# Lecture 12: Self-Supervised Learning

# The Promises of Unsupervised Learning

- The promise of unsupervised learning
  - Learn representations of data from unlabeled data
  - Use a bit of supervision to solve many tasks

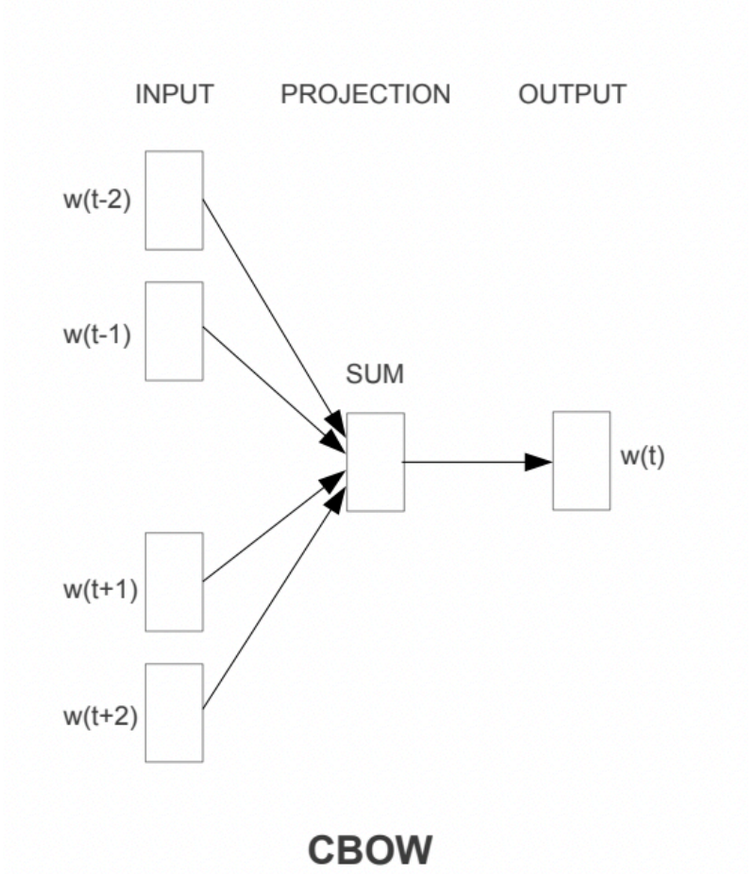
# Self-Supervision

- Self-Supervision is a class of unsupervised learning methods
  - In general it means we construct a supervised learning problem using the unlabeled data (e.g. withholding part of it and predicting another)
  - Definition can vary slightly
  - Typically distinct from generative modelling approaches to unsupervised representation learning

# Naming

- Self-supervision is an ubiquitous unsupervised representation learning method in NLP
  - Word2vec
  - BERT
  - Skip-thoughts
- The term “self-supervision” however is more used in applications in computer vision and speech

# Word2Vec

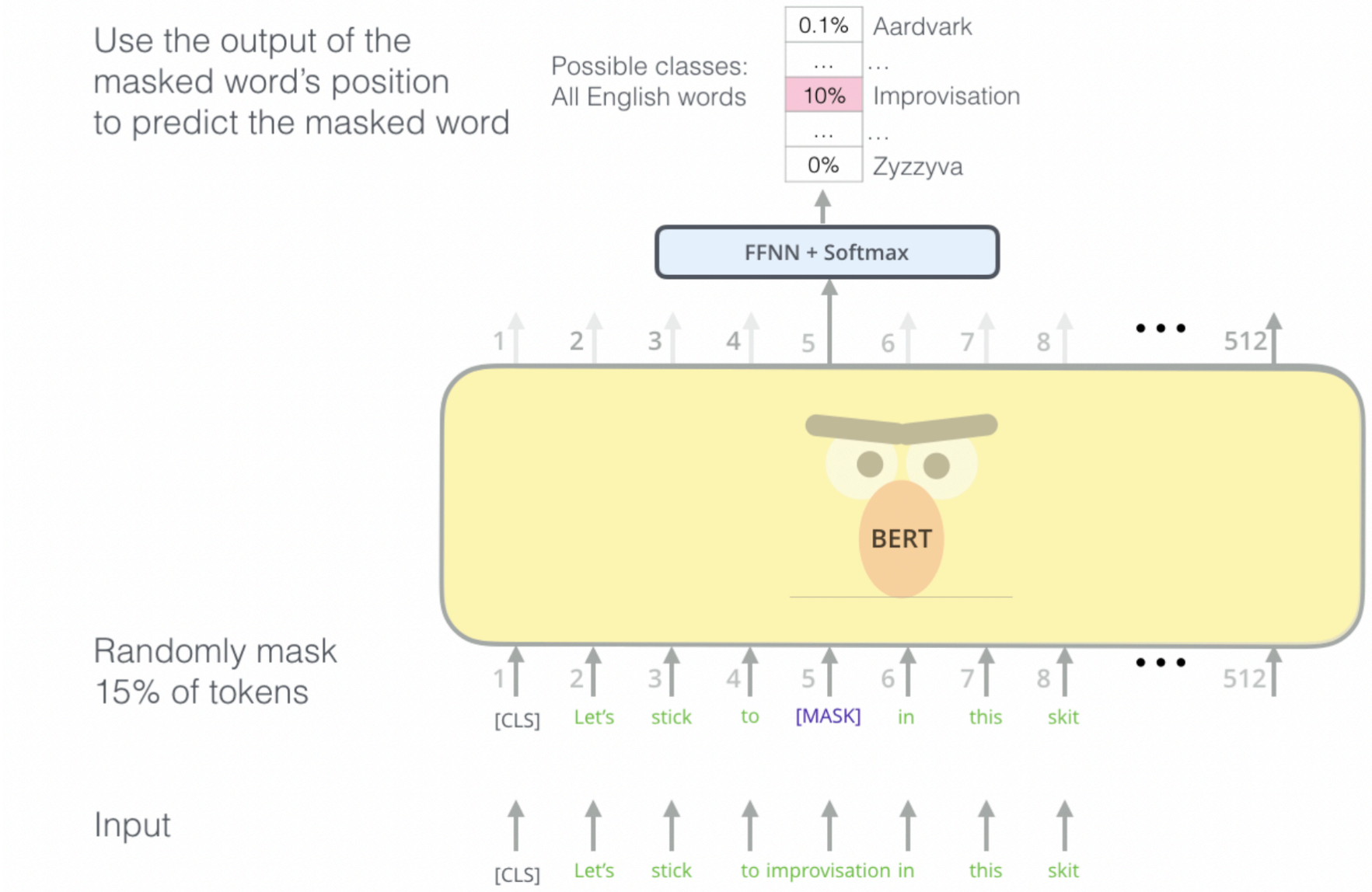



**I love this _____ and I've seen it many times**

**many this _____ love I I've it and times seen**

- word2vec CBOW model: predict a word given its neighbors
- Use data itself to form the prediction problem

# BERT




- BERT essentially uses the same idea as CBOW but with fancy non-linear and order aware modeling

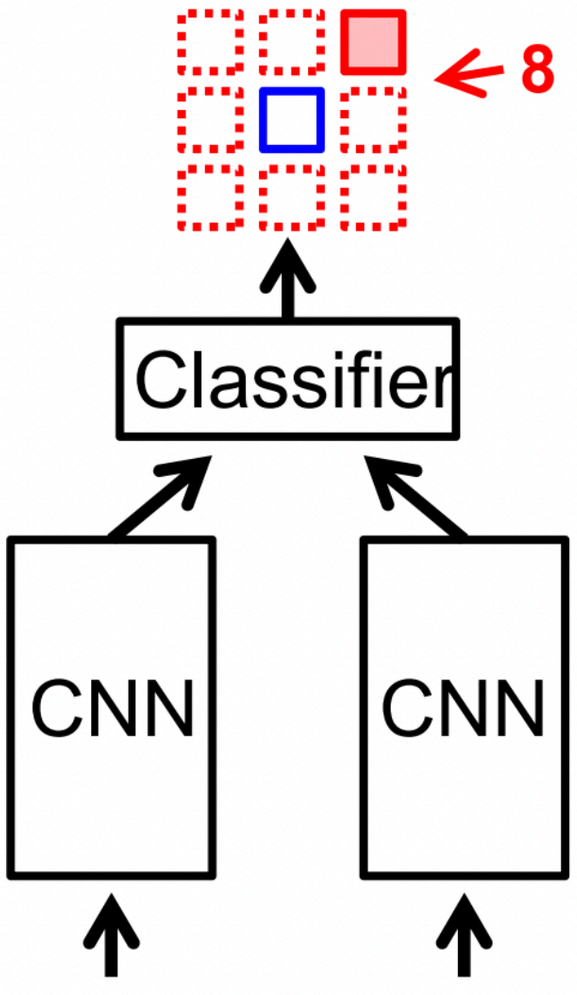
- Allows fine-tuning on downstream language tasks

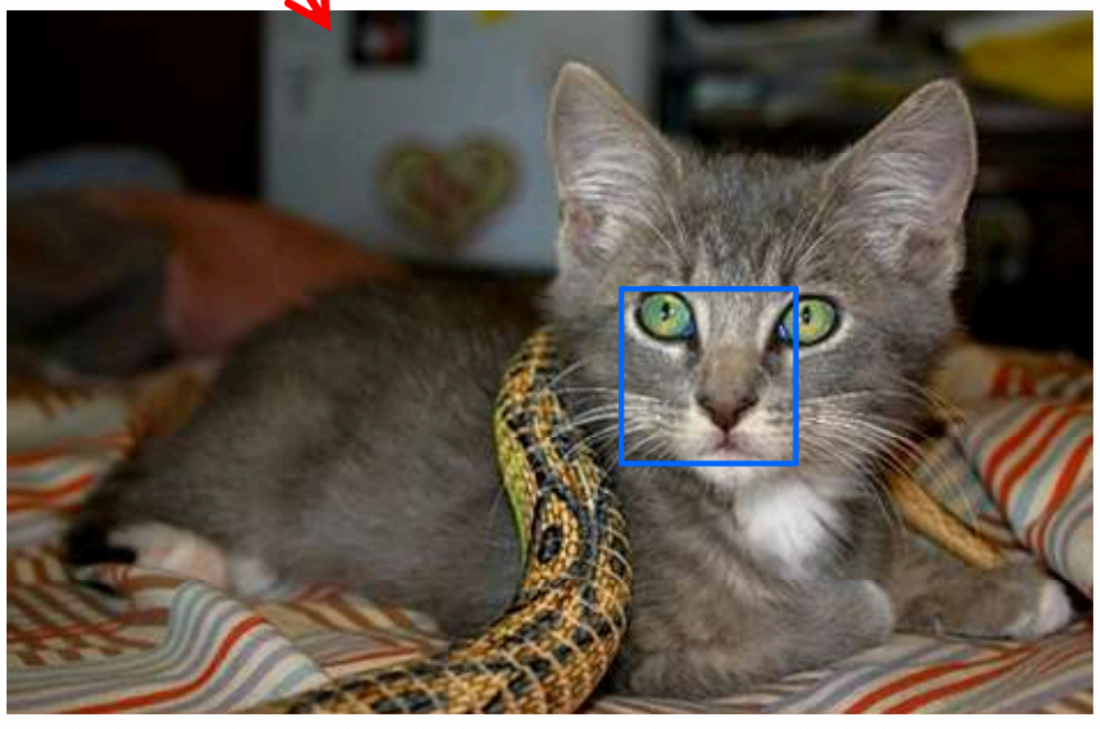
# Unsupervised learning in vision

- Progress in unsupervised learning for vision was rather slow until ~2018
- Despite great success in producing visually coherent images generative model based representations have not been fruitful for downstream tasks especially when compared to supervised representation learning

# Auxiliary Task: Relative Position

Train network to predict relative position of two regions in the same image



**Randomly Sample Patch**

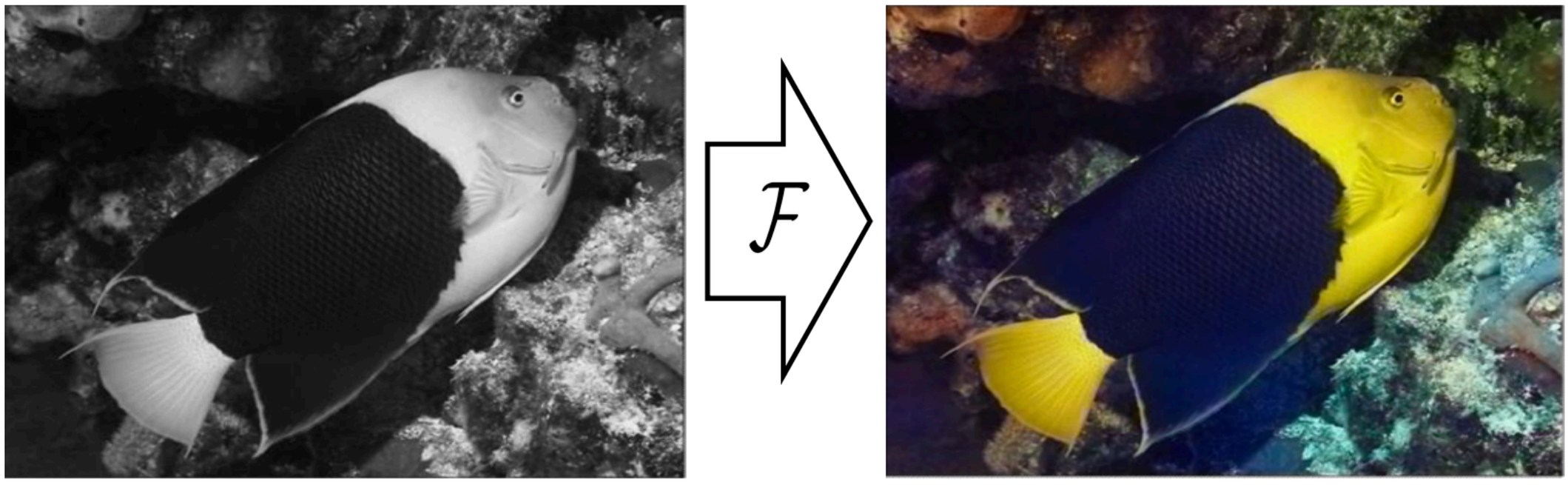
**Sample Second Patch**

Unsupervised visual representation learning by context prediction,
Carl Doersch, Abhinav Gupta, Alexei A. Efros, ICCV 2015

**Slide Credit: Andrew Zisserman**

# Auxiliary Task: Color Self-Supervision

Train network to predict pixel colour from a monochrome input

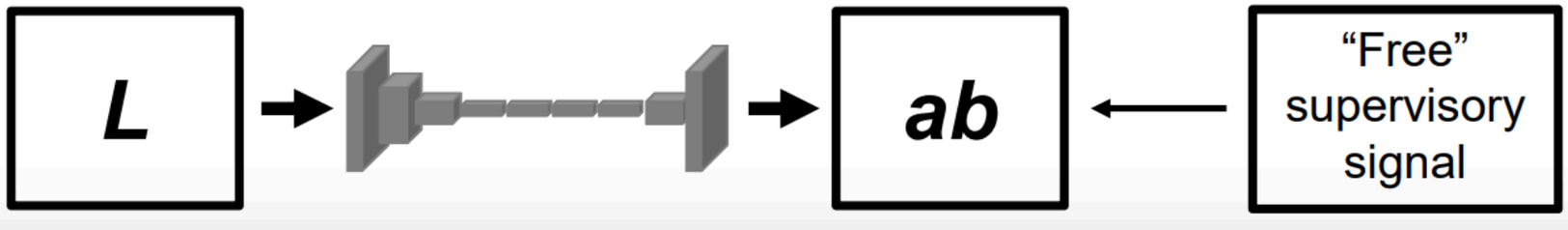



Grayscale image: *L* channel

$$\mathbf{X} \in \mathbb{R}^{H \times W \times 1}$$

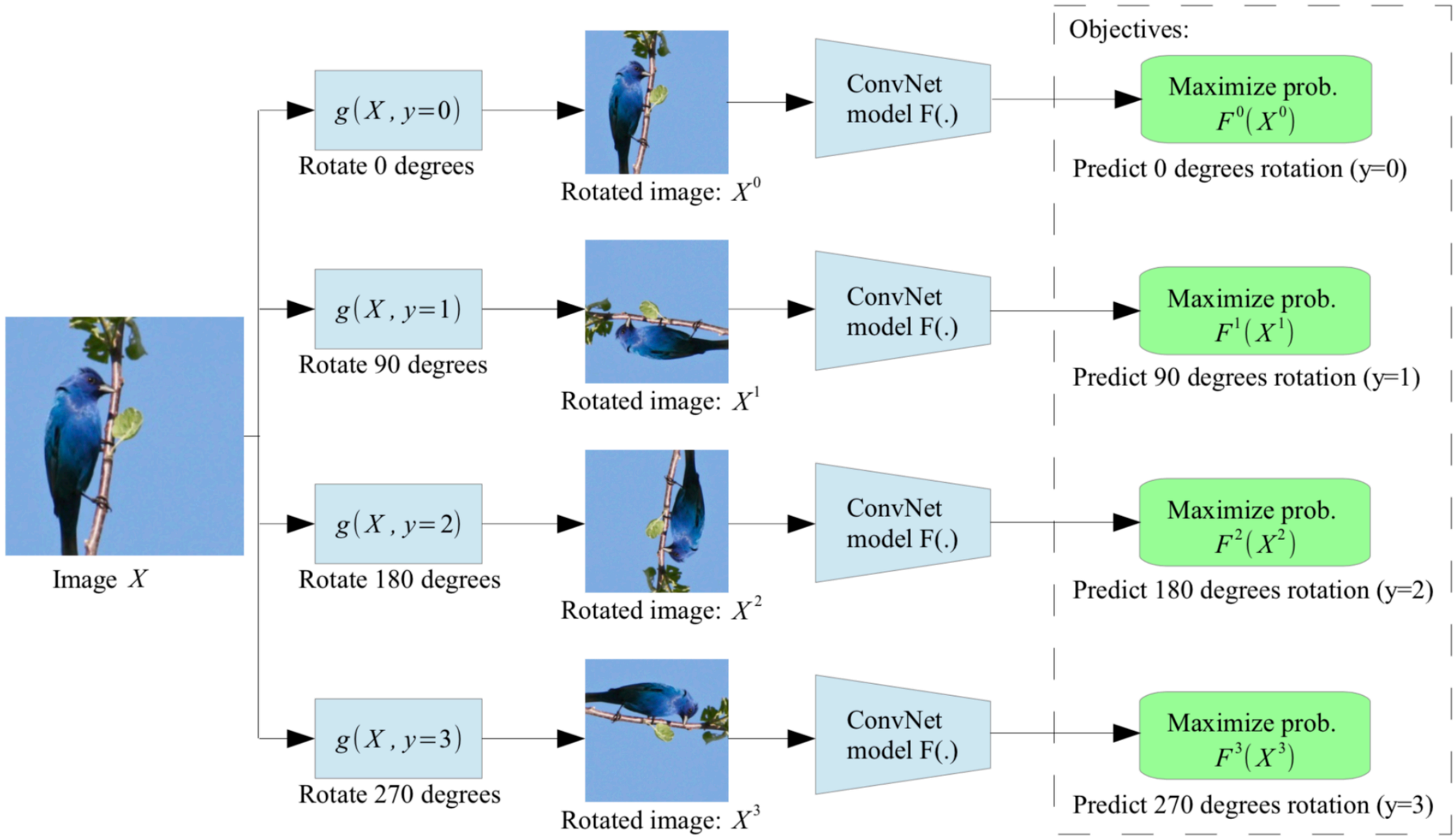
Concatenate (L,ab)

$$(\mathbf{X}, \hat{\mathbf{Y}})$$




**Slide Credit: Andrew Zisserman**

# Rotation




## UNSUPERVISED REPRESENTATION LEARNING BY PREDICTING IMAGE ROTATIONS

**Spyros Gidaris, Praveer Singh, Nikos Komodakis**
University Paris-Est, LIGM
Ecole des Ponts ParisTech
{spyros.gidaris,praveer.singh,nikos.komodakis}@enpc.fr

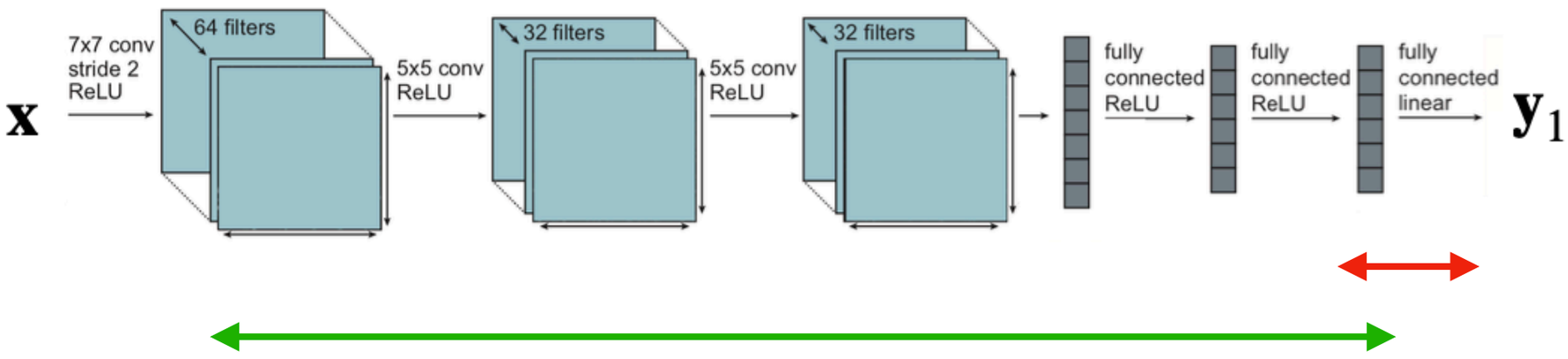
# Evaluating

- Evaluating unsupervised learning algorithms isn't a well defined problem
- Generative Models
  - We can evaluate by visual quality
  - We can evaluate by likelihood — for some models a bound on it
  - We can use the learned features for downstream tasks
- Self-supervised learning
  - Typically assumes a downstream task such as classification
  - Evaluation can be done by transfer learning or direct use of learned features in supervised tasks

# Evaluating

- Common evaluation framework on image classification used in literature
  - Approach 1 — used with Imagenet/CIFAR-10 data
    - Train model unsupervised (e.g. imagenet data without labels + CNN)
    - Use trained model (all parameters frozen) to give features to a linear classifier
    - Sometimes a non-linear classifier such as a small MLP is used as well
  - Approach 2 — when another task data is available (more realistic scenario)
    - Train the model unsupervised  (e.g. imagenet data without labels + CNN)
    - Fine-tune the model on a new task (e.g. detection)
- In both of these evaluations the goal is to get similar performance as starting from a large scale supervised model
  - Remember we have more unlabeled data by orders of magnitude then labeled

# Evaluating Unsupervised Learning




Imagenet
Unlabeled Train Set

Imagenet labeled
Train Set

# Evaluating

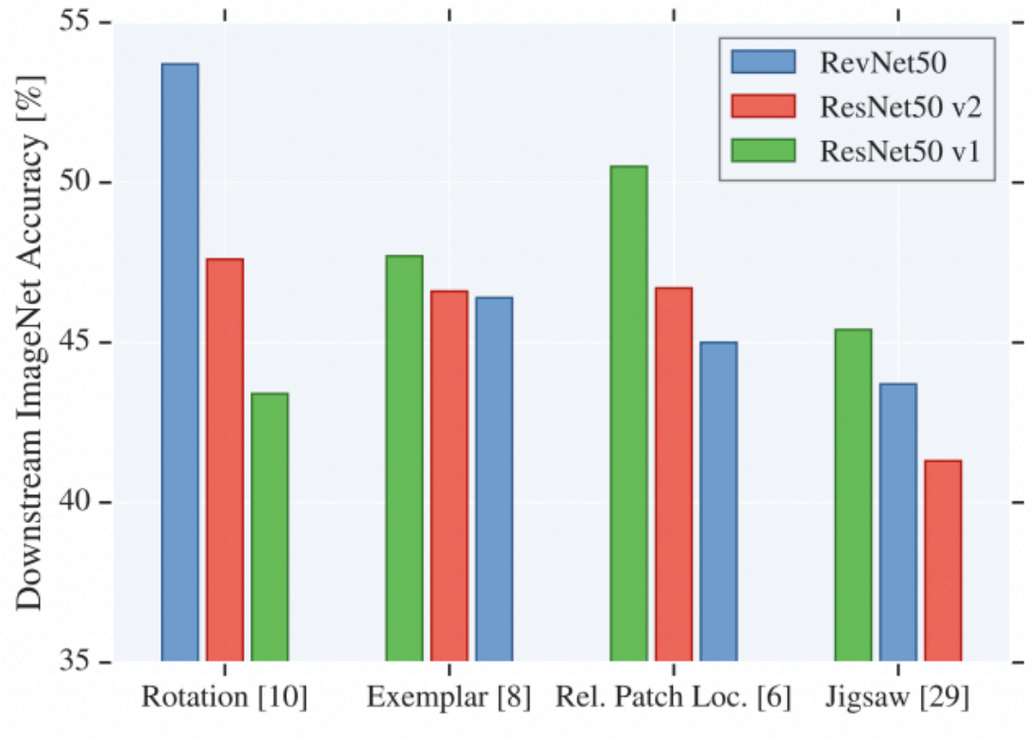



Table 5: **Task Generalization: ImageNet top-1 classification with linear layers**. We compare our unsupervised feature learning approach with other unsupervised approaches by training logistic regression classifiers on top of the feature maps of each layer to perform the 1000-way ImageNet classification task, as proposed by Zhang et al. (2016a). All weights are frozen and feature maps are spatially resized (with adaptive max pooling) so as to have around 9000 elements. All approaches use AlexNet variants and were pre-trained on ImageNet without labels except the ImageNet labels and Random entries.

| Method | Conv1 | Conv2 | Conv3 | Conv4 | Conv5 |
|---|---|---|---|---|---|
| ImageNet labels | 19.3 | 36.3 | 44.2 | 48.3 | 50.5 |
| Random | 11.6 | 17.1 | 16.9 | 16.3 | 14.1 |
| Random rescaled Krähenbühl et al. (2015) | 17.5 | 23.0 | 24.5 | 23.2 | 20.6 |
| Context (Doersch et al., 2015) | 16.2 | 23.3 | 30.2 | 31.7 | 29.6 |
| Context Encoders (Pathak et al., 2016b) | 14.1 | 20.7 | 21.0 | 19.8 | 15.5 |
| Colorization (Zhang et al., 2016a) | 12.5 | 24.5 | 30.4 | 31.5 | 30.3 |
| Jigsaw Puzzles (Noroozi & Favaro, 2016) | 18.2 | 28.8 | 34.0 | 33.9 | 27.1 |
| BIGAN (Donahue et al., 2016) | 17.7 | 24.5 | 31.0 | 29.9 | 28.0 |
| Split-Brain (Zhang et al., 2016b) | 17.7 | 29.3 | 35.4 | 35.2 | 32.8 |
| Counting (Noroozi et al., 2017) | 18.0 | 30.6 | 34.3 | 32.5 | 25.7 |
| (Ours) RotNet | **18.8** | **31.7** | **38.7** | **38.2** | **36.5** |

# Multi-Task Self-Supervised Learning

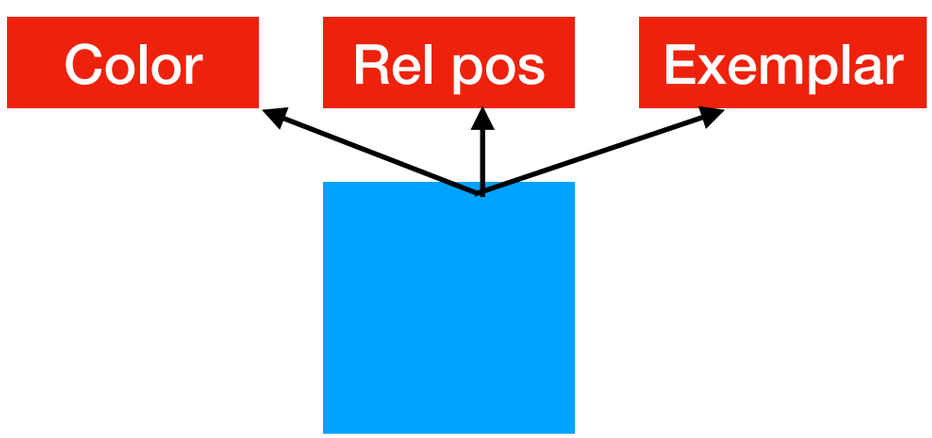
## Multi-Task Self-Supervised Learning

Procedure:

- ImageNet-frozen: self-supervised training, network fixed, classifier trained on features
- PASCAL: self-supervised pre-training, then train Faster-RCNN
- ImageNet labels: strong supervision

NB: all methods re-implemented on same backbone network (ResNet-101)

| Self-supervision task | ImageNet Classification top-5 accuracy | PASCAL VOC Detection mAP |
|---|---|---|
| Rel. Pos | 59.21 | 66.75 |
| Colour | 62.48 | 65.47 |
| Exemplar | 53.08 | 60.94 |
| Rel. Pos + colour | 66.64 | 68.75 |
| Rel. Pos + Exemplar | 65.24 | 69.44 |
| Rel. Pos + colour + Exemplar | 68.65 | 69.48 |
| ImageNet labels | 85.10 | 74.17 |

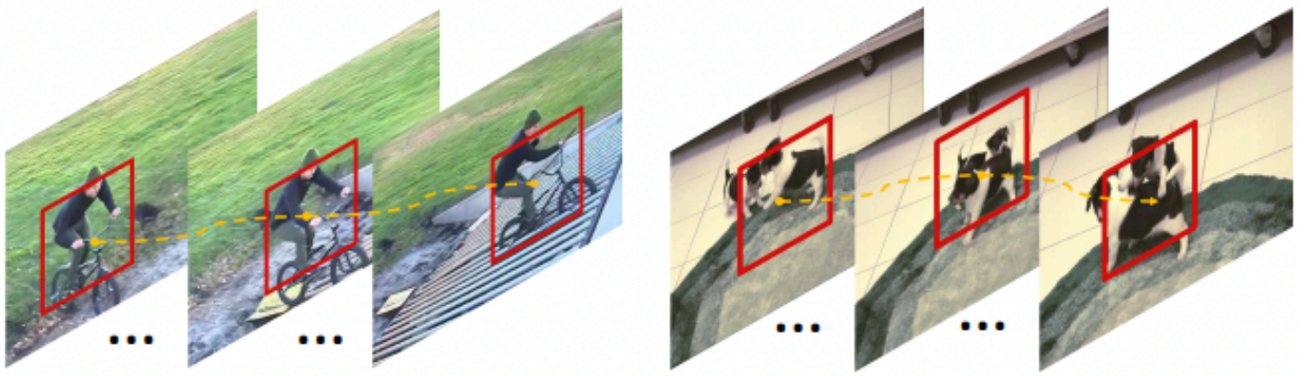



**Slide Credit: Andrew Zisserman**

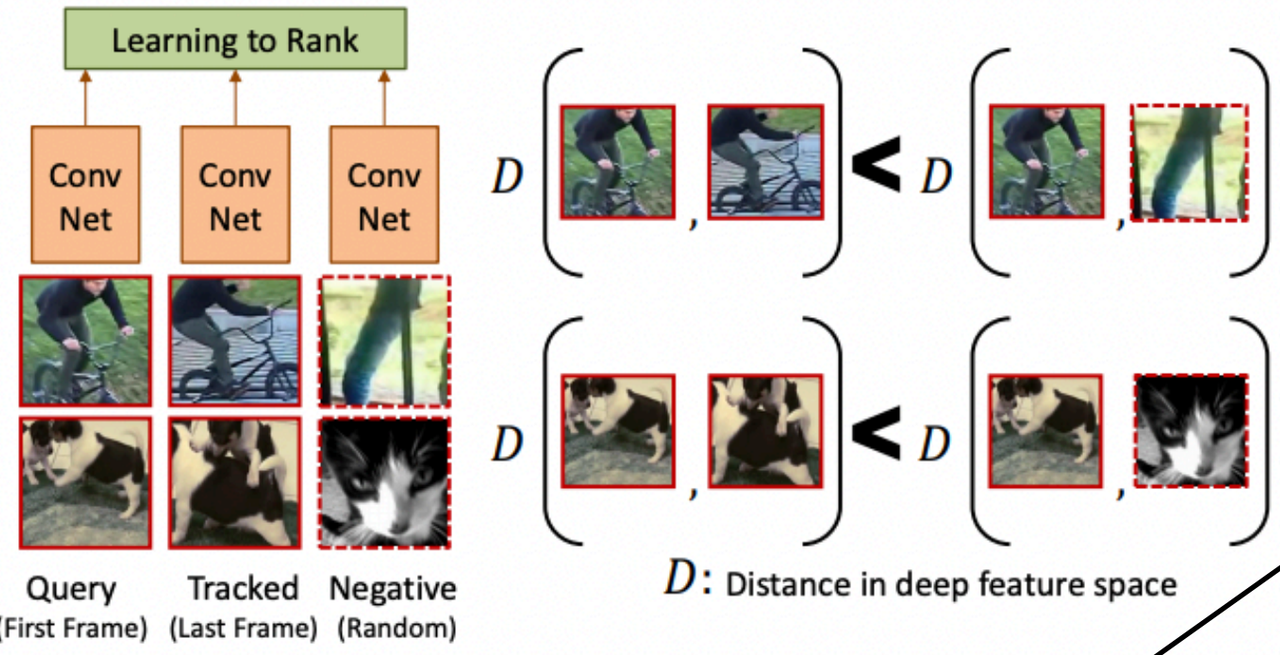
# Metric Learning Loss for Self-Supervised Learning

- Approach to self-supervised methods in video is often based on using triplet loss  or pairwise loss with adjacent frames
- Since 2018-2019 breakthrough self-supervised methods are based on using contrastive losses with augmented versions of data

# Self-Supervised Learning from Video



(a) Unsupervised Tracking in Videos




- Learn model of 2d images using unsupervised video
- Contrastive or triplet loss are used to encourage similarity of close by frames
- Can also use unsuperivised tracking algorithms to further isolate objects
- Model can be used on downstream 2d image tasks
- Similar idea can be applied for multi-view data

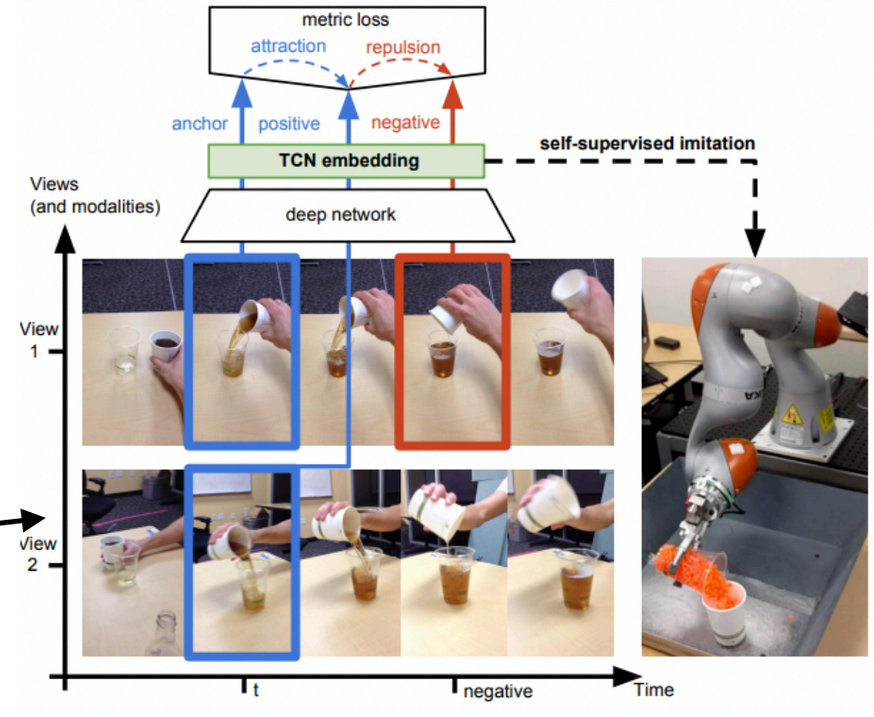
$$\min_W \frac{\lambda}{2} \parallel W \parallel_2^2 + \sum_{i=1}^{N} \max\{0, D(X_i, X_i^+) - D(X_i, X_i^-) + M\},$$

**Unsupervised Learning of Visual Representations using Videos**

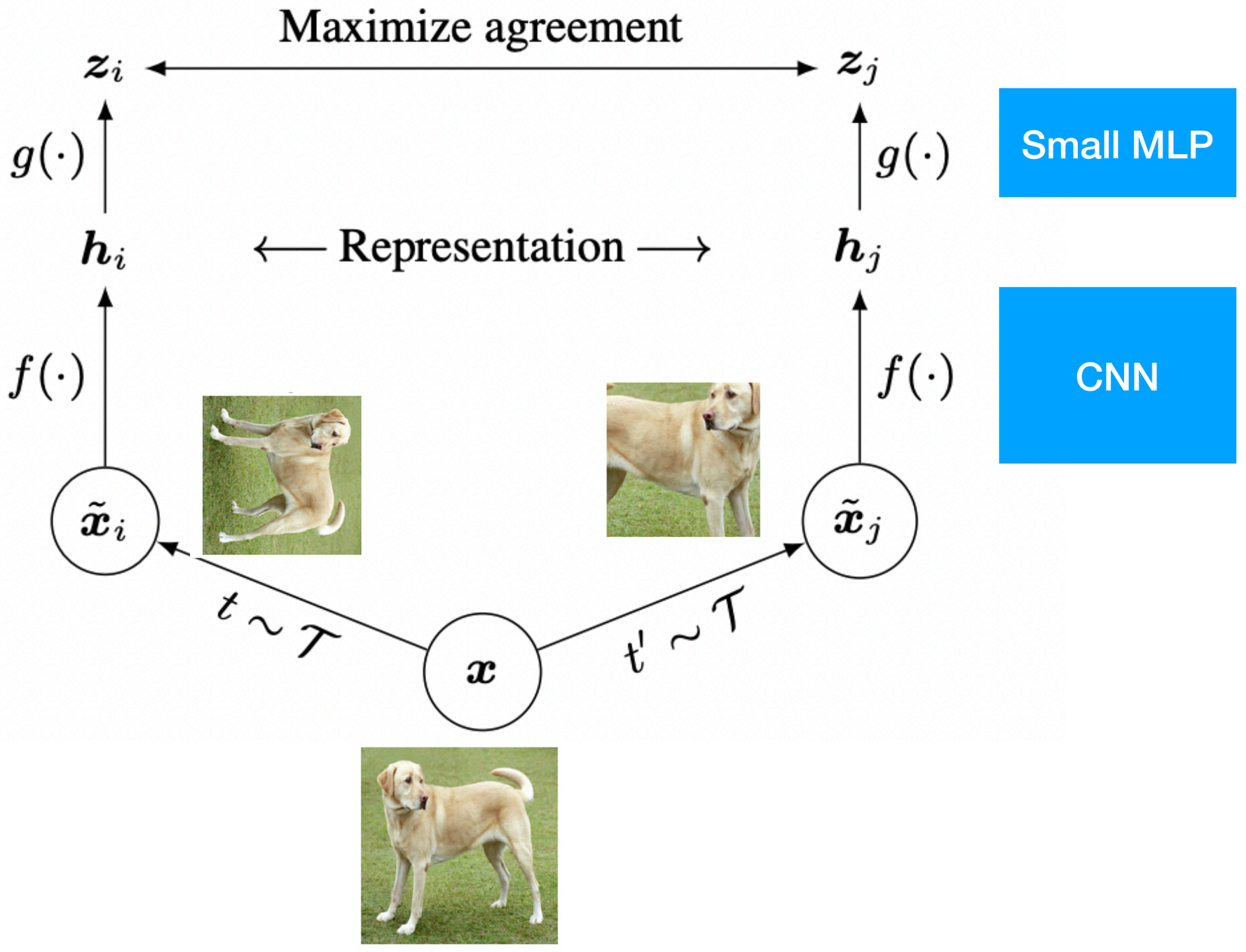
Xiaolong Wang, Abhinav Gupta
Robotics Institute, Carnegie Mellon University

**Time-Contrastive Networks: Self-Supervised Learning from Video**

Pierre Sermanet[1*@] Corey Lynch[1R*] Yevgen Chebotar[2*]
Jasmine Hsu[1] Eric Jang[1] Stefan Schaal[2] Sergey Levine[1]



# SimCLR: Contrastive Loss + Augmentation




Small MLP

CNN

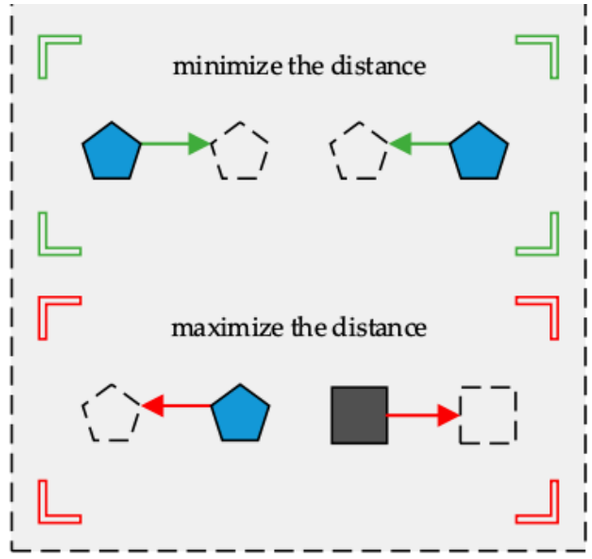
# SimCLR: “Contrastive” Loss + Augmentation

**Recall**

$$z_1 = f_\theta(x_1) \quad z_2 = f_\theta(x_2)$$

$$l(z_1, z_2, y) = y \parallel z_1 - z_2 \parallel + (1 - y)\max(0,\ m - \parallel z_1 - z_2 \parallel)$$

$y - >$ **Same or not**




**We use another metric learning loss:**
**Info Noise Contrastive Estimation**

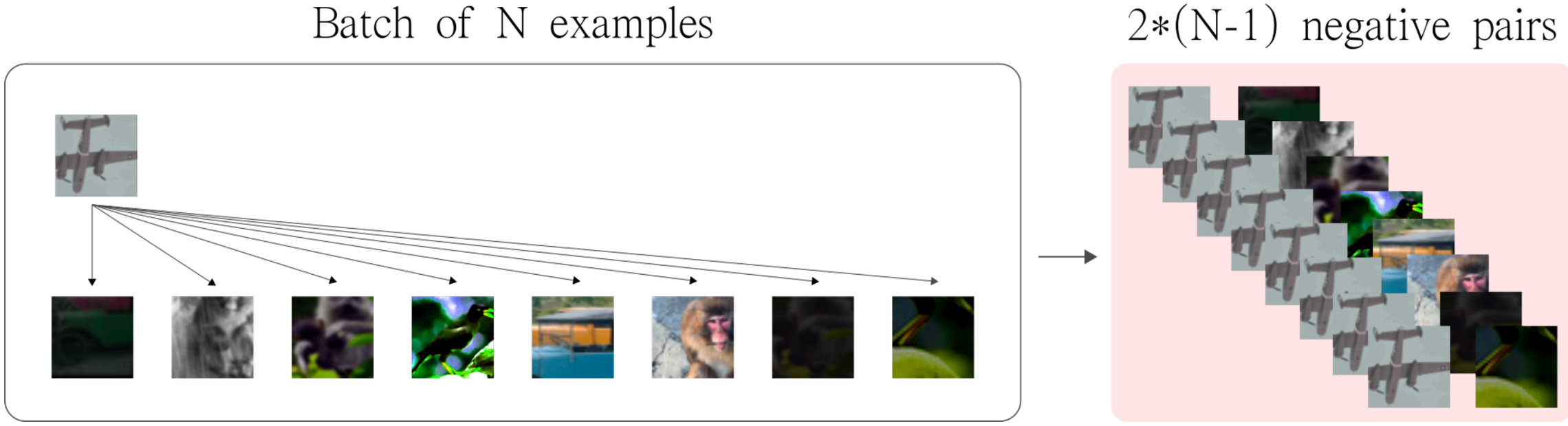
$$\ell_{i,j} = -\log \frac{\exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)}, \qquad \mathrm{sim}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}^\top \boldsymbol{v} / \|\boldsymbol{u}\| \|\boldsymbol{v}\|$$

**N is the batch size**

**2*N augmentations**

- For each  matching pair (z_i , z_j) we have a lot of negative pairs z_i, z_k in the denominator
- Works well with mini batch processing
- Can be implemented using Softmax + CrossEntropy like in classification (Also called Normalized Temperature-scaled Cross Entropy)
    - Another interpretation - for a given image label the augmentation as 1 and , treat as multi class classification
- Distance metric is cosine distance
- T is a temperature parameter allows to compensate for maximum 1 value of cosine similarity

# SimCLR: Efficient Use of Mini-Batch




**Algorithm 1** SimCLR's main learning algorithm.

**input:** batch size $N$, constant $\tau$, 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 $g(\cdot)$

**Most of the Compute Cost Here**

**(2N)^2 Comparisons for 2N forward/backward passes**

Image from: https://sthalles.github.io/simple-self-supervised-learning/

# SimCLR

- Results far exceed previous unsupervised learning on Imagenet and get close to match Supervised learning benchmarks

- A cluster of methods basically doing this same thing with slight variations came out circa 2018-2020 (CPC, Moco, AMDIM).

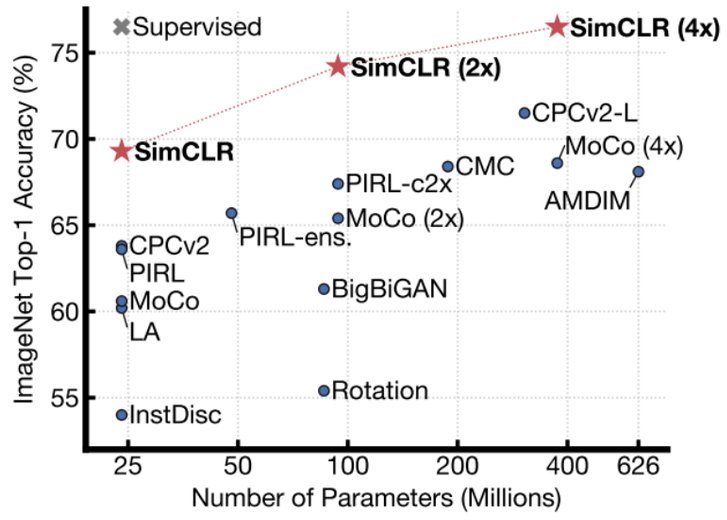



*Figure 1.* ImageNet Top-1 accuracy of linear classifiers trained on representations learned with different self-supervised methods (pretrained on ImageNet). Gray cross indicates supervised ResNet-50. Our method, SimCLR, is shown in bold.

| | Food | CIFAR10 | CIFAR100 | Birdsnap | SUN397 | Cars | Aircraft | VOC2007 |
|---|---|---|---|---|---|---|---|---|
| *Linear evaluation:* | | | | | | | | |
| SimCLR (ours) | **76.9** | **95.3** | 80.2 | 48.4 | **65.9** | 60.0 | 61.2 | **84.2** |
| Supervised | 75.2 | **95.7** | **81.2** | **56.4** | 64.9 | **68.8** | **63.8** | 83.8 |
| *Fine-tuned:* | | | | | | | | |
| SimCLR (ours) | **89.4** | **98.6** | **89.0** | **78.2** | **68.1** | **92.1** | **87.0** | **86.6** |
| Supervised | 88.7 | 98.3 | **88.7** | **77.8** | 67.0 | 91.4 | **88.0** | 86.5 |
| Random init | 88.3 | 96.0 | 81.9 | **77.0** | 53.7 | 91.3 | 84.8 | 69.4 |

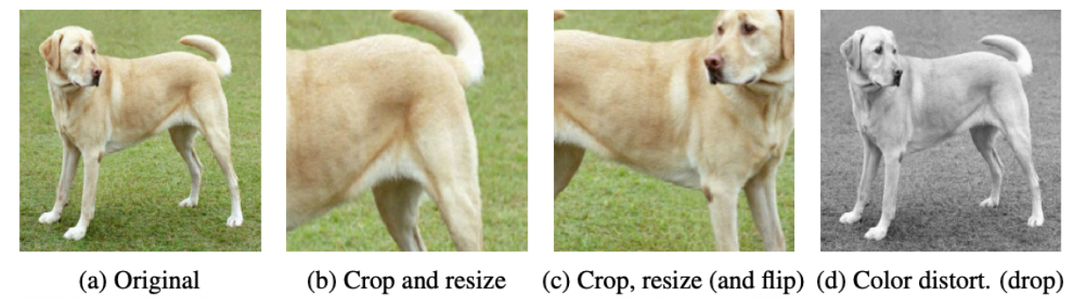
- Similar to related “earlier” (by a few months) methods:

Oord et al Representation Learning with Contrastive Predictive Coding

He et al Momentum Contrast for Unsupervised Visual Representation Learning

# SimCLR

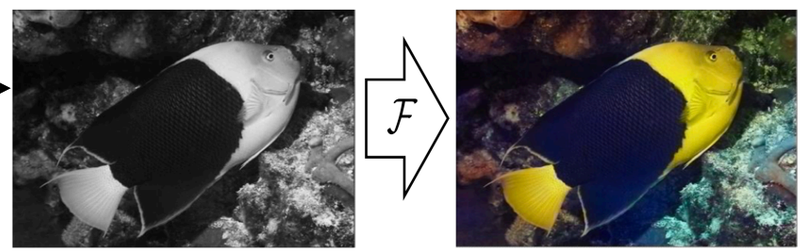
- Many augmentations are needed for strong results
- If auxiliary task has simple solution network will find it without having to build features to understand images/objects
- Variety of augmentations can be made analogous to having many auxiliary tasks



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

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

**Similar to rotation task**




**Similar to Color Prediction Auxiliary Task**

# NT-Xent/InfoNCE

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

| Name | Negative loss function | Gradient w.r.t. $\boldsymbol{u}$ |
|---|---|---|
| NT-Xent | $\boldsymbol{u}^T\boldsymbol{v}^+/\tau - \log \sum_{\boldsymbol{v} \in \{\boldsymbol{v}^+, \boldsymbol{v}^-\}} \exp(\boldsymbol{u}^T\boldsymbol{v}/\tau)$ | $(1 - \frac{\exp(\boldsymbol{u}^T\boldsymbol{v}^+/\tau)}{Z(\boldsymbol{u})})/\tau\boldsymbol{v}^+ - \sum_{\boldsymbol{v}^-} \frac{\exp(\boldsymbol{u}^T\boldsymbol{v}^-/\tau)}{Z(\boldsymbol{u})})/\tau\boldsymbol{v}^-$ |
| 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 $\boldsymbol{0}$ |

**SimCLR authors argue NT-Xent can automatically weight up hard negatives**

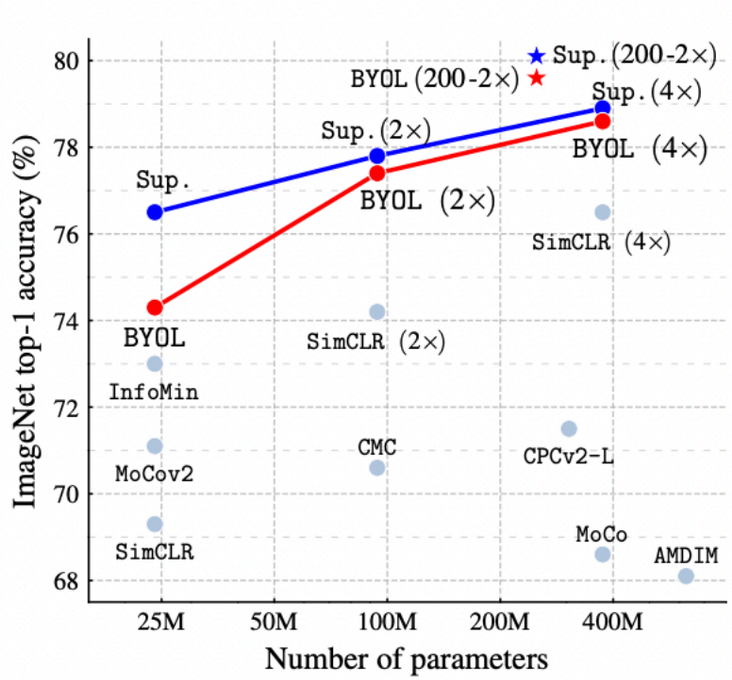
# Bring Your Own Latents BYOL

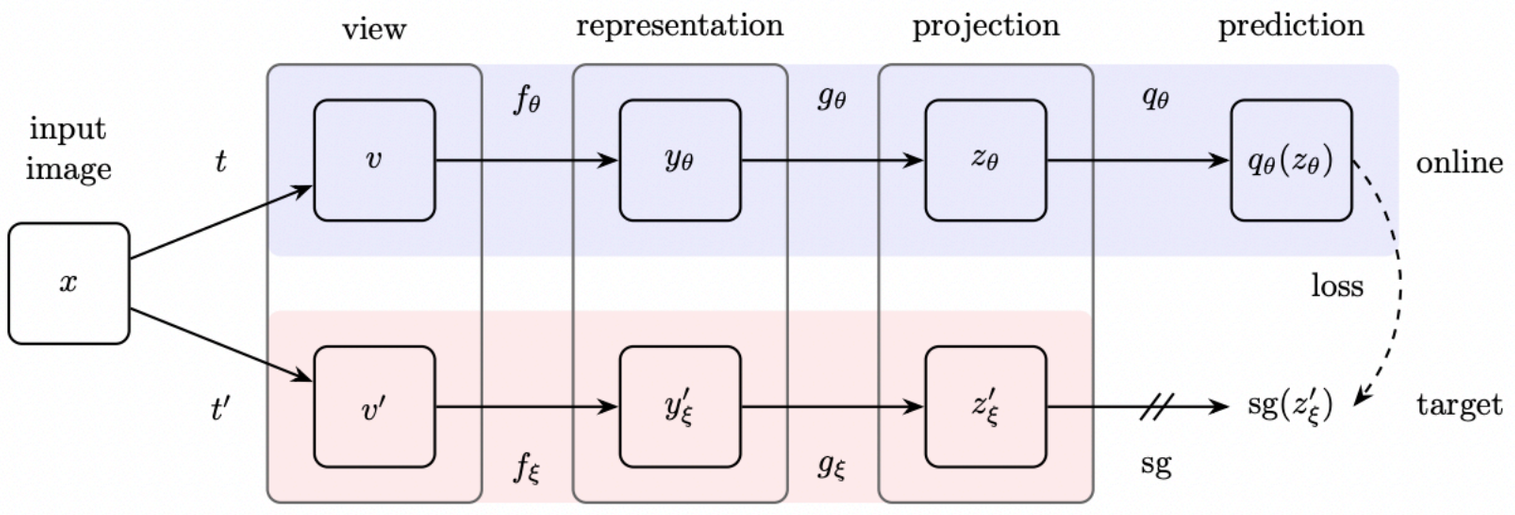
- Recent works shows how to do everything without negatives!

$$u^T v^+ / \tau - \log \sum_{v \in \{v^+, v^-\}} \exp(u^T v / \tau)$$

**u and v come from slightly different networks**

- Underlying principles still not fully understood




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

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


**Jean-Bastien Grill**[*,1] **Florian Strub**[*,1] **Florent Altché**[*,1] **Corentin Tallec**[*,1] **Pierre H. Richemond**[*,1,2]
**Elena Buchatskaya**[1] **Carl Doersch**[1] **Bernardo Avila Pires**[1] **Zhaohan Daniel Guo**[1]
**Mohammad Gheshlaghi Azar**[1] **Bilal Piot**[1] **Koray Kavukcuoglu**[1] **Rémi Munos**[1] **Michal Valko**[1]

# Semi-Supervised Learning vs Unsupervised Pre-training

- Semi-supervised learning
  - Usually means we jointly learn over labeled and unlabeled data
  - E.g. we can train on unlabeled data using self-supervised loss and add another “head”

$$\min_{\theta} \mathcal{L}_l(D_l, \theta) + w\mathcal{L}_u(D_u, \theta),$$

  - More recently though SimCLR does this in 2 steps
- Unsupervised Pre-training / Transfer learning
  - Learn on unlabeled data then train on new task (fine-tune or use features directly)
  - More practical in settings with very large unlabeled data or lack of access to the original unlabeled data

# Conclusions/Remarks

- Self-supervised learning is an exciting emerging area in deep learning

- Solutions are somewhat domain and problem specific however some general principles can be reused (e.g. word2vec objective is similar to  relative position in images)

- Sometimes hard to understand why one method works over another in this area