Rank aggregation from Pairwise Preferences

Sandip Sinha (09907) **Pranav Mundada (09083)** **Sameer Shah (09062)**

Abstract

Rank aggregation refers to the process of using information about pair-wise preferences among a collection of objects to obtain a global ranking that is consistent with the data. In this project, we look at Rank Centrality (RC), Nuclear Norm Minimization (NNM) and SVM rank aggregation (SVM-RA) and propose a modification of RC which we call SSP1. We studied, implemented and compared these algorithms for varying degrees of completeness of the comparison matrix.

1 Preliminaries

Let the number of objects to be ranked be n. Without loss of generality, assume the objects are $\{1, ..., n\}$, denoted as $[n]$. The algorithms requires information regarding pairwise preferences of items. Let A be a $n \times n$ matrix such that A_{ij} is the proportion of times j is preferred over i.

1.1 Bradley-Terry-Luce (BTL) Model

In the BTL model, we assume there is a true score vector $w = (w_1, ..., w_n) \in \mathbb{R}^n$, $w_i > 0$ for all i. And data is drawn from distribution generated by this true score vector. We scale w such that $\Sigma_i w_i = 1$. Whenever objects i and j are compared, the probability that j beats i is $\frac{w_j}{w_i+w_j}$. Concretely, let d be the number of pairs compared and let k be the number of comparisons done for each pair. For $i, j \in [n], l \in [k]$, let Y_{ij}^l be the indicator random variable for the event that j is preferred over i. Then $\mathbb{P}(Y_{ij}^l = 1) = \frac{w_j}{w_i+w_j}$. Also, Y_{ij}^l's are independent across i, j and l.

1.2 Error Metrics

Let $\hat{(}\sigma)$ be the estimated ranking, σ be the true ranking and w be the true score vector. In case of real datasets, we consider the ranking we get from the complete matrix as the true ranking. We look at two error metrics. The second one is introduced by us.

1. $D_w(\hat{\sigma}) := \left(\frac{1}{2n||w||^2} \sum_{i<j} (w_i - w_j)^2 \mathbb{I}((w_i - w_j)(\hat{\sigma_i} - \hat{\sigma_j}) > 0) \right)^{1/2}$
2. $D_{frac}(\sigma, \hat{\sigma}) := 1 - \frac{1}{n} \sum_i \mathbb{1}(\sigma(i) = \hat{\sigma}(i))$

2 Rank Centrality

This is converted into a stochastic matrix P which is provided to the algorithm. The computation done by the algorithm can be understood as the evolution of a rapidly mixing Markov chain corresponding to a random walk on the graph induced by P, and the score vector turns out to be the stationary distribution of this Markov chain.

2.1 Main idea for Rank Centrality

Given d and k, the algorithm chooses d random pairs to be compared k times each. Let $a_{ij} = \frac{1}{k}\Sigma_{l=1}^k Y_{ji}^l$. Let $A_{ij} = \frac{a_{ij}}{a_{ij}+a_{ji}}$. Then $A_{ij} \to \frac{w_j}{w_i+w_j}$ as $k \to \infty$. Let $A_{ij} = 0$ if i and j are not compared. Consider a directed

graph $G = ([n], E, A)$ with n nodes (corresponding to objects), and weighted edges between pairs of objects which are compared. Let d_{max} be the maximum out-degree of a node. Define a stochastic matrix P by

$$\begin{aligned} P_{ij}^{(RC)} &= \frac{1}{d_{max}} A_{ij} && \text{if} \quad i \neq j \\ &= 1 - \frac{1}{d_{max}} \Sigma_{k \neq i} A_{ik} && \text{if} \quad i = j \end{aligned} \tag{1}$$

Then P is a stochastic matrix which represents a time-homogeneous Markov Chain which is aperiodic and irreducible, hence ergodic and rapidly mixing. Moreover, P is reversible, and w is the stochastic vector which satisfies the detailed balance condition:

$$w_i P_{ij} = w_j P_{ji} \forall i, j$$

Hence, w is a stationary distribution of P. w is an eigenvector of P with eigenvalue 1, and the eigenvalues of P are at most 1 in absolute value. When P has a unique eigenvector w corresponding to the largest eigenvalue 1, then the Markov chain converges rapidly to w. Hence, it is enough to compute the leading eigenvector by power iteration. The mixing rate of the Markov Chain is affected by the spectral gap ξ of the graph.

2.2 Algorithm

Input: $G = ([n], E, A)$
Output: Score w and Rank $\pi \in S_n$
1: Compute the transition matrix Pas described above.
2: Compute the stationary distribution π by power iteration.

2.3 Main Theorems

Theorem 2.1. Given n objects and a connected comparison graph $G = ([n], E, A)$, let each pair $(i, j) \in E$ be compared for k times with outcomes produced as per a BTL model with parameters $w_1, ..., w_n$. Then, for some positive constant $C \geq 8$ and when $k \geq 4C^2(b^5\kappa^2/d_{max}\xi^2) \log n$, with probability at least $1 - 4n^{-C/8}$,

$$\frac{||\pi - \tilde{\pi}||}{||\tilde{\pi}||} \leq \frac{Cb^{5/2}k}{\xi} \sqrt{\frac{\log n}{kd_{max}}}$$

where $\tilde{\pi}(i) = w_i/\Sigma_l w_l, b = max_{i,j} w_i/w_j$, and $\kappa = d_{max}/d_{min}$.

Theorem 2.2. Given n objects, let the comparison graph $G = ([n], E)$ be generated by selecting each pair to be in E with probability e/n independently of everything else. Each such chosen pair of objects is compared k times with outcomes produced as per a BTL model with parameters $w_1, ..., w_n$. Then, if $e \geq 10C^2 \log n$ and $ke \geq 128C^2b^5 \log n$, the following holds with probability at least $1 - 10n^{-C/8}$:

$$\frac{||\pi - \tilde{\pi}||}{||\tilde{\pi}||} \leq 8Cb^{5/2} \sqrt{\frac{\log n}{ke}}$$

Note that Theorem 2 is nearly order-optimal since Erdos-Renyi graphs are connected with high probability if $e = \Omega(logn)$, i.e. the total number of pairs compared is $\Omega(n \log n)$.

2.4 Experimental Results

We considered the following variations of constructing the P matrix and checked their performance against Rank centrality.

$$\begin{aligned} P_{ij}^{(RC)} &= (1/d_{max})A_{ij}, && \text{if} \quad i \neq j \\ &= 1 - (1/d_{max}) \sum_{k \neq i} A_{ik}, && \text{if} \quad i = j \end{aligned} \qquad P_{ij}^{(MC2)} = \frac{a_{ij}}{\Sigma_{k \neq i} a_{ik}}$$

$$\begin{aligned} P_{ij}^{(SSP1)} &= \frac{A_{ij}}{\max_i \Sigma_{k \neq i} A_{ik}} && \text{if} \quad i \neq j \\ &= 1 - \Sigma_{l \neq i} \frac{A_{il}}{\max_i \Sigma_{k \neq i} A_{ik}} && \text{if} \quad i = j \end{aligned} \qquad \begin{aligned} P_{ij}^{(MC3)} &= \frac{a_{ij}}{deg(i)}, && \text{if} \quad i \neq j \\ &= 1 - \Sigma_{k \neq i} \frac{a_{ik}}{deg(i)} && \text{if} \quad i = j \end{aligned}$$

Note: The SSP1 algorithm is introduced by us. It is a variation in the way we define the P matrix. We observed it works as good as Rank Centrality, if not better. We have tested it on both synthetic and real data. While implementing RC, MC3 and SSP1 it is important to ensure that none of the diagonal elements of the P matrix are one. In case they are, it implies that that object is preferable over all other objects with complete certainty. In such situations we give that object highest rank and recursively repeat our algorithm for remaining objects.
MC2 and MC3 are spectral ranking algorithms which were introduced by Dwork et al. (2001a). SSP1 algorithm is introduced by us. First we tested the four algorithms mentioned above on synthetically generated data-sets with n=30. Their behavior for one such generated data is provided in Fig. 1. Note that d is the edge sampling rate.

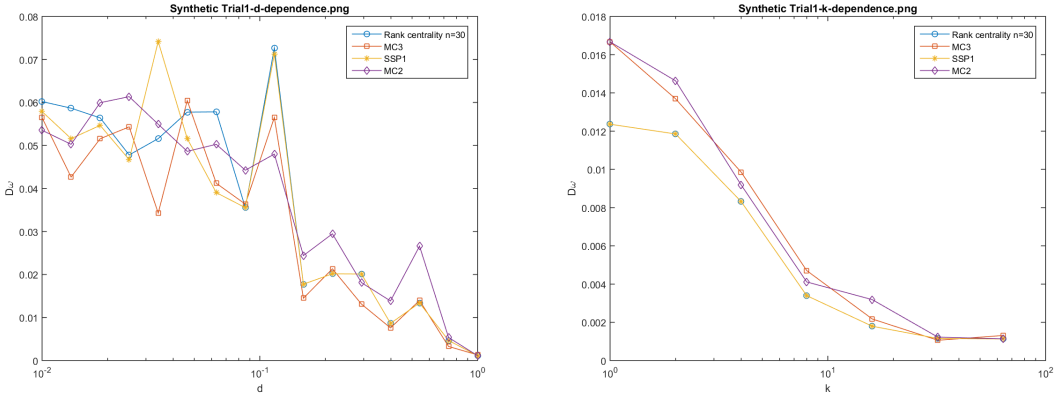

Figure 1: For Synthetic data: The figure on the left keeps k=64 fixed, while the figure on the right keeps d=1 fixed.

Since all the four algorithms have shown similar trend in the error function, we have implemented all the four algorithms on 444 different data-sets obtained from Preflib database. Generally the algorithms appear to be working as reported in the reference paper.

3 Nuclear Norm Minimization

In 2011, David F. Gleich and Lek-Heng Lim introduced a new norm on a matrix, the Nuclear norm which converted the NP-hard matrix completion problem into a convex problem. Using this new norm they gave a ranking algorithm. Given a rating matrix R we compute the pairwise comparison matrix $\hat{Y}$. Then we use a matrix completion algorithm on Y (Y is the sub-matrix of $\hat{Y}$ where only the terms which were calculated from at least c-many individual comparisons are retained) to get X, which is used to compute scores of individual items. Ranking is based on this score vector s.

3.1 Matrix Completion Problem

Given a matrix Y, the goal is to find the lowest rank matrix X that agrees with Y in all the non-zeros. Let Ω be the index set corresponding to the known entries of Y. A matrix operation is defined w.r.t this indexing which maps these known entries to a vector.

$$[A(Y)]_i = Y_{w_i}$$

where is $w_i = (r, s)$ is an index in the matrix Y. Let $b = \mathcal{Y}$ The problem of matrix completion corresponds to finding solution of

$$\begin{aligned} \text{minimize} \quad & rank(X) \\ \text{subject to} \quad & \mathcal{A}(X) = b \end{aligned}$$

Unfortunately this problem is NP-hard. And that is where the significance of "Nuclear Norm Model" lies. This NP-hard problem of matrix completion is converted into a convex problem by defining a new norm as follows : For a matrix Y, the nuclear norm is defined as

$$\|Y\|_* = \sum_{i=1}^{rank(Y)} \sigma_i(Y)$$

where $\sigma_i(Y)$ is the ith singular value if Y. This norm is a convex under-estimator of the rank function, i.e., $\|Y\|_* \leq rank(Y)\sigma_{max}(Y)$. So now the matrix completion problem is a convex problem.

$$\begin{aligned} \text{minimize} \quad & \|X\|_* \\ \text{subject to} \quad & \mathcal{A}(X) = b \end{aligned}$$

3.2 Algorithm

For the implementation of matrix completion algorithm, LASSO formulation is used.

$$\begin{aligned} \text{minimize} \quad & \|\mathcal{A}(X) - b\|_2 \\ \text{subject to} \quad & \|X\| \leq 2 \quad \text{and} \quad X = -X^T \end{aligned}$$

And to solve optimization the 'Singular Value Projection' (SVP) algorithm is used.

SVP algorithm is as follows :

Let $\Omega(X)$ denote output of $\mathcal{A}(X)$
Input: index set Ω, target values b, target rank k
Parameters: maximum rank k, step length η, tolerance ϵ Initialize : $X^{(0)} = 0, t = 0$

Iterate:

Set $U^{(t)}\Sigma^{(t)}{V^{(t)}}^T$ to be rank-k SVD of a matrix with non-zeros Ω and values $\Omega(X^{(t)}) - \eta(\Omega(X^{(t)} - b)$
$X^{(t+1)} \leftarrow U^{(t)}\Sigma^{(t)}{V^{(t)}}^T$
$t \leftarrow t + 1$

Until $\|\Omega(X^{(t)} - b\|_2 > \epsilon$

SVP gives X in the form USV^T which is also skew-symmetric. We use X to calculate the score vector as

$$s = \frac{1}{n}(USV^T)\mathbf{e}$$

where $\mathbf{e}$ is a vector of all ones of size n.

4 SVM-Rank Aggregation Algorithm

In 2014, Agarwal and Rajkumar proposed a new SVM-based algorithm which gives optimal performance under more general conditions than the BTL model. We say a non-negative $n \times n$ matrix P is a pairwise preference matrix if $P_{ij} = 1 - P_{ji}$ for all $i \neq j$ and $P_{ii} = 0$ for all $i \in [n]$.

Definition 4.1. Low-Noise (LN) Condition:
We say a pairwise preference matrix P satisfies the Low-Noise (LN) Condition if

$$\forall i \neq j P_{ij} > P_{ji} \Rightarrow \sum_{k=1}^{n} P_{kj} > \sum_{k=1}^{n} P_{ki}$$

Definition 4.2. Generalized Low-Noise (GLN) Condition:
We say a pairwise preference matrix P satisfies the Generalized Low-Noise (LN) Condition if $\exists \alpha \in \mathbb{R}^n$ such that

$$\forall i \neq j : P_{ij} > P_{ji} \Rightarrow \sum_{k=1}^{n} \alpha_k P_{kj} > \sum_{k=1}^{n} \alpha_k P_{ki}$$

Observation 1: If P satisfies BTL condition, then P satisfies LN condition. If P satisfies LN condition, then any permutation π that ranks the objects in decreasing order of f, given by $f_i = \sum_{k=1}^{n} P_{ki}$ for $i \in [n]$, is optimal.

Observation 2: If P satisfies LN condition, then P satisfies GLN condition with $\alpha_k = 1 \forall k \in [n]$. If P satisfies GLN condition, any permutation π that ranks the objects in decreasing order of f, given by $f_i = \sum_{k=1}^{n} \alpha_k P_{ki}$ is optimal.

Definition 4.3. (P-induced dataset):
For a matrix $P \in \{0, 1\}^n$, the P-induced dataset $S_P = \{v_{ij}, z_{ij}\}_{i<j}$ consists of the $\binom{n}{2}$ vectors $v_{ij} = P_i - P_j \in \mathbb{R}^n$ $(i < j)$, where P_i denotes the i^{th} column of P, together with binary labels $z_{ij} = \text{sign}(P_{ji} - P_{ij}) \in \{\pm 1\}$.

The objective of running SVM is to determine the vector α for which the empirical pairwise comparison matrix P satisfies the GLN condition, from which the ranking can be obtained.

4.1 SVM-RA algorithm

Result: Ranking $\pi \in S_n$

Construct $\hat{P}$-induced dataset $S_{\hat{P}}$
If $\hat{P}$ is linearly separable by hyperplane through origin **then**
train hard-margin linear SVM on $S_{\hat{P}}$ to get $\hat{\alpha} \in \mathbb{R}^n$;
Else train soft-margin linear SVM (with any suitable value for regularization parameter) on $S_{\hat{P}}$ to get $\hat{\alpha} \in \mathbb{R}^n$;

For $i = 1$ to n:

$$\hat{f}_i = \sum_{k=1}^{n} \hat{\alpha}_k \hat{P}_{ki};$$

Return Ranking $\pi \in \text{argsort}(\hat{f})$

5 Conclusion

Figure 2, shows that error in NNM implementation reduces very slowly with the number of comparisons for a given pair. After running the algorithms over 100+ real datasets from Preflib we conclude that RC works better than SVM-RA whenever BTL model is satisfied. Figure 3 shows SVM-RA is more robust than RC whenever BTL fails. We also observe that our proposed modification of RC works better than RC for low number of pair comparisons.

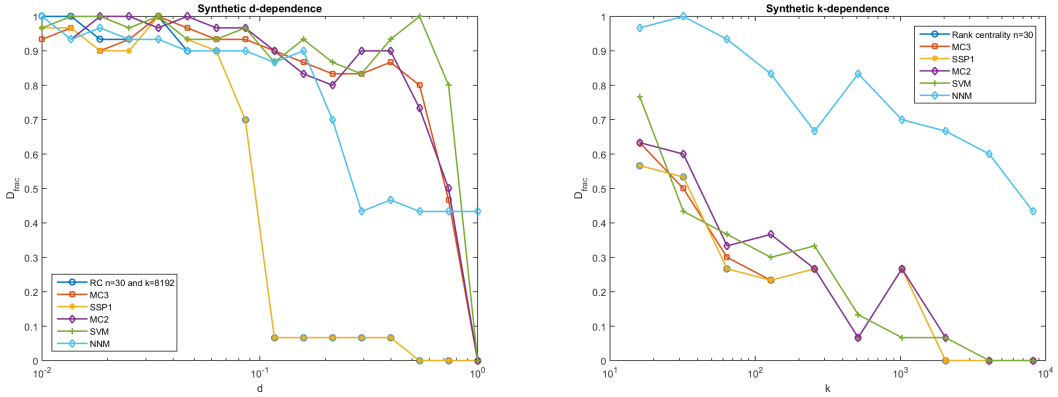

Figure 2: Results for data taken from `http://www.preflib.org/data/`

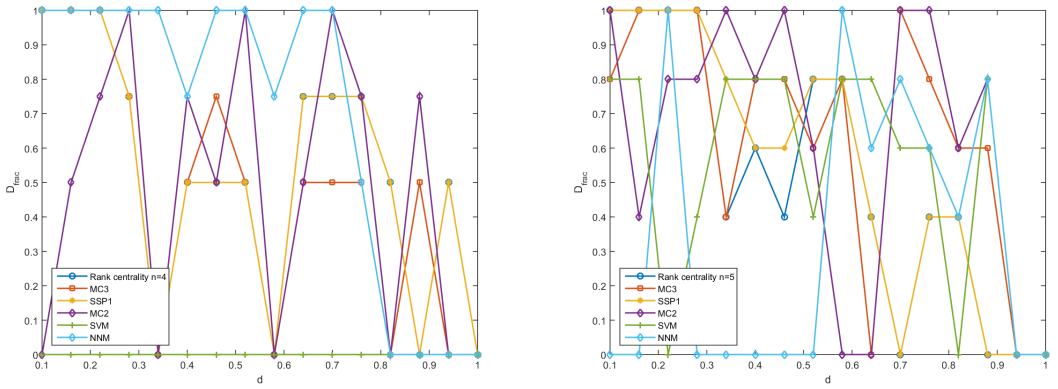

Figure 3: Left figure: LN dataset; Right figure: GLN dataset. The datasets were taken from reference [2]

References

[1] Sahand Negahban & Sewoong Oh & Devavrat Shah (2015) Rank Centrality: Ranking from Pair-wise Comparisons, arXiv:1209.1688v4 [cs.LG] 12 Nov 2015.
[2] Arun Rajkumar & Shivani Agarwal (2014) A Statistical Perspective of Algorithms for Rank Aggregation from Pairwise Data, JMLR: W&CP volume 32.
[3] David F. Gleich & Lek-Heng Lim (2011) Rank Aggregation via Nuclear Norm Minimization, KDD'11, August 21-24,2011,San Diego, CA,USA.