # representation
        z_{2k} = g(h_{2k})                          # projection
    end for
    for all i ∈ {1, ..., 2N} and j ∈ {1, ..., 2N} do
        s_{i,j} = z_i^⊤ z_j / (τ‖z_i‖‖z_j‖)         # pairwise similarity
    end for
    define ℓ(i, j) as −s_{i,j} + log Σ_{k=1}^{2N} 1_[k≠i] exp(s_{i,k})
    L = 1/(2N) Σ_{k=1}^{N} [ℓ(2k−1, 2k) + ℓ(2k, 2k−1)]
    update networks f and g to minimize L
end for
return encoder network f
```

$$\ell(i,j) = -s_{i,j} + \log \sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k})$$

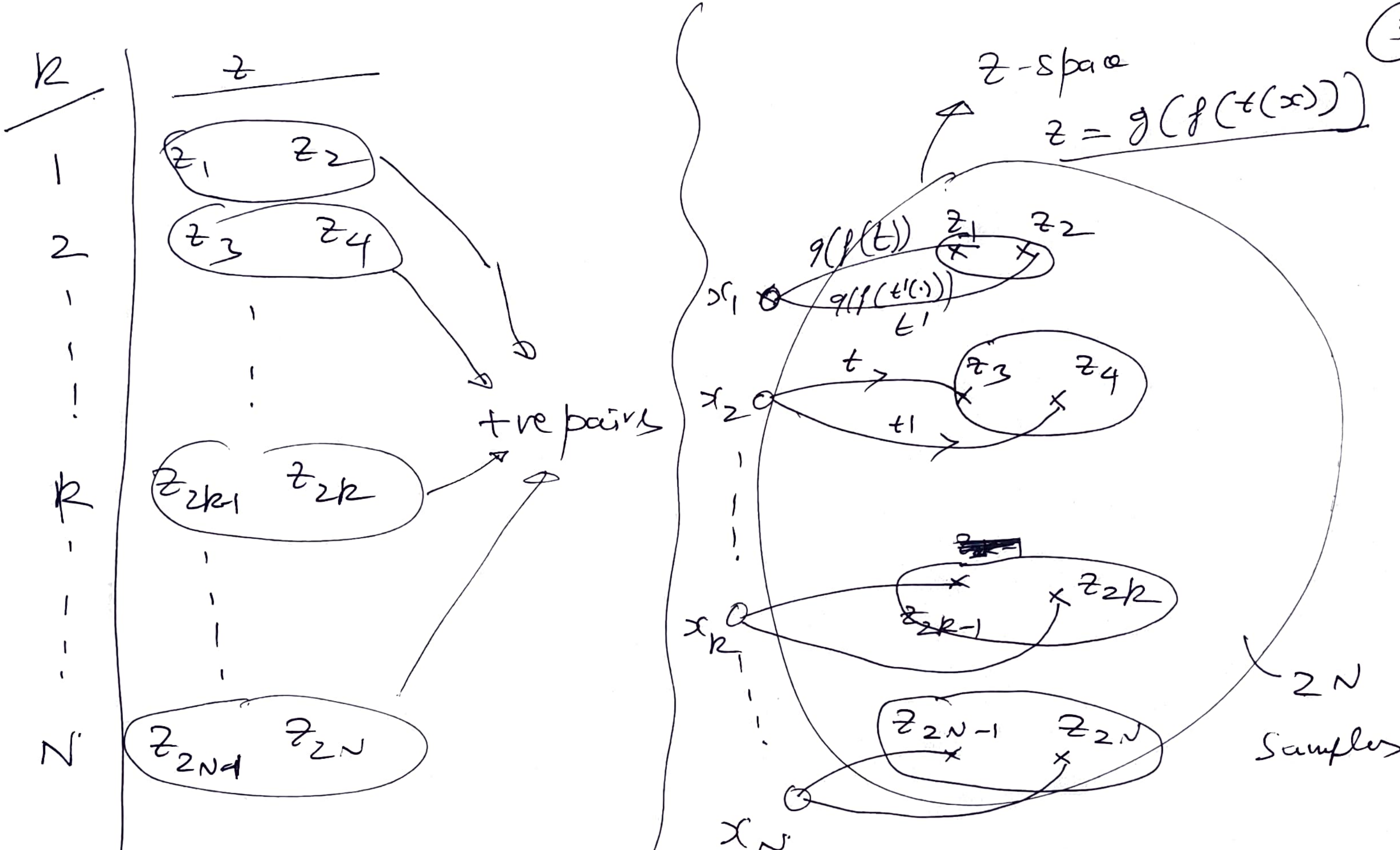
$$\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} \left[\ell(2k-1, 2k) + \ell(2k, 2k-1)\right]$$

SimCLR



Random Sampling of a mini batch of N examples $\{x_k\}_{k=1}^{N}$

k	x_k		$t(x_k)$	$t'(x_k)$		$z = g(f(\tilde{x}))$	
1	x_1		$\tilde{x}_1$	$\tilde{x}_2$		z_1	z_2
2	x_2		$\tilde{x}_3$	$\tilde{x}_4$		z_3	z_4
$\vdots$	$\vdots$		$\vdots$	$\vdots$		$\vdots$	$\vdots$
k	x_k	Transform $\longrightarrow \boxed{t, t'} \longrightarrow$ Augment	$\tilde{x}_{2k-1}$	$\tilde{x}_{2k}$	$\xrightarrow{g(f(\cdot))}$	z_{2k-1}	z_{2k}
$\vdots$	$\vdots$		$\vdots$	$\vdots$		$\vdots$	$\vdots$
N	x_N		$\tilde{x}_{2N-1}$	$\tilde{x}_{2N}$		z_{2N-1}	z_{2N}



Set

$i = \{1, \dots, i, \dots 2N\}$

$j = \{1, \dots, \dots j \dots 2N\}$

$\forall\, i = \{1, \ldots i, \ldots 2N\}$ & $j = \{1, \ldots, j, \ldots 2N\}$

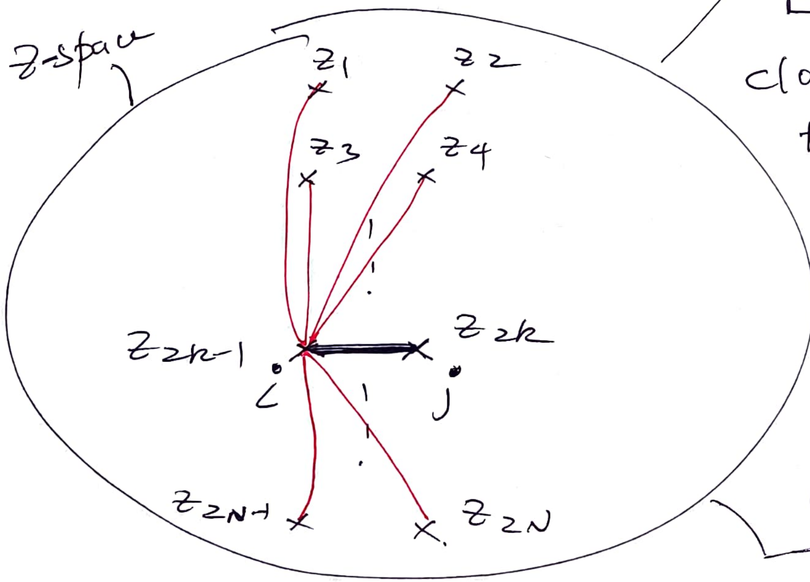
Define & Compute <u>Cosine Similarity</u> [Dot product between L_2 normalized z_i, z_j]

$$s_{ij} = \frac{z_i^T z_j}{\|z_i\| \, \|z_j\|} \qquad \forall\, i, j \in [1, \ldots, 2N]$$

<u>+ve pairs</u>

$i = 2k-1$

$j = 2k$

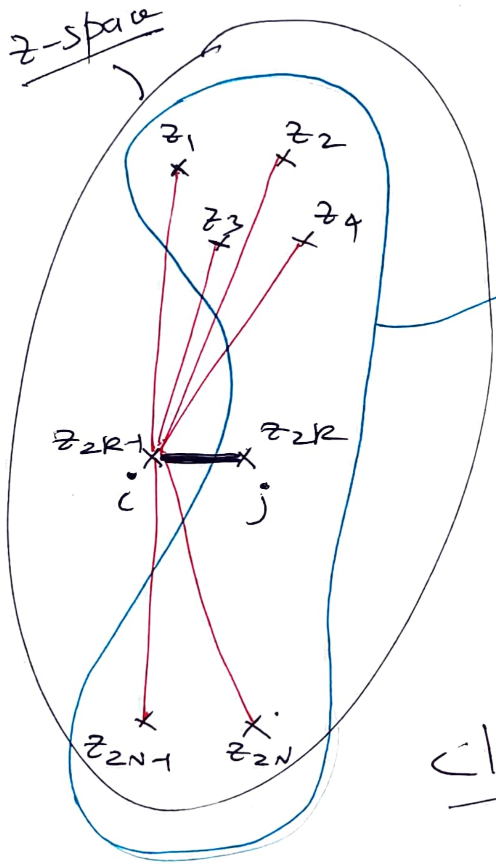


<u>Contrastive loss</u>

- Reduce
- Increase

Ensure $\tilde{x}_j$ is closer to $\tilde{x}_i$ than all $\tilde{x}_k$, $k \neq i, k \neq j$

Identify $\tilde{x}_j$ in $\{\tilde{x}_k\}$ for a given $\tilde{x}_i$



$$\ell(\overset{2k-1}{i}, \overset{2k}{j}) = -\log \frac{e^{\frac{s_{ij}}{\tau}}}{\sum\limits_{\substack{m=1 \\ m \neq i}}^{2N} e^{\frac{s_{im}}{\tau}}}$$

Range of index m

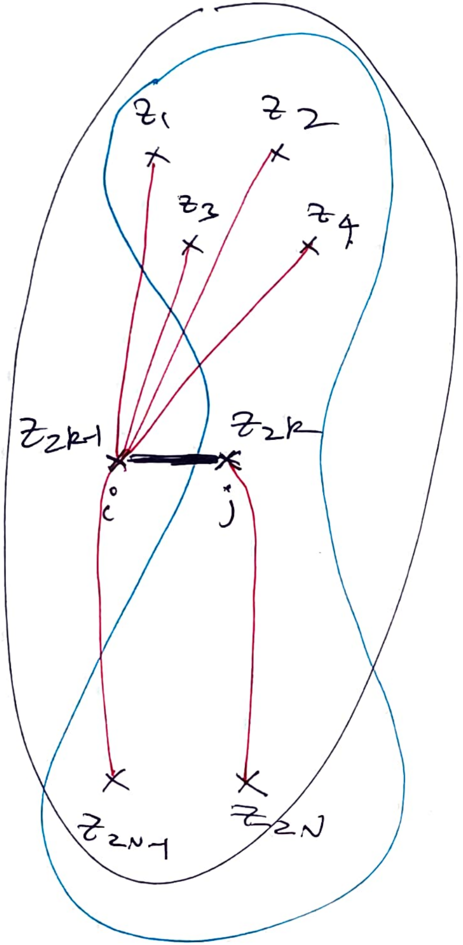
τ: Temperature

$\ell(\cdot,\cdot)$: Normalized Temperature-Scaled Cross-Entropy Loss

Closer $i \Leftrightarrow j$ or $j \Leftrightarrow i$ $\quad s_{ij} \uparrow \rightarrow 1$

$\longrightarrow e^{s_{ij}} \uparrow$

Farther $i \Leftrightarrow m$ $\quad s_{im} \downarrow \rightarrow 0$ or $-1 \rightarrow e^{s_{im}} \Downarrow$



$$\ell(i,j) = -\log \frac{e^{\frac{s_{ij}}{\tau}}}{e^{\frac{s_{ij}}{\tau}} + \sum\limits_{\substack{m=1 \\ m \neq j \\ m \neq i}}^{2N} e^{\frac{s_{im}}{\tau}}}$$

+ve pair CS

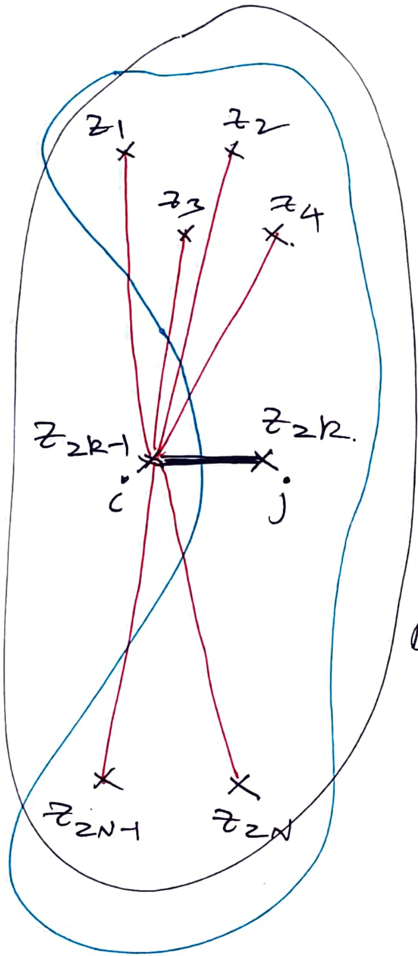
+ve pair CS

−ve pair CS

$$\ell(i,j) = -\log \frac{CS^{+}}{CS^{+} + CS^{-}}$$

Sum of 2N-2 terms

$$= \frac{\text{Black-Line}}{\text{Black-Line} + \sum\limits_{2N-2} \text{Red-Lines}}$$



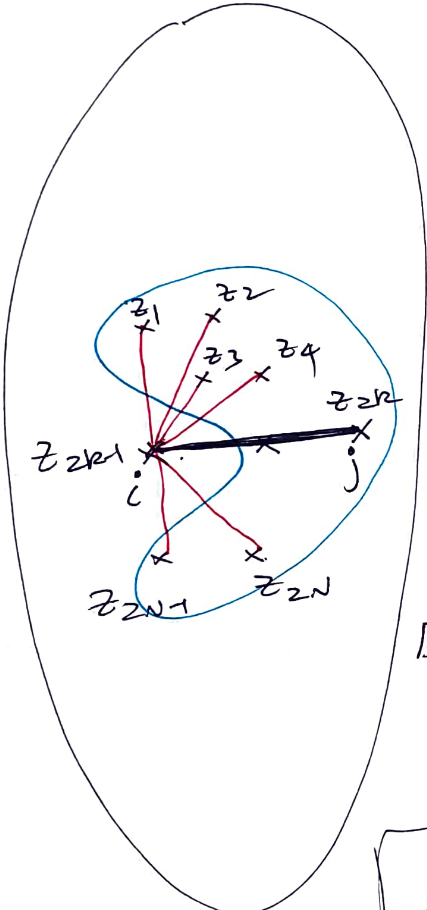
Identify $\tilde{x}_j$ (ie z_{2k}) in $\{\tilde{x}_k\}$ for a given $\tilde{x}_i \to z_{2k-1}$

Ensure $\tilde{x}_j$ is closer to $\tilde{x}_i$ than all $\tilde{x}_k$, $k \neq i$, $k \neq j$

How

1. $\{x_i, x_j\}$ are the pairs e closer than $\{x_i, x_m\}$

$$\frac{\overbrace{CS^+}^{\text{Black-Line}}}{\underbrace{CS^+}_{\text{Black-Line}} + \underbrace{CS^-}_{\sum \text{Red-Lines}}} = \frac{\text{High}}{\text{High} + \text{Low}} < 1 \to 1$$



2. when $\{x_i, x_j\}$ the pairs are not yet close

[when $g(f(\cdot))$ is not effective yet]

$$\frac{CS^+}{\underbrace{CS^+}_{\text{Black-line}} + \underbrace{CS^-}_{\Sigma\ \text{Red-lines}}} = \frac{Low}{Low + High} \sim 0 \to 0$$

$$\boxed{\ell(i,j) = -\log \frac{CS^+}{CS^+ + CS^-}}$$

Contrasting

+ve pairs w.r.t -ve pairs

Poor embeddings //learnings of $f(\cdot)$ e $g(\cdot)$

$$\frac{\text{Low}}{\text{Low} + \text{High}}$$

Good embeddings of $f(\cdot)$ e $g(\cdot)$ & learning with epochs.

$$\frac{\text{High}}{\text{High} + \text{Low}}$$

Higher the better

Loss

$$l(i,j) = -\log \left[\frac{e^{cs^{+}/\tau}}{e^{cs^{+}/\tau} + \sum\limits_{\substack{\text{Red lines} \\ (2N-2) \text{ terms}}} e^{\frac{cs^{-}}{\tau}}} \right] = -\log \left[\frac{\text{Black-Line}}{\text{Black-Line} + \sum\limits_{\substack{2N-2 \\ \text{terms}}} \text{Red lines}} \right]$$

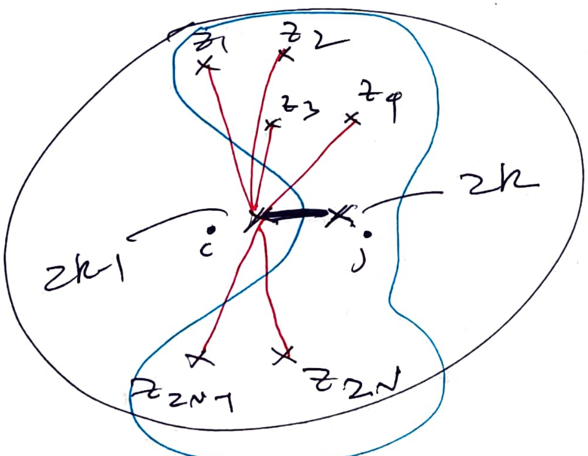
Lower the Better

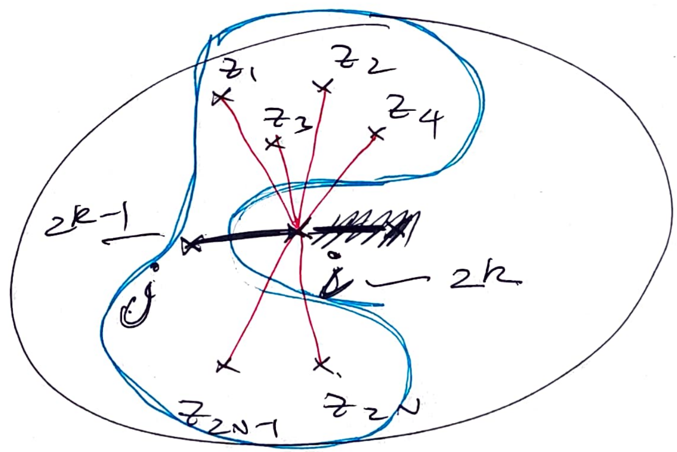
Since

$$l(i,j) \neq l(j,i)$$

Note the Denominator difference on the $\sum$ Red lines

$$L = \frac{1}{2N} \sum_{k=1}^{N} \left[l(\underset{i}{2k-1}, \underset{j}{2k}) + l(\underset{i}{2k}, \underset{j}{2k-1}) \right]$$





SimCLR pseudo code

Algorithm 1 SimCLR's main learning algorithm.

input: batch size N, temperature τ, form of f, g, $\mathcal{T}$.
for sampled mini-batch $\{\boldsymbol{x}_k\}_{k=1}^{N}$ **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** $-s_{i,j} + \log \sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k})$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network f

Thank you !!