Computer Vision; Image Classification; Data-efficient Learning

Maziar Raissi

Assistant Professor

Department of Applied Mathematics

University of Colorado Boulder

maziar.raissi@colorado.edu

– un/self/semi-supervised learning

– representation learning and/or label propagation

Self-training with Noisy Student improves ImageNet classification

Boulder

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 θ^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

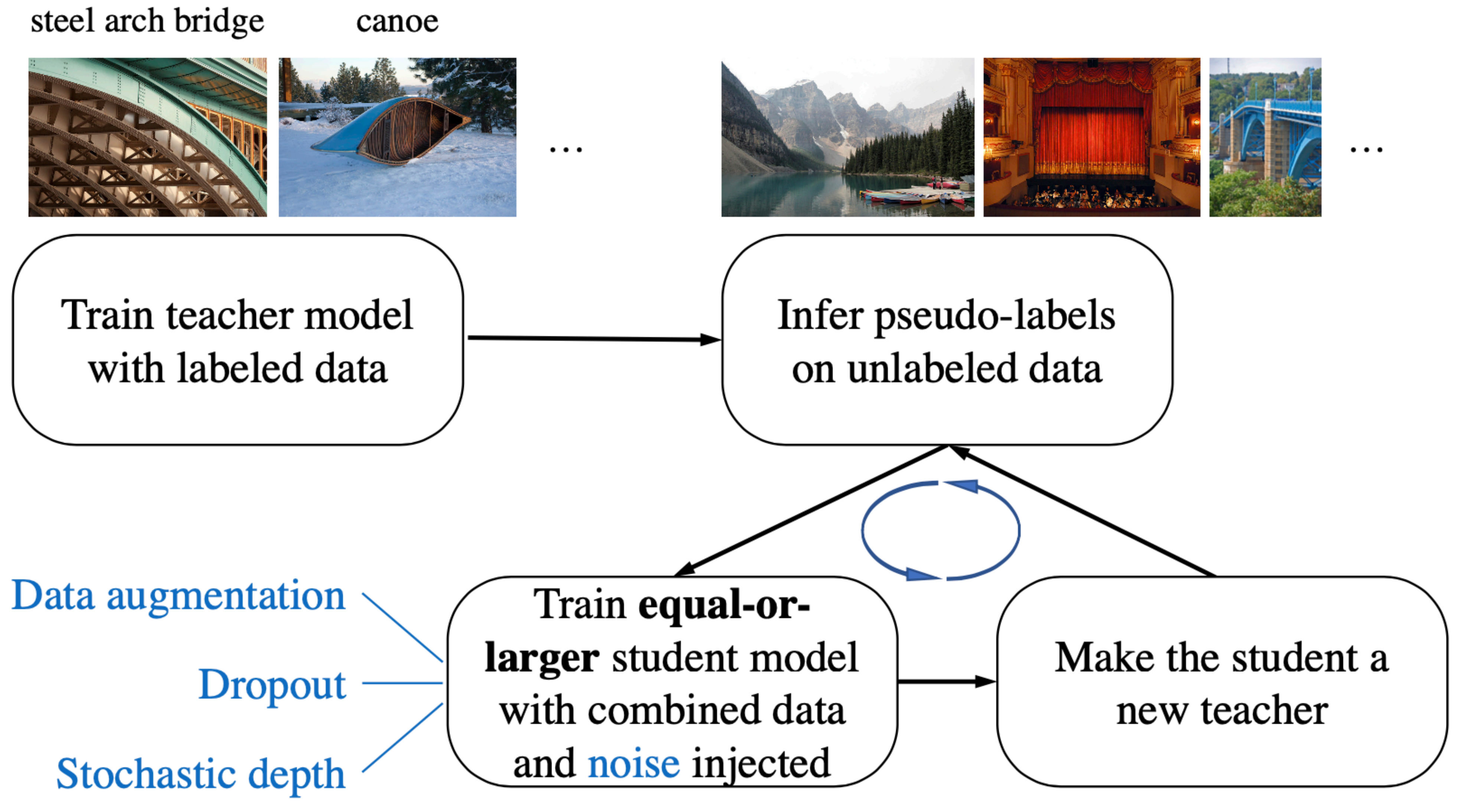
$$\tilde{y}_i = f(\tilde{x}_i, \theta^t_*), \forall i = 1, \cdots, m$$

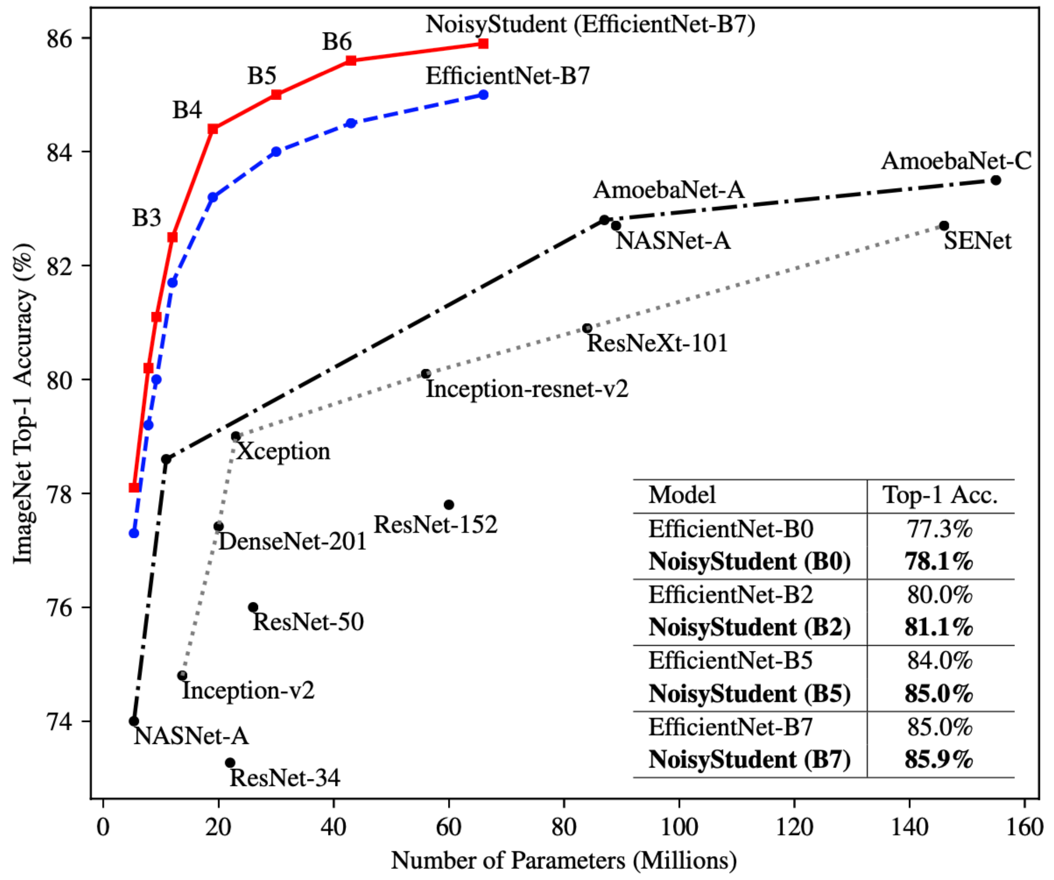
3: Learn an **equal-or-larger** student model θ^s_* which minimizes the cross entropy loss on labeled images and unlabeled images with **noise** added to the student model

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

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

	ImageNet top-1 acc.	ImageNet-A top-1 acc.	ImageNet-C mCE	ImageNet-P mFR
Prev. SOTA	86.4%	61.0%	45.7	27.8
NoisyStudent	**88.4%**	**83.7%**	**28.3**	**12.2**





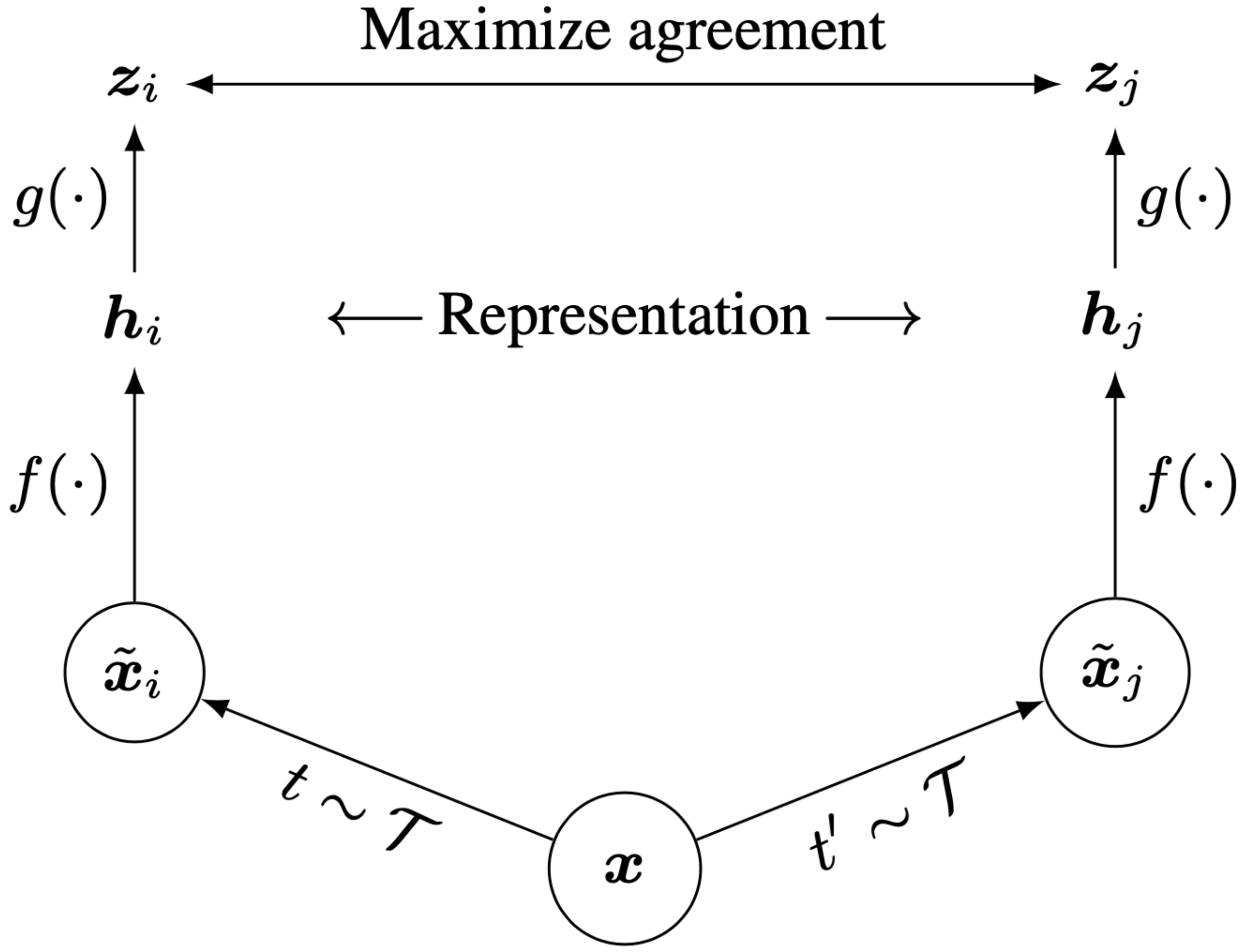
ImageNet-C and ImageNet-P test sets include images with common corruptions and perturbations such as blurring, fogging, rotation and scaling. ImageNet-A test set consists of difficult images that cause significant drops in accuracy to state-of-the-art models. These test sets are considered as “robustness” benchmarks.

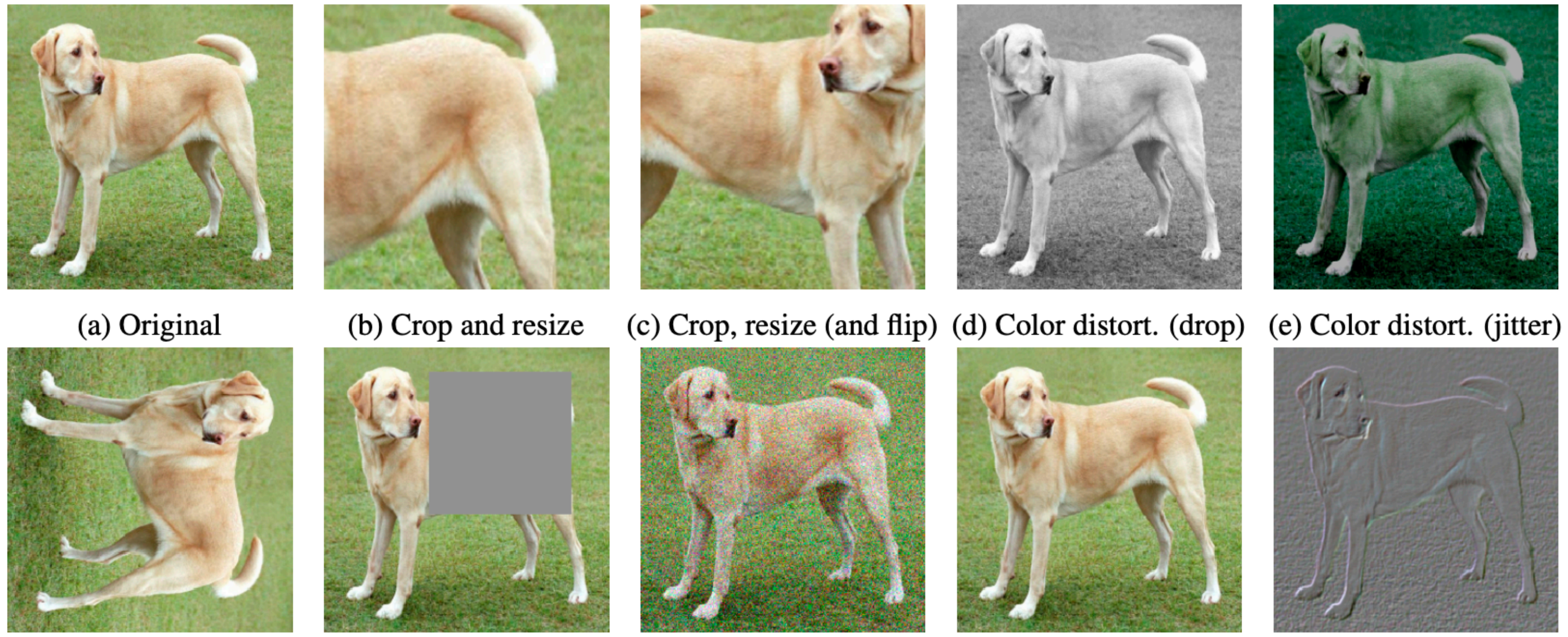
mCE (mean corruption error) is the weighted average of error rate on different corruptions, with AlexNet’s error rate as a baseline (lower is better). mFR (mean flip rate) measures the model's probability of flipping predictions under perturbations with AlexNet as a baseline (lower is better).

(a) ImageNet-A (b) ImageNet-C (c) ImageNet-P

Xie, Qizhe, et al. "Self-training with noisy student improves imagenet classification." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.

A Simple Framework for Contrastive Learning of Visual Representations





(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

$\tilde{x}_i, \tilde{x}_j \rightarrow$ two correlated views of the same example

Data Augmentation

Random cropping and resizing to the original size
Random color distortions
Random Gaussian blur

Base Encoder

$h_i = f(\tilde{x}_i) = \mathrm{ResNet}(\tilde{x}_i)$

$h_i \in \mathbb{R}^d \rightarrow$ output after average pooling layer

Projection Head

$$z_i = g(h_i) = W^2 \underbrace{\sigma}_{\text{ReLU}}(W^1 h_i)$$

Contrastive Loss

Contrastive Prediction Task: Given a set $\{\tilde{x}_k\}$ including a positive pair of examples $\tilde{x}_i$ and $\tilde{x}_j$, identify $\tilde{x}_j$ in $\{\tilde{x}_k\}_{k \neq i}$ for a given $\tilde{x}_i$.

minibatch: N examples $\implies 2N$ data datapoints

Given a positive pair, treat the other $2(N-1)$ augmented examples as negatives

$\mathrm{sim}(u, v) = u^T v / \|u\| \|v\| \rightarrow$ cosine similarity

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

NT-Xent Loss (normalized temperature-scaled cross entropy loss)

$\tau \rightarrow$ temperature

The final loss is computed over all positive pairs (i, j) & (j, i).

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

Self-supervised learning on ImageNet

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**

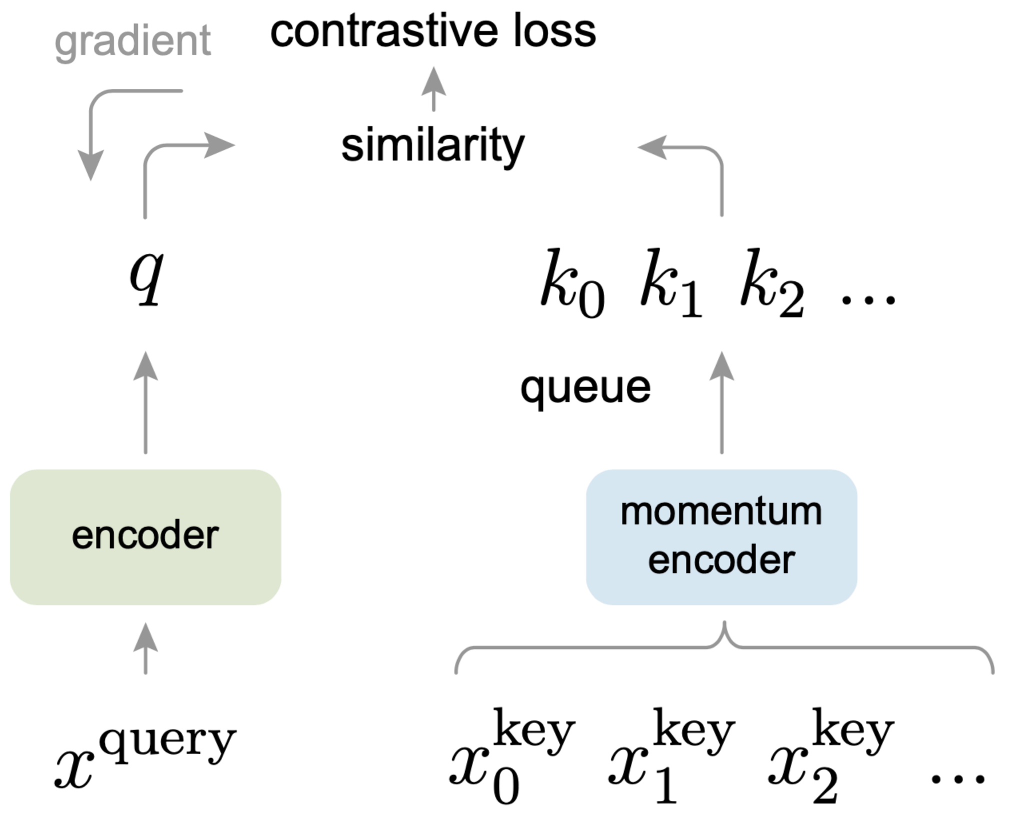
Semi-supervised learning on ImageNet

Method	Architecture	Label fraction 1%	10%
		Top 5	
Supervised baseline	ResNet-50	48.4	80.4
Methods using other label-propagation:			
Pseudo-label	ResNet-50	51.6	82.4
VAT+Entropy Min.	ResNet-50	47.0	83.4
UDA (w. RandAug)	ResNet-50	-	88.5
FixMatch (w. RandAug)	ResNet-50	-	89.1
S4L (Rot+VAT+En. M.)	ResNet-50 (4×)	-	91.2
Methods using representation learning only:			
InstDisc	ResNet-50	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**

Chen, Ting, et al. "A simple framework for contrastive learning of visual representations." *arXiv preprint arXiv:2002.05709* (2020).

Momentum Contrast for Unsupervised Visual Representation Learning

Boulder



Keys (tokens): samples from the data (e.g., images or patches) represented by an encoder network

Unsupervised Learning: dictionary look-up

```
An encoded "query" should be
similar to its matching "key"
and dissimilar to others
```

Instance discrimination task

A query matches a key if they are encoded views (e.g., different crops) of the same image

Contrastive learning

$q \to$ encoded query

$\{k_0, k_1, \ldots\} \to$ set of encoded samples (keys of a dictionary)

$k_+ \to$ the single key in the dictionary that matches q

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)}$$

InfoNCE (noise-contrastive estimation) loss

$\tau \to$ temperature hyper-parameter

$q = f_q(x^q)$ query sample

encoder network

$k = f_k(x^k)$

Momentum Contrast (MoCo)

① dynamic ② large ③ consistent dictionary

dictionary as queue

```
The current mini-batch is enqueued to the dictionary,
and the oldest mini-batch in the queue is removed
```

$\theta_k \to$ parameters of f_k

$\theta_q \to$ parameters of f_q

$\theta_k \leftarrow m\theta_k + (1-m)\theta_q, m \in [0, 1), m = 0.999$

$\theta_q \to$ updated by backprop

Object detection fine-tuned on PASCAL VOC

pre-train	AP_{50}	AP	AP_{75}
random init.	60.2	33.8	33.1
super. IN-1M	81.3	53.5	58.8
MoCo IN-1M	81.5 (+0.2)	55.9 (**+2.4**)	62.6 (**+3.8**)
MoCo IG-1B	82.2 (**+0.9**)	57.2 (**+3.7**)	63.7 (**+4.9**)

Faster R-CNN

ImageNet-1M (IN-1M), Instagram-1B (IG-1B), super. (supervised)

Algorithm 1 Pseudocode of MoCo in a PyTorch-like style.

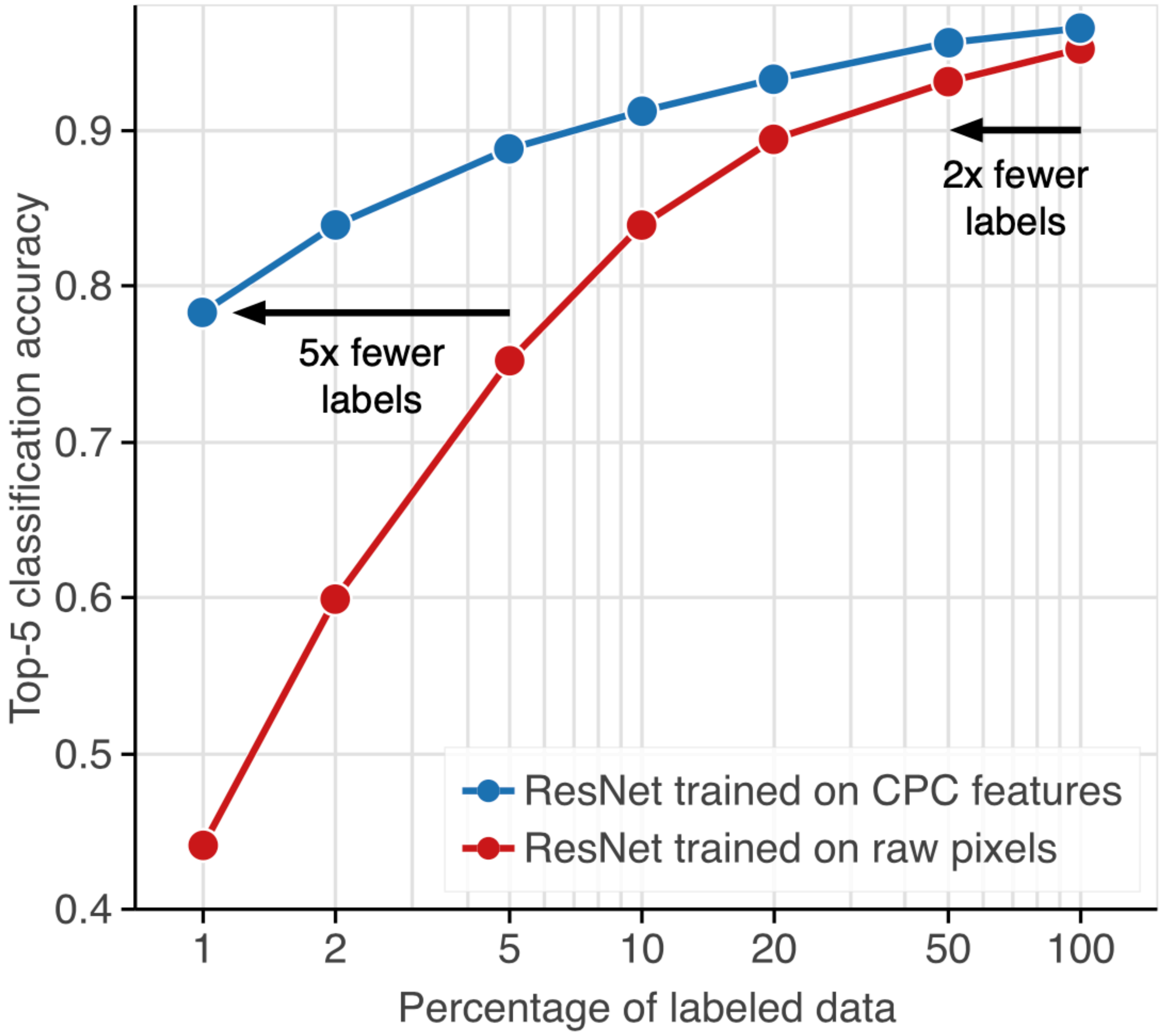
```
# f_q, f_k: encoder networks for query and key
# queue: dictionary as a queue of K keys (CxK)
# m: momentum
# t: temperature

f_k.params = f_q.params # initialize
for x in loader: # load a minibatch x with N samples
   x_q = aug(x) # a randomly augmented version
   x_k = aug(x) # another randomly augmented version

   q = f_q.forward(x_q) # queries: NxC
   k = f_k.forward(x_k) # keys: NxC
   k = k.detach() # no gradient to keys

   # positive logits: Nx1
   l_pos = bmm(q.view(N,1,C), k.view(N,C,1))

   # negative logits: NxK
   l_neg = mm(q.view(N,C), queue.view(C,K))

   # logits: Nx(1+K)
   logits = cat([l_pos, l_neg], dim=1)

   # contrastive loss, Eqn.(1)
   labels = zeros(N) # positives are the 0-th
   loss = CrossEntropyLoss(logits/t, labels)

   # SGD update: query network
   loss.backward()
   update(f_q.params)

   # momentum update: key network
   f_k.params = m*f_k.params+(1-m)*f_q.params

   # update dictionary
   enqueue(queue, k) # enqueue the current minibatch
   dequeue(queue) # dequeue the earliest minibatch
```

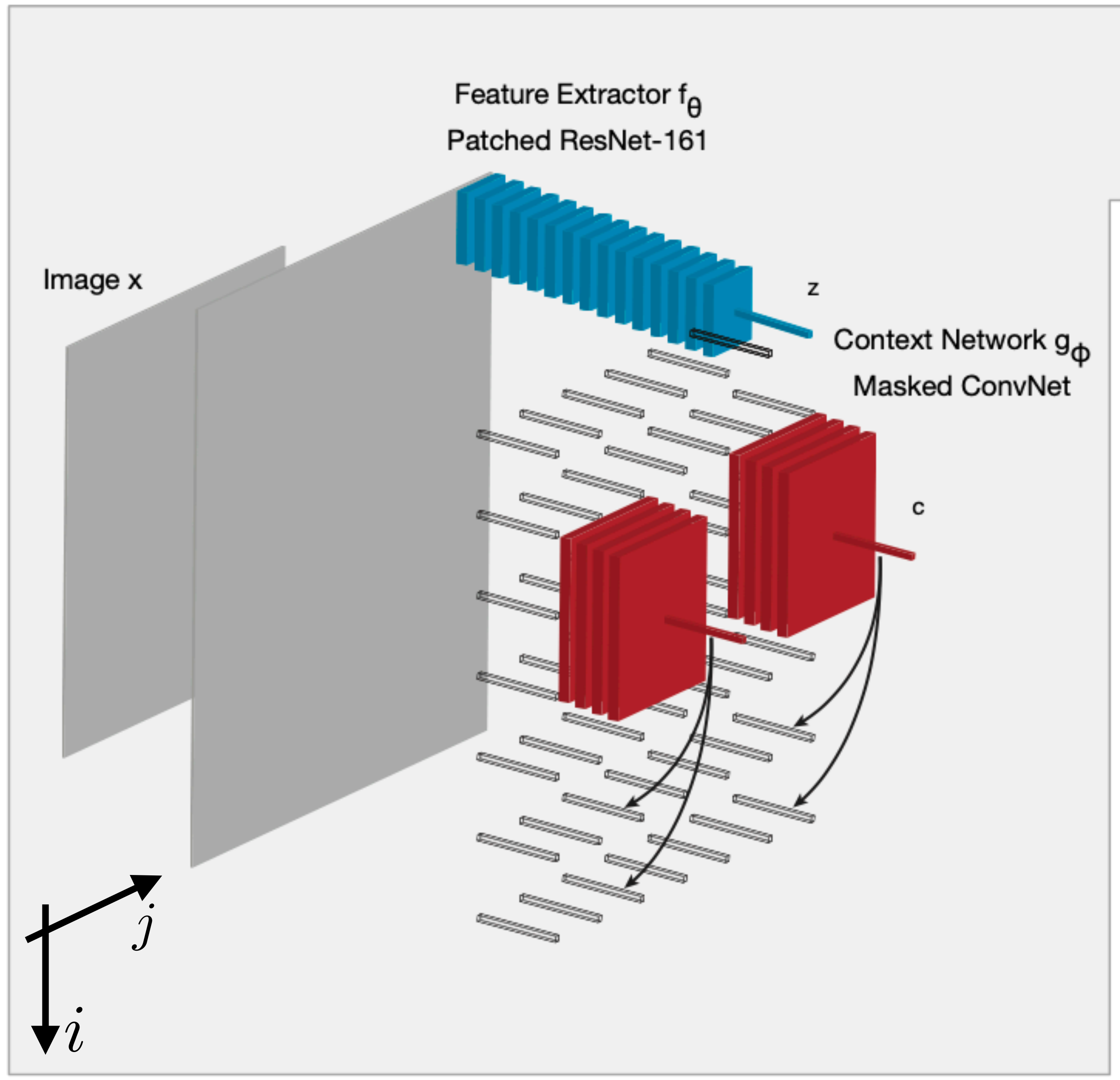
bmm: batch matrix multiplication; mm: matrix multiplication; cat: concatenation.

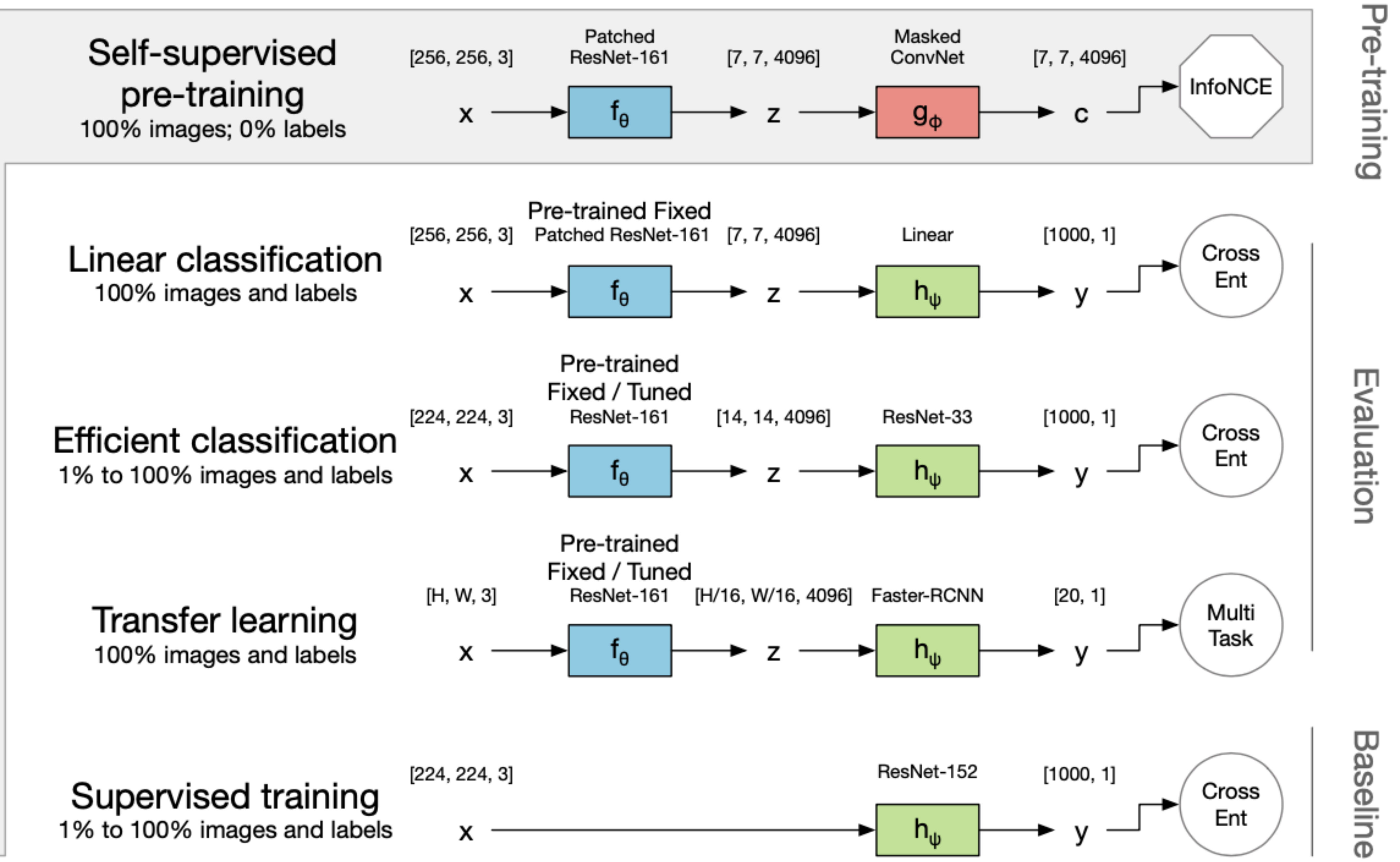
He, Kaiming, et al. "Momentum contrast for unsupervised visual representation learning." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.

Data-Efficient Image Recognition with Contrastive Predictive Coding

Boulder







$\{x_{i,j}\}$ → a grid of overlapping patches dividing each input image

i, j → location of the patch

$z_{i,j} = f_\theta(x_{i,j})$ → encoded patch

neural network

g_ϕ → masked convolutional network (applied to the grid of feature vectors)

$c_{i,j} = g_\phi(\{z_{u,v}\}_{u \le i,v})$ → context vector

feature vectors that lie above i, j

$z_{i+k,j}$ → "future" feature vectors to be predicted from current context vector $c_{i,j}$

k → prediction length

$\hat{z}_{i+k,j} = W_k c_{i,j}$

prediction matrix

Correctly recognize the target $z_{i+k,j}$ among a set of randomly sampled feature vectors $\{z_l\}$ from the dataset (i.e., other locations in the image and other images in the mini-batch)

$$\mathcal{L}_{\text{CPC}} = -\sum_{i,j,k} \log p(\boldsymbol{z}_{i+k,j} | \hat{\boldsymbol{z}}_{i+k,j}, \{\boldsymbol{z}_l\})$$

$$= -\sum_{i,j,k} \log \frac{\exp(\hat{\boldsymbol{z}}_{i+k,j}^T \boldsymbol{z}_{i+k,j})}{\exp(\hat{\boldsymbol{z}}_{i+k,j}^T \boldsymbol{z}_{i+k,j}) + \sum_l \exp(\hat{\boldsymbol{z}}_{i+k,j}^T \boldsymbol{z}_l)}$$

InfoNCE (Noise Contrastive Estimation)

– increase depth & width – layer normalization – making predictions in all four directions – extensive patch-based data augmentation

Henaff, Olivier. "Data-efficient image recognition with contrastive predictive coding." *International Conference on Machine Learning*. PMLR, 2020.

Questions?