
بسم الله الرحمن الرحیم

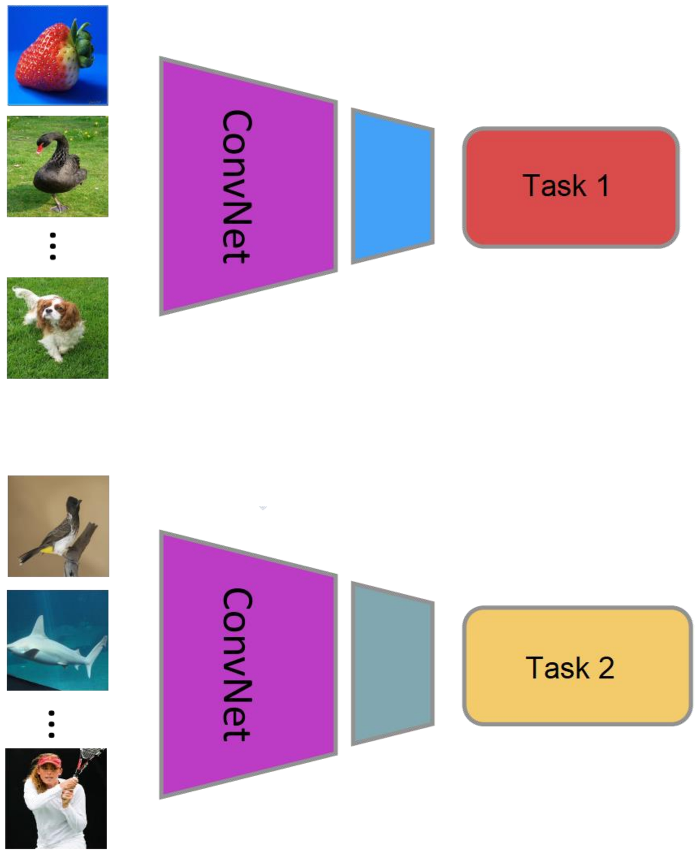
Deep Learning

Mohammad Reza Mohammadi

2021

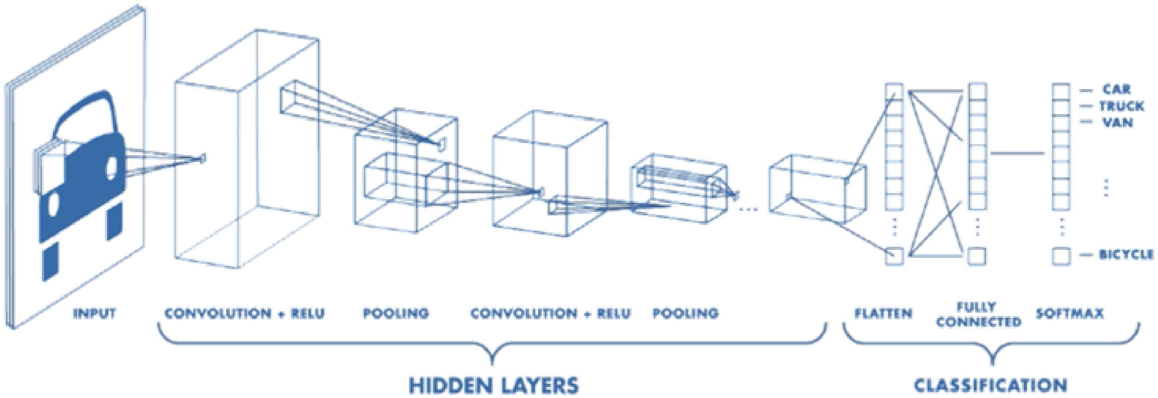
Representation learning

- How learning algorithms share statistical strength across different tasks?
  - Including using information from unsupervised tasks to perform supervised tasks
  - Transfer learned knowledge to tasks for which few or no examples are given but a task representation exists
  - Shared representations are useful to handle multiple modalities or domains



Representation learning

- What makes one representation better than another?
  - A good representation is one that makes a subsequent learning task easier
- We can think of feedforward networks trained by supervised learning as performing a kind of representation learning
  - The last layer of the network is typically a linear classifier
  - The rest of the network learns to provide a representation to this classifier



Representation learning

- Supervised training does not involve explicitly imposing any condition on the learned intermediate features
  - Orthogonal CNNs impose orthogonality on convolutional filters
- We often have very large amounts of unlabeled training data and relatively little labeled training data
  - Semi-Supervised and Self-Supervised Learning
  - Humans and animals are able to learn from very few labeled examples

Wang, Jiayun, et al. "Orthogonal convolutional neural networks." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.

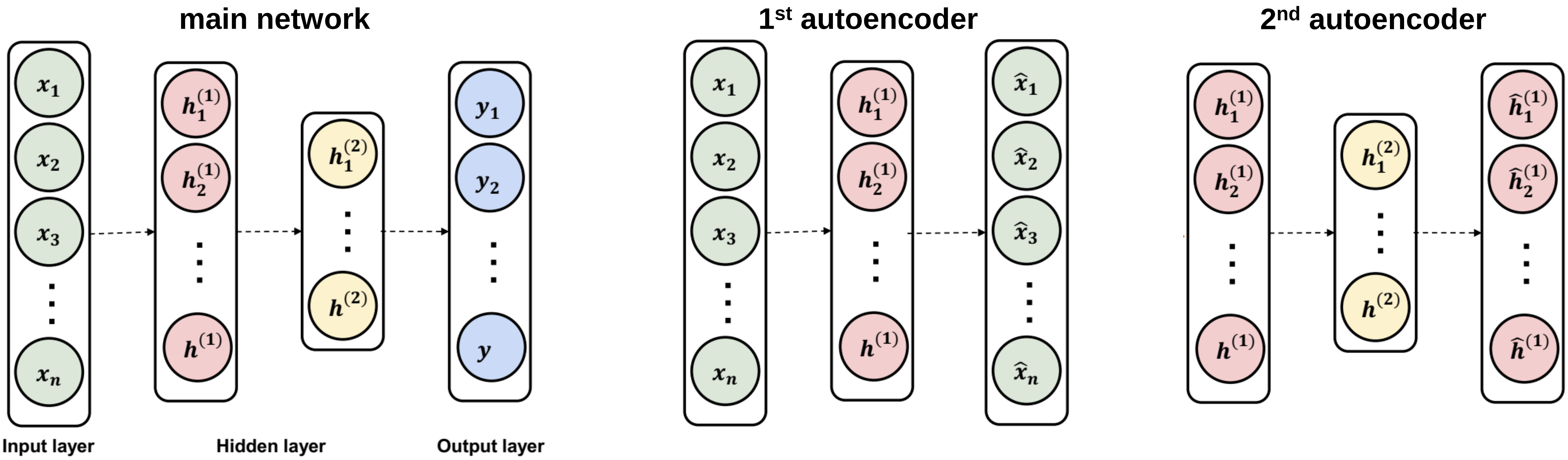
Greedy layer-wise unsupervised pretraining

- Weight initialization is an important design choice when developing deep learning neural network models
- Greedy layer-wise UP have long been used to sidestep the difficulty of jointly training the layers of a deep neural net for a supervised task

- Greedy: UL per a single layer basis, it progresses by layer
- Pretraining: It runs only once before joint training

Greedy layer-wise unsupervised pretraining

- Pretraining proceeds one layer at a time
- It is then possible to perform supervised fine-tuning

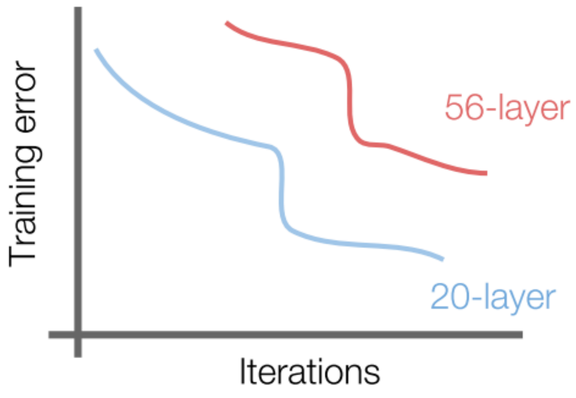


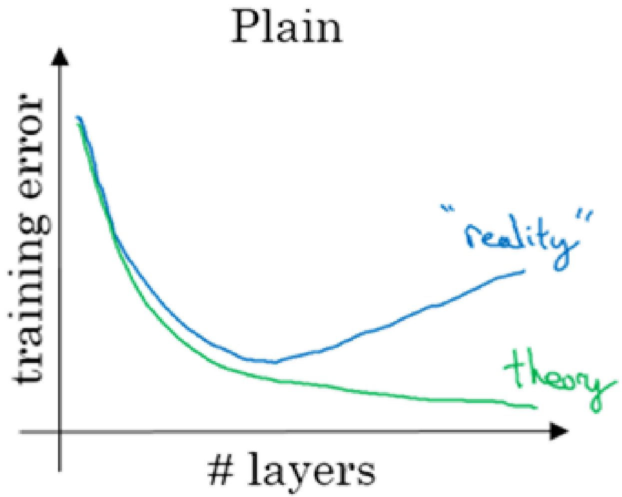
Greedy layer-wise unsupervised pretraining

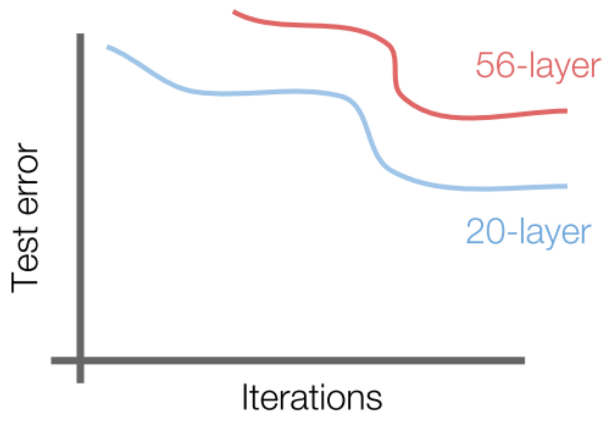
- This approach was performed before the invention and popularization of modern techniques for training very deep networks (ReLU, batch normalization, better optimizers, better architectures, …)
- Modern approaches typically use simultaneous unsupervised learning and supervised learning rather than two sequential stages
- Most helpful when the number of labeled examples is very small

ResNet

- Such degradation is not caused by overfitting, and adding more layers to a suitably deep model leads to higher training error
- A deeper model should produce no higher training error than its shallower counterpart
- The degradation (of training accuracy) indicates that not all systems are similarly easy to optimize



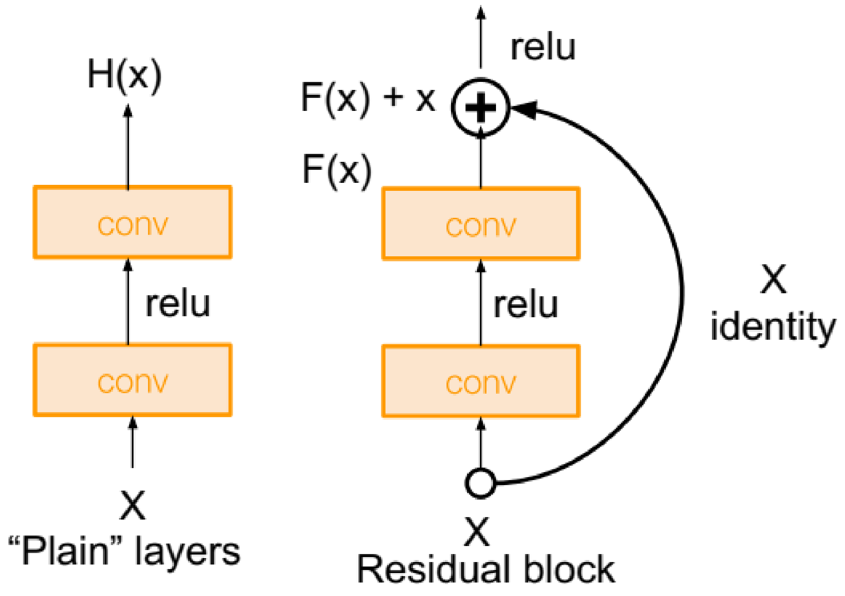




K. He, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

ResNet

- Let these layers fit a residual mapping $F(x) = H(x) - x$
- We hypothesize that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping
- To the extreme, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers



Batch normalization (BN)

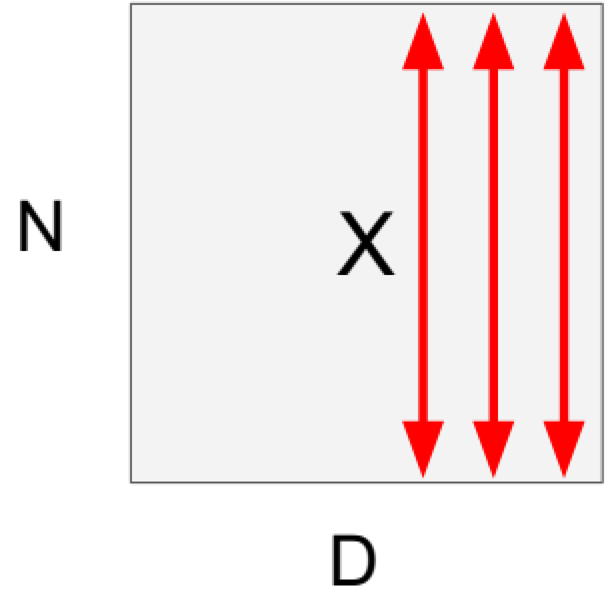
- Dramatic effect on optimization performance
- Especially for convolutional networks and networks with sigmoidal nonlinearities
- Consider a batch of activations at some layer
  - To make each dimension zero-mean unit-variance, apply:
    - This is a vanilla differentiable function

$$\hat{x}^{(k)} = \frac{x^{(k)} - E\left[x^{(k)}\right]}{\sqrt{Var\left[x^{(k)}\right]}}$$

S. Ioffe and C. Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *International conference on machine learning*. 2015.

Batch normalization

- Compute the empirical mean and variance independently for each dimension
- Input: $x: N \times D$
- Per-channel mean, shape is D: $$\mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{ij}$$
- Per-channel var, shape is D: $$\sigma_j^2 = \frac{1}{N}\sum_{i=1}^{N}\left(x_{ij} - \mu_j\right)^2$$
- Normalized x, shape is $N \times D$ $$\hat{x}_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$
- Output, Shape is $N \times D$ $$y_{ij} = \gamma_j \hat{x}_{ij} + \beta_j$$



Learnable scale and shift

Batch normalization: test time

- Estimates depend on minibatch
  - can't do this at test-time!
- (Running) average of values (μ and σ^2) seen during training
- During testing batchnorm becomes a linear operator!
- Can be fused with the previous fully-connected or conv layer

$$\mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{ij}$$

$$\sigma_j^2 = \frac{1}{N}\sum_{i=1}^{N} \left(x_{ij} - \mu_j\right)^2$$

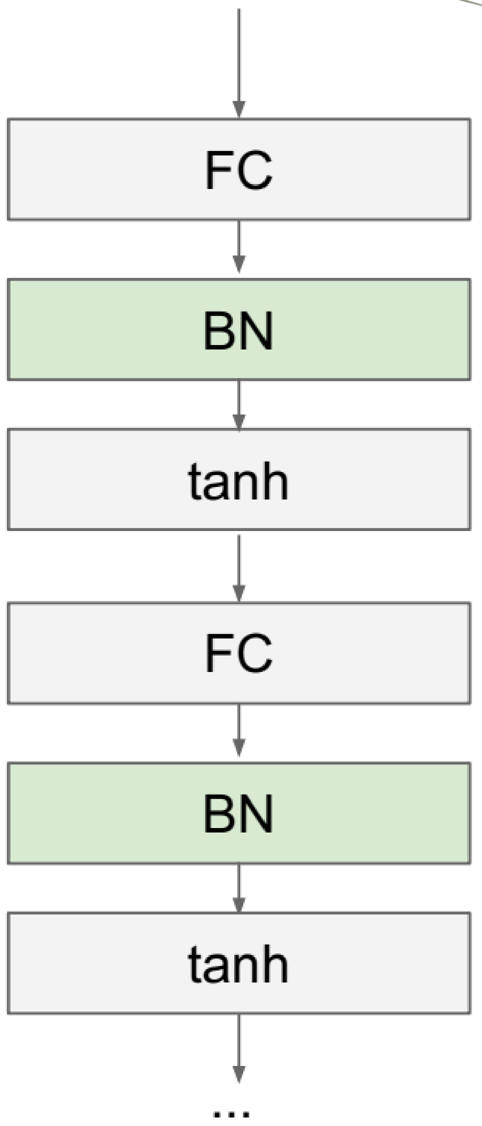
$$\hat{x}_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

$$y_{ij} = \gamma_j \hat{x}_{ij} + \beta_j$$

Batch normalization

- Usually inserted after Fully Connected or Convolutional layers, and before nonlinearity
- Makes deep networks much easier to train!
- Improves gradient flow
- Allows higher learning rates, faster convergence
- Networks become more robust to initialization
- Acts as regularization during training
- Zero overhead at test-time
  - can be fused with conv!

$$\hat{x}^{(k)} = \frac{x^{(k)} - E\left[x^{(k)}\right]}{\sqrt{Var\left[x^{(k)}\right]}}$$



Batch normalization for ConvNets

Batch Normalization for fully-connected layers

$$x:\ N \times D$$

$$\downarrow$$

$$\mu, \sigma:\ 1 \times D$$

$$\gamma, \beta:\ 1 \times D$$

$$y = \gamma \frac{x - \mu}{\sigma} + \beta$$

Batch Normalization for convolutional layers

$$x:\ N \times W \times H \times C$$

$$\downarrow \quad \downarrow \quad \downarrow$$

$$\mu, \sigma:\ 1 \times 1 \times 1 \times C$$

$$\gamma, \beta:\ 1 \times 1 \times 1 \times C$$

$$y = \gamma \frac{x - \mu}{\sigma} + \beta$$

Layer normalization

Batch Normalization for fully-connected layers

$$x\colon\ N \times D$$

$$\downarrow$$

$$\mu, \sigma\colon\ 1 \times D$$

$$\gamma, \beta\colon\ 1 \times D$$

$$y = \gamma \frac{x - \mu}{\sigma} + \beta$$

Layer Normalization for fully-connected layers

$$x\colon\ N \times D$$

$$\downarrow$$

$$\mu, \sigma\colon N \times 1$$

$$\gamma, \beta\colon\ 1 \times D$$

$$y = \gamma \frac{x - \mu}{\sigma} + \beta$$

Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer normalization." *arXiv preprint arXiv:1607.06450* (2016).

Instance normalization

Batch Normalization for convolutional layers

$$x\colon\ N \times W \times H \times C$$

$$\mu, \sigma\colon\ 1 \times 1 \times 1 \times C$$

$$\gamma, \beta\colon\ 1 \times 1 \times 1 \times C$$

$$y = \gamma \frac{x - \mu}{\sigma} + \beta$$

Instance Normalization for convolutional layers

$$x\colon\ N \times W \times H \times C$$

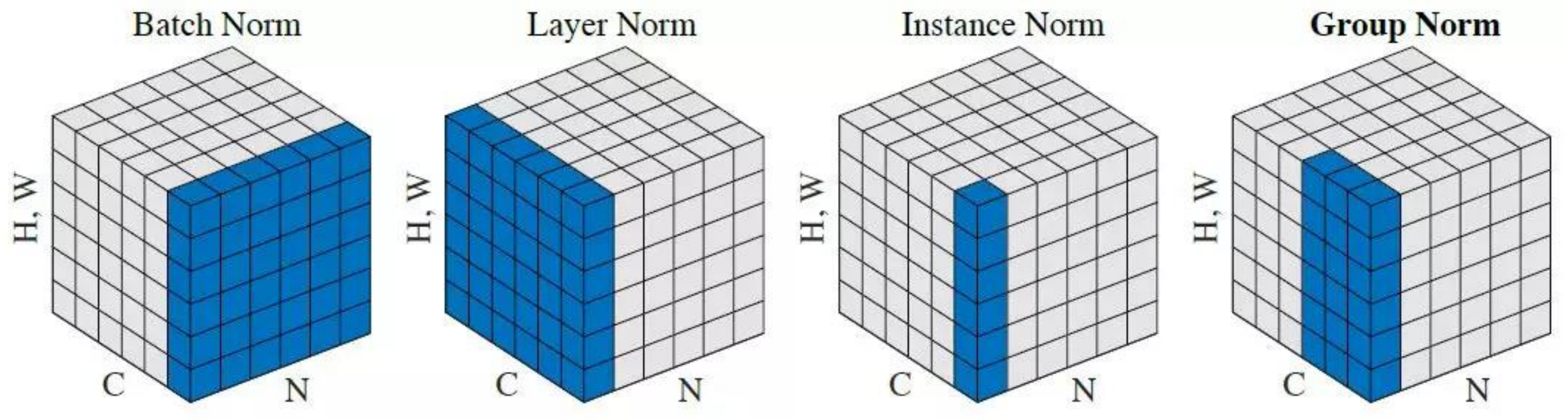
$$\mu, \sigma\colon N \times 1 \times 1 \times C$$

$$\gamma, \beta\colon\ 1 \times 1 \times 1 \times C$$

$$y = \gamma \frac{x - \mu}{\sigma} + \beta$$

D. Ulyanov, A. Vedaldi, and V. Lempitsky. "Instance normalization: The missing ingredient for fast stylization." *arXiv preprint arXiv:1607.08022* (2016).

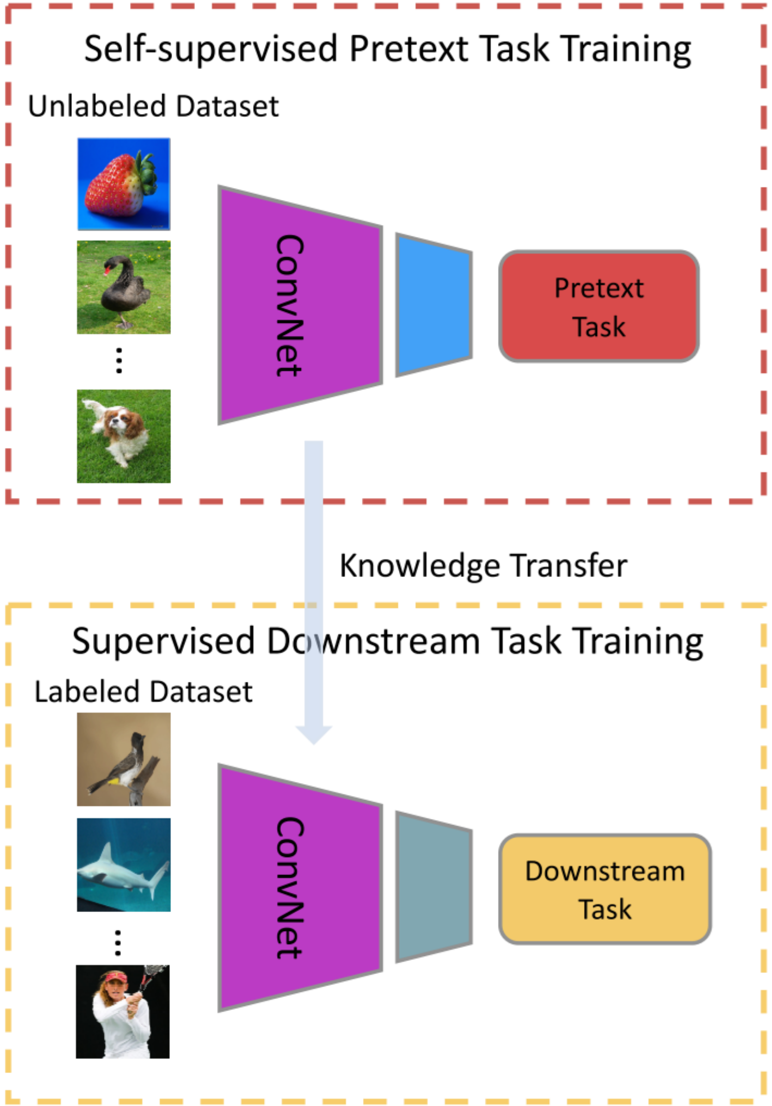
Comparison of normalization layers



Y. Wu and K. He, "Group normalization." Proceedings of the European conference on computer vision (ECCV). 2018.

Self-supervised learning

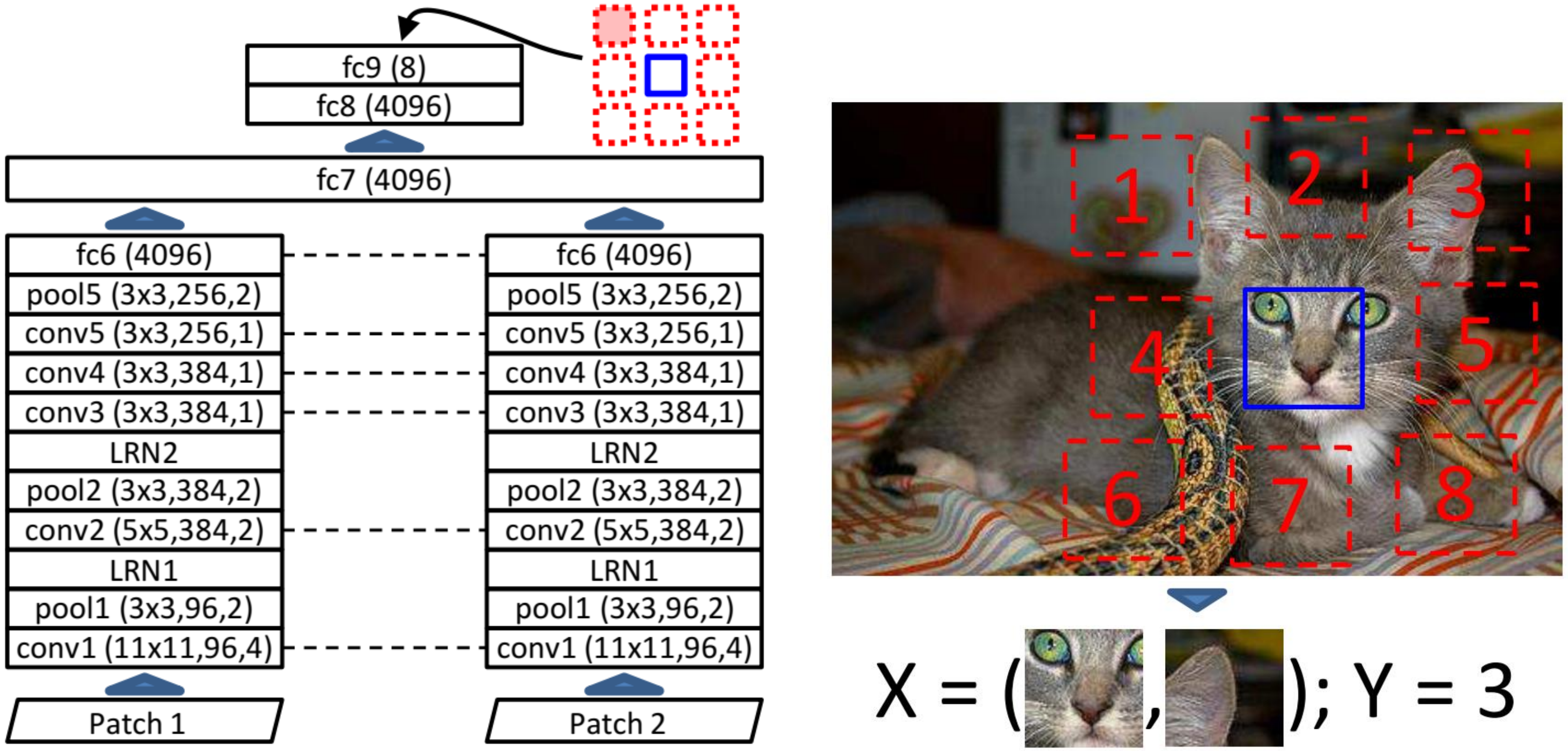
- Self-supervised learning methods are proposed to learn general features from large-scale unlabeled data without using any human-annotated labels
- The pretext tasks share two common properties:
  - Visual features need to be captured by ConvNets to solve the pretext tasks
  - Pseudo labels for the pretext task can be automatically generated based on the attributes of images or videos



L. Jing, and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey." *IEEE Transactions on PAMI* (2020).

Sample pretext: context prediction

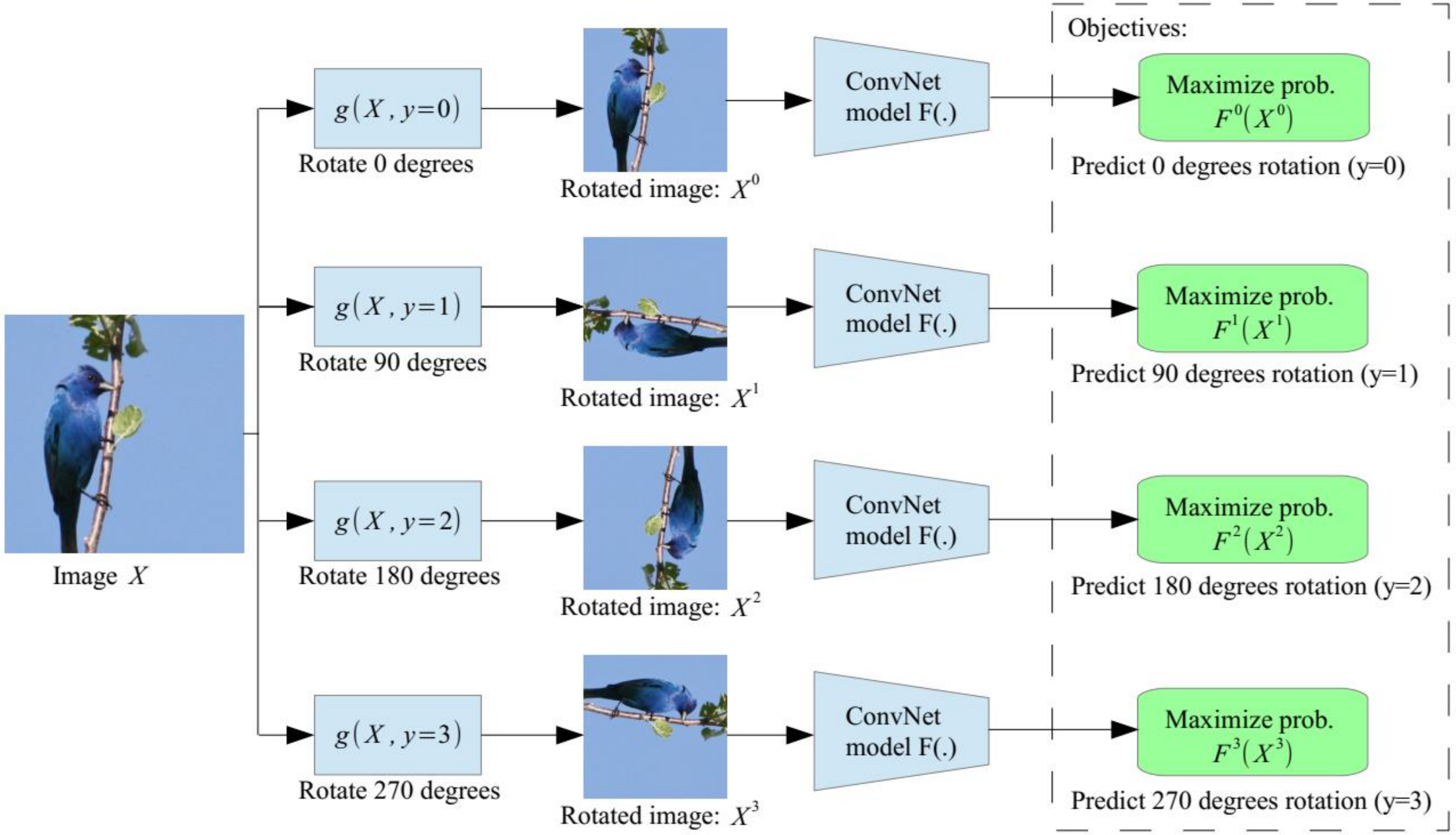
- Spatial context as a source of free and plentiful supervisory signal for training a rich visual representation



C. Doersch, A. Gupta, and A. Efros. "Unsupervised visual representation learning by context prediction." *Proceedings of the IEEE ICCV*. 2015.

Sample pretext: rotation prediction

- Learn image features by training ConvNets to recognize the 2d rotation



S. Gidaris, P. Singh, and N. Komodakis. "Unsupervised representation learning by predicting image rotations." *arXiv preprint arXiv:1803.07728* (2018).