21) Transfer Learning

TL via Model Copying

- Transfers a trained model to another dataset/problem
- Purpose: Saves computational & data resources to go further
- When to use: ① ur problem is similar to the source model's
  ② ur data is insufficient ③ Source model is deep
- Process: Fine-Tuning the model: Freezes some layers (≥0) & Retrain remaining
  ↑ more ① = more layers to freeze

Famous Pretrained CNNs

- PyTorch provides importable pretrained CNNs

① LeNet: standard convolution-pooling block

Flatten

x → Convolution → pooling → Convolution → pooling → dense → dense → dense

(28x28) (6@28x28) (6@14x14) (16@10x10) (16@5x5) (120,84) (84,10) (10,1)

② AlexNet: recognizes images in 1000 categories w/ human-level performance
- Model trained on 2 separate GPUs & Convolutions stacked together w/ ReLU
- more suitable for transfer learning

③ VGGNet: similar to AlexNet

④ ResNet: similar to AlexNet but some layers have residual/skipped connections to other layers

TL via Autoencoder

- Steps: ① Pretrain AE model ② Transfer encoder part of AE
  ③ Attach it to prediction layers (ex. FFN) ④ Fine-Tune this model
- Pros & Cons: ① Learn features tailored to ur data
  ② Can reuse data from Step 1 & 4 without overfitting
  ③ Do not necessarily learn features relevant for prediction