Dealing with small data

**Data augmentation**
- For image classification tasks: add rotation, scale, flip, contrast, color balance, noise...
- Model the "noise" non-meaningful transformations

- Use a simulator to produce quantities of input data

**Multi-tasking**
- Consider (at the same training time) another task

**Transfer learning**
- Sequential transfer: First \( C_s \) - Second \( C_R \)
- Pick a pre-trained network, pre-train \( V_{66} \) on ImageNet
- For computer vision task, \( V_{66} \) on ImageNet
- Small data (few labeled examples)
- Big data (with training from scratch)

**Forms of weak supervision**
- Few labeled examples
- Semi-supervision
- No labeled samples (small part)
- Example: when labeling is costly (requires time, expert)
Several approaches:

1) unsupervised training (on full set) → good representation → supervised training
   Typically: auto-encoders

2) supervised training → label some of the unlabeled samples
   Use
   \[ D_u \rightarrow \text{ newly build} \]

Issues: SF mistakes, learn from wrongly labeled data

Weak supervision

Less general: ex: labels could be noisy

Self-supervision

→ unsupervised pretraining
   → supervised task (on \( F \)) → with a fake task
   with labels for all samples in \( F \)

ex: image classification
→ image puzzle: extract patches from some image and ask for geometrical relation

→ add a relation to the image → task: retrieve the single (random)

→ data augmentation
  1. define “classes”:
     1 class = \{ all augmented input data coming from same sample \}

ex: video classification

by auxiliary task:
→ predict next frame: fully supervised
→ give 3 frames, ask whether temporal order is correct
Active learning

Same setting as semi-supervision, + ask some samples to be labeled

- Large dataset \( \mathcal{D} \subset \{ \mathbf{x}_i \} \)
- Lists for few: \( \{ y_{1i}, \ldots, y_{p_1} \} \) with \( p \ll |\mathcal{D}| \)
- Which \( x_i \) (with \( i \in [p] \)) to pick? tradeoff for labels

\[ \text{\( \hat{y}_i = \arg \min_{c \in \mathcal{C}} \sup_{\mathbf{x}_c} \hat{g}_i \quad \text{if classification task} \)} \]

Local methods

- To quantify the impact of the choice on the chosen sample only
- Uncertainty sampling:
  - Pick \( x_i \) for which the model is the most uncertain \( \rightarrow \) lowest prediction confidence
  \[ \arg \min_{\mathbf{x}_i} \sup_{c \in \mathcal{C}} \hat{g}_i \]

- Margin sampling
  \[ \hat{y}_i = \arg \min_{c \in \mathcal{C}} \hat{g}_i \]

- Entropy sampling:
  \[ H(\hat{g}_i) = -\sum_{c \in \mathcal{C}} \hat{g}_{ci} \log \hat{g}_{ci} \]
  - High: if prob. well dispersed over classes
  - Low: if Dirac peak

- Query by committee
  \[ \hat{y}_{ik} = \arg \min_{c \in \mathcal{C}} \frac{1}{k} \sum_{k=1}^{n} \hat{g}_{ci} \]
  - Do models agree?

\[ \text{\( \text{If most agree, pick \( x_i \)} \)} \]

Global methods

- To quantify impact of the choice over all dataset samples
- Expected model change
  \[ \text{do one gradient-descent step with chosen sample} \]
  \[ \theta_{t+1} = \theta_t - \eta \nabla \text{Loss}(\hat{g}_i, \hat{x}_i) \]
  \[ \text{for large \( \theta \) update} \]

- Direct pull or push
  \[ \text{use predictions as class probe estimate} \]
### Incorporation of Priors

- Small data
- Help the training of the network by adding priors from physical knowledge

#### A) Invariance

**Enforcement of invariance by design**

- Symmetry of the problem: Group of transformations $G$
  
  $V g \in G, \quad F(x) = F(g x)$

  - No need to learn it
  - Easier training vs. data augmentation
- Translation equivariance: 
  \[ F(\text{translated}(x)) = \text{Translated}(F(x)) \]

- Permutation invariance:
  \[ \text{one input: } x \xrightarrow{\text{permutation}} x' \text{ (raw permutation)} \]

[DeepSets]

Theorem: any permutation-invariant function can be re-written as:

\[ F(x) = g\left( \sum \psi(x^r) \right) \]

\[ \forall \psi, g \text{ ...} \]

\[ \sum \rightarrow g \rightarrow \text{ universal permut-} \]

\[ \text{equivariant approximator} \]

- Row-wise equivariant

- Stack 

- one block equivariant

- Ex: point clouds \(\rightarrow\) pointNet++

- Input: laser measurements on an object \(\rightarrow\) 3D object classification

- Ex: population genetics

- Input: DNA

- DNA from individuals living now \(\rightarrow\) lose order-invariant

- Task: demography site inference: to how many people were living 5000 years ago?

- Invariance to rotations

- Input: molecule

- Predict toxicity

- Input: equi-biases, inv. biases

- Predict
Learning the invariance → Spatial Transformers

(image) input \( x \) → Predict rotation angle \( \theta(x) \) → Network → Return \( \theta(x) \) → Works angle-insensitive

→ Capsule networks action: equiv

B) by task design ← metrics → data