Detecting Extreme Rank Anomalous Collections

Hanbo Dai Feida Zhu Ee-Peng Lim Hwee Hwa Pang
School of Information Systems, Singapore Management University
{hanbo.dai.2008, fdzhu, eplim, hhpang}@smu.edu.sg

Abstract

Anomaly or outlier detection has a wide range of applications, including fraud and spam detection. Most existing studies focus on detecting point anomalies, i.e., individual, isolated entities. However, there is an increasing number of applications in which anomalies do not occur individually, but in small collections. Unlike the majority, entities in an anomalous collection tend to share certain extreme behavioral traits. The knowledge essential in understanding why and how the set of entities becomes outliers would only be revealed by examining at the collection level. A good example is web spammers adopting common spamming techniques. To discover this kind of anomalous collections, we introduce a novel definition of anomaly, called *Extreme Rank Anomalous Collection*. We propose a statistical model to quantify the anomalousness of such a collection, and present an exact as well as a heuristic algorithms for finding top-K extreme rank anomalous collections. We apply the algorithms on real Web spam data to detect spamming sites, and on IMDB data to detect unusual actor groups. Our algorithms achieve higher precisions compared to existing spam and anomaly detection methods. More importantly, our approach succeeds in finding meaningful anomalous collections in both datasets.

1 Introduction

When a set of entities are ranked by various features, a small subset of the entities could cluster toward the top or bottom ranks on some features. We call such an entity subset an *extreme rank anomalous collection* or *ERAC*. ERACs are prominent in many real-life applications.

Web spammers are good examples of ERACs. As reported in [13], [6] and [16], web spammers often adopt many spamming strategies to boost the ratings of their pages. For example, they stuff the pages with many popular keywords and anchor texts that are unrelated to one another. They may also generate pages from similar templates on the fly in order to perform "link spam". As a result, when measured by those characteristics or features, spammer hosts consistently demonstrate very extreme behaviors and form an identifiable anomalous collection, compared to normal web hosts.

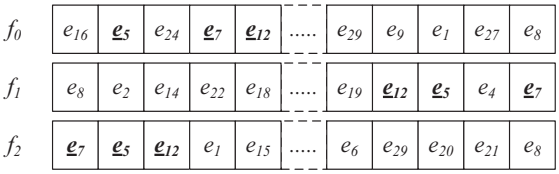

Figure 1: An example of ERAC. 30 entities $\{e_0, ..., e_{29}\}$ are ranked according to each 3 features $\{f_0, f_1, f_2\}$. In this example, $\{e_5, e_7, e_{12}\}$ is an ERAC

To illustrate, suppose we have 30 web hosts $\{e_0, ..., e_{29}\}$ as shown in Figure 1. The three host features $\{f_0, f_1, f_2\}$ reflect the aforementioned spamming strategies: f_0 represents the average number of popular keywords, f_1 is the variance of the word count, and f_2 captures the average fraction of anchor text. For each feature, we rank all the web hosts in descending order by their feature values. We can then identify $\{e_5, e_7, e_{12}\}$ as an ERAC because all of its entities appear at the top positions for features f_0 and f_2, and at the bottom positions for feature f_1. The fact that e_5, e_7 and e_{12} collectively display extreme behaviors across all the indicative features is strong evidence that they are likely to be web spammers.

As another example, ERACs can also be groups of fraudulent users in online marketplaces. A fraudulent user is likely to create many low-price transactions with other "accomplice" accounts in a very short time to gain credibility, before performing fraud transactions involving large sums of money [8] [21]. Consequently, they may be ranked at extreme positions with respect to features such as average number of transactions and transaction rate.

In the above examples, the features that find entities at extreme positions are not known ahead of time. This makes the ERAC detection challenging.

Existing anomaly detection approaches fall short in the above applications because they either focus on

single point anomalies, or they are not optimized to detect collections with extreme characteristics. Note that a set of single point anomalies does not always form an extreme rank anomalous collection, because neither does every entity in an ERAC appear at very extreme position, nor does every single point anomaly share common pattern with other anomalies. For example, in Figure 1, e_{12} is not very extreme by itself although it is part of an extreme rank anomalous collection $\{e_5, e_7, e_{12}\}$. In contrast, e_8 is quite extreme as it appears at extreme positions on all three features, but it does not form an extreme cluster with any other entities. Our problem therefore cannot be solved by simply grouping the single point anomalies found by existing approaches, as the effort to discover various common patterns among individual outliers is exponential in the number of outliers and the number of features.

After certain entities are found to form an ERAC, users can investigate their commonalities to understand the underlying reason. For example, ERACs could reveal the common spamming patterns and strategies adopted by different groups of spammers.

In this paper, we propose to model ERAC by the hypergeometric distribution. Our anomaly score is derived from the statistical p-value to measure how extremely ranked a collection is, which captures the following principles — A collection is more anomalous if: (I) it contains a larger number of entities that are ranked at more extreme positions on some feature; and (II) it contains entities that are consistently ranked at extreme positions across more features.

We summarize our contributions as follows:

- We propose a new kind of anomaly called extreme rank anomalous collection, together with a statistical model to measure the anomalousness of a collection by how extremely ranked it is with respect to any chosen feature set. With this model, we can explain statistically why a collection is anomalous, and how anomalous it is.
- We develop both exact and heuristic algorithms to find the top-K anomalous collections. We provide algorithm for coping with independent and dependent feature sets.
- We apply our approach on a web host graph to detect web spammers, and on the IMDB dataset to detect unusual actor groups. The results show that our approach finds meaningful anomalous collections in both data sets. For the web spam case with labeled true spammers, our approach discovers unique spammer collections while achieving higher precision, compared to existing works in both spam and anomaly detection.

The rest of the paper is organized as follows. Section 2 discusses related work. After introducing the problem formulation in Section 3, Section 4 presents our algorithms for both independent and dependent feature sets. Section 5 reports on experiments. Section 6 concludes the paper.

2 Related Work

Most of the studies on anomaly detection focus on point anomalies [7], using a variety of classification-based [6], distance-based [19] and density-based [5] approaches. Since these approaches assume that anomalies appear in sparse regions or are far away from the normal entities, true anomalous collections that are dense or are close to the normal entities may escape detection.

Clustering algorithms have been applied to detect both point anomalies and anomalous clusters. [12] and [14] detect point anomalies, on the assumption that they do not lie in any cluster. For anomalous cluster detection, [9], [17] and [20] assume that normal entities belong to large and dense clusters, whereas outliers form small or sparse clusters. Thus they apply a clustering algorithm on the data set, and find small clusters that are below some size threshold. In [9] and [17], small clusters are the smaller ones that together constitute less than 10% of the population. In [20], the threshold is half of the average cluster size. These clustering-based approaches require a threshold for the size of anomalous clusters, which is not trivial to set. Furthermore, while clustering aims to return entity clusters based on similarity, we focus on finding collections of extreme behaviors without an explicit distance function defined. We show later in the experiments that there exist ERACs that cannot be detected by the clustering objective of minimizing intra-ERAC/cluster distance while maximizing inter-ERAC/cluster distance.

Statistical techniques have also been applied to discover anomalies on the assumption that normal entities occur in high probability regions of a statistical model, while anomalies occur in the low probability regions. This technique utilizes statistical inference tests to determine if an entity is generated by the statistical model [3]. For example, Grubb's test and the student's t-test have been applied under the assumption that the data is generated by a Gaussian distribution. However, most of the existing studies mainly focus on statistical tests for a single data point, and are not suitable for modeling a collection of entities. Nevertheless, [2] uses a statistical model to detect in a network an anomalous cluster in which the node features have a different distribution than the rest of the nodes. The work assumes there is only one anomalous cluster in the network, and only one feature can be attached to each node.

Supervised approach is used in [18] to detect a group of anomalous objects, assuming the normal data is labeled beforehand. A Bayesian network is learned from the labeled normal data and another Bayesian network of the same structure but different parameters is also learned and assumed to generate outliers. However, since anomalous collections are often of small sizes, their outlier generation model trained during the search process would suffer from over-fitting.

3 Extreme Rank Anomalous Collection

Let E denote the universal entity set, and F a set of features. For an entity $e \in E$, $e.f$ denotes the feature value of e for feature $f \in F$. Entities can be ranked with respect to any feature. $rank_f(e)$ denotes the rank of entity e in E w.r.t. feature f, which is simplified to $rank(e)$ when the context is clear. For example, on feature f_1, $rank(e_8) = 1$ and $rank(e_7) = 30$ in Figure 1, assuming all feature values are distinct.

An entity may lie in top or bottom rank positions on different features. For the purpose of exposition, the following discussion focuses on the top rank position case. Our analysis extends readily to the bottom rank position case.

3.1 Measuring Anomalousness for A Single Feature We begin by measuring how anomalous an entity collection is w.r.t. a single feature. Our approach is based on the following principle: *on a given feature f*, an entity collection S is more anomalous, or more extremely ranked, if more entities in S appear in extreme regions of the ranked list of the universal entity set E w.r.t. f.

We use an **extremity index** r to refer to an extreme region, and denote as $S_f(r)$ the subset of entities in S which appear in top r rank positions w.r.t. feature f,

$$S_f(r) = \{e \mid e \in S, rank_f(e) \leq r\}$$

Continuing the example in Figure 1, for $S = \{e_5, e_7\}$ and $r = 2$, we have $S_{f_0}(r) = \{e_5\}$. Similarly, for $S = E$ and extremity index $r = 3$, $E_{f_1}(r) = \{e_8, e_2, e_{14}\}$. Note that $|E_f(r)| = r$ when the entities have distinct feature values. Where there is a tie, $|E_f(r)|$ may be larger than r.

The extremity of any given entity collection $S \subset E$ can be quantified by the cardinality of $S_f(r)$, which is a random variable following the hypergeometric distribution. This is because $S_f(r) = S \cap E_f(r)$, and if we randomly pick $|S|$ entities from $|E|$ entities, the number of entities in S that belong to $E_f(r)$ follows the hypergeometric distribution. Thus the probability of observing $|S_f(r)|$ common entities shared by S and $E_f(r)$ is:

$$prob(|S_f(r)|, |E|, |E_f(r)|, |S|) = \frac{\binom{|E_f(r)|}{|S_f(r)|} \cdot \binom{|E|-|E_f(r)|}{|S|-|S_f(r)|}}{\binom{|E|}{|S|}}$$

We now define the **p-value**[1] of S w.r.t. extremity index r and feature f, denoted as $p_f(S, r)$, as the probability of observing at least $|S_f(r)|$ common entities between a random collection of size $|S|$ and $E_f(r)$.

$$p_f(S, r) = \sum_{i=|S_f(r)|}^{\min(|E_f(r)|,|S|)} prob(i, |E|, |E_f(r)|, |S|)$$

Thus, a given S leads to different p-values with different r. Intuitively, for any given r and f, the smaller the p-value of S, the more extremely ranked S is w.r.t. r. Therefore, among all the choices of r, we pick the one which gives the smallest p-value. This particular r measures the maximum extremity of ranking that S could possibly have, which is by our definition also the maximum anomalousness of S w.r.t. f. Formally, we call this r the **representative extremity index** of S w.r.t. f, which is defined as:

$$r_f(S) = argmin_{0<r<|E|/2}\, p_f(S, r)$$

The choice of r ranges from 1 to half of the population of E, because we now focus on the top rank position case. The bottom rank position case, which we omit due to space constraint, is easy to derive similarly.

Correspondingly, the **representative p-value** of S w.r.t. f is denoted as $\widehat{p}_f(S)$, i.e., $\widehat{p}_f(S) = p_f(S, r_f(S))$.

Referring to Figure 1, for $S = \{e_5, e_7\}$, $r = 2$ and $r' = 4$, we have $|S_{f_0}(r)| = 1$ and $|S_{f_0}(r')| = 2$. Moreover, $p_{f_0}(S, r) = \sum_{i=1}^{\min(2,2)} prob(i, 30, 2, 2) = 0.131$ and $p_{f_0}(S, r') = \sum_{i=2}^{\min(4,2)} prob(i, 30, 4, 2) = 0.013$. As $p_{f_0}(S, r') < p_{f_0}(S, r)$, r' better represents how extremely ranked S is than r does for f_0. Suppose $r_{f_0}(S) = r' = 4$, we have the representative p-value of S w.r.t. f_0 as $\widehat{p}_{f_0}(\{e_5, e_7\})) = 0.013$.

Statistically, the representative p-value reflects how anomalous (i.e. extremely ranked) a collection S is w.r.t. a particular feature f. The smaller the representative p-value, the more anomalous a collection is. For example, in Figure 1, we compare $\{e_5, e_7\}$ with $\{e_7, e_{12}\}$ by their representative p-values for f_0. $\widehat{p}_{f_0}(\{e_7, e_{12}\}) = p_{f_0}(\{e_7, e_{12}\}) = 0.023$. Since $\widehat{p}_{f_0}(\{e_5, e_7\}) = 0.013$ is

[1] The p-value defined here is the right ended p-value of the hypergeometric distribution. The right ended p-value is used instead of the left ended one because in our case the more (instead of less for the left ended case) S overlaps with $E_f(r)$, the more extremely ranked S is w.r.t. f.

smaller than $\widehat{p}_{f_0}(\{e_7, e_{12}\})$, we conclude that $\{e_5, e_7\}$ is more anomalous than $\{e_7, e_{12}\}$ w.r.t. f_0; this is intuitive as $\{e_5, e_7\}$ sits more towards the top positions than $\{e_7, e_{12}\}$ on f_0. Note that the p-value measures directly at the collection level, which is different from measuring on the entity level followed by aggregating across the individual measures.

We are now ready to give our definition for an *Extreme Rank Anomalous Collection*.

DEFINITION 1. ***[Extreme Rank Anomalous Collection (ERAC)]*** *Given a universal entity set E and an entity set S s.t. $S \subset E, 1 < |S| < |E|/2$, a feature set F and a threshold α, we say S is an* Extreme Rank Anomalous Collection *w.r.t. F if $\exists f \in F$ s.t. $\widehat{p}_f(S) \leq \alpha$.*

The definition of ERAC corresponds to a global null hypothesis of multiple hypothesis tests, where each test is associated with one feature. We reject the null hypothesis that S is not anomalous w.r.t. any feature in F, if we witness a rare event happening in one test (i.e. on one feature) as indicated by a p-value that is smaller than a predefined significance level[2] α. Note that when we reject the null hypothesis and say that S is anomalous, S may not be significant for every test. We impose the condition $1 < |S| < |E|/2$, as an anomalous collection should contain more than one entity and yet remain the minority of the population.

3.2 Measuring Anomalousness for Multiple Features To measure how extreme an entity collection is ranked on a set of features, we generalize our principle as follows: *For a given feature set F, a collection is more anomalous if it is extremely ranked for more features in the given feature set.*

As the representative p-value measures how anomalous an ERAC is for a single feature, we define the **anomaly score** of an ERAC S for F as the product of the representative p-values for all the features in F. As the probabilistic value tend to be very small, we take the log form:

$$\Omega(S, F) = -\sum_{f \in F} \log \widehat{p}_f(S)$$

This definition is consistent with the principle that the more features S is extremely ranked against, the more anomalous it is.

We formulate our *top-K ERAC detection problem* as:

DEFINITION 2. ***[Top-K ERAC]*** *Given an entity universe set E, a feature set F, a target collection size N ($N < |E|/2$) and K, find the top-K most anomalous Extreme Rank Anomalous Collections of size at most N.*

We do not require the elements in different extreme rank anomalous collections to be mutually exclusive.

4 ERAC Detection Algorithms

To find the top-K ERACs of size at most N, a naive way is to enumerate all collections of size up to N, sort them by anomaly score in decreasing order, and return the top K ones in the ranking. It is easy to see the infeasibility of this approach as the search space is exponential in the size of the universe E, i.e., $\binom{|E|}{2} + ... + \binom{|E|}{N}$. Therefore we propose a bottom-up approach that successively generates larger ERAC candidates from smaller ones, maintaining a current top-K list and pruning unpromising growth paths whenever possible.

Out of this approach arise two main challenges. *(I) How to generate candidates from the current set of ERACs of smaller sizes?* We adopt a classic approach used in the Apriori Algorithm [1] to generate candidates for frequent item-sets. Candidates are generated level by level from single entities to collections of target size N. To generate a collection of size $n+1$, we only combine two collections of size n sharing the same first $n-1$ entities, assuming all entities are sorted alphabetically; this has been shown to ensure unique generation of each candidate collection. *(II) How to prune unnecessary growth paths and generate fewer candidates?* A crucial pruning technique used in Apriori is to prune any current candidate from future consideration whenever it is found to fall below the threshold set by the least one ranked in the current top-K list, thus avoiding potential traversal of the entire exponential search space. Unfortunately, in general, the anomaly score of ERACs does not enjoy this downward closure property to support the standard pruning strategy in top-K computation. Specifically, for any collection S, even when we find the anomaly score $\Omega(S, F)$ is less than the least one in the current top-K list, we cannot conclude that for all super-sets S' of S, the anomaly score $\Omega(S', F) \leq \Omega(S, F)$ and therefore safely prune S. The absence of this monotonicity on the anomaly score poses a difficulty to our bottom-up search approach. It seems that we have to keep all possible collections of size n to generate candidates of size $n+1$

[2] In order to control the type 1 error (false positive), the significance level for each individual test should be adjusted. Our model can accommodate all existing adjustment techniques including Bonferroni Correction, Holm-Bonferroni and Westfall-Young step-down. We adopt the Bonferroni Correction [10] to adjust the significance level to $\alpha/|F|$ to be conservative. For example, assuming 0.05 is the intended significance level for each single test, α would be set to $0.05/|F|$.

to guarantee the completeness of the mining result.

To tackle this problem, we develop two new pruning techniques based on different bounding techniques respectively in Sections 4.1 and 4.2. The former provides a more conservative and safe bound that leads to an exact algorithm *ERAC_E* which guarantees the completeness of the result. The latter derives a more aggressive and tighter bound used in a heuristic algorithm *ERAC_H* with good approximate result. The more general case of dependent features is discussed in Section 4.3.

4.1 The Exact Algorithm ERAC_E Given a collection S, despite the absence of monotonicity on the anomaly score precludes an upper-bound on $\Omega(S')$ for all super-sets S' of S, it is possible to derive an upper-bound on $\Omega(S')$ for those super-sets of S of a given size n. We denote this **upper-bound with size-constraint** n as $\hat{\Omega}(S, F, n)$, which will be formally defined shortly. Intuitively, the most anomalous super-set S' of S can be formed by adding to S exactly $|S'| - |S|$ entities which are ranked the most extreme with respect to S. Formally, given a collection S, size n, ($|S| < n \leq |E|/2$), and a feature f, the most anomalous super-set of S is defined as $\hat{S}_n(f) = S \cup S^*$, where $|S^*| = n - |S|$ and $\forall e' \in S^*, \forall e \in E \setminus \hat{S}_n, rank_f(e') \leq rank_f(e)$. To illustrate with the example in Figure 1, suppose $S = \{e_5, e_{12}\}$. Assuming no tie in the ranking, we have $\hat{S}_4(f_0) = \{e_{16}, e_5, e_{24}, e_{12}\}$ because, on f_0, e_{16} and e_{24} are the two entities ranked the most extreme to the top positions, excluding the entities already in S.

Accordingly, given F and S, the upper-bound with size-constraint n for S is defined as:

$$\widehat{\Omega}(S, F, n) = -\sum_{f \in F} \log \widehat{p}_f(\hat{S}_n(f))$$

Before we present the theorem to show that $\widehat{\Omega}(S, F, n)$ indeed represents the upper-bound on the anomaly score of all super-sets of S of size n, we need the following property of p-value.

PROPERTY 4.1. *Given any feature f, collections S and S' and extremity indices r and r', if $|S| = |S'|$, $|E_f(r)| = |E_f(r')|$ and $|S_f(r)| > |S'_f(r')|$, then $p_f(S, r) < p_f(S', r')$.*

Property 4.1 can be intuitively explained as, with all other parameters kept constant, the larger the number of entities in S that fall into the extreme positions, the smaller the p-value is. Now we have Theorem 4.1.

THEOREM 4.1. *Given S, $0 < |S| \leq |E|/2$, $\forall S'$ such that $S \subset S'$ and $|S'| \leq |E|/2$, we have $\Omega(S', F) \leq \widehat{\Omega}(S, F, |S'|)$*

Proof. Suppose S and S' are two collections s.t. $S \subset S'$. To prove $\Omega(S', F) = -\sum_{f \in F} \log \widehat{p}_f(S') \leq \widehat{\Omega}(S, F, |S'|) = -\sum_{f \in F} \log \widehat{p}_f(\hat{S}_{|S'|}(f))$, we need to show that for any f, $\widehat{p}_f(S') \geq \widehat{p}_f(\hat{S}_{|S'|}(f))$. Let $\widehat{S}$ denote $\hat{S}_{|S'|}(f)$.

Since $|\widehat{S}| = |S'|$, $E_f(r_f(S')) = E_f(r_f(S'))$ and $|\widehat{S} \cap E_f(r_f(S'))| \geq |S' \cap E_f(r_f(S'))|$. According to Property 4.1, we have $p_f(S', r_f(S')) \geq p_f(\widehat{S}, r_f(S'))$ By definition, $\forall 0 < r < |E|/2, p_f(\widehat{S}, r_f(\widehat{S})) \leq p_f(\widehat{S}, r)$. So we have $p_f(\widehat{S}, r_f(\widehat{S})) \leq p_f(\widehat{S}, r_f(S'))$. Thus $p_f(\widehat{S}, r_f(S')) \geq p_f(\widehat{S}, r_f(\widehat{S}))$. Therefore, we have $p_f(S', r_f(S')) \geq p_f(\hat{S}_{|S'|}(f), r_f(\hat{S}_{|S'|}(f)))$. Hence, $\Omega(S', F) \leq \widehat{\Omega}(S, F, |S'|)$.

Based on the upper-bound with size-constraint, we propose an exact algorithm to incorporate the following pruning strategy. At any time in the bottom-up search process, there are three data structures in the system: (I) the current top-K list; (II) the set of ERACs of size n, which are used to generate candidates of size $n+1$; and (III) the set of single entities which have yet to grow into any super-set yet. Denote Ω_t as the anomaly score of the least anomalous one in the current top-K list.

PRUNING TECHNIQUE 1. *When generating candidates of size $n+1$, only those collections S with anomaly score upper-bound for size-constraint $n+1$, i.e., $\hat{\Omega}(S, F, n+1)$, larger than Ω_t are grown.*

Why is this pruning strategy sound? Observe that due to the absence of the downward closure property, a standard Apriori-style bottom-up cannot drop any collection S even if its anomaly score falls below Ω_t. On the other hand, the upper-bound derived directly from the target size N, i.e., $\hat{\Omega}(S, F, N)$, would almost always prevail over Ω_t, leaving us with little pruning power if any. In contrast, with our pruning strategy, the upper-bound is computed with a size-constraint which increases consistently with the size of candidates being generated. Consequently, (I) when the size is small, an upper-bound with the same small size-constraint is more likely to fall below Ω_t; (II) as the size grows, Ω_t increases monotonically as well, continually pushing the bar higher for the upper-bound to beat. This accounts for the greater pruning power of our method. The *ERAC_E* algorithm for computing top-K ERACs of size up to N is shown in Algorithm 1, followed by a running example in Table 1 to illustrate the pruning techniques applied on the example in Figure 2.

Step 6 of Algorithm 1 applies Theorem 4.1, where collections with upper-bounds smaller than the threshold will not be included in $\mathbb{S}$, which maintains the set

Algorithm 1 *ERAC_E* for independent features

Input: E, F, K, N
Output: Top-K ERACs: $\mathbb{S}^*$
{$\mathbb{S}^*$ is implemented by a priority queue of max length K. Ω_t is the smallest anomaly score of collections in $\mathbb{S}^*$, and is set to zero if $\mathbb{S}^* = \emptyset$. $\mathbb{S}(i)$ is the current selected collections of size i, implemented by a hash tree.}

1: $n = 1$; $\mathbb{S}^* = \emptyset$; $\mathbb{S}(i) = \emptyset$, for $i = 1..N$; $\mathbb{S} = \{\{e\} \mid e \in E\}$
2: **while** $\mathbb{S} \neq \emptyset$ && $n < N$ **do**
3: update $\mathbb{S}^*$ by elements in $\mathbb{S}$ {Ω_t is updated accordingly}
4: $\mathbb{S} = \{\{e\} \mid e \in E\} - \mathbb{S}(1)$
5: **for** $i = 1$ to n **do**
6: $\mathbb{S} = \{S \in \mathbb{S} \mid \widehat{\Omega}(S, F, n+1) > \Omega_t\}$ {$\mathbb{S}$ now keeps the set of eligible collections}
7: $\mathbb{S}(i) = \mathbb{S} \cup \mathbb{S}(i)$
8: $\mathbb{S} = join(\mathbb{S}(i), \mathbb{S}(i))$ {generate new candidates of size $i+1$, and update $\mathbb{S}^*$ accordingly by elements in $\mathbb{S}$}
9: $n++$
10: **return** $\mathbb{S}^*$

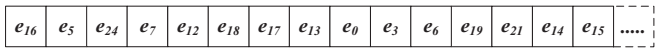

Figure 2: Top 15 entities ranked according to f_0 in Figure 1

of eligible collections. In Step 8 using $join()$, the supersets of collections with anomaly score upper-bounds smaller than the threshold are excluded from $\mathbb{S}$. The $join(\mathbb{S}(i), \mathbb{S}(i))$ function computes $S1 \bigotimes S2$ for each $(S1, S2)$ pair derived from $\mathbb{S}(i)$, where the operation $\bigotimes$ combines two size-i collections with identical $i-1$ elements to a size-$(i+1)$ collection (and is implemented similarly as in the Apriori Algorithm [1]).

Running Example. Figure 2 shows the top 15 entities ranked by f_0 out of a universe of 30 entities. Suppose $K = 1$ and N= 3. Table 1 shows the execution of Algorithm 1 with the changes in $\mathbb{S}(i)$, $\mathbb{S}^*$ and Ω_t.

When $n = 1$, $\mathbb{S}^*$ is updated to keep the current most anomalous collection after Step 3: $\{e_{16}\}$. Ω_t is updated to $\Omega(\{e_{16}\}, \{f_0\}) = -\log \widehat{p}(\{e_{16}\}) = 3.40$. At Step 4, since $\mathbb{S}(1) = \emptyset$, $\mathbb{S}$ contains all the singular sets. At Step 6, the set of selected collections of size 1 is $\mathbb{S} = \{\{e_{16}\}, \{e_5\}, \{e_{24}\}, \{e_7\}, \{e_{12}\}\}$ by comparing

n	$\mathbb{S}(i)$	$\mathbb{S}^*$ (Ω_t)
1	$\mathbb{S}(1) = \{\{e_{16}\}, \{e_5\}, \{e_{24}\}, \{e_7\}, \{e_{12}\}\}$	$\{\{e_{16}\}\}$(3.4)
2	$\mathbb{S}(1)$: same as above $\mathbb{S}(2) = \{\{e_{16}, e_5\}, \{e_{16}, e_{24}\}, \{e_{16}, e_7\}, \{e_{16}, e_{12}\}, \{e_5, e_{24}\}, \{e_5, e_7\}, \{e_5, e_{12}\}, \{e_{24}, e_7\}, \{e_{24}, e_{12}\}, \{e_7, e_{12}\}\}$	$\{\{e_{16}, e_5\}\}$ (6.08)
3	$\mathbb{S}(1)$, $\mathbb{S}(2)$: same as above $\mathbb{S}(3) = \{\{e_{16}, e_5, e_{24}\}, \{e_{16}, e_5, e_7\}, \{e_5, e_{24}, e_7\}\}$	$\{\{e_{16}, e_5, e_{24}\}\}$ (8.31)

Table 1: Algorithm 1 run on the example in Figure 2

their upper-bounds and Ω_t. $\{e_{18}\}$ is not eligible as $\widehat{\Omega}(\{e_{18}\}, \{f_0\}, 2) = -\log \widehat{p}(\{e_{16}, e_{18}\}) = 3.37 < \Omega_t$. In Step 8, we get $\mathbb{S}$ containing 10 collections. When $n = 2$, $\mathbb{S}^*$ is updated to $\{\{e_{16}, e_5\}\}$ and $\Omega_t = \Omega(\{e_{16}, e_5\}, \{f_0\}) = 6.08$. $\mathbb{S}$ contains the remaining singular collections from $\{e_{18}\}$ to $\{e_{15}\}$ along the ranked list. This time i goes from 1 to 2. When $i = 1$, at Step 6, the algorithm tries to pick out the previous left over singular entities that may be selected to generate collections of size 3. However, even $\widehat{\Omega}(\{e_{18}\}, \{f_0\}, 3) = -\log \widehat{p}(\{e_{16}, e_5, e_{18}\}) = 5.31 < \Omega_t$, meaning none of them is selected for now. When $i = 2$, in Step 6, we get $\mathbb{S} = \{\{e_{16}, e_5\}, \{e_{16}, e_{24}\}, \{e_{16}, e_7\}, \{e_5, e_{24}\}, \{e_5, e_7\}, \{e_{24}, e_7\}\}$. After executing the join function in Step 8, $\mathbb{S} = \{\{e_{16}, e_5, e_{24}\}, \{e_{16}, e_5, e_7\}, \{e_5, e_{24}, e_7\}\}$. Finally, we get the top-1 ERAC of size no greater than 3 on feature f_0: $\{e_{16}, e_5, e_{24}\}$ with anomaly score 8.31. Note that if we had set $N = 6$, previous left-over singular entities e_{18} would be selected and the corresponding nodes in the lattice tree would be grown from level 1 to level 6. Due to space limit, we only show the results for $N = 3$.

In this example, a standard Apriori algorithm without Pruning Strategy 1 would have to traverse the entire search space of all the candidates up to size 3, visiting altogether $\binom{15}{1} + \binom{15}{2} + \binom{15}{3} = 575$ nodes in the search lattice, due to the absence of the downward closure property. In comparison, we only visit 5+10+3=18 nodes in total, saving the visit to 96% of nodes even in this small example.

Efficient Anomaly Score Computation. By the definition of $\Omega(S, F)$, for a given S and each feature $f \in F$, we need to compute $\widehat{p}_f(S)$, which is decided by the representative extremity index of S w.r.t f. The naive approach to find this representative extremity index would examine all $|E|/2$ possible indices. However, the following property of the p-value allows us to avoid checking all extemity indices.

PROPERTY 4.2. *Given any feature f, collections S and S', and extremity indices r and r', if $|S| = |S'|$, $|S_f(r)| = |S'_f(r')|$ and $|E_f(r)| > |E_f(r')|$, then $p_f(S, r) > p_f(S', r')$.*

Property 4.2 suggests that with all other parameters kept constant, the smaller the extremity index, the smaller the p-value is.

With Property 4.2, we check only those extremity indices "indicated" by the rank positions of entities in S. For example, in Figure 1, for $S = \{e_5, e_7\}$ and f_0, we just need to check $r = 2$ and $r' = 4$ corresponding to the positions of e5 and e7 respectively. The other extremity indices can be skipped. Take

$r'' = 3$ for example, as $|S_{f_0}(r'')| = |S_{f_0}(r)|$ but $|E_f(r'')| = 3 > |E_f(r)| = 2$, according to Property 4.2, we have $p_{f_0}(S, r) < p_{f_0}(S, r'')$. Thus, we do not need to consider r''. Extremity indices that do not correspond to the rank of any entity in S have larger p-values than the extremity index corresponding to the rank of some entity in S. Therefore, to compute the representative p-value $\widehat{p}_f(S)$ for each $f \in F$, we examine only $O(|S|)$ extremity indices instead of $O(|E|)$.

It can be shown the total running time of Algorithm 1 is of $O(|\overline{\mathbb{S}}|^2 + |\overline{\mathbb{S}}| \cdot N^4 \cdot |F|)$, where $|\overline{\mathbb{S}}|$ denote the average size of $\mathbb{S}(i)$ for all i and all n.

4.2 The Heuristic Algorithm: ERAC_H The bound given in Section 4.1 could be rather loose and gives little pruning power. We now show another pruning technique by exploiting our Apriori-style candidate generation.

As stated in Section 3, given a collection S, the p-value of S is determined by the extremity index r ($r < |E|/2$) and the number of entities in S that appear in $E_f(r)$, i.e., $|S_f(r)|$. We denote $|S_f(r)|$ as i. For any given p-value and size n, we can explicitly express the underlying i and r according to the p-value formula by denoting the p-value as $p_f(i, r, n)$, or just $p(i, r, n)$ when the context is clear.

Let S_x be the ERAC realizing x (i.e., $\Omega(S_x, F) = x$). Since the anomaly score is the sum of the negative logarithm of the representative p-values of every feature, we want to derive for each feature $f \in F$, the corresponding representative p-value $p_f(i^x, r^x, n)$ for S_x. From there, we check whether, for feature f, combining S_1 and S_2 will achieve a representative p-value that is even smaller than $p_f(i^x, r^x, n)$. If so, combining S_1 and S_2 will achieve a higher anomaly score than x. We will discuss how to decompose x into a set of p-values later in this section. For now, we assume $p_f(i^x, r^x, n)$ is known for any f.

For any feature f, given any representative p-value $p_f(i^x, r^x, n)$ of some collection S of size n, we take the heuristic that the representative p-values of S's subsets of size n' that generate S is at most $p_f(i^x - (n - n'), r^x, n')$. This heuristic is based on some properties of p-value, which are omitted here due to space limit. Given an anomaly score x of any size-n collection and a collection size $n', n' < n$, we define the **lower-cut with score-size constraint** (x, n) as $\Omega^*(x, n, n') = -\sum_{f \in F} \log p_f(i^x - (n - n'), r^x, n')$.

We have the following pruning heuristic.

PRUNING TECHNIQUE 2. *Given an anomaly score threshold x for collections of size n, and its correspondent representative p-value $p_f(i^x, r^x, n)$ for each feature $f \in F$, we can use $p_f(i^x - (n - n'), r^x, n')$ to derive the lower-cut $\Omega^*(x, n, n')$ and prune away any collection of size $n', n' < n$, such that its anomaly score is smaller than $\Omega^*(x, n, n')$.*

With this pruning heuristic, our heuristic algorithm works as follows. Given a target size N, we first estimate a threshold Ω_N of collections of size N by the anomaly score of the collection comprising the top-N anomalous singular entities, which is a valid candidate collection to be used as a reasonable start. It is possible that this estimated initial bound Ω_N is too aggressive and is even higher than the true Ω_t of the final top-K result. To remedy, we first run with this initial Ω_N to obtain a preliminary top-K result, then set the smaller one between the initial Ω_N and this preliminary Ω_t as the new threshold Ω_N to reboot the algorithm.

With this new threshold Ω_N and its correspondent set of $p_f(i^x, r^x, n)$, we compute the sequence of lower-cuts $\Omega^*(\Omega_N, N, i)$ for all levels $1 \leq i < N$. Another trick is that, when we are to generate candidates of size n, it could be that the anomaly score of the least one in the current top-K list (i.e., Ω_t) can provide a better cut i.e., $\Omega^*(\Omega_t, n+1, n) > \Omega^*(\Omega_N, N, n)$. Using the better cut, we can prune away current ERACs of size n before trying to combine any two of them to generate candidates of size $n+1$. The details are shown in Algorithm 2.

Algorithm 2 *ERAC_H* for independent features

Input: E, F, K, N
Output: Top-K ERACs: $\mathbb{S}^*$
{$\mathbb{S}^*$ is implemented by a priority queue of max length K. Ω_t is the smallest anomaly score of collections in $\mathbb{S}^*$, and is set to zero if $\mathbb{S}^* = \emptyset$ }

1: $n = 1$; $\mathbb{S}^* = \emptyset$; $\mathbb{S} = \{\{e\} \mid e \in E\}$
2: $\Omega_N = \Omega(S_N, F)$, where S_N is the union of the top-N anomalous elements in $\mathbb{S}$.
3: **repeat**
4: $\quad \Omega'_N = \Omega_N$
5: $\quad$ **for** $n = 1$ to $N - 1$ **do**
6: $\quad\quad \mathbb{S} = \{S \in \mathbb{S} \mid \Omega(S, F) > \max(\Omega^*(\Omega_N, N, n), \Omega^*(\Omega_t, n+1, n))\}$
7: $\quad\quad \mathbb{S} = join(\mathbb{S}, \mathbb{S})$ {$\mathbb{S}^*$ and Ω_t are updated by elements in $\mathbb{S}$ whenever necessary}
8: $\quad \Omega_N = \Omega_t$
9: **until** $\Omega_N \geq \Omega'_N$
10: **return** $\mathbb{S}^*$

Running Example. Setting $K = 1$ and $N = 3$ again, we show how Algorithm 2 executes on the same example in Figure 2. The algorithm estimates Ω_N as $\Omega(\{e_{16}, e_5, e_{24}\}, \{f_0\}) = 8.31$, since these three entities are the top-3 anomalous singular entities. When $n = 1$, the algorithm checks whether each collection of size 1 can beat the current Ω_t by Pruning Technique 2. As $\Omega^*(8.31, 3, 1) = -\log p(1, 3, 1)$, only $\{e_{16}\}$,$\{e_5\}$ and

$\{e_{24}\}$ are selected to generate collections of size 2. After the join step, $\mathbb{S} = \{\{e_{16}, e_5\}, \{e_{16}, e_{24}\}, \{e_5, e_{24}\}\}$. The algorithm goes on to find $\{e_{16}, e_5\}$ as the current top-1 ERAC. When $n = 2$, Step 7 finds all elements in the current $\mathbb{S}$ are eligible to generate collections of size 3. The join() generates the collection $\{e_{16}, e_5, e_{24}\}$ and keeps it as the top-1 ERAC. Since the current Ω_t is equal to the estimated threshold $\Omega(\{e_{16}, e_5, e_{24}\}, \{f_0\})$, the algorithm stops and returns $\{e_{16}, e_5, e_{24}\}$ as the final result.

For the same example, we have shown that the exact algorithm *ERAC_E* visits 18 nodes in total, whereas the heuristic algorithm *ERAC_H* only visits 3+3+1=7 nodes, saving the visit to 61% of nodes over the exact algorithm.

We now analyze the time complexity of Algorithm 2. Let $|\overline{\mathbb{S}'}|$ denote the average size of $\mathbb{S}$ for all n. The time complexity of Algorithm 2 is of $O(|\overline{\mathbb{S}'}|^2 + |\overline{\mathbb{S}'}| \cdot N^3 \cdot |F|)$, lower than that of Algorithm 1 as $N^3 < N^4$ and as we expect $|\overline{\mathbb{S}'}|$ in Algorithm 2 to be much smaller than the counterparts $|\overline{\mathbb{S}}|$ in Algorithm 1.

Anomaly Score Decomposition. We now discuss how to derive representative value $p_f(i^x, r^x, n)$ for each feature $f \in F$ from a given anomaly score x. Ideally, we aim to obtain the set of p-values that minimize $\Omega^*(x, n, n')$, subject to (1) $\mathbb{X} \leq -\sum_{f \in F} \log x_f$; (2) $\nexists i' < i^x$ with $p(i', r', n) < p(i^x, r^x, n)$.

For efficiency purpose, we approximate the minimum value of $\Omega^*(x, n, n')$ by assuming the anomaly score x is evenly divided across features up to a degree of approximation. We first compute $e^{-\frac{x}{|F|}}$, then we find the $p_f(i^x, r^x, n)$ such that (1) $p_f(i^x, r^x, n) \leq e^{-\frac{x}{|F|}}$; (2) $\nexists i' < i^x$ with $p(i', r', n) < p(i^x, r^x, n)$; (3) $\nexists r' < r^x$ with $p(i^x, r', n) < e^{-\frac{x}{|F|}}$. Condition (1) guarantees $x \leq -\sum_{f \in F} \log x_f$. Condition (2) is required in the heuristic. Condition (3) is based on Property 4.2. As i^x is fixed, we choose the larger r^x so that the corresponding $p(i^x - (n - n'), r^x, n')$ is larger, which in turn corresponds to a smaller anomaly score.

4.3 Handling Dependent Features

We have so far assumed that features in F are independent of one other. Now we handle the more general case of dependant features. A feature set F is said to be an Independent Feature Set or IFS, if $|F| > 1$ and $\forall f_i, f_j \in F$, f_i is independent of f_j. However, it is not necessary to compute the anomaly score of a collection for every IFS, as many of them are subsumed by others. We hence focus on maximal IFSs. An IFS F is a **maximal IFS** if $\nexists$ an IFS F' s.t. $F \subset F'$. Given a feature set F, the set of all maximal independent feature sets derived from F is denoted as $\mathbb{F}$.

In Section 3, the anomaly score is measured on one independent feature set. Given $\mathbb{F}$, the overall anomaly score of S is aggregated by taking the largest anomaly score across all maximal IFSs in $\mathbb{F}$ as $\Omega(S) = \max_{F' \in \mathbb{F}} \Omega(S, F')$.

We now detect ERACs for any dependent feature set F based on the algorithms for independent features. We denote the algorithm using *ERAC_E* as *ERACD_E* and the one using *ERAC_H* as *ERACD_H*. In both *ERACD_E* and *ERACD_H*, we first generate all maximal IFS from F. For each maximal IFS, we call *ERAC_E* or *ERAC_H*. Finally, we get the top-K collections of size up to N by aggregating the top-K collections across all maximal IFSs. Since we are presented with multiple maximal IFSs, we return not only the ERACs but also their corresponding maximal IFSs. We still use a priority queue of maximum length of K as before, but with composite elements $\langle S, F' \rangle$. Each element keeps the anomalous ERACs and the corresponding maximal IFSs that make S most anomalous. We constrain the priority queue to have distinct collections only. As to compute all maximal IFSs $\mathbb{F}$ is NP-hard [22], we adopt the algorithm in [11] for an approximation.

5 Experiments

In this section, we apply our ERAC detection framework on a web host graph to detect web spam collections, and on an IMDB dataset to detect anomalous actor groups. We show the effectiveness and efficiency of our proposed algorithms. In the case of the web spam where the ground truth is available, we compare the precision of our framework with the existing web spam detection approach, and anomaly detection approaches. We lastly show the results on IMDB dataset.

5.1 Detecting Web Spam Collections

As reported in [4], spammers often try to game the search result ranking by fabricating incoming links from link farms, which are usually also spammers deploying the same spamming strategies. Moreover, these incoming spammer pages are often created by the same web page template at a very low cost.

As a result, the incoming neighborhoods of spammers are extremely homogeneous or heterogeneous compared to those of normal ones that are gradually built up. In other words, web spammers are expected to be ranked at top or bottom by their incoming neighborhood's homogeneity.

Given a node e, we define the incoming **Neighborhood Feature** or iNF as:
$iNF(f, e) = median_{e', e'' \in iNBR(e), e' \neq e'' \neq e} dist(e'.f, e''.f)$,
where $dist(e'.f, e''.f) = |e'.f - e''.f|$, and $iNBR(e)$ denote node e's 1-hop incoming neighborhood.

Intuitively, a node with a small iNF value has a more homogeneous incoming neighborhood, whereas a node with a large iNF value has a more heterogeneous incoming neighborhood.

We extract our web host graph from WEBSPAM-UK2006[3] published by Yahoo! Research Barcelona. We adopt 96 content features calculated by [6] [4], where the features of a host are represented by its home page as well as the page with the highest PageRank score on the host. We compute 6 structural features at the host level, including the number of 1-hop and 2-hop incoming neighbors. These features are further processed to derive the corresponding neighborhood features. We iteratively remove entities with less than 2 incoming neighbors, assuming they are not spammers. This leaves one big connected web host graph, with 5634 nodes (1709 spammers) associated with 102 features.

We first compare the top-K ERACs found by our exact and heuristic algorithms in Section 4, then demonstrate the effectiveness of our approach by comparing to other existing techniques.

5.1.1 Exact Versus Heuristic Algorithms As the exact algorithm takes a long time to execute on the entire graph, we sample several subgraphs for comparing the exact and heuristic algorithms. We are interested to know how different they are, in terms of the ERACs returned and efficiency.

We extract subgraphs by uniformly sampling the edges at random so that the density of connected components is preserved. We extract four groups of subgraphs, with each group having five subgraphs generated by 50, 80, 100 and 150 random edges respectively. For each subgraph, we iteratively remove entities with less than 2 incoming neighbors as mentioned before. The average number of nodes of the resultant graphs for each setting is shown in Table 2. We then extract maximal IFSs for each subgraph and randomly select one maximal IFS, with which we apply both the exact and heuristic algorithms. As for the other parameters, we empirically set N to 3, K to one fifth of the number of nodes in the sample.

To evaluate how close the results of the two algorithms are, we compute the overlap between the top-K ERACs $\mathbb{S}^*_{exact}$ and $\mathbb{S}^*_{heuristic}$ as $\frac{|\mathbb{S}^*_{exact} \cap \mathbb{S}^*_{heuristic}|}{K}$. To evaluate efficiency, we define the speed-up as the ratio of the execution time of the exact algorithm to that of the heuristic one.

The average overlap and average speed-up is shown in Table 2. As we see in the table, the heuristic algorithm produces almost identical top-K ERACs as the exact algorithm, in a much shorter time.

Table 2: Average overlap and speed-up

# of sampled edges (Avg # of nodes)	Avg Speed-up	Avg Overlap
50 (81)	51	0.99
80 (135)	101	0.99
100 (182)	110	0.99
150 (246)	154	1.0

5.1.2 Effectiveness of ERACD_H Since some of the 102 features in the web spam data set are dependent on each other, we apply the algorithm designed for dependent feature set to compute the top-K ERACs of size no greater than N. As the exact algorithm is too costly to run on the web host graph, we show results from the heuristic algorithm $ERACD\_H$, which should output almost identical results as the exact one. We set K=1000, around one fifth of the total number of hosts as we did before and set N=12. We keep N small as anomalous collections normally are not large.

Applying the $ERACD\_H$ algorithm, we formed 258 maximal IFSs from the 102 features. Among the maximal IFSs, 21 of them are of size 3, the largest size of all. The top most ERAC returned by $ERACD\_H$ identifies 12 hosts, all true spammers (including *www.englandguide.co.uk* and *www.posters.co.uk* that are still actively spamming despite being labeled as spammers in 2006). This collection is associated with the maximal IFS of {"Number of words", "Top 100 corpus precision", "Independent LH" }, where corpus precision refers to the fraction of words that appear in the set of popular terms, and Independent LH is a measure of the independence of the distribution of trigrams.

Now we look further into the representative extremity indices of this collection, and explain why it is anomalous. It turns out that the web hosts in this collection are clustered in the top 18 positions on "Number of words", top 22 on "Top 100 corpus precision" and top 45 on "Independent LH". This means that the neighborhood of each host in this collection is very homogeneous in terms of number of words, tendency to use very popular keywords, and pattern of using many unrelated keywords. Therefore our approach is able to discover true spammer collections as well as explain why they are anomalous.

5.1.3 Comparison to Spam Detection Approaches Next, we compare our $ERACD\_H$ algorithm, denoted as ERACD, with the unsupervised TrustRank [16] [15] and the supervised decision tree techniques employed in [6] and [4]. These spam detection approaches

[3] http://barcelona.research.yahoo.net/webspam/datasets

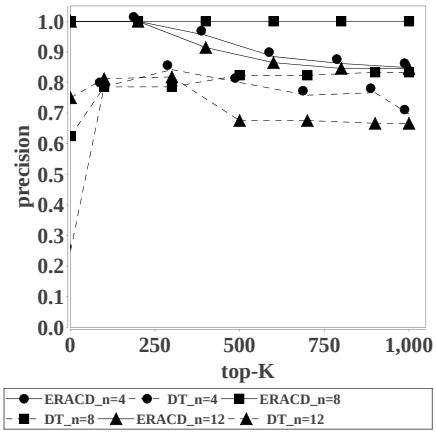

Figure 3: ERACD v.s.DT

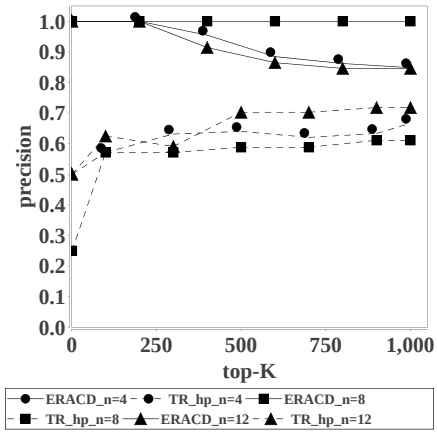

Figure 4: ERACD v.s.TR_hp

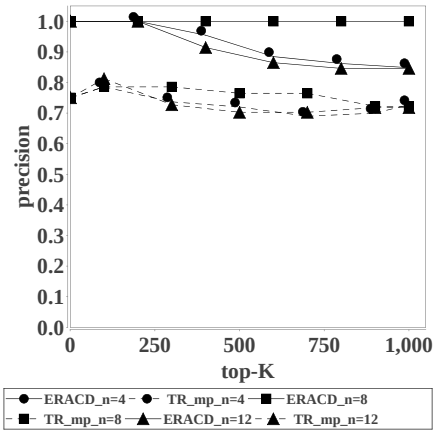

Figure 5: ERACD v.s.TR_mp

aim at detecting individual spammers, not spam collections. We therefore assume that all the entities in top-K ERACs are spammers, and compare the precision of the competing methods.

To compute the trustrank score, we follow the guidance of [15] and [6].We then rank web hosts in ascending order by the trustrank score of their respective home page (hp), as well as the page with the highest page rank (mp). Since the lower the trustrank score of a node, the more likely it is a spammer, the hosts at the top are more likely to be spammers. We denote the approach involving home pages as TR_hp and the ranked list it produces as $EL(\mathsf{TR\_hp})$. The approach involving the pages with the highest page rank is denoted as TR_mp and its ranked list as $EL(\mathsf{TR\_mp})$.

For the decision tree denoted as DT, we use J48 of Weka[4] with 5-fold cross validation. The features used are the neighborhood features used in ERACD. To derive a ranked list of web hosts for comparing with other approaches, we sort the hosts in descending order of the prediction values assigned to them by the decision tree. The ranked list is denoted as $EL(\mathsf{DT})$.

Since our heuristic algorithm works incrementally, in each loop of n from 1 to N, it outputs the top-K ERACs of size no greater than n denoted as $\mathbb{S}^*(n, K)$. We therefore are able to compare the ERACD approach with collections of various sizes by keeping the intermediate top-K ERACs results of *ERACD_H* for various $n \leq N$. In the experiments, we try $n \in \{4, 8, 12\}$.

Given n and K, let $\tau(n, K) = |\bigcup_{e \in \bigcup_{S \in \mathbb{S}^*(n,K)} e}|$ denote the number of distinct entities in $\mathbb{S}^*(n, K)$. We assume that all entities in $\mathbb{S}^*(n, K)$ are spammers and compare the top $\tau(n, K)$ entities of each approach. Let $EL(O, \tau(n, K))$ denote the top $\tau(n, K)$ entities returned by approach $O \in \{\mathsf{ERACD}, \mathsf{TR\_hp}, \mathsf{TR\_mp}, \mathsf{DT}\}$. The precision of the top $\tau(n, K)$ entities is defined as: $\frac{|EL(O,\tau(n,K)) \cap \Gamma|}{|EL(O,\tau(n,K))|}$, where Γ denotes the set of true spammers in E.

Figures 3, 4 and 5 plot the precision against K for ERACD versus {DT, TR_hp and TR_mp}. In the figures, ERACD is represented by the solid line, while the competing approaches are in doted lines.

As we observe from the plots, ERACD outperforms DT, TR_hp and TR_mp across the n settings. This demonstrates that our approach, although not specifically designed for detecting spammers, still outperforms the other methods in precision.

The recall levels achieved by our approach are all around 0.03 for $n = 4, 8, 12$ respectively and with $K = 1000$. The low recall levels are expected, as our approach is intended for discovering the most anomalous collections of entities, with no attempt to avoid overlap between collections.

Next, to see whether our approach finds unique spammers, we check the overlap between the top $\tau(n, K)$ entities returned by each pair of approaches. Specifically, the overlap ratio is $\frac{|EL(\mathsf{ERACD},\tau(n,K)) \cap EL(O,\tau(n,K)|}{\tau(n,K)}$, where $O \in \{\mathsf{TR\_hp}, \mathsf{TR\_mp}, \mathsf{DT}\}$. For $K = 1000$ and $n \in \{4, 8, 12\}$, all the overlap ratios are smaller than 0.05, suggesting ERACD detects unique spammer hosts that are missed by the competing methods.

5.1.4 Comparison to Anomaly Detection Approaches Having compared our approach with spammer detection approaches that produce point anomalies, we compare with the general density-based and clustering-based anomaly detection approaches. Note that the density-based approach returns point anoma-

[4]www.cs.waikato.ac.nz/ml/weka

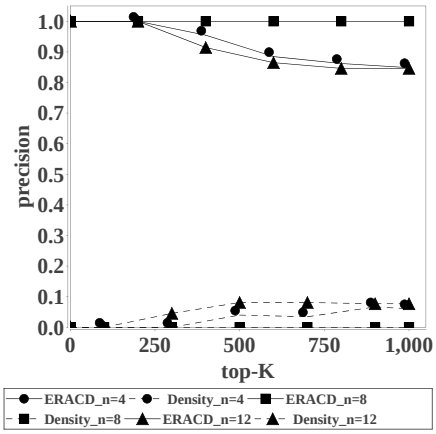

Figure 6: ERACD v.s. Density-based method on web spam dataset

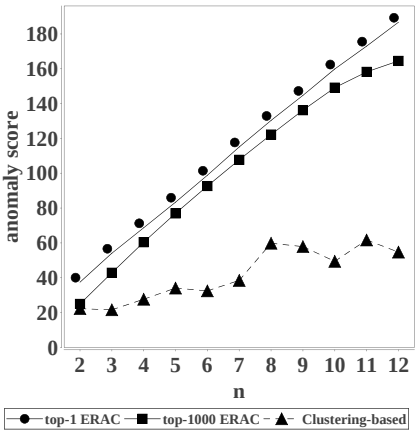

Figure 7: ERACD v.s. Clustering-based method on web spam dataset

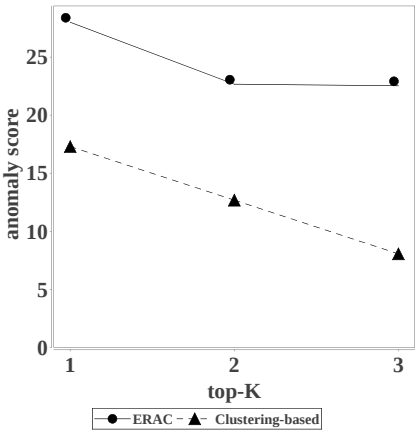

Figure 8: ERACD v.s. Clustering-based method on IMDB dataset

lies whereas the clustering-based one returns collections of entities. We are interested to know whether the anomaly score and precision of the collections returned by our approach are better.

Figure 6 shows the comparison with the density-based approach [5]. The top entities returned by the density-based approach are mostly non-spammers. This is because the density-based approach assumes anomalies appear in sparse regions. However, true spammers are likely to employ common spamming tricks, making them less likely to appear in sparse regions of the feature space.

Next, we consider clustering-based outlier detection approaches, which consider "small clusters" to be anomalous. We follow [20] in defining a small cluster as one with a size that is smaller than half of the average cluster size.

We apply agglomerative hierarchical clustering with complete link and Euclidian distance to cluster E. The clustering algorithm is run on each maximal IFS calculated by our approach so as to find the most anomalous cluster across all maximal IFSs. In clustering for a given maximal IFS, we stop growing the cluster tree if any two lower level clusters being combined makes the average size of all clusters larger than $2 \cdot N$. This is to make sure that all resultant clusters of size smaller or equal to N are "small" clusters by the definition of the clustering-based approach. Since the clusters returned by our approach are of size no greater than N, we can make a fair comparison with the "small" clusters returned by the clustering-based approach.

In this comparison, we keep the intermediate top-K results when n varies, and show the top-1 ERAC and top-1000 ERAC of ERACD for each n together with the most anomalous cluster discovered by the clustering-based approach.

In Figure 7, we see that for all n, the collections discovered by our approach are more anomalous than the clustering-based ones, because our ERACD is optimized to find clusters sitting at extreme positions.

As for precision, we compute first for each given maximal IFS, the ratio of the number of true spammers in all the small clusters over the total number of entities in all the small clusters. The maximal ratio is selected as the precision across all maximal IFSs. We also compute this precision for all n from 2 to 12. The results show that the precision of the small clusters is quite low for each n, with a maximum of 0.35 at n=10. This suggests that most of the entities in small clusters are not necessarily spammers. Thus we cannot rely on clustering-based anomaly detection techniques to find spammer collections.

5.2 Detecting Anomalous Collections in IMDB

From the IMDB data set obtained from its portal[5], we extracted actors and actresses participating in movies shown between 1990 and 2008. We extract actors playing non-trivial roles in each movie by taking only those appearing among the top 10 names in the cast list. We extracted 6 actor features including number of movies, average rating of all movies, average salary, average movie budget, average movie box office and average payback (the ratio of average box office to average salary). We remove actors with missing feature values. In the end, we get altogether 183 actors.

We apply our ERACD on this preprocessed IMDB dataset. The maximal independent feature set generated consists of {number of movies, average movie bud-

[5] http://www.imdb.com/interfaces

get, average payback}. With this maximal IFS, we also run the clustering-based approach on this IMDB dataset. We set $N = 3$, meaning the clusters returned that are of size no greater than 3 are considered anomalous by the clustering-based approach. Altogether, the clustering-based approach returned 29 clusters, including 3 anomalous ones. Figure 8 shows that the top-3 ERACs returned by ERACD are more anomalous than those returned by the clustering-based approach.

Interestingly, the top-1st ERAC returned by our ERACD is {Grint Rupert, Radcliffe Daniel, Watson Emma}, containing the three major actors in Harry Potter movies. The most anomalous cluster returned by the clustering-based approach is {Lohan Lindsay, Witherspoon John, Madonna}. We plot their relative rankings along the three features in Figure 9. The center point of each star chart represents the middle ranking. The distance to the center point represents the extremity. Therefore, the further away an actor's ranking is from the center point along a feature, the more extremely ranked he or she is w.r.t that feature. By visual inspection, we can see that {Grint Rupert, Radcliffe Daniel, Watson Emma} is anomalous because the actors in the collection are consistently ranked extremely in all three features. However, the collection returned by the clustering-based approach is less anomalous. This illustrates again that, while clustering-based approaches could return sets of similar entities, they are not designed to capture those interesting collections exhibiting extreme behavior as in our problem setting.

Our findings are useful especially when groups of extremely ranked actors w.r.t a certain combination of features are of interest. For example, a movie producer may want to find actors who perform in few large-budget movies but have big payback. Our results suggest the producer should go for the top ERACs.

6 Conclusions

In this paper, we introduce the problem of discovering extreme rank anomalous collections (ERAC). We proposed both exact and heuristic algorithms for detecting top-K ERACs, for both independent and dependent feature sets. We applied our approach to detect web spammers in web host graph as well as anomalous actor groups in IMDB. The results on both data sets showed that the top ERACs are meaningful. Interestingly, the anomalous entities discovered by our approach are largely distinct from those found with existing methods. Moreover, for web spam detection, our approach not only detects web spammer collections with higher precision than existing approaches, but also explains the anomaly statistically.

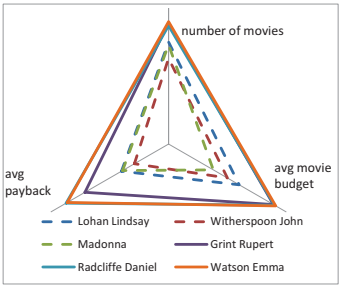

Figure 9: top-1st ERACD v.s. top-1st cluster by Clustering-based method

References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *VLDB Conf.*, 1994.

[2] E. Arias-Castro, E. J. Cands, and A. Durand. Detection of an anomalous cluster in a network. *Ann. Statist.*, 39(1), 2011.

[3] V. Barnett and T. Lewis. *Outliers in statistical data*. John Wiley and Sons, 1994.

[4] L. Becchetti, C. Castillo, D. Donato, R. Baeza-YATES, and S. Leonardi. Link analysis for web spam detection. *ACM Trans. Web*, 2(1), 2008.

[5] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: identifying density-based local outliers. In *SIGMOD Conf.*, 2000.

[6] C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri. Know your neighbors: web spam detection using the web topology. In *SIGIR Conf.*, 2007.

[7] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3), 2009.

[8] C. E. H. Chua and J. Wareham. Fighting internet auction fraud: An assessment and proposal. *Computer*, 37(10), 2004.

[9] L. Duan, L. Xu, Y. Liu, and J. Lee. Cluster-based outlier detection. *Annals of Operations Research*, 168(1), 2009.

[10] C. W. Dunnett. A multiple comparison procedure for comparing several treatments with a control. *Journal of the American Statistical Association*, 50(272), 1955.

[11] D. Eppstein. All maximal independent sets and dynamic dominance for sparse graphs. In *SODA Conf.*, 2005.

[12] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *ICDM Conf.*, 1996.

[13] D. Fetterly, M. Manasse, and M. Najork. Spam, damn spam, and statistics: using statistical analysis to locate spam web pages. In *Proc. of the 7th International Workshop on the Web and Databases*, 2004.

[14] S. Guha, R. Rastogi, and K. Shim. Rock: A robust clustering algorithm for categorical attributes. In *ICDE Conf.*, 1999.

[15] Z. Gyöngyi, P. Berkhin, H. Garcia-Molina, and J. Pedersen. Link spam detection based on mass estimation. In *VLDB Conf.*, 2006.

[16] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with trustrank. In *VLDB Conf.*, 2004.

[17] Z. He, X. Xu, and S. Deng. Discovering cluster-based local outliers. *Pattern Recogn. Lett.*, 24(9), 2003.

[18] D. Kaustav, S. Jeff, and B. N. Daniel. *Detecting Anomalous Groups in Categorical Datasets*. CMU Technical Report, 2009.

[19] E. M. Knorr and R. T. Ng. Algorithms for mining distance-based outliers in large datasets. In *VLDB Conf.*, 1998.

[20] A. Loureiro, L. Torgo, and C. Soares. Outlier detection using clustering methods: a data cleaning application. In *Proc. of the data mining for business workshop*, 2004.

[21] S. Pandit, D. H. Chau, S. Wang, and C. Faloutsos. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *WWW Conf.*, 2007.

[22] J. M. Robson. Algorithms for maximum independent sets. *Journal of Algorithms*, 7(3), 1986.

[23] T. Wu. An accurate computation of the hypergeometric distribution function. *ACM Trans. Math. Softw.*, 19(1), 1993.