Improving the data efficiency in self-supervised representation learning

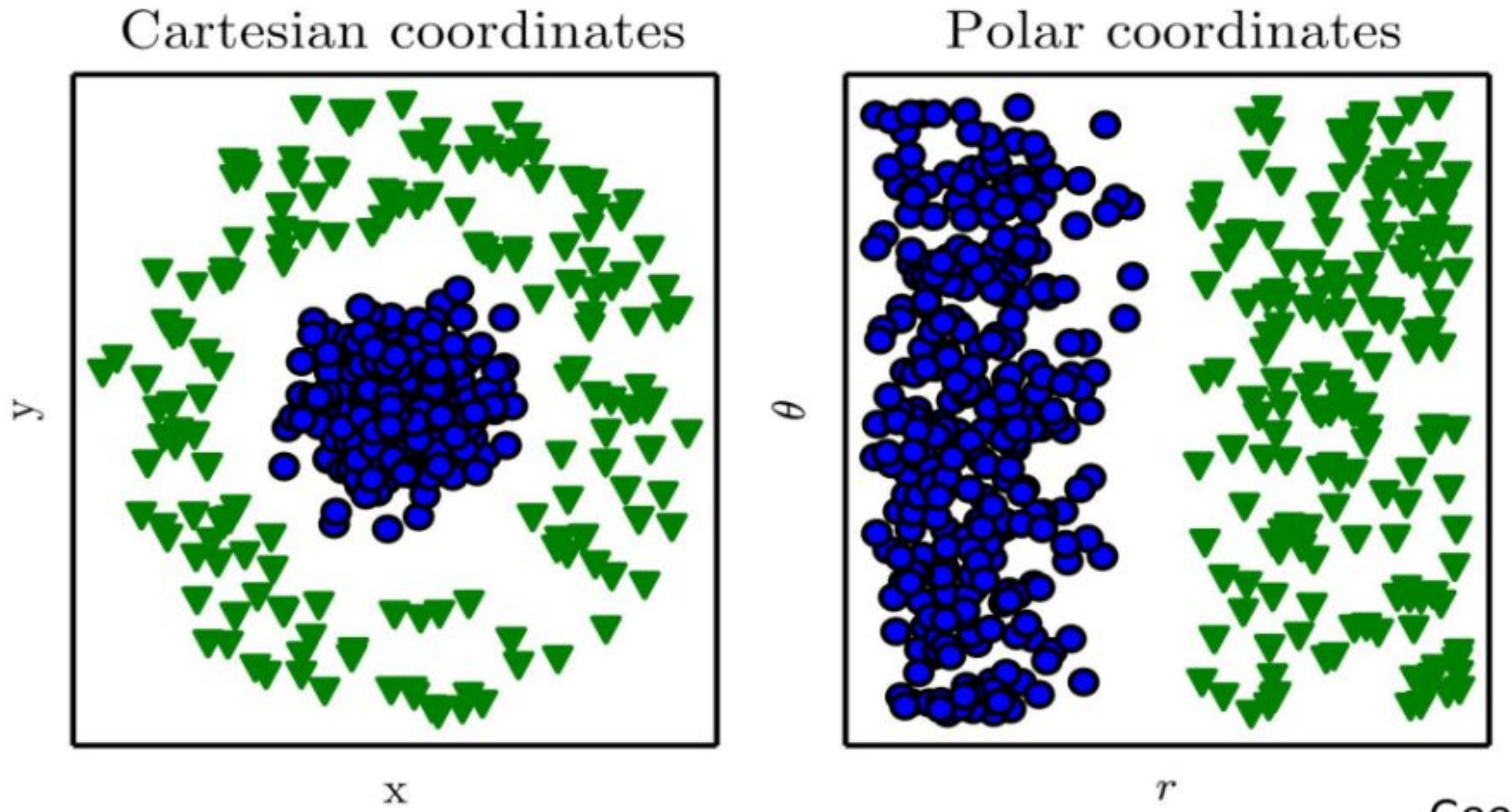
Roberto Halpin Gregorio

May 9, 2022

Learning Representations

- Not all data is numerical but machine learning needs numerical representations!
- For different data types, want to learn meaningful representations that are:
  - Model parsable
  - Efficient / compact
  - Informative for downstream tasks

Making Learning Easier



Goodfellow

Unsupervised Representation Learning

- Given a set of unlabeled data, can we learn about the structure of said data?
  - Clustering
  - Data compression / Dimensionality reduction

- Self-supervised learning
  - Branch of unsupervised learning that uses a created **pretext task** to learn representations of the data

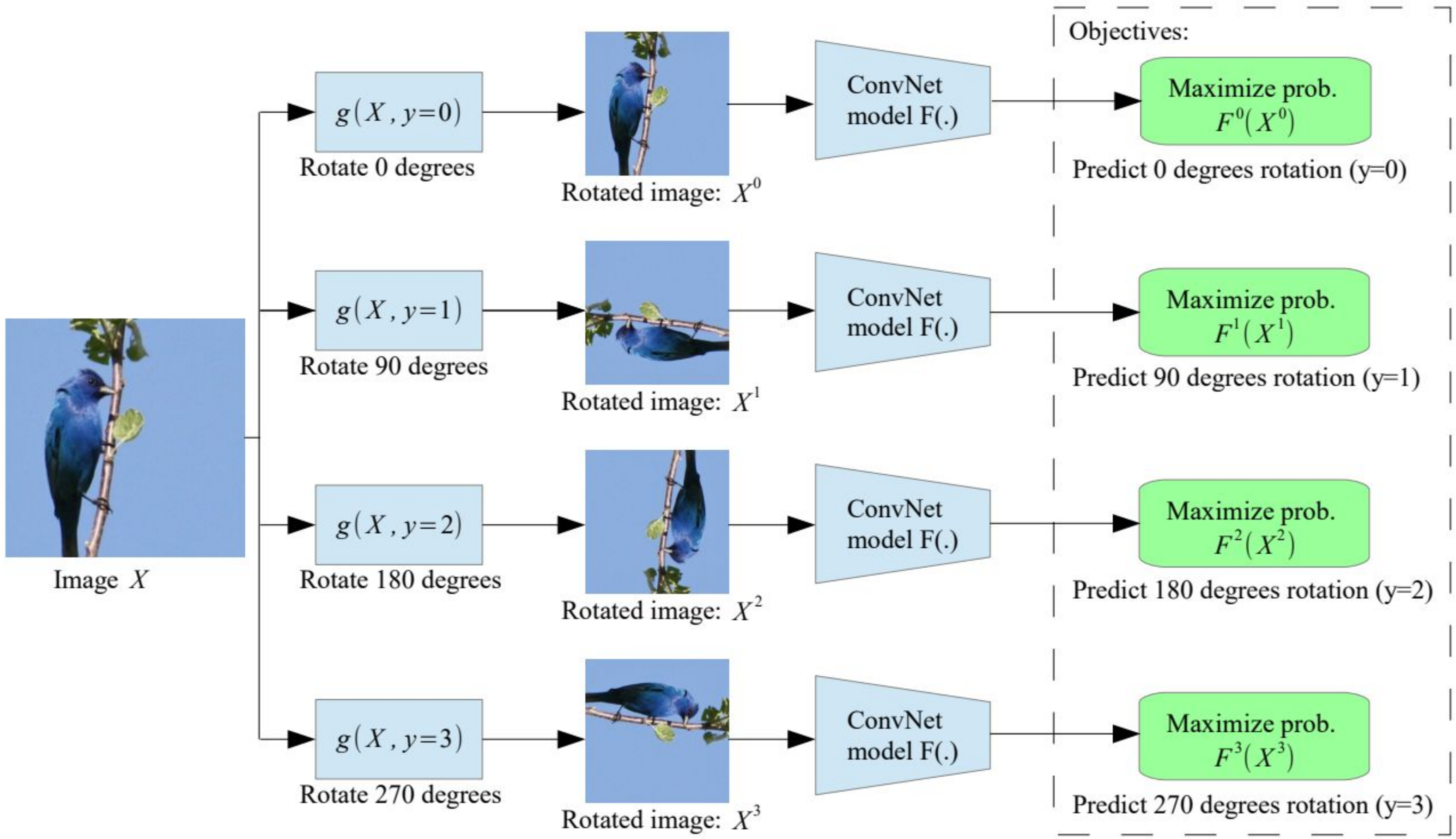
Self-supervised Learning (SSL)

- **Pretext Task**
  - Pre-designed tasks for networks to solve.
  - Learning the objective function produces useful features.

- **Downstream Task**
  - Applications of interest where the pretrained model can be utilized.
  - Greatly benefit from the pretrained models when training data are scarce.

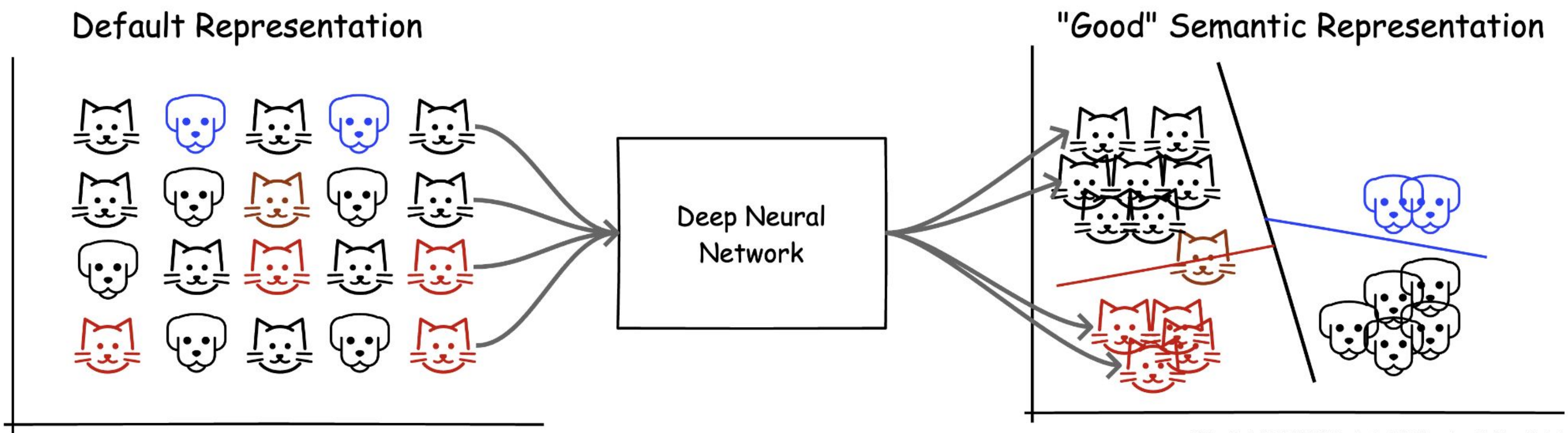
Self-supervised Learning (SSL)

- **RotNet** (Gidaris et al. 2018)



Why self-supervised learning in vision?

- Images are in a continuous, high-dimensional space.
- No need for labeled data.
- Longtail problem.
  - Most labeled images correspond to very few label classes.



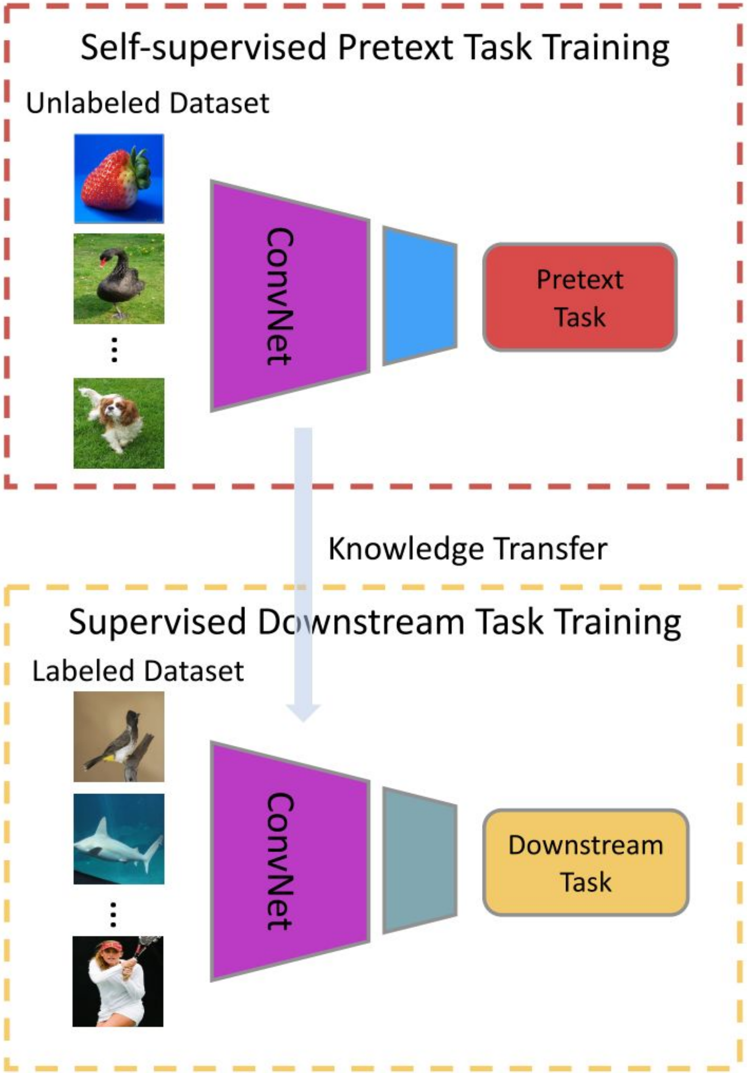
Cat by Martin LEBRETON, Dog by Serhii Smirnov from the Noun Project

Victor Dibia

Self-supervised Learning in Computer Vision

Common workflow

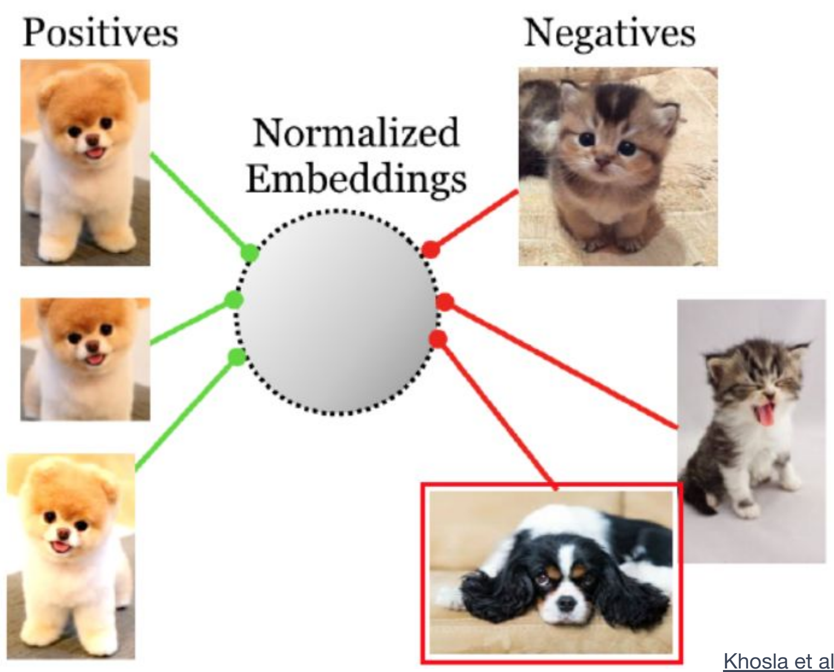
1. Pretext task used to train model.
   a. Unlabeled images.
2. Extract representation network.
3. Representation network used for downstream tasks.



Self-supervised Visual Feature Learning with Deep Neural Networks: A Survey

Self-supervised Learning in Computer Vision

- <u>Contrastive Learning</u>
  - Learn an embedding space where
    - Similar (positive) sample pairs stay close to each other.
    - Dissimilar (negative) sample pairs are far apart.



Khosla et al. 2004

Self-supervised Learning in Computer Vision

- Contrastive Learning
  - Learn an embedding space where
    - Similar (positive) sample pairs stay close to each other.
    - Dissimilar (negative) sample pairs are far apart.
  - How to create positive and negative sample pairs?
  - How do we create the embedding space with these desired properties?

Self-supervised Learning in Computer Vision

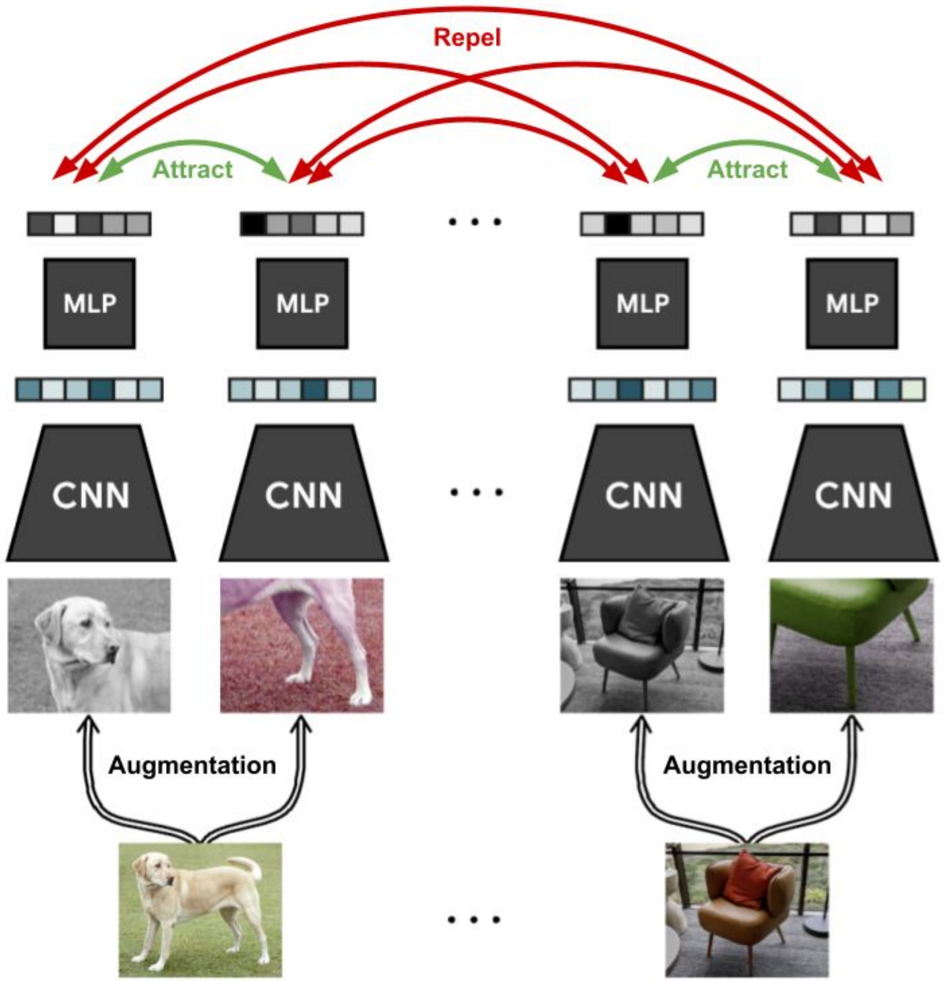
- **SimCLR** (Chen et al. 2020)

A Simple Framework for Contrastive Learning of Visual Representations

Ting Chen [1] Simon Kornblith [1] Mohammad Norouzi [1] Geoffrey Hinton [1]

Self-supervised Learning in Computer Vision

- **SimCLR** (Chen et al. 2020)



Limitations of SSL

- How to identify the important invariances and symmetries?

Limitations of SSL

- How to identify the important invariances and symmetries?
- How do we learn representations that follow these properties?

Limitations of SSL

- How to identify the important invariances and symmetries?
- How do we learn representations that follow these properties?
- Requires a large amount of pre-training iterations (time inefficient).

Limitations of SSL

- How to identify the important invariances and symmetries?
- How do we learn representations that follow these properties?
- Requires a large amount of pre-training iterations (time inefficient).
- Requires a large amount of data to learn good quality representations (data inefficient).

Limitations of SSL

- How to identify the important invariances and symmetries?
- How do we learn representations that follow these properties?
- Requires a large amount of pre-training iterations (time inefficient).
- Requires a large amount of data to learn good quality representations (data inefficient).

Improving the data efficiency of SSL

- Focus on reducing the amount of real data needed.
- Standard approach is to use data augmentation.
- Most SSL methods use simple random data transformations:
  - Flipping
  - Cropping
  - Color jittering
  - Gaussian blur

Improving the data efficiency of SSL

- **Idea:** Generate fake images using our real training data and use these fake images as data augmentation.

Improving the data efficiency of SSL

- **Idea:** Generate fake images using our real training data and use these fake images as data augmentation.

- **Goal:** Be able to beat the performance on the original, real dataset.

Improving the data efficiency of SSL

- **Idea:** Generate fake images using our real training data and use these fake images as data augmentation.

- **Goal:** Be able to beat the performance on the original, real dataset.

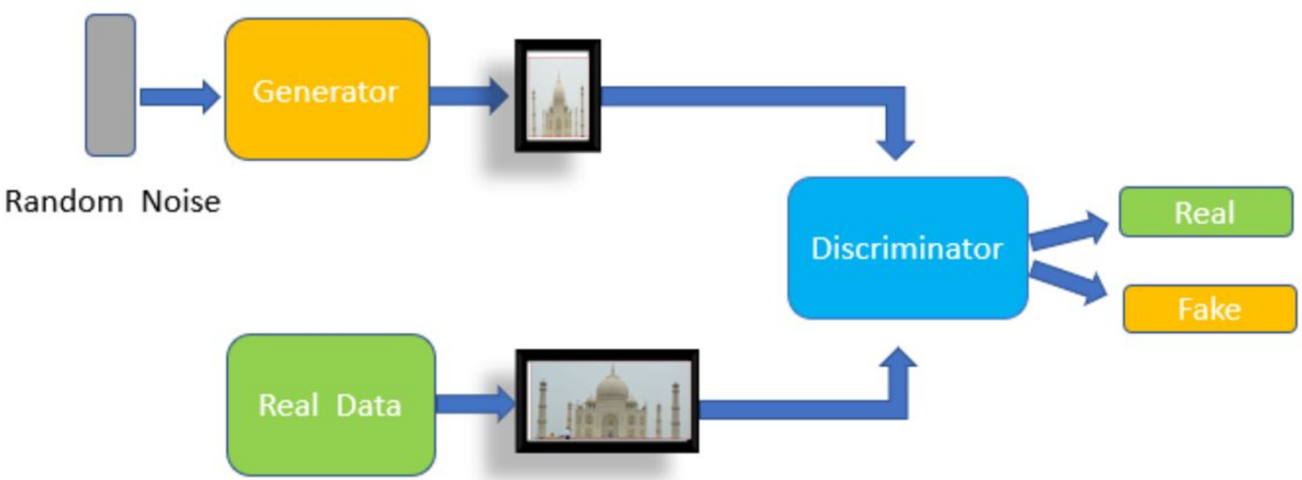
- **Requirement:** A generative model that produces *good* samples when trained with limited data.

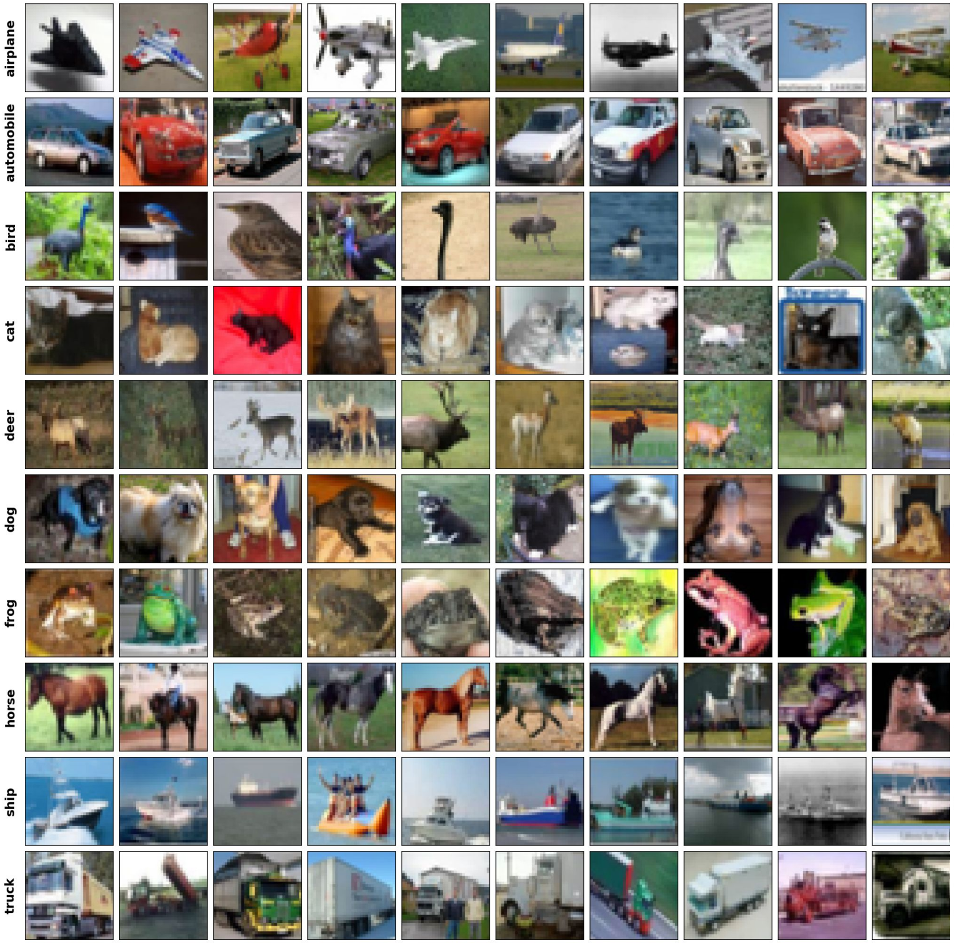
Problem Setup

1. Given an image dataset X without labels.
2. Train a generative model on X to generate a set of fake images Z.
3. Use $X \cup Z$ to pretrain a self-supervised representation learner, such as SimCLR.

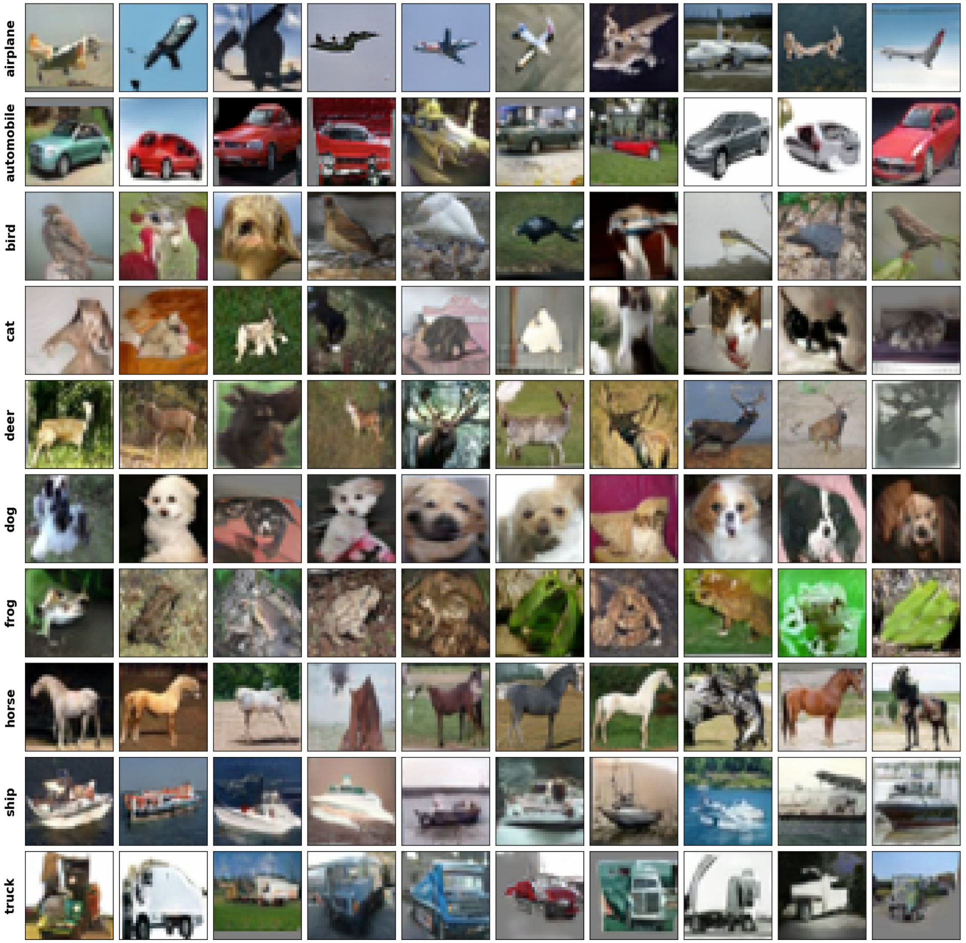
Generating the fake data

- Generative Adversarial Networks (GAN)
- StyleGAN2
  - Unconditional generative image modeling.
  - Know for good image quality.
- Data-efficient GANs
  - Framework that improves GAN training efficiency.





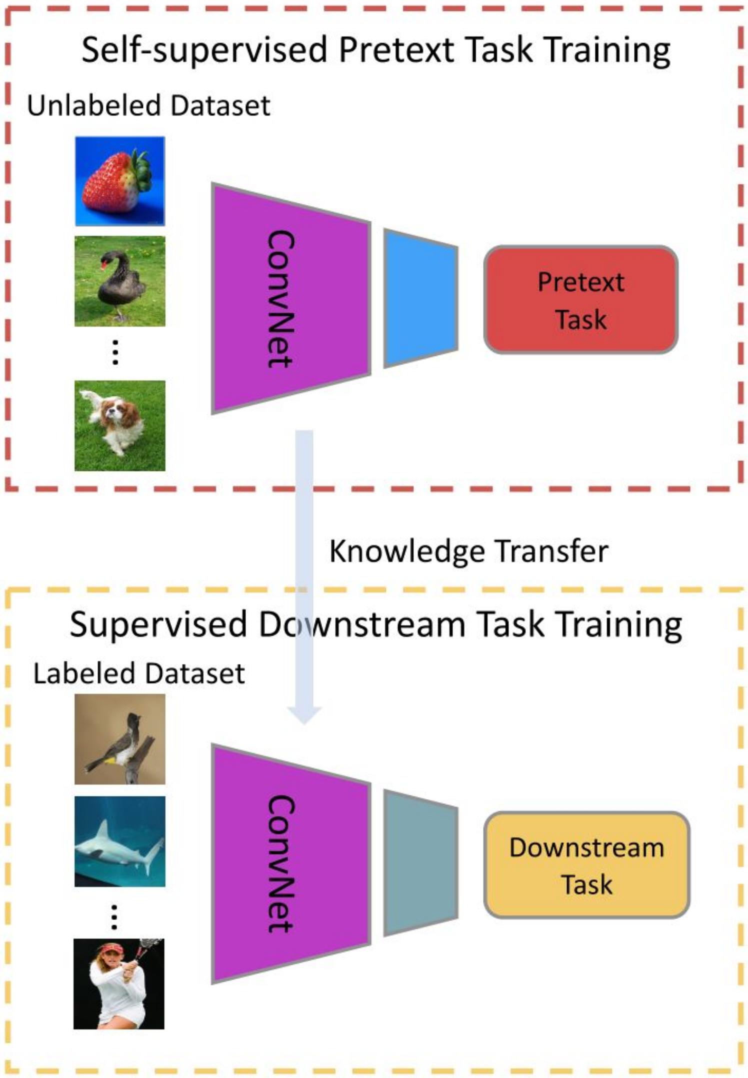
CIFAR-10 Real Images



CIFAR-10 Fake Images

Evaluation

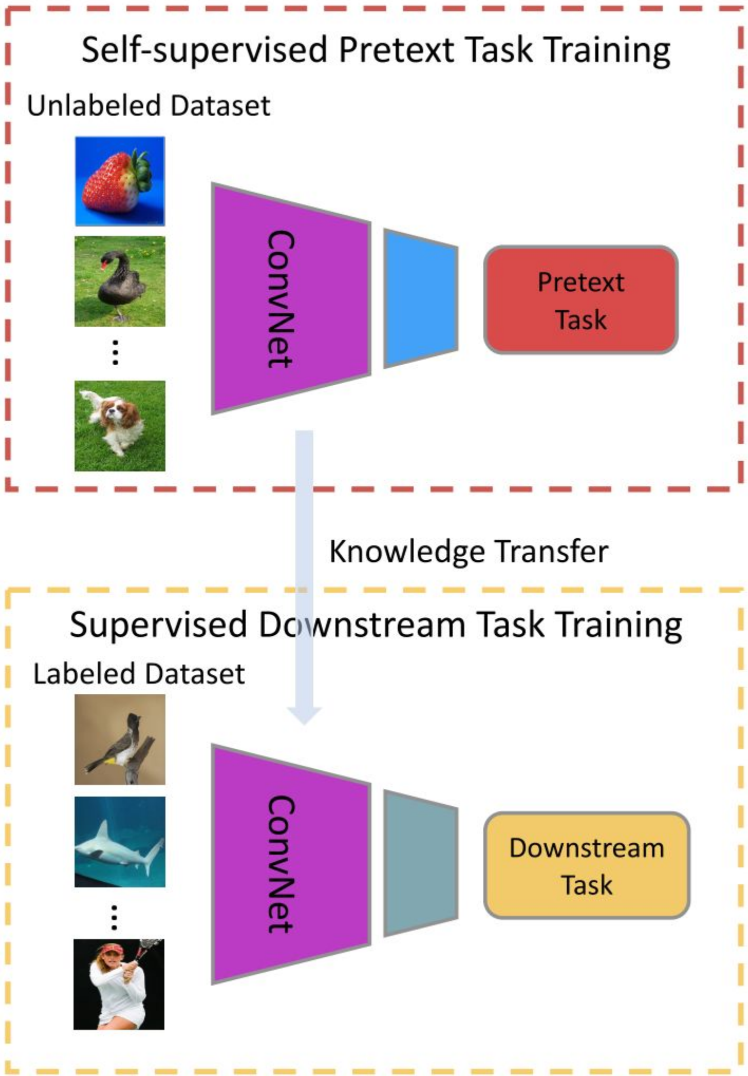
Linear Evaluation Protocol



Evaluation

Linear Evaluation Protocol

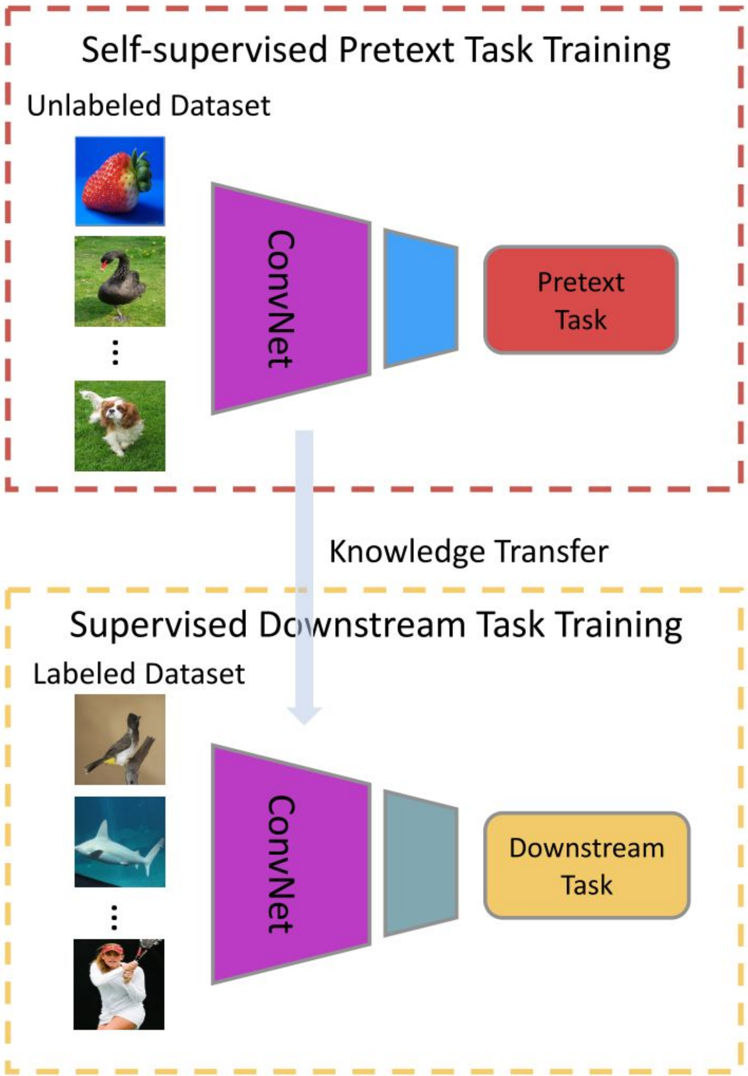
1. Freeze the pretrained representation network.



Evaluation

Linear Evaluation Protocol

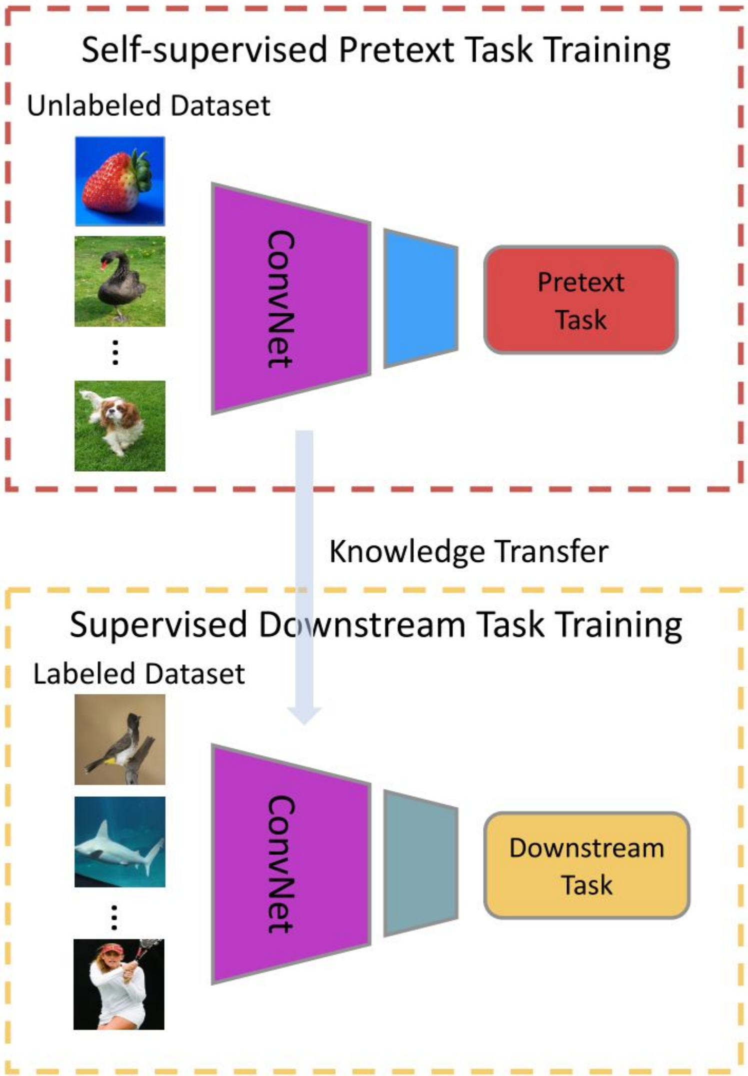
1. Freeze the pretrained representation network.
2. Attach a linear classifier on top of the frozen representation.



Evaluation

Linear Evaluation Protocol

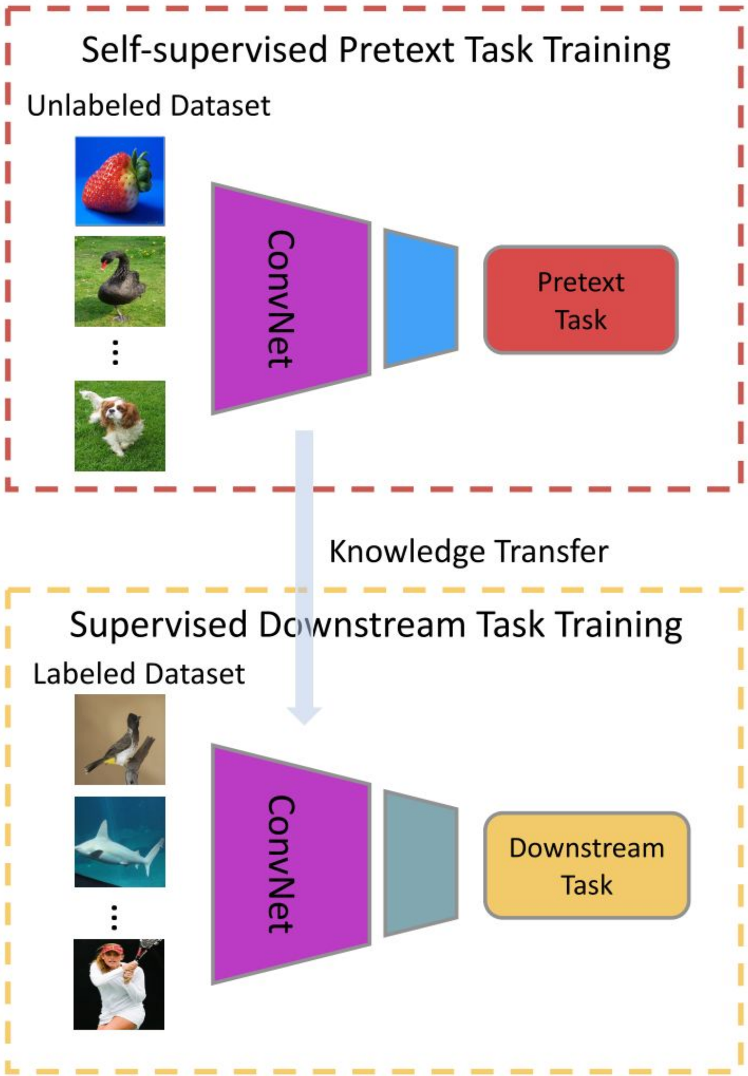
1. Freeze the pretrained representation network.
2. Attach a linear classifier on top of the frozen representation.
3. Train the linear classifier using a labeled train dataset.



Evaluation

Linear Evaluation Protocol

1. Freeze the pretrained representation network.
2. Attach a linear classifier on top of the frozen representation.
3. Train the linear classifier using a labeled train dataset.
4. Evaluate the classifier's accuracy on a test dataset.



Experimental Setup

- Datasets
  - CIFAR-10/100 (+ STL-10 & Tiny ImageNet)
- SSL Algorithm
  - SimCLR (+ MoCo)
- Generative Model
  - Data Efficient StyleGAN2

Experimental Setup

- Datasets
  - CIFAR-10/100 (+ STL-10 & Tiny ImageNet)
- SSL Algorithm
  - SimCLR (+ MoCo)
- Generative Model
  - Data Efficient StyleGAN2
- Data Hyperparameters
  - How many labeled samples (real)?
  - How many generated samples (fake)?

Learning Efficiency

- How quickly does this method train representations that perform well?
- Fix the compute of both this method and the baseline during pretraining, what do we observe in terms of accuracy?
- In the low data regime, there is a possibility that this method also improves the learning efficiency.

Questions?

Thank you!