Supervising Unsupervised Learning

Vikas K. Garg and Adam Kalai

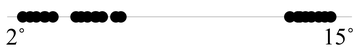
vgarg@csail.mit.edu, adum@microsoft.com

Microsoft

Massachusetts Institute of Technology

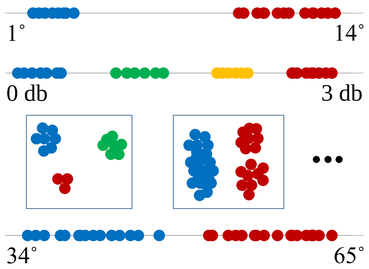
Motivation

Clustering problem in isolation:



How many clusters?

Clustering repository:



- No supervision ⇒ hard to define and evaluate learning problems
- Often annotations (e.g. cluster labels) available for diverse problems
- Patterns seen previously could be used to inform new unsupervised data
- Transfer across different representations, dimensionalities, and domains!
- Allows us to *provably learn* to perform unsupervised learning (UL)!

Setting

- **Given**
  - Annotated repository of n datasets $T = \{(X_i, Y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1}^{n}$
  - T drawn IID from (unknown) distribution μ over problems (X, Y)
  - Datasets $(X_i, Y_i) \in T$ may differ in representation, dimensions, etc.
  - Bounded loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to [0, 1]$
  - Loss of classifier $c \in \mathcal{Y}^{\mathcal{X}}$: $\ell_\mu(c) = \mathbb{E}_{(X,Y)\sim\mu}\, \ell(Y, c(X))$
- **Goal**
  - Learner $L : (\mathcal{X} \times \mathcal{Y})^* \to \mathcal{Y}^{\mathcal{X}}$ that outputs a classifier $L(T)$
  - Want the expected loss $\mathbb{E}_{T\sim\mu^n}\, \ell_\mu(L(T))$ to be low
- **Agnostic Learning**
  - Consider an empirical minimizer over UL algorithms $U \in \mathcal{U}$

$$\mathrm{ERM}_{\mathcal{U}}(T) \in \arg\min_{U\in\mathcal{U}} \sum_{(X,Y)\in T} \ell(Y, U(X))$$

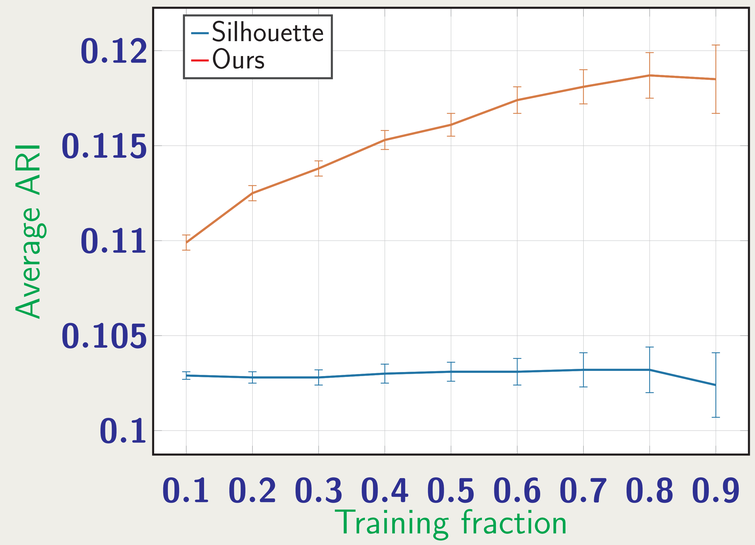
  - For any finite family $\mathcal{U}$, any distribution μ over problems $(X, Y) \in \mathcal{X} \times \mathcal{Y}$, and any $n \geq 1, \delta > 0$,

$$\Pr_{T\sim\mu^n}\left[\ell_\mu(\mathrm{ERM}_{\mathcal{U}}(T)) \leq \min_{U\in\mathcal{U}} \ell_\mu(U) + \sqrt{\frac{2}{n}\log\frac{|\mathcal{U}|}{\delta}}\right] \geq 1 - \delta$$

- **Leads to efficient procedures...**
  - Selecting clustering algorithm
  - Estimating number of clusters
  - Fitting threshold in single-linkage clustering
  - Removing outliers, recycling problems, etc.

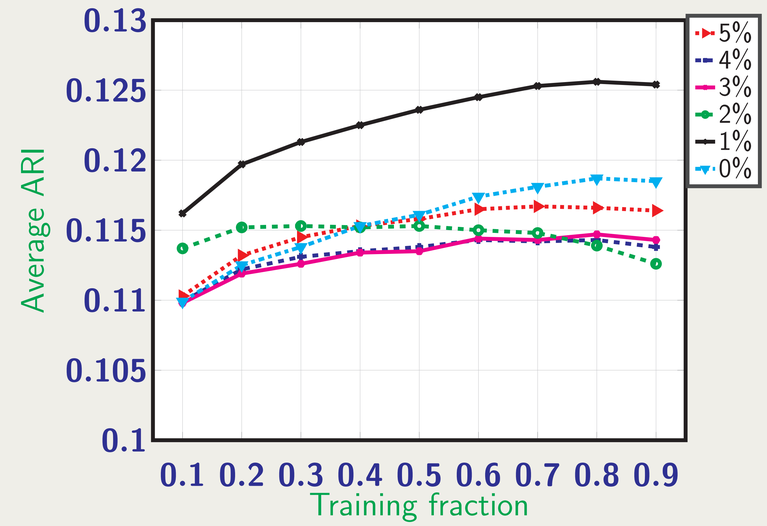
Example: selecting number of clusters

- Run k-means algorithm with different k on each train dataset
- Learn mapping from Silhouette Index (SI) to Adjusted Rand Index (ARI)
- Choose the number with maximum predicted ARI on any test dataset



Example: predicting fraction of outliers

- Remove points with large norms, cluster other points, and compute SI
- Put the removed points into clusters, and compute ARI
- Output the candidate fraction that predicts maximum ARI on test set

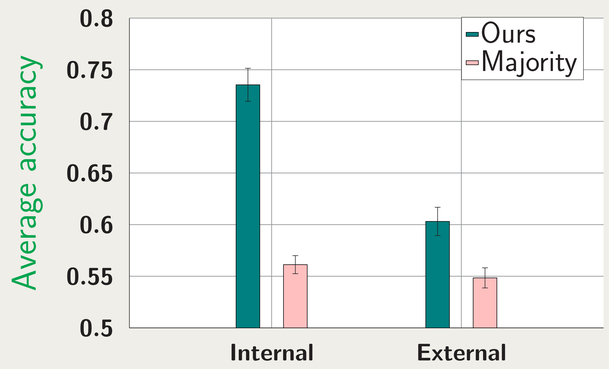


Good meta-clustering is possible!

- Desirable clustering properties
  - *Scale invariance (S)*: scaling the data should not change clustering
  - *Richness (R)*: any target clustering achievable by adjusting the distances between points
  - *Consistency (C)*: no change if points within a cluster pulled closer
- Kleinberg's impossibility theorem (2002)
  - No algorithm can satisfy all of S, R, and C!
- Intuition into impossibility result
  - Consider two 1D points a and b
  - If a and b too close, we can let them be in same cluster (R1)
  - If too far, we could force them to be in different clusters (R2)
  - But distances in (R1) and (R2) multiples of each other
  - Thus, scale invariance gets violated!
- Our idea : let previous datasets inform the scale on unseen datasets
  - We introduce Meta-scale invariance (MS)
    - Fix any distance functions $d_1, d_2, \ldots, d_t$ and ground truth clusterings $C_1, \ldots, C_t$ on sets $X_1, \ldots, X_t$. For any $\alpha > 0$, and any distance function d, if $M(d_1, C_1, \ldots, d_t, C_t) = A$ and $M(\alpha \cdot d_1, C_1, \ldots, \alpha \cdot d_t, C_t) = A'$, then $A(d) = A'(\alpha \cdot d)$
  - Our result : There exists a meta-clustering algorithm that satisfies (MS) and whose output always satisfies richness and consistency!

Deep learning binary similarity with multiple small datasets

- Sample pairs of examples from each small dataset.
- For each pair, also include covariance features specific to its dataset.
- Label 1 if the sampled pair comes from same cluster, 0 otherwise.
- Train a deep net classifier on all the pairs together.
- Predict whether test pair comes from same cluster or not.



External test datasets did not include any examples from training datasets!