A Faster Sampling Algorithm for Spherical k-means

Rameshwar Pratap, Anup Deshmukh*, Pratheeksha Nair*, Tarun Dutt*

*International Institute of Information Technology – Bangalore, India

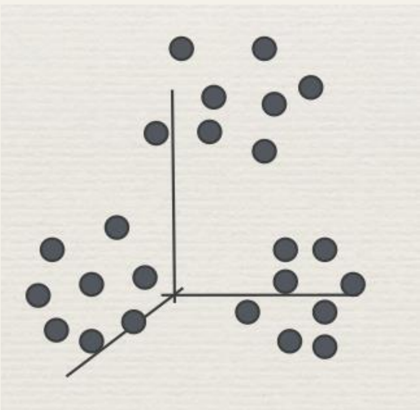
Presenter – Anup Deshmukh

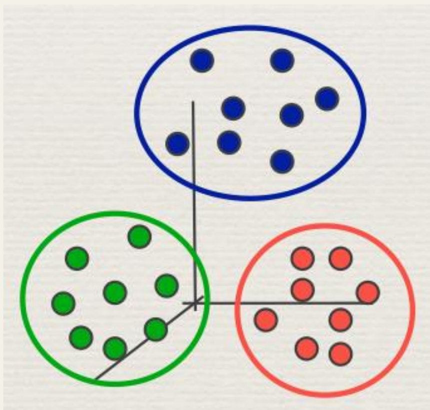
Introduction to the World of Clustering Algorithms

Partitioning unlabeled data objects(examples) into disjoint clusters such that:

- data objects within a cluster are very similar
- data objects in different clusters are very different

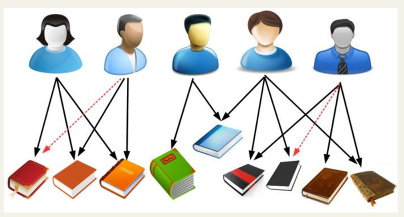
Clustering algorithms discover new categories in an unsupervised man ner(unlabeled data)





Importance and Applications

- Pre-processing for fast search - Text Clustering
- Summarizing news articles along with headlines
- Collaborative filtering(CF) algorithms in Recommender Systems
- CF makes recommendations based on interactions between users and items.



- To illustrate, consider an example of Netflix, which has over 8, 000 titles in the U.S. content library and around 130 million streaming subscribers worldwide.
- In real world computational environment, calculating similarities between these large number of users or items then becomes difficult!

Data Representation

- Text document corpuses are represented in the vector space model as matrices where each row corresponds to a document and each column represents a topic/word in the document

Data mining items	Linear Algebra Items	Neutral Items
Text	Linear	Analysis
Mining	Algebra	Application
Clustering	Matrix	Algorithm
Classification	Vector	
Retrieval	Space	
Information		

Document vector

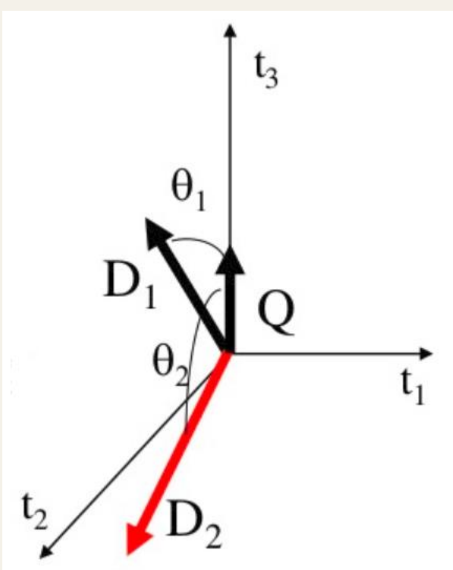
Word vector

10	0	**5**	0	0
0	0	**0**	4	0
0	0	**0**	0	0
6	15	**14**	0	7
0	0	**0**	13	0
0	**0**	**6**	**0**	**3**
0	9	**0**	0	0
0	0	**0**	5	0
0	1	**0**	0	0

- The similar Bag of Words approach can be used to represent user-rating matrices in Collaborative Filtering(CF)!
- Once we represent the data, next important step is to measure similarity between two documents

$$\cos(D_i, D_j) = \frac{\langle D_i, D_j \rangle}{||D_i||.||D_j||}$$

- Cosine similarity is proven to be appropriate for determining similarity between documents



Problem Statement

- Develop a faster algorithm which assigns data points to *k* clusters while maintaining approximation to the optimal clustering cost
- Goal: **Find a set of cluster centers that maximizes the cosine similarity between each point and its closest cluster center**

Baseline algorithms

SPKM:

- In this method we choose *k* arbitrary initial centers uniformly randomly.
- Then EM-type local search is performed till convergence
- SPKM is a simple algorithm but
- It takes many Lloyd's iterations to converge and
- It is sensitive to initialization (may get stuck in a local optimum)

SPKM++:

- All data points are first normalized to unit norm
- In this method main idea is to spread out the initial chosen cluster centers
- First center is chosen randomly
- Remaining k-1 points chosen from the following distribution

$$\frac{(1-\cos(x', C))}{\phi_{\mathcal{X}}(C)} \propto (1-\cos(x', C))$$

- Provides better clustering quality compared to SPKM but
- Needs *k* passes over the data as opposed to one in SPKM and
- For large datasets, *k* is typically large and hence SPKM++ is not scalable

Proposed algorithm - SPKM MCMC

In this work, we propose a Markov chain based sampling algorithm that

- takes only one pass over the data for choosing *k* initial seeds and
- gives close to optimal clustering similar to SPKM++

SPKM-MCMC reduces the complexity by approximating angular-sampling, i.e, it uses a sampling method where sampling probabilities *q(x)* are close to the underlying angular sampling distribution *p(x)*.

Simply put, the theoretical guarantee on the clustering cost of SPKM-MCMC algorithm is close to SPKM++ while simultaneously achieving a significant speed-up in the seeding time

Input: Data set $\mathcal{X}$, chain-length m, number of clusters k.
Output: A set of initial cluster centers (seeding points) $\mathbf{C} = \{\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_k}\}$.

Preprocessing step:
$\mathbf{c_1} \leftarrow$ a vector sampled uniformly at random from $\mathcal{X}$.
for $x \in \mathcal{X}$ **do**
  $q(\mathbf{x}|\mathbf{c_1}) = \frac{\mathrm{d}(\mathbf{x},\mathbf{c_1})}{2\sum_{\mathbf{x}' \in \mathcal{X}} \mathrm{d}(\mathbf{x}',\mathbf{c_1})} + \frac{1}{2|\mathcal{X}|}$
end

Main algorithm:
$\mathbf{C} \leftarrow \{\mathbf{c_1}\}$
for $i = 2, 3, \ldots, k$ **do**
  $x \leftarrow$ point sampled from $q(x)$
  $d_x \leftarrow \mathrm{d}(x, \mathbf{C})$
  for $j = 2, 3, \ldots, m$ **do**
    $y \leftarrow$ point sampled from $q(y)$
    $d_y \leftarrow \mathrm{d}(y, \mathbf{C})$
    if $\frac{d_y q(x)}{d_x q(y)} > \mathrm{Unif}(0, 1)$ **then**
      $x \leftarrow y, d_x \leftarrow d_y$
    end
  end
  $\mathbf{C} \leftarrow \mathbf{C} \cup \{x\}$
end

To summarize, we now have a faster sampling algorithm which maintains almost the same approximation as of the clustering quality of SPKM++.

	SPKM++	SPKM-MCMC
Seeding step	Requires *k* passes over the data.	Requires one pass over the data
Clustering cost	*O(log(k))* approximation	*Additive error due to Markov approximation*

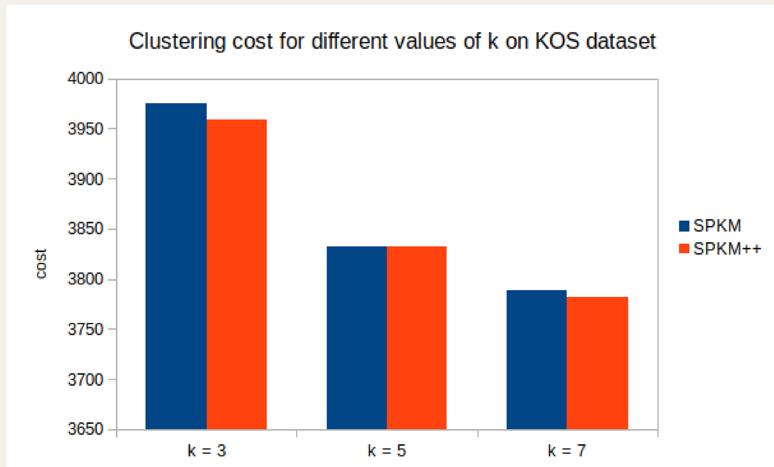
Experiments

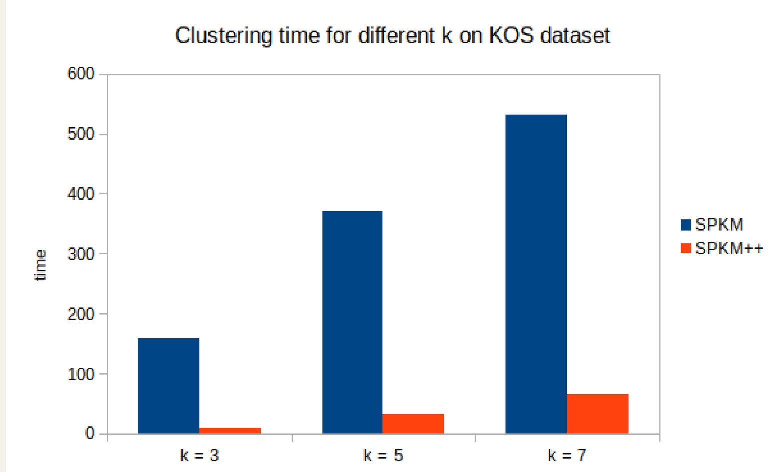
- The first set of experiments compare the performance of SPKM and SPKM++
- Second set of experiments compare the performance of SPKM++ and SPKM-MCMC

DATASET DESCRIPTION

Dataset	No of documents	No of words in the vocab (dimension)	Max no of words in a document (sparsity)
KOS blog entries	3430	6906	457
BBC	9635	2225	128
NIPS full papers	1500	12419	914
20 Newsgroups	1700	56916	734

COMPARISON BETWEEN SPKM AND SPKM++





SEEDING TIME COMPARISON BETWEEN SPKM++ AND SPKM-MCMC

k = 10	KOS	BBC	NIPS	20NEWS
SPKM++	1	1	1	1
SPKM-MCMC(m=5)	x8.0	x7.5	x5.4	x4.8
SPKM-MCMC(m=30)	x7.6	x7.0	x5.0	x3.3
SPKM-MCMC(m=100)	x6.6	x5.7	x4.2	x1.8
SPKM-MCMC(m=500)	x4.0	x2.7	x2.2	x0.5

CLUSTERING COST COMPARISON BETWEEN SPKM++ AND SPKM-MCMC

k = 10	KOS	BBC	NIPS	20NEWS
SPKM++	0.00%	0.00%	0.00%	0.00%
SPKM-MCMC(m=5)	-0.03%	0.07%	0.08%	0.48%
SPKM-MCMC(m=30)	-0.07%	-0.03%	0.08%	0.03%
SPKM-MCMC(m=100)	-0.06%	-0.03%	0.09%	-0.14%
SPKM-MCMC(m=500)	-0.43%	0.06%	-0.13%	-0.08%

Conclusions

- We experimentally validate SPKM++ on publicly available datasets
- We showed its superior performance over SOTA SPKM
- We proposed a Markov Chain based sampling algorithm for initial seeding of *k* data points
- This algorithm retains an *O(log(k))* multiplicative approximation guarantee with respect to optimal clustering results
- We experimentally evaluate our algorithm on public datasets and obtained significant speed-up
- The speed-up in seeding time is more prominent with increase in value of *k*
- The proposed algorithm is simple and easy to implement

References

- David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.
- Olivier Bachem, Mario Lucic, Seyed Hamed Hassani, and Andreas Krause. Fast and provably good seedings for k-means. In Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain, pages 55–63, 2016
- Inderjit S. Dhillon and Dharmendra S. Modha. Concept decompositions for large sparse text data using clustering. Machine Learning, 42(1/2):143–175, 2001. doi: 10.1023/A: 1007612920971
- Yasunori Endo and Sadaaki Miyamoto. Spherical k-means++ clustering. In Modeling Decisions for Artificial Intelligence - 12th International Conference, MDAI 2015, Skövde, Sweden, September 21-23, 2015, Proceedings, pages 103–114, 2015. doi: 10.1007/978-3-319-23240-9 9

Contact Information

Anup Anand Deshmukh

Email: deshmukh.anand@iiitb.org

Web: https://anup-deshmukh.github.io/

LinkedIn: anup-deshmukh-263872a7

Affiliation: IIIT-Bangalore, India