Rank Aggregation for Anomaly Detection

***Abstract*—Most anomaly detection algorithms rank input data instances based on the degree of anomalousness. These algorithms might be biased (due to model assumptions) or have high variance (due to stochastic nature). Therefore, ensemble techniques that use more than one algorithm are often preferred. Here, individual algorithms first generate their own rankings. Next, these rankings are aggregated to produce a final ranking. The Plackett-Luce (PL) model is popularly used for such rank aggregation. In this project we extended the PL model by adding Gamma prior and also tried another technique called Generalized Method of Moments (GMM) for producing better anomaly ranking.**

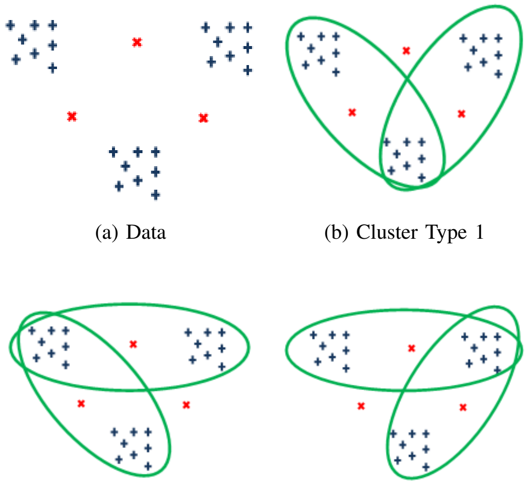
I. Introduction

Ranking is an inherent task in many fields such as web page ranking by search engines, team ranking in sports, document suggestion by recommender systems and many other e-commerce applications. In this project we are mainly focusing on ranking anomalies.

Anomaly detection is an important research area and is being widely applied in fields like computer security, fraud detection in credit card transactions, medical diagnosis, etc. Due to their nature, each anomaly must be investigated by a human supervisor whose time and effort is expensive. Therefore, both false positives and false negatives must be kept to a minimum. To detect anomalies, usually the first task is to model the data using a reasonable set of assumptions and then have the model return an anomaly score for each instance. These assumptions play a critical role in the accuracy of the anomaly detection system.

Fig. 1 shows a simple illustration where a mis-specified model might fail to detect all anomalies. Here the points marked in red are anomalies. We can see that there are three clusters in the data (Fig. 1a). If we model this data as a mixture of Gaussians with two components, then the data will be under-fit resulting in models similar to those in Fig. 1b, Fig. 1c, and Fig. 1d with high probability. Due to the stochastic nature of the model fitting algorithm (using Expectation-Maximization), we can generate (say) 100 different model instances with random initializations. Clearly, most of these models cannot report all anomalies by themselves. Moreover, a simple averaging of anomaly scores across such models might smudge out the anomaly signals.

The above example motivates us to find a better aggregation mechanism that preserves most anomaly signals. A promising approach is to first untangle the major ranking preferences among the detectors. For instance, in the above example there might be only three such preferences corresponding to the three highly probable cluster types. Once the major preference types have been identified, the top anomalies under each of them can be prioritized to detect all anomalies sooner. To identify the individual ranking preferences, we model the output anomaly ranks from all detectors as a mixture of Plackett-Luce [1] distributions.



(a) Data

(b) Cluster Type 1

(c) Cluster Type 2

(d) Cluster Type 3

Fig. 1: Illustration of model mis-specification. The points marked in red are anomalies.

II. Related Work

The general field that deals with the ranking of objects is called learning to rank[2]. A lot of different approaches have been tried to this purpose. In [3] they used mixture models along with PL distribution and for inferring the parameters using maximum likelihood estimation (MLE) they used the MM (Minorize-Maximization) algorithm[4] which is a generalized version of popular Expectation Maximization (EM) algorithm. They applied their idea on the Irish college degree applications data to find groups of similar applicants. In [5] they used Generalized Method of Moments (GMM) for computing parameters of PL distribution. In [6] they used Bayesian inference technique along with expectation propagation for inferring PL distribution parameters.

The Plackett-Luce distribution (PL) ([1], [7]) which is an generalization of the Bradley-Terry model ([8]) is commonly used for modeling item rankings in the statistics and machine learning community. It can help smooth the noisy rankings from multiple judges and output a consolidated rank even when each judge ranks only a subset of the items. The PL distribution can also be extended to model a mixture of more than one underlying ranking preference ([3], [9]). However, there has been relatively less work in using PL distribution to improve anomaly rankings from multiple detectors and moreover, most of the previous work has focused on scenarios where the total number of items being ranked is small (a few tens or hundreds). One of our objectives in this project is also to rank thousands of items so that our system is more useful in practice.

III. Problem Description and Formulation

Assume that we have *N* objects and *D* anomaly detectors. Each detector provides its own ranking over (a subset of) the *N* objects. Under the PL distribution, we assume that the underlying preference is defined by a vector of positive real numbers $\mathbf{v} = \{v_1, ..., v_N\}$ where the i-th object is associated with v_i. The joint probability of ranking the i-th and k-th objects in the first and second place respectively is $(\frac{v_i}{\sum_{j=1}^{N} v_j} \times \frac{v_k}{\sum_{j \neq i}^{N} v_j})$. In this manner, the detector d's output rank likelihood is $\prod_{i=1}^{N_d} \frac{v_{d_i}}{\sum_{t=i}^{N} v_{d_t}}$ where d_i is the i-th ranked item under d. We have assumed that detector d ranks only N_d items. The PL distribution assumes that there exists only one underlying preference distribution. However, it is common to have situations with more than one type of preference e.g., when ranking courses, some students would rank medical courses higher while others would prefer engineering courses. It is then helpful to learn a mixture of PL distributions. If we assume there are *K* types of preferences among the *D* detectors, and if we knew the preference of each detector, then the likelihood of all rankings would be:

$$l = \prod_{d=1}^{D} \pi_{z_d} \prod_{i=1}^{N_d} \frac{v_{z_d d_i}}{\sum_{t=i}^{N} v_{z_d d_t}} \tag{1}$$

$$\Rightarrow l_c = \prod_{d=1}^{D} \prod_{k=1}^{K} \left(\pi_k \prod_{i=1}^{N_d} \frac{v_{k d_i}}{\sum_{t=i}^{N} v_{k d_t}} \right)^{z_{dk}} \tag{2}$$

where z_{dk} is an indicator variable which is 1 when the judge d belongs to component k, and 0 otherwise. d_t is the t-th ranked item under the d-th judge. l_c is referred to as the complete data likelihood.

The log-likelihood is:

$$\begin{aligned} L_c &= \sum_{d=1}^{D} \sum_{k=1}^{K} z_{dk} \left(\log(\pi_k) + \sum_{i=1}^{N_d} \log(\frac{v_{kd_i}}{\sum_{t=i}^{N} v_{kd_t}}) \right) \\ &= \sum_{d=1}^{D} \sum_{k=1}^{K} z_{dk} \left(\log(\pi_k) + \sum_{i=1}^{N_d} \log(v_{kd_i}) - \sum_{i=1}^{N_d} \log(\sum_{t=i}^{N} v_{kd_t}) \right) \end{aligned} \tag{3}$$

Expectation Maximization (EM) is frequently used to infer parameters that maximize log-likelihoods like Equation 3 However, Equation 3 is hard to optimize since it is not convex when the v_{kd}'s are fixed; instead, the *MM algorithm* ([10]) is used which is derived next. We know that for a concave function:

$$f(x) \leq f(y) + f'(y)(x - y) \tag{4}$$

since $f(x) = \log(x)$ is a concave function:

$$\Rightarrow \log(x) \leq \log(y) + \frac{1}{y}(x - y)$$

$$\Rightarrow \log(x) \leq \log(y) + \frac{x}{y} - 1$$

Hence, we have:

$$-\log(x) \geq -\log(y) + 1 - \frac{x}{y} \tag{5}$$

Now, let $x = \sum_{t=i}^{N} v_{kd_t}$ and $y = \sum_{t=i}^{N} v_{kd_t}^{(l)}$. Then, plugging these into Equation 5 we get:

$$-\log(\sum_{t=i}^{N} v_{kd_t}) \geq -\log(\sum_{t=i}^{N} v_{kd_t}{}^{(l)}) + 1 - \frac{\sum_{t=i}^{N} v_{kd_t}}{\sum_{t=i}^{N} v_{kd_t}^{(l)}} \tag{6}$$

$$\Rightarrow -\log(\sum_{t=i}^{N} v_{kd_t}) \geq -\frac{\sum_{t=i}^{N} v_{kd_t}}{\sum_{t=d_t}^{N} v_{kd_t}^{(l)}} + const. \tag{7}$$

Equation 7 is true up to a constant – this follows from the fact that $v_{kd_t}^{(l)}$'s are constants in the maximization. Now, plugging Equation 7 into Equation 3 we have:

$$\begin{aligned} L_c \geq & \sum_{d=1}^{D} \sum_{k=1}^{K} z_{dk} \log(\pi_k) + \\ & \sum_{d=1}^{D} \sum_{k=1}^{K} \sum_{i=1}^{N_d} z_{dk} \left(\log(v_{kd_i}) - \frac{\sum_{t=i}^{N} v_{kd_t}}{\sum_{t=i}^{N} v_{kd_t}^{(l)}} \right) \end{aligned} \tag{8}$$

The right hand side of 8 acts as the surrogate objective function in the *MM algorithm*.

IV. Adding Priors to PL-Mix

One of the short-comings of the PL model is that if the data can be partitioned into two sets such that no item in one set is ever ranked higher than any item in the other set by any detector, then the v_i's for the items in the second set can become unbounded under MLE. This can be avoided by adding priors that somehow bound the v_i's. Since the v_i's are always positive, we can put *Gamma* priors on each of these. Let us assume $Gamma(\alpha_0, \beta_0)$ as priors on all v_i parameters.

$$\begin{aligned} l_c = & \left(\prod_{k=1}^{K} \prod_{n=1}^{N} \frac{\beta^{\alpha_0}}{\Gamma(\alpha_0)} v_{kn}^{\alpha_0 - 1} e^{-\beta_0 v_{kn}} \right) \\ & \times \prod_{d=1}^{D} \prod_{k=1}^{K} \left(\pi_k \prod_{i=1}^{N_d} \frac{v_{kd_i}}{\sum_{t=i}^{N} v_{kd_t}} \right)^{z_{dk}} \end{aligned} \tag{9}$$

The log-likelihood is:

$$\begin{aligned} L_c = & \left(\sum_{k=1}^{K} \sum_{n=1}^{N} \log(\frac{\beta_0^{\alpha_0}}{\Gamma(\alpha_0)} v_{kn}^{\alpha_0 - 1} e^{-\beta_0 v_{kn}}) \right) \\ & + \sum_{d=1}^{D} \sum_{k=1}^{K} z_{dk} \log \left(\pi_k \prod_{i=1}^{N_d} \frac{v_{kd_i}}{\sum_{t=i}^{N} v_{kd_t}} \right) \end{aligned} \tag{10}$$

Ignoring the constants, the log-likelihood that needs to be maximized is:

$$
\begin{aligned}
L_c = &\left(\sum_{k=1}^{K}\sum_{n=1}^{N}\log(v_{kn}^{\alpha_0-1}e^{-\beta_0 v_{kn}})\right)\\
&+\sum_{d=1}^{D}\sum_{k=1}^{K} z_{dk}\log\left(\pi_k\prod_{i=1}^{N_d}\frac{v_{kd_i}}{\sum_{t=i}^{N} v_{kd_t}}\right)
\end{aligned}
$$

$$
\begin{aligned}
&= (\alpha_0-1)\sum_{k=1}^{K}\sum_{n=1}^{N}\log(v_{kn}) - \sum_{k=1}^{K}\sum_{n=1}^{N}\beta_0 v_{kn}\\
&\quad+\sum_{d=1}^{D}\sum_{k=1}^{K} z_{dk}\Bigg(\log(\pi_k)\\
&\quad+\sum_{i=1}^{N_d}\log(v_{kd_i}) - \sum_{i=1}^{N_d}\log(\sum_{t=i}^{N} v_{kd_t})\Bigg)
\end{aligned} \tag{11}
$$

Now, plugging equation 7 to 11 we have:

$$
\begin{aligned}
L_c \geq &(\alpha_0-1)\sum_{k=1}^{K}\sum_{n=1}^{N}\log(v_{kn}) - \sum_{k=1}^{K}\sum_{n=1}^{N}\beta_0 v_{kn}\\
&+\sum_{d=1}^{D}\sum_{k=1}^{K} z_{dk}\Bigg(\log(\pi_k)\\
&+\sum_{i=1}^{N_d}\log(v_{kd_i}) - \sum_{i=1}^{N_d}\frac{\sum_{t=i}^{N} v_{kd_t}}{\sum_{t=i}^{N} v_{kd_t}^{(l)}}\Bigg)
\end{aligned} \tag{12}
$$

The computations required in the E- and M-steps at the (l+1)-th iteration of the *MM algorithm* are shown below.

E-Step

$$
\hat{z}_{dk} = \frac{\pi_k \prod_{i=1}^{N_d}\frac{v_{kd_i}}{\sum_{t=i}^{N} v_{kd_t}}}{\sum_{k'=1}^{K}\pi_{k'}\prod_{i=1}^{N_d}\frac{v_{k'd_i}}{\sum_{t=i}^{N} v_{k'd_t}}} \tag{13}
$$

M-Step

Maximizing L_c using Lagrangian w.r.t π_k subject to the constraint that $\sum_{k=1}^{K}\pi_k = 1$:

$$
\begin{aligned}
\hat{\pi}_k &= \frac{\sum_{d=1}^{D} z_{dk}}{\sum_{k'=1}^{K}\sum_{d=1}^{D} z_{dk'}}\\
&= \frac{\sum_{d=1}^{D} z_{dk}}{\sum_{d=1}^{D}\sum_{k'=1}^{K} z_{dk'}}\\
&= \frac{\sum_{d=1}^{D} z_{dk}}{D} \text{ (since } \sum_{k'=1}^{K} z_{dk'} = 1)
\end{aligned} \tag{14}
$$

Maximizing L_c w.r.t v_{kn}:

$$
\begin{aligned}
\frac{\partial L_c}{\partial v_{kn}} = 0 = &\frac{\partial}{\partial v_{kn}}\Bigg((\alpha_0-1)\sum_{k=1}^{K}\sum_{n=1}^{N}\log(v_{kn}) - \sum_{k=1}^{K}\sum_{n=1}^{N}\beta_0 v_{kn}\\
&+\sum_{d=1}^{D}\sum_{i=1}^{N_d} z_{dk}\log(v_{kd_i}) - \sum_{d=1}^{D}\sum_{i=1}^{N_d} z_{dk}\left(\frac{\sum_{t=i}^{N} v_{kd_t}}{\sum_{t=i}^{N} v_{kd_t}^{(l)}}\right)\Bigg)\\
\Rightarrow 0 = &\frac{\alpha_0-1}{v_{kn}} - \beta_0 + \frac{1}{v_{kn}}\sum_{d=1}^{D} z_{dk}\sum_{i=1}^{N_d}\mathbf{1}[d_i = n]\\
&-\sum_{d=1}^{D}\sum_{i=1}^{N_d} z_{dk}\mathbf{1}[n \in \{d_i, ..., d_N\}]\left(\sum_{t=i}^{N} v_{kd_t}^{(l)}\right)^{-1}\\
\Rightarrow v_{kn}^{l+1} = &\\
&\frac{\sum_{d=1}^{D} z_{dk}\sum_{i=1}^{N_d}\mathbf{1}[d_i = n] + \alpha_0 - 1}{\sum_{d=1}^{D}\sum_{i=1}^{N_d} z_{dk}\mathbf{1}[n \in \{d_i, ..., d_N\}]\left(\sum_{t=i}^{N} v_{kd_t}^{(l)}\right)^{-1} + \beta_0}
\end{aligned} \tag{15}
$$

Where, $\mathbf{1}[d_i = n]$ and $\mathbf{1}[n \in \{d_i, ..., d_N\}]$ are indicator functions and evaluate to 1 if the corresponding condition [.] is true and evaluate to 0 otherwise.

V. Mixture of PL with Generalized Method of Moments (GMM)

Recently, an alternate estimation technique was proposed by [5] using GMM. In this approach, we try to find a set of parameters which can match the expected pair-wise win/loss proportions. The proportions to be matched are estimated from the observed ranking data.

As in previous sections, assume we have a set D of detectors, each of which provides a ranking over N items and let $\gamma = \{\gamma_1, ..., \gamma_N\}$ be the PL distribution parameters (in previous sections we denoted these parameters as $\{v_1, ..., v_N\}$). Let $(c_i \succ c_j)$ be the event that item c_i is ranked better than item c_j. It can be shown by marginalization under the PL distribution that $P(c_i \succ c_j) = \sum_{d:c_i \succ c_j} P(d|\gamma) = \frac{\gamma_i}{\gamma_i + \gamma_j}$. For each detector $d \in D$, define a matrix $P(d)$ (which is $N \times N$ dimension) as:

$$
P(d)_{ij} = \begin{cases} X_d^{c_i \succ c_j} & \text{if } i \neq j \\ -\sum_{l \neq i} X_d^{c_l \succ c_i} & \text{if } i = j \end{cases} \tag{16}
$$

Let $P(D) = \frac{1}{|D|}\sum_{d \in D} P(d)$.
We can then verify that

$$
(E_{d|\gamma*}[P(d)])_{ij} = \begin{cases} \frac{\gamma_i*}{\gamma_i* + \gamma_j*} & \text{if } i \neq j \\ -\sum_{l \neq i}\frac{\gamma_i*}{\gamma_i* + \gamma_l*} & \text{if } i = j \end{cases} \tag{17}
$$

which implies: $E_{d|\gamma*}[P(d)].\gamma* = \mathbf{0}$.
For example, if $N = 3$ and $D = \{[c_1 \succ c_2 \succ c_3], [c_2 \succ c_3 \succ c_1]\}$, then we have:

$$
P(D) = \begin{bmatrix} -1 & 1/2 & 1/2 \\ 1/2 & -1/2 & 1 \\ 1/2 & 0 & -3/2 \end{bmatrix} \tag{18}
$$

Generalized Method of Moments (GMM): Assume that we have a column vector valued function $g(d, \gamma) \in R^N$ and a $N\times N$ matrix W. Let $g(D, \gamma) = \frac{1}{|D|}\sum_{d\in D} g(d, \gamma)$. GMM is a generic technique that computes the optimal set of parameters $\gamma* = \inf_{\gamma\in\Omega} g(D, \gamma)^T W^{-1} g(D, \gamma)$, where Ω is the parameter space. We can see from the above setup that the PL distribution parameters can be inferred using GMM if we set W as the Identity matrix, and $g(D, \gamma) = P(D).\gamma$.

GMM has been shown to be faster than the algorithm discussed earlier for inferring parameters of a single PL distribution [5]. Therefore, we would like to extend it for inferring the parameters of a mixture of PL distributions as well. Similar to Section III we again assume we have K distinct preferences defined by K sets of PL distribution parameters and assign *soft*-memberships to each detector – $\mathbf{Z}_{[K\times D]} = \{\mathbf{z}_1, ..., \mathbf{z}_D\}$, $\mathbf{z}_d = \{z_{1d}, ..., z_{Kd}\}$, $d = 1..D$ where z_{kd} is the probability that detector d belongs to group k, and $\sum_{k=1}^{K} z_{kd} = 1 \; \forall d = 1..D$. The PL parameters under the mixture model are defined as $\mathbf{\Gamma}_{[N\times K]} = \{\gamma_1, ..., \gamma_K\}$, and $\gamma_k = \{\gamma_{k1}, ..., \gamma_{kN}\}$. The optimization function under this setup is:

$$\begin{aligned}\mathbf{\Gamma}*, \mathbf{Z}* &= \inf_{\Gamma, Z} \sum_{k=1}^{K} (\mathbf{S}_k\gamma_k)^T \mathbf{S}_k\gamma_k \\ &= \inf_{\Gamma, Z} \sum_{k=1}^{K} {\gamma_k}^T (\mathbf{S}_k^T \mathbf{S}_k)\gamma_k \qquad (19)\\ \text{s.t.} & \sum_{k=1}^{K} z_{kd} = 1, z_{kd} \geq 0, \gamma_{ki} \geq 0\end{aligned}$$

where

$$\mathbf{S}_k = \frac{1}{\sum_{d=1}^{D} z_{kd}} \sum_{d=1}^{D} z_{kd} P(d) \qquad (20)$$

One alternative for solving Equation 19 is to use co-ordinate descent as shown in Algorithm 1.

```
Initialize all $\{\gamma_k, \mathbf{z}_d\}$ to starting values;
for $l + 1$-th iteration do
    ${\gamma_k}^{l+1}$ = Compute $\gamma_k$ using GMM $|({\gamma_k}^l, {\mathbf{z}_d}^l)$;
    ${\mathbf{z}_d}^{l+1}$ = Compute $\mathbf{z}_d$ $|({\gamma_k}^{l+1}, {\mathbf{z}_d}^l)$;
end
```

Algorithm 1: Co-ordinate Descent for PL Mixture Inference with GMM

While the above objective function 19 seems convex under assumption that the latent variables z_{kd} are constant, this model has a fundamental flaw: the $\mathbf{S}_k$ matrices are *singular* by construction. This implies that for any arbitrary set of z_{kd} parameters in the [feasible] parameter space the optimization will return the set of eigen vectors corresponding to the minimum possible value (i.e., 0). Therefore, this model will not work in theory and had to be abandoned. We implemented Algorithm 1 in MATLAB using the *cvx* library. When computing γ_k by fixing $\mathbf{z_d}$, the objective function always converges very close to 0 in one iteration of coordinate descent even before optimizing $\mathbf{z_d}$. Investigating why this was the case, we realized the above flaw in the model.

Nevertheless, the model works fine under the assumption that there is a single underlying preference (i.e., K=1). Therefore in this project we have compared this setting of GMM with the other algorithms.

VI. Experiment and Results

A. Dataset

Our objective is to detect anomalies. Therefore, for our experiments with real-world data, we selected a subset of the UCI Shuttle dataset which has 4000 total instances with 20 known anomalies. We used 20 independent density based (Gaussian mixture models) anomaly detectors to score the anomalousness of each instance based on probability density.

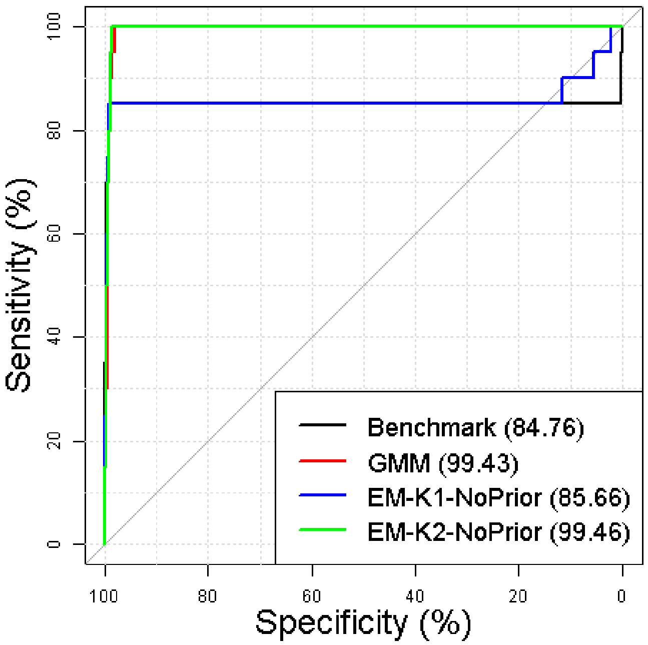


Fig. 2: AUC of the algorithms

B. Methodology

Four algorithms were used to rank each data instance based on the anomaly scores computed by the detectors:

- *EM-K1-NoPrior* : PL parameters estimated using *MM algorithm* with no Gamma priors and K=1
- *EM-K2-NoPrior* : PL parameters estimated using *MM algorithm* with no Gamma priors and K=2
- *Benchmark* : Items ranked in increasing order of the average probability density obtained from the detectors
- *GMM* : PL parameters estimated with GMM using K=1

For *EM-K1-NoPrior*, *EM-K2-NoPrior*, and *GMM* we first ranked each item based on probability density under each detector. The resulting set of 20 complete rankings were modeled as PL distributions for which the distribution parameters were computed. An item that has a higher value for the corresponding PL parameter is more anomalous than another item which has a lower value. Therefore, all items are finally ranked on the descending order of their PL distribution parameter values. For *EM-K2-NoPrior* the model returns two values for each item. The final ranking is based on the *max* of these.

The AUC is a commonly used measure for anomaly detection performance. Hence we have used the same for comparing the four algorithms.

C. Observations

First, we did not see any difference in performance in *MM algorithm* between using *Gamma* priors vs. not using the priors. This is likely because there were no set of items that were consistently ranked lower than other items. Due to this, we have only shown results in Fig. 2 for those algorithms which do not use priors.

Secondly, we note that the parameters inferred by *GMM* and *EM-K1-NoPrior* are different although both are trying to infer parameters for the same underlying distribution. This is not surprising since the number of detectors is only 20 and hence the true expectation of the pair-wise win/loss proportions will likely be different from that estimated from the data. Moreover, *EM-K1-NoPrior* is also prone to local optima. Looking at the individual detector rankings (Fig. 3), we found that three anomalous items were in some cases ranked very low (close to bottom) by four detectors and were otherwise ranked very high. These four detectors confused *EM-K1-NoPrior*. As illustrated in Fig. 4, *EM-K2-NoPrior* is able to overcome these confusing results and performs better.

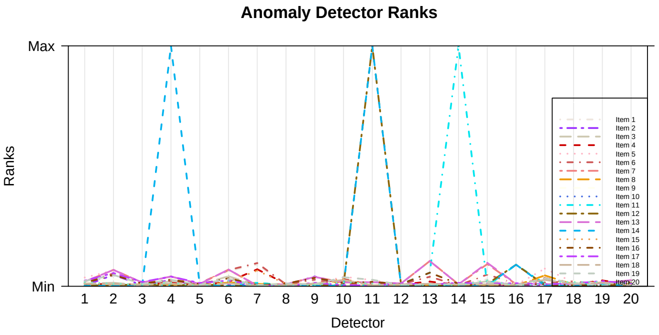


Fig. 3: Parallel Plot of Anomaly Ranks. Each unique colored line represents one anomaly. *Min* corresponds to Rank 1 and *Max* corresponds to Rank 4000. Item 11 (light blue) was ranked low by Detector 14, Item 12 (dark brown) was ranked low by Detector 11, and Item 14 (blue) was ranked low by Detectors 4 and 11. In all other cases, anomalies were ranked high by the detectors.

Finally, we note that contrary to what has been argued in [5], each iteration of the *MM algorithm* can be implemented in $O(DN^2)$ instead of (DN^3). This can be seen from Equation 15 where the summation in the denominator can be precomputed for each detector at the start of each *MM algorithm* iteration as a cumulative sum vector in $O(DN)$ time with a reasonable memory cost of $O(DN)$. While the *GMM* algorithm took around 25 minutes to execute, *EM-K2-NoPrior* took 16 minutes. *EM-K1-NoPrior* takes 3 minutes.

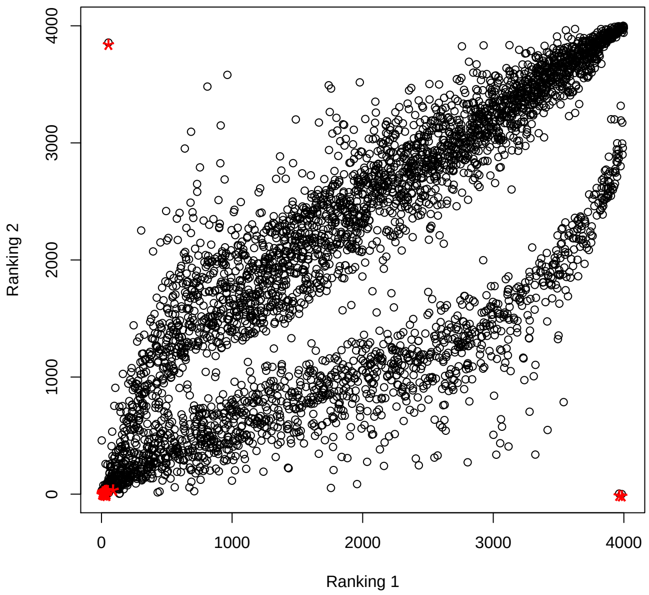


Fig. 4: Ranks under the two individual components in *EM-K2-NoPrior*. The points marked in red are anomalies. The overall rankings under both components of the mixture are very similar, however the three confusing anomaly items are ranked high under one component and low under the other. Therefore taking the best ranking for all items from either component improves the overall detection accuracy.

VII. Conclusion

We have been successful in showing the potential for improving anomaly rankings using PL distributions. In the process we have realized that the *MM algorithm* can be practical in a real-world scenario where the number of detectors is small and is competitive with GMM.

References

[1] R. L. Plackett. The analysis of permutations. *Applied Statistics*, 1975.
[2] Tie-Yan Liu. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3):225–331, March 2009.
[3] Murphy T. B. Gormley, I. C. Analysis of irish third-level college applications data, 2006.
[4] David R. Hunter, Kenneth Lange, Departments Of Biomathematics, and Human Genetics. A tutorial on mm algorithms. *Amer. Statist*, pages 30–37, 2004.
[5] Chen W. Parkes D. C. Xia L. Soufiani, H. A. Generalized method-of-moments for rank aggregation. *NIPS*, 2013.
[6] John Guiver and Edward Snelson. Bayesian inference for plackett-luce ranking models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 377–384, New York, NY, USA, 2009. ACM.
[7] R. D. Luce. *Individual Choice Behavior: A Theoretical Analysis*. 1959.

[8] Terry M. E. Bradley, R. A. Rank analysis of incomplete block designs. i. the method of paired comparisons, 1952.

[9] Tardella L. Mollica, C. Epitope profiling via mixture modeling of ranked data. *arXiv:1401.0404 [stat.ME]*, 2014.

[10] R. Hunter. Mm algorithms for generalized bradley-terry models. *The Annals of Statistics*, 32, 2004.