A Computational Theory of Clustering

Avrim Blum
Carnegie Mellon University

Based on work joint with Nina Balcan, Anupam Gupta, and Santosh Vempala
Point of this talk

• A new way to theoretically analyze and attack problem of clustering. Fixes a disconnect in previous formulations.

• Interesting theoretical structure. Will show results in this framework, but also many open questions too!

• Motivated by machine learning but you don’t need to know any ML for this talk.
Clustering comes up everywhere

• Given a set of documents or search results, cluster them by topic.

• Given a collection of protein sequences, cluster them by function.

• Given a set of images of people, cluster by who is in them.

• ...

...
Standard theoretical approach

- View data as nodes in weighted graph.
  - Weights based on some measure of similarity (like # keywords in common, edit distance,...)
- Pick some objective to optimize like k-median, k-means, min-sum,...
Standard theoretical approach

• View data as nodes in weighted graph.
  - Weights based on some measure of similarity (like # keywords in common, edit distance,...)
• Pick some objective to optimize like k-median, k-means, min-sum,...
  - E.g., k-median asks: find center pts $c_1, c_2, \ldots, c_k$ to minimize $\sum_x \min_i d(x,c_i)$
Standard theoretical approach

- View data as nodes in weighted graph.
  - Weights based on some measure of similarity (like # keywords in common, edit distance,...)
- Pick some objective to optimize like k-median, k-means, min-sum,...
- Develop algorithm that approximates this objective. (E.g., best known for k-median is $3+\varepsilon$ approx. Beating $1 + 2/e \approx 1.7$ is NP-hard.)

A bit of a disconnect... isn’t our real goal to get the points right??
"We couldn't get a psychiatrist, but perhaps you'd like to talk about your skin. Dr. Perry here is a dermatologist."
Well, but..

- Could say we’re implicitly hoping that any c-approx to k-median objective is \( \varepsilon \)-close pointwise to truth.
- This is an assumption about how the similarity info relates to the target clustering.
- Why not make it explicit?

Example of result: for any \( c>1 \), this assumption implies structure we can use to get \( O(\varepsilon) \)-close to truth. Even for values where getting c-approx is NP-hard!

(Even \( \varepsilon \)-close, if all clusters are “sufficiently large”.)

“Approximate clustering without the approximation”
Well, but..

• Could say we’re implicitly hoping that any $c$-approx to $k$-median objective is $\varepsilon$-close pointwise to truth.

• This is an assumption about how the similarity info relates to the target clustering.

• Why not make it explicit?

More generally: what natural properties of similarity info are sufficient to cluster well, and by what kinds of algorithms?

Give guidance to designers of similarity measures, & about what algs to use given beliefs about them.
Well, but..

- Could say we’re implicitly hoping that any $c$-approx to $k$-median objective is $\varepsilon$-close pointwise to truth.

- This is an assumption about how the similarity info relates to the target clustering.

- Why not make it explicit?

More generally: what natural properties of similarity info are sufficient to cluster well, and by what kinds of algorithms?

Analogy to learning: what concept classes are learnable and by what algorithms?
General Framework

S set of n objects. [web pages, protein seqs]

∃ ground truth clustering. $x, \ell(x)$ in \{1,\ldots,t\}. [topic, function]

Goal: clustering h of low error pointwise. $\text{err}(h) = \min_{\sigma} \Pr_{x\in S}[\sigma(h(x)) \neq \ell(x)]$

Given a pairwise similarity function $K(x,y)$ between objects.

**Question:** how related does K have to be to target to be able to cluster well?
Using “similarity” instead of distance since don’t want to require metric. Usually based on some heuristic.

- “cosine similarity” between documents (size of intersection / size of union)
- Smith-Waterman score for bio sequence data.
- In general, might not even be symmetric.

In learning, very common as kernel functions. K is a kernel if corresponds to dot-product in implicit space. \( K(x,y) = \Phi_K(x) \cdot \Phi_K(y) \). [this is why we use “K”]

Given a pairwise similarity function \( K(x,y) \) between objects.

Question: how related does \( K \) have to be to target to be able to cluster well?
What conditions on a similarity measure would be enough to allow one to cluster well?

• Using “similarity” instead of distance since don’t want to require metric. Usually based on some heuristic.
  - “cosine similarity” between documents (size of intersection / size of union)
  - Smith-Waterman score for bio sequence data.
  - In general, might not even be symmetric.

• In learning, very common as kernel functions. K is a kernel if corresponds to dot-product in implicit space. \( K(x,y) = \Phi_K(x) \cdot \Phi_K(y) \). [this is why we use “K”]

Given a pairwise similarity function \( K(x,y) \) between objects.

Question: how related does \( K \) have to be to target to be able to cluster well?
What conditions on a similarity measure would be enough to allow one to cluster well?

Will lead to something like a PAC model for clustering.

Alternatively, model data as mixture of Gaussians or other distributions. (Generative model)

Here, we don’t want to make distributional assumptions. (compare to learning a linear separator). Can view as advice to designer of similarity function.
What conditions on a similarity measure would be enough to allow one to cluster well?

Will lead to something like a PAC model for clustering.

This talk is based on two pieces of work:

- Formulation of framework and analysis of different natural properties [with Nina Balcan and Santosh Vempala]
- Looking specifically at implicit properties used in approximation algorithms [with Nina Balcan and Anupam Gupta]
What conditions on a similarity measure would be enough to allow one to cluster well?

**Protocol**

∃ ground truth clustering for S

i.e., each x in S has \( \ell(x) \) in \{1,\ldots,t\}.

**Input**

S, a similarity function K.

**Output**

Clustering of small error.

The similarity function K has to be related to the ground-truth.
What conditions on a similarity measure would be enough to allow one to cluster well?

Here is a condition that trivially works:

Suppose $K$ has property that:

- $K(x,y) > 0$ for all $x,y$ such that $l(x) = l(y)$.
- $K(x,y) < 0$ for all $x,y$ such that $l(x) \neq l(y)$.

If we have such a $K$, then clustering is easy. Now, let’s try to make this condition a little weaker....
What conditions on a similarity measure would be enough to allow one to **cluster** well?

Suppose $K$ has property that all $x$ are more similar to points $y$ in their own cluster than to any $y'$ in other clusters.

- **Still a very strong condition.**

Problem: the same $K$ can satisfy for two very different clusterings of the same data!
What conditions on a similarity measure would be enough to allow one to cluster well?

Suppose K has property that all x are more similar to points y in their own cluster than to any y' in other clusters.

• Still a very strong condition.

Problem: the same K can satisfy for two very different clusterings of the same data!

Unlike learning, you can’t even test your hypotheses!
Let’s weaken our goals a bit...

1. OK to produce a hierarchical clustering (tree) such that correct answer is apx some pruning of it.
   - E.g., in case from last slide:
     - Can view as starting at top and saying “if any of these clusters is too broad, just click and I will split it for you”
Let’s weaken our goals a bit...

1. OK to produce a hierarchical clustering (tree) such that correct answer is apx some pruning of it.
   - E.g., in case from last slide:

2. OK to output a small # of clusterings such that at least one has low error.
   - Define clustering complexity of a property as minimum list length needed to guarantee at least one clustering is ε-close to target.
Then you can start getting somewhere....

1. “all x more similar to all y in their own cluster than to any y' from any other cluster”

is sufficient to get hierarchical clustering such that target is some pruning of tree. (Kruskal’s / single-linkage works)

Proof: laminar before ⇒ laminar after.
Then you can start getting somewhere....

1. “all x more similar to all y in their own cluster than to any y’ from any other cluster”

is sufficient to get hierarchical clustering such that target is some pruning of tree. (Kruskal’s / single-linkage works)

Proof: laminar before $\Rightarrow$ laminar after.
Then you can start getting somewhere....

1. “all x more similar to all y in their own cluster than to any y' from any other cluster”

is sufficient to get hierarchical clustering such that target is some pruning of tree. (Kruskal’s / single-linkage works)

2. Weaker condition: ground truth is “stable”:

For all clusters C, C’, for all A ⊂ C, A’ ⊂ C’: A and A’ are not both more similar to each other than to rest of their own clusters.

[E.g., property 1 plus internal noise]
**Analysis for slightly simpler version**

Assume for all $C, C'$, all $A \subseteq C$, $A' \subseteq C'$, we have

$$K(A,C-A) > K(A,A'),$$

and say $K$ is symmetric.

**Algorithm:** average single-linkage

- Like Kruskal, but at each step merge pair of clusters whose average similarity is highest.

**Analysis:** (all clusters made are laminar wrt target)

- Failure iff merge $C_1, C_2$ s.t. $C_1 \subseteq C$, $C_2 \cap C = \emptyset$. 
Analysis for slightly simpler version

Assume for all $C, C'$, all $A \subset C, A' \subset C'$, we have

\[ K(A, C-A) > K(A, A'), \]

and say $K$ is symmetric.

Algorithm: average single-linkage

- Like Kruskal, but at each step merge pair of clusters whose average similarity is highest.

Analysis: (all clusters made are laminar wrt target)

- Failure iff merge $C_1, C_2$ s.t. $C_1 \subset C$, $C_2 \cap C = \emptyset$.
- But must exist $C_3 \subset C$ s.t. $K(C_1, C_3) \geq K(C_1, C-C_1)$, and $K(C_1, C-C_1) > K(C_1, C_2)$. Contradiction.
What if asymmetric?

Assume for all $C, C', A \subset C, A' \subset C'$, we have

$$K(A,C-A) > K(A,A'),$$

[Think of $K$ as “attraction”]

and say $K$ is symmetric.

Algorithm breaks down if $K$ is not symmetric:

Instead, run “Boruvka-inspired” algorithm:

- Each current cluster $C_i$ points to $\arg\max_{C_j} K(C_i,C_j)$
- Merge directed cycles. (not all components)
Relaxed conditions

Going back to:

1. “all x more similar to all y in their own cluster than to any z from any other cluster”

“strict separation”

Let’s consider a relaxed version:

1’. “Exists $S' \subseteq S$, $|S'| \geq (1-\alpha)|S|$, satisfying 1.”

Can show two interesting facts:

A. Can still efficiently get a tree of error $\alpha$. (assuming all target clusters are large).

B. This property is implied by $(2,\varepsilon)$ k-median property, for $\alpha=4\varepsilon$. (Assume metric. “more similar” = “closer”)
Relation to apx k-median assumption

• Suppose any 2-apx k-median solution must be $\varepsilon$-close to the target. (for simplicity, assume target=OPT)

• But doesn’t satisfy 1. $x$ is closer to $z$ in other cluster than to $y$ in own cluster.
  - Delete & repeat.
  - Can’t repeat $>\varepsilon n$ times.

  - Else, move all $x$’s to corresponding $z$’s cluster: at most doubles objective.

  • $d(x,c_z) \leq d(x,z) + \text{cost}(z) \leq d(x,y) + \text{cost}(z) \leq \text{cost}(x) + \text{cost}(y) + \text{cost}(z)$. 
Relation to apx k-median assumption

• \((2, \varepsilon)\) k-median property \(\Rightarrow\) O(\(\varepsilon\))-relaxed separation property.

• O(\(\varepsilon\))-relaxed separation property \(\Rightarrow\) produce tree s.t. some pruning is O(\(\varepsilon\))-close. (assuming all target clusters are large).

• Can actually directly go from \((c, \varepsilon)\) k-median property to single O(\(\varepsilon\))-close clustering, for any \(c>1\). (\(\varepsilon\)-close if all target clusters are large).
  - Also for k-means, min-sum.
How can we use the \((c, \varepsilon)\) k-median property to cluster, without solving k-median?
Clustering from \((c, \varepsilon)\) k-median prop

- Suppose any \(c\)-apx k-median solution must be \(\varepsilon\)-close to the target. (and for simplicity say target is k-median opt, & all cluster sizes \(> 2\varepsilon n\))
- For any \(x\), let \(w(x) = \text{dist to own center, } w_2(x) = \text{dist to } 2^{\text{nd}}-\text{closest center.}\)
- Let \(w_{\text{avg}} = \text{avg}_x w(x)\).
- Then:
  - At most \(\varepsilon n\) pts can have \(w_2(x) < (c-1)w_{\text{avg}}/\varepsilon\).
  - At most \(5\varepsilon n/(c-1)\) pts can have \(w(x) \geq (c-1)w_{\text{avg}}/5\varepsilon\).
- All the rest (the good pts) have a big gap.
Clustering from \((c, \varepsilon)\) k-median prop

- At most \(\varepsilon n\) pts can have \(w_2(x) < (c-1)w_{\text{avg}}/\varepsilon\).
- At most \(5\varepsilon n/(c-1)\) pts can have \(w(x) \geq (c-1)w_{\text{avg}}/5\varepsilon\).

• All the rest (the good pts) have a big gap.
Clustering from \((c, \varepsilon) k\)-median prop

- At most \(\varepsilon n\) pts can have \(w_2(x) < (c-1)w_{\text{avg}}/\varepsilon\).
- At most \(5\varepsilon n/(c-1)\) pts can have \(w(x) \geq (c-1)w_{\text{avg}}/5\varepsilon\).

- All the rest (the good pts) have a big gap.
- Define critical distance \(d_{\text{crit}} = (c-1)w_{\text{avg}}/5\varepsilon\).
- So, a \(1-O(\varepsilon)\) fraction of pts look like:
Clustering from \((c, \varepsilon)\) k-median prop

- So if we define a graph \(G\) connecting any two pts within distance \(\leq 2d_{crit}\), then:
  - Good pts within cluster form a clique
  - Good pts in different clusters have no common nbrs
- So, a \(1-O(\varepsilon)\) fraction of pts look like:
Clustering from \((c, \varepsilon)\) k-median prop

- So if we define a graph \(G\) connecting any two pts within distance \(\leq 2d_{\text{crit}}\), then:
  - Good pts within cluster form a clique
  - Good pts in different clusters have no common nbrs
- So, the world now looks like:
Clustering from \((c,\varepsilon)\) k-median prop

- If all clusters have size \(> 2b+1\), where \(b = \# \text{ bad pts} = O(\varepsilon n/(c-1))\), then:
  - Create graph \(H\) where connect \(x, y\) if share \(> b\) nbrs in common in \(G\).
  - Output \(k\) largest components in \(H\).

- So, the world now looks like:
Clustering from \((c, \varepsilon)\) k-median prop

- If clusters not so large, then need to be a bit more careful but can still get error \(O(\varepsilon)\).
- E.g., now could have some clusters dominated by bad pts....

- So, the world now looks like:
\(O(\varepsilon)\)-close \(\rightarrow\) \(\varepsilon\)-close

- Back to the large-cluster case: can actually get \(\varepsilon\)-close. (for any \(c>1\), but "large" depends on \(c\)).
- Idea: Really two kinds of bad pts.
  - At most \(\varepsilon n\) "confused": \(w_2(x) - w(x) < (c-1)w_{\text{avg}} / \varepsilon\).
  - Rest not confused, just far: \(w(x) \geq (c-1)w_{\text{avg}} / 5\varepsilon\).
- Can recover the non-confused ones...
\[ \Omega(\varepsilon) \text{-close} \rightarrow \varepsilon \text{-close} \]

- Back to the large-cluster case: can actually get \( \varepsilon \)-close. (for any \( c > 1 \), but “large” depends on \( c \)).
- Given output \( C' \) from alg so far, reclassify each \( x \) into cluster of lowest median distance
  - Median is controlled by good pts, which will pull the non-confused points in the right direction.
<table>
<thead>
<tr>
<th>Property</th>
<th>Model, Algorithm</th>
<th>Clustering Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strict Separation</td>
<td>Hierarchical, Linkage based</td>
<td>$\Theta(2^k)$</td>
</tr>
<tr>
<td>Stability, all subsets.</td>
<td>Hierarchical, Linkage based</td>
<td>$\Theta(2^k)$</td>
</tr>
<tr>
<td>(Weak, Strong, etc)</td>
<td>Hierarchical, Linkage based</td>
<td></td>
</tr>
<tr>
<td>Average Attraction</td>
<td>List, Sampling based &amp; NN</td>
<td>$[k^{\Omega(k/\gamma)}, k^{O(k/\gamma^2)}]$</td>
</tr>
<tr>
<td>(Weighted)</td>
<td>Hierarchical, list and refine (running time $k^{O(k/\gamma^2)}$)</td>
<td>$\Theta(2^k)$</td>
</tr>
<tr>
<td>Stability of large subsets</td>
<td>Hierarchical, list and refine</td>
<td></td>
</tr>
<tr>
<td>relaxed separation</td>
<td>Hierarchical, list and refine</td>
<td>$\Theta(2^k)$</td>
</tr>
<tr>
<td>$(c, \varepsilon)$ k-median</td>
<td>Greedy + refining</td>
<td>1</td>
</tr>
</tbody>
</table>
How about weaker conditions?

What if just have: all (most) $x$ satisfy
\[ E_{x' \in C(x)}[K(x, x')] > E_{x' \in C'}[K(x, x')] + \gamma \quad (\forall C' \neq C(x)) \]
Not sufficient for hierarchy.

But can produce a small list of clusterings s.t. at least one is good:

- Upper bound $k^{O(k/\gamma^2 \ldots)}$. [doesn’t depend on $n$]
- Lower bound $\approx k^{\Omega(k/\gamma)}$.

Upper and lower bounds on “clustering complexity” of this property.
Can also analyze inductive setting

• View $S$ as just a random subset from larger instance space $X$.
• Property holds wrt $X$.
• Given $S$, want to:
  - A. produce good clustering of $S$.
  - B. be able to insert new points in streaming manner as they arrive.
Can also analyze inductive setting

Assume for all $C, C'$, all $A \subseteq C, A' \subseteq C'$, we have

$$K(A, C - A) > K(A, A') + \gamma$$

Draw sample $S$:

- Need to argue that whp $K(A, S \cap C - A)$ is good estimate of $K(A, C - A)$ for all $A \subseteq S$ for suff $\log S$.
- A sample cplx type argument using “regularity” type results of [AFKK].

Once $S$ is hierarchically partitioned, can insert new points as they arrive.
Like a PAC model for clustering

- PAC learning model: basic object of study is the concept class (a set of functions). Look at which are learnable and by what algs.
- In our case, basic object of study is a property: like a data-dependent concept class. Want to know which allow clustering and by what algs.
Conclusions

What properties of a similarity function are sufficient for it to be useful for clustering?

- Target function as ground truth rather than graph as ground truth. Graph is just produced using a heuristic!
- To get interesting theory, helps to relax what we mean by “useful”.
- Can view as a kind of PAC model for clustering.
- A lot of interesting directions to explore.
Conclusions

- Natural properties (relations between sim fn and target) that motivate spectral methods?
- Efficient algorithms for other properties? E.g., “stability of large subsets”, (c,ε) property for other clustering objectives.
- Other notions of “useful”.
- A lot of interesting directions to explore.
  - Produce a small DAG instead of a tree?
  - Others based on different kinds of feedback?
- ...