A Scalable Clustering Algorithm for Serendipity in Recommender Systems

Pratheeksha Nair
Anup Anand Deshmukh
Shrisha Rao

International Institute of Information Technology, Bangalore

Pratheeksha.Nair@iiitb.org
Deshmukh.Anand@iiitb.org

17 November 2018

Summary

Introduction

- Two main challenges faced by Collaborative Filtering (CF) algorithms in Recommender Systems:
  - High Sparsity of data points
  - Problem of overspecialization
- This paper is novel in two fronts and tackles the above two problems by
  - Using a *scalable spherical k-means* algorithm for addressing high sparsity.
  - Introducing SC-CF(*Serendipitous Clustering for CF*) that focuses on the overspecialization problem.

Introduction

- CF makes recommendations based on interactions between users and items.
- Users are typically represented by the items they have interacted with

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0	3	0	3	0
User 2	4	0	0	2	0
User 3	0	0	3	0	0
User 4	3	0	4	0	3
User 5	4	3	0	4	0

- Each vector corresponds to a user and each value represents the user's rating for that item.

Introduction

High Sparsity

Introduction

High Sparsity

- Over $8,000$ titles in the U.S. content library and around 130 million streaming subscribers worldwide.

Introduction

High Sparsity

- Over $8,000$ titles in the U.S. content library and around 130 million streaming subscribers worldwide.
- Calculating similarities between users or items then becomes difficult!

- Recommended items are evaluated based on their alignment to a user's rating information.
- Only those items are recommended that are highly similar to items rated highly previously.
- Users live in a *filter bubble* failing to explore *diverse* items
- **Serendipitous** recommendations come into play.

Algorithm

SPKM || clustering

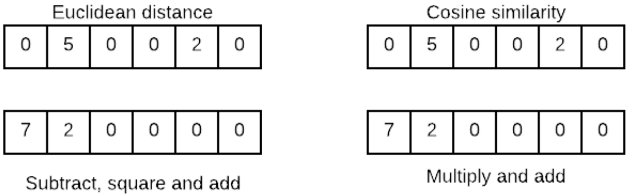
- Spherical k-means(SPKM) clustering uses normalized data points represented on a unit hyper-sphere
- **Cosine similarity** is used as a similarity/distance measure whose computation for sparse vectors is easier

$$cosine\_sim(\mathbf{a}, \mathbf{b}) = \langle \mathbf{a}, \mathbf{b} \rangle = \frac{\mathbf{a}.\mathbf{b}}{|\mathbf{a}|.|\mathbf{b}|}$$

Algorithm
SPKM || clustering

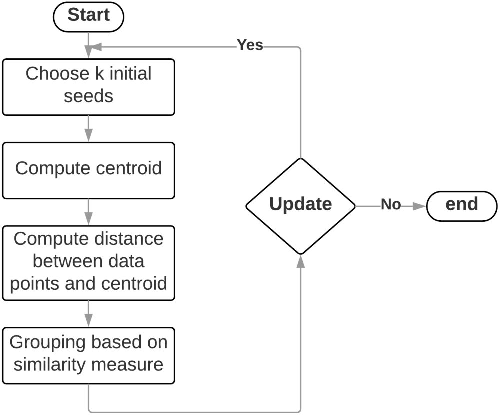
- Spherical k-means(SPKM) clustering uses normalized data points represented on a unit hyper-sphere
- **Cosine similarity** is used as a similarity/distance measure whose computation for sparse vectors is easier

$$cosine\_sim(\mathbf{a}, \mathbf{b}) = \langle \mathbf{a}, \mathbf{b} \rangle = \frac{\mathbf{a}.\mathbf{b}}{|\mathbf{a}|.|\mathbf{b}|}$$



Algorithm

SPKM || clustering



- Choosing k initial cluster centers is important for clustering quality
- Centers should be such that inter-cluster distances are maximized

SPKM ++

- Samples k points based on a probability distribution that maximizes the inter-cluster distances
- The seeding step here runs for k iterations, sampling one point at a time

SPKM ||

- In each iteration, l centers are chosen as opposed to one center in $++$ seeding
- Requires only $\mathcal{O}(\log n)$ number of passes through the data where n is the number of data points

- STEP 1: The l centers in every iteration are chosen based on the probability distribution given below (left side)
- STEP 2: The $(l \times r) + 1$ points in U_x are then re-grouped into k clusters (C) using weighted K-Means (right side)

Algorithm

SPKM || clustering

- STEP 1: The l centers in every iteration are chosen based on the probability distribution given below (left side)
- STEP 2: The $(l \times r) + 1$ points in U_x are then re-grouped into k clusters (C) using weighted K-Means (right side)

while $i \leq \mathcal{O}(\log n)$ **do**
    $L \leftarrow$ sample l points from $x \in X$ each with probability $p_x = \frac{l.d(x, U_x)}{J(U_x)}$
    $U_x \leftarrow U_x \cup L$
    $i \leftarrow i + 1$
end while

$\mathbf{C} \leftarrow$ sample a single $\mathbf{c} \in U_x$ with probability $w_{\mathbf{x}} / \sum_{\mathbf{c}' \in U_x} w_{\mathbf{c}'}$
for $i = 2, \ldots k$ **do**
    Sample $\mathbf{c} \in U_x$ with probability $\frac{w_{\mathbf{x}}(\mathbf{c}, \mathbf{C})}{\sum_{\mathbf{c}' \in U_x} w_{\mathbf{c}'}(\mathbf{c}', \mathbf{C})}$
    $\mathbf{C} \leftarrow \mathbf{C} \cup \{\mathbf{c}\}$
end for
return $\mathbf{C}$

- Goal is to give *unexpected*, *diverse*, yet *relevant* recommendations.
- Apart from regular clusters, data points are assigned to **serendipitous** clusters.
- Both serendipitous and *conventional* clusters are together considered as the "neighbourhood" in the prediction ,

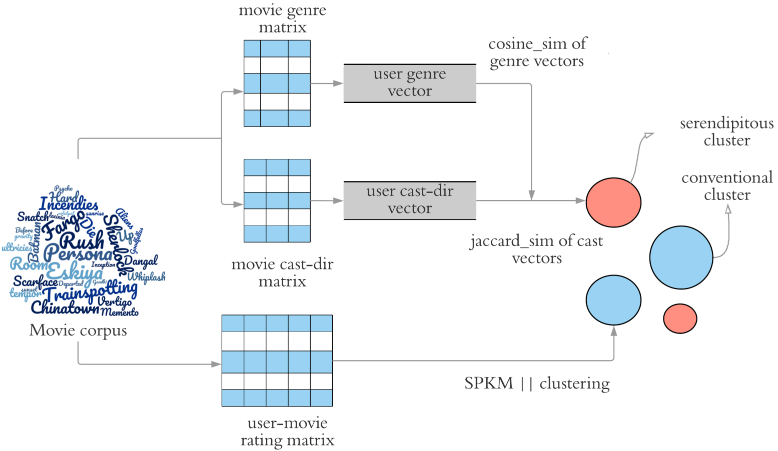
$$r_{uj} = \mu_u + \frac{\sum_{v \in N} \langle u, v \rangle . (r_{vj} - \mu_u)}{\sum_{v \in N} |\langle u, v \rangle|}$$

r_{uj} is the predicted rating for item j by user u
μ_u is the average of all ratings given by user u
N is the neighbourhood of users taken into consideration

Algorithm
SC-CF

- Apart from user-item interactions, a user profile is generated consisting of a *genre profile* and a *cast profile*.
- For each cluster, a *concept* vector is obtained – data point closest to the cluster center (mean of all points in the cluster).
- This concept vector will have an associated genre and cast profile.
- The serendipitous cluster a user belongs to is constituted of:
  - Top p concept vectors whose genre profiles are **least** similar to that of the user.
  - Top q concept vectors whose cast profiles are **most** similar to that of the user.

Algorithm
SC-CF



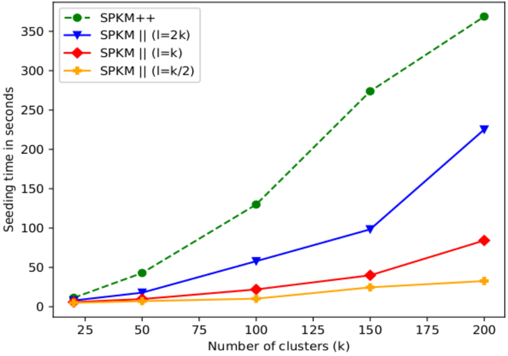
- Real-world dataset description

Data Set	# documents	# words in the vocab (dimension)	max # words in a document (sparsity)
KOS blogs	3430	6906	457
ENRON emails	7000	28102	2021

- Serendipity 2018 dataset
  - 5000 users, 49000 movies
  - Contains users' responses/ratings to serendipitous recommendations

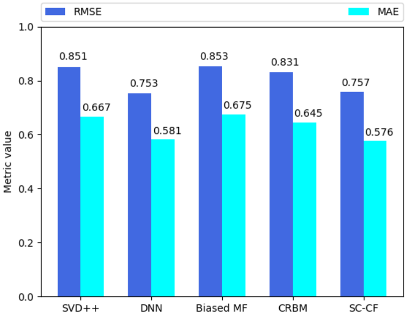
- Table shows the relative improvement in the SPKM || with respect to the cost of SPKM ++
- Graph shows the comparison between the seeding time of SPKM || and SPKM ++

k	SPKM++	SPKM || (Clustering cost)		
		l = k/2	l = k	l = 2k
20	0.00%	3.33%	-6.79%	3.3%
50	0.00%	0.04%	0.08%	-0.12%
100	0.00%	0.11%	0.19%	0.08%
150	0.00%	0.06%	0.09%	-0.03%
200	0.00%	0.12%	0.08%	0.04%

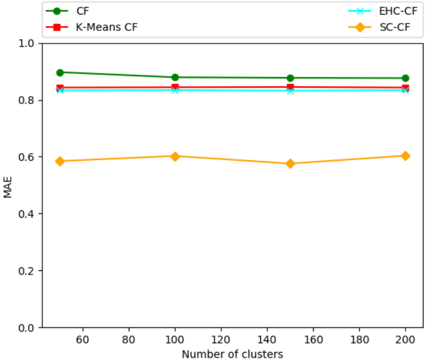


- The algorithms must consider only those users who have a positive correlation with the target user – The neighbourhood selection!

- The algorithms must consider only those users who have a positive correlation with the target user – The neighbourhood selection!



- The best performing algorithm, SC-CF ($l = 2k$ and $k = 150$) achieves 0.757 RMSE and 0.576 of MAE.



- The SC-CF ($l = 2k$ and $k = 150$) reduces the average prediction error by 31% compared to the other clustering methods. technique.

Conclusion

- Apart from user-item interaction profile similarity, diversity and unexpectedness are also important.
- SPKM || gives improved clustering quality with lower number of rounds compared to the widely used ++ seeding technique.
- SC-CF includes serendipitous clusters in the neighbourhood while making predictions and consistently outperforms the popular existing methods by reducing the average prediction error by 9%.
- Results provide empirical evidence that recommendations generated by SC-CF account for diversity and unexpectedness, and thus bear an intrinsic added value.
- Collaborative filtering algorithms can be enriched with serendipity in recommendations.

References

Dhillon, Inderjit S and Modha, Dharmendra S, (2001)
Concept decompositions for large sparse text data using clustering
Machine learning Volume 42, 143 – 175, Springer

Arthur, David and Vassilvitskii, Sergei (2007)
k-means++: The advantages of careful seeding
Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms 1027–1035, Society for Industrial and Applied Mathematics

Endo, Yasunori and Miyamoto, Sadaaki (2015)
Spherical k-means++ clustering
Modeling Decisions for Artificial Intelligence 103 – 114, Springer

Bahmani, Bahman and Moseley, Benjamin and Vattani, Andrea and Kumar, Ravi and Vassilvitskii, Sergei (2012)
Scalable k-means++
Proceedings of the VLDB Endowment 5, 7, 622 – 633, VLDB Endowment

Questions/Comments?

Thank You