Hierarchical sampling for active learning

Sanjoy Dasgupta and Daniel Hsu
University of California, San Diego
Active learning

Unlabeled data (raw signal): cheap, plentiful
e.g. text (web), speech (microphone), images (Flickr)

Labels (quantity to predict): often expensive
e.g. read/categorize articles, transcribe audio, identify/locate objects

Given: pool of unlabeled data, access to human labeler
Goal: learn an accurate classifier, requesting as few labels as possible
1. Efficient search through hypothesis space
   Label queries reduce set of likely hypotheses;
   Query points so as to shrink this set as quickly as possible
   (e.g. Query-by-committee [FSST93], region-of-disagreement [CAL93],
   agnostic active learners [BBL06, Han07, DHM07])

2. Exploit cluster structure in data
   Data is often “clustered” by class label;
   Need just a few queries in each cluster to identify an
   appropriate labeling of all of the data
   (e.g. Bayesian method with flexible priors [ZGL03])
General active learning strategies

1. **Efficient search through hypothesis space**
   Label queries reduce set of likely hypotheses;
   Query points so as to shrink this set as quickly as possible
   (e.g. Query-by-committee [FSST93], region-of-disagreement [CAL93],
   agnostic active learners [BBL06,Han07,DHM07])

2. **Exploit cluster structure in data**
   Data is often “clustered” by class label;
   Need just a few queries in each cluster to identify an
   appropriate labeling of all of the data
   (e.g. Bayesian method with flexible priors [ZGL03])
Typical active learning heuristics

- Start with a pool of unlabeled data.
- Query the labels of a few initial points
- Repeat:
  - Train a classifier on current set of labeled data
  - Choose unlabeled point closest to decision boundary
    (the most uncertain point, the point with smallest margin, ...)

Typical active learning heuristics

• Start with a pool of unlabeled data.
• Query the labels of a few initial points
• Repeat:
  - Train a classifier on current set of labeled data
  - Choose unlabeled point closest to decision boundary
    (the most uncertain point, the point with smallest margin, ...)

Unlabeled data distribution:

| 45% | 5% | 5% | 45% |
Typical active learning heuristics

- Start with a pool of unlabeled data.
- Query the labels of a few initial points
- Repeat:
  - Train a classifier on current set of labeled data
  - Choose unlabeled point closest to decision boundary (the most uncertain point, the point with smallest margin, ...)

Unlabeled data distribution:

\[ h_{\text{current}} \]
Typical active learning heuristics

• Start with a pool of unlabeled data.
• Query the labels of a few initial points
• Repeat:
  - Train a classifier on current set of labeled data
  - Choose unlabeled point closest to decision boundary
    (the most uncertain point, the point with smallest margin, ...)
Typical active learning heuristics

- Start with a pool of unlabeled data.
- Query the labels of a few initial points
- Repeat:
  - Train a classifier on current set of labeled data
  - Choose unlabeled point closest to decision boundary (the most uncertain point, the point with smallest margin, ...)

Labeled data distribution:

- $h_{opt}$
- $h_{current}$

- 45%
- 5%
- 5%
Typical active learning heuristics

- Start with a pool of unlabeled data.
- Query the labels of a few initial points
- Repeat:
  - Train a classifier on current set of labeled data
  - Choose unlabeled point closest to decision boundary
    (the most uncertain point, the point with smallest margin, ...)

Labeled data distribution:

- $h_{opt}$: 45% (blue)
- $h_{current}$: 5% (red)

Set of labeled data is not representative of underlying distribution!

“Missed cluster effect” (Schütze et al, 2006)

$\text{err}(h_{opt}) = 2.5\%, \text{ err}(h_{current}) \geq 5\%$
Consistency with active learning

- Should never do worse than random sampling (passive supervised learning)
- General methodology
  - Balance *random sampling* with *selective (active) sampling* so that sampling bias is properly managed
- Various tricks available to implement this
  - e.g. rejection sampling, confidence intervals [BBL06, DHM07]
Cluster-adaptive sampling

**Goal:** label every data point by assigning the majority label of each cluster to its constituents.

Result is a *fully* labeled data set (with mostly correct labels). Now use *any* supervised learning method to train a classifier!
Cluster-adaptive sampling

Initial pool of unlabeled data:
Cluster-adaptive sampling

Cluster the unlabeled data:
Cluster-adaptive sampling

Query the label of a few points in each cluster:
Cluster-adaptive sampling

Query the label of a few points in each cluster:

A mixed bag ...

Relatively pure ...

... seem to be stuck.
Cluster-adaptive sampling

Recourse: use a *hierarchical* clustering

Since cluster 2 is relatively pure, focus sampling on cluster 1 in hopes of discovering a better pruning.
Cluster-adaptive sampling

Recourse: use a *hierarchical* clustering

Since cluster 2 is relatively pure, focus sampling on cluster 1 in hopes of discovering a better pruning.
Cluster-adaptive sampling

Recourse: use a hierarchical clustering

Better pruning: \{ 2, 3, 4 \}
Cluster-adaptive sampling

Main idea:

Search for a pruning of the tree (hierarchical clustering) with “pure” nodes (clusters)

- Maintain a pruning $P$ of the tree
- Opportunistically choose a node (cluster) $v$ to sample from, then choose a random leaf (data point) $z$ within $v$
- Query label of $z$, update empirical counts of observed labels for each cluster containing $z$
- Empirical counts (+ confidence intervals) used to assess “purity” of a node
- Choose the best pruning of a cluster after sampling from it
Algorithm

- **INPUT:** hierarchical clustering $T$
- **INITIALIZE:** pruning $P = \{\text{root}\}$, labeling $L(\text{root}) = +1$
- **FOR** $t = 1, 2, \ldots$:
  - Set $v = \text{select-node}(P)$
  - Pick a random point $z$ in subtree $T_v$
  - Query $z$’s label
  - Update empirical counts for all nodes along path from $z$ to $v$
  - Choose best pruning and labeling $(P', L')$ of $T_v$;
    Set $P = (P \setminus \{v\}) \cup P'$, and $L(u) = L'(u)$ for all $u \in P'$
- **FOR EACH** $v \in P$: assign each leaf in $T_v$ the label $L(v)$
- **RETURN** the resulting fully-labeled data set
Algorithm

• INPUT: hierarchical clustering \( T \)
• INITIALIZE: pruning \( P = \{ \text{root} \} \), labeling \( L(\text{root}) = +1 \)
• FOR \( t = 1, 2, \ldots \):
  - Set \( v = \text{select-node}(P) \)
  - Pick a random point \( z \) in subtree \( T_v \)
  - Query \( z \)'s label
  - Update empirical counts for all nodes along path from \( z \) to \( v \)
  - Choose best pruning and labeling \((P', L')\) of \( T_v \);
    Set \( P = (P \setminus \{ v \}) \cup P' \), and \( L(u) = L'(u) \) for all \( u \in P' \)
• FOR EACH \( v \in P \): assign each leaf in \( T_v \) the label \( L(v) \)
• RETURN the resulting fully-labeled data set
Algorithm details

1. Building a hierarchical clustering:
   - Standard agglomerative (linkage) methods
   - Divisive methods (binary space partitioning)
   - Domain-specific distance measures (e.g. KL-divergence, manifold geodesic distance)
   - Bayesian methods

Just need that the resulting hierarchical clustering have a small, pure (in class label) pruning.
Algorithm details

2. Choosing a pruning and labeling:

Estimated error from assigning label \( l \) to node \( v \) is \( 1 - \hat{p}_{v,l} \)

Dynamic program cost function \( s(v) \) (roughly):

\[
s(v) = \min \begin{cases} 
1 & \forall \text{“well-estimated” } p_{v,l} \\
1 - \hat{p}_{v,l} & \text{if } v \text{ has children } a, b \\
\frac{|a|}{|v|} s(a) + \frac{|b|}{|v|} s(b) & \text{and some } p_{v,l} \text{ “well-estimated”}
\end{cases}
\]
Algorithm details

3. Selecting a node to sample from:

Many variations of \texttt{select-node}(P) possible

1. Choose node $v \in P \text{ w.p. } \propto |v|$
2. Choose node $v \in P \text{ w.p. } \propto |v| \cdot \left(1 - \hat{p}^{LB}_{v,l}\right)$

- Essentially random sampling
- Active sampling: avoids sampling from relatively pure nodes

Can also combine with:
- “PAC-Bayes”-style priors
- Sampling rules via hypothesis search (e.g. margin-based rules)
- ...
Consistency guarantees

• **With random sampling rule:**
  If there is a pruning of the tree to $k$ clusters with error $\eta$, the algorithm discovers a pruning with error $O(\eta)$ after $O(k/\eta)$ label queries.

• **With active sampling rule:**
  Never worse than a constant factor away from guarantees of random sampling.
Immediate extensions

- **Multi-class**: track multiple empirical counts; use multinomial confidence intervals
- **Batch-mode**: repeatedly call `select-node(P)`
- **Rare-category detection**:
  - Goal: discover “rare” classes (those with class priors < 0.01%)
  - e.g. uncover new fraud patterns, anomalies
  - Active sampling rule: helps balance “coverage” of data space; directs sampling away from “pure” majority-class regions
Experiments

• Tested cluster-adaptive sampling with active sampling rule
  - Used logistic regression to train a linear classifier on resulting labeled data set

• Compared to:
  - Random sampling (passive learning)
  - Margin-based sampling (query for labels near boundary of current classifier)
  - Both use logistic regression as base learner
Experiments

Newsgroup text (bag-of-words features)

alt.atheism/talk.religion.misc

10-class MNIST OCR digits

“Error” is test error on held-out sample of final resulting classifier
Future work

• Characterization of sample complexity improvements
  - What is the optimal sampling rule?
  - When are exponential savings possible?

• Generalize method to other structures discovered with unsupervised learning
Summary

- Cluster-adaptive sampling method for active learning
  - Discovers viable clustering if it exists (at any level) in a hierarchical clustering
  - Manages sampling bias by combining valid confidence intervals (error bounds)
  - Fall-back consistency guarantee
  - Empirically outperforms random sampling and competitive with unsafe heuristics
Thanks!