Stanford University

CS 329P : Practical Machine Learning (2021 Fall)

11. Transfer Learning

Qingqing Huang, Mu Li, Alex Smola

https://c.d2l.ai/stanford-cs329p

Transfer learning

- Motivation
  - Exploit a model trained on one task for a related task
  - Popular in deep learning as DNNs are data hungry and training cost is high
- Approaches
  - Feature extraction (e.g. Word2Vec, ResNet-50 feature, I3D feature)
  - Train a model on a related task and reuse it
  - Fine-tuning from a pertained model (focus of this lecture)
- Related to
  - Semi-supervised learning
  - In the extreme, zero-shot / few-shot learning
  - Multi-task learning, where some labeled data is available for each task

Stanford University

CS 329P : Practical Machine Learning (2021 Fall)

11.1 Fine-tuning in CV

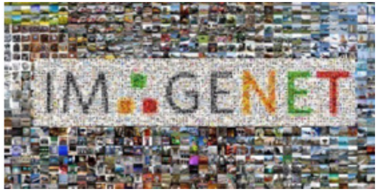
Qingqing Huang, Mu Li, Alex Smola

https://c.d2l.ai/stanford-cs329p

Transferring Knowledge

- There exists large-scale labeled CV datasets
  - Especially for image classification, the cheapest one to label
- Transfer knowledge from models trained on these datasets to your CV applications (with 10-100X smaller data)

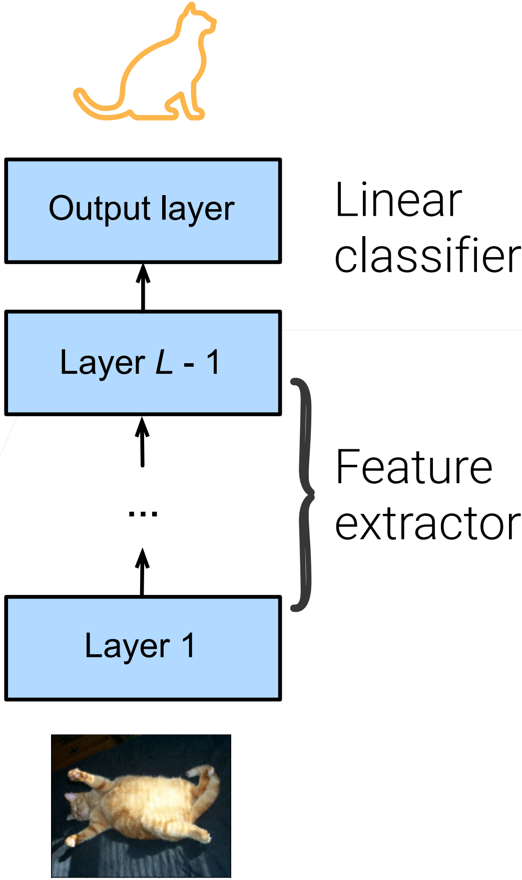
Your dataset



# examples	1.2 M	50K	60 K
# classes	1,000	100	10

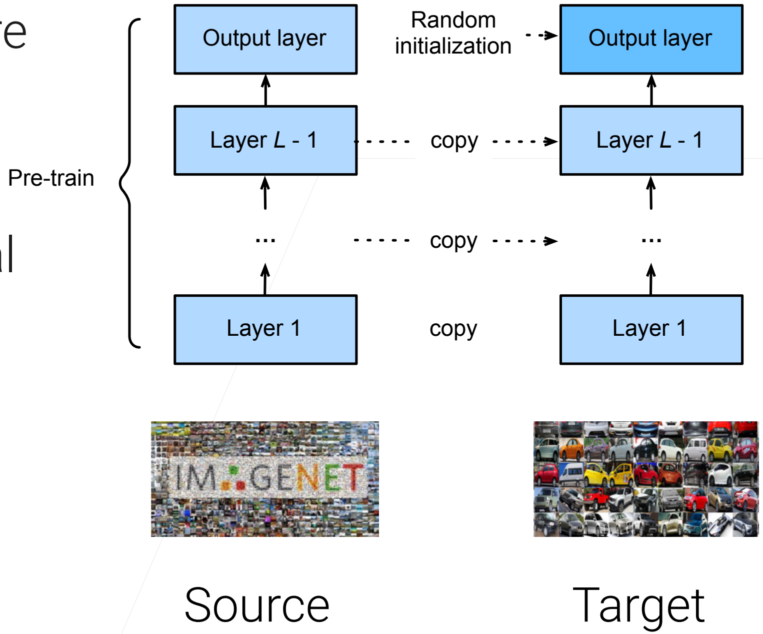
Pre-trained Models

- Partition a neural network into:
  - A feature extractor (encoder) maps raw pixels into linearly separable features
  - A linear classifier (decode) makes decisions
- Pre-trained model
  - a neural network trained on a large-scale and general enough dataset
  - The feature extractor may generalize well to
    - other datasets (e.g. medical/satellite images)
    - other tasks (e.g. object detection, segmentation)



Fine-Tuning techniques

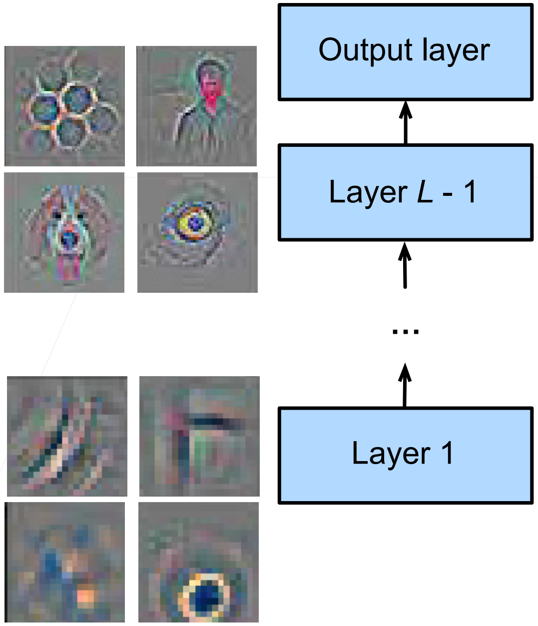
- Initialize the new model:
  - Initialize the feature extractor with the feature extractor parameters of a pre-trained model
  - Randomly initialize the output layer
  - Start the parameter optimization near a local minimal
- Train with a small learning rate with just a few epochs
  - Regularize the search space



Freeze Bottom Layers

- Neural networks learn hierarchical features
  - Low-level features are universal, generalize well, e.g. curves /edges / blobs
  - High-level features are more task and dataset specific, e.g. classification labels
- Freeze bottom layers during fine tuning

  Train the top layers from scratch
  - Keep low-level universal features intact
  - Focus on learning task specific features
  - A strong regularizer



Where to Find Pre-trained Models

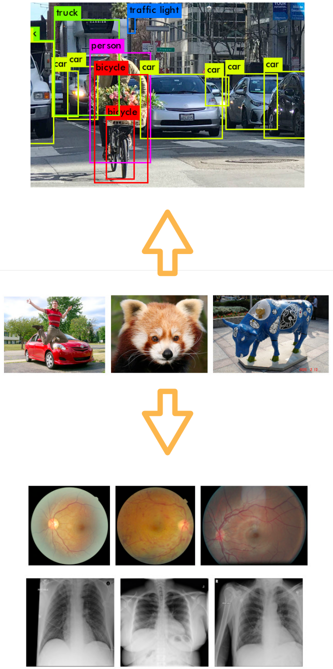
- Tensorflow Hub: https://tfhub.dev/
  - Tensorflow models submitted by users
- TIMM: https://github.com/rwightman/pytorch-image-models
  - PyTorch models collected by Ross Wightman

```
import timm
from torch import nn

model = timm.create_model('resnet18', pretrained=True)
model.fc = nn.Linear(model.fc.in_features, n_classes)
# Train model as a normal training job
```

Applications

- Fine-tuning pre-trained models (on ImageNet) is widely used in various CV applications:
    - Detection/segmentation (similar images but different targets)
    - Medical/satellite images (same task but very different images)
- Fine-tuning accelerates convergence
- Though not always improve accuracy
    - Training from scratch could get a similar accuracy, especially when the target dataset is also large



Summary

- Pre-train models on large-scale datasets (often image classification)
- Initialize weights with pre-trained models for down-stream tasks
- Fine-tuning accelerates converges and (sometimes) improves accuracy