Core-set: Summary and Open Problems

Hu Ding

Computer Science and Engineering, Michigan State University
Clustering in High Dimension

- K-means/median clustering: use core-set and JL-transform, reduce the number of points from $n$ to roughly $\tilde{O}(k^2(\log n)^2/\epsilon^4)$. “On Coresets for k-Median and k-Means Clustering in Metric and Euclidean Spaces and Their Applications” by Chen.

- Streaming model: a very similar idea to dynamically updating hash table:
  - construct a sequence of buckets with doubling capacities, where only the first one stores the new arriving data, and others store core-sets.
  - The basic idea in Chen’s paper is partition+uniform sampling. It was improved by a more sophisticated adaptive sampling strategy. “A unified framework for approximating and clustering data” by Feldman and Langberg.
Clustering in High Dimension

- K-means/median clustering: use core-set and JL-transform, reduce the number of points from $n$ to roughly $\tilde{O}(k^2(\log n)^2/\epsilon^4)$. “On Coresets for k-Median and k-Means Clustering in Metric and Euclidean Spaces and Their Applications” by Chen.

- Streaming model: a very similar idea to dynamically updating hash table:
Clustering in High Dimension

- K-means/median clustering: use core-set and JL-transform, reduce the number of points from $n$ to roughly $\tilde{O}(k^2(\log n)^2/\epsilon^4)$. “On Coresets for k-Median and k-Means Clustering in Metric and Euclidean Spaces and Their Applications” by Chen.

- Streaming model: a very similar idea to dynamically updating hash table:
  - construct a sequence of buckets with doubling capacities, where only the first one stores the new arriving data, and others store core-sets.
Clustering in High Dimension

- **K-means/median clustering**: use core-set and JL-transform, reduce the number of points from \( n \) to roughly \( \tilde{O}(k^2(\log n)^2/\epsilon^4) \). “On Coresets for k-Median and k-Means Clustering in Metric and Euclidean Spaces and Their Applications” by Chen.

- **Streaming model**: a very similar idea to dynamically updating hash table:
  - construct a sequence of buckets with doubling capacities, where only the first one stores the new arriving data, and others store core-sets.

- The basic idea in Chen’s paper is **partition + uniform sampling**. It was improved by a more sophisticated **adaptive sampling** strategy. “A unified framework for approximating and clustering data” by Feldman and Langberg.
Projective clustering: instead of $k$ cluster centers, it seeks $k$ $j$-dimensional flats (shifted subspaces) to minimize the clustering cost.

This problem is extremely hard, imagine the generalization for both k-means/median and PCA in high dimension.

$(1 + \epsilon, \log n)$-bicriteria approximation: resulting in at most $(1 + \epsilon)k\log n$ flats.

"Bi-criteria linear-time approximations for generalized k-mean/median/center" by Feldman et al.

The basic idea is about uniform sampling + peeling.

If $k$ is constant, there exists $(1 + \epsilon)$-approximation.

"A unified framework for approximating and clustering data" by Feldman and Langberg.
Clustering in High Dimension (cont.)

- Projective clustering: instead of $k$ cluster centers, it seeks $k$ $j$-dimensional flats (shifted subspaces) to minimize the clustering cost.
  - This problem is extremely hard, imagine the generalization for both k-means/median and PCA in high dimension.
• Projective clustering: instead of $k$ cluster centers, it seeks $k$ $j$-dimensional flats (shifted subspaces) to minimize the clustering cost.

• This problem is extremely hard, imagine the generalization for both k-means/median and PCA in high dimension.

• $(1 + \epsilon, \log n)$-bicriteria approximation: resulting in at most $(1 + \epsilon)$ times the optimal clustering cost and $k \log n$ flats. “Bi-criteria linear-time approximations for generalized k-mean/median/center” by Feldman et al.
• Projective clustering: instead of $k$ cluster centers, it seeks $k$ $j$-dimensional flats (shifted subspaces) to minimize the clustering cost.

• This problem is extremely hard, imagine the generalization for both $k$-means/median and PCA in high dimension.

• $(1 + \epsilon, \log n)$-bicriteria approximation: resulting in at most $(1 + \epsilon)$ times the optimal clustering cost and $k \log n$ flats.

  “Bi-criteria linear-time approximations for generalized $k$-mean/median/center” by Feldman et al.

• The basic idea is about uniform sampling + peeling.
• Projective clustering: instead of $k$ cluster centers, it seeks $k$ $j$-dimensional flats (shifted subspaces) to minimize the clustering cost.
  
  • This problem is extremely hard, imagine the generalization for both k-means/median and PCA in high dimension.
  
  • $(1 + \epsilon, \log n)$-bicriteria approximation: resulting in at most $(1 + \epsilon)$ times the optimal clustering cost and $k \log n$ flats.
  
  “Bi-criteria linear-time approximations for generalized k-mean/median/center” by Feldman et al.
  
  • The basic idea is about uniform sampling + peeling.
  
  • If $k$ is constant, there exists $(1 + \epsilon)$-approximation. “A unified framework for approximating and clustering data” by Feldman and Langberg.
• Stochastic model for uncertain and noisy data.
• Stochastic model for **uncertain** and **noisy** data.
  
  1. Existential uncertainty model: each data point has a fixed location but with an **existential probability**.
• Stochastic model for **uncertain** and **noisy** data.
  ① Existential uncertainty model: each data point has a fixed location but with an existential probability.
  ② Locational uncertainty model: each data point has a set of location candidates.
Stochastic model for uncertain and noisy data.

1. Existential uncertainty model: each data point has a fixed location but with an existential probability.
2. Locational uncertainty model: each data point has a set of location candidates.

Current state-of-the-art techniques can only handle low dimensional case. “Stochastic k-Center and j-Flat-Center Problems” by Huang and Li; “epsilon-Kernel Coresets for Stochastic Points” by Huang et al.
SVM And Sparse Representation

• SVM is equivalent to a polytope distance problem, and we have learned Gilbert algorithm for polytope distance. “Coresets for polytope distance” by Gartner and Jaggi.
SVM And Sparse Representation

- SVM is equivalent to a polytope distance problem, and we have learned Gilbert algorithm for polytope distance. "Coresets for polytope distance" by Gartner and Jaggi.
- But what about non-separable case? "Random Gradient Descent Tree: A Combinatorial Approach for SVM with Outliers" by Ding and Xu.
Our Main Ideas

Key Observation: It is not necessary to select the single point in a deterministic way at each step of Gilbert algorithm.

Gradient Descent + Randomness
Our Main Ideas

**Key Observation:** It is not necessary to select the single point in a *deterministic* way at each step of Gilbert algorithm.

**Gradient Descent + Randomness**

- Preserve fast convergence via *gradient descent*.
- Explicitly remove the influence of outliers via *randomness*. 
Our Main Ideas

Gilbert Algorithm

Our idea for the case with outliers

• Induce a data structure called Random Gradient Descent-Tree.

• Quality guarantee: $(1 - \epsilon)$-approximation with respect to separating margin.
Our Main Ideas

Gilbert Algorithm  
Our idea for the case with outliers

\[ x_0 \]
Our Main Ideas

Gilbert Algorithm

Our idea for the case with outliers

\[ \mathbf{x}_0 \]
Our Main Ideas

Gilbert Algorithm

Our idea for the case with outliers

• Induce a data structure called Random Gradient Descent-Tree.

• Quality guarantee: $(1 - \epsilon)$-approximation with respect to separating margin.
Our Main Ideas

Gilbert Algorithm

Our idea for the case with outliers

- Induce a data structure called Random Gradient Descent-Tree.
- Quality guarantee: $(1 - \epsilon)$-approximation with respect to separating margin.
Our Main Ideas

- Induce a data structure called Random Gradient Descent-Tree.
- Quality guarantee: $(1 - \epsilon)$-approximation with respect to separating margin.
Our Main Ideas

Gilbert Algorithm

Our idea for the case with outliers

- Induce a data structure called Random Gradient Descent-Tree.
- Quality guarantee: $(1 - \epsilon)$-approximation with respect to separating margin.
Our Main Ideas

Gilbert Algorithm

Our idea for the case with outliers

- Induce a data structure called Random Gradient Descent-Tree.
- Quality guarantee: \((1 - \epsilon)\)-approximation with respect to separating margin.
Our Main Ideas

Gilbert Algorithm

Our idea for the case with outliers

- Induce a data structure called Random Gradient Descent-Tree.
- Quality guarantee: $(1 - \epsilon)$-approximation with respect to separating margin.
Our Main Ideas

Gilbert Algorithm

Our idea for the case with outliers

- Induce a data structure called **Random Gradient Descent-Tree**.
- **Quality guarantee:** $(1 - \epsilon)$-approximation with respect to separating margin.
How to Achieve Sub-linear Time Complexity

• From linear to sub-linear time:
How to Achieve Sub-linear Time Complexity

- From linear to sub-linear time:

  - Top $k$ selection
  - Sampling
How to Achieve Sub-linear Time Complexity

- From linear to sub-linear time:

  *Top k selection*  
  *Sampling*  
  *Sampling*  
  *Top k' selection*
How to Achieve Sub-linear Time Complexity

- From linear to sub-linear time:

  - Top $k$ selection
  - Sampling

  Sampling → Top $k'$ selection

  $x_0$
How to Achieve Sub-linear Space Complexity

- Build the data structure via breadth-first search.
- Only two levels are needed to store $\Rightarrow$ sub-linear extra space complexity.
Actually Gilbert algorithm is a special case of Frank-Wolfe algorithm. "Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm" by Clarkson.
• Actually Gilbert algorithm is a special case of Frank-Wolfe algorithm. “Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm” by Clarkson.

• Another typical case of Frank-Wolfe algorithm is computing Minimum Enclosing Ball in high dimension.
• Actually Gilbert algorithm is a special case of Frank-Wolfe algorithm. “Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm” by Clarkson.

• Another typical case of Frank-Wolfe algorithm is computing Minimum Enclosing Ball in high dimension.

• The result of Frank-Wolfe algorithm usually is a sparse representation, with potential applications in dictionary learning.
Other Applications of Core-set

- Compress GPS/LiDAR data, and help data mining on GPS datasets with applications in traffic management, autopilot, etc. “The Single Pixel GPS: Learning Big Data Signals from Tiny Coresets” by Feldman et al.
- Large scale training for Gaussian Mixture Model (GMM). “Scalable Training of Mixture Models via Coresets” by Feldman et al.
- Speedup alignment/registration for computer vision and robotics. “Low-cost and Faster Tracking Systems Using Core-sets for Pose-Estimation” by Nasser et al.
Other Applications of Core-set

- Compress GPS/LiDAR data, and help data mining on GPS datasets with applications in traffic management, autopilot, etc. “The Single Pixel GPS: Learning Big Data Signals from Tiny Coresets” by Feldman et al.
Other Applications of Core-set

- Compress GPS/LiDAR data, and help data mining on GPS datasets with applications in traffic management, autopilot, etc. “The Single Pixel GPS: Learning Big Data Signals from Tiny Coresets” by Feldman et al.


- Large scale training for Gaussian Mixture Model (GMM). “Scalable Training of Mixture Models via Coresets” by Feldman et al.
Other Applications of Core-set

- Compress GPS/LiDAR data, and help data mining on GPS datasets with applications in traffic management, autopilot, etc. “The Single Pixel GPS: Learning Big Data Signals from Tiny Coresets” by Feldman et al.
- Large scale training for Gaussian Mixture Model (GMM). “Scalable Training of Mixture Models via Coresets” by Feldman et al.
- Speedup alignment/registration for computer vision and robotics. “Low-cost and Faster Tracking Systems Using Core-sets for Pose-Estimation” by Nasser et al.
Open Problems

- **Constrained** Core-set, such as the applications in social network (e.g., users or products may have mutual conflict).
Open Problems

- **Constrained** Core-set, such as the applications in social network (e.g., users or products may have mutual conflict).
- Stochastic model in high dimension.
Open Problems

- **Constrained** Core-set, such as the applications in social network (e.g., users or products may have mutual conflict).
- Stochastic model in high dimension.
- Algorithm for computing Core-set via GPU (highly parallel).
Open Problems

- **Constrained** Core-set, such as the applications in social network (e.g., users or products may have mutual conflict).
- Stochastic model in high dimension.
- Algorithm for computing Core-set via GPU (highly parallel).
- Any other new applications, or theoretical improvement.
Thank You!

Any Question?