Vision: Representation Learning

EECE454 Intro. to Machine Learning Systems

Fall 2024

Recap

- **Last class.** Network architectures for learning from images
    - Convolutional Layer
    - Convolutional Networks
        - Basic CNNs: Conv + FCs, ...
        - Deeper CNNs: Residual Connections, Bottlenecks ...
        - Lighter CNNs: Inverted Bottlenecks, ...
        - Scalable CNNs: Scaling depth / width / resolution, removing BN
- **Focus.** We can train large neural networks, using minimal computations

Recap

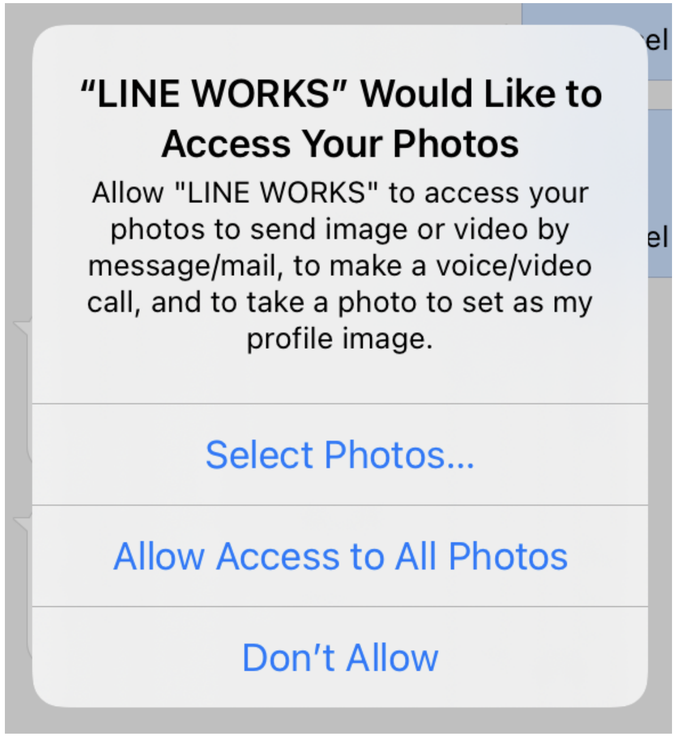
- **Today.** Key ideas in visual representation learning
  (i.e., learning useful features, with minimal supervision)
  - Data augmentations
  - Transfer learning
  - Semi-supervised Learning
  - Self-supervised Learning

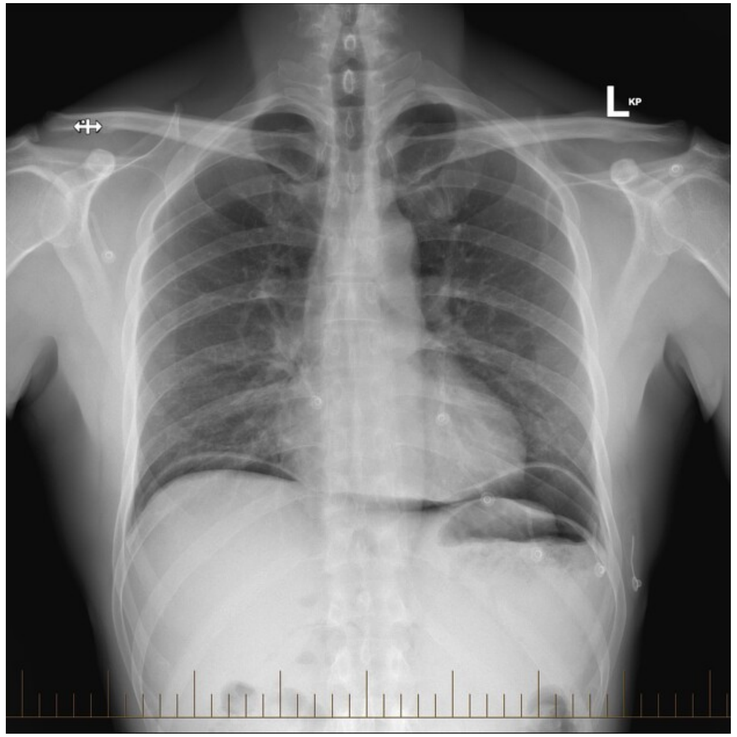
- **Focus.** We can utilize very large datasets, with minimal human labeling efforts

Visual Supervised Learning: Ideas for enlarging the dataset

Problem

- To get a high performance, we need both the large model & large data
- **Problem.** Collecting large-scale annotated visual dataset is quite difficult
  - Image collection
    - License & Privacy issues
    - Special domains (e.g.,medical)
  - Pre-processing
    - Resize & Formatting



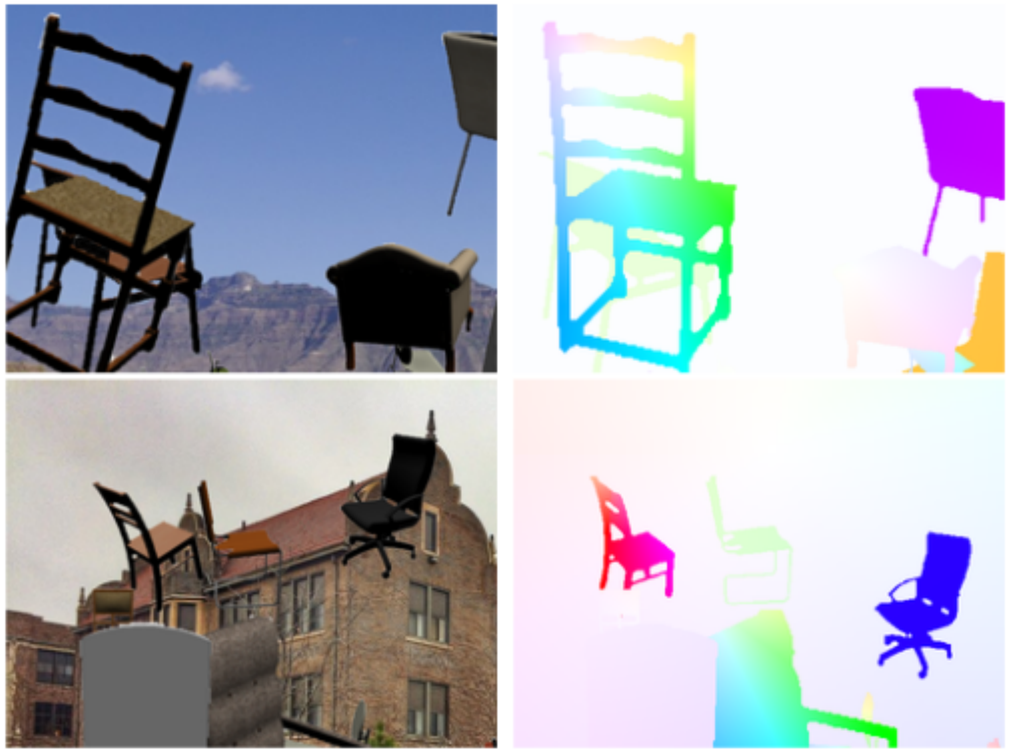


Problem

- To get a high performance, we need both the large model & large data
- **Problem.** Collecting large-scale annotated visual dataset is quite difficult
  - Image collection
    - License & Privacy issues
    - Special domains (e.g.,medical)
  - Pre-processing
    - Resize & Formatting
  - Labeling
    - Hiring multiple annotators
    - Requires expertise / labor
  - Post-processing
    - Quality review & balancing …

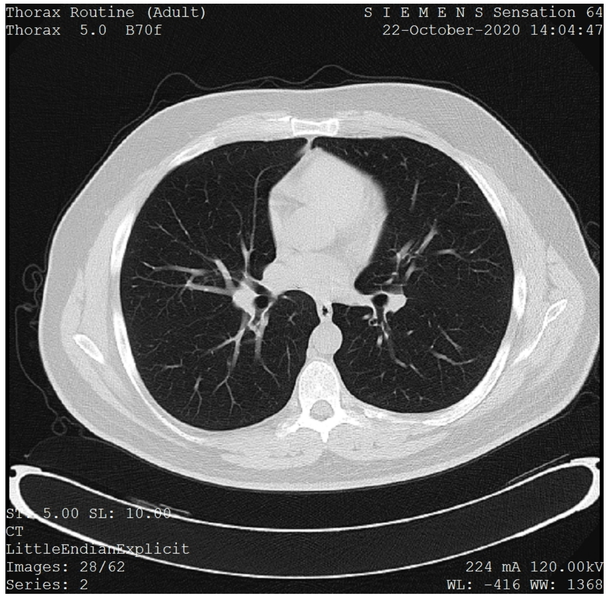
Overview

- **Idea.** Develop effective algorithms to utilize alternate data sources
- 1. Fake but realistic data
  - Data augmentations ✅
  - Synthetic dataset
  - Generated images



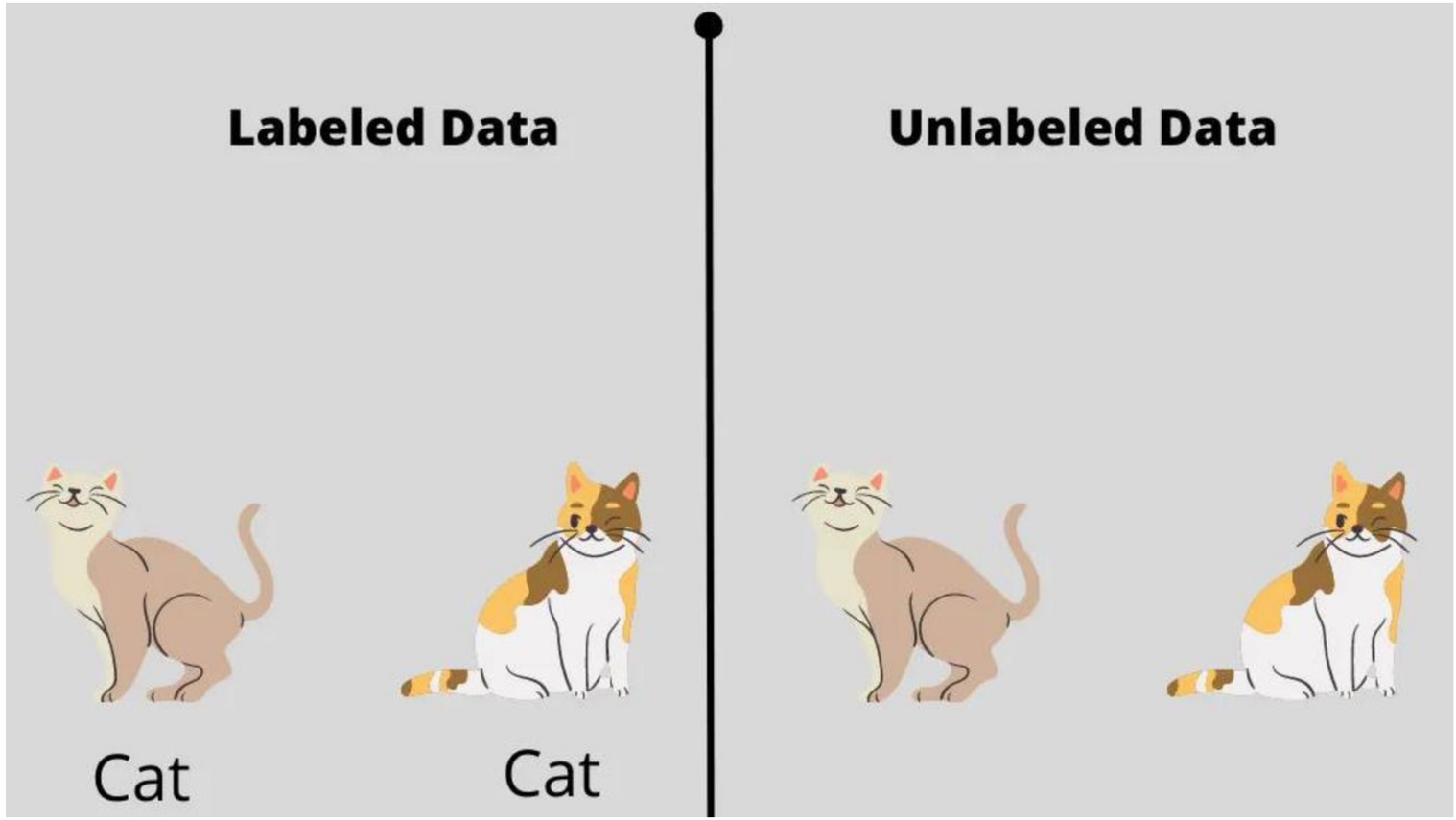
Overview

- 2. Data from related domains
  - Transfer learning ✅
  - Few-shot learning
  - Continual learning
  - Meta-learning



Overview

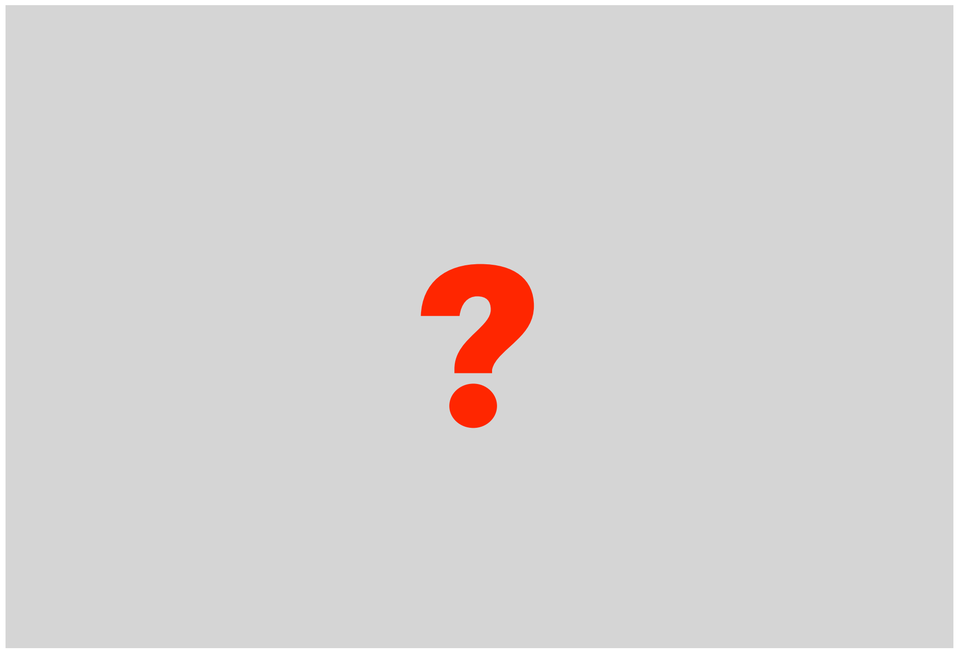
- 3. Utilize unlabeled data
  (recall: K-Means, PCA)
  - Semi-supervised Learning ✅
  - Self-supervised Learning ✅
  - Generative Modeling
    - Next class
  - Weakly supervised learning
    - when we talk about “multimodal learning”



Data augmentation

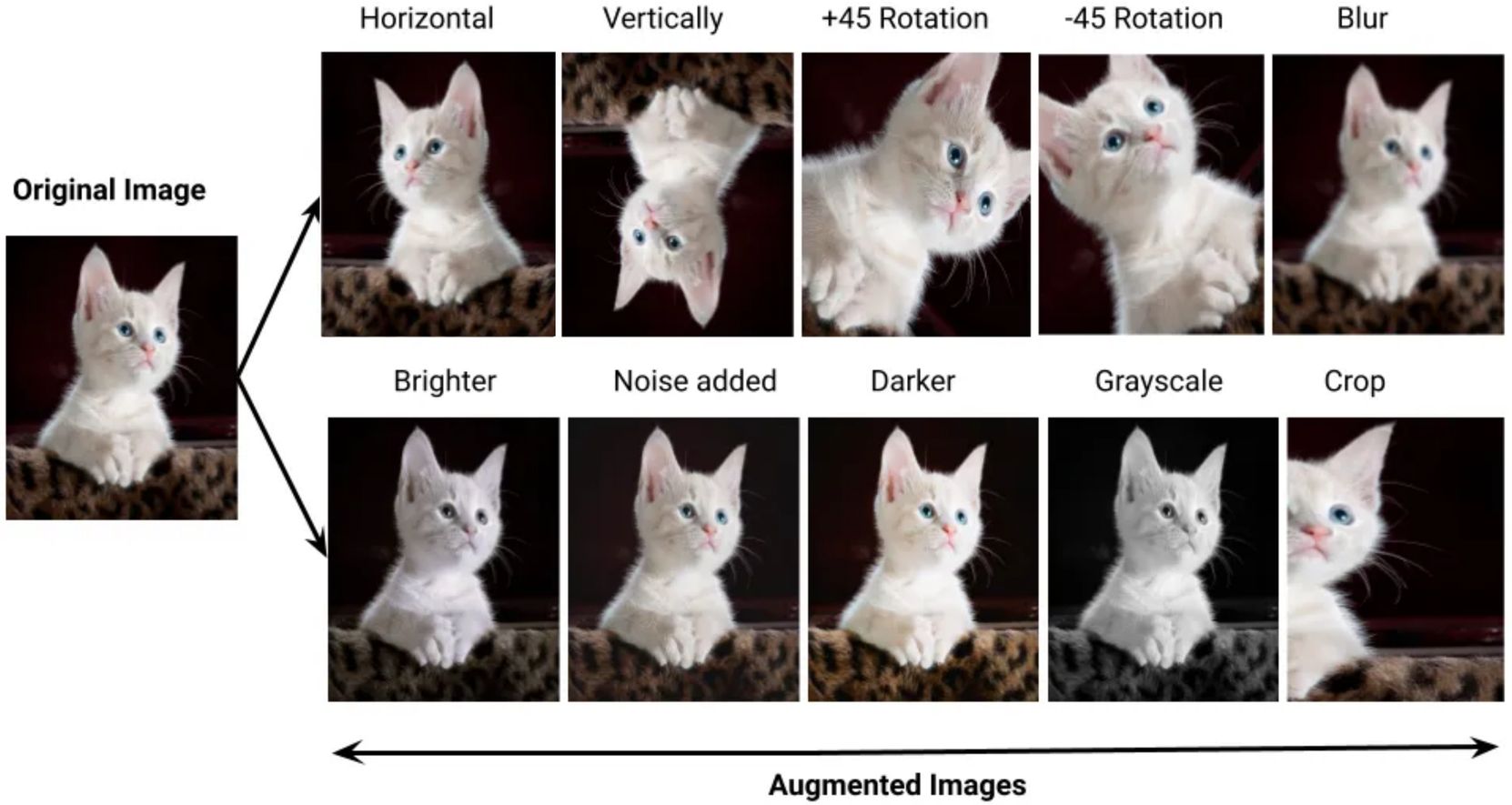
Data augmentation

- **Idea.** Apply generic operations on existing labeled data, for making a realistic yet new images
  - Important. Choose the operations that does not alter “how human view” the data

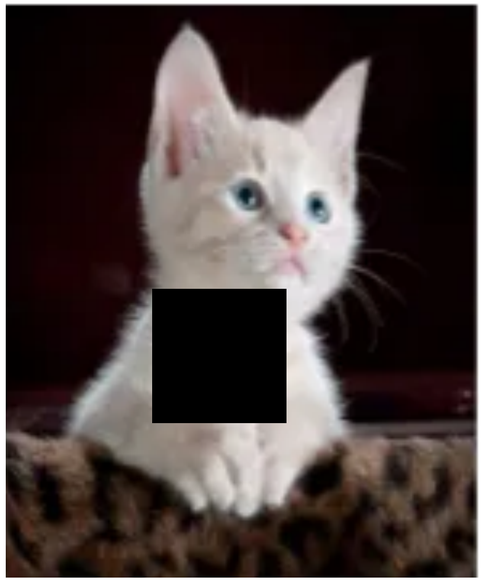


Data augmentation

- **Basic Operations.** Flipping, Resizing, Shifting, Cropping, Rotating, Blurring, Grayscale, Darker, …
  - Caution. In many cases, we avoid vertical flipping or extreme resizing (why?)
  - CutOut has also been popular, but less used nowadays

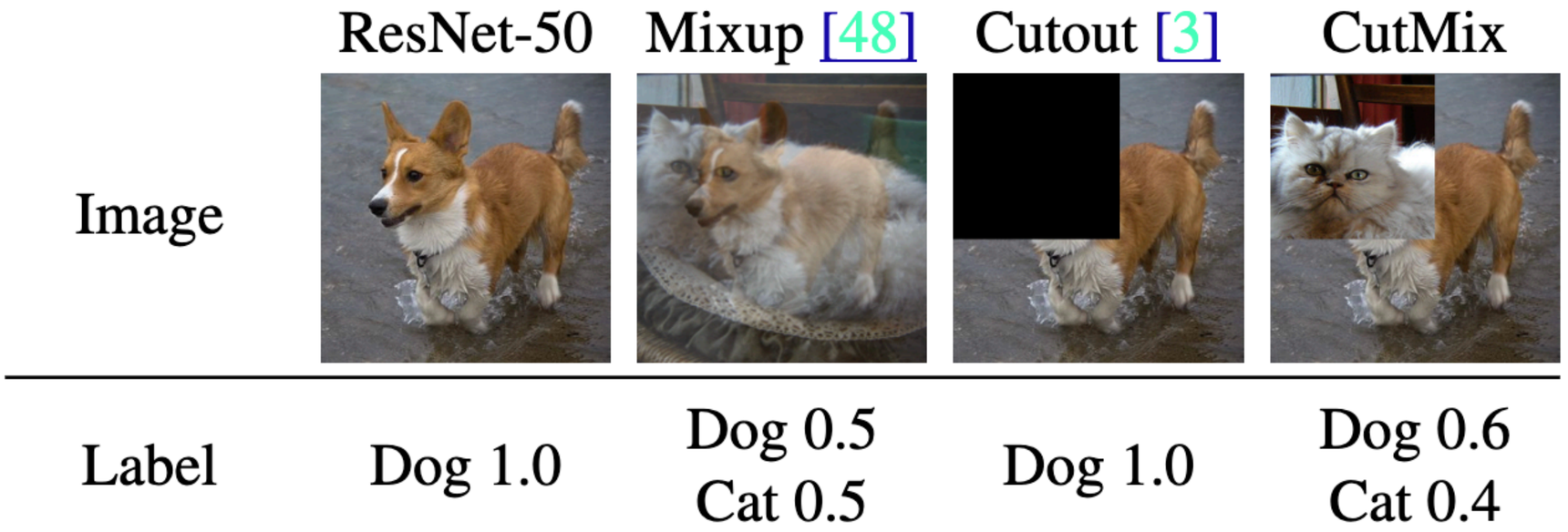


CutOut



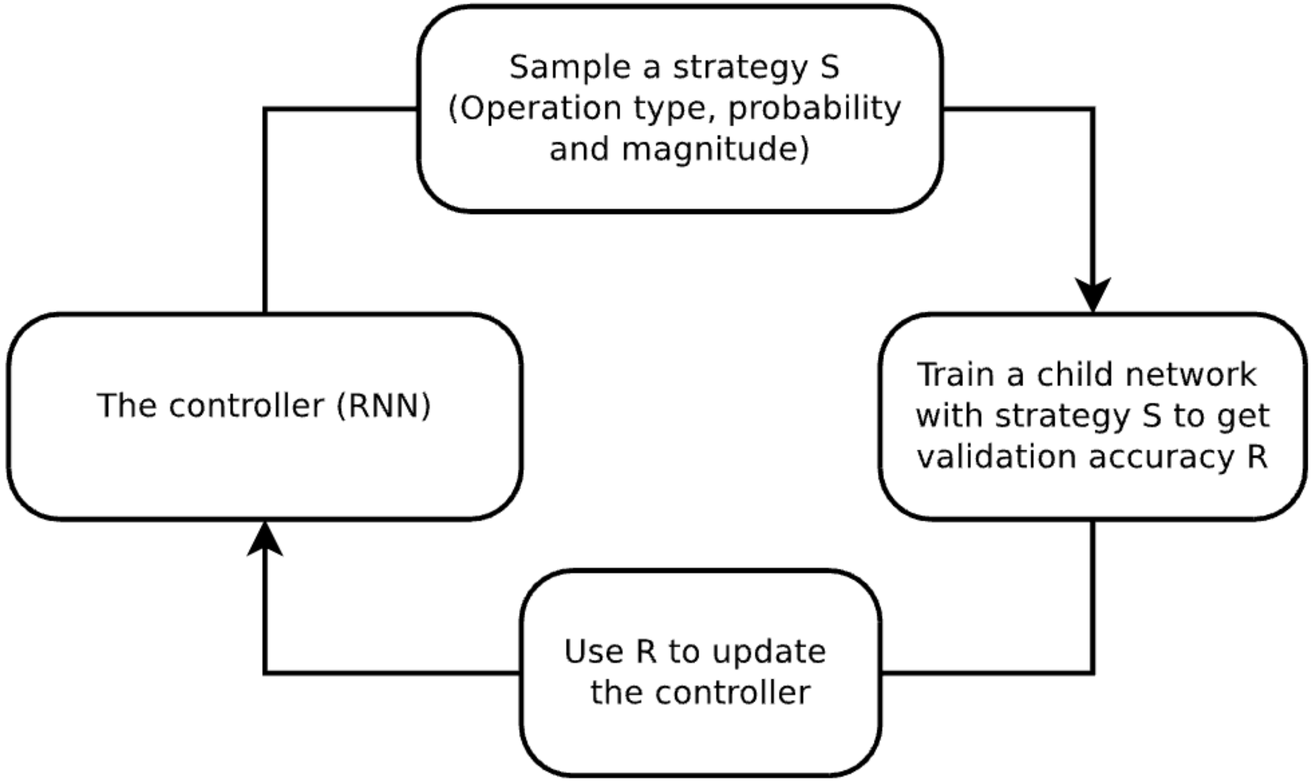
Data augmentation

- **Mixing Images.** It is also typical to mix multiple images to make one
  - SMOTE / Mixup. Overlay two or more images into one, and give mixed labels.
  - CutMix. Combine multiple pieces of images without overlapping.



Data augmentation

- **AutoAugment.** Train a image augmentation strategy that works best with the target dataset and the model pair
  - Requires a lot of training; followed by RandAugment, Fast AutoAugment ...

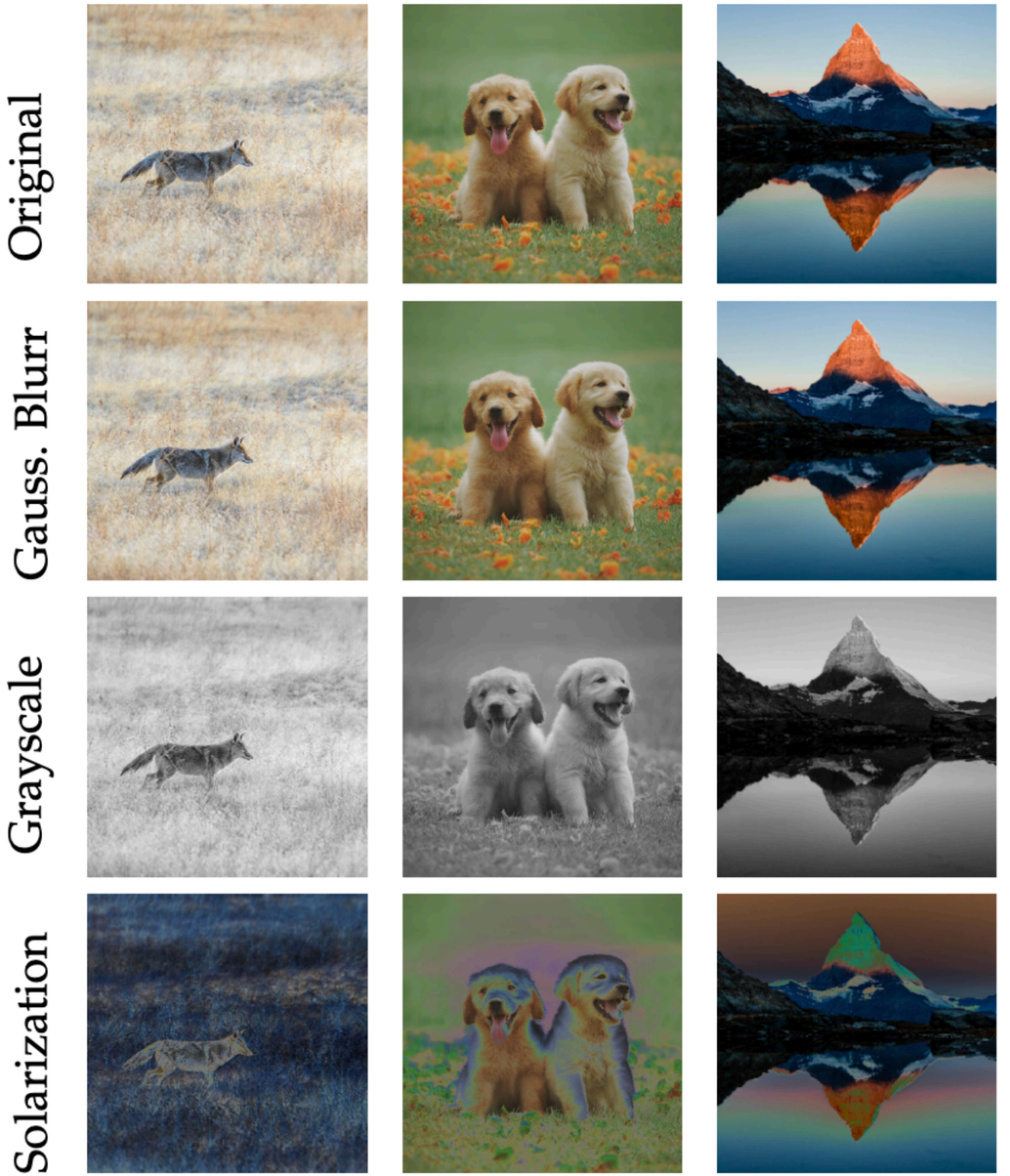


Touvron et al., “DeiT III: Revenge of the ViT,” ECCV 2022

Data augmentation

- Very recently, people avoid excessive augmentations
- Reason.
  - Augmentation makes us lose some information anyways, which is potentially very important
  - Self-supervised training works well, so we have less motivation for augmentation…
- Typical to use just:

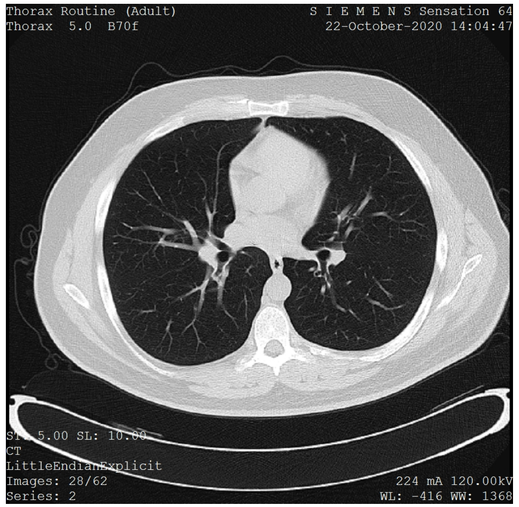
  “3 augmentations” + horizontal flip + color jitter



Transfer Learning

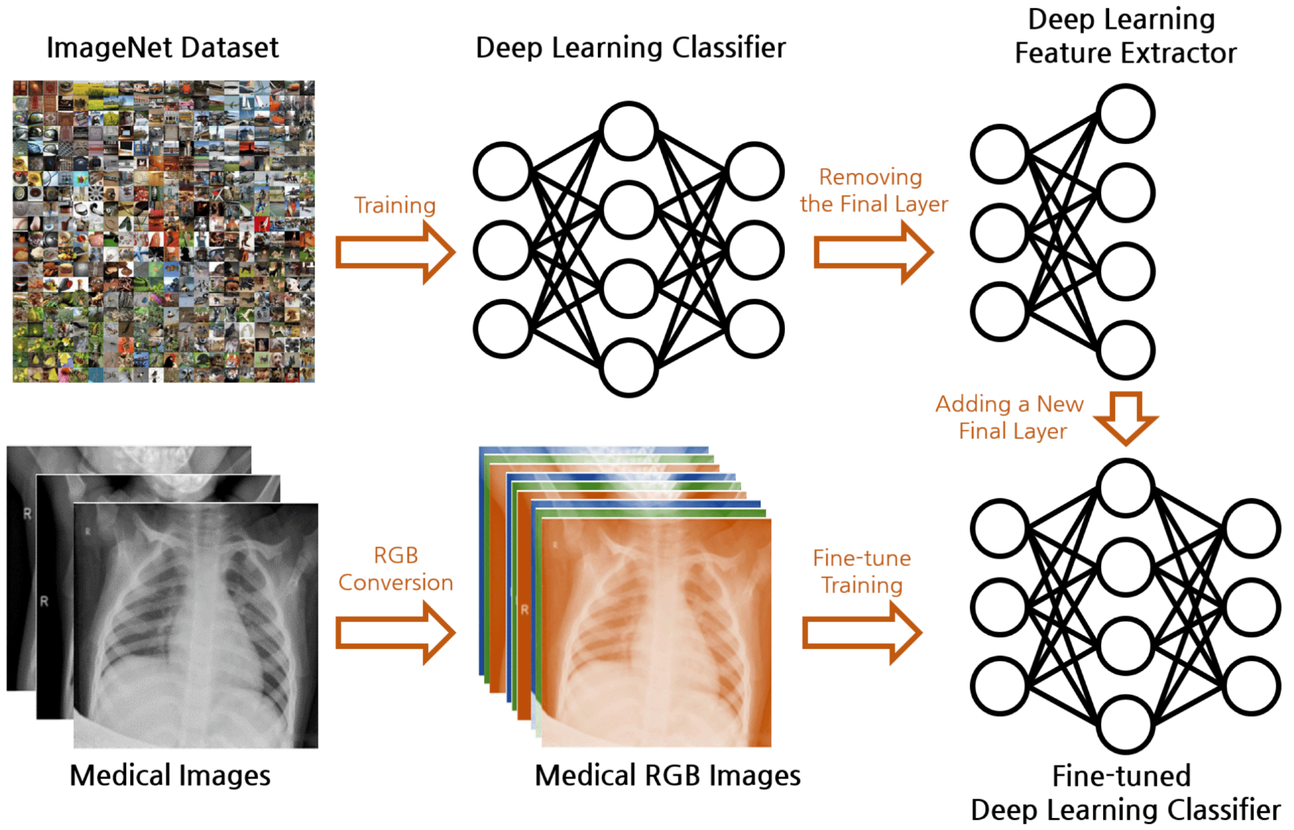
Transfer Learning

- **Setting.**
  - We have some domain with very scarce data
  - There exists some relevant domain with plenty of data
    - Natural images — Google has a 3B-scale dataset (JFT-3B)



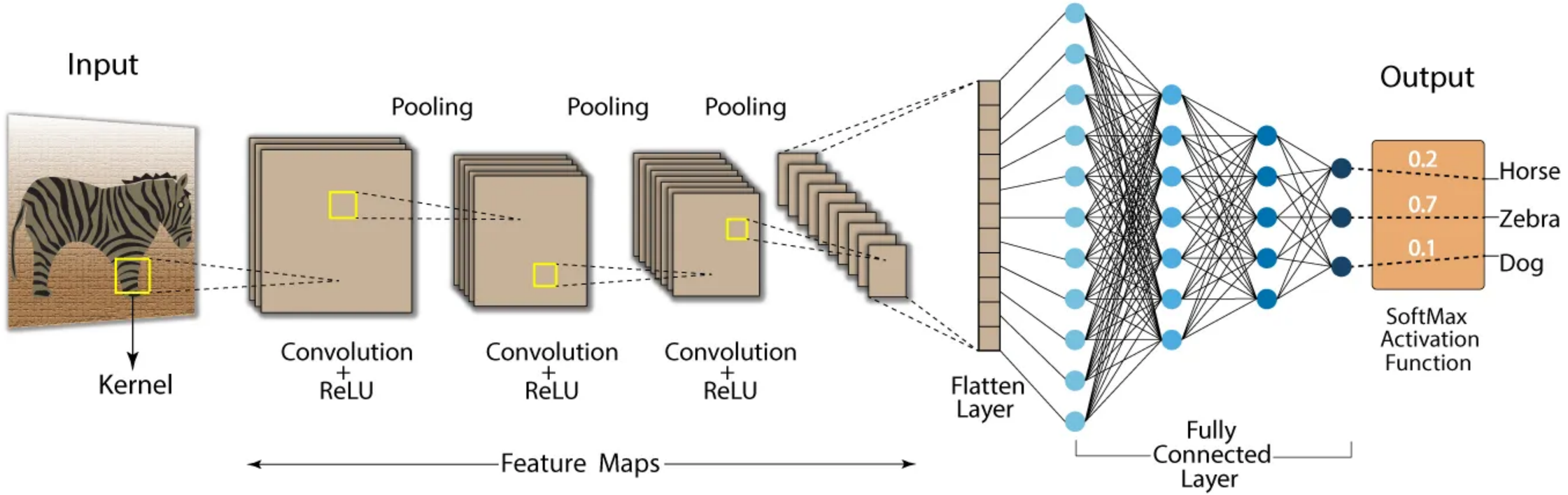
Transfer Learning

- **Setting.**
  - We have some domain with very scarce data
  - There exists some relevant domain with plenty of data
    - Natural images — Google has a 3B-scale dataset (JFT-3B)
- **Idea.**
  - First, train a model on the relevant domain (called pretraining)
  - Re-use the model weights and train further on original data
  - Intuition. There is some common knowledge shared across domains
    - neurons that discern shapes, such as “circles” or “triangles”



Transfer Learning

- There are many different ways to transfer the parameters.
- **Common.** Cut the final layer—the classification layer—off from the model.
  - Reason. The number of classes does not match

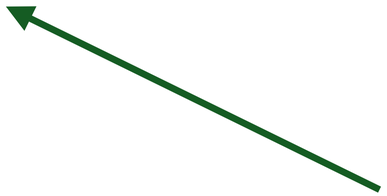


Transfer Learning

- There are many different ways to transfer the parameters.
- **Common.** Cut the final layer—the classification layer—off from the model.
  - Reason. The number of classes does not match
- **Difference.** Choose one option:
  - Freeze the pretrained weights and train only the newly added classification layer
  - Fine-tune all weights on the new data, without freezing them
  - Add additional neurons / layers and train them together, with or without freezing pretrained weights

- few target domain data
- small domain discrepancy

(worries about "forgetting useful info")

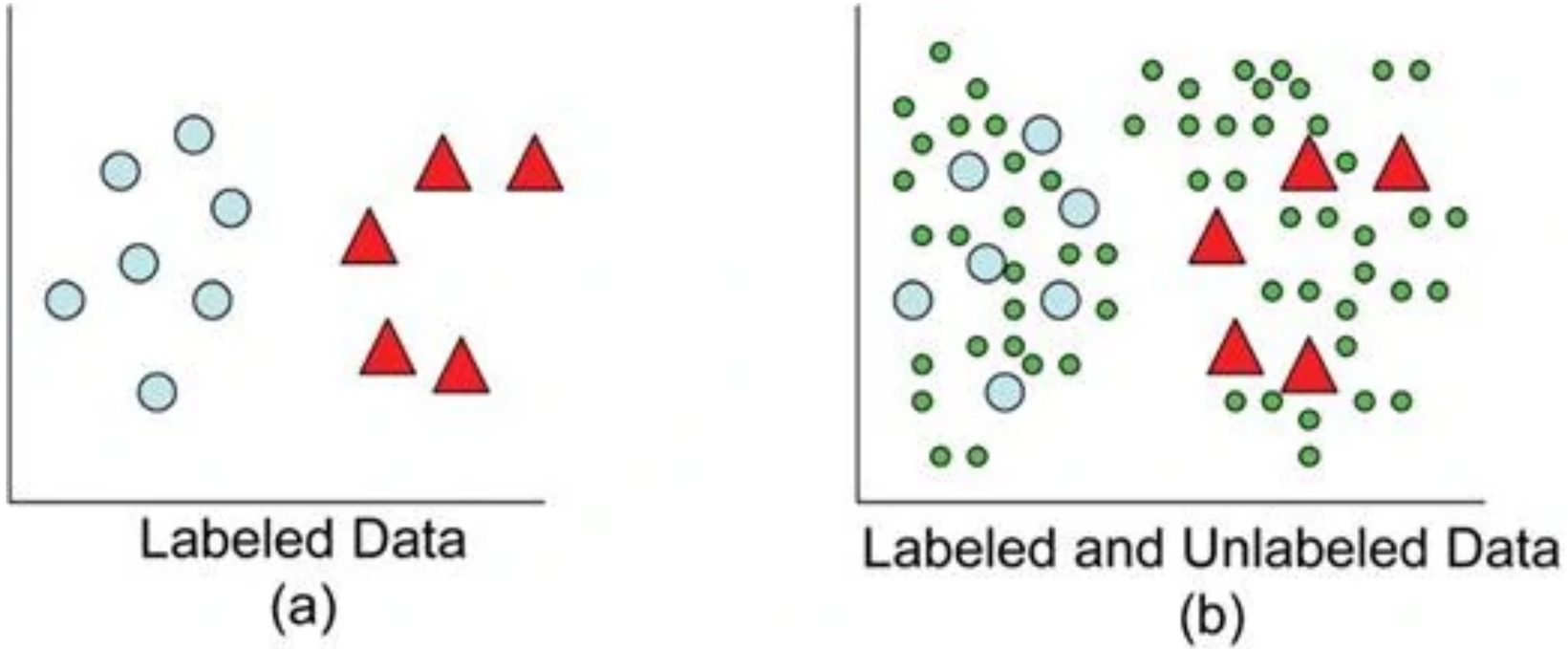


- many target domain data
- large domain discrepancy

Semi-Supervised Learning: Pseudo-labeling approach

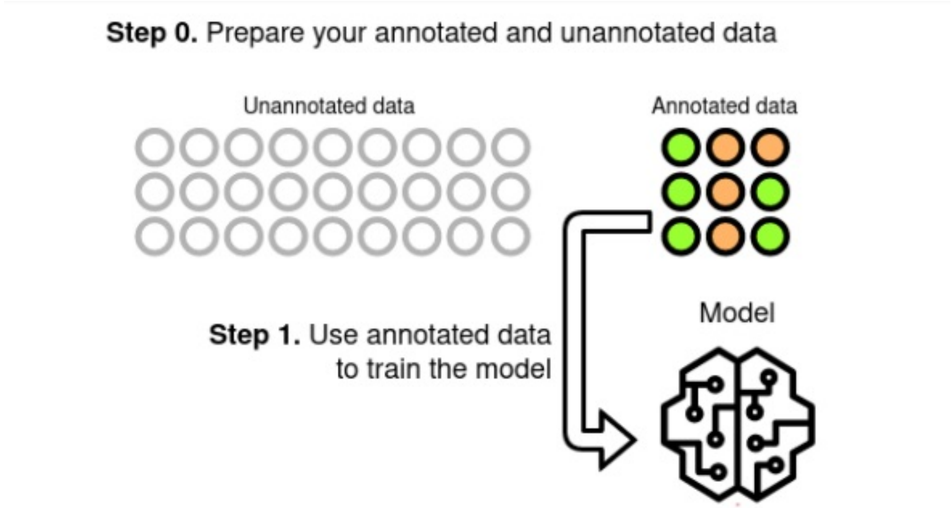
Semi-supervised Learning

- **Setting.** We have both:
  - small number of labeled data
  - large number of unlabeled data
- **Question.** How can we utilize the unlabeled data most effectively?



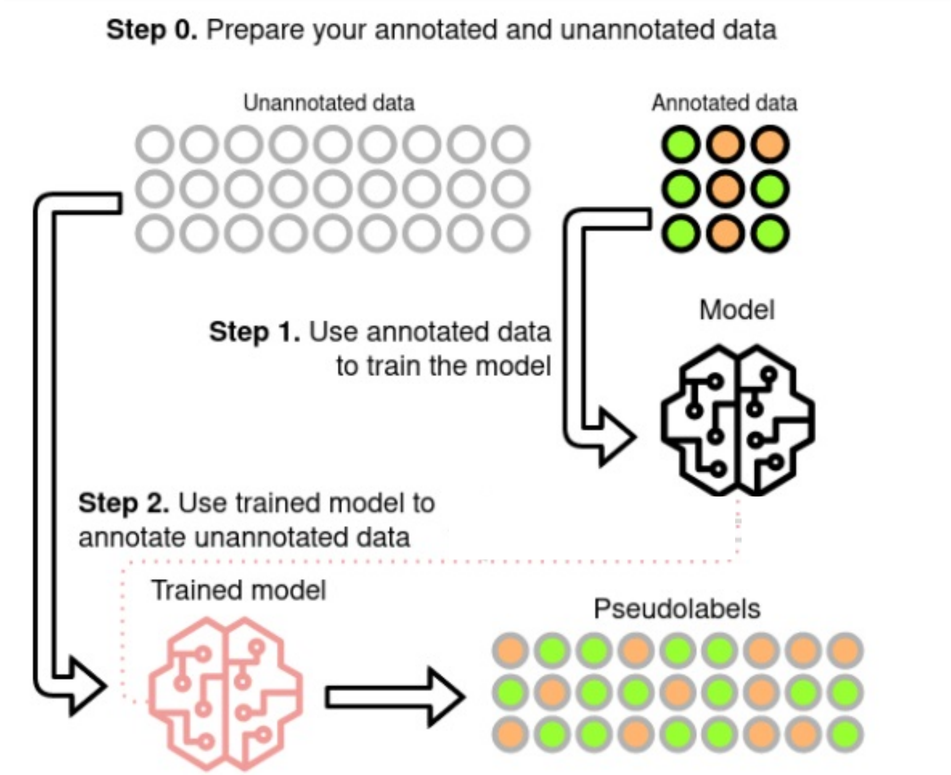
Semi-supervised Learning

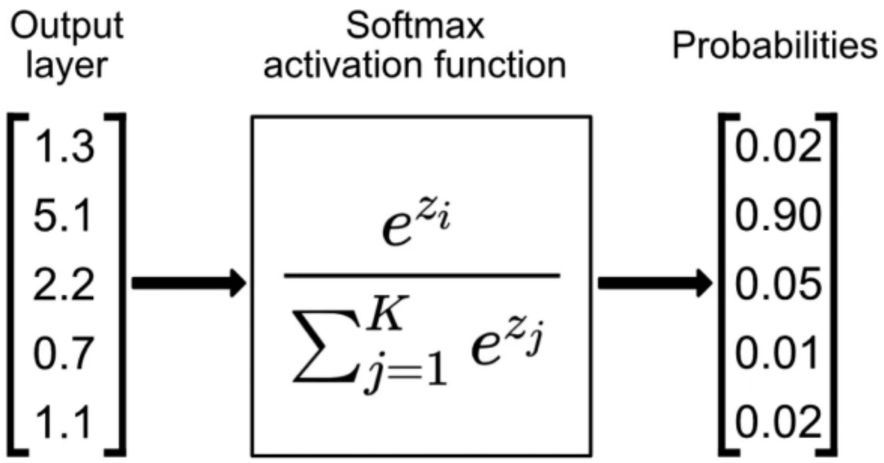
- **Idea.** Give pseudo-labels for the confident samples.
  - Train a model on labeled dataset



Semi-supervised Learning

- **Idea.** Give pseudo-labels for the confident samples.
  - Train a model on labeled dataset
  - Pick some unlabeled samples, whose prediction confidence is large.
    - Give these pseudo-labels



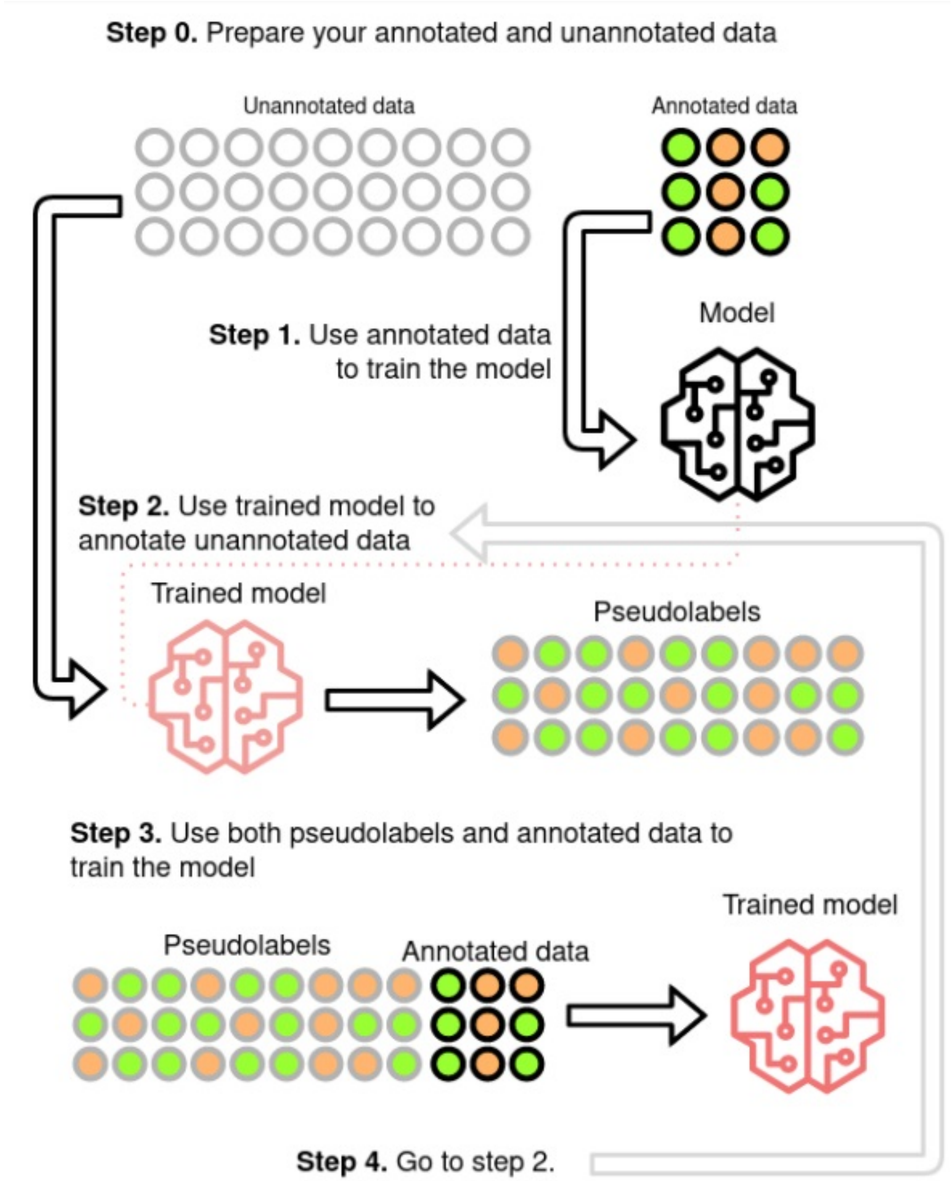


Semi-supervised Learning

- **Idea.** Give pseudo-labels for the confident samples.
  - Train a model on labeled dataset
  - Pick some unlabeled samples, whose prediction confidence is large.
    - Give these pseudo-labels.
  - Add pseudo-labeled samples to the training dataset and repeat

- Examples. MixMatch (2019), FixMatch (2020)

- Disclaimer. Not the only way, but rest are inherited by the self-supervised learning.



Self-Supervised Learning

Self-supervised Learning

- **Setting.** We have a lot of unlabeled data.
  - Goal. Train a feature map (representation function) that can be transferred well
    - Transfer to the same domain: Semi-supervised learning
    - Transfer to the other domain: Transfer learning

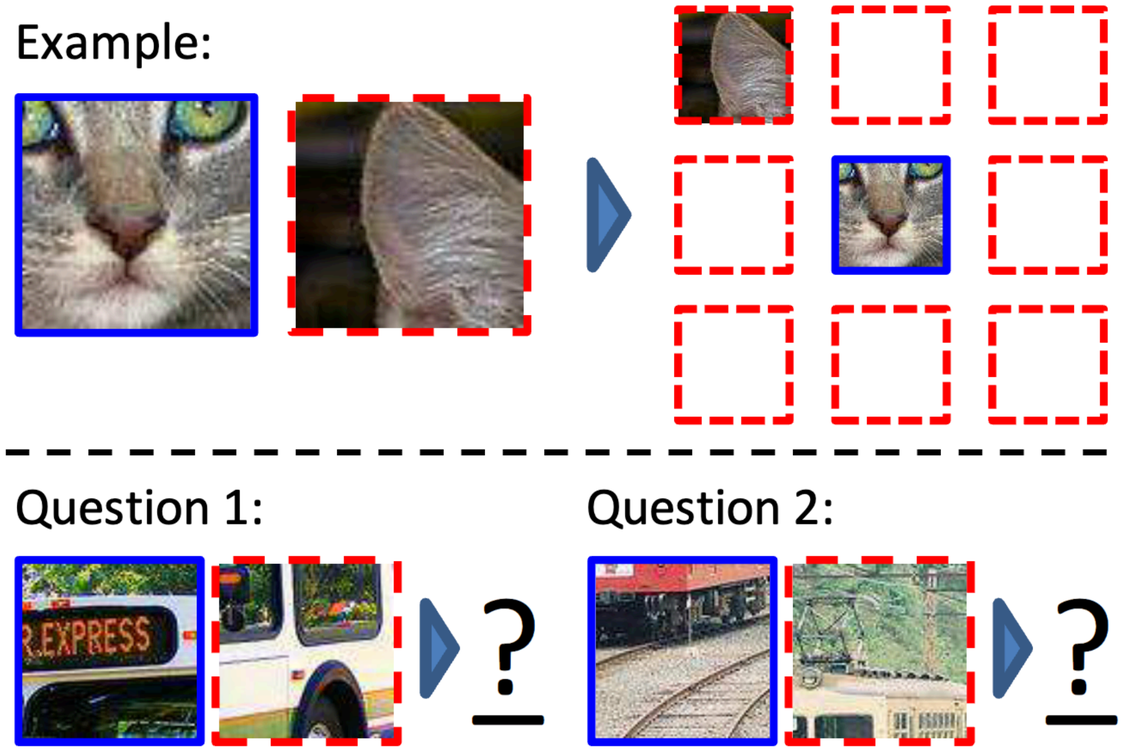
Self-supervised Learning

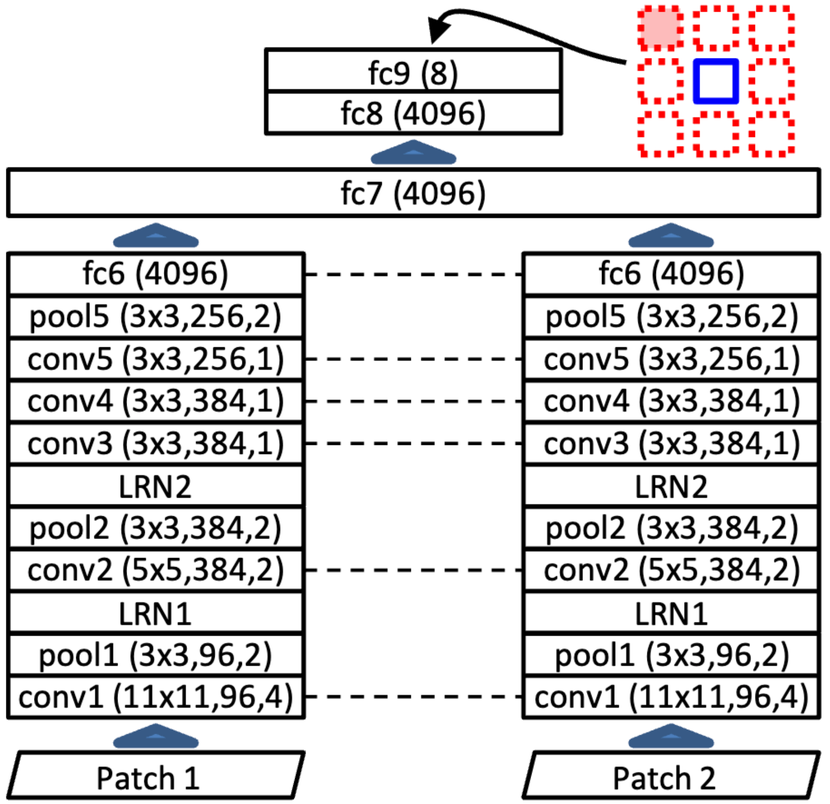
- **Setting.** We have a lot of unlabeled data.
  - Goal. Train a feature map (representation function) that can be transferred well
    - Transfer to the same domain: Semi-supervised learning
    - Transfer to the other domain: Transfer learning
- **Idea.** There are two major approaches:
  - Pretext Task
  - Joint embedding

Doersch et al., “Unsupervised visual representation learning by context prediction” ICCV 2015

Pretext Task

- **Idea.** Train a model to solve certain synthetic (but useful) tasks
- Example. Context prediction (2015)
  - Predict the relative location of the patch w.r.t. another patch

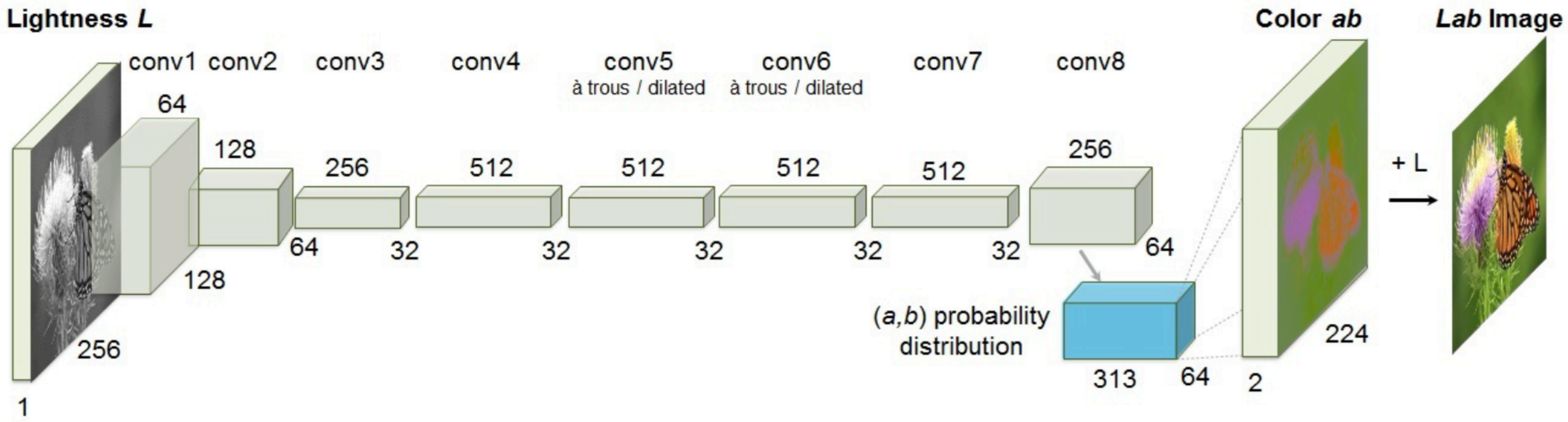




Zhang et al. “Colorful image colorization” ECCV 2016

Pretext Task

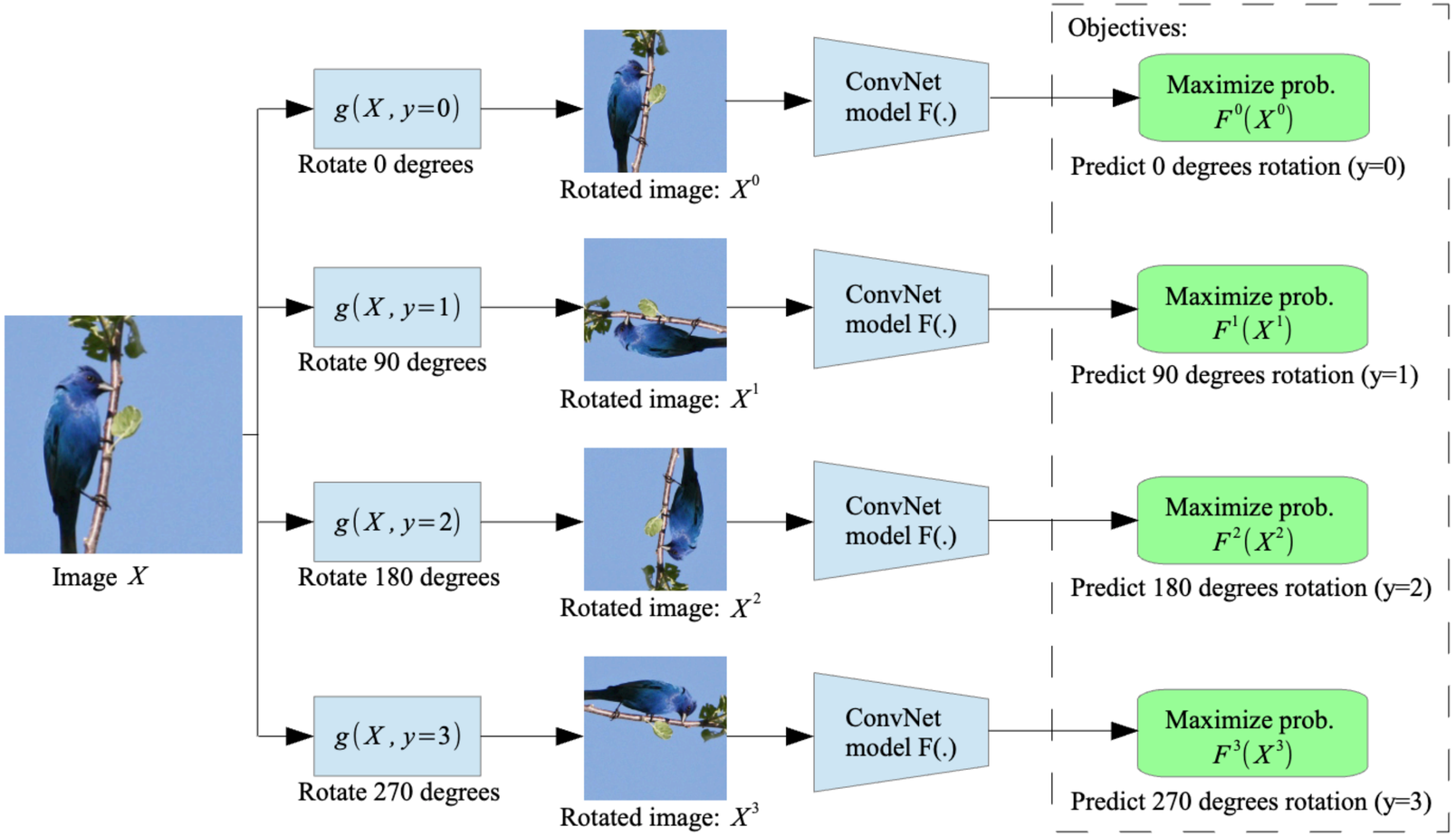
- Example. Colorization (2016)
  - Predict the colors of each pixel in the gray-scaled image.



Gidaris et al. “Unsupervised representation learning by predicting image rotations,” ICLR 2018

Pretext Task

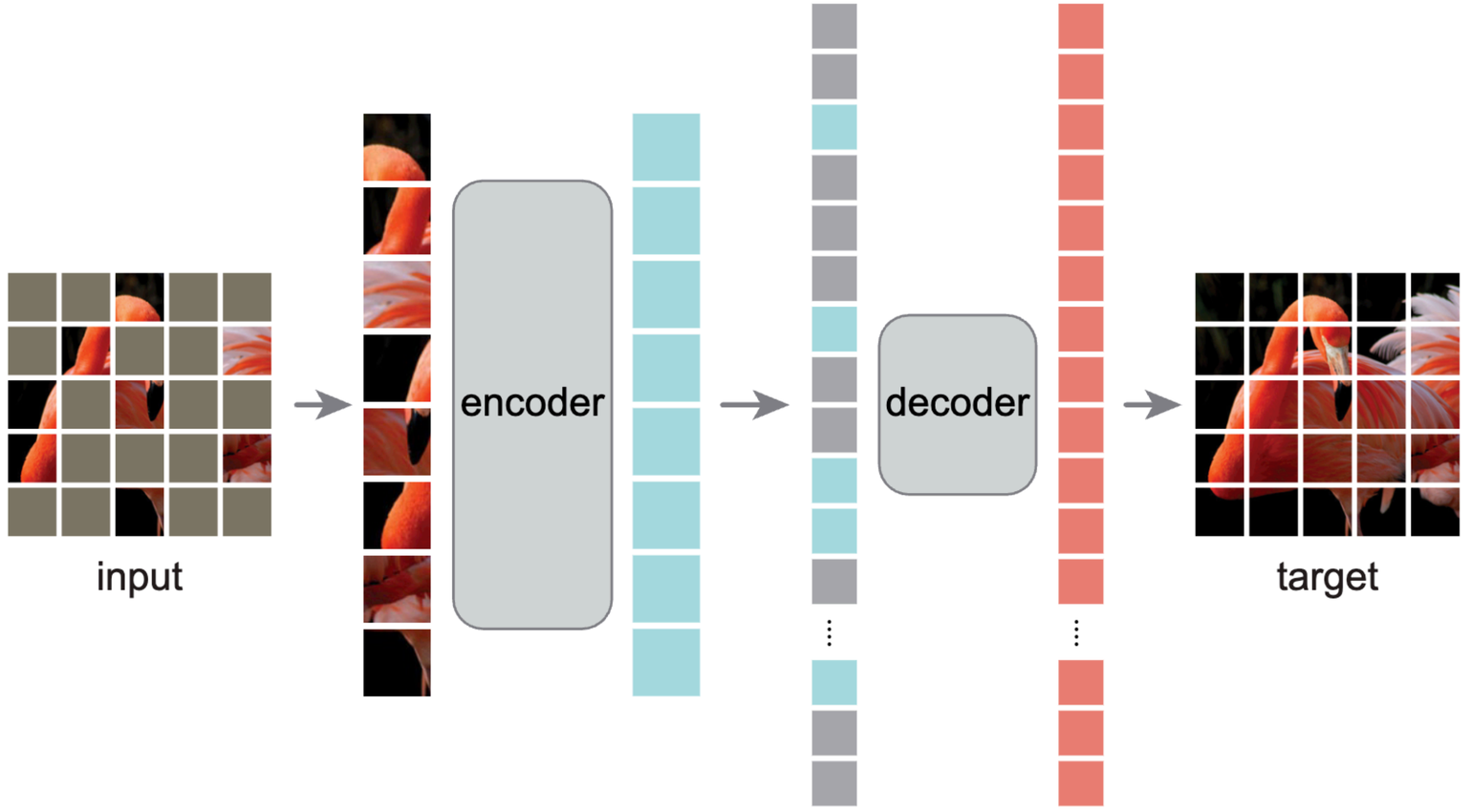
- Example. Rotation (2018)
  - Predict how much the target image has been rotated



He et al. “Masked autoencoders are scalable vision learners,” CVPR 2022

Pretext Task

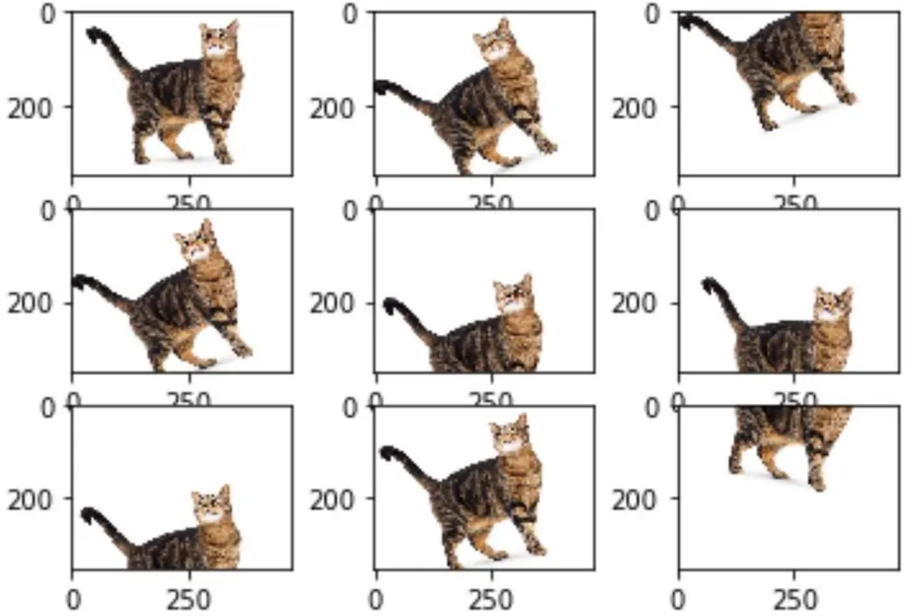
- Example. Masking (2022)
  - Complete the masked-out details from the given image
    (will be revisited later; requires knowing transformers)



Joint embedding

- **Idea.** Train a model to be indifferent to certain operations
  - The representation of an image should be similar to a slightly perturbed image (positive sample)
  - Problem. This becomes a minimization problem with trivial solution:

$$\min_{\theta} \mathbb{E} \|f_\theta(\mathbf{x}) - f_\theta(\mathbf{x}_{\text{perturbed}})\|^2$$

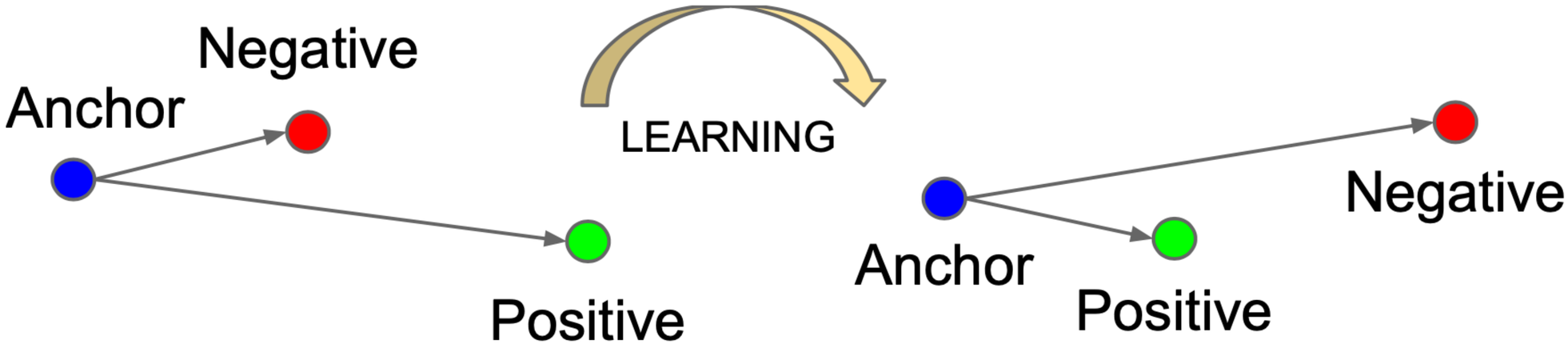


Joint embedding

- **Solution.** Add a negative sample, from which we want to maximize the distance.
  - The optimization problem may now be something like:

$$\min_{\theta} \left(\mathbb{E} \|f_\theta(\mathbf{x}) - f_\theta(\mathbf{x}_{\text{perturbed}})\|^2 - \mathbb{E} \|f_\theta(\mathbf{x}) - f_\theta(\mathbf{y})\|^2 \right)$$

    where $\mathbf{y}$ is an independently drawn image (extremely less likely to be a cat).
    - Called “triplet loss”



Chen et al., "A Simple Framework for Contrastive Learning of Visual Representations" ICML 2020

Joint embedding

- Example. Simple Contrastive Learning (2020)
  - Draw a large batch of data: $\mathbf{x}_1, \ldots, \mathbf{x}_N$
  - Randomly augment each sample twice, and get their representations: $\mathbf{x}_i \mapsto \mathbf{z}_{2k}, \mathbf{z}_{2k-1}$
  - Use the cross-entropy-like loss function

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

    where $\mathbf{sim}(\mathbf{u}, \mathbf{v}) = \mathbf{u}^\top \mathbf{v} / \|\mathbf{u}\| \|\mathbf{v}\|$ is the cosine similarity.

  - Total loss:

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

- Other examples. BYOL, MoCo (2020)

Self-supervised learning

- **As of 2024.** Masked Autoencoders are slightly more preferred over joint embedding approaches
    - Contrastive learning require large batch & large memory
    - Contrastive learning utilizes human-derived concepts more severely
        - Heavily depends on which augmentation we use

- **Next class.** Visual Generative Models

Cheers