Approximate Clustering without the Approximation Algorithm & a new angle on Optimization

Avrim Blum
Carnegie Mellon University
3 great areas that go great together
Approximate Clustering without the Approximation Algorithm & a new angle on Optimization

Avrim Blum
Carnegie Mellon University

Based on work joint with Nina Balcan, Pranjal Awasthi, Anupam Gupta, Or Sheffet, and Santosh Vempala

Indo-US Symposium on Machine Learning, Game Theory and Optimization 2010
Theme of this talk

- What if worst-case instances are hard, even to apx, but don’t want to make distributional assumptions?

- Often there are assumptions will need to make anyway when it comes time to use your solution.

- If make these explicit up front, can give alg more to work with, and sometimes get around computational hardness barriers.
Clustering
Clustering comes up in many places

- 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 approach

• 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 approach

• Come up with some measure of similarity (like # keywords in common, edit distance,...)
• Use to view data as nodes in weighted graph
• Run clustering algorithm on graph. Hope it gives a good output.
Standard theoretical approach

• Come up with some measure of similarity (like # keywords in common, edit distance,...)
• Use to view data as nodes in weighted graph
• Pick some objective to optimize like k-median, k-means, min-sum,...
Standard theoretical approach

- Come up with some measure of similarity (like # keywords in common, edit distance,...)
- Use to view data as nodes in weighted graph
- 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 $\sum_x \min_i d(x,c_i)$
  - k-means asks: find $c_1, c_2, \ldots, c_k$ to minimize $\sum_x \min_i d^2(x,c_i)$
Standard theoretical approach

• Come up with some measure of similarity (like # keywords in common, edit distance,...)
• Use to view data as nodes in weighted graph
• Pick some objective to optimize like k-median, k-means, min-sum,...
• Develop algorithm to (approx) optimize this objective. (E.g., best known for k-median is $3+\varepsilon$ approx [AGKMMP04]. Beating $1 + 1/e$ is NP-hard [JMS02].)

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 in error to truth.
- This is an assumption about how the distance measure relates to the target clustering.
- Why not make it explicit?

Example of result: for any $c>1$, this property implies structure we can use to get $O(\varepsilon)$ error. Even for values where getting c-approx is NP-hard! (Even $\varepsilon$ error, if all clusters are “sufficiently large”.) As well as if we could approximate to NP-hard value!
Could say we’re implicitly hoping that any $c$-approx to $k$-median objective is $\varepsilon$-close in error to truth.

This is an assumption about how the distance measure relates to the target clustering.

Why not make it explicit?

More generally: have one objective you can measure, and a different one you care about.

Implicitly assuming they are related.

Let’s make it explicit.
Formal Setup

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

Ground truth clustering. $C_1^*, C_2^*, \ldots, C_k^*$. [true clustering by topic]

Goal: clustering $C_1, \ldots, C_k$ of low error. $error(C') = \min_{\sigma \in S_k} \frac{1}{n} \sum_{i=1}^{k} |C_i^* - C_{\sigma(i)}|$
Set $S$ of $n$ objects.  

Ground truth clustering.  $C_1^*, C_2^*, \ldots, C_k^*$.  

Goal: clustering $C_1, \ldots, C_k$ of low error.  

$$\text{error}(C') = \min_{\sigma \in S_k} \frac{1}{n} \sum_{i=1}^{k} |C_i^* - C_{\sigma(i)}|$$

Given a distance metric $d(x,y)$ on objects.

Satisfies $(c, \varepsilon)$-approximation-stability for objective $\Phi$ if any $c$-approximation to $\Phi$ has error at most $\varepsilon$.  

[web pages, protein seqs] [true clustering by topic]
Approximation-stability

- Instance is \((c, \varepsilon)\)-apx-stable for objective \(\Phi\): any \(c\)-approximation to \(\Phi\) has error \(\leq \varepsilon\).
- Focus on \(\Phi = \text{“k-median objective”}\).
  - (Also results for k-means, min-sum)

How are we going to use this to cluster well if we don’t know how to get a \(c\)-approximation?
\((c, \varepsilon)\) k-median stability

We're assuming any \(c\)-apx k-median solution must be \(\varepsilon\)-close to the target, \(c > 1\).

Two approaches that don't work:

1. Hope that \((1.1, \varepsilon)\) stable \(\Rightarrow\) \((3, O(\varepsilon))\) stable
   - But for any \(c_1 < c_2\) can construct dataset and target s.t. all \(c_1\) apx to k-median have error < \(\varepsilon\), but exists \(c_2\) apx that has error 0.49.
(c,ε) k-median stability

We’re assuming any c-apx k-median solution must be ε-close to the target, c>1.

Two approaches that don’t work:

1. Hope that (1.1,ε) stable ⇒ (3,O(ε)) stable
2. Hope that c-apx is easy under (c,ε) stability
   - Unfortunately, c-apx is as hard as in general case. (if min cluster size small vs εn - will get back to this...)

Instead, want to skip this proxy, just use properties implied by assumption.
Clustering from \((c, \varepsilon)\) k-median stability

- 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 \(w_{\text{avg}} = \text{avg}_x w(x)\). \([\text{OPT} = n \cdot w_{\text{avg}}]\)
- 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 stability

- Define critical distance \(d_{\text{crit}} = (c-1)w_{\text{avg}} / 5\varepsilon\).
- So, a \(1 - O(\varepsilon)\) fraction of pts look like:
  - At most \(\varepsilon n\) pts can have \(w_2(x) < (c-1)w_{\text{avg}} / \varepsilon\).
  - At most \(5n/(c-1)\) 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 stability

- 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, a \(1 - O(\varepsilon)\) fraction of pts look like:
Clustering from \((c, \varepsilon)\) \(k\)-median stability

- 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 stability

- If furthermore 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 \(H\). (only makes mistakes on bad points)

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

If clusters not so large, then need to be a bit more careful but can still get error \(O(\varepsilon/(c-1))\).

Could have some clusters dominated by bad pts...

Actually, just need to modify algorithm a bit, but analysis more involved.
$O(\epsilon)$-close $\Rightarrow$ $\epsilon$-close

- Back to the large-cluster case: can actually get $\epsilon$-close. (for any $c>1$, but “large” depends on $c$).

- Idea: Really two kinds of bad pts.
  - At most $\epsilon n$ “confused”: $w_2(x)-w(x) < (c-1)w_{avg}/\epsilon$.
  - Rest not confused, just far: $w(x) \geq (c-1)w_{avg}/5\epsilon$.

- Can recover the non-confused ones...
$O(\varepsilon)$-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...

\[ \leq d_{\text{crit}} \]

\[ w(x) \]

\[ w_2(x) \]

\[ w_2(x) - w(x) \geq 5 d_{\text{crit}} \]

\[ \text{non-confused bad pt} \]
$O(\varepsilon)$-close $\Rightarrow$ $\varepsilon$-close

- Back to the large-cluster case: can actually get $\varepsilon$-close. (for any $c > 1$, but "large" depends on $c$).
  - 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.

\[
\text{w}(x) - \text{w}_2(x) \geq 5 \text{d}_{crit}
\]
**$O(\varepsilon)$-close $\Rightarrow \varepsilon$-close**

- Back to the large-cluster case: can actually get $\varepsilon$-close. (for any $c>1$, but “large” depends on $c$).
  - 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.

A bit like 2-rounds of k-means/Lloyd’s algorithm
Stepping back...

- Assumption that any $c$-apx to $k$-median is $\varepsilon$-close allows us to get $\varepsilon$-close (for large clusters) or $O(\varepsilon)$-close (for general cluster sizes).

- Can also get similar guarantees for $k$-means, min-sum objectives.

See [Balcan-Braverman09] for best results.
Stepping back...

- Assumption that any c-apx to k-median is $\varepsilon$-close allows us to get $\varepsilon$-close (for large clusters) or $O(\varepsilon)$-close (for general cluster sizes)

Useful in practice?

- [Voevodski-Balcan-Roglin-Teng-Xia UAI’10]
  - Show how algorithm can be adapted to be very fast in setting of 1-vs-all queries.
  - Apply to protein sequence clustering problems (Pfam, SCOP databases)
  - Fast and high accuracy.
Extensions

[Awasthi-B-Sheffet’10]

All $\varepsilon$-far solutions are not $c$-approximations

All $k-1$ clusterings are not $c$-approximations

(strictly weaker condition in the “large clusters” case)

Under this condition, can get a PTAS: $1+\alpha$ apx in polynomial time (exponential in $1/\alpha$, $1/(c-1)$)
Extensions

[Awasthi-B-Sheffet'10]

- All \( \varepsilon \)-far solutions are not \( c \)-approximations
- All \( k-1 \) clusterings are not \( c \)-approximations

Implications:
- Under approx stability, get exactly \( \varepsilon \)-close in “large clusters” case for \( k \)-means too.
- Only need clusters of size \( \geq 2\varepsilon n \) vs \( \Omega(\varepsilon n/(c-1)) \).
Other problems?

One proposal: Nash equilibria.

So, what’s a Nash equilibrium?
Nash equilibrium

- Set of (randomized) strategies for players in a multiagent interaction (i.e., game) such that no one has any incentive to deviate.
- Appears to be computationally hard to find even in 2-player n-action games.
- A lot of interest in whether we can compute \( \varepsilon \)-equilibria efficiently (best general alg: time \( n^{O((\log n)/\varepsilon^2)} \) )
Nash equilibrium

• Why do we want to find an (apx) equilibrium?
  – One reason: predict long-term behavior (e.g., proposing design of new system), which we believe will be at (apx) equilibrium.

• In that case, natural to focus on cases where all (apx) equilibria are close to each other.
Nash equilibrium

So, what can we say?

• In that case, natural to focus on cases where all (apx) equilibria are close to each other.
Nash equilibrium

• In that case, natural to focus on cases where all (apx) equilibria are close to each other.
Nash equilibrium

- Get poly-time for all $\Delta = O(\epsilon)$?
Nash equilibrium

- Get poly-time for all $\Delta = O(\epsilon)$?
- Recent work of [Balcan-Braverman]:
  - $\Delta = O(\epsilon^{1/4})$ as hard as general case for poly-time.
  - Extensions to reversed-quantifier condition.
  - Connections to perturbation-stability.
Other problems where approach might make sense?

A few ideas:

- Sparsest cut
  - Best apx is $O((\log n)^{1/2})$ [ARV]
  - What if assume any 10-apx has error $\leq \varepsilon$?

Minimize $e(A,B)/(|A|*|B|)$
Other problems where approach might make sense?

A few ideas:

- Evolutionary tree reconstruction
  - Often posed as a Steiner-tree-like problem.
  - To have confidence in solution, would hope that near-optimal answers are close in structure.
  - Brings up related Q: what if only part of input is stable. Maybe can identify & output solution for that?
Summary

For clustering, can say “if data has the property that a 1.1 apx to [pick one: k-median, k-means, min-sum] would be sufficient to cluster well, then we can cluster well” ...even though you might think NP-hardness results for approximating these objectives would preclude this.

Suggests an approach to other optimization problems where **objective function** may be a proxy for something else, or esp care about **stable** instances

- Nash equilibria
- Market equilibria?
- Sparsest cut? Evolutionary trees?

Many interesting problems to explore!