# Computer Vision

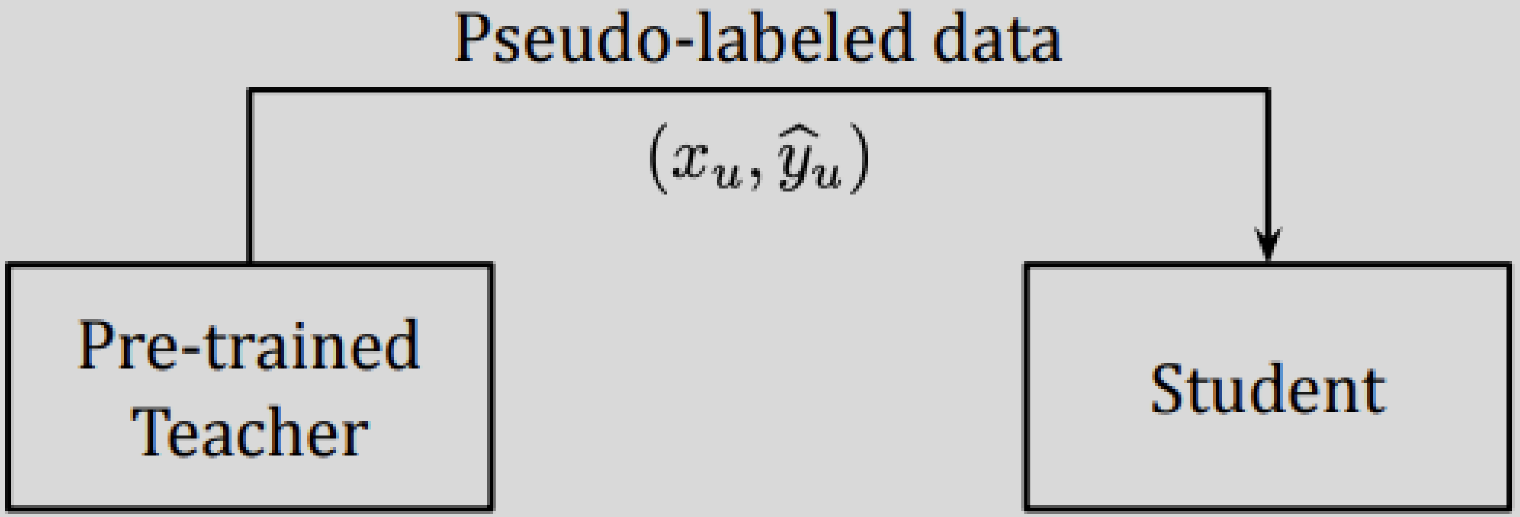
## Lecture 08: Data-efficient Training

# Pseudo labels

- Networks are trained in a supervised fashion jointly with labeled and unlabeled data.

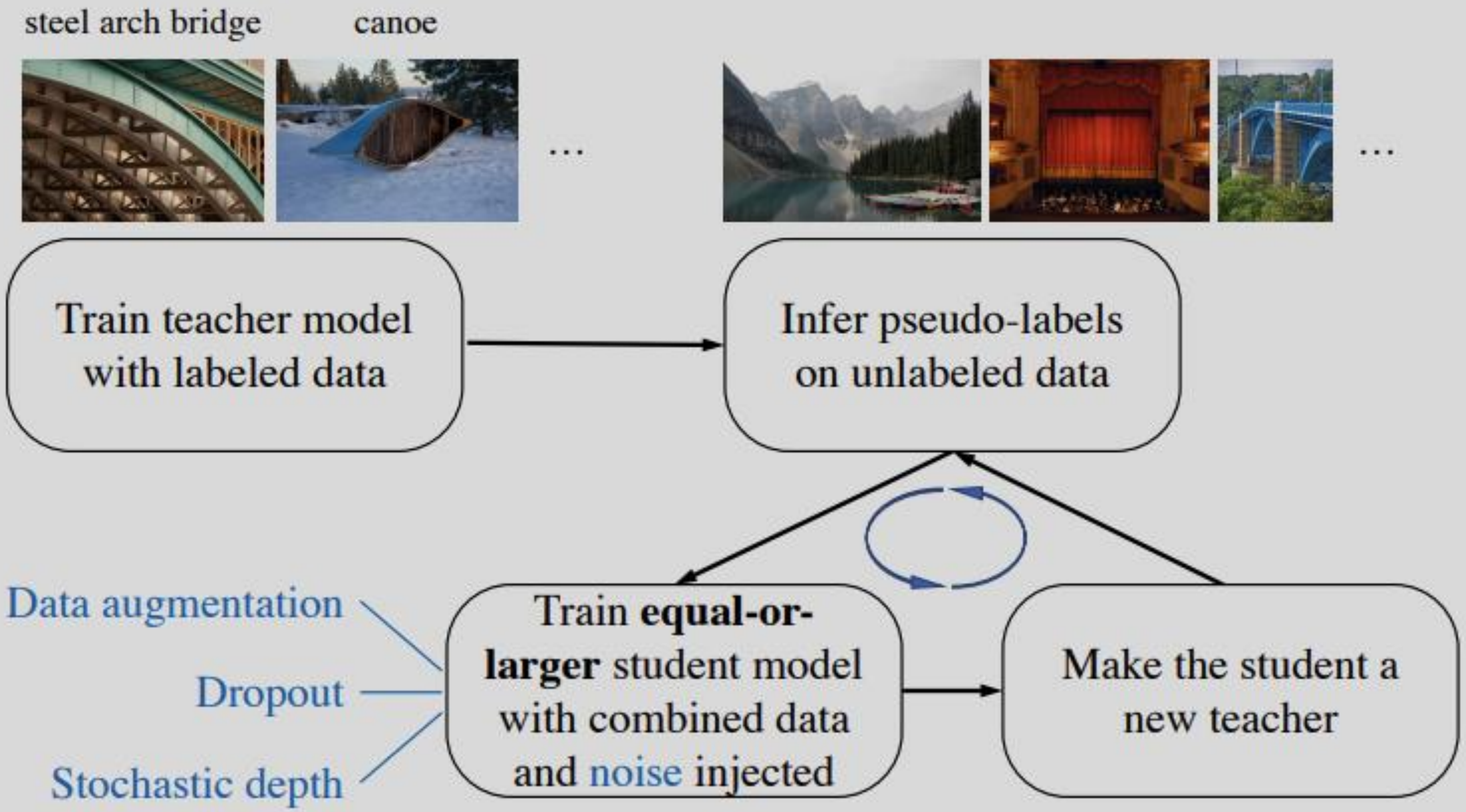
- Pseudo-Labels are target classes for unlabeled data predicted from another network as if they were true labels.

# Pseudo labels




$$\theta_S^{\mathrm{PL}} = \underset{\theta_S}{\operatorname{argmin}} \; \underbrace{\mathbb{E}_{x_u}\Big[\mathrm{CE}\big(T(x_u;\theta_T), S(x_u;\theta_S)\big)\Big]}_{:=\mathcal{L}_u\left(\theta_T,\theta_S\right)}$$

# Self-training




Self-training with noisy student improves imagenet classification, CVPR'20.

# Self-training

**Require:** Labeled images $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ and unlabeled images $\{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_m\}$.

1: Learn teacher model $\theta_*^t$ which minimizes the cross entropy loss on labeled images

$$\frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f^{noised}(x_i, \theta^t))$$

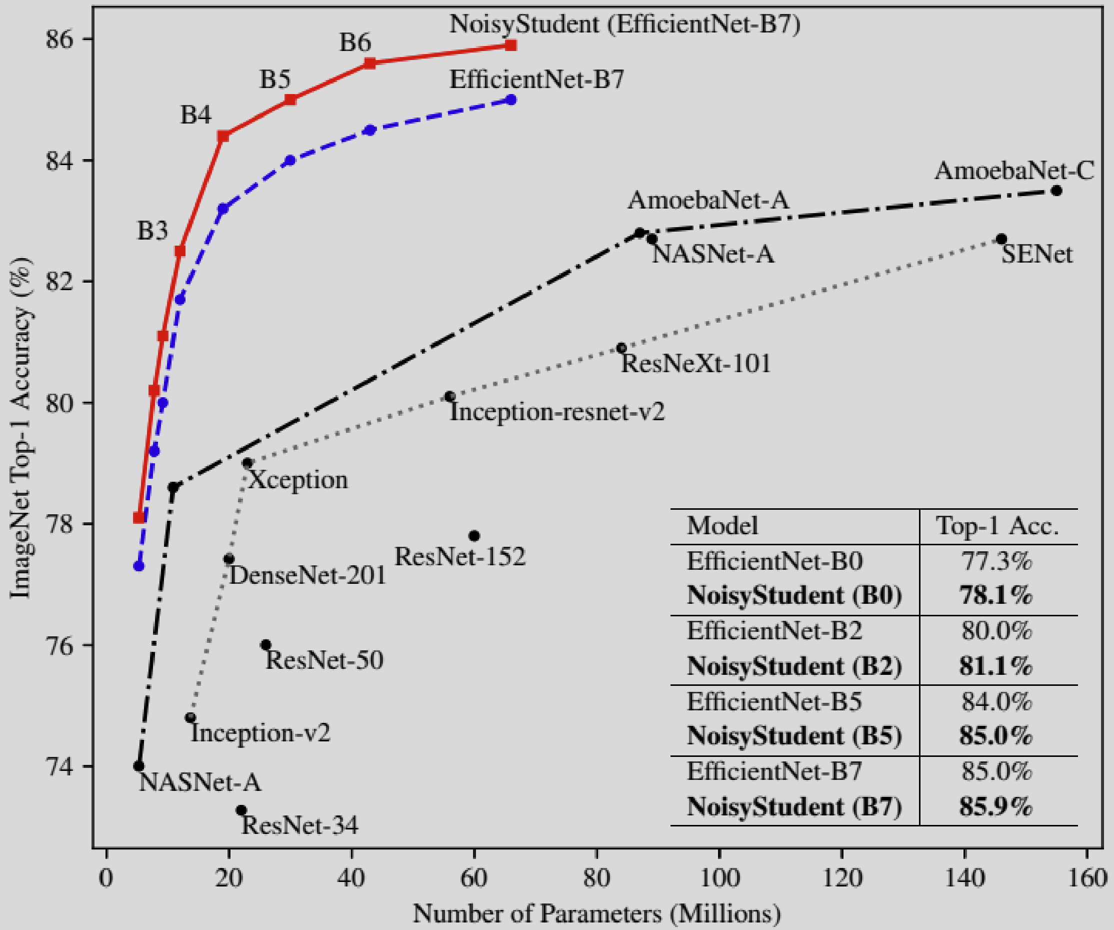
2: Use an unnoised teacher model to generate soft or hard pseudo labels for unlabeled images

$$\tilde{y}_i = f(\tilde{x}_i, \theta_*^t), \forall i = 1, \cdots, m$$

3: Learn an **equal-or-larger** student model $\theta_*^s$ which minimizes the cross entropy loss on labeled images and unlabeled images with **noise** added to the student model

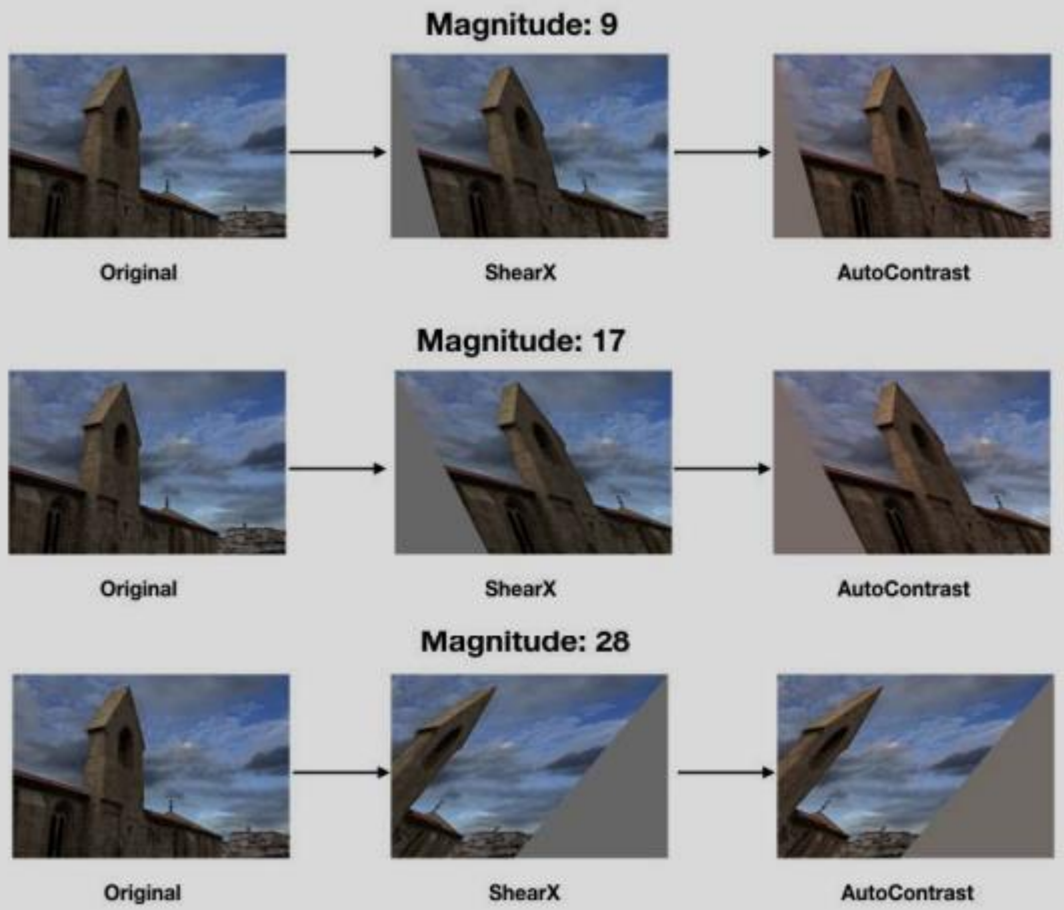
$$\frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f^{noised}(x_i, \theta^s)) + \frac{1}{m}\sum_{i=1}^{m} \ell(\tilde{y}_i, f^{noised}(\tilde{x}_i, \theta^s))$$

4: Iterative training: Use the student as a teacher and go back to step 2.




| Model | Top-1 Acc. |
|---|---|
| EfficientNet-B0 | 77.3% |
| **NoisyStudent (B0)** | **78.1%** |
| EfficientNet-B2 | 80.0% |
| **NoisyStudent (B2)** | **81.1%** |
| EfficientNet-B5 | 84.0% |
| **NoisyStudent (B5)** | **85.0%** |
| EfficientNet-B7 | 85.0% |
| **NoisyStudent (B7)** | **85.9%** |

# Self-training (Noise)



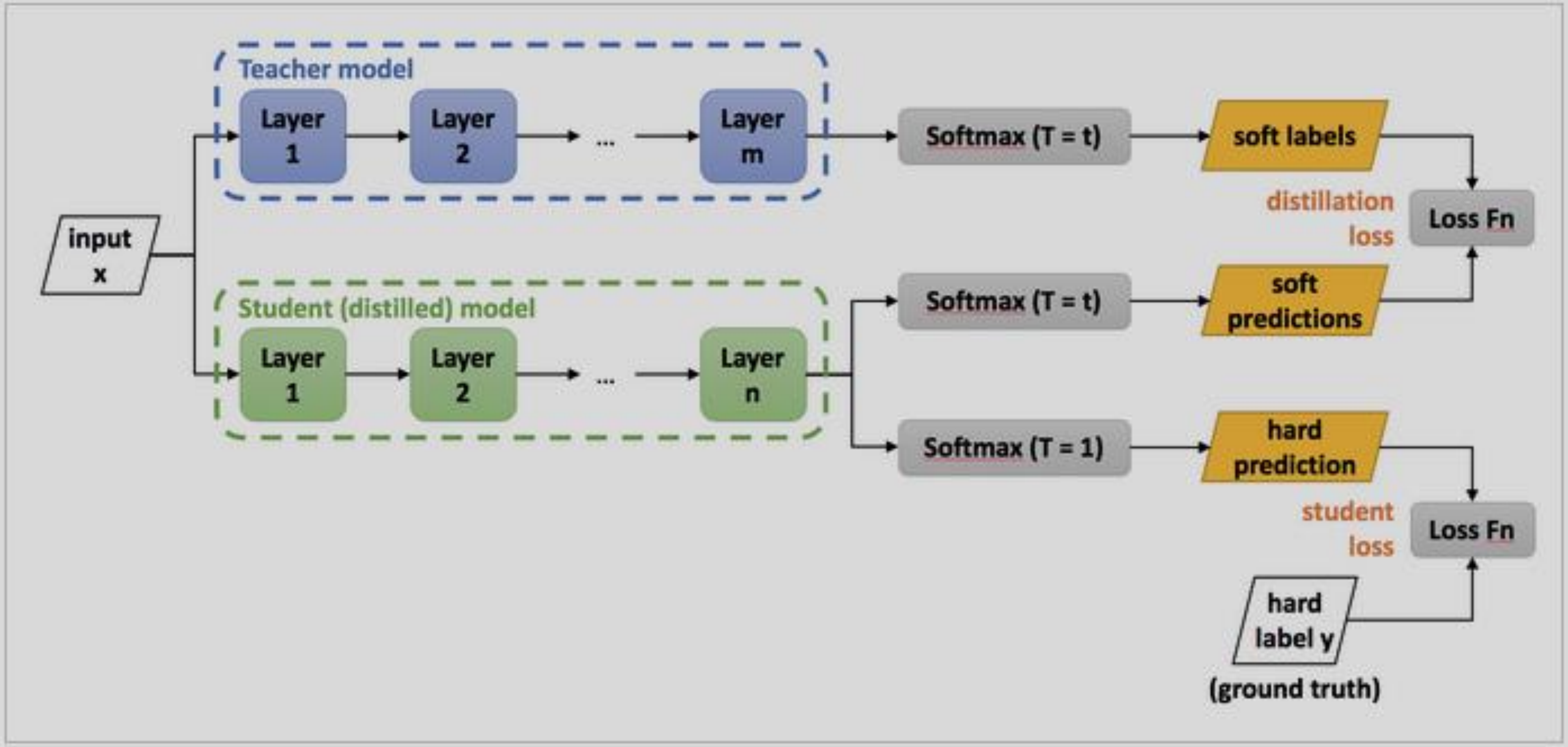

```
transforms = [
'Identity', 'AutoContrast', 'Equalize', 'Rotate',
    'Solarize', 'Color',
'Posterize', 'Contrast', 'Brightness', 'Sharpness',
    'ShearX', 'ShearY',
'TranslateX', 'TranslateY']

def randaugment(N, M):
"""Generate a set of distortions.

  Args:
    N: Number of augmentation transformations to apply
        sequentially.
    M: Magnitude for all the transformations.
"""

  sampled_ops = np.random.choice(transforms, N)
  return [(op, M) for op in sampled_ops]
```

# Knowledge distillation

- The teacher network provides a richer supervisory signal than the data supervision.

- KD guides the training of a student network by encouraging it to mimic some aspect of a teacher network.

# Knowledge distillation



# Knowledge distillation

| cow | dog | cat | car |
|---|---|---|---|
| 0 | 1 | 0 | 0 |

| cow | dog | cat | car |
|---|---|---|---|
| $10^{-6}$ | .9 | .1 | $10^{-9}$ |

| cow | dog | cat | car |
|---|---|---|---|
| .05 | .3 | .2 | .005 |

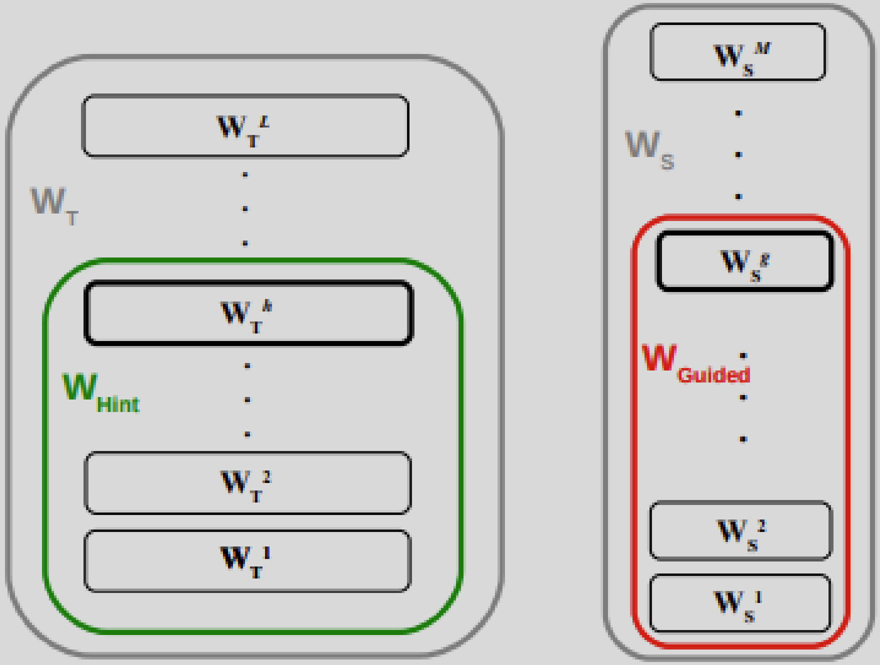
`Dark knowledge'

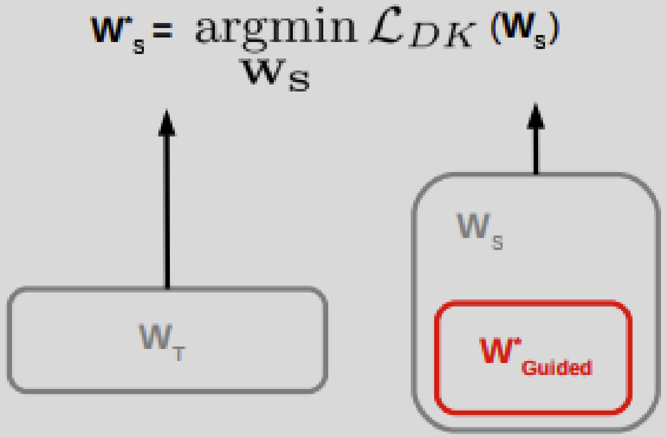
Which classes the teacher found more similar to the predicted class.

$$q_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)}$$

q : probability
T : temperature

# Knowledge distillation




$$\mathrm{P}_{\mathrm{T}}^{\tau} = \mathrm{softmax}\left(\frac{\mathbf{a}_T}{\tau}\right), \quad \mathrm{P}_{\mathrm{S}}^{\tau} = \mathrm{softmax}\left(\frac{\mathbf{a}_S}{\tau}\right) \quad \mathcal{L}_{KD}(\mathbf{W_S}) = \mathcal{H}(\mathbf{y_{true}}, \mathrm{P_S}) + \lambda \mathcal{H}(\mathrm{P}_{\mathrm{T}}^{\tau}, \mathrm{P}_{\mathrm{S}}^{\tau})$$

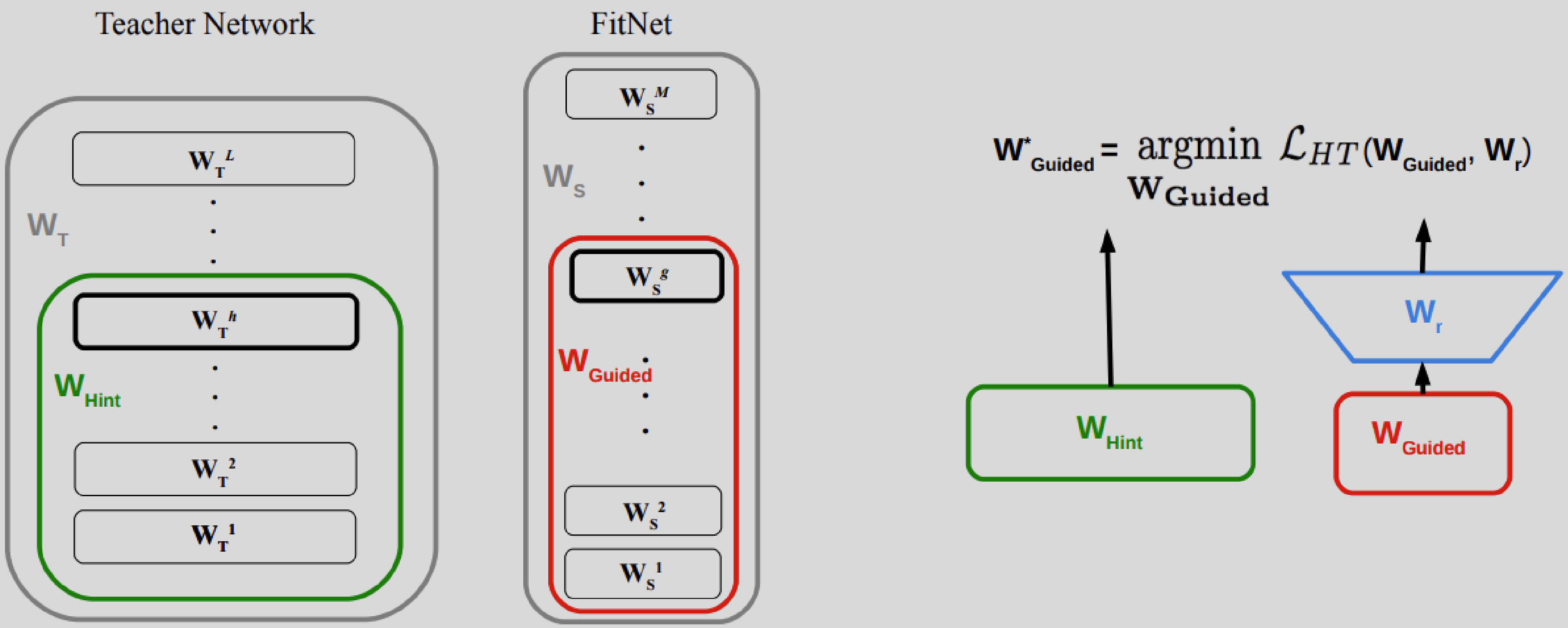
Distilling the Knowledge in a Neural Network, NeurIPS'14

# Hints

- KD fails when the depth of the student network getting deeper.

- *Hint* is defined as the output of a teacher's hidden layer.

# Learning hints




$$\mathcal{L}_{HT}(\mathbf{W_{Guided}}, \mathbf{W_r}) = \frac{1}{2}||u_h(\mathbf{x}; \mathbf{W_{Hint}}) - r(v_g(\mathbf{x}; \mathbf{W_{Guided}}); \mathbf{W_r})||^2$$

FitNets: hints for thin deep nets, ICLR'15.

# Learning hints

**Input:** $\mathbf{W_S}, \mathbf{W_T}, g, h$

**Output:** $\mathbf{W_S^*}$

1: $\mathbf{W_{Hint}} \leftarrow \{\mathbf{W_T}^1, \ldots, \mathbf{W_T}^h\}$

2: $\mathbf{W_{Guided}} \leftarrow \{\mathbf{W_S}^1, \ldots, \mathbf{W_S}^g\}$

3: Intialize $\mathbf{W_r}$ to small random values

4: $\mathbf{W^*_{Guided}} \leftarrow \underset{\mathbf{W_{Guided}}}{\operatorname{argmin}} \; \mathcal{L}_{HT}(\mathbf{W_{Guided}}, \mathbf{W_r})$

5: $\{\mathbf{W_S}^1, \ldots, \mathbf{W_S}^g\} \leftarrow \{\mathbf{W_{Guided}}^{*1}, \ldots, \mathbf{W_{Guided}}^{*g}\}$

6: $\mathbf{W_S^*} \leftarrow \underset{\mathbf{W_S}}{\operatorname{argmin}} \, \mathcal{L}_{KD}(\mathbf{W_S})$

# Intermediate representation

| Algorithm | # params | Accuracy |
|---|---|---|
| *Compression* | | |
| FitNet | ~2.5M | **91.61%** |
| Teacher | ~9M | 90.18% |
| Mimic single | ~54M | 84.6% |
| Mimic single | ~70M | 84.9% |
| Mimic ensemble | ~70M | 85.8% |
| *State-of-the-art methods* | | |
| Maxout | | 90.65% |
| Network in Network | | 91.2% |
| Deeply-Supervised Networks | | **91.78%** |
| Deeply-Supervised Networks (19) | | 88.2% |

Table 1: Accuracy on CIFAR-10

| Algorithm | # params | Accuracy |
|---|---|---|
| *Compression* | | |
| FitNet | ~2.5M | **64.96%** |
| Teacher | ~9M | 63.54% |
| *State-of-the-art methods* | | |
| Maxout | | 61.43% |
| Network in Network | | 64.32% |
| Deeply-Supervised Networks | | **65.43%** |

Table 2: Accuracy on CIFAR-100

# Intermediate representation

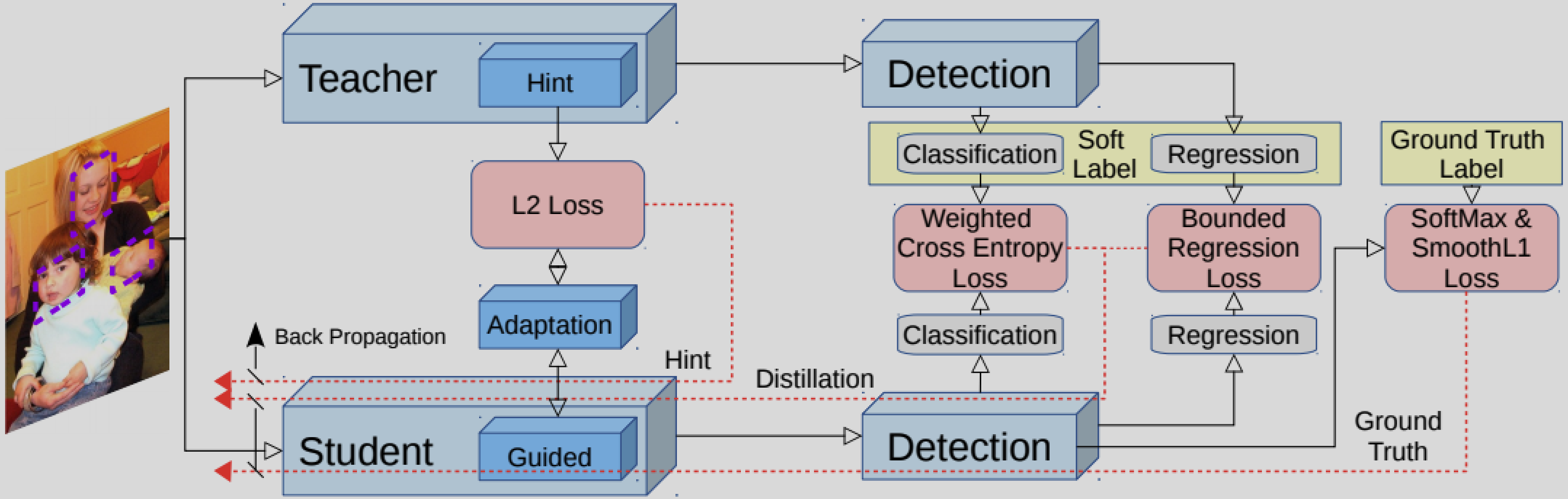
| Algorithm | # params | Misclass |
|---|---|---|
| *Compression* | | |
| FitNet | ~1.5M | 2.42% |
| Teacher | ~4.9M | **2.38%** |
| *State-of-the-art methods* | | |
| Maxout | | 2.47% |
| Network in Network | | 2.35% |
| Deeply-Supervised Networks | | **1.92%** |

Table 3: SVHN error

| Algorithm | # params | Misclass |
|---|---|---|
| *Compression* | | |
| Teacher | ~361K | 0.55% |
| Standard backprop | ~30K | 1.9% |
| KD | ~30K | 0.65% |
| FitNet | ~30K | **0.51%** |
| *State-of-the-art methods* | | |
| Maxout | | 0.45% |
| Network in Network | | 0.47% |
| Deeply-Supervised Networks | | **0.39%** |

Table 4: MNIST error

# KD for object detection




Learning Efficient Object Detection Models with Knowledge Distillation, NIPS’17.

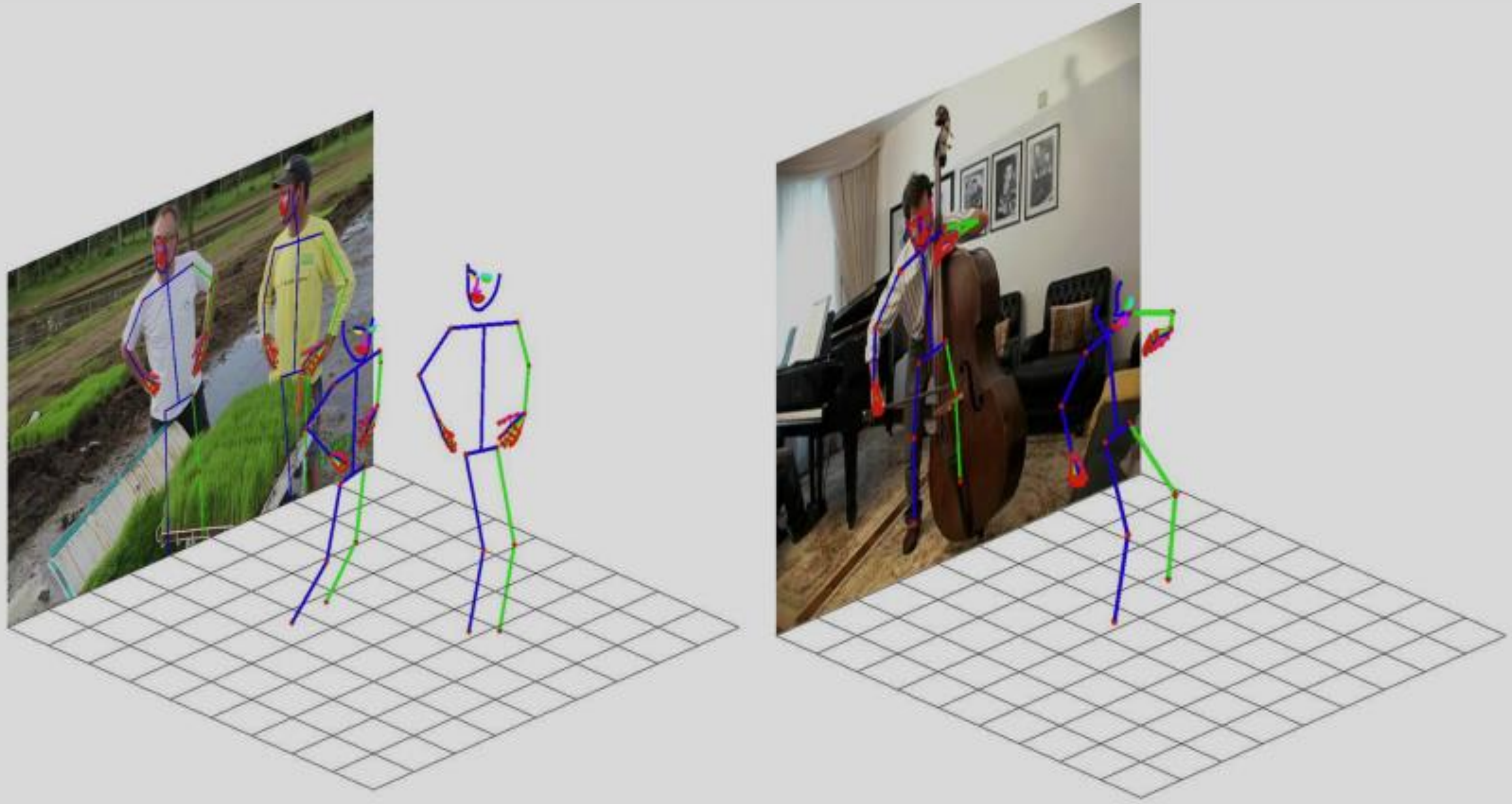
# KD for object detection

| | | Baseline | Distillation | Hint | Distillation + Hint |
|---|---|---|---|---|---|
| PASCAL | Trainval | 79.6 | 78.3 | 80.9 | 83.5 |
| | Test | 54.7 | 58.4 | 58 | 59.4 |
| COCO | Train | 45.3 | 45.4 | 47.1 | 49.6 |
| | Val | 25.4 | 26.1 | 27.8 | 28.3 |

learning on different datasets with Tucker and VGG16 pair.

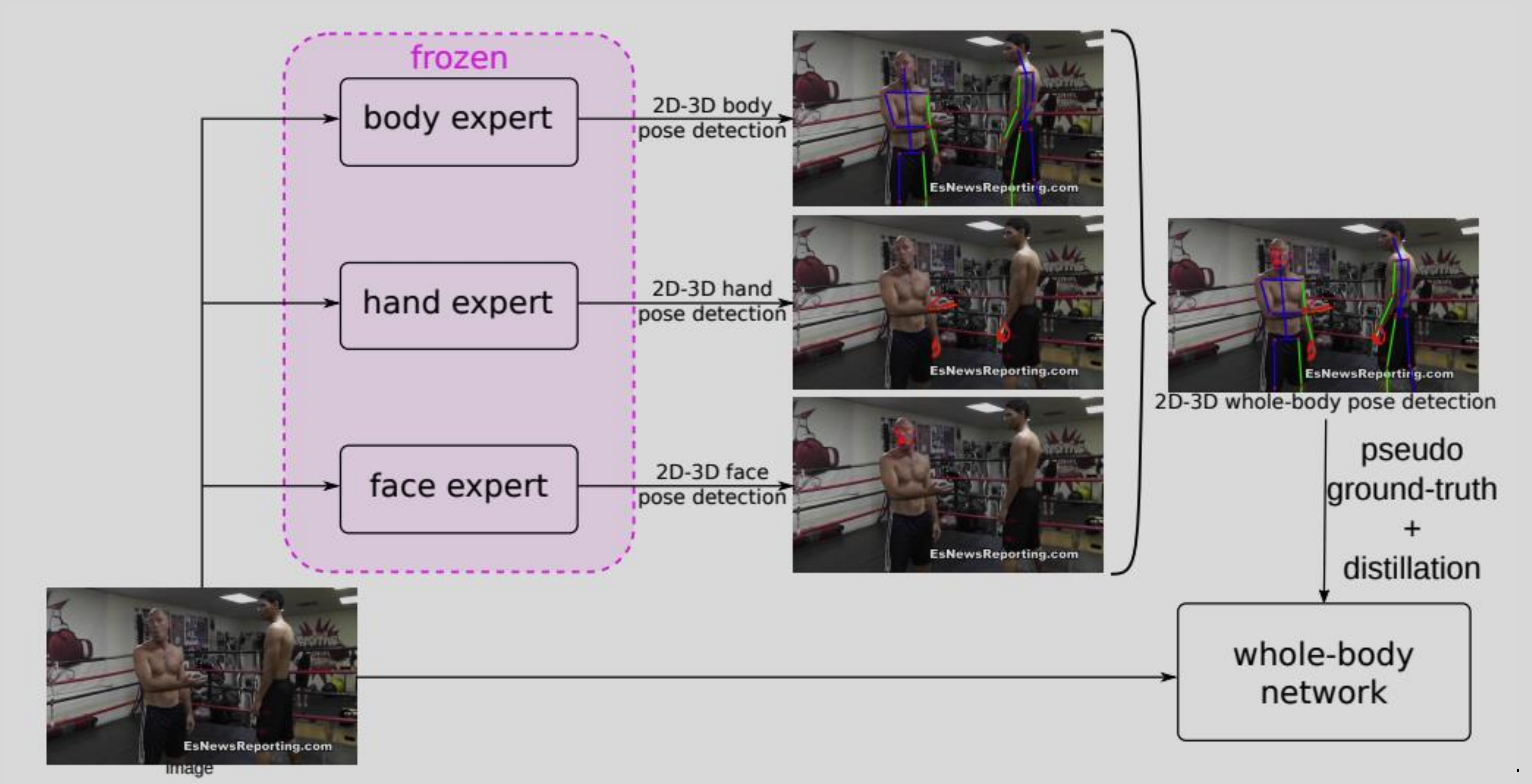
Learning Efficient Object Detection Models with Knowledge Distillation, NIPS'17.

# Distillation of part experts

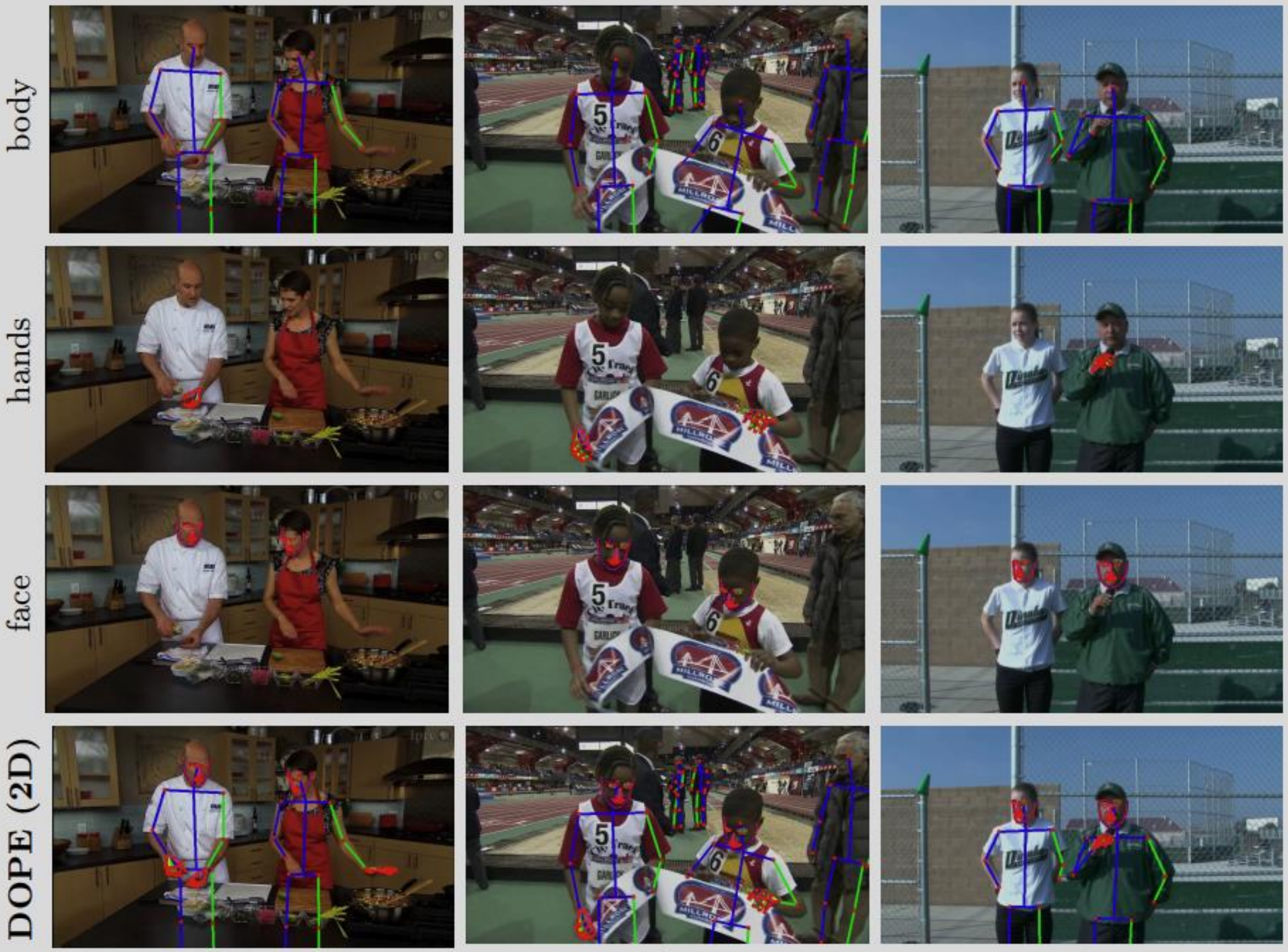


DOPE: Distillation Of Part Experts for whole-body 3D pose estimation in the wild, ECCV'20.

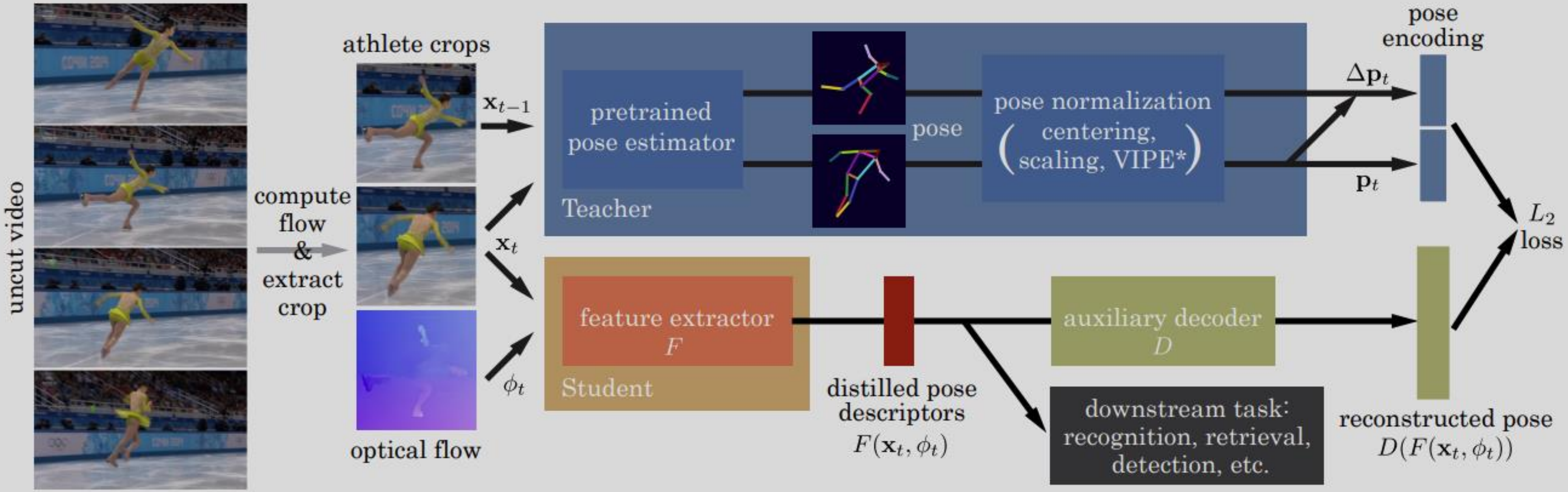
# Distillation of part experts



# Distillation of part experts



# Video pose distillation




Video Pose Distillation for Few-Shot, Fine-Grained Sports Action Recognition, ICCV'21.

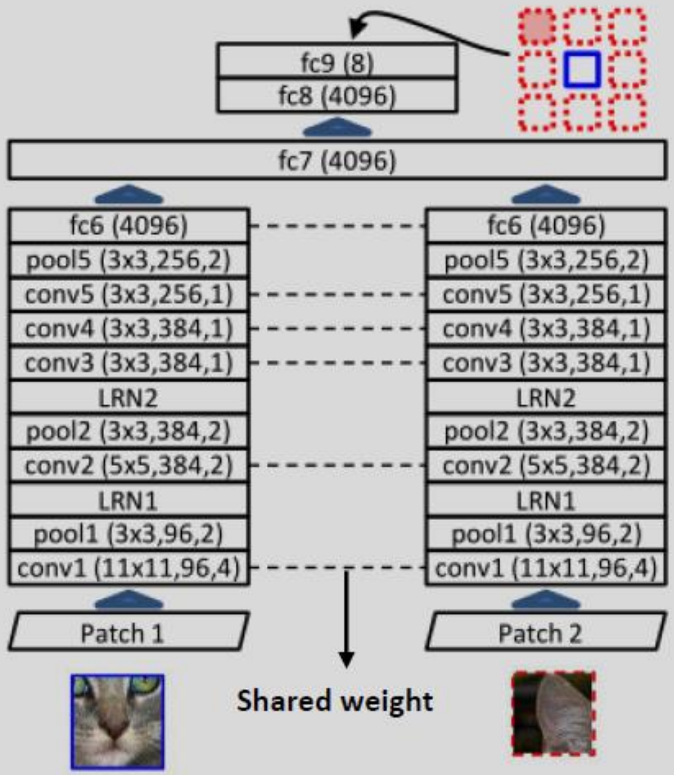
# Video pose distillation

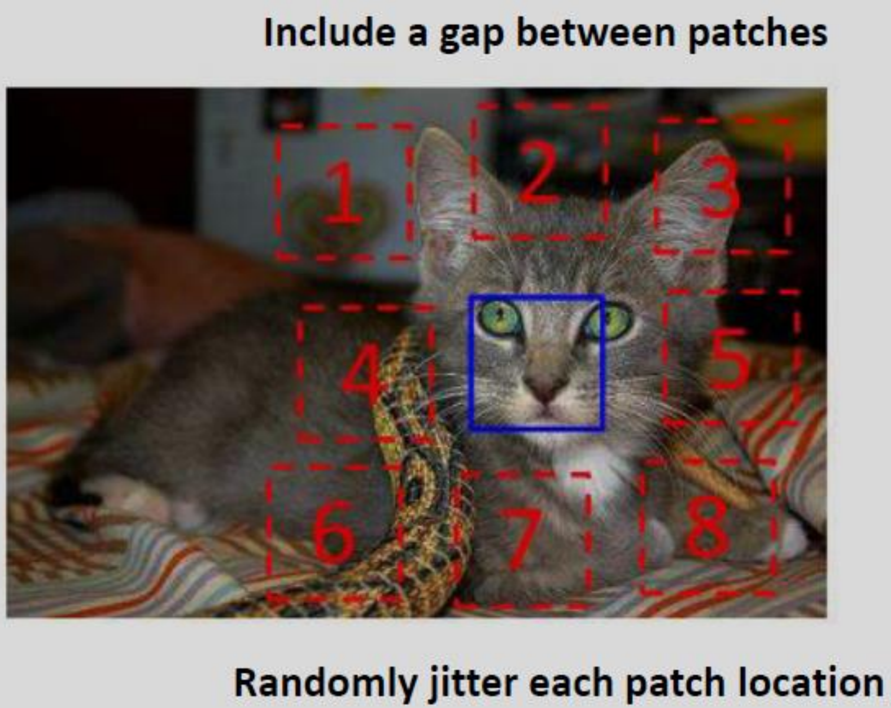
$$\Delta \mathbf{p}_t := \mathbf{p}_t - \mathbf{p}_{t-1}$$

$$\underset{F,D}{\text{minimize}} \sum_{t=1}^{N} \left\| D\left(F\left(\mathbf{x}_t, \phi_t\right)\right) - \begin{bmatrix} \mathbf{p}_t \\ \Delta \mathbf{p}_t \end{bmatrix} \right\|_2^2$$

Video Pose Distillation for Few-Shot, Fine-Grained Sports Action Recognition, ICCV'21.

# Self-supervised learning




Unsupervised Visual Representation Learning by Context Prediction. *ICCV 2015*

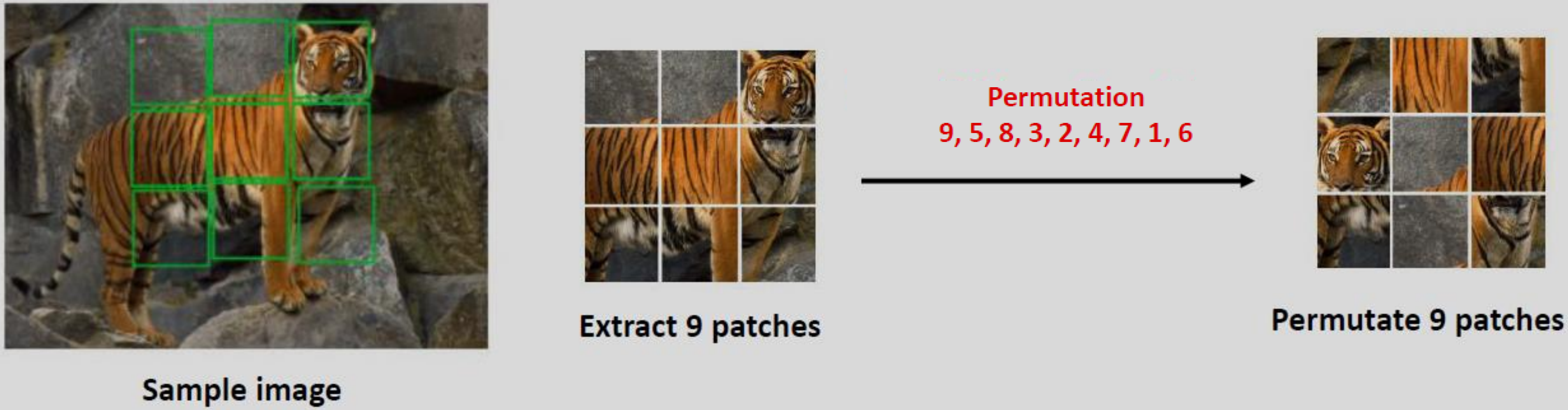
# Self-supervised learning

Context Prediction: Predict relative positions of patches

- You have to understand the object to solve this problem!
- Be aware of trivial solution! CNN is especially good at it


Unsupervised Visual Representation Learning by Context Prediction. *ICCV 2015*

# Self-supervised learning

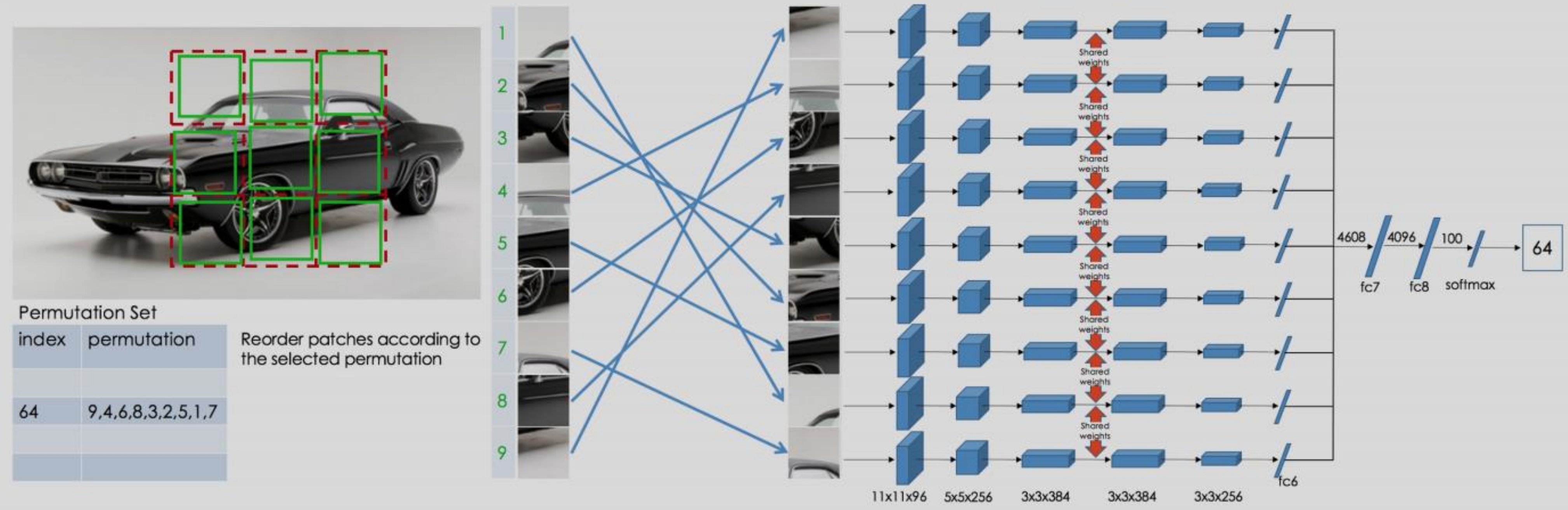



Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV 2016*.

# Self-supervised learning

## Solving the Jigsaw

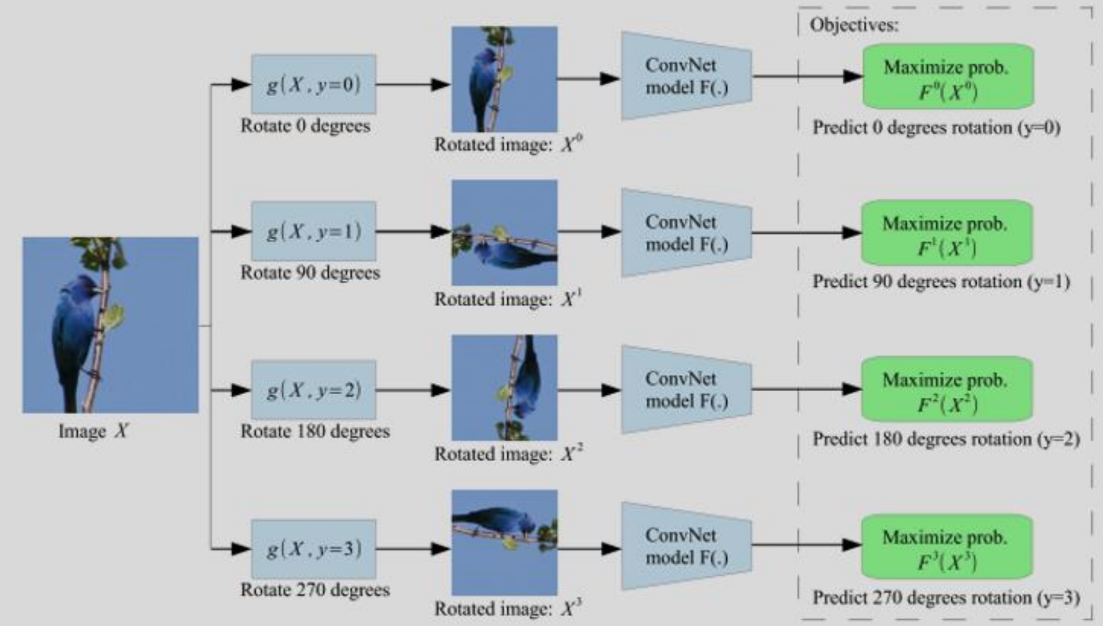
- Use stronger supervision, solve the real jigsaw problem
- Harder problem, better ability for networks



# Self-supervised learning

## Predicting the rotations

- Predict the 4 types of rotation angles.



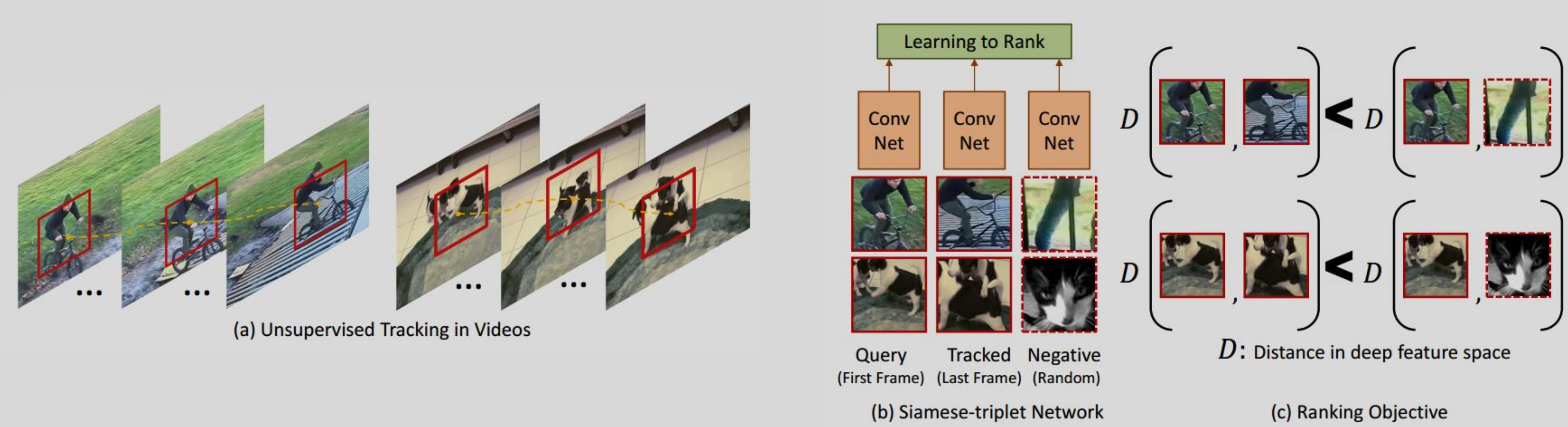

| Method | Conv1 | Conv2 | Conv3 | Conv4 | Conv5 |
|---|---|---|---|---|---|
| ImageNet labels | 19.3 | 36.3 | 44.2 | 48.3 | 50.5 |
| Random | 11.6 | 17.1 | 16.9 | 16.3 | 14.1 |
| Random rescaled Krähenbühl et al. (2015) | 17.5 | 23.0 | 24.5 | 23.2 | 20.6 |
| Context (Doersch et al., 2015) | 16.2 | 23.3 | 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** |

**ImageNet classification top-1 accuracy**

Unsupervised representation learning by predicting image rotations. In *ICLR 2018*.

# Self-supervision for video

Find corresponding pairs using visual tracking

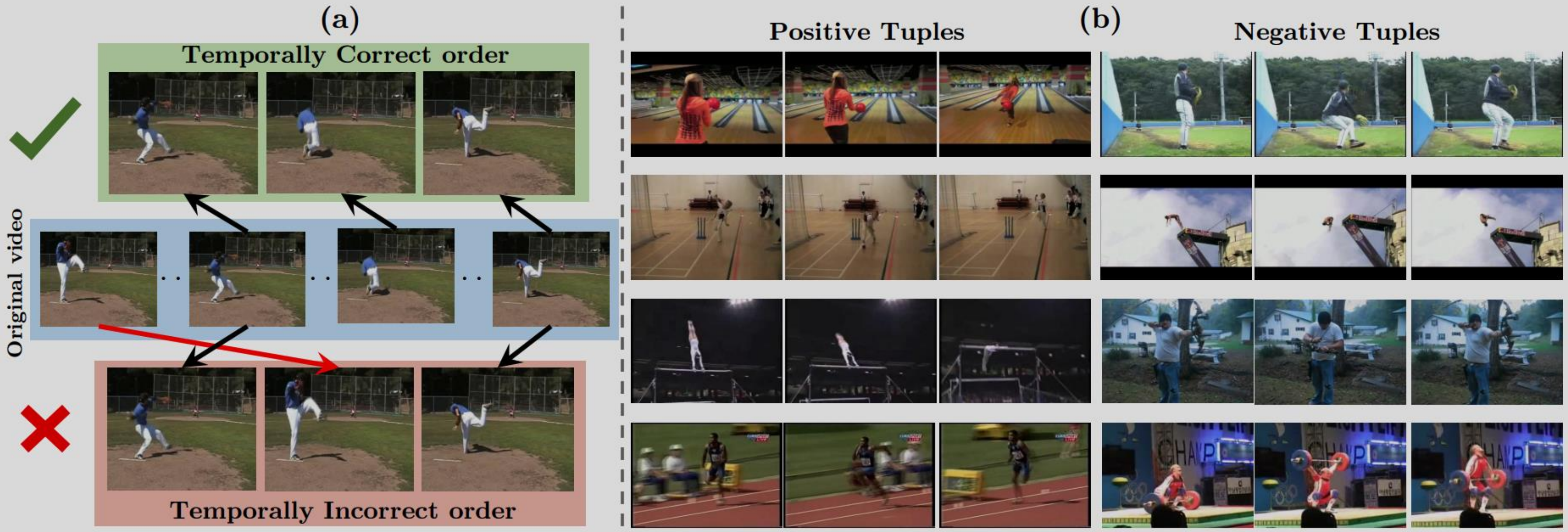



Wang, X., & Gupta, A. (2015). Unsupervised learning of visual representations using videos. In *ICCV2015*

# Self-supervision for video

Is the temporal order of a video correct?

- Encode the cause and effect of action

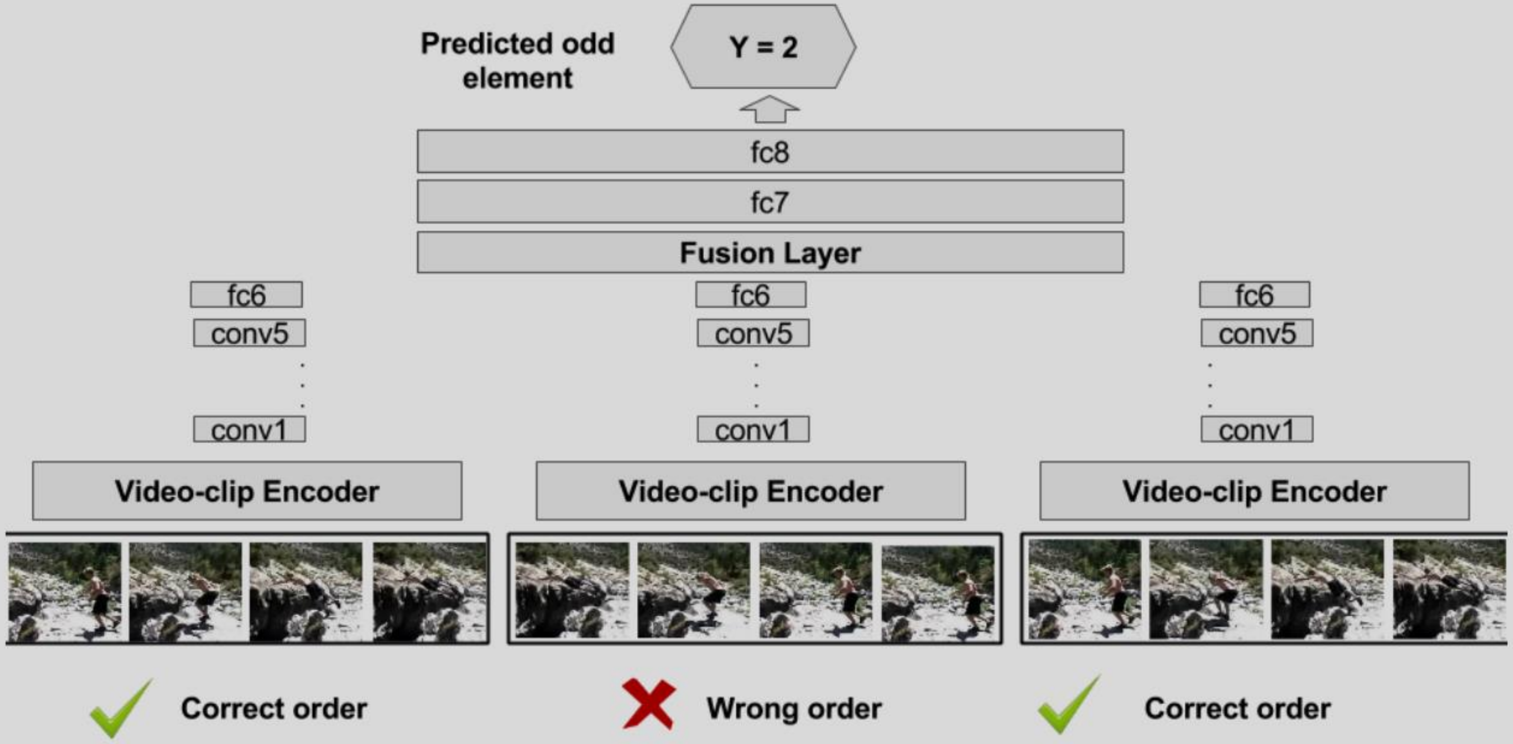



Misra, I., Zitnick, C. L., & Hebert, M. Shuffle and learn: unsupervised learning using temporal order verification. In *ECCV 2016*.

# Self-supervision for video

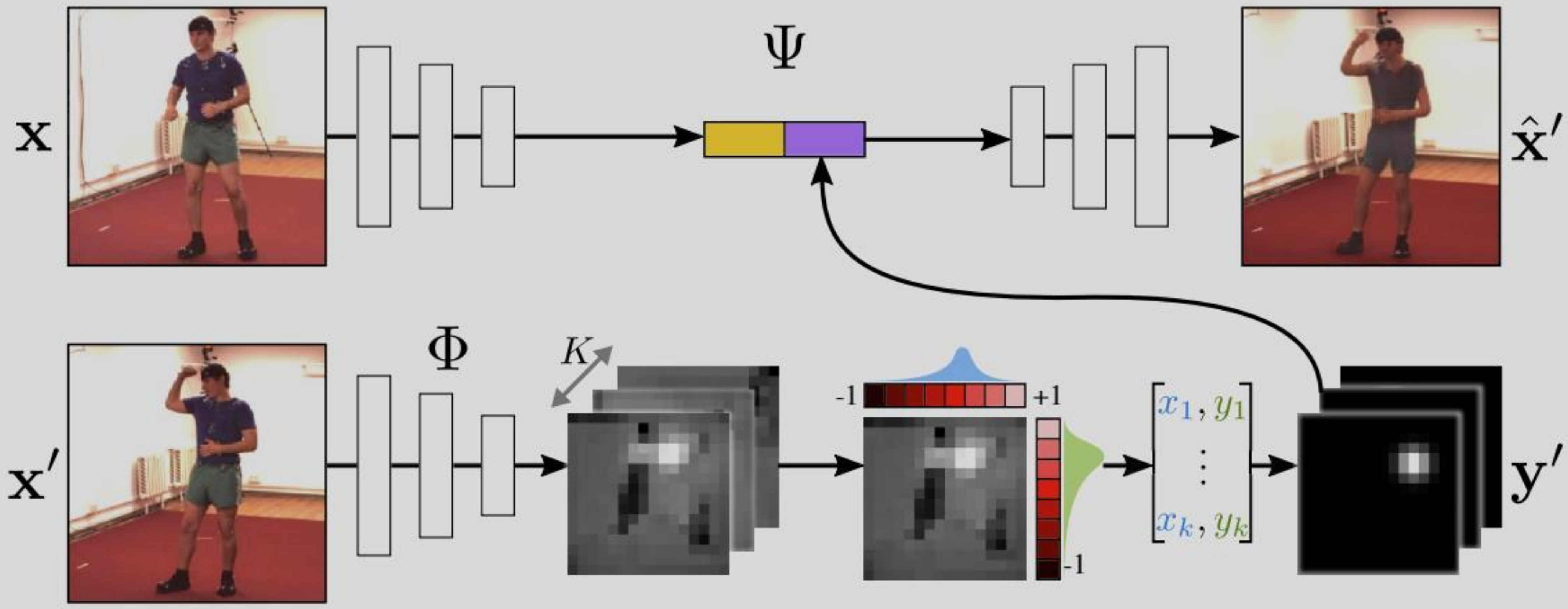
Is the temporal order of a video correct?

- Find the odd sequence



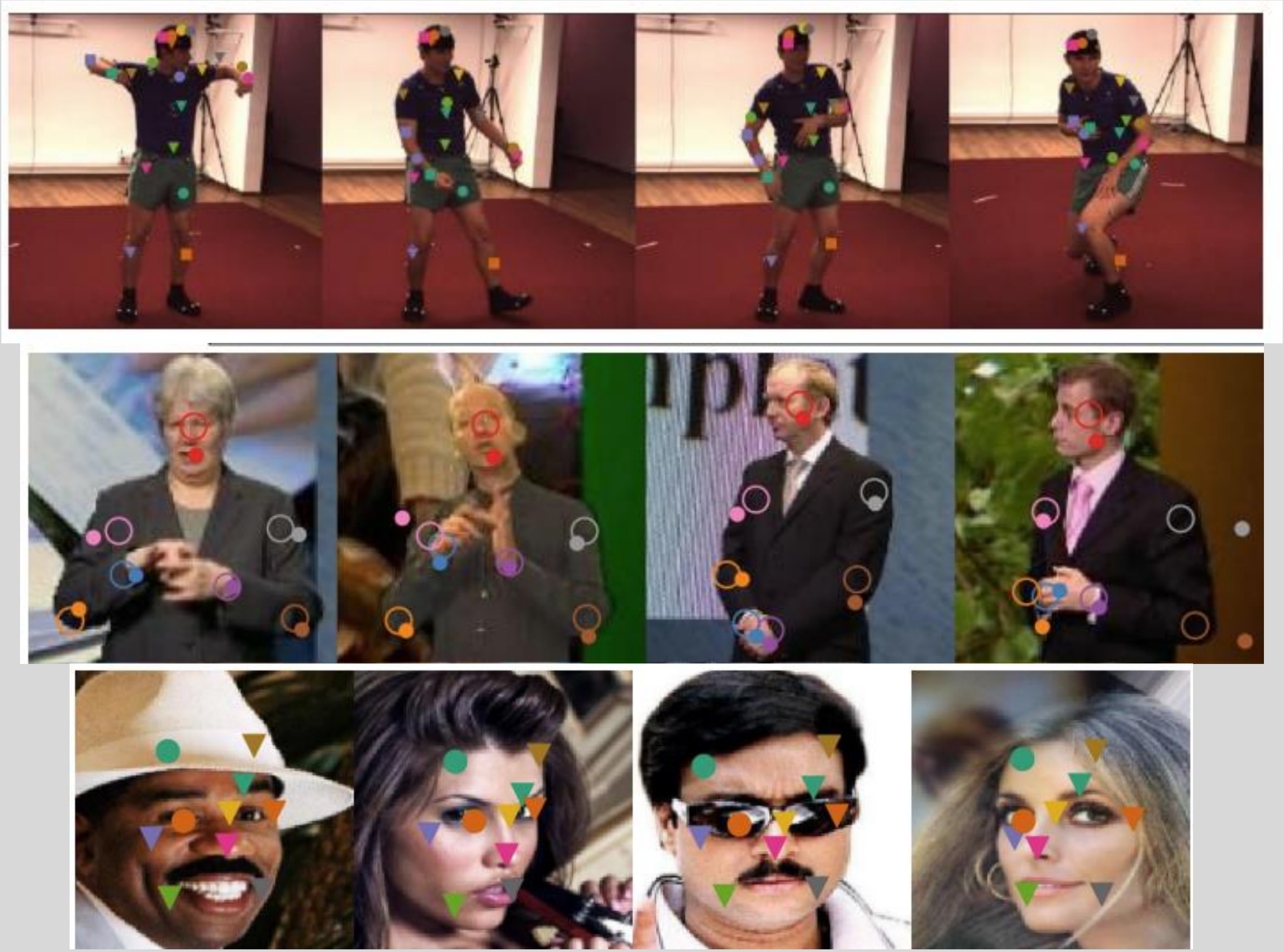

Fernando, B., Bilen, H., Gavves, E., & Gould, S. Self-Supervised Video Representation Learning With Odd-One-Out Networks. *In CVPR2017*.

# Self-supervision for pose



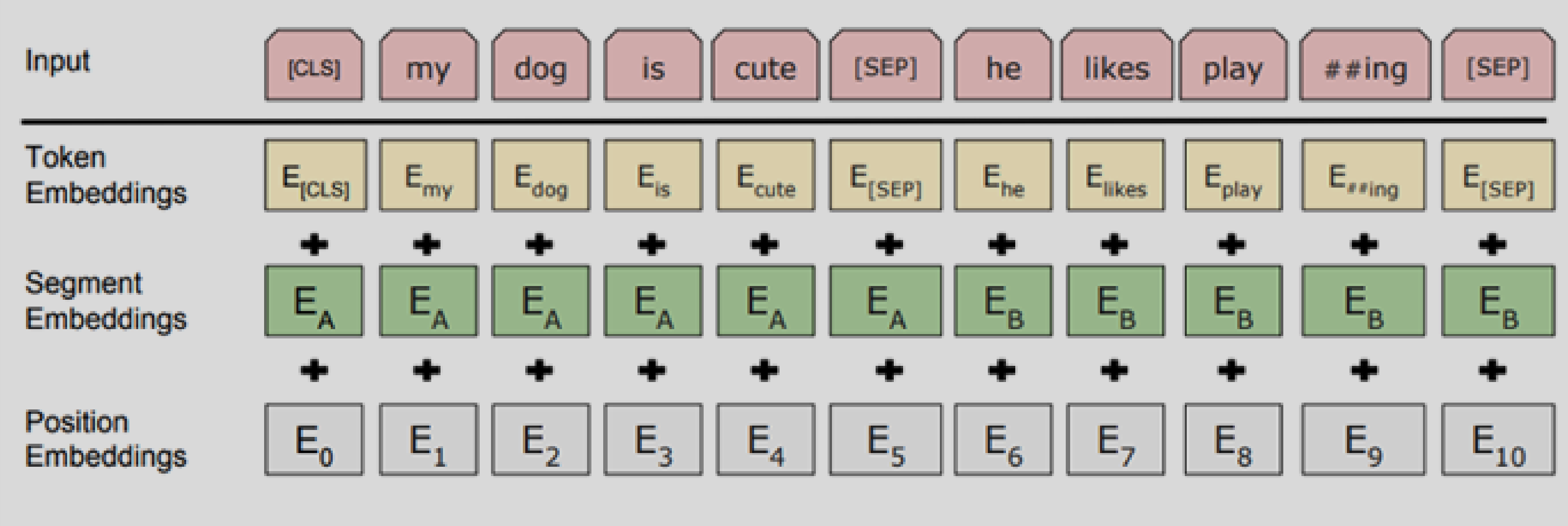

Unsupervised Learning of Object Landmarks through Conditional Image Generation, NeurIPS'18.

# Self-supervision for pose

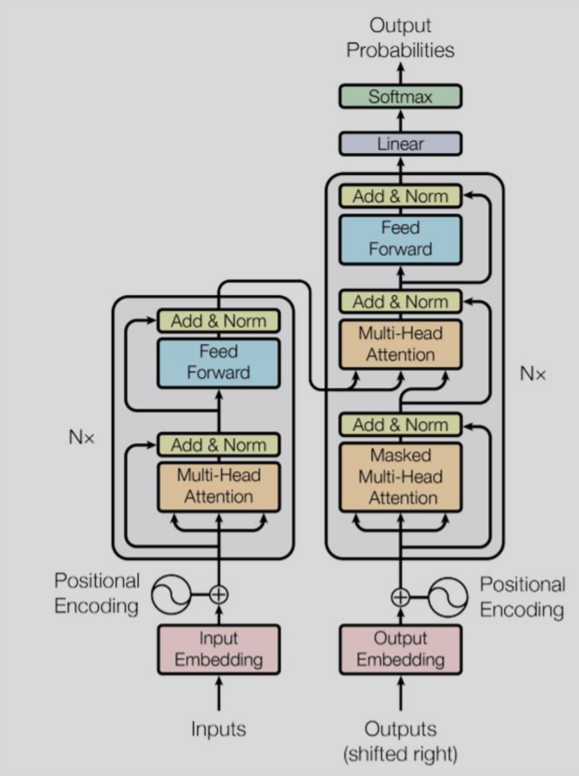


# Self-supervised learning

- Self-supervision: Learning without tagged data.
- The method could be applied to any inputs.
  - Speech, image, video, text and etc.



# Self-supervised learning




Input = [CLS] the man went to [MASK] store [SEP]

he bought a gallon [MASK] milk [SEP]

Label = IsNext

Input = [CLS] the man [MASK] to the store [SEP]

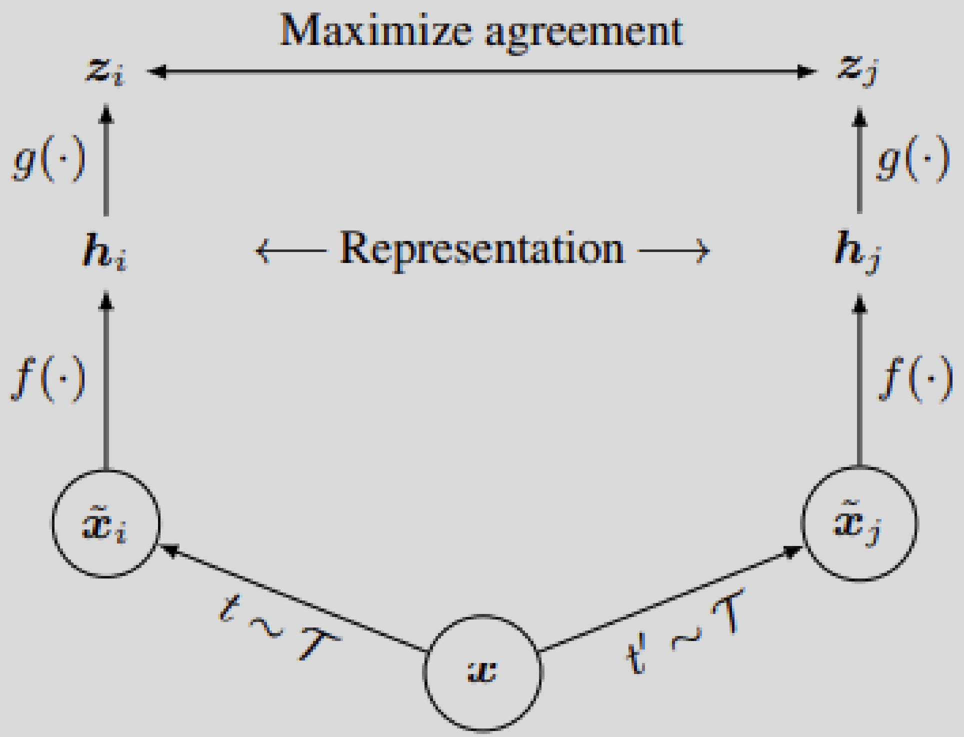
penguin [MASK] are flight ##less birds [SEP]

Label = NotNext

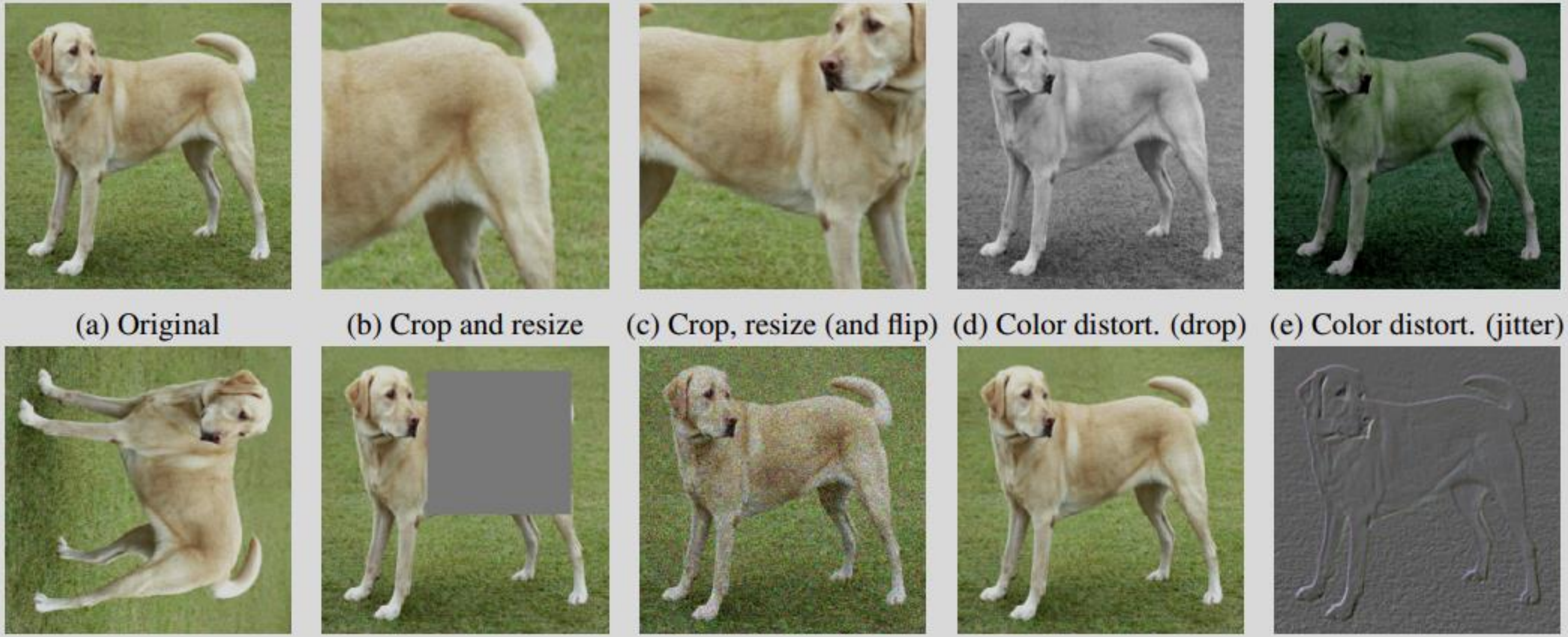
Transformer architecture is trained by 1) Masked language model, 2) Next sentence prediction

# Contrastive learning

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



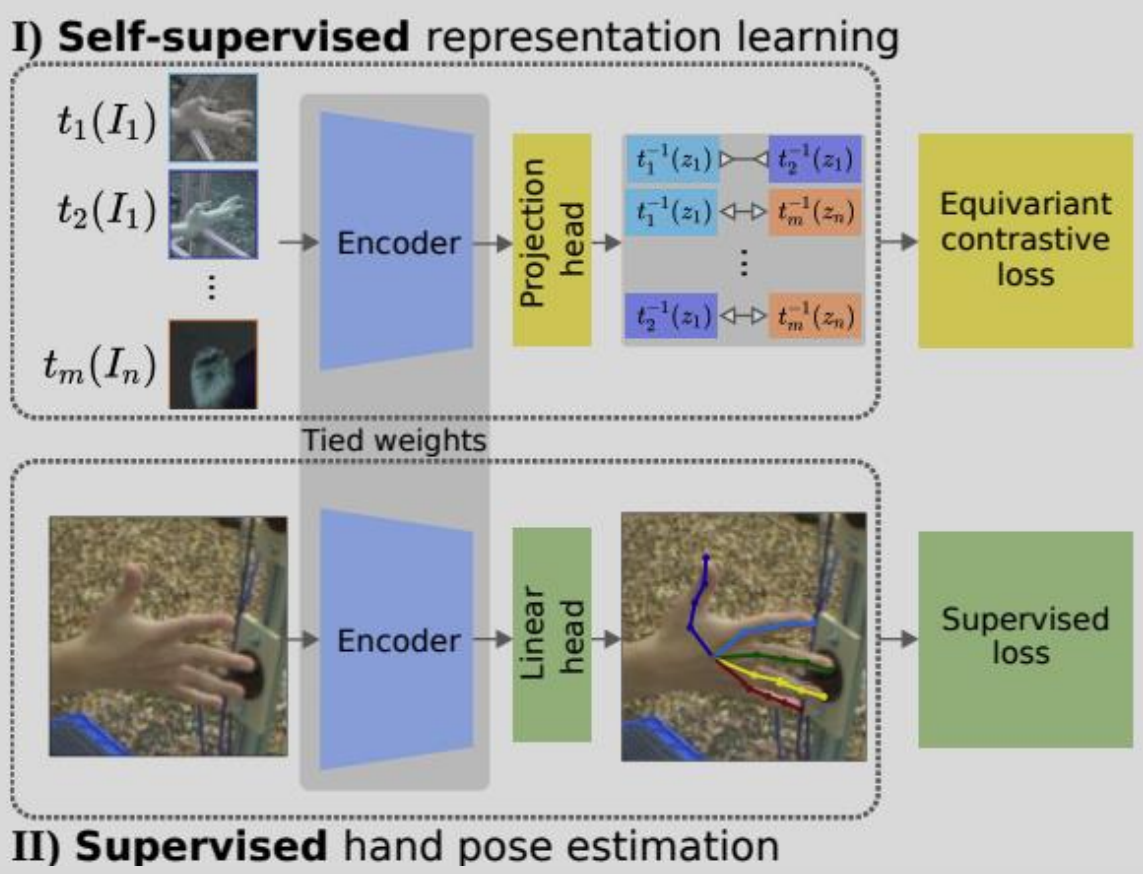
# Contrastive learning

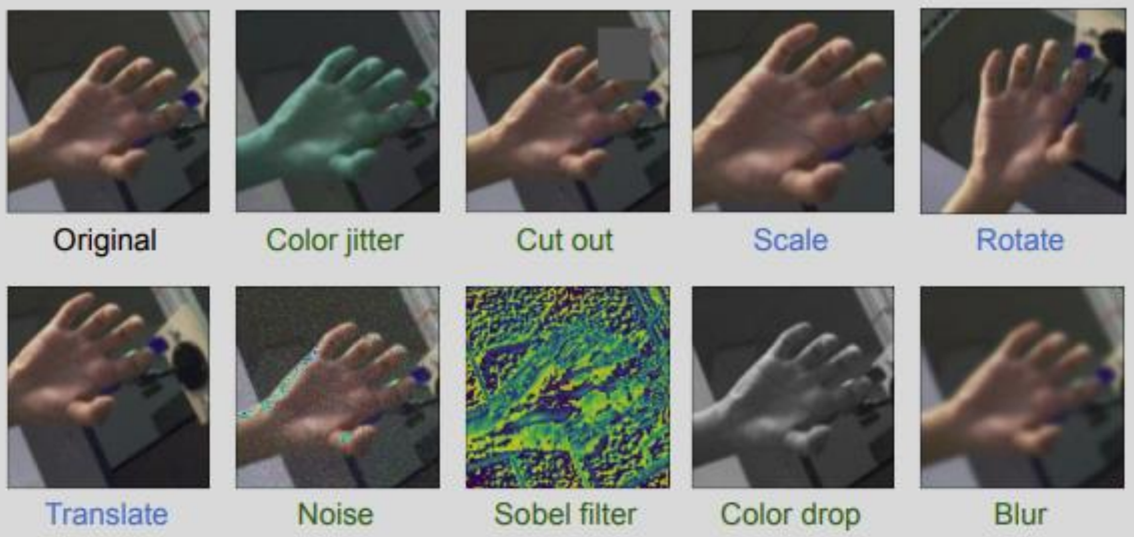

(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)
(f) Rotate {90°, 180°, 270°} (g) Cutout (h) Gaussian noise (i) Gaussian blur (j) Sobel filtering

# Contrastive learning

| | 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 |

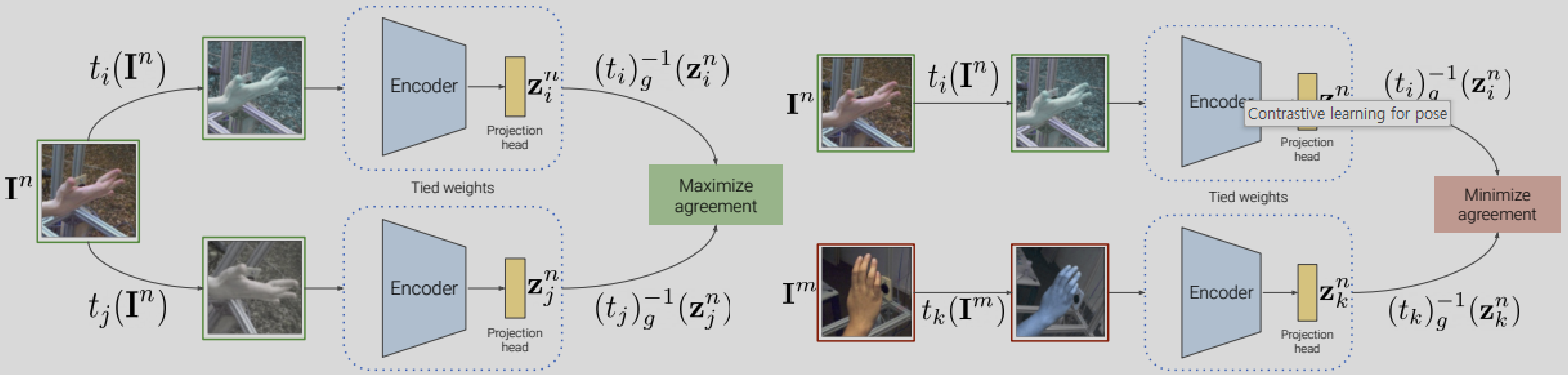
# Contrastive learning for pose



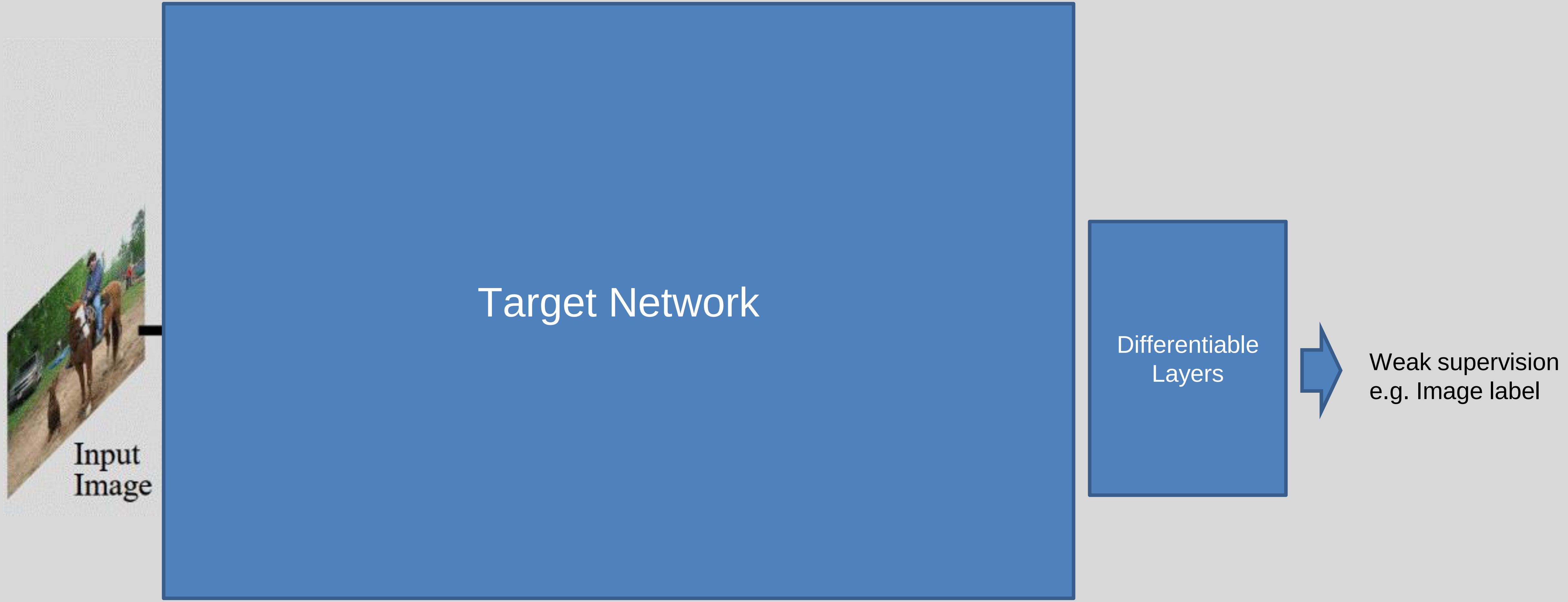

Self-Supervised 3D Hand Pose Estimation from monocular RGB via Contrastive Learning, ICCV’21

# Contrastive learning for pose

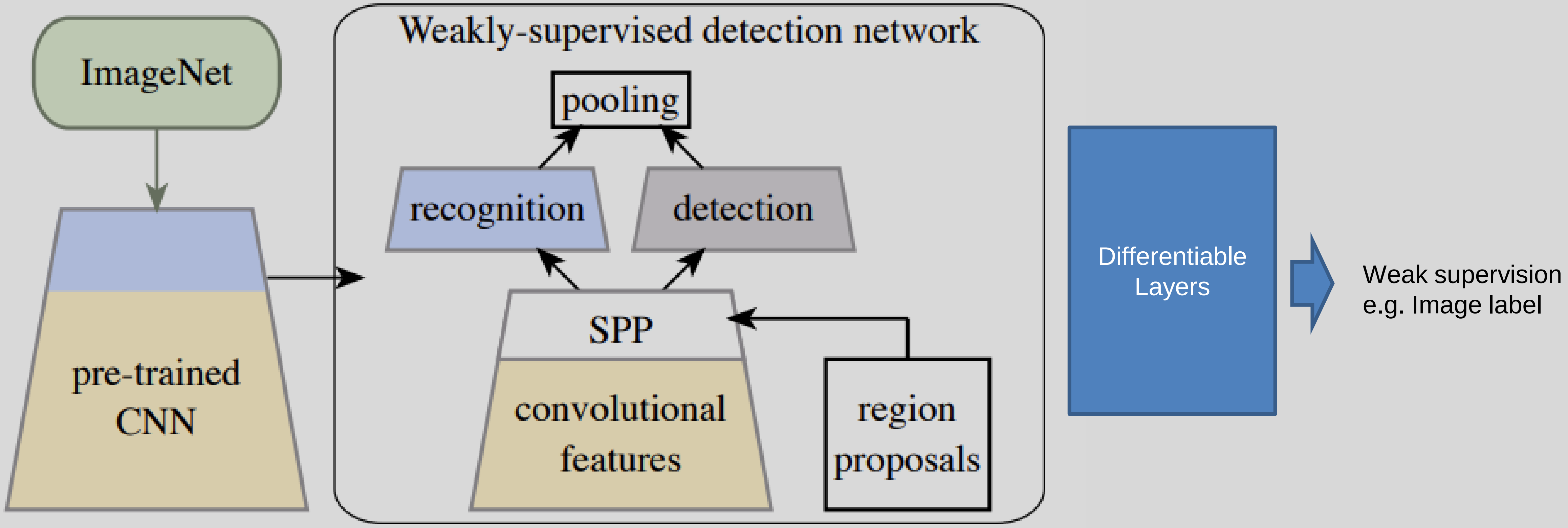



**The agreement between projections from the same input image is maximized (left) and agreements amongst projections from different input images are minimized (right)**

# Weakly-supervised learning



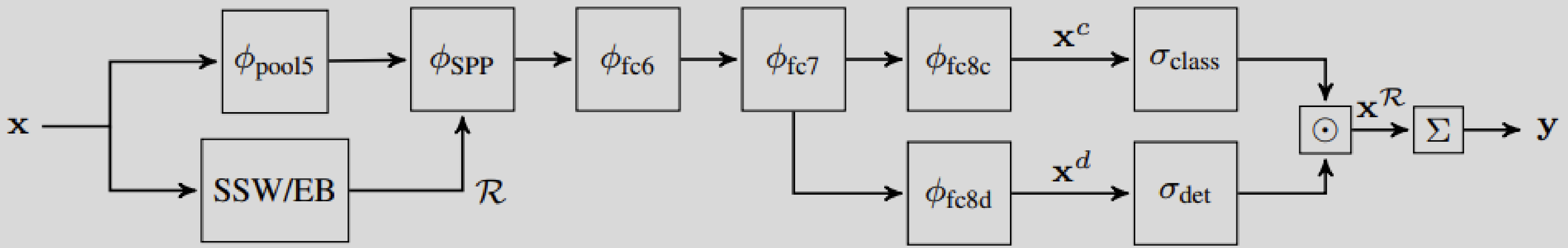
# Weakly-supervised object detection




Weakly Supervised Deep Detection Networks, CVPR'16

# Weakly-supervised object detection

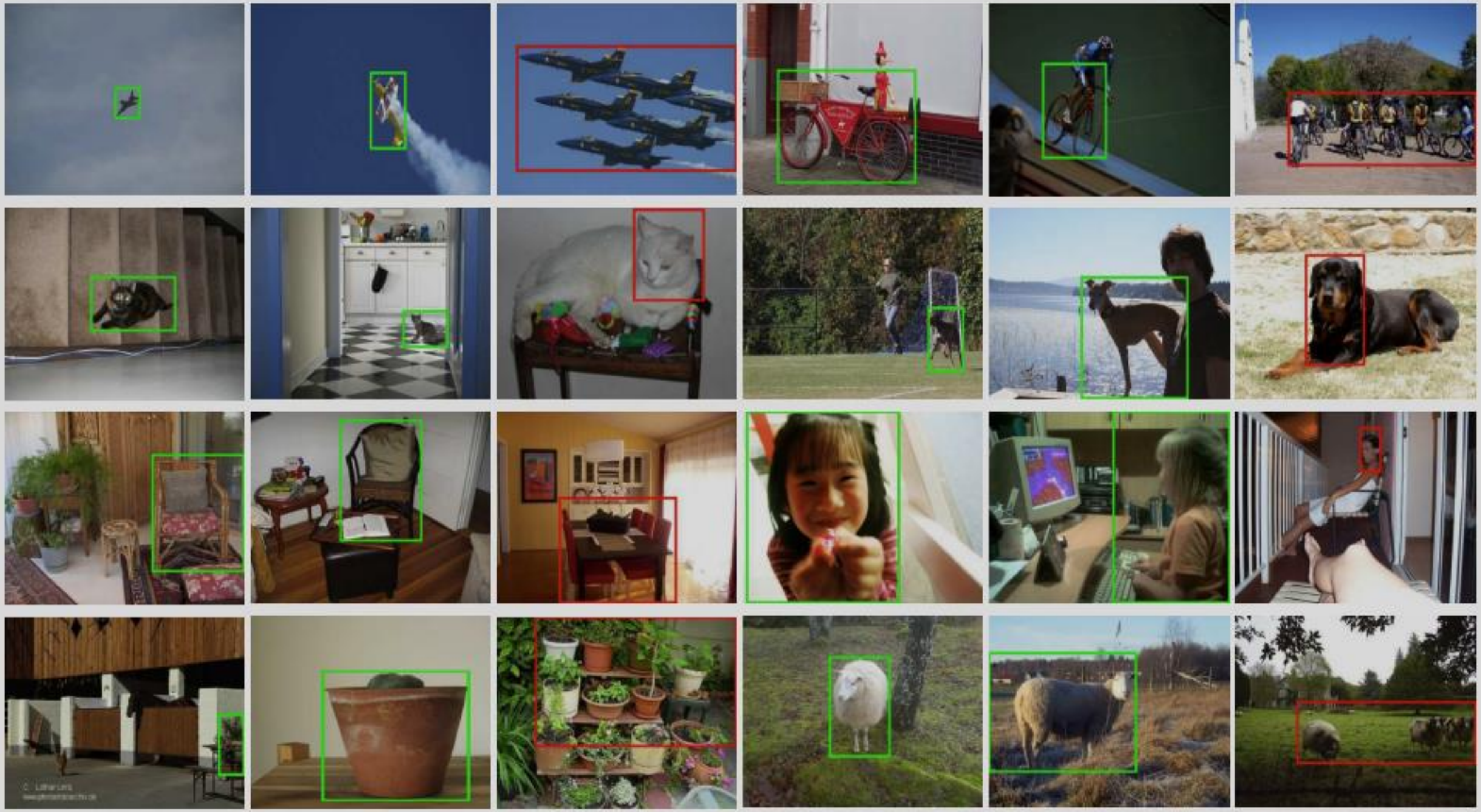
$$[\sigma_{\text{class}}(\mathbf{x}^c)]_{ij} = \frac{e^{x^c_{ij}}}{\sum_{k=1}^{C} e^{x^c_{kj}}}$$



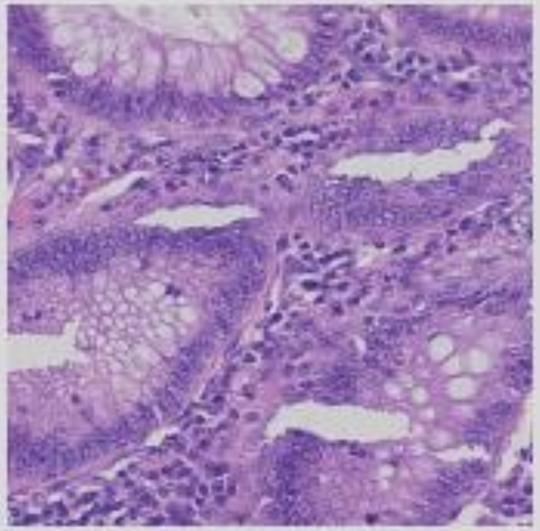

$$y_c = \sum_{r=1}^{|\mathcal{R}|} x^{\mathcal{R}}_{cr}$$

$$[\sigma_{\text{det}}(\mathbf{x}^d)]_{ij} = \frac{e^{x^d_{ij}}}{\sum_{k=1}^{|\mathcal{R}|} e^{x^d_{ik}}}$$

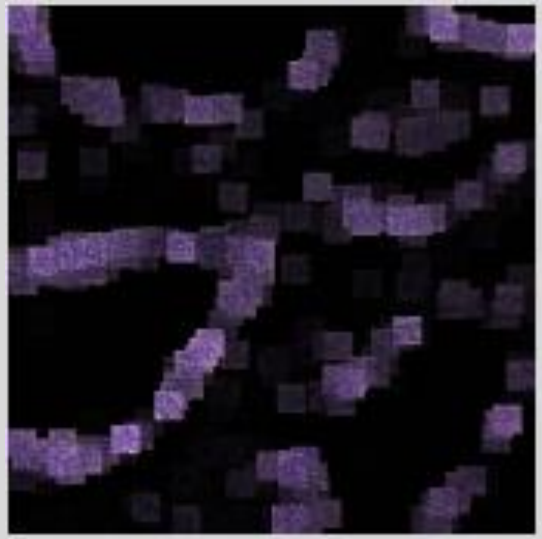
# Weakly-supervised object detection



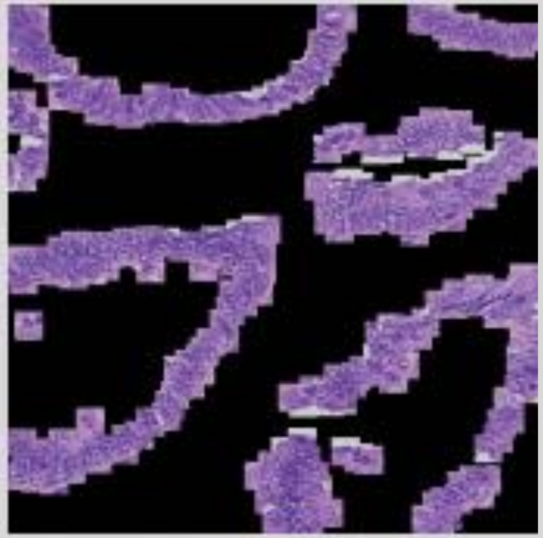
# Weakly-supervised segmentation



Original image



Predicted patch weights



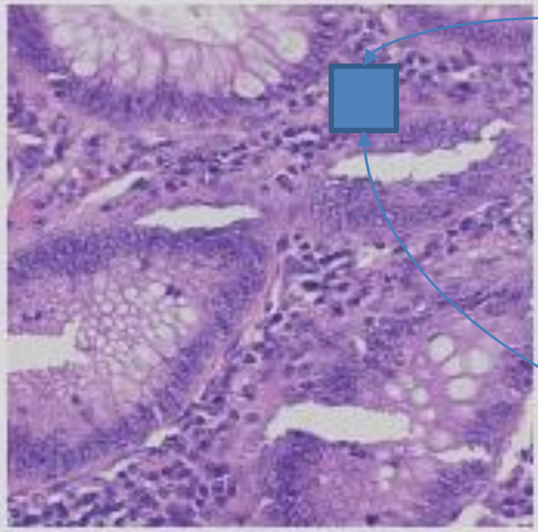
Ground-truth patches

Attention-based Deep Multiple Instance Learning, ICML'18

# Weakly-supervised segmentation

$$\mathbf{z} = \sum_{k=1}^{K} a_k \mathbf{h}_k$$

$$a_k = \frac{\exp\{\mathbf{w}^\top \tanh\left(\mathbf{V}\mathbf{h}_k^\top\right)\}}{\sum_{j=1}^{K} \exp\{\mathbf{w}^\top \tanh\left(\mathbf{V}\mathbf{h}_j^\top\right)\}}$$



Original image

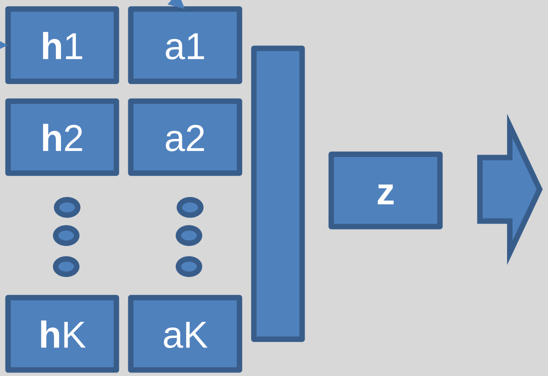
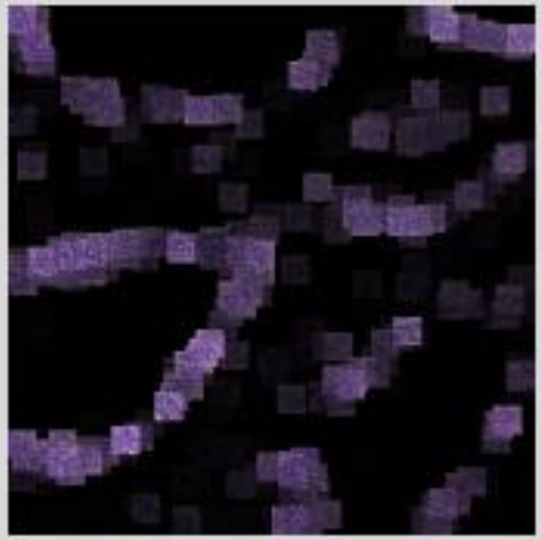



Image label



Predicted patch weights

# Thank you!