5-2014

Detecting Anomaly Collections using Extreme Feature Ranks

Hanbo DAI
Hubei University

Feida ZHU
Singapore Management University, fdzhu@smu.edu.sg

Ee Peng LIM
Singapore Management University, eplim@smu.edu.sg

Hwee Hwa PANG
Singapore Management University, hhpang@smu.edu.sg

Citation

DAI, Hanbo; ZHU, Feida; LIM, Ee Peng; and PANG, Hwee Hwa. Detecting Anomaly Collections using Extreme Feature Ranks. (2014). *Data Mining and Knowledge Discovery*. 29, (3), 689-731.
Available at: https://ink.library.smu.edu.sg/sis_research/2534

Detecting anomaly collections using extreme feature ranks

Hanbo Dai · Feida Zhu · Ee-Peng Lim · HweeHwa Pang

Abstract Detecting anomaly collections is an important task with many applications, including spam and fraud detection. In an anomaly collection, entities often operate in collusion and hold different agendas to normal entities. As a result, they usually man-ifest collective extreme traits, i.e., members of an anomaly collection are consistently clustered toward the top or bottom ranks on certain features. We therefore propose to detect these anomaly collections by extreme feature ranks. We introduce a novel anomaly definition called *Extreme Rank Anomalous Collection* or ERAC. We propose a new measure of anomalousness capturing collective extreme traits based on a statisti-cal model. As there can be a large number of ERACs of various sizes, for simplicity, we first investigate the *ERAC detection problem* of finding top-*K* ERACs of a predefined size limit. We then tackle the follow-up *ERAC expansion problem* of uncovering the supersets of the detected ERACs that are more anomalous without any size constraint. Algorithms are proposed for both ERAC detection and expansion problems, followed by studies of their performance in four datasets. Specifically, in synthetic datasets, both ERAC detection and expansion algorithms demonstrate high precisions and recalls.

H. Dai (✉)
The School of Computer Science and Information Engineering, Hubei University, Wuhan, China
e-mail: daihanbo@hubu.edu.cn

F. Zhu · E-P. Lim · H. Pang
The School of Information Systems, Singapore Management University, Singapore, Singapore e-mail: fdzhu@smu.edu.sg

E-P. Lim
e-mail: eplim@smu.edu.sg

H. Pang
e-mail: hhpang@smu.edu.sg

In a web spam dataset, both ERAC detection and expansion algorithms discover web spammers with higher precisions than existing approaches. In an IMDB dataset, both ERAC detection and expansion algorithms identify unusual actor collections that are not easily identified by clustering-based methods. In a Chinese online forum dataset, our ERAC detection algorithm identifies suspicious "water army" spammer collections agreed by human evaluators. ERAC expansion algorithm successfully reveals two larger spammer collections with different spamming behaviors.

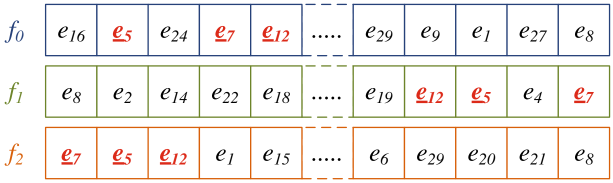
Keywords Anomaly collection · Extreme feature rank · Anomaly cluster · Outlier group · Spam detection · Spam cluster

1 Introduction

According to Barnett and Lewis (1994), an anomaly or outlier is a data instance or subset of data instances which appears to be inconsistent with the remainder of that set of data. In general, an anomaly can be classified as point anomaly or anomaly collection. A point anomaly usually lies in a sparse region or is far away from normal ones, whereas an anomaly collection are formed by similar entities, which as a whole is inconsistent with the rest. In practice, this inconsistency often implies different agendas from normal entities such as fraudulent activities or spamming campaigns.

In this paper, we detect anomaly collections by their extreme feature ranks, based on the observation that members in an anomaly collection often collaborate with each other and they manifest extreme traits. A good example is web spammer collections. As reported by Fetterly et al. (2004), Castillo et al. (2007) and Gyöngyi et al. (2004), web spammers adopt spamming strategies to boost the ratings of their pages. For example, they stuff the pages with popular keywords and anchor texts that are unrelated to one another. They 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 traits and form an identifiable anomalous collection, in contrast to normal web hosts.

To illustrate, Fig. 1 shows 30 web hosts $\{e_0, \ldots, e_{29}\}$ with three host features $\{f_0, f_1, f_2\}$, reflecting 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 anomaly collection because all of its entities appear at the top positions on features



f_0	e_{16}	e_5	e_{24}	e_7	e_{12}	.....	e_{29}	e_9	e_1	e_{27}	e_8
f_1	e_8	e_2	e_{14}	e_{22}	e_{18}	.....	e_{19}	e_{12}	e_5	e_4	e_7
f_2	e_7	e_5	e_{12}	e_1	e_{15}	.....	e_6	e_{29}	e_{20}	e_{21}	e_8

Fig. 1 An example of ERAC. 30 entities $\{e_0, \ldots, e_{29}\}$ 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

f_0 and f_2, and at the bottom positions on feature f_1. The fact that e_5, e_7 and e_{12} collectively display extreme traits across the three features is a strong evidence that they are likely to be web spammers.

As another example, groups of fraudulent users in online marketplaces are anomaly collections with extreme traits. A fraudulent user would create sufficient low-price transactions with other accomplice accounts in a short time in order to gain credibility, before performing fraud transactions involving large sums of money according to Chua and Wareham (2004) and Pandit et al. (2007). Consequently, they are likely to rank at extreme positions with respect to features such as average number of transactions and transaction rate.

To better study this kind of anomaly collections, we propose a novel definition, *Extreme Rank Anomalous Collection* or *ERAC*. An ERAC is an entity subset clustered toward the top or bottom ranks, when entities are ranked on certain features. ERACs cannot be easily detected by existing anomaly detection approaches, 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 ERAC, because not every entity in an ERAC may appear at extreme positions. For example, in Fig. 1, e_{12} is not very extreme by itself although it is part of an ERAC $\{e_5, e_7, e_{12}\}$. In contrast, e_8 is very anomalous as a single entity, since it appears at extreme positions on all three features, but it does not form an extreme cluster with any other entities. ERACs therefore cannot be discovered by simply grouping single point anomalies found by existing approaches.

To detect ERACs, Dai et al. (2012) propose to model an ERAC by the hypergeometric distribution. The anomalousness of an ERAC is quantified by a statistical p value. Due to the large number of ERACs of various sizes, Dai et al. (2012) tackle the *ERAC detection problem* of discovering top-K ERACs with a predefined size limit, which is set to small values for efficiency reasons. Nevertheless, after being offered with the top ERACs of a predefined size, users may want to further explore the most anomalous supersets of some detected top-K ERACs. For example, in the webspam case, users may find the detected ERAC $\{e_5, e_7, e_{12}\}$ of interest as they have the common spamming strategy of using lots of popular keywords, with very little variance on the word count, and high fraction of anchor text. It is natural to ask, can we detect the superset of this ERAC that are even more anomalous with similar sets of spamming strategies?

Therefore, in this paper, we not only explore the ERAC detection problem, but also propose the *ERAC expansion problem* to uncover the supersets of the detected ERACs that are more anomalous than the original top-K ones. Unlike the ERAC detection problem, ERAC expansion is done without predefined size constraints.

We summarize our contributions as follows:

- We are the first to detect anomaly collections by their extreme feature ranks. We measure the anomalousness of a collection by how extremely ranked it is with respect to any feature set.
- We develop both exact and heuristic algorithms to find the top-K anomalous collections of a predefined size limit on different pruning strategies under the feature

independence assumption. We then provide algorithms for coping with the more general situations with dependent feature sets.

- We propose the follow-up problem of ERAC expansion to uncover the anomalous supersets of the detected ERACs. We also design efficient greedy algorithms to solve the expanding problem without having to specify the size limit.
- We propose an exploratory scheme for searching ERACs, making use of the algorithms developed in both ERAC detection and expansion problems.
- We apply our ERAC detection approach on synthetic datasets with injected ERACs and on three real datasets including a web host graph, a movie dataset and a Chinese online forum dataset. In synthetic datasets, our proposed heuristic algorithm scales well with large dataset and at the same time maintains high precisions and recalls. For the web spam dataset with labeled true spammers, our approach discovers spammer collections that are more anomalous while achieving higher precisions, compared to existing approaches in both spam detection and anomaly detection. In the movie dataset, we detect unusual actor collections that are more anomalous than clustering-based method. User evaluation shows our approach successfully finds opinion spammer collections in the Chinese forum dataset.
- We also apply our ERAC expansion algorithm in all datasets. The results demonstrate that in the synthetic datasets, the injected ERACs are retrieved with high success rate; in web host dataset, the algorithm achieves higher precision than existing methods; in the movie dataset, the expansion reveals larger anomalous actor collections that cannot be discovered by the clustering-based approach; in the Chinese online forum dataset, the expansion uncovers larger spammer collections of coherent unusual behaviors.

The rest of the paper is organized as follows. Section 2 discusses related work. After introducing the problem formulation in Sects. 3 and 4 presents our ERAC detection algorithms for both independent and dependent feature sets. Section 5 describes our ERAC expansion problem and its algorithms. Section 6 discusses an exploratory scheme for searching ERACs. Section 7 reports on experiments on both ERAC detection and expansion. Finally, Sect. 8 concludes the paper and discusses limitations and feature work.

2 Related work

According to a survey by Chandola et al. (2009), most of the studies on anomaly detection focus on point anomalies. A handful of approaches are proposed such as a classification-based one by Castillo et al. (2007), a distance-based one by Knorr and Ng (1998), a density-based one by Breunig et al. (2000) and clustering based ones by Ester et al. (1996) and Guha et al. (1999). Since these approaches assume that anomalies appear in sparse regions or are far away from the normal entities, anomalous collections that are dense or are close to the normal entities may escape detection.

Duan et al. (2009), He et al. (2003) and Loureiro et al. (2004) use clustering based approach for anomalous collection detection, assuming that normal entities belong to large and dense clusters, whereas outliers form small or sparse clusters. According to Duan et al. (2009) and He et al. (2003), anomalous clusters are the smaller ones that together constitute less than 10 % of the population. Loureiro et al. (2004) claim that the threshold is half of the average cluster size. However, the assumption that small or sparse clusters are anomalous may not hold in many real datasets. Furthermore, these algorithms do not consider collective extreme traits.

Liu et al. (2010) detect anomaly collections by a data structure called isolation forest. The work assumes that after data points are projected to some hyperplane, the anomalous points follow one distribution while the normal points follow a different distribution. The two distributions are further assumed to be separable by minimizing their dispersions. However, these assumptions do not always hold, especially for datasets with multiple similar extreme patterns.

Arias-Castro et al. (2011) use a statistical model to detect in a graph an anomaly collection in which the node features have a different distribution than the rest of the nodes. This work takes the assumption that only one anomalous cluster exists in the whole graph, and only one feature can be attached to each node.

Detecting anomalous collections is also related to the task of subgroup discovery proposed by Klösgen (1996) and Wrobel (1997). A survey done by Herrera et al. (2011) summarizes the task as to discover the subgroups of the entity population that are statistically "most interesting" with respect to the class labels of entities. The subgroups are induced by rules, which are a conjunctive or disjunctive of attribute-value pairs. However, the anomaly collection of interest in this paper is not representable by the rules. This is because, (i) not all members of an anomaly collection satisfy rules; (ii) entities satisfying rules may not be members of an anomaly collection. Moreover, the anomalousness we use to measure an anomaly collection does not involve the class labels, whereas the interestingness measure used for subgraph discovery usually does.

3 Extreme rank anomalous collection

Dai et al. (2012) have shown that the anomalousness of a collection is better measured directly at collection level, instead of measuring individually on entity level followed by aggregating over the whole collection. Here we adopt that definition. Table 1 summarizes notations introduced below.

Let E denote the universal entity set, and F a set of features. $rank_f(e)$ denotes the rank of entity e in E w.r.t. feature f. In Fig. 1, $rank(e_8) = 1$ and $rank(e_7) = 30$ on feature f_1. An **extremity index** r refers to an extreme region, and $S_f(r)$ denotes the set of entities in S which appear in top r rank positions w.r.t. feature f,

Table 1 Notations

Notation	Meaning	Notation	Meaning
E	The universal entity set	F	A feature set $\{f\}$
S	An entity collection	r	An extremity index
$Rank_f(e)$	The ranking of e w.r.t. f	$S_f(r)$	Entities in S within r on f
$p_f(S, r)$	p value of S w.r.t. r on f	$\widehat{p}_f(S)$	S' representative p value on f
$r_f(S)$	Representative r of S on f		

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

In Fig. 1, suppose $S = \{e_5, e_7\}$ and $r = 2$, we have $S_{f_0}(r) = \{e_5\}$. Similarly, suppose $S = E$ and $r = 3$, $E_{f_1}(r) = \{e_8, e_2, e_{14}\}$ the top 3 entities on f_1.

The extremity of any collection $S \subset E$ can be quantified by $|S_f(r)|$, which is a random variable following the hypergeometric distribution. Thus the probability of observing $|S_f(r)|$ common entities shared by S and $E_f(r)$ is:

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

The p value of S w.r.t. extremity index r and feature f, denoted as $p_f(S, r)$, is the probability of observing at least $|S_f(r)|$ common entities between a random collection of size $|S|$ and $E_f(r)$.

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

Among all choices of r, we call the one which gives the smallest p value **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 corresponding representative p value of S w.r.t. f is denoted as $\widehat{p}_f(S)$, i.e., $\widehat{p}_f(S) = p_f(S, r_f(S))$. The smaller the representative p value, the more anomalous (i.e., extremely ranked) the collection is. In Fig. 1, we compare $S = \{e_5, e_7\}$ with $S' = \{e_7, e_{12}\}$ 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 smaller than $\widehat{p}_{f_0}(\{e_7, e_{12}\})$, $\{e_5, e_7\}$ 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.

Given a universal entity set E and an entity set S s.t. $S \subset E$, $1 < |S| < |E|/2$, a set of independent features F and a threshold α, we say S is an **Extreme Rank Anomalous Collection (ERAC)** w.r.t. F if (i) $\exists F^S \subseteq F$ such that $|F^S| > 1$ and $\forall f \in F^S$, $\widehat{p}_f(S) \le \alpha$; (ii) $1 < |S| < |E|/2$.

The definition is based on a global null hypothesis of multiple hypothesis tests, where each test corresponds to one feature. S is anomalous w.r.t. F, if the derived p values are smaller than a predefined significance level[1] α in at least two tests (i.e., $|F^S| > 1$).

Note that when we reject the null hypothesis and say that S is anomalous, S may not be significant for every feature. We call F^S the significant features of S. We impose

[1] In order to control the type 1 error (false positive), the significance level for each individual test should be adjusted. We adopt the Bonferroni Correction by Dunnett (1955) to adjust the significance level to $\alpha/|F|$.

the condition $1 < |S| < |E|/2$, as an anomalous collection should contain more than one entity and yet remain the minority of the population.

The definition also requires a set of independent features F. We define the **dependency** of any two features $f, f' \in F$ based on the statistic of Kendall Tau rank correlation coefficient Kendall (1948).

$$dep(f, f') = \frac{|3(n_c - n_d)|}{\sqrt{|E| \cdot (|E| - 1) \cdot (2|E| + 5)/2}}$$

where n_c is the number of concordant pairs of entities and n_d is the number of discordant pairs of entities in the entity lists, which are generated by ranking all entities on feature f and f' respectively. As the dependency statistic is expected to follow the standard normal distribution, we compute the z-score value λ corresponding to any given significance level of a two tailed test. For example, if the significance level is 0.05, the z-score is 1.96. Therefore, given f and f', if the $dep(f, f')$ score is not greater than λ, then f and f' are independent of each other. Note that cases with dependent features will be dealt in the following section.

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 a set of features F as the product of the representative p values for all the significant features in F. As the probability value tends to be very small, we take the log form:

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

This definition reflects that the more features S is extremely ranked against, the more anomalous it is.

With the anomaly score defined, we formulate our **ERAC detection problem** as follows: Given an entity universe set E, an independent feature set F, a target collection size N ($N < |E|/2$) and K, find the top-K most anomalous ERACs of size at most N.

As there are potentially large number of ERACs in a large population E, we focus on the top-K ERACs. We require a predefined size limit N of small values, as the anomalous collections are by definition the minority in the population anyway.

4 ERAC detection algorithms

In this section, to tackle the ERAC detection problem, we develop an exact algorithm in Sect. 4.1 and two heuristic algorithms in Sects. 4.2 and 4.3. The more general case of dependent features is discussed in Sect. 4.4.

Before we dive into details of these algorithms, we first analyze high level strategies to guide our algorithm design. A naive way to find the top-K ERACs of size at most N, is to enumerate all collections of size up to N, sort them by anomaly score in decreasing order, and return those with the top-K ranks. This approach is obviously infeasible due to its combinatorial nature. Therefore, a better strategy is to successively generate larger ERAC candidates from smaller ones, maintaining a current top-K list and pruning unpromising candidates. Since Agrawal and Srikant (1994) have proposed a proven way of generating candidates for frequent item-sets level by level from single entities to collections of target size N, we adopt his approach. However, for Apriori it is safe to prune any candidate from future consideration whenever it is found to fall below the threshold set by the last collection in the current top-K list, whereas our anomaly score of ERAC does not enjoy this downward closure property to support the standard pruning strategy in top-K computation. This is because for any collection S, even when 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 property on the anomaly score poses a difficulty to our bottom-up search approach.

4.1 An exact algorithm ERAC_E

Based on the analysis above, it seems that we would actually have to keep all collections of size n to generate candidates of size $n+1$ to guarantee the completeness of the mining result. Fortunately, despite the absence of downward closure property on the anomaly score which precludes setting 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. Intuitively, the most anomalous super-set S' of S, carrying this upper-bound can be formed by adding to S exactly $|S'| - |S|$ entities which are ranked the most extreme positions.

Formally, given a collection S, size n, ($|S| < n \leq |E|/2$), and a feature f, the most anomalous super-set of S is denoted as $\hat{S}_n$, and defined as $\hat{S}_n(f) = S \cup S^*$, where $|S^*| = n - |S|$ and $\forall e' \in S^*, \forall e \in E - \hat{S}_n, rank_f(e') \leq rank_f(e)$. Following the example in Fig. 1, suppose $S = \{e_5, e_{12}\}$, 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 in 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:

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

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 first introduce a property of p value.

Property 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 $p_f(S, r) < p_f(S', r')$.

Proof By definition, we have $p_f(S, r) = \sum_{i=|S_f(r)|}^{\min(|E_f(r)|,|S|)} \Pr(i, |E|, |E_f(r)|, |S|)$ and $p_f(S', r') = \sum_{i=|S'_f(r')|}^{\min(|E_f(r')|,|S'|)} \Pr(i, |E|, |E_f(r')|, |S'|)$.

Since $|S| = |S'|$, $|E_f(r)| = |E_f(r')|$ and $|S_f(r)| > |S'_f(r')|$, we have $p_f(S', r') = p_f(S, r) + \sum_{i=|S'_f(r')|}^{|S_f(r)|-1} \Pr(i, |E|, |E_f(r)|, |S|)$.

As $\Pr(i, |E|, |E_f(r)|, |S|) > 0$, we have $p_f(S, r) < p_f(S', r')$. □

Property 1 can be 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 arrive at the following Theorem 1 stating $\widehat{\Omega}(S, F, n)$ is indeed the upper-bound on the anomaly score of all super-sets of S of size n.

Theorem 1 *Given S, $0 < |S| < |E|/2$, $\forall S'$ such that $S \subset S'$ and $|S'| < |E|/2$, 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 $|\widehat{S}| = |S'|$, $|\widehat{S} \cap E_f(r_f(S'))| \geq |S' \cap E_f(r_f(S'))|$. According to Property 1, we have $p_f(S', r_f(S')) \geq p_f(\widehat{S}, r_f(S'))$. By definition, $p_f(\widehat{S}, r_f(\widehat{S})) \leq p_f(\widehat{S}, r)$. So we have $p_f(\widehat{S}, r_f(\widehat{S})) \leq p_f(\widehat{S}, r_f(S'))$.

Thus, $p_f(S', r_f(S')) \geq p_f(\hat{S}_{|S'|}(f), r_f(\hat{S}_{|S'|}(f)))$, and $\Omega(S', F) \leq \widehat{\Omega}(S, F, |S'|)$. □

Based on the upper-bound with size-constraint, we propose an exact algorithm that incorporates the following pruning strategy.

Pruning technique 1 *In 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.*

Although the absence of downward closure property creates issues, with our pruning strategy, the upper-bound is computed with a size-constraint which increases in tandem with the size of candidates being generated. It has several advantages: (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. 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 2 to illustrate the pruning techniques applied on the example in Fig. 2.

Step 6 of Algorithm 1 applies Theorem 1 in excluding from $\mathbb{S}$ those collections with upper-bounds smaller than the threshold. Moreover, through $join()$, 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 compute $S_1 \bigotimes S_2$ for each (S_1, S_2) 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 by Agrawal and Srikant (1994)).

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 2 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 of Algorithm 1, 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

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 $K$. $\Omega_t$ 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 $\mathbb{S}$ {$\Omega_t$ 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 $\mathbb{S}^*$
```

Table 2 Algorithm 1 running on the example shown in Fig. 2

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

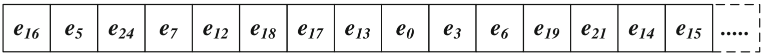


Fig. 2 Top 15 entities ranked according to f_0 in Fig. 1

comparing 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 $\{e_{15}\}$ in 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 leftover singular entities that may be selected to generate collections of size 3. However, even $\widehat{\Omega}(\{e_{18}\}, \{f_0\}, 3) = -\log p(\{e_{16}, e_5, e_{18}\}) = 5.31 < \Omega_t$, 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 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.

In this example, a standard Apriori algorithm without our 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. In comparison, we only visit $5+10+3=18$ nodes in total, saving the visit to 96 % of the nodes even in this small example.

Efficient anomaly score computation. To calculate $\Omega(S, F)$, for a given S and a feature f, we compute $\widehat{p}_f(S)$, which is decided by the representative extremity index of S w.r.t f. It seem that we need to scan through all $|E|/2$ possible indices so as to find this representative extremity index. Fortunately, the following property of the p value allows us to avoid checking all the extremity indices.

Property 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 $p_f(S,r)>p_f(S',r')$.

Proof Suppose $|E_f(r)|=|E_f(r')|+1=r_0+1$ and $|S_f(r)|=|S'_f(r')|=i_0$,

$$
\begin{aligned}
(1-p_f(S,r))\cdot\binom{|E|}{|S|} &= \sum_{i=0}^{i_0-1}\binom{r_0+1}{i}\cdot\binom{|E|-r_0-1}{|S|-i} \text{ (by definition)}\\
&= \sum_{i=0}^{i_0-1}\left(\binom{r_0}{i}+\binom{r_0}{i-1}\right)\cdot\binom{|E|-r_0-1}{|S|-i} \text{ (Pascal's triangle)}\\
&= \sum_{i=0}^{i_0-1}\binom{r_0}{i}\cdot\binom{|E|-r_0-1}{|S|-i}+\sum_{i=0}^{i_0-1-1}\binom{r_0}{i}\cdot\binom{|E|-r_0-1}{|S|-i-1}
\end{aligned}
$$

Similarly, $(1-p_f(S',r'))\cdot\binom{|E|}{|S|}=(1-p_f(S,r))\cdot\binom{|E|}{|S|}+\binom{r_0}{i_0-1}\cdot\binom{|E|-r_0-1}{|S|-(i_0-1)-1}$. Thus, we have

$$
\begin{aligned}
p_f(S,r) &= p_f(S',r')+\frac{\binom{r_0}{i_0-1}\cdot\binom{|E|-r_0-1}{|S|-(i_0-1)-1}}{\binom{|E|}{|S|}}\\
&= p_f(S',r')+\frac{\binom{|E_f(r)|-1}{|S_f(r)|-1}\cdot\binom{|E|-|E_f(r)|}{|S|-|S_f(r)|}}{\binom{|E|}{|S|}}
\end{aligned}
$$

In general, for any given $E_f(r)$ and $E_f(r')$ such that $|E_f(r)|>|E_f(r')|$,

$$
p_f(S,r)=p_f(S',r')+\sum_{j=0}^{|E_f(r)|-|E_f(r')|-1}\frac{\binom{|E_f(r)|-1-j}{|S_f(r)|-1}\cdot\binom{|E|-|E_f(r)|+j}{|S|-|S_f(r)|}}{\binom{|E|}{|S|}}
$$

As the second part $\sum_{j=0}^{|E_f(r)|-|E_f(r')|-1}\frac{\binom{|E_f(r)|-1-j}{|S_f(r)|-1}\cdot\binom{|E|-|E_f(r)|+j}{|S|-|S_f(r)|}}{\binom{|E|}{|S|}}>0$, we hence have $p_f(S,r)>p_f(S',r')$. □

Property 2 suggests that with all the other parameters kept constant, the smaller the extremity index, the smaller the p value is. It also suggests us to check only those extremity indices corresponding to the ranking of an entity in S. For example, in Fig. 1, for $S = \{e_5, e_7\}$ and f_0, we just check $r = 2$ and $r' = 4$; 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 2, we have $p_{f_0}(S, r) < p_{f_0}(S, r'')$. Thus, we do not need to consider r''. Similarly, all other extremity indices that do not correspond to the rank of any entity in S can be proven to 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 $O(|E|)$.

Time Complexity. As proposed by Wu (1993), the p value $p_f(S, r)$ can be calculated in $O(\min(|E_f(r)|, |S|))$ steps by recursion and factorial acceleration. Thus, the total time complexity of computing $\Omega(S, F)$ is $O(|S|^2 \cdot |F|)$, assuming $\min(|E_f(r)|, |S|) = |S|$. Similarly, by definition the time complexity of $\widehat{\Omega}()$ is on the same order as $\Omega(S, F)$. Step 6 takes $O(|\mathbb{S}| \cdot n^2 \cdot |F|)$. Step 8 can be implemented by a hash tree and thus is of $O(|\mathbb{S}(i)|^2)$. The size of $\mathbb{S}(i)$ is data dependent, which in the worse case is $|E|^i$. Let $|\mathbb{S}|$ denote the average size of $\mathbb{S}(i)$ for all i and all n. The running time of the "for" loop is $O(|\mathbb{S}|^2 + |\mathbb{S}| \cdot n^3 \cdot |F|)$. Therefore, the total running time of Algorithm 1 is $O(|\mathbb{S}|^2 + |\mathbb{S}| \cdot N^4 \cdot |F|)$.

4.2 A naive heuristic algorithm: ERAC_N

The number of seeds selected in Step 6 of the ERAC_E algorithm may be large and it may grow exponentially as the collection size increases. To further prune the potential candidates, we take the naive heuristic that only the top-m most anomalous potential candidates are selected as seeds to generate the collections of larger sizes.

With this heuristic, we add one more step: "$\mathbb{S}$ = top $m \cdot |E|$ most anomalous collections in $\mathbb{S}$" after Step 6 in Algorithm 1 with $0 < m < 1$. We thus obtain ERAC_N, a naive heuristic algorithm to detect top-K ERACs with independent feature set with m as additional parameter.

Since the naive heuristic fixes the number of seeds for collections of all sizes, it is expected to be much faster than the exact algorithm. Suppose on average we have $m \cdot |E|$ number of collections producing at most $(m \cdot |E|)^2$ potential candidates in the $join()$ step, the running time of the "for" loop is $O(m^2 \cdot |E|^2 + |E| \cdot n^3 \cdot |F|)$. Therefore, the time complexity of the heuristic algorithm is $O(m^2 \cdot |E|^2 + |E| \cdot N^4 \cdot |F|)$. The choice of m, which directly affects running time and the anomalousness of the top-K results, will be studied in Sect. 7.

4.3 A more sophisticated heuristic algorithm: ERAC_H

Since the bound given in Sect. 4.1 could be rather loose and gives little pruning power, and the assumption taken in the naive heuristic may be too strong and miss many anomalous ERACs, we now show a more sophisticated pruning technique that exploits our Apriori-style candidate generation.

Suppose we are at the stage of generating candidates of size n, and we know these candidates have anomaly scores that are no smaller than x. Now if we are examining two size-$(n-1)$ collections S_1 and S_2, and conclude from our computation that the upper-bound on the anomaly score of the resultant collection from combining S_1 and S_2 is still less than x, then it is unnecessary to combine them.

For simplicity, we denote $|S_f(r)|$ as i. For any given p value and size n, we represent 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 denote 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 for every feature, we derive for each feature $f \in F$, the corresponding representative p value $p_f(i^x, r^x, n)$ for S_x. We then 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. The process of decomposing x into a set of p values is discussed later in this section. For now, we assume $p_f(i^x, r^x, n)$ is known for any f.

Our task is to compute, for any given feature f, the bound on the representative p value of the resultant collection from combining S_1 and S_2, and compare against $p_f(i^x, r^x, n)$.

We first examine the following p value Property 3.

Property 3 Given any feature f, collections S and S' and extremity indexes r and r', if $|S_f(r)| = |S'_f(r')|$, $|E_f(r)| = |E_f(r')|$ and $|S| > |S'|$, then $p_f(S, r) > p_f(S', r')$.

According to Harkness (1965), the role of $|S_f(r)|$ and $|E_f(r)|$ are interchangeable in hypergeometric distribution, Property 3 can be proven similarly to Property 2. Property 3 suggests that with all the other parameters kept constant, the smaller the collection size, the smaller the p value is.

We now show the following Lemma stating that the combination of S_1 and S_2 is warranted, i.e., the resultant collection's p value is smaller than $p_f(i^x, r^x, n)$, only when each of them has a p value that is at most $p(i^x - 1, r^x, n-1)$.

Lemma 1 *For any feature f, given any representative p value $p(i, r, n)$ of some collection S of size n, the representative p values of S's subsets of size $n-1$ that generate S is at most $p(i-1, r, n-1)$.*

Proof Let S_1 and S_2 denote the generating subsets of S. We have $|S_1| = |S_2| = n-1$, $|S_1 \cap S_2| = |S_1| - 1$, $|S_1 - S_2| = 1$ and $S_1 \cup S_2 = S$. According to the definition of $p(i, r, n)$, S has i entities in $E(r)$. Hence S_1 and S_2 must have either i or $i-1$ entities in $E(r)$. Therefore, S_1 and S_2 must have p value of either $p(i-1, r, n-1)$ or $p(i, r, n-1)$. According to Property 2, we know that $p(i-1, r, n-1) \geq p(i, r, n-1)$. Hence, $p(i-1, r, n-1)$ is the largest possible p value among S_1 and S_2, since the representative p value is smaller than or equal to $p(i-1, r, n-1)$ by definition. We therefore proved that the upper-bound on the representative p values of S's subsets of size $n-1$ that generate S (i.e., S_1 and S_2) is $p(i-1, r, n-1)$. □

An illustration of Lemma 1 is as follows. Suppose $|E| = 30$, there exists a collection S of size $n = 4$ whose representative p value indicates that it has 3 members (i.e., $i = 3$) ranked among the top 8 (i.e., $r = 8$). Thus we have $p(3, 8, 4) = 0.048$. A collection (e.g., S_1) that can generate S must have at least 2 entities ranked among the top 8. This means $p(i-1, r, n-1) = p(2, 8, 3) = 0.166$ must be one of the p values of S_1. By definition, S_1's representative p value is therefore no greater than $p(2, 8, 3)$, which we conclude to be the upper-bound.

Investigating deeper into the relationship between $p(i, r, n)$ and $p(i-1, r, n-1)$, we arrive at the following:

Property 4 $p(i_1-1, r_1, n-1) - p(i_1, r_1, n) = \frac{n-i_1+1}{n} \cdot \Pr(i_1-1, |E|, r_1, n)$.

Proof

$$p(i, r, n) = p(i-1, r, n) - \Pr(i-1, |E|, r, n) \text{ (by definition of p value)}$$
$$= p(i-1, r, n-1) + \frac{\binom{n-1}{i-1-1} \cdot \binom{|E|-n}{r-(i-1)}}{\binom{|E|}{r}} - \Pr(i-1, |E|, r, n) \text{ (Property 3)}$$

Therefore, we have $p(i_1-1, r_1, n-1) - p(i_1, r_1, n)$

$$= -\frac{\binom{n-1}{i_1-1-1} \cdot \binom{|E|-n}{r_1-(i_1-1)}}{\binom{|E|}{r_1}} + \frac{\binom{n}{i_1-1} \cdot \binom{|E|-n}{r_1-(i_1-1)}}{\binom{|E|}{r_1}}$$
$$= -\frac{\binom{n-1}{i_1-1-1} \cdot \binom{|E|-n}{r_1-(i_1-1)}}{\binom{|E|}{r_1}} + \frac{(\binom{n-1}{i_1-1} + \binom{n-1}{i_1-1-1}) \cdot \binom{|E|-n}{r_1-(i_1-1)}}{\binom{|E|}{r_1}}$$
$$= \frac{\binom{n-1}{i_1-1} \cdot \binom{|E|-n}{r_1-(i_1-1)}}{\binom{|E|}{r_1}} = \frac{n-i_1+1}{n} \cdot \Pr(i_1-1, |E|, r_1, n).$$

□

It is important to note that the bound we derived is for the particular size-n collection S generated from combining its size-$n-1$ subsets S_1 and S_2. It does not generalize to the case where S_1 and S_2 are arbitrary collections of size $n-1$. For example, $p(1, 2, 3) = 0.284 > p(3, 10, 3) = 0.1053$, but the collections of size 4 generated from $p(1, 2, 3)$ is $p(2, 2, 4) = 0.0315$, which is smaller than $p(4, 10, 4) = 0.043$. This suggests although $p(i-1, r, n-1)$ is the upper-bound on collections of size $n-1$ for generating the particular collection of $p(i, r, n)$, it may not be the upper-bound for generating any collections that have a p value smaller than $p(i, r, n)$. We need to carefully examine all possible relationships between different r and i.

For any i_1, i_2, r_1 and r_2, we have 9 possible cases chosen from $\{i_1 < i_2, i_1 = i_2, i_1 > i_2\} \times \{r_1 < r_2, r_1 = r_2, r_1 > r_2\}$. Given two representative p values $p(i_1-1, r_1, n-1)$ and $p(i_2-1, r_2, n-1)$ of collections of size $n-1$, and the condition $p(i_1-1, r_1, n-1) > p(i_2-1, r_2, n-1)$, there are 4 cases that do not satisfy the condition. $\{i_1 = i_2\} \times \{r_1 = r_2, r_1 < r_2\}$ violates the condition according to Property 2. $\{i_1 > i_2\} \times \{r_1 = r_2, r_1 < r_2\}$ is not possible according to Property 1 and Property 2. The rest of the cases are addressed by the following two lemmas:

Lemma 2 *Given two representative p values $p(i_1-1, r_1, n-1)$ and $p(i_2-1, r_2, n-1)$ of collections of size $n-1$, and $p(i_1-1, r_1, n-1) > p(i_2-1, r_2, n-1)$, if*

$(i_1 = i_2$ and $r_1 > r_2)$ or $(i_1 < i_2$ and $r_1 = r_2)$ or $(i_1 < i_2$ and $r_1 > r_2)$, then $p(i_1, r_1, n) > p(i_2, r_2, n)$.

Proof For $i_1 = i_2$ and $r_1 > r_2$, we have $p(i_1, r_1, n) > p(i_2, r_2, n)$, according to Property 2.
For $i_1 < i_2$ and $r_1 = r_2$, we have $p(i_1, r_1, n) > p(i_2, r_2, n)$, according to Property 1.

For $i_1 < i_2$ and $r_1 > r_2$, we have $p(i_1, r_1, n) > p(i_2, r_2, n)$, according to Property 1 and Property 2. □

Lemma 3 *Given two representative p values $p(i_1-1, r_1, n-1)$ and $p(i_2-1, r_2, n-1)$ of collections of size $n-1$, and $p(i_1-1, r_1, n-1) > p(i_2-1, r_2, n-1)$, if $(i_1 < i_2$ and $r_1 < r_2)$ or $(i_1 > i_2$ and $r_1 > r_2)$ and $(n - i_1 + 1) \cdot \Pr(i_1 - 1, |E|, r_1, n) < (n - i_2 + 1) \cdot prob(i_2 - 1, |E|, r_2, n)$, then $p(i_1, r_1, n) > p(i_2, r_2, n)$.*

Proof According to Property 4, we have

$$p(i_1, r_1, n) - p(i_1 - 1, r_1, n - 1) = -\frac{n - i_1 + 1}{n} \cdot \Pr(i_1 - 1, |E|, r_1, n), \text{ and}$$
$$p(i_2, r_2, n) - p(i_2 - 1, r_2, n - 1) = -\frac{n - i_2 + 1}{n} \cdot \Pr(i_2 - 1, |E|, r_2, n).$$

Since $(n - i_1 + 1) \cdot \Pr(i_1 - 1, |E|, r_1, n) < (n - i_2 + 1) \cdot \Pr(i_2 - 1, |E|, r_2, n)$, we have $p(i_1, r_1, n) - p(i_1 - 1, r_1, n - 1) > p(i_2, r_2, n) - p(i_2 - 1, r_2, n - 1)$. With $p(i_1-1, r_1, n-1) > p(i_2-1, r_2, n-1)$, we therefore have $p(i_1, r_1, n) > p(i_2, r_2, n)$. □

In Lemma 3, if we impose on i_2 such that $\forall r', \nexists i' < i_2$ with $p(i', r', n) < p(i_2, r_2, n)$, then for the case of $i_1 < i_2$ and $r_1 < r_2$, we always have $p(i_1, r_1, n) > p(i_2, r_2, n)$ by definition.

Therefore, according to Lemma 2 and Lemma 3, for a given representative p value $x = p(i^x, r^x, n)$, such that $\nexists i' < i^x$ with $p(i', r', n) < p(i^x, r^x, n)$ for any r', we can compute the lower-bound on the p value of collections of size $n-1$ by $p(i^x-1, r^x, n-1)$. Except for the case of $i_1 > i^x, r_1 > r^x$ and $p(i^x-1, r^x, n-1) < p(i_1-1, r_1, n-1)$, we need to further check whether $p(i^x, r^x, n) < p(i_1 - 1, r_1, n - 1)$ still holds. If it no longer holds, we need to increase our upper-bound from $p(i^x - 1, r^x, n - 1)$ to $p(i_1 - 1, r_1, n - 1)$. This guarantees the true upper-bound.

However, this additional checking is costly. We therefore heuristically take $p(i^x - 1, r^x, n - 1)$ as the upper-bound on p value of collections of size $n - 1$. With $p(i^x - 1, r^x, n-1)$, we take $p(i^x-2, r^x, n-2)$ as the upper-bound on p value of collections of size $n - 2$, and so on.

Thus, 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

$$\Omega^*(x, n, n') = -\sum_{f \in F} \log p_f(i^x - (n - n'), r^x, n')$$

After computing these lower-cuts for collections of size n' along each feature, we sum negative log values of the bounds to get a lower-cut on the anomaly score of the

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 $K$. $\Omega_t$ 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:     $\Omega'_N = \Omega_N$
 5:     for $n = 1$ to $N - 1$ do
 6:         $\mathbb{S} = \{S \in \mathbb{S} \mid \Omega(S, F) > \max(\Omega^*(\Omega_N, N, n), \Omega^*(\Omega_t, n + 1, n))\}$
 7:         $\mathbb{S} = join(\mathbb{S}, \mathbb{S})$ {$\mathbb{S}^*$ and $\Omega_t$ are updated by elements in $\mathbb{S}$ whenever necessary}
 8:     $\Omega_N = \Omega_t$
 9: until $\Omega_N \geq \Omega'_N$
10: return $\mathbb{S}^*$
```

collections; any collections of size n' with anomaly score below this lower-cut can be pruned.

We arrive at the following pruning strategy.

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 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 $\Omega^*(x, n, n')$.*

Overall our heuristic algorithm works as follows. Given a target size N, we first estimate a threshold Ω_N for 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 start with. 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. As 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 in generating 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.

Running example. Setting $K = 1$ and $N = 3$ again, we show how Algorithm 2 executes on the example in Fig. 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.

Time complexity. We now analyze the time complexity of Algorithm 2. Let $|\overline{\mathbb{S}'}|$ denote the average size of $\mathbb{S}$ for all n. Since Step 6 is $O(|\overline{\mathbb{S}'}| \cdot n^2 \cdot |F|)$ and Step 7 is $O(|\overline{\mathbb{S}'}|^2)$, the time complexity of Algorithm 2 is $O(|\overline{\mathbb{S}'}|^2 + |\overline{\mathbb{S}'}| \cdot N^3 \cdot |F|)$, 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 counterpart $|\overline{\mathbb{S}}|$ in Algorithm 1.

Anomaly score decomposition. We now describe 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 (i) $x \leq -\sum_{f \in F} \log x_f$; (ii) $\nexists i' < i^x$ 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. We first compute $e^{-\frac{x}{|F|}}$, then find $p_f(i^x, r^x, n)$ such that (i) $p_f(i^x, r^x, n) \leq e^{-\frac{x}{|F|}}$; (ii) $\nexists i' < i^x$ with $p(i', r', n) < p(i^x, r^x, n)$; (iii) $\nexists r' < r^x$ with $p(i^x, r', n) < e^{-\frac{x}{|F|}}$. Condition (i) guarantees that $x \leq -\sum_{f \in F} \log x_f$. Condition (ii) is required in the heuristic. Condition (iii) is based on Property 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 leads to a smaller anomaly score.

4.4 Handling dependent features

So far, we have assumed that features in F are independent of each other. The more general case of dependant features needs to be studied. We say a feature set F is an **Independent Feature Set** or IFS, if $|F| > 1$ and $\forall f_i, f_j \in F$, f_i is independent of f_j. The dependency of any two features is computed as its definition in Sect. 1. As there are potentially multiple IFS alternatives, we focus on the 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 $\mathbb{F}$.

With $\mathbb{F}$, the overall anomaly score of S is aggregated by taking the largest anomaly score across all maximal IFSs in $\mathbb{F}$:

$$\Omega(S) = \max_{F' \in \mathbb{F}} \Omega(S, F')$$

We propose two algorithms shown in Algorithm 3 to find ERACs involving a dependent feature set F based on the exact and heuristic 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 IFSs 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. The algorithm returns both the ERACs and their associated maximal

Algorithm 3 ERACD_E and ERACD_H

```
Input: $E$, $F$, $K$ and $N$
Output: top-$K$ ERACs and maximal IFSs: $\mathbb{C} = \{\langle S, F' \rangle\}$
{$\mathbb{C}.F(S)$ is a function that returns the corresponding maximal IFS of any top-$K$ ERAC $S$ in $\mathbb{C}$}
 1: $\mathbb{C} = \emptyset$
 2: compute dependency relation of every two features in $F$ defined in Section 3 and keep
    the results in $R$.
 3: $\mathbb{F}$= generateMIFS($F$, $R$)
 4: for all $F' \in \mathbb{F}$ do
 5:     $\mathbb{S}^* =$ ERAC_E$(E, F', K, N)$ {for ERACD_H, $\mathbb{S}^* =$ ERAC_H$(E, F', K, N)$ }
 6:     for all $S \in \mathbb{S}^*$ do
 7:         if $S$ is already in top-$K$ ERACs then
 8:             if $\Omega(S, \mathbb{C}.F(S)) < \Omega(S, F')$ then
 9:                 delete $\langle S, \mathbb{C}.F(S) \rangle$ from $\mathbb{C}$
10:                 add $\langle S, F' \rangle$ to $\mathbb{C}$
11:         else
12:             add $\langle S, F' \rangle$ to $\mathbb{C}$
13: return $\mathbb{C}$
```

IFSs. We use a priority queue of maximum length of K as before, but with composite elements $\langle S, F' \rangle$, where S is an ERAC and F' is the maximal IFSs that make S most anomalous. We constrain the priority queue to have distinct collections only.

Note that in Step 3, we need to compute all maximal IFSs $\mathbb{F}$. If we represent each feature as a node and each dependency relation between two features as an edge between the corresponding two nodes, we transfer the problem of finding all maximal IFSs to that of computing all maximal independent set in a graph, which is a well-shown NP-hard problem according to Lawler et al. (1980) or the survey by Bomze et al. (1999). We adopt the algorithm by Eppstein et al. (2005) for an approximation, which proposes an efficient algorithm based on Avis and Fukuda (1993)'s reverse search paradigm with a novel data structure and a "parent" operation for quickly testing whether certain subsets of the vertices of the graph is maximal. When applied to our problem, the complexity of this algorithm is $O(|F|^{2-1/|F|})$ per generated maximal independent feature set, where $|F|$ is the total number of features.

We now analyze the complexities of ERACD_E and ERACD_H. The two algorithms differ only in Step 5, calling ERAC_E and ERAC_H respectively, whose complexities are analyzed in previous sections. Step 2 takes $O(|F|^2)$ time, and Step 3 takes sub-quadratic time $O(|F|^{2-1/|F|})$ by the algorithm of Eppstein et al. (2005).

Therefore, the complexity of algorithm ERACD_E is $O(|F|^2 + |F|^{2-1/|F|} \cdot |\mathbb{F}| + |\mathbb{F}| \cdot (|\mathbb{S}|^2 + |\mathbb{S}| \cdot N^4 \cdot |F| + K \cdot N^2 \cdot |F|))$, and the complexity of algorithm ERACD_H is $O(|F|^2 + |F|^{2-1/|F|} \cdot |\mathbb{F}| + |\mathbb{F}| \cdot (|\mathbb{S}|^2 + |\mathbb{S}| \cdot N^3 \cdot |F| + K \cdot N^2 \cdot |F|))$. As before, $|\mathbb{S}|$ denotes the average size of candidate collections processed in Algorithm ERAC_E and ERAC_H.

5 ERAC expansion

With the aforementioned algorithms, users are able to find top-K ERACs of size no greater than a predefined N. It is natural to wonder if the identified ERACs can be expanded to larger, more anomalous ERACs. For example, after seeing some identified top-K spammer collections, users may be interested in supersets of these spammer collections, which are even more anomalous.

One way to derive the larger ERACs is to rerun the ERAC detection algorithms with a larger N. However, users often have little idea about the exact size of the larger ERACs to set, hence a trial and error strategy is non-ideal. Moreover, it is possible that some interesting ERAC in the top-K list for N may disappear in the top-K list of a larger N with the current K setting. This makes it even harder to track down the superset of the ERAC of interest.

In this section, we propose the problem of expanding an ERAC to the most anomalous superset of any size. We then propose algorithms to solve the problem.

The **ERAC expansion problem** is defined as follows: Given an ERAC S, find $S' \subset E - S$ such that (i) $S \cup S'$ is an ERAC; (ii) $\Omega(S \cup S', F) > \Omega(S, F)$; (iii) $\forall S'' \subset E - S$, we have $\Omega(S \cup S', F) \geq \Omega(S \cup S'', F)$.

The definition states that expanding an ERAC S will produce the superset of S having the largest anomaly score among all of S's supersets.

5.1 ERAC expansion algorithms

With little modification, the algorithms in Sect. 4 for detecting top-K ERACs of size no greater than N can be applied to expand the ERACs. Specifically, given an ERAC S to be expanded, for the exact algorithm ERACD_E, we first set $N = |E|/2$ and $K = 1$. Then in the process of computing the upper-bound of the supersets of any candidate collection S', we compute the upper-bound of $S \cup S'$ instead. According to the analysis in Sect. 4.1, the complexity is $O(|\overline{\mathbb{S}}|^2 + |\overline{\mathbb{S}}| \cdot E^4 \cdot |F|)$.

For the sophisticated heuristic algorithm ERACD_H, we also set $N = |E|/2$ and $K = 1$. When computing the lower-cut using S_x, we use $S \cup S_x$ instead. According to the analysis in Sect. 4.3, the complexity is $O(|\overline{\mathbb{S}'}|^2 + |\overline{\mathbb{S}'}| \cdot E^3 \cdot |F|)$.

As the modified ERACD_E and ERACD_H are still costly, we propose below more efficient algorithms for ERAC expansion.

5.1.1 A greedy algorithm

We design a greedy algorithm to add one entity at a time until no further expansion can produce a more anomalous ERAC. In each step, we add one entity such that the resultant ERAC gives the highest anomaly score.

This greedy scheme is based on the observation that members of an ERAC behave like each other and most likely occupy the extreme positions of a similar set of features. Thus, given a subset of an ERAC, the rest of the members of this ERAC are expected to have similar extreme behavior as the subset.

A naive way of choosing an entity to join S is to try out every entity in E and select the one that gives the largest anomaly score after joining with S. This is costly, as each insertion of an entity to S entails scanning through all the entities in E. A more efficient approach only needs to try out the entities within the extremity index at each insertion step, which can be easily derived by the existing p values of S on each feature.

Specifically, for any feature f, we know the corresponding $p(i, r, n)$ for S on f. To insert into S an entity that gives a larger anomaly score, we want $p(i+1, r', n+1)$ to be smaller than $p(i, r, n)$. Since there can be many r' that satisfy the condition, we define the upper-bound on r', $r(S, f)$, to be the largest extremity index such that adding any entity within $r(S, f)$ produces a smaller p value than $p(i, r, n)$ on feature f. Note that these p values can be pre-computed and sorted for efficient retrieval. Thus finding the right $r(S, f)$ can be done in constant time.

Given S and F, we define the **candidate pool** $A(S, F)$ as the set of entities within $r(S, f)$ for all $f \in F$. That is, $A(S, F) = \bigcup_{f \in F} E_f(r(S, f))$, where $E_f(r)$ is the set of entities within $r(S, f)$ defined in Sect. 3.

Lemma 4 *$\forall e \in E - A(S, F)$, we have $\Omega(S \cup \{e\}, F) < \Omega(S, F)$.*

This lemma states that we only need to check the entities in the candidate pool instead of all entities in E to guarantee we find the entity e that gives the largest anomaly score after joining S.

The lemma can be easily proven. Since $r(S, f)$ is the upper-bound on the extremity index for feature f, each entity in $A(S, F)$ when joining S will give a smaller p value in at least one feature than that of S alone. Thus, every entity in $A(S, F)$ has a chance to produce a more anomalous superset after joining S. On the other hand, every entity in $E - A(S, F)$ has no chance to produce a more anomalous superset, as any resultant superset is less anomalous than S w.r.t every feature in F. Since the final anomaly score is the sum of the anomaly scores w.r.t. all significant features, each entity in $E - A(S, F)$ when joining with S will give a smaller anomaly score than that of S.

With this lemma, we propose the following greedy algorithm for expanding S.

The algorithm is given an initial collection S^0 to expand and outputs the resultant S, which is the most anomalous superset of S^0 of any size. In each loop of i in Algorithm 4, a new X (i.e., $A(S, F)$) is computed and entities in X are visited to find the one that boosts the anomaly score the most. If the resultant anomaly score still increases, the size of S is incremented by one.

We now analyze the complexity of Algorithm 4. In Step 7, suppose $|\bigcup_{f \in F} E_f (r(S, f))| = \overline{r} \cdot |F|$, where $\overline{r}$ is the average number of entities within the extremity index across all features. Given that computing $\Omega(S \cup \{e\}, F)$ is $O((|S| + 1)^2 \cdot |F|)$, Step 7 takes $O(\overline{r} \cdot |F| \cdot (|S| + 1)^2 \cdot |F|)$.

Therefore, the complexity of the whole algorithm is $O(|E| \cdot |F| + \overline{r} \cdot |F| \cdot |E|^3 \cdot |F| + |E|^3 \cdot |F|)$. Recall the complexity of the modified sophisticated heuristic algorithm for detecting ERACs is $O(|\overline{\mathbb{S}'}|^2 + |\overline{\mathbb{S}'}| \cdot E^3 \cdot |F|)$. Since $|\overline{\mathbb{S}'}|$ is $O(|E|^N)$, the expansion algorithm here is much faster.

5.1.2 A heuristic-based greedy algorithm

Note that the time consuming part of the expansion algorithm is in each loop of i computing the anomaly score for all entities in the updated candidate pool $A(S, F)$ in Step 7, which has complexity $O(\overline{r} \cdot |F| \cdot |E|^2 \cdot |F|)$. If we can expedite this step, we can achieve an even more efficient algorithm. This is possible if we avoid computing the value of $\Omega(S \cup \{e\}, F)$ for each entity in $A(S, F)$. Instead we choose from $A(S, F)$

Algorithm 4 ERAC Expansion ERAC_exp

```
Input: $E$, $F$, an ERAC $S^0$ to be expanded
Output: Expanded ERAC $S$ from $S^0$
$S = S^0$
for $i = |S^0| to |E|/2$ do
    $A = \emptyset$ {$A$ keeps the candidate pool $A(S, F)$}
    for all $f \in F$ do
        find $r(S, f)$ in the pre-computed and sorted p-value list
        $A = A \cup E_f(r(S, f))$
    Find entity $e$ in $A$ s.t. $\Omega(S \cup \{e\}, F)$ is the largest.
    if $\Omega(S \cup \{e\}, F) > \Omega(S, F)$ then
        $S = S \cup \{e\}$
    else
        break
return $S$
```

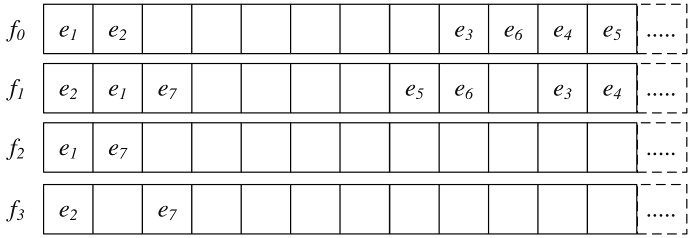
the entity that gives the largest anomaly score when joining S^0, which is pre-computed for each entity e in E.

We introduce the following **expansion heuristic**: entity e in $A(S, F)$ that gives the largest $\Omega(S^0 \cup \{e\}, F)$ value will also give the largest $\Omega(S \cup \{e\}, F)$.

Since our algorithm greedily adds entities to S^0 one after another, this heuristic ignores all entities added to S^0 in the expansion process and always select the entity in $A(S, F)$ that contributes the most to S^0 instead of S. Suppose S^0 is in fact a subset of a larger ERAC S, the heuristic has a better chance of retrieving S from S^0, if S^0 already has a similar extreme pattern as S, i.e., both of them occupy extreme positions on similar features.

However, S^0 may not always have a similar extreme pattern as S, i.e., S^0 may occupy extreme positions of a few more or less features than S does, which is more likely to happen when the size of S^0 is much smaller than S. When this happens, the expansion heuristic may fail as it favors only the entities that are similar to all members in S^0, which are not necessarily similar to all members in S.

For example, Fig. 3 shows the top-12 positions of four features with seven entities e_1 to e_7. The empty positions are occupied by other entities, which are not relevant and are not shown for simplicity. Suppose the target ERAC is $S = \{e_1, e_2, e_3, e_4, e_5, e_6\}$. If we expand $S^0 = \{e_1, e_2\}$ with the expansion heuristic, the first entity joining S^0 is e_7, which produces the highest $\Omega(S^0 \cup \{e\}, F)$ value compare to other entities. This will lead us to a different superset than S. However, if we start with $S^0 = \{e_1, e_2, e_3\}$, the



f_0	e_1	e_2							e_3	e_6	e_4	e_5	.....
f_1	e_2	e_1	e_7					e_5	e_6		e_3	e_4	.....
f_2	e_1	e_7											.....
f_3	e_2		e_7										.....

Fig. 3 An illustration of the expansion heuristic. Suppose $S = \{e_1, e_2, e_3, e_4, e_5, e_6\}$

Algorithm 5 Combined ERAC algorithm ERAC_comb

```
Input: $E$, $F$, $K$ and $N$
Output: $\mathbb{C}$
$\mathbb{C} = \emptyset$
$\mathbb{C}' =$ ERACD_H$(K, N)$ {run the ERAC detection algorithm}
for all $S \in \mathbb{C}'$ do
    $S$=ERAC_exp$(E, F, S)$ {run the ERAC expansion algorithm}
    add $S$ to $\mathbb{C}$
return $\mathbb{C}$
```

following entities to join S^0 are then from among $\{e_4, e_5, e_6, e_7\}$, as they all produce high $\Omega(S^0 \cup \{e\}, F)$ values. This way, the heuristic will successfully lead us to the anomalous superset S.

To incorporate this heuristic, we change Step 7 in Algorithm 4 to "find the entity e in X such that $\Omega(S^0 \cup \{e\}, F)$ is the largest". We also assume that the corresponding anomaly score $\Omega(S^0 \cup \{e\}, F)$ for each entity e in E is pre-computed. As a result, the new Step 7 has complexity $O(\bar{r} \cdot |F|)$.

Therefore, the complexity of the whole heuristic-based greedy expansion algorithm is $O(|E| \cdot |F| + |E| \cdot \bar{r} \cdot |F| + |E|^3 \cdot |F|)$, faster than without the heuristic. We apply the more efficient heuristic based greedy algorithm, denoted exp, in the experiments to demonstrate the effectiveness of expanding ERACs.

6 Discussion on exploratory search for ERACs

In this section, we discuss an exploratory search strategy for more anomalous ERACs, making use of the algorithms developed in both ERAC detection and expansion problems.

Ideally, users want to search for more anomalous ERACs regardless of collection size. Since the search space grows exponentially with the population size, for efficiency reasons, we propose to first investigate the ERAC detection problem of finding top-K ERACs of size less than a predefined N of small values. The detected ERACs are treated as "seeds" in the follow-up ERAC expansion problem to retrieve the larger and more anomalous ERACs. In this strategy, the ability of discovering more anomalous ERACs largely depends on whether the seeds contains any subset of the more anomalous ERACs.

A direct way of incorporating more informative seeds is to set a large K and N. However, we have shown in Sect. 7 that larger K and N will increase the execution time of the ERAC detection algorithms. Moreover, setting the right K and N is not trivial. Due to the lack of downward closure property, it is possible that the subsets of a less anomalous ERAC are even more anomalous than the subsets of a more anomalous ERAC. Thus, one cannot easily predict the proper K and N by any monotonic property.

In this section, we therefore propose an exploratory search strategy that gradually increases K and N based on a gain function. Since the gain function is derived based on the proposed ERAC detection and expansion algorithms, we first describe the combined ERAC algorithm comb in Algorithm 5. $\mathbb{C}$ denotes the resultant set of ERACs after expanding the top-K ERACs of size no greater than N.

Algorithm 6 Exploratory ERAC search

```
Input: $E$, $F$, initial $K^0$ and $N^0$ selected by user for $K$ and $N$
Output: $\mathbb{C}$, the resultant set of ERACs
Set $K = K^0$ and $N = N^0$
Run ERAC_comb$(E, F, K, N)$; set $t = T(K, N)$ and $g = G(K, N)$
while exploratory search continues do
    Run $\mathbb{C}_1$ =ERAC_comb$(E, F, K+1, N)$; set $t_1 = T(K+1, N)$ and $g_1 = G(K+1, N)$
    Run $\mathbb{C}_2$ =ERAC_comb$(E, F, K, N+1)$; set $t_2 = T(K, N+1)$ and $g_2 = G(K, N+1)$
    if $\frac{g_1-g}{g} - \frac{t_1-t}{t} > \frac{g_2-g}{g} - \frac{t_2-t}{t}$ then
        $\mathbb{C} = \mathbb{C}_1$
        $K = K+1$; $t = t_1$; $g = g_1$;
    else
        $\mathbb{C} = \mathbb{C}_2$
        $N = N+1$; $t = t_2$; $g = g_2$;
return $\mathbb{C}$
```

We denote $T(K, N)$ as the time required to run comb for a given K and N. The goal function, denoted as $G(K, N)$, can be defined according to users' search goal. For example, if users want to get the most anomalous ERACs, the goal function is $\max_{S \in \mathbb{C}}(\Omega(S, F))$. If users want as many anomalous ERACs as possible, the goal function is $\sum_{S \in \mathbb{C}}(\Omega(S, F))$.

Given (K, N) and (K', N') with $K' \geq K$ and $N' \geq N$, corresponding to two runs of the comb algorithm, the **gain function** $I(K', N', K, N)$ is defined as:

$$\frac{G(K', N') - G(K, N)}{G(K, N)} - \frac{T(K', N') - T(K, N)}{T(K, N)}$$

The first part of the gain function captures the change ratio in gain function and the other capture the change ratio in execution time. This gain function has larger value if the second run of comb with (K', N') have larger gain $G(K', N')$ and smaller execution time $T(K', N')$ than the first run with (K, N). We use the ratio to normalize the changes so that the change on gain and time are on the same scale.

Now, we describe our high-level search strategy in Algorithm 6. The increase of K or N is decided by the gain function.

7 Experiments

In this section, we examine the performance of our ERAC detection and ERAC expansion algorithms on four datasets, including the synthetic datasets, a web spam dataset, a movie dataset and a Chinese online forum dataset. For each dataset, we first report the results of both our ERAC detection and ERAC expansion algorithms. We are interested to know (i) How well can our algorithms retrieve the injected ERACs in synthetic data of various population sizes; (ii) Whether our algorithms can retrieve meaningful collections in the real-life datasets; (iii) Can our algorithms retrieve more anomalous collections compared to existing approaches; (iv) Can our algorithms achieve higher precision where ground truth is available.

Algorithm 7 *Generate synthetic data with injected ERACs*

```
Input: $E$, $F$, size of injected ERACs $n$, number of injected ERACs $y$
Output: the set of entity lists $\{EL_f\}$ and the set of injected ERACs $\mathfrak{S}$
 1: Generate for each feature $f$, a random entity list over the same population $E$ as $\{EL_f\}$.
 2: Let $\mathfrak{S} \leftarrow \emptyset$
 3: while $\{EL_f\}$ are independent of each other and $|\mathfrak{S}| < y$ do
 4:     Randomly select $F' \subseteq F$ to be the significant features for a new injected ERAC.
 5:     Estimate the upper bound $z$ on the extremity index of this ERAC, such that this
        ERAC is more anomalous than the top-1 ERAC found in the original randomized
        entity lists in step 1.
 6:     For each feature $f \in F'$, randomly generate extremity index $r_f$, such that $r_f < z$.
 7:     Randomly select a feature $\widehat{f} \in F'$, and randomly select a collection $S$ whose repre-
        sentative p-value is $p(n, r_{\widehat{f}}, n)$ w.r.t. $\widehat{f}$ and $S$ is disjoint with all previously injected
        ERACs in $\mathfrak{S}$.
 8:     Add $S$ to $\mathfrak{S}$. {$S$ is one of the injected ERACs}
 9:     for each $f \in F'$ and $f \neq \widehat{f}$ do
10:         Randomly select a collection $S'$ whose representative p-value is $p(n, r_f, n)$ w.r.t.
            $r_f$ and is disjoint with all previously injected ERACs in $\mathfrak{S}$.
11:         Switch the positions of the elements in $S$ to those of elements in $S'$ in $EL_f$.
12: return $\{EL_f\}$ and $\mathfrak{S}$
```

7.1 Results on synthetic data

7.1.1 Synthetic data generation

The input to the algorithm includes population size and number of features. In addition, we assume the size of any injected ERAC and the total number of injected ERACs are given. The output is the set of entity lists $\{EL_f\}$ and the set of injected ERACs. We begin with randomized entity lists and inject ERACs of various sizes until the entity lists are no longer independent of each other. The dependency between entity lists is computed according to Kendall (1948). Algorithm 7 shows the detail.

7.1.2 ERAC detection in synthetic data

We compare the exact and heuristic algorithms in terms of effectiveness and efficiency by varying the parameter settings and generating synthetic datasets with different ground truths.

We fix the number of features at 10, the size of any injected ERAC at 10, and the number of injected ERACs at 5. Different synthetic datasets are generated by varying the population size $|E|$ from 100 to 300. For each population size, we generate three datasets for measuring the average performance of our exact ERAC_E, naive heuristic ERAC_N and sophisticated heuristic ERAC_H algorithms.

For all our algorithms, we fix $N = 10$ and $K = 5$ according to the synthetic data generation parameters. As for ERAC_N, we vary m that controls the number of candidate collections, i.e., $m \in \{0.1, 0.5, 1.0\}$. A larger m requires more candidate collections to be processed.

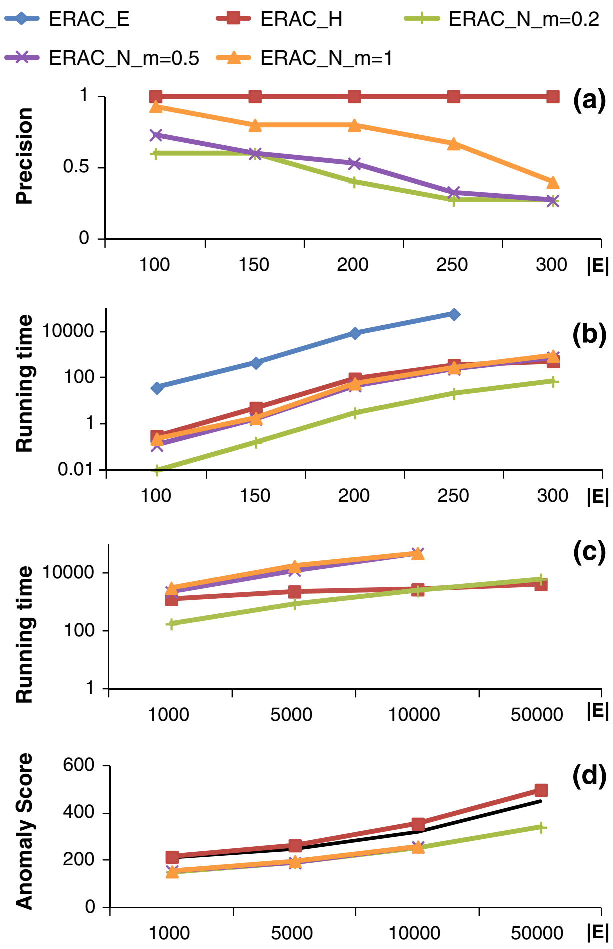


Fig. 4 Results on synthetic data

Figure 4a and b report the precision and running time of each algorithm respectively, averaged over 3 generated datasets for each population size. Note that the precision here equals recall as the number of injected ERACs equals to the number of anomalous collections returned. When $|E| = 300$, the exact algorithm takes more than 24 h, thus its results are excluded.

We see from Figure 4a that both ERAC_E and ERAC_H are able to retrieve all injected ERACs and achieve perfect precision. On the other hand, the precision of ERAC_N drops as m decreases and the population size $|E|$ increases. These precisions are in general lower than those of ERAC_E and ERAC_H. Figure 4b confirms that ERAC_H and ERAC_N require much lower execution time than the exact one as expected. ERAC_N needs more execution time when m decreases. The figures show that our sophisticated algorithm ERAC_H achieves comparable precisions with much less execution time than ERAC_E. It also achieves higher precisions although it incurs a longer running time than ERAC_N.

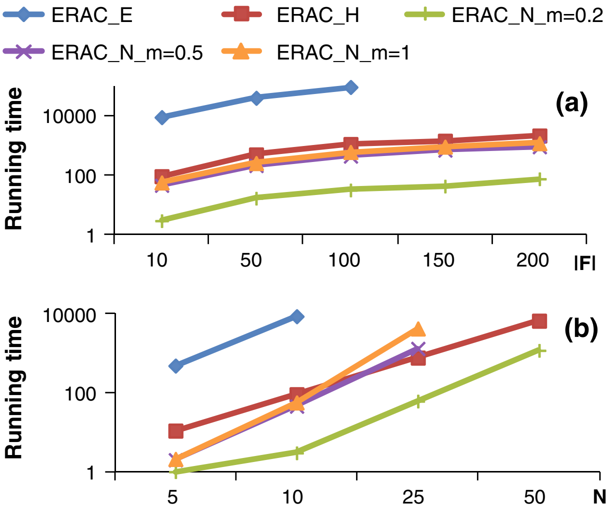


Fig. 5 The impact of $|F|$ and N on running time of proposed algorithms

To test the scalability of ERAC_N and ERAC_H, while keeping $N = 10$, $K = 5$ and $|F| = 10$, we gradually increase the population size up to 50,000, where ERAC_N with $m = 0.5$ and $m = 1.0$ cannot complete within 24 h. For each population size, we again generate three datasets. Besides running time, we also report the anomaly score of the K-th most anomalous ERAC identified by each algorithm and the anomaly score of the K-th most anomalous injected ERAC, denoted as injected. We plot the results in Fig. 4c and d.

We see that ERAC_H scales well with data size. In all the settings, ERAC_H is able to retrieve either all the injected ERACs or the ones that are even more anomalous than the injected ERACs. It is possible that as the population size goes larger, there exist ERACs more anomalous than the injected ones. This is because if any injected ERAC is not very anomalous, its subsets are more likely to form more anomalous ERACs with other entities. However, ERAC_N cannot even retrieve collections that are as anomalous as the injected ones. This shows that the pruning technique used in ERAC_H takes good advantage of the properties of the ERACs and render much better results than the naive heuristics.

Next, we investigate the impact of the size of the feature set F and the collection size N on the execution time of ERAC_N (with $m \in \{0.1, 0.5, 1.0\}$ as before) and ERAC_H. According to the result of above experiments, we fix the population size $|E|$ to 200, so that ERAC_E can finish in a reasonable period. We first set $N = 10$, $K = 5$ and vary $|F|$ from 10 to 200. For each choice of $|F|$, we generate three datasets as before. The running time is averaged across datasets and shown in Fig. 5a. We can see that the number of features has less impact on the running time of all algorithm than population size $|E|$ does as shown in Fig. 4. We then set $|F| = 10$, $K = 5$ and vary N from 5 to 50. For each choice of N, we also generate three datasets and plot the averaged running time in Fig. 5b. We can observe from this figure that N has much larger impact on the running time than $|F|$ does, which is in line with the time

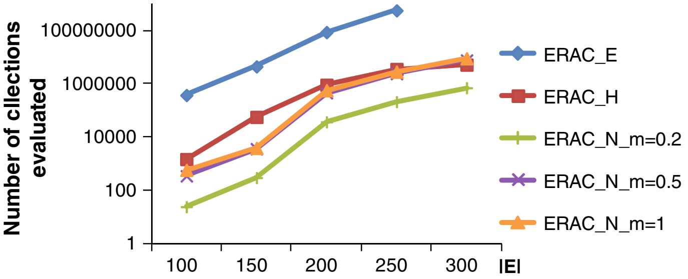


Fig. 6 The pruning power of proposed algorithms

complexity analysis. Note that in Fig. 5, if any run takes more than 24 h, its results are excluded.

We also study the pruning power of the proposed algorithms by measuring the number of collections evaluated. The fewer the number of collections being evaluated, the large pruning power an algorithm has. We fix $N = 10$, $K = 5$, $|F| = 10$ and vary the population size $|E|$ from 100 to 300 as before for the purpose of cross checking. We generate three datasets for each setting of $|E|$, the results are plotted in Fig. 6, which shows our heuristic algorithms ERAC_H and ERAC_N has larger pruning power than the exact one as expected.

7.1.3 ERAC expansion in synthetic data

Here we demonstrate the effectiveness of expanding ERACs on synthetic data using the algorithm proposed in Sect. 5.1. We are interested to know whether the expanding algorithm ERAC_exp can retrieve the injected ERACs from the top-K ERACs identified by the detection algorithm ERAC_H.

For the synthetic data generation, we set $|E| = 200$, $|F| = 10$ and the size of the injected ERAC to 10. For simplicity, we inject one ERAC into each synthetic data. We run the synthetic generation algorithm five times so that we have 5 synthetic datasets, each with one injected ERAC.

Next we apply the ERACD_H with $N = 3, 5, 7, 9$ and $K = 10$ on each of the five synthetic datasets. It turns out that for each setting of N, all top-10 ERACs returned are subsets of the corresponding injected ERAC.

For each of the five synthetic datasets, we expand top-10 ERACs for each setting of $N = 3, 5, 7, 9$ and run our expanding algorithm ERAC_exp. Thus, we run it 200 ($5*4*10=200$) times altogether. To measure the performance of the algorithm, we measure the "success rate" of the algorithm in retrieving exactly the injected ERAC over the five synthetic datasets. Otherwise, we give a zero score.

We observe in Table 3 that when N gets larger, ERAC_exp successfully retrieves the injected ERAC for all the five synthetic datasets. When $N = 3$, there are 15 cases out of 200 where the algorithm fails to expand some top-K ERACs to the original injected ERAC of size 10. This is because, when N is small, the detected ERAC of size N is likely to have extreme patterns dissimilar with the injected ERAC. Thus, in the greedy expansion process, the heuristic in ERAC_exp does not favor the other

Table 3 The results of ERAC expansion in five synthetic datasets

N	3	5	7	9
Success rate	35/50	50/50	50/50	50/50

Success rate reflects the ratio of successfully retrieving the injected ERAC by exp in five synthetic datasets

members in the injected ERACs and ends up with a different expanded ERAC than the injected one. When N is larger, it is more likely for the detected ERACs to have similar extreme pattern as the injected ERAC, leading to a very high success rate.

7.2 Results on web spam data

As reported by Becchetti et al. (2008), 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 from 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 the top or bottom as measured by the homogeneity of their incoming neighborhood.

Given a node e, we define the incoming **Neighborhood Feature** or iNF of a feature f in such a way that a node with homogeneous incoming neighborhood has a small iNF value, while a node with heterogeneous incoming neighborhood has a large iNF value. Specifically,

$$iNF(f, e) = median_{e', e'' \in iNBR(e), e' \neq e'' \neq e} dist(e'.f, e''.f)$$

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 the web host graph from the WEBSPAM-UK2006 dataset[2] published by Yahoo! Research. We adopt the 96 content features provided by Castillo et al. (2007) and Becchetti et al. (2008), 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. The incoming neighborhood features are derived from these content and structural features. These iNFs are used to rank web hosts in the following experiments.

We iteratively remove entities with less than 2 incoming neighbors, assuming they are not spammers. This is because spammers are more likely to have many incoming

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

neighbors and those spammers having few incoming neighbors are of little spamming power anyway. This leaves one big connected web host graph, with 5634 nodes (1709 spammers) associated with 102 features.

Since some of the 102 features in the web spam dataset are dependent on each other, we apply algorithm ERACD_H designed for dependent feature set to compute the top-K ERACs of size no greater than N in real-life datasets.

7.2.1 Effectiveness of ERAC detection algorithm

Applying the ERACD_H algorithm with $K = 1000$ and $N = 12$, ERACD_H finishes in 5225 seconds, less than 2 h. It returns 258 maximal IFSs from the 102 features. Among the maximal IFSs, 21 are of size 3, the largest size of all. The top ERAC returned by ERACD_H consists of 12 hosts, all true spammers (including *englandguide.co.uk* and *posters.co.uk* that are still actively spamming despite having been labeled as spammers since 2006). This collection is associated with the maximal IFS {"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 in the page content.

Moreover, we use representative extremity indices of this collection to 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 suggests 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.

7.2.2 Comparison to spam detection approaches

We now compare our ERACD_H algorithm with unsupervised TrustRank in the work of Gyöngyi et al. (2004) and Gyöngyi et al. (2006); and supervised decision tree techniques employed by Castillo et al. (2007) and Becchetti et al. (2008). Note that these spam detection approaches aim at detecting individual spammers, not spam collections. We therefore treat the websites in our top-K ERACs as individual spammers to compare with the precisions of those of TrustRank and decision tree based methods.

TrustRank starts with a seed set of trusted nodes, and propagates their scores by simulating a random walk with restart to the trusted pages. The estimated non-spam mass of a page used by Gyöngyi et al. (2006) is the amount of score it receives from trusted pages. We refer to this non-spam mass as the trustrank score. The lower the trustrank score of a node, the more likely it is a spammer.

To select the trusted nodes, we follow the guidance of Castillo et al. (2007) and randomly sampled 3800 hosts from the **.UK** domain in the Open Directory Project.[3] In computing the trustrank score, we set the probability of following an out-link from a trusted web page to 0.85. 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

[3] http://rdf.dmoz.org/.

(mp). Thus, the hosts at the top are 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 decision tree DT, we use J48 of Weka[4] with 5-fold cross validation. The features used are the same set of derived neighborhood features for ERACD_H. 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 $EL(\mathsf{DT})$.

Our heuristic algorithm works incrementally such that each loop of n from 1 to N 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_H approach with collections of various sizes by keeping the intermediate top-K ERACs produced by ERACD_H for various $n \leq N$. In the following experiments, we set $n \in \{4, 8, 12\}$.

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

Figure 7a, b and c plot the precision against K for ERACD_H versus {DT, TR_hp and TR_mp}. In the figures, ERACD_H is represented by a solid line, while the competing approaches are in doted lines.

As we observe in the figures, ERACD_H outperforms DT, TR_hp and TR_mp for all n settings. This demonstrates that our approach, although not specifically designed to detect 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 $K = 1{,}000$. The low recall levels are expected, as our approach is designed to discover anomalous collections of websites sharing collective extreme trait, not spammers of all types with or without collective extreme trait.

Next, to check whether spammer with collective extreme traits detected by our approach can be discovered by others, we compute the overlap between the top-$\tau(n, K)$ websites returned by our approach and that of each competing approach.

The overlap ratio as $\frac{|EL(\mathsf{ERACD\_H},\tau(n,K)) \cap EL(O,\tau(n,K)|}{\tau(n,K)}$, where $O \in \{\mathsf{TR\_hp}, \mathsf{TR\_mp}, \mathsf{DT}\}$. For $K = 1{,}000$ and $n \in \{4, 8, 12\}$, all the overlap ratios are smaller than 0.05, indicating ERACD_H detects unique spammer hosts with collective trait that are missed by the competing methods.

7.2.3 Comparison to anomaly detection approaches

The general density-based and clustering-based anomaly detection approaches are relevant state of the art to compare with. The density-based approach returns individual websites whereas the clustering-based one returns collections of websites. We evaluate

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

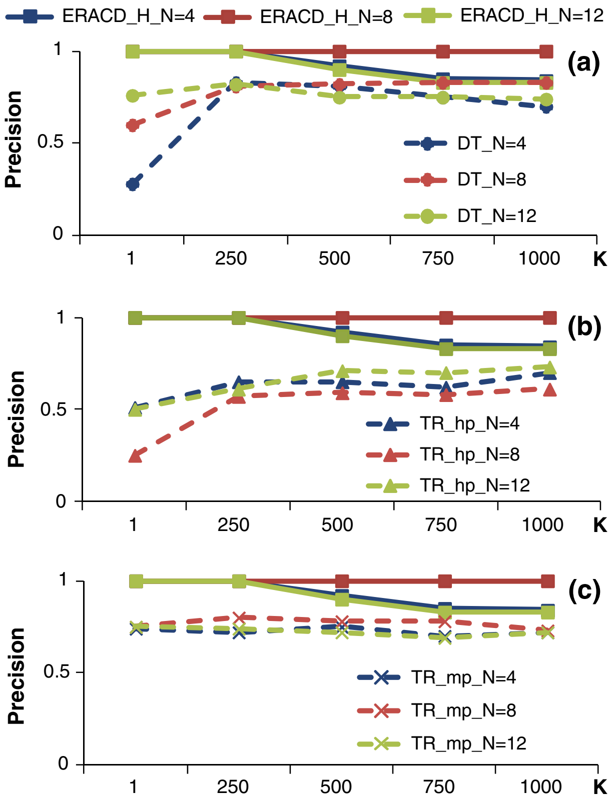


Fig. 7 Comparison of ERACD_H to other approaches in terms of precision. **a** ERACD_H versus DT, **b** ERACD_H versus TR_hp, **c** ERACD_H versus TR_mp

the anomaly score and precision of the collections returned by our approach against these two approaches.

We first show the comparison with the density-based approach proposed by Breunig et al. (2000). Following the experiment design of the previous section, we compare the results of ERACD_H with collections of various sizes ($n \in \{4, 8, 12\}$) and $K = 1{,}000$. Figure 8 shows the precision curves, which suggest that the top websites 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.

On the other hand, clustering-based outlier detection approaches consider "small clusters" to be anomalous. We follow Loureiro et al. (2004) in defining a small cluster as the one with a size smaller than half of the average cluster size.

We apply agglomerative hierarchical clustering with complete link and Euclidean 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 once the

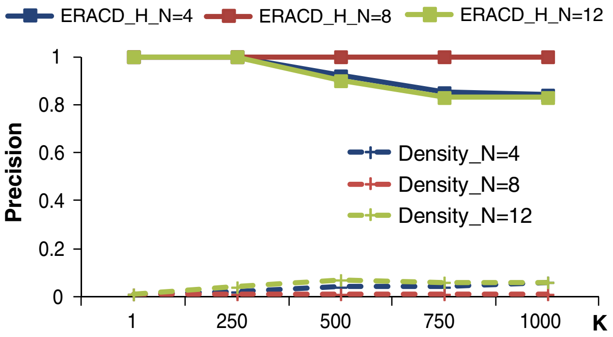


Fig. 8 ERACD_H versus density-based method on web spam data in terms of precision

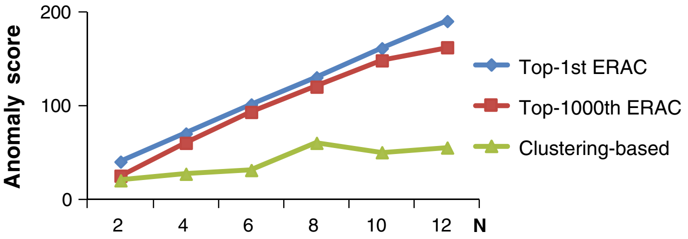


Fig. 9 ERACD_H versus clustering-based method on web spam data in terms of anomaly scores

combination of the next two clusters would cause the average size of all the clusters to rise above $2 \cdot N$. This is to make sure that all resultant clusters of size smaller than 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 the comparison, we keep the intermediate top-K results as N varies, and show the top-1 ERAC and top-1000 ERAC of ERACD_H for each N together with the most anomalous cluster discovered by the clustering-based approach.

In Fig. 9, we see that for all $N = \{2, 4, 6, 8, 10, 12\}$, the collections discovered by our approach are more anomalous than the clustering-based ones, because ERACD_H is optimized to collections that exhibit extreme behaviors.

To take a close look of the extremity of the top ERAC and the most anomalous cluster produced by the clustering-based approach on each of the three features in the maximal IFS, we use star charts to plot the relative rankings of the members in the two collections. For better visualization, we select the corresponding ERAC and cluster for $N = 4$. The result is shown in Fig. 10. The center point of each star chart represents the middle ranking (i.e., 2,817), as we have 5,634 websites in total. The distance to the center represents the extremity in ranking for a feature. It is easy to see that although the members in the most anomalous cluster by the clustering-based approach are similar to each other, they are not extreme on any of the features. In contrast, members of the top ERAC are extreme on all the three features. Interestingly, all the four websites in the top ERAC are spammers, whereas all the websites in the cluster are normal websites.

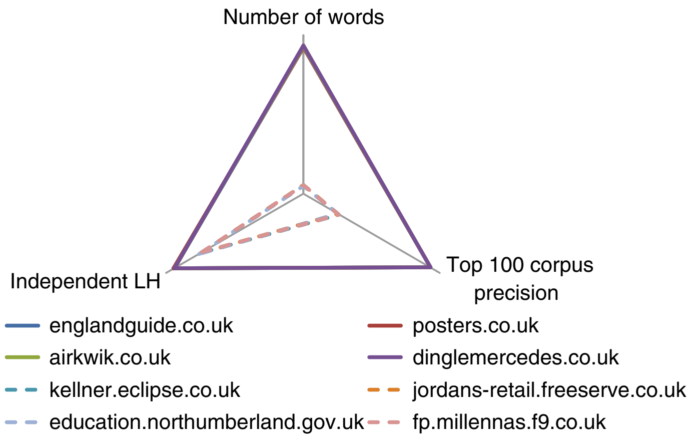


Fig. 10 Top ERAC of size 4 by ERACD_H (*in solid lines*) versus the most anomalous cluster of size 4 by clustering-based method (*in dotted lines*) on web spam data

As for precision, we compute for each given maximal IFS the ratio of the number of true spammers in all the small clusters over the total number of websites in all the small clusters. The maximal ratio is selected as the precision across all maximal IFSs. We also compute this precision for all $N = \{2, 4, 6, 8, 10, 12\}$. The results show that the precision of the small clusters is quite low for each N, with a maximum of 0.35 at $N = 10$ which is as good as random guess. This suggests that most of the websites in small clusters are not necessarily spammers. Obviously we cannot rely on clustering-based anomaly detection techniques to find spammer collections.

7.2.4 ERAC expansion in web spam data

We also apply our expansion algorithm ERAC_exp on this web spam data after we retrieve the top-1000 ERACs for each parameter setting of $N = \{4, 8, 12\}$. For each setting of N, we feed every top-1000 ERACs into ERAC_exp.

Interestingly, for all settings of N, the top-1000 ERACs are expanded to the same ERAC of size 79. This suggests the top-1000 ERACs are part of a larger cohesive ERAC. Note that all detected ERAC of different sizes have the same extreme pattern. In particular, they are located at the extreme top positions of the maximal IFS, {"Number of words", "Top 100 corpus precision", "Independent LH" }. This is the reason why the expansion algorithm ERAC_exp is able to retrieve the same ERAC for all the settings. The anomaly score of the expanded ERAC of size 79 is 1345.34, much larger than the original ERACs of small sizes, which suggests that ERAC_exp successfully retrieve the much more anomalous superset.

We also measure the precision of this expanded ERAC and compare with the competing methods {DT, TR_hp and TR_mp}. As the expanded ERAC has a size of 79, we take the top-79 spammers returned by each approach and measure its precision. As shown in Table 4, our expansion is still more effective in identifying spammers than the competing methods.

Table 4 The precisions of ERAC expansion and other competing methods

Method	ERAC expansion	DT	TR_hp	TR_mp
Precision	0.80	0.76	0.71	0.69

7.3 Results on IMDB data

From the IMDB dataset,[5] we focus on actors and actresses participating in movies shown between 1990 and 2008. We extract the 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). After dropping those actors who have missing feature values, we are left with 183 actors.

7.3.1 ERAC detection in IMDB data

We apply ERACD_H on this preprocessed IMDB dataset with $K = 3$ and $N = 3$. We set K and N to be small so that the results are easier to analyze. ERACD_H finishes in 202 seconds. There are 12 maximal IFSs extracted from the 6 features, including two maximal IFSs with the largest size 3. Each maximal IFS returns a top ERAC. ERACD_H chooses the maximal IFS that leads to the most anomalous ERAC, which is {number of movies, average movie budget, average payback}.

Using this maximal independent feature set, we also run the clustering-based approach on the IMDB dataset. Since we set $N = 3$ for ERACD_H, it is only fair for the clustering-based approach to assume that clusters of size no greater than 3 are anomalous. Altogether, the clustering-based approach returned 29 clusters, including 3 anomalous ones, all of which are less anomalous than the top-3 ERACs returned by ERACD_H.

Specifically, the three anomalous clusters returned by the clustering-based approach are {Lohan Lindsay, Witherspoon John, Madonna}, {Hudson Kate, Walker Paul, Crudup Billy} and {Duchovny David, Hawke Ethan}. The top-3 ERACs are {Grint Rupert, Radcliffe Daniel, Watson Emma}, {Bloom Orlando, Grint Rupert, Watson Emma}, {Brando Marlon, Grint Rupert, Watson Emma}. We plot their relative feature rankings in Fig. 11a, b and c. The center point of each star chart represents the middle ranking (i.e., 91), as we have 183 actors. The distance to the center point represents the extremity in ranking for a feature. By visual inspection, we find that the ERACs are more extreme on all the three features, compared to the three clusters returned by the clustering-based approach. This demonstrates again that while clustering-based approaches could return sets of similar actors, they are not designed to capture interesting collections that exhibit extreme behavior.

We note that several of the ERACs are potentially useful in practice. For example, a movie producer may want to find actors who perform in few large-budget movies

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

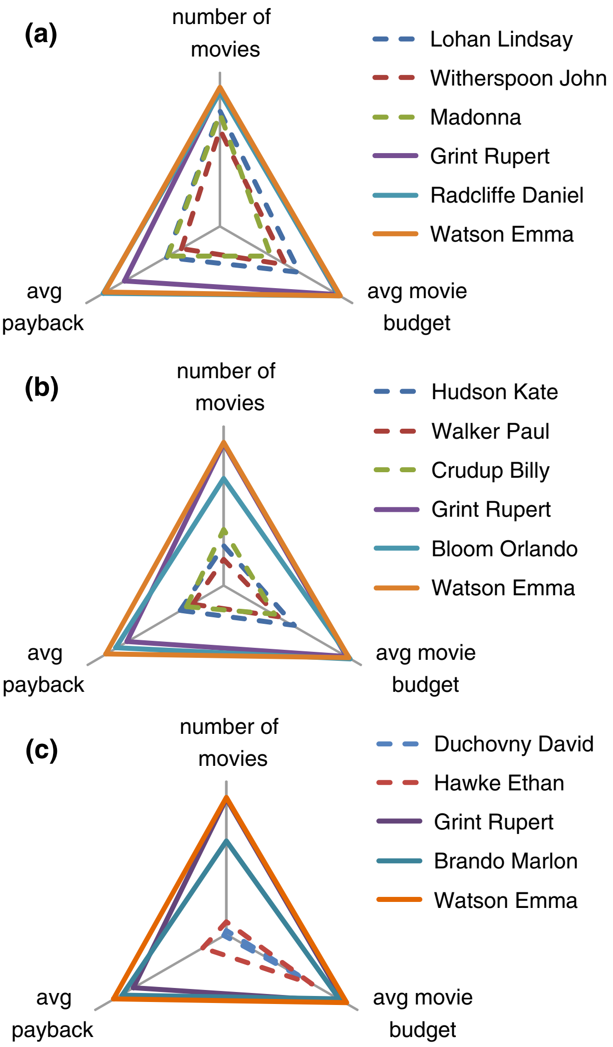


Fig. 11 ERACD_H versus clustering-based method on IMDB data in terms of top-1st collections shown in (**a**), top-2nd collections shown in (**b**), and top-3rd collections in (**c**)

but have big payback. Our results suggest the producer should go for the Harry Potter actors, or the other top ERACs.

7.3.2 ERAC expansion in IMDB data

Algorithm ERAC_exp is applied to the top-3 ERACs detected by ERACD_H. Since all the top-3 ERACs have the same extreme patterns on the maximal IFS {number of movies, average movie budget, average payback}, they expand to the same ERAC of size 12, which is {Grint Rupert, Radcliffe Daniel, Watson Emma, Bloom Orlando, Brando Marlon, Dane Eric, Donner Richard, Englund Robert, Frakes Jonathan, Gibson Tyrese, Howard Bryce Dallas, Jackson Janet}. As expected, this expanded ERAC has the same extreme pattern as the top-3 ERACs. The anomaly score grows from 37.3 for the original top ERAC of size 3, to 53.18 for the expanded superset of size 12.

To compare fairly with our ERAC detection algorithm, the clustering-based algorithm assumes that clusters that are no larger than 12 are anomalous, as our expanded ERAC is of size 12. Altogether, the clustering-based approach returned 8 clusters, including 2 anomalous ones. One cluster is of size 12 with anomaly score of 13.7, which contains {Gere Richard, Chase Chevy, Baldwin William, Jackman Hugh, Zellweger Renee, Daniels Jeff, Penn Sean, Barrymore Drew, Hackman Gene, Duchovny David, Vaughn Vince, Grant Hugh} The other is of size 8 with anomaly score of 6.84, which contains {Scorsese Martin, Silverstone Alicia, Jackson Samuel L., Farrell Colin, Kirkland Sally, Bower Michael, Brosnan Pierce, Ryder Winona}. They are much less anomalous than the expanded ERAC. These results show that our ERAC expansion successfully retrieves a larger ERAC that is more anomalous than the original ERACs as well as the anomalous clusters returned by the competing method.

7.4 Results on Chinese online forum data

In this section, we apply our algorithm ERACD_H to identify the infamous Chinese online "water army" (网络水军) spammers,[6] who are hired to post or comment on threads in many popular online forums, with the aim of influencing public opinion on targeted events or products. It is reported that during the Qihoo 360[7] versus Tencent QQ[8] dispute, both sides hired water armies to post favorable comments on themselves while disparaging the other.[9] Another reported event is in the Chinese dairy industry, where the brand manager of MengNiu (蒙牛) company was arrested for hiring online water army to frame the competing company YiLi (伊利).[10]

Water army operates by soliciting various tasks from internet public relations agencies (e.g., shuijunwang.com) and get paid according to the quality and quantity of posts or comments generated on popular online forums. It is often the case that hundreds or thousands of spammers are hired to form a water army that participates very actively in many popular online forums.

Each water army spammer commonly post comments from multiple user accounts. This helps to hide his/her true identity, as readers may not suspect that these comments come from the same person due to the different account names. If some lazy spammers simply copy and paste his message multiple times in the same post or in multiple posts, they do not get paid by the agencies, as the spammers are instructed not to perform such easily detectable behavior but to post relevant and meaningful messages using

[6] http://en.wikipedia.org/wiki/Internet_Water_Army.

[7] www.360.cn, the number 1 computer network security service provider in China.

[8] www.QQ.com, the number 1 instant messaging and online community service provider in China.

[9] http://www.chinadaily.com.cn/bizchina/2010-11/05/content_11509557.htm.

[10] http://www.chinadaily.com.cn/china/2010-10/21/content_11437735.htm.

Table 5 Three focused events for detecting water army in Tianya.cn

Events	Time period	Num of threads	Num of users
360 vs QQ	2010.9-2010.12	413	3545
Yaojiaxin (药家鑫)	2010.10-2011.4	345	6443
Ligangmen (李刚门)	2010.10-2011.1	359	3269

different accounts. This makes detecting water army spammers difficult, as they put in effort to post non-repetitive and relevant comments to appear like normal users.

We detect spammer collections in Tianya.cn[11], one of the most popular online Chinese forums. We focus on three hot events namely, 360 versus QQ, Yaojiaxin (药家鑫), and Ligangmen (李刚门), which are highly suspected[12] to involve water army activities. For each event, we submit the event name as query to the search interface of Tianya.cn, which returns the 750 most relevant threads. We keep only those threads that are posted within the relevant period of each event. The start and end dates are determined from news reports and are shown in Table 5.

With the relevant threads, we extract all the user accounts that have posted in the threads. We then filter accounts by their membership scores. The membership score assigned by Tianya.cn reflects how active an account is and how much contribution it has made writing acknowledged posts and getting involved in different interactive activities. Since a water army spammer normally has many accounts and would not take too much effort to build up membership scores for each of them, we remove the top 50 percent of accounts that have high membership scores. We also filter out accounts with only one post or comment, as they are also not likely to be involved in water army. Table 5 summarizes the profile of the three events after the aforementioned preprocessing.

Given a particular event, we treat each thread of this event as a feature, and the number of times that an account comments in this thread as the feature value. When accounts are ranked according to each thread in descending order, we expect collections of water army spammers to appear in top positions of a few threads, as a few spammers are collaborating, or the same spammer is using multiple accounts in these threads. This collective extreme behavior of water army spammers fits our ERAC principles. Note that a normal active account may also appear in the top positions of a few threads, but not likely in the top position of many threads consistently.

7.4.1 Effectiveness of ERAC Detection Algorithm

We set $N = 5$ and $K = 1$ for our heuristic algorithm ERACD_H, which finishes in a couple of hours for each event. The results of the top ERAC returned for each event along with running time are listed in Table 6. The third column $|F^s|$ denotes the number of threads w.r.t. which the ERAC is significant (i.e., the corresponding representative

[11] http://www.tianya.cn.

[12] http://www.shuijunshiwan.com/wenku/.

Table 6 Results for computing top ERAC for each event in Tianya.cn

Events	Top ERAC	$\|F^S\|$	avg-r (avg-i)	Running time (s)
360 vs QQ	湃大性,ayaya118,时光倒溜 大红鹰的翅膀,让读书人有钱	13	50.2(3.7)	4,675
Yaojiaxin (药家鑫)	武者刺,probit,国强民负 洪德阀紫,天易在线	35	67.3(2.9)	7,834
Ligangmen (李刚门)	www9w,热血春秋,muzi840719 曾经是败类，苍天不开眼	15	45.8(3.6)	3,980

p value is below α). For each significant feature, we find the corresponding extremity index r and intersection i value as in $p(i, r, n)$. avg-r and avg-i in the fourth column shows their average values.

For example, in the event of 360 versus QQ, the top ERAC is {{湃大性,ayaya118,时光倒溜,大红鹰的翅膀,让读书人有钱}}. By checking the representative p value of this collection w.r.t. each of the 357 threads, we know their collective extreme behavior is significant in 13 threads, with average $r = 50.2$ and average $i = 3.7$. This suggests that they actively commented on the 13 threads. Reading through the comments of the five accounts, it is clear that their posts are either against 360 or for QQ and the posts are generated within a 5-day period. Furthermore, most of their comments were posted to the threads related to 360 versus QQ. All the above observations clearly indicate collaboration among the five accounts.

7.4.2 User evaluation

Since we do not have the ground truth on water army spammers in Tianya.cn, we recruit human evaluators to judge the accounts on both individual level and collection level.

We hired four Chinese evaluators who are familiar with Tianya.cn forum and the three hot events. They are requested to read the Chinese Wiki page and news reports on water army beforehand. For each user account, the evaluators are provided with the account's homepage in Tianya.cn, which contains the membership score, number of visits, last visit time, registration time and all the threads the account has posted or commented on. As Tianya.cn lists each account's threads page by page, it is very tedious for the evaluators to navigate through the threads and the comments posted from the account. We therefore crawled each account's home page, all the threads as well as the comments in each thread, and presented them together to the evaluators.

User evaluation at individual level

Firstly, we are interested to know whether the user accounts in the top-K ERACs really are water army spammers on Tianya.cn.

To alleviate the demand placed on the evaluators, we only select accounts in the top ERAC of each focused event, altogether 15 of them, for judging. The evaluators are asked to judge whether each account is a spammer according to his understanding of how a normal account in any online forum should behave. The evaluators assign a

Table 7 User evaluation at individual level

Event	User accounts	E1	E2	E3	E4	Avg.
360 vs QQ	大红鹰的翅膀	5	5	5	5	5
	湃大性	3	4	4	4	3.75
	让读书人有钱	5	3	5	4	4.25
	时光倒溜	4	4	4	4	4
	ayaya118	5	3	5	5	4.5
Ligangmen (李刚门)	www9w	3	3	3	4	3.25
	热血春秋	3	1	1	2	1.75
	曾经是败类	4	4	5	3	4
	苍天不开眼	5	5	4	4	4.5
	muzi840719	5	4	4	4	4.25
Yaojiaxin (药家鑫)	国强民负	3	5	4	2	3.5
	武者刺	3	2	2	3	2.5
	天易在线	4	4	4	4	4
	洪德阀紫	2	4	5	5	4
	probit	3	5	5	4	4.25

E1–E4 denote the four evaluators

score between 1 and 5 to each account, with 1 being normal, 5 being spammer and 3 being not-sure. The final score of an account is the average score assigned by the evaluators. We consider the accounts with final score greater than 3 to be spammers and the rest to be normal.

The scores from the evaluators are shown in Table 7. Out of 15 accounts identified by our approach, 13 of them are judged by the evaluators to be spammers. This gives a precision of 0.87.

For user account "大红鹰的翅膀", the evaluators report that it has not logged in after the event of 360 versus QQ, almost all of its threads are about 360 versus QQ, and almost all of its comments contain links to a voting website supporting 360. These observations cast strong suspicions on the account.

There are two accounts "热血春秋" and "武者刺" that the evaluators give low scores as they find that although the accounts are involved in many threads on the corresponding event, they also commented on other events, which makes them less suspicious. Our approach considers only the number of comments across threads but not the actual content of the comments and therefore misclassified them.

User evaluation at collection level

Now we conduct our user evaluation at collection level. We are interested to know (i) Are the accounts in the top ERAC spamming in the corresponding significant threads? (ii) Are the ERAC rankings consistent with the evaluators' perception? In this user evaluation, we focus on the 360 versus QQ event.

We derive a random set of accounts to compare with our top ERAC. We randomly choose from the users whose number of comments are comparable to the

Table 8 User evaluation at collection level

ERACs	E1	E2	E3	E4	avg
Top-1	2.56	3.18	4.01	3.08	3.21
Random collection	1.14	1.21	1.91	2.17	1.61

E1–E4 denote the four evaluators

average number of comments of the accounts in the top ERAC. This way, the chosen accounts are as active as the members in top ERAC. We end up with the ERAC {珠珠雨粒,treegreen12010,爱顶嘴,再见能否,quanyeke}.

We present the 5 accounts in the top ERAC with its 13 significant threads, together with the 5 accounts in the random user collection with its 16 significant threads to the evaluators. For each account, the evaluators are given its comments in all the significant threads and asked to judge whether the account is spamming in a particular thread based on the comments it posted.

For each account, the evaluator assigns a score from 1 to 5 according to each of the significant thread, with 1 being non-spamming, 5 being spamming and 3 being not-sure. The evaluators also need to provide their reasons.

The final score of an ERAC is the average score of its member accounts across the significant threads. Table 8 shows the results. As we see from the table, the average score of the top ERAC is larger than 3, indicating that its members indeed are spamming the significant threads. We also observe that the evaluators consider the top ERAC more suspicious than the less anomalous random account collection, suggesting our ranking are in line with the evaluators' perception.

7.4.3 ERAC Expansion in Chinese Online Forum Data

The previous experiments on detecting water army spammer collections revealed ERACs with predefined size $N = 5$. To uncover water army collections that are supersets of the detected ERACs, we expand the top-K ERACs returned for the event of 360 versus QQ by the ERAC_exp algorithms described in Sect. 5.1.

We first set $N = 5$, $K = 100$ and run ERACD_H to return the top-100 ERACs of size up to 5. We then apply ERAC_exp to expand each ERAC.

Interestingly, 67 of the 100 ERACs expand to the same ERAC of size 11, whereas the remaining 33 ERACs expand to another ERAC of size 14. The first expanded ERAC of size 11 is significant on 13 features, which are exactly the same 13 features on which the original 67 ERACs of size 5 are significant. On the other hand, the second expanded ERAC of size 14 are significant on 15 features, which overlap with the significant features of the original 33 ERACs on 14.63 features on average.

These observations indicate that the added members in the expanded ERACs have very similar extreme patterns as members in the original ERACs. Specifically, in Tianya.cn, the accounts in the same ERAC posted many times on almost the same set of threads, which substantiates our expansion heuristic.

The 67 ERACs of size 5 have a maximum anomaly score of 314.98, whereas the expanded ERAC of size 11 scores 661.41. On the other hand, the maximum anomaly score of the 33 ERACs is 309.67, much less than 754.36, the anomaly score of the ERAC of size 14 after expansion. This shows that ERAC expansion indeed produces much more anomalous supersets.

Another observation is that the two expanded ERACs do not overlap, neither do their significant features. After reading through the posts from users in each expanded ERAC on their corresponding significant features (i.e., threads), we discover that the two collections of users had condemned 360 harshly with emotive remarks, but providing little factual support. This suggests the existence of at least two independent water army collections spamming on Tianya.cn on behalf of QQ.

8 Conclusions and future work

In this paper, we detect a new type of anomaly collections, called extreme rank anomaly collections (ERAC). Members of an ERAC exhibit similar extreme behaviors and thus appear at extreme ranking positions of multiple features. Due to the existence of large number of ERACs of various sizes, for efficiency reasons, we first propose the problem of discovering top-K ERACs with a predefined size limit. To uncover the anomalous supersets of the detected ERACs, we then propose the problem of ERAC expansion without having to specify the size of supersets.

We apply ERAC detection and expansion algorithms to discover injected ERACs in synthetic datasets, web spammer collections in a web spam dataset, unusual actor collections in an IMDB dataset and water army spammer collections in a Chinese online forum dataset. The results show that our algorithms are able to uncover the anomalous collections in all datasets. Moreover, we achieve higher precision in web spam detection than existing approaches. We detect anomalous actor collections that are not easily identified by other approaches. We reveal collaborating water army spammer collections in the Chinese online forum, which are corroborated by human evaluators.

Limitations and future directions

Due to the fact that our proposed anomalousness measure of collections does not enjoy the downward closure property, even the most efficient algorithm proposed ERACD_H takes a few hours to run on real datasets of thousands of entities. Our naive heuristic is faster than our sophisticated heuristic but with much lower precision/recall. For future work, more efficient algorithms could be developed that better balance between precision/recall and running time.

The second future direction could be to create a real-life dataset with ground truth on anomalous collections instead on individual anomalies. A dataset with ground truth on collection level is more suitable for evaluating our approach by both precision and recall than any dataset with ground truth on individual level, where high precision and low recall is often expected. This is because our approach is not tuned to detect all individual anomalies with or without collective traits.

We also plan to embark on generalizing our anomaly collection definition. Our ERAC detection frameworks are based on the "extreme rank" concept, where members of an anomaly collection consistently appear at extreme ranking positions of certain features. However, it is possible to have an anomalous collection defined to have members appearing consistently at some non-extreme positions. This general definition of anomaly collection would be able to utilize features where the behaviors are not extreme. One solution is to change the "extreme index" concept to a "closeness interval" in the entity ranking list. Given a collection and a closeness interval, the number of entities of this collection appearing in this closeness interval still follows the hypergeometric distribution. It would also be interesting to develop both accurate and efficient algorithms for such anomalous collections and evaluate them on real data.

Acknowledgments Part of the work was done when the first author was pursuing PhD in School of Information Systems, Singapore Management University, Singapore. This work is partly supported by National Nature Science Foundation of China (NSFC Grant No. 61300126).

References

Agrawal R, Srikant R (1994) Fast algorithms for mining association rules in large databases. In: Very Large Data Bases Conference (VLDB), pp 487–499

Arias-Castro E, Candes EJ, Durand A (2011) Detection of an anomalous cluster in a network. Ann Stat 39(1):278–304

Avis D, Fukuda K (1993) Reverse search for enumeration. Discret Appl Math 65:21–46

Barnett V, Lewis T (1994) Outliers statistical data. Wiley, New York

Becchetti L, Castillo C, Donato D, Baeza-Yates R, Leonardi S (2008) Link analysis for web spam detection. ACM Trans Web 2(1):2:1–2:42

Bomze IM, Budinich M, Pardalos PM, Pelillo M (1999) The maximum clique problem. Kluwer Academic Publishers, Norwell

Breunig MM, Kriegel HP, Ng RT, Sander J (2000) Lof: identifying density-based local outliers. In: ACM sigmod record, pp 93–104

Castillo C, Donato D, Gionis A, Murdock V, Silvestri F (2007) Know your neighbors: web spam detection using the web topology. In: International Conference on Research and Development in Information Retrieval (SIGIR), pp 423–430

Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: a survey. ACM Comput Surv 41(3):15:1–15:58

Chua CEH, Wareham J (2004) Fighting internet auction fraud: an assessment and proposal. Computer 37(10):31–37

Dai H, Zhu F, Lim EP, Pang H (2012) Detecting extreme rank anomalous collections. In: SIAM International Conference on Data Mining (SDM), pp 883–894

Duan L, Xu L, Liu Y, Lee J (2009) Cluster-based outlier detection. Ann Oper Res 168(1):151–168

Dunnett CW (1955) A multiple comparison procedure for comparing several treatments with a control. J Am Stat Assoc 50(272):1096–1121

Eppstein D (2005) All maximal independent sets and dynamic dominance for sparse graphs. In: Symposium on Discrete Algorithms (SODA), pp 451–459

Ester M, Kriegel HP, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: International Conference on Knowledge Discovery and Data Mining (KDD), pp 226–231

Fetterly D, Manasse M, Najork M (2004) Spam, damn spam, and statistics: using statistical analysis to locate spam web pages. In: Proceedings of the international workshop on the web and databases, pp 1–6

Guha S, Rastogi R, Shim K (1999) Rock: a robust clustering algorithm for categorical attributes. In: International Conference on Data Engineering (ICDE), pp 512–521

Gyöngyi Z, Garcia-Molina H, Pedersen J (2004) Combating web spam with trustrank. In: Very Large Data Bases Conference (VLDB), pp 576–587

Gyöngyi Z, Berkhin P, Garcia-Molina H, Pedersen J (2006) Link spam detection based on mass estimation. In: Very Large Data Bases Conference (VLDB), pp 439–450

Harkness WL (1965) Properties of the extended hypergeometric distribution. Ann Math Stat 36(3):938–945

He Z, Xu X, Deng S (2003) Discovering cluster-based local outliers. Pattern Recogn Lett 24(9):1641–1650

Herrera F, Carmona CJ, Gonzalez P, del Jesus MJ (2011) An overview on subgroup discovery: foundations and applications. Knowl Inf Syst 29(3):495–525

Kendall M (1948) Rank correlation methods. Griffin

Klösgen W (1996) Advances in knowledge discovery and data mining. American Association for Artificial Intelligence

Knorr EM, Ng RT (1998) Algorithms for mining distance-based outliers in large datasets. In: Very Large Data Bases Conference (VLDB), pp 392–403

Lawler EL, Lenstra JK, Kan AHGR (1980) Generating all maximal independent sets: Np-hardness and polynomial-time algorithms. SIAM J Comput 9(3):558–565

Liu FT, Ting KM, Zhou ZH (2010) On detecting clustered anomalies using sciforest. In: European Conference on Principles of Data Mining and Knowledge Discovery (PKDD), pp 274–290

Loureiro A, Torgo L, Soares C (2004) Outlier detection using clustering methods: a data cleaning application. In: Proceedings of the data mining for business workshop

Pandit S, Chau DH, Wang S, Faloutsos C (2007) Netprobe: a fast and scalable system for fraud detection in online auction networks. In: International World Wide Web Conference (WWW), pp 201–210

Wrobel S (1997) An algorithm for multi-relational discovery of subgroups. In: European Conference on Principles of Data Mining and Knowledge Discovery (PKDD), pp 78–87

Wu T (1993) An accurate computation of the hypergeometric distribution function. ACM Trans Math Softw 19(1):33–43