Towards a theory of interactive learning

Sanjoy Dasgupta

University of California, San Diego
Interactive learning

Adaptive engagement between a learning agent and information source(s).
Interactive learning

Adaptive engagement between a learning agent and information source(s).
Interactive learning

Adaptive engagement between a learning agent and information source(s).
Interactive learning

Adaptive engagement between a learning agent and information source(s).
Outline

1. Interactive structure learning
2. Learning from partial correction
3. Structural query-by-committee
4. Interactive hierarchical clustering
Example: active learning of classifiers

Unlabeled data is often plentiful and cheap: documents off the web, speech samples, images, video. *But labeling can be expensive.*

Active learning: Machine queries just a few labels, choosing wisely and adaptively.

- Good querying schemes?
- Tradeoff between # labels and error rate of final classifier?
Example: interaction for unsupervised learning

Lots of progress on algorithms for unsupervised learning tasks, like

- Clustering
- Embedding
- Topic modeling
- ...
Example: interaction for unsupervised learning

Lots of progress on algorithms for unsupervised learning tasks, like

- Clustering
- Embedding
- Topic modeling
- ...

But these could all benefit from interaction!

- What kind of feedback?
- How to incorporate?
Other examples

- Interactive learning of structured-output predictors
- Interactive knowledge graph construction
- Interactive scientific discovery

...
Other examples

- Interactive learning of structured-output predictors
- Interactive knowledge graph construction
- Interactive scientific discovery

Plan: Fit all these into a general framework.

Desirable outcomes:

- Generic interactive learning algorithms
- Bounds on “interaction complexity”
- Formal relationship with existing models of learning
Interactive structure learning

Components of the learning problem:

- **Space of instances \( \mathcal{X} \).**
  
  Input space for classifier, or points to cluster, or sentences to tag, or items on which to build a knowledge graph.
Interactive structure learning

Components of the learning problem:

- **Space of instances** $\mathcal{X}$.
  Input space for classifier, or points to cluster, or sentences to tag, or items on which to build a knowledge graph.

- **Want to learn a structure** over $\mathcal{X}$, chosen from a set $\mathcal{G}$. Examples:
  - classifiers on $\mathcal{X}$
  - hierarchical clusterings of $\mathcal{X}$
  - embeddings of $\mathcal{X}$
  - part-of-speech taggers for $\mathcal{X}$
  - knowledge graphs on $\mathcal{X}$

There is some target $g^* \in \mathcal{G}$ that meets the user's needs. In fact, there may be many. Call them $\mathcal{G}^* \subseteq \mathcal{G}$. 
Interactive structure learning

Components of the learning problem:

- Space of instances $\mathcal{X}$.
  Input space for classifier, or points to cluster, or sentences to tag, or items on which to build a knowledge graph.

- Want to learn a **structure** over $\mathcal{X}$, chosen from a set $\mathcal{G}$.

  Examples:
  - classifiers on $\mathcal{X}$
  - hierarchical clusterings of $\mathcal{X}$
  - embeddings of $\mathcal{X}$
  - part-of-speech taggers for $\mathcal{X}$
  - knowledge graphs on $\mathcal{X}$

- There is some target $g^* \in \mathcal{G}$ that meets the user’s needs.
  In fact, there may be many. Call them $\mathcal{G}^* \subseteq \mathcal{G}$. 
Loss function on structures

Which structure would be chosen in the absence of interaction?

1. Loss function $L(g)$ over structures $g \in \mathcal{G}$
   \[
   \min L(g) \text{ subject to expert-supplied constraints}
   \]
   Examples:
   - $L(g) =$ cost function for clusterings $g$
   - $L(g) =$ regularization term for classifier $g$
   - $L(g) =$ smoothness of metric $g$ wrt default distance

2. Prior distribution $\pi(g)$ over $\mathcal{G}$
   \[
   \max \pi(g) \text{ subject to expert-supplied constraints}
   \]
   E.g. $\pi(g) \propto e^{-L(g)}$. 

What kind of interaction is allowed?
Loss function on structures

Which structure would be chosen in the absence of interaction?

1. Loss function \( L(g) \) over structures \( g \in \mathcal{G} \)

\[
\min L(g) \text{ subject to expert-supplied constraints}
\]

Examples:
- \( L(g) = \) cost function for clusterings \( g \)
- \( L(g) = \) regularization term for classifier \( g \)
- \( L(g) = \) smoothness of metric \( g \) wrt default distance

2. Prior distribution \( \pi(g) \) over \( \mathcal{G} \)

\[
\max \pi(g) \text{ subject to expert-supplied constraints}
\]

E.g. \( \pi(g) \propto e^{-L(g)} \).

What kind of interaction is allowed?
Example: feedback for clustering

\( \mathcal{X} \): points to be clustered; \( \mathcal{G} \): space of possible clusterings

Machine has chosen some clustering \( g \in \mathcal{G} \) and wants feedback.

\begin{itemize}
  \item Look at protocols for which interaction time is constant.
  \item Show expert the restriction of \( g \) to \( O(1) \) points from \( \mathcal{X} \).
\end{itemize}
Example: feedback for clustering

\( \mathcal{X} \): points to be clustered; \( \mathcal{G} \): space of possible clusterings

Machine has chosen some clustering \( g \in \mathcal{G} \) and wants feedback.

- Look at protocols for which interaction time is constant.
- Show expert the restriction of \( g \) to \( O(1) \) points from \( \mathcal{X} \).
Example: feedback for clustering

\( \mathcal{X} \): points to be clustered; \( \mathcal{G} \): space of possible clusterings

Machine has chosen some clustering \( g \in \mathcal{G} \) and wants feedback.

- Look at protocols for which interaction time is constant.
- Show expert the restriction of \( g \) to \( O(1) \) points from \( \mathcal{X} \).

E.g. must-link dolphin-whale
Feedback, more generally

The learner wants feedback on some structure $g \in G$. Interacts with an information source: “expert”.
Feedback, more generally

The learner wants feedback on some structure $g \in \mathcal{G}$. Interacts with an information source: “expert”.

Difficult to fathom $g$ in its entirety:

- Too large
- Incomprehensible parametrization
Feedback, more generally

The learner wants feedback on some structure $g \in \mathcal{G}$. Interacts with an information source: “expert”.

Difficult to fathom $g$ in its entirety:
- Too large
- Incomprehensible parametrization
Feedback, more generally

The learner wants feedback on some structure $g \in \mathcal{G}$. Interacts with an information source: “expert”.

Difficult to fathom $g$ in its entirety:
- Too large
- Incomprehensible parametrization

Constant-time rounds of interaction:
- Learner displays a snapshot of $g$. For instance: the restriction of $g$ to a small subset $S \subseteq \mathcal{X}$.
- Expert either accepts this snapshot or fixes part of it. These corrections serve as constraints.
Feedback, more generally

The learner wants feedback on some structure \( g \in \mathcal{G} \). Interacts with an information source: “expert”.

Difficult to fathom \( g \) in its entirety:
- Too large
- Incomprehensible parametrization

Constant-time rounds of interaction:
- Learner displays a snapshot of \( g \).
  For instance: the restriction of \( g \) to a small subset \( S \subseteq \mathcal{X} \).
- Expert either accepts this snapshot or fixes part of it.
  These corrections serve as constraints.

Requirement on snapshots:
\[
g \in \mathcal{G}^* \text{ iff expert accepts all snapshots}
\]
Example: hierarchical clustering

\[ \mathcal{G} = \text{all hierarchies on } \mathcal{X}, \text{ and } g^* = \text{specific target hierarchy}. \]
Example: hierarchical clustering

\[ \mathcal{G} = \text{all hierarchies on } \mathcal{X}, \text{ and } g^* = \text{specific target hierarchy.} \]

Learner’s current best guess: \( g \)
Example: hierarchical clustering

\[ \mathcal{G} = \text{all hierarchies on } \mathcal{X}, \text{ and } g^* = \text{specific target hierarchy.} \]

Learner’s current best guess: \( g \)
Shows expert the restriction of \( g \) to a small set of points

![Hierarchical Clustering Diagram]

- zebra
- dolphin
- elephant
- whale
- mouse
- rabbit

Expert either:
- Accepts, i.e. \( g \) coincides with \( g^* \) on these points
- Or supplies a triplet that is violated by \( g \).
Example: hierarchical clustering

\[ \mathcal{G} = \text{all hierarchies on } \mathcal{X}, \text{ and } g^* = \text{specific target hierarchy.} \]

Learner’s current best guess: \( g \)
Shows expert the restriction of \( g \) to a small set of points

Expert either:

- Accepts, i.e. \( g \) coincides with \( g^* \) on these points
- Or supplies a triplet that is violated by \( g \).
Example: hierarchical clustering

$\mathcal{G} = \text{all hierarchies on } \mathcal{X}$, and $g^* = \text{specific target hierarchy}$.

Learner’s current best guess: $g$
Show expert the restriction of $g$ to a small set of points

Expert either:
- Accepts, i.e. $g$ coincides with $g^*$ on these points
- Or supplies a triplet that is violated by $g$.

Key property: $g = g^*$ iff they agree on all triplets
Questions and atomic subquestions

Learner’s current model: $g$

![Diagram showing a tree with zebra, dolphin, elephant, whale, mouse, and rabbit nodes.](image)
Questions and atomic subquestions

Learner’s current model: $g$

```
zebra         dolphin

elephant    whale    mouse    rabbit
```

**Snapshot:** $g(\{\text{dolphin, elephant, mouse, rabbit, whale, zebra}\})$.

- That is, treat $g$ as a function: $g : (x^6) \to \{\text{trees on six leaves}\}$.
Questions and atomic subquestions

Learner’s current model: \( g \)

\[ \text{Snapshot: } g(\{\text{dolphin, elephant, mouse, rabbit, whale, zebra}\}). \]

- That is, treat \( g \) as a function: \( g : \left( \binom{X}{6} \right) \rightarrow \{\text{trees on six leaves}\}. \)
- Questions: sets of six points. \( Q = \left( \binom{X}{6} \right) \)
- Learner picks some \( q \in Q \) and shows expert \( g(q) \)
Questions and atomic subquestions

Learner’s current model: $g$

```
zebra          dolphin
  /   \
elephant whale mouse rabbit
dolphin whale
```

Snapshot: $g$({dolphin, elephant, mouse, rabbit, whale, zebra}).

- That is, treat $g$ as a function: $g : \binom{X}{6} \rightarrow \{\text{trees on six leaves}\}$.
- Questions: sets of six points. $Q = \binom{X}{6}$
- Learner picks some $q \in Q$ and shows expert $g(q)$
Questions and atomic subquestions

Learner’s current model: \( g \)

\[
\text{Snapshot: } g(\{\text{dolphin, elephant, mouse, rabbit, whale, zebra}\}).
\]

- That is, treat \( g \) as a function: \( g : \binom{X}{6} \rightarrow \{\text{trees on six leaves}\} \).
- Questions: sets of six points. \( Q = \binom{X}{6} \)
- Learner picks some \( q \in Q \) and shows expert \( g(q) \)
- There are also smaller \textit{atomic} questions, \( A = \binom{X}{3} \).
- And \( g \) is also a function \( g : A \rightarrow \{\text{trees on 3 leaves}\} \).
Questions and atomic subquestions

Learner’s current model: $g$

**Snapshot:** $g(\{\text{dolphin, elephant, mouse, rabbit, whale, zebra}\})$.

- That is, treat $g$ as a function: $g : \binom{X}{6} \rightarrow \{\text{trees on six leaves}\}$.
- Questions: sets of six points. $Q = \binom{X}{6}$
- Learner picks some $q \in Q$ and shows expert $g(q)$
- There are also smaller *atomic* questions, $A = \binom{X}{3}$.
- And $g$ is also a function $g : A \rightarrow \{\text{trees on 3 leaves}\}$.
- Each $q \in Q$ contains atomic subquestions $A(q) \subseteq A$.
- Expert provides feedback on one of these subquestions, $a \in A(q)$, for which $g(a) \neq g^*(a)$. 
Summary of protocol

Learning problem:
- Instance space $\mathcal{X}$, structures $\mathcal{G}$ over $\mathcal{X}$
- Target structures: $\mathcal{G}^* \subseteq \mathcal{G}$

Protocol for learning:
Initial set of candidate structures: $\mathcal{G}_0 = \mathcal{G}$
For $t = 0, 1, 2, \ldots$:
- Learner selects $g_t \in \mathcal{G}_t$, e.g. $\text{arg min}_{g \in \mathcal{G}_t} L(g)$.
- Learner shows expert a snapshot of $g_t$
  (picks a question $q \in Q$ and shows expert $q$ and $g_t(q)$)
- If snapshot is correct:
  - Expert accepts it
- Else:
  - Expert corrects a piece of it
    (provides $g^*(a)$ for some subquestion $a \in A(q)$ on which $g_t$ is wrong)
- $\mathcal{G}_{t+1} = \text{structures in } \mathcal{G}_t \text{ that meet the new constraints}$
1. Reduction to multiclass classification

E.g. Think of any hierarchical clustering as a function from (subsets of $s$ points) to (trees with $s$ leaves):

$$\{\text{dolphin, elephant, mouse, whale}\} \rightarrow \text{elephant, mouse, dolphin, whale}$$
1. **Reduction to multiclass classification**

E.g. Think of any hierarchical clustering as a function from (subsets of $s$ points) to (trees with $s$ leaves):

\[
\{\text{dolphin, elephant, mouse, whale}\} \rightarrow
\]

\[\text{elephant} \quad \text{mouse} \quad \text{dolphin} \quad \text{whale}\]

Suggests many algorithms for interactive structure learning.
2. Partial correction

Benefits over the usual question-answer paradigm:
• Natural and intuitive interface that provides more context
• Gives the expert a chance to provide a teaching signal: identify key errors rather than minor ones
• More likely to contain an error than a single atomic subquestion
• More choice ⇒ more reliable feedback?
2. Partial correction

Benefits over the usual question-answer paradigm:

• Natural and intuitive interface that provides more context
• Gives the expert a chance to provide a teaching signal: identify key errors rather than minor ones
• More likely to contain an error than a single atomic subquestion
• More choice ⇒ more reliable feedback?
2. Partial correction

Benefits over the usual question-answer paradigm:

- Natural and intuitive interface that provides more context
- Gives the expert a chance to provide a teaching signal: identify key errors rather than minor ones
- More likely to contain an error than a single atomic subquestion
- More choice $\Rightarrow$ more reliable feedback?
Summary of protocol

Learning problem:
- Instance space $\mathcal{X}$, structures $\mathcal{G}$ over $\mathcal{X}$
- Target structures: $\mathcal{G}^* \subseteq \mathcal{G}$

Protocol for learning:
Initial set of candidate structures: $\mathcal{G}_0 = \mathcal{G}$
For $t = 0, 1, 2, \ldots$:
- Learner selects $g_t \in \mathcal{G}_t$, e.g. $\arg\min_{g \in \mathcal{G}_t} L(g)$.
- Learner shows expert a snapshot of $g_t$ (picks a question $q \in Q$ and shows expert $q$ and $g_t(q)$)
- If snapshot is correct:
  - Expert accepts it
- Else:
  - Expert corrects a piece of it (provides $g^*_a$ for some subquestion $a \in A(q)$ on which $g_t$ is wrong)
  - $\mathcal{G}_{t+1} = \text{structures in } \mathcal{G}_t \text{ that meet the new constraints}$
Outline

1. Interactive structure learning
2. Learning from partial correction (with Mike Luby)
3. Structural query-by-committee
4. Interactive hierarchical clustering
Toy example

Structures to learn: threshold classifiers on $\mathcal{X} = [0, 1]$.

$\mathcal{G} = \{g_w : w \in [0, 1]\}$, $g_w(x) = 1(x \geq w)$.

Target $g^* = g_0$, i.e. everywhere 1.
Toy example

Structures to learn: threshold classifiers on $\mathcal{X} = [0, 1]$.

$$
\mathcal{G} = \{g_w : w \in [0, 1]\}, \quad g_w(x) = 1(x \geq w).
$$

Target $g^* = g_0$, i.e. everywhere 1.

Learning algorithm:

- Initially take threshold $w_1 = 1$.
- Later, threshold $w_t = \text{smallest } x \text{ for which a 1 label has been seen}$
Toy example

Structures to learn: threshold classifiers on $\mathcal{X} = [0, 1]$.

$$\mathcal{G} = \{g_w : w \in [0, 1]\}, \quad g_w(x) = 1(x \geq w).$$

Target $g^* = g_0$, i.e. everywhere 1.

Learning algorithm:
- Initially take threshold $w_1 = 1$.
- Later, threshold $w_t = \text{smallest } x \text{ for which a 1 label has been seen}$

Expert sees $c$ points chosen at random from $[0, 1]$, labeled by current $w_t$. 
Toy example

Structures to learn: threshold classifiers on $\mathcal{X} = [0, 1]$.

$\mathcal{G} = \{g_w : w \in [0, 1]\}, \quad g_w(x) = 1(x \geq w)$.

Target $g^* = g_0$, i.e. everywhere 1.

Learning algorithm:

- Initially take threshold $w_1 = 1$.
- Later, threshold $w_t = \text{smallest } x \text{ for which a 1 label has been seen}$

Expert sees $c$ points chosen at random from $[0, 1]$, labeled by current $w_t$.

Which error will the expert point out?
The two extremal policies for the expert:

- LEFT: pick the leftmost (smallest) misclassified point.
- RIGHT: pick the rightmost misclassified point.
Toy example, cont’d

The two extremal policies for the expert:

- **LEFT**: pick the leftmost (smallest) misclassified point.
- **RIGHT**: pick the rightmost misclassified point.
Convergence rates for partial correction

Here $Q = \binom{\mathcal{X}}{6}$ and $\mathcal{A} = \binom{\mathcal{X}}{3}$

- Each query $q$ contains $c = \binom{6}{3} = 20$ atomic subquestions $A(q)$
- Pick a distribution $\mu$ over $Q$, e.g. uniform
- This induces a distribution $\nu$ over $\mathcal{A}$ (also uniform)
- Error rate of any hierarchy $g$: fraction of incorrect triples,

$$\text{err}(g) = \Pr_{a \sim \nu}(g(a) \neq g^*(a)).$$

Goal: want $\text{err}(g) \leq \epsilon$.

- Random (i.i.d.) labeled triples: $O\left(\frac{1}{\epsilon} \ln |\mathcal{G}|\right)$ suffice.

But what if the triples are generated by partial correction?
Convergence rates for partial correction

If we received random triples, we’d need $O\left(\frac{1}{\varepsilon} \ln |G|\right)$ of them to get an $\varepsilon$-good structure.
If we received random triples, we’d need $O\left(\frac{1}{\epsilon} \ln |\mathcal{G}| \right)$ of them to get an $\epsilon$-good structure.

But: even if snapshots are chosen at random, the feedback triples are not i.i.d.!
If we received random triples, we’d need $O\left(\frac{1}{\epsilon} \ln |\mathcal{G}| \right)$ of them to get an $\epsilon$-good structure.

**But: even if snapshots are chosen at random, the feedback triples are not i.i.d.**!

Sanity check: no matter what subquestions the expert chooses, sample complexity is $\tilde{O}\left(\frac{1}{\epsilon} \ln |\mathcal{G}| \right)$. 
Let $\nu$ be the desired distribution over atomic subquestions $\mathcal{A}$. Let $c$ be the maximum number of atomic questions in each query.

1. The distribution induced by partial correction on round $t$ is some $\Gamma_t$ such that:

$$\Gamma_t(a) \leq c \cdot \nu(a).$$

Therefore, at least $(1/c)$ fraction of the space $\mathcal{A}$ gets sampled.

2. Structures that have high error in the sampled region will be eliminated.

3. The sampling region keeps moving. Once a region has been thoroughly sampled, structures that are bad in that region are removed. Subsequently-chosen structures $g_t$ are bad elsewhere.
Outline

1. Interactive structure learning
2. Learning from partial correction
3. Structural query-by-committee (with Chris Tosh)
4. Interactive hierarchical clustering
Intelligent querying, by committee

**QBC** (Freund, Seung, Sompolinsky, Tishby)

\( \mathcal{H}_0 \): family of binary classifiers  
\( \pi \): prior on \( \mathcal{H}_0 \)  
\( \mu \): distribution on \( \mathcal{X} \)  

At time \( t = 0, 1, 2, \ldots \):

- Get a new data point \( x_t \sim \mu \)
- Pick \( h, h' \sim \pi|_{\mathcal{H}_t} \)
- If \( h(x_t) \neq h'(x_t) \):
  - Query the label \( y_t \)
  - \( \mathcal{H}_{t+1} = \{ h \in \mathcal{H}_t : h(x_t) = y_t \} \)
- Else: \( \mathcal{H}_{t+1} = \mathcal{H}_t \)

Structural QBC

\( \mathcal{G}_0 \): family of structures  
\( \pi \): prior on \( \mathcal{G}_0 \)  
\( \mu \): distribution on \( Q \)  

At time \( t = 0, 1, 2, \ldots \):

- Get a new query \( q_t \sim \mu \)
- Pick \( g, g' \sim \pi|_{\mathcal{G}_t} \)
- With probability \( d(g, g'; q_t) \):
  - Present \( q_t, g(q_t) \) to expert
  - Receive atomic constraints
- \( \mathcal{G}_{t+1} = \{ g \in \mathcal{G}_t : g \text{ satisfies } C_t \} \)
- Else: \( \mathcal{G}_{t+1} = \mathcal{G}_t \)

\( d(g, g'; q) \) is the fraction of atomic subquestions of \( q \) on which \( g, g' \) disagree.

Statistical guarantees – convergence, rates – continue to hold.
Intelligent querying, by committee

**QBC** (Freund, Seung, Sompolinsky, Tishby)

\( \mathcal{H}_0 \): family of binary classifiers
\( \pi \): prior on \( \mathcal{H}_0 \)
\( \mu \): distribution on \( \mathcal{X} \)

At time \( t = 0, 1, 2, \ldots \):
- Get a new data point \( x_t \sim \mu \)
- Pick \( h, h' \sim \pi|_{\mathcal{H}_t} \)
- If \( h(x_t) \neq h'(x_t) \):
  - Query the label \( y_t \)
  - \( \mathcal{H}_{t+1} = \{ h \in \mathcal{H}_t : h(x_t) = y_t \} \)
- Else: \( \mathcal{H}_{t+1} = \mathcal{H}_t \)

**Structural QBC**

\( \mathcal{G}_0 \): family of structures
\( \pi \): prior on \( \mathcal{G}_0 \)
\( \mu \): distribution on \( \mathcal{Q} \)

At time \( t = 0, 1, 2, \ldots \):
- Get a new query \( q_t \sim \mu \)
- Pick \( g, g' \sim \pi|_{\mathcal{G}_t} \)
- With probability \( d(g, g'; q_t) \):
  - Present \( q_t, g(q_t) \) to expert
  - Receive atomic constraints \( C_t \)
  - \( \mathcal{G}_{t+1} = \{ g \in \mathcal{G}_t : g \text{ satisfies } C_t \} \)
- Else: \( \mathcal{G}_{t+1} = \mathcal{G}_t \)

Statistical guarantees – convergence, rates – continue to hold.
Intelligent querying, by committee

**QBC** (Freund, Seung, Sompolinsky, Tishby)

\( \mathcal{H}_0 \): family of binary classifiers  
\( \pi \): prior on \( \mathcal{H}_0 \)  
\( \mu \): distribution on \( \mathcal{X} \)

At time \( t = 0, 1, 2, \ldots \):
- Get a new data point \( x_t \sim \mu \)
- Pick \( h, h' \sim \pi|_{\mathcal{H}_t} \)
- If \( h(x_t) \neq h'(x_t) \):
  - Query the label \( y_t \)
  - \( \mathcal{H}_{t+1} = \{ h \in \mathcal{H}_t : h(x_t) = y_t \} \)
- Else: \( \mathcal{H}_{t+1} = \mathcal{H}_t \)

**Structural QBC**

\( \mathcal{G}_0 \): family of structures  
\( \pi \): prior on \( \mathcal{G}_0 \)  
\( \mu \): distribution on \( \mathcal{Q} \)

At time \( t = 0, 1, 2, \ldots \):
- Get a new query \( q_t \sim \mu \)
- Pick \( g, g' \sim \pi|_{\mathcal{G}_t} \)
- With probability \( d(g, g'; q_t) \):
  - Present \( q_t, g(q_t) \) to expert
  - Receive atomic constraints \( C_t \)
  - \( \mathcal{G}_{t+1} = \{ g \in \mathcal{G}_t : g \text{ satisfies } C_t \} \)
- Else: \( \mathcal{G}_{t+1} = \mathcal{G}_t \)

\( d(g, g'; q) \) = fraction of atomic subquestions of \( q \) on which \( g, g' \) disagree.

Statistical guarantees – convergence, rates – continue to hold.
Volume versus diameter

QBC (and many other schemes) pick queries to quickly shrink the volume of the version space: its probability mass under the prior $\pi$.

$G$: 

[Diagram of a square]
Volume versus diameter

QBC (and many other schemes) pick queries to quickly shrink the volume of the version space: its probability mass under the prior $\pi$.

$G$: 

![Diagram showing the volume versus diameter concept]
Volume versus diameter

QBC (and many other schemes) pick queries to quickly shrink the volume of the version space: its probability mass under the prior $\pi$.

$\mathcal{G}$:
Volume versus diameter

QBC (and many other schemes) pick queries to quickly shrink the volume of the version space: its probability mass under the prior $\pi$.

$G$: [Diagram of a geometric shape]
Volume versus diameter

QBC (and many other schemes) pick queries to quickly shrink the volume of the version space: its probability mass under the prior $\pi$.

$\mathcal{G}$:
Volume versus diameter

QBC (and many other schemes) pick queries to quickly shrink the volume of the version space: its probability mass under the prior $\pi$.

Better idea: decrease the diameter of the version space, where

$$d(g, g') = \Pr_{a \sim \nu}(g(a) \neq g'(a)).$$
Volume versus diameter

QBC (and many other schemes) pick queries to quickly shrink the volume of the version space: its probability mass under the prior $\pi$.

Better idea: decrease the diameter of the version space, where

$$d(g, g') = \Pr_{a \sim \nu}(g(a) \neq g'(a)).$$

Work in progress: extending this from active learning of binary classifiers to the general structure learning model.
Outline

1 Interactive structure learning
2 Learning from partial correction
3 Structural query-by-committee
4 Interactive hierarchical clustering (with Sharad Vikram)
Hierarchical clustering

Useful tool for exploratory data analysis:
- Capture structure at all scales
- Well-established algorithms like average linkage.

As usual, the trees returned by these algorithms aren’t necessarily aligned with the user’s needs.
Hierarchical clustering with interaction

\( X = \) a set of points, \( \mathcal{G} = \) all hierarchies on these points.

Three ingredients needed:
Hierarchical clustering with interaction

\[ X = \text{a set of points}, \ G = \text{all hierarchies on these points}. \]

Three ingredients needed:


Feedback: triplet constraint like \((\{\text{dolphin, whale}\}, \text{zebra})\)
Hierarchical clustering with interaction

\( X = \) a set of points, \( \mathcal{G} = \) all hierarchies on these points.

Three ingredients needed:


Feedback: triplet constraint like \( (\{\text{dolphin}, \text{whale}\}, \text{zebra}) \)

2. A cost function \( L : \mathcal{G} \to \mathbb{R} \) over hierarchies.
Hierarchical clustering with interaction

\[ X = \text{a set of points, } \mathcal{G} = \text{all hierarchies on these points.} \]

Three ingredients needed:


Feedback: triplet constraint like \((\{\text{dolphin, whale}\}, \text{zebra})\)

2. A cost function \(L : \mathcal{G} \rightarrow \mathbb{R}\) over hierarchies.
   
   Oops... we don’t have this!
Hierarchical clustering with interaction

\( X = \) a set of points, \( \mathcal{G} = \) all hierarchies on these points.

Three ingredients needed:


Feedback: triplet constraint like \((\{ \text{dolphin, whale} \}, \text{zebra})\)

2. A cost function \( L : \mathcal{G} \rightarrow \mathbb{R} \) over hierarchies.
   
   Oops... we don’t have this!

3. An algorithm for \( \min\{ L(T) : T \in \mathcal{G} \text{ satisfies constraints} \} \)
A cost function for hierarchical clustering

Input: a similarity function on $X = \{x_1, \ldots, x_n\}$
A cost function for hierarchical clustering

Input: a similarity function on $X = \{x_1, \ldots, x_n\}$

Can represent as an undirected graph with weights $w_{ij}$. Here’s an example with unit weights:

![Graph Example](image)
A cost function for hierarchical clustering

Input: a similarity function on \( X = \{x_1, \ldots, x_n\} \)

Can represent as an undirected graph with weights \( w_{ij} \). Here’s an example with unit weights:

![Graph representation]

Idea for a cost function:

- Charge for edges that are cut.
A cost function for hierarchical clustering

Input: a similarity function on $X = \{x_1, \ldots, x_n\}$

Can represent as an undirected graph with weights $w_{ij}$. Here’s an example with unit weights:

![Graph example](image)

Idea for a cost function:

- Charge for edges that are cut.
  But: in a hierarchical clustering, all edges are cut.
A cost function for hierarchical clustering

Input: a similarity function on $X = \{x_1, \ldots, x_n\}$

Can represent as an undirected graph with weights $w_{ij}$. Here’s an example with unit weights:

![Graph example](image)

Idea for a cost function:

- Charge for edges that are cut.
  - But: in a hierarchical clustering, all edges are cut.
- **Charge more the “higher up” an edge is cut.**
Cost function, cont’d

\[ L(T) = \sum_{i,j} w_{ij} \cdot \#(\text{ancestors of } i, j) \]
Cost function, cont’d

\[ L(T) = \sum_{i,j} w_{ij} \cdot \#(\text{descendants of lowest common ancestor of } i, j) \]
Cost function, cont’d

\[ L(T) = \sum_{i,j} w_{ij} \cdot \# \text{(descendants of lowest common ancestor of } i,j) \]
Properties of cost function

\[ L(T) = \sum_{i,j} w_{ij} \cdot \#(\text{descendants of lowest common ancestor of } i, j) \]

- There is always an optimal tree that is binary.
Properties of cost function

\( L(T) = \sum_{i,j} w_{ij} \cdot \#(\text{descendants of lowest common ancestor of } i, j) \)

- There is always an optimal tree that is binary.

- If the similarity graph is disconnected, the top split of the optimal tree must cut no edges.
Three canonical examples

\[ L(T) = \sum_{i,j} w_{ij} \cdot \# \text{(descendants of lowest common ancestor of } i, j) \]

1 Line graph on \( n \) nodes.

Unbalanced tree: \( \Omega(n) \). Balanced tree: \( O(\log n) \).

Complete graph. All trees have the same cost.

Planted partition model. Correct clustering in expectation.
Three canonical examples

\[ L(T) = \sum_{i,j} w_{ij} \cdot \#(\text{descendants of lowest common ancestor of } i, j) \]

1 Line graph on \( n \) nodes.

Unbalanced tree: cost \( \Omega(n) \). Balanced tree: \( O(\log n) \).

Complete graph. All trees have the same cost.

Planted partition model. Correct clustering in expectation.
Three canonical examples

\[ L(T) = \sum_{i,j} w_{ij} \cdot \#(\text{descendants of lowest common ancestor of } i, j) \]

1. Line graph on \( n \) nodes.

Unbalanced tree: cost \( \Omega(n) \). Balanced tree: \( O(\log n) \).
Three canonical examples

\[ L(T) = \sum_{i,j} w_{ij} \cdot \#(\text{descendants of lowest common ancestor of } i, j) \]

1. Line graph on \( n \) nodes.

Unbalanced tree: cost \( \Omega(n) \). Balanced tree: \( O(\log n) \).

2. Complete graph. All trees have the same cost.
### Three canonical examples

\[
L(T) = \sum_{i,j} w_{ij} \cdot \# \text{(descendants of lowest common ancestor of } i, j) \]

1. **Line graph on** \(n\) **nodes.**

   ![Line graph on n nodes](image)

   Unbalanced tree: cost \(\Omega(n)\). Balanced tree: \(O(\log n)\).

2. **Complete graph.** All trees have the same cost.

3. **Planted partition model.** Correct clustering in expectation.
Algorithm for hierarchical clustering

NP-hard to minimize the cost function

\[ L(T) = \sum_{i,j} w_{ij} \cdot \#(\text{descendants of lowest common ancestor of } i, j) \]
Algorithm for hierarchical clustering

NP-hard to minimize the cost function

\[ L(T) = \sum_{i,j} w_{ij} \cdot \#(\text{descendants of lowest common ancestor of } i, j) \]

A heuristic: treat input as weighted graph \((V, E)\), and recursively split using sparse/normalized cuts (e.g. using spectral partitioning).
Algorithm for hierarchical clustering

NP-hard to minimize the cost function

$$L(T) = \sum_{i,j} w_{ij} \cdot \#(\text{descendants of lowest common ancestor of } i, j)$$

A heuristic: treat input as weighted graph \((V, E)\), and recursively split using sparse/normalized cuts (e.g. using spectral partitioning).

```plaintext
function MakeTree(V)
If |V| = 1: return leaf containing the singleton element in V
Let \((S, V \setminus S)\) be an \(\alpha\)-approximation to the sparsest cut of \(V\)
LeftTree = MakeTree(S)
RightTree = MakeTree(V \setminus S)
Return [LeftTree, RightTree]
```

This is an \((\alpha \log n)^\alpha\)-approximation to the optimal cost. Actually [Charikar-Chatziafratis, Cohen-Kanade-Mathieu]: just \(O(\alpha)\).
Algorithm for hierarchical clustering

NP-hard to minimize the cost function

\[ L(T) = \sum_{i,j} w_{ij} \cdot \# \text{(descendants of lowest common ancestor of } i, j) \]

A heuristic: treat input as weighted graph \((V, E)\), and recursively split using sparse/normalized cuts (e.g. using spectral partitioning).

function MakeTree(V)
If \(|V| = 1\): return leaf containing the singleton element in \(V\)
Let \((S, V \setminus S)\) be an \(\alpha\)-approximation to the sparsest cut of \(V\)
LeftTree = MakeTree(S)
RightTree = MakeTree(V \setminus S)
Return [LeftTree, RightTree]

This is an \((\alpha \log n)\)-approximation to the optimal cost.
Algorithm for hierarchical clustering

NP-hard to minimize the cost function

\[ L(T) = \sum_{i,j} w_{ij} \cdot \#(\text{descendants of lowest common ancestor of } i, j) \]

A heuristic: treat input as weighted graph \((V, E)\), and recursively split using sparse/normalized cuts (e.g. using spectral partitioning).

function MakeTree(V)
If \(|V| = 1\): return leaf containing the singleton element in \(V\)
Let \((S, V \setminus S)\) be an \(\alpha\)-approximation to the sparsest cut of \(V\)
LeftTree = MakeTree(S)
RightTree = MakeTree(V \setminus S)
Return [LeftTree, RightTree]

This is an \((\alpha \log n)\)-approximation to the optimal cost.

Actually [Charikar-Chatziafratis, Cohen-Kanade-Mathieu]: just \(O(\alpha)\).
Hierarchical clustering with interaction

\[ X = \text{a set of points}, \ G = \text{all hierarchies on these points}. \]

Three ingredients needed:


Feedback: triplet constraint like ({\textit{dolphin}, \textit{whale}}, \textit{zebra})

2. A cost function \( L : \mathbb{G} \rightarrow \mathbb{R} \) over hierarchies.
   We have this now.

3. An algorithm for \( \min\{L(T) : T \in \mathbb{G} \text{ satisfies constraints}\} \)
Animals with attributes, before interaction
Interaction example

Constraint: (\{tiger, collie\}, gorilla)
Interaction example

Constraint: (\{tiger, collie\}, gorilla)
Constraint: ({tiger, collie}, gorilla)
Intelligent querying

Structural QBC:
- Prior on trees: Dirichlet diffusion tree.
- Sample using Metropolis-Hastings walk with subtree-prune-and-regraft moves.
- Easy to incorporate constraints (and maintains strong connectedness of state space)
- Query every 100 iterations of the sampler.
20 Newsgroups

---

**Top Graph:**
- **Y-axis:** Data Log Likelihood
- **X-axis:** Iterations
- **Legend:**
  - Smart
  - Interleaved
  - Simple
  - Random
  - Active
  - Vanilla DDT

**Bottom Graph:**
- **Y-axis:** Triplet Distance from $T^*$
- **X-axis:** Iterations
- **Legend:**
  - Average Linkage

---
Outline

1. Interactive structure learning
2. Learning from partial correction
3. Structural query-by-committee
4. Interactive hierarchical clustering
Interesting directions

- reinforcement learning
- interactive structure learning
- co-adaptive learning
- bandits
- active learning
- preference elicitation
- imitation learning
- curriculum learning
- intelligent tutoring
- peer grading
- explanation-based learning
- teaching
- crowdsourced learning
Bibliography

• A similarity-based cost function for hierarchical clustering. STOC 2016.

• With Sharad Vikram
  Interactive Bayesian hierarchical clustering. ICML 2016.

• With Mike Luby
  Learning from partial corrections. 2017.

• With Chris Tosh