Efficiently Discovering Unexpected Pattern-Co-Occurrences

Roel Bertens° Jilles Vreeken• Arno Siebes°

Abstract
Our world is filled with both beautiful and brainy people, but how often does a Nobel Prize winner also wins a beauty pageant? Let us assume that someone who is both very beautiful and very smart is more rare than what we would expect from the combination of the number of beautiful and brainy people. Of course there will still always be some individuals that defy this stereotype; these beautiful brainy people are exactly the class of anomaly we focus on in this paper. They do not posses intrinsically rare qualities, it is the unexpected combination of factors that makes them stand out.

In this paper we define the above described class of anomaly and propose a method to quickly identify them in transaction data. Further, as we take a pattern set based approach, our method readily explains why a transaction is anomalous. The effectiveness of our method is thoroughly verified with a wide range of experiments on both real world and synthetic data.

1 Introduction

The recognition of anomalies provides useful application-specific insights [1]. More specifically, the field of anomaly detection focusses on the identification of data that significantly differ from the rest of the dataset — so different that it gives rise to the suspicion that it was generated by a different mechanism. Such an anomaly may, e.g., occur because of an error, it may be an outlier, or it may be a highly unexpected data point. It is hard, if not impossible, to automatically distinguish between such different possible origins. Hence, anomalies should be inspected manually to decide whether it should, e.g., be removed, corrected, or simply remain in the data "as is". One should thus preferably not report an overly large list of potentially anomalous data points and, at the very least, that list should be ordered such that the most anomalous data points appear on top.

For transactional data anomaly detection usually boils down to pointing out those transactions that show unexpected behaviour. This unexpected behaviour can manifest itself in different ways and each detection algorithm is limited to find only those anomalies which fit the corresponding framework. For example, much work has been done to detect unexpected behaviour which can be expressed by the compressed size of a transaction given a pre-processed model [23, 3]. That is, transactions that badly fit the norm of the data are deemed to be anomalous. Another example is to score transactions based on the number of frequent patterns that reside in it [13]. Yet another method scores transactions based on items missing from a transaction which were expected given the set of mined association rules [19]. All these methods have their own advantages, however, none of them is able to detect an anomaly based on the presence of multiple items in a single transaction that are not expected to occur together. Therefore in this work we focus on this class of anomalies, not to improve existing methods, but to improve the field of anomaly detection by making it more comprehensive. Since there are many ways in which a transaction can be anomalous, there should be a wide variety of algorithms detecting complementary sets of anomalies.

In addition to highlighting the transactions that show anomalous behaviour, our method describes anomalies in more detail by providing the most unlikely co-occurrence of patterns in that transaction. As an example consider a dataset containing people's drinking habits where roughly half of the people drinks soft drink $\mathbb{C}$ and the other half drinks soft drink $\mathbb{P}$. Now each individual who drinks $\mathbb{C}$ or $\mathbb{P}$ is not surprising. Moreover, someone drinking both $\mathbb{C}$ and $\mathbb{P}$ also does not seem surprising as it can be compressed well using the methods of [23, 3], it contains multiple frequent patterns [13] and there is nothing missing [19]. However, in this dataset almost everyone drinks either $\mathbb{C}$ or $\mathbb{P}$, but not both. Therefore, someone drinking both $\mathbb{C}$ and $\mathbb{P}$ is an anomaly, as drinking both is unexpected. We propose to score each transaction based on the most unlikely co-occurrence between patterns and therefore our method is able to find the described class of anomalies.

For this example, the score we introduce is based on the well-known interestingness measure *lift* of an association rule [26] (also known as its *interest* [5] and closely related to the *novelty* of an association rule [15]). More precisely, we take minus the log of the minimal lift of the two association rules $\mathbb{C} \to \mathbb{P}$ and $\mathbb{P} \to \mathbb{C}$. So, the difference is that we do not score a rule, but a transaction and do so by the minimal lift of all the rules that apply to this transaction.

The higher the score, the more anomalous the transaction. So, if both $\mathbb{C}$ and $\mathbb{P}$ are frequent, a transaction t containing both is anomalous if $\{\mathbb{C}, \mathbb{P}\}$ is infrequent, and the more infrequent it is, the more anomalous t is. Note that this is the opposite of *rare* patterns, or rare association rules [14]. For rare rules either $\mathbb{C}$, $\mathbb{P}$, or both should be infrequent, while the confidence of, e.g., $\mathbb{C} \to \mathbb{P}$ should be high.

For rare association rules either $\mathbb{C}$ or $\mathbb{P}$ is expected to

°Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands, {R.Bertens, A.P.J.M.Siebes}@uu.nl.
•Max Planck Institute for Informatics and Saarland University, Saarland Informatics Campus, Saarbrücken, Germany, jilles@mpi-inf.mpg.de.

be infrequent, and multiple or adaptive minimal support thresholds can be used for efficient discovery. Since we assume both $\mathbb{C}$ and $\mathbb{P}$ to be frequent and only $\{\mathbb{C}, \mathbb{P}\}$ to be infrequent such ideas cannot be used here. Rather, an exhaustive algorithm requires all frequent sets with a support equal or larger than 1, since any of these may be formed by an unexpected combination of patterns. Finding the most surprising transactions using this humongous set is infeasible on all but the most trivial data sets. For this reason we introduce a heuristic algorithm based on the code tables computed by algorithms such as KRIMP [27] or SLIM [24].

In extensive experiments we firstly show that this heuristic algorithm finds all anomalies we hide in synthetic data. Secondly, we show that the transactions found to be anomalous in real world data sets are indeed strange. For example, in the well-known *Adult* data set, the top-ranked transaction contains the very unexpected co-occurrence of someone whose attribute sex is female yet whose relationship status is husband. It is highly probable that this is a mistake, but it is certainly an anomaly the data scientist should be aware of before analysing the data set.

2 Notation

We consider transaction datasets D containing $|D|$ transactions. Each transaction t contains a subset, of size $|t|$, of the items from the alphabet Ω. Categorical data consists of $|A|$ attributes, where each attribute $A_i \in A$ has a domain Ω_i, and can also be regarded as transaction data by mapping each attribute value pair to a different item. We assume there is no missing data. All logarithms are to base 2, and by convention $0 \log 0 = 0$. We use $P(\cdot)$ to denote a probability function.

3 Anomalies in Transaction Data

What is an Anomaly? Anomalies are also referred to as abnormalities, discordants, deviants, or outliers in the data mining and statistics literature [1]. As we consider transaction data we use the following definition.

DEFINITION 3.1. *A transaction is anomalous when it deviates from what we expect considering the whole dataset.*

Given this definition an anomaly can manifest itself in different ways, resulting in multiple classes of anomalies for transaction data. In this section we recall two familiar classes of anomalies, define one new class, and we show how to identify all of them by formalising appropriate anomaly scores. We want to emphasise again that the scores for different classes of anomalies are complementary to each other. Further, for ease of interpretation and computation we take the negative log-likelihood for the scores in each class.

3.1 Class 0: Unexpected Transaction Lengths A transaction can be anomalous not as a result of the patterns it contains, but solely on the basis of its deviating length.

DEFINITION 3.2. *A class 0 anomaly is a transaction with significantly deviating transaction length.*

We propose an anomaly score which represents the number of bits needed to describe the transaction length given all transaction lengths in the data, i.e. for a transaction t we have

$$score_0(t) = -\log(P(|t|)) = -\log\left(\frac{|\{t' \in D \mid |t'| = |t|\}|}{|D|}\right) .$$

The intuition behind the subscript 0 for this score is that we take no patterns into account to identify these anomalies. As it is a fairly trivial score we will not further evaluate it.

3.2 Class 1: Unexpected Transactions When a transaction contains very little structure, i.e. few or no frequent patterns, it can be regarded to be anomalous.

DEFINITION 3.3. *A class 1 anomaly is a transaction that contains very little of the regularity conveyed by the rest of the dataset.*

The state of the art in transaction anomaly detection focusses on what we call class 1 anomalies. For example, OC^3 [23] scores transactions using a descriptive pattern set $\mathcal{S}$. Transactions containing few of these patterns but mostly singletons will get a higher score. That is, because such a transaction cannot be explained well by the pattern set that is descriptive for the data. We generalise this idea by defining a score based on the probability of a transaction. More formally, $score_1$ scores each transaction based on the number of bits needed to describe it, i.e. for a transaction t we have

$$score_1(t) = -\log P(t) \quad .$$

For compression based methods such as OC^3 this score is defined by the compressed length of the transaction given the model of the data. However, any method that can assign a probability to a transaction based on the whole data can be used here. Note that, as transactions are scored as a whole, this approach will unlikely detect unexpected co-occurrences of patterns. For example, using OC^3, all patterns that describe a transaction will contribute to its score independently. As much work has been done to detect these anomalies we will not further evaluate their identification, but focus on the next class of anomalies.

3.3 Class 2: Unexpected Co-occurrences The focus of this paper lies on identifying unexpected co-occurrences of patterns.

DEFINITION 3.4. *A transaction contains a class 2 anomaly when it contains two patterns that occur much less frequently together in the data than what could be expected from their individual supports.*

As this definition is somehow the opposite of that of a pattern, which is formed when two smaller patterns occur together more frequently than expected, we can also use the terms negative pattern or negative interaction pattern.

To identify anomalous transactions based on class 2 anomalies we would like to score a transaction based on the unexpectedness of the co-occurrences of the patterns contained in it. That is, we propose to rank a transaction based on its most unexpected pattern co-occurrence. Intuitively this means that for each transaction we compute the number of bits we need to explain the most unlikely co-occurrence given a pattern set $\mathcal{S}$ and the data. For a transaction t we thus have

$$score_2(t) = \max_{\{X,Y \in \mathcal{S} \mid X,Y \subseteq t\}} -\log P(XY) + \log\big(P(X) \times P(Y)\big) .$$

In the remainder of this paper we refer to $score_2$ as the UPC score, for Unexpected Pattern Co-occurrence. We compute $P(X)$ as X's support or relative frequency in the data.

Given a UPC score for a transaction we can readily explain its anomalousness as we know which co-occurrence of patterns is responsible for the score. Therefore our method has the nice property of producing very interpretable rankings.

Our score is related to the concept of lift [21] used in the context of association rules. In our setting we use it to describe the difference between two patterns appearing together in a transaction and what would be expected if they were statistically independent. Therefore, the higher our score the more unexpected the pattern co-occurrence.

Scores that are constructed to identify class 1 anomalies are not able to detect these class 2 anomalies as they look at all patterns *independently*. For example, OC^3 [23] and COMPREX [3] will not give a class 2 anomaly a higher score as both individual patterns are frequent and will add little to the anomaly score. Similarly, the frequent pattern based method from He et al. [13] and the method from Narita et al. [19] have no means to give higher scores to class 2 anomalies. As a result, methods for identifying class 1 anomalies do not identify unexpected co-occurrences, while these actually do indicate anomalous behaviour.

Which patterns to consider Given the relation between our score and the lift of association rules, a straightforward way to find high scoring transactions may seem to simply mine for low-lift association rules. However, to maximize the score the individual patterns X and Y should have a support as high as possible while XY should have a support as low as possible. That is, we should mine for all rules — including those with a support of 1 — to ensure that we do not miss the most interesting, most anomalous transactions.

Clearly this quickly becomes infeasible for all but the smallest data sets. Not only because discovering all these rules will take an inordinate amount of time, but also since the post-processing of all these rules necessary to identify the most surprising transactions becomes a rather daunting task.

The alternative we take is by starting from a set of patterns $\mathcal{S}$. We compute the score of each pair of patterns from $\mathcal{S}$ and identify those transaction in which pairs with a (very) high score occur.

Clearly, not just any pattern set will do as we want to find the highest scoring transactions. The set of all frequent patterns $\mathcal{F}$ will be far too large to be able to consider the interactions between each pair of patterns. In the worst case we need to consider each co-occurrence of patterns for each transaction, thus leading to a computational complexity of

$$O(|D| \times |\mathcal{F}| \times |\mathcal{F}|) \quad .$$

Choosing a higher minimum support will yield smaller pattern sets but as a result we might miss important patterns. We could use condensed representations such as closed [20] or non-derivable [6] frequent patterns to remove as much redundancy as possible, however these sets will still be too large. By sampling [12] patterns we can attain small sets of patterns, however, the choice of the size of the sample determines which anomalies one will (likely) find. A set that is too small might miss some important patterns, but a set that is too large probably contains redundancy and again becomes a bottleneck in our approach. Since it is not straightforward to choose the right size for the required pattern set, we choose to use KRIMP [27] or SLIM [24] to automatically find small descriptive pattern sets that describe the data well without containing noise or redundancy. Using these pattern sets it will hold that $|\mathcal{S}| \lll |\mathcal{F}|$. We thus dramatically reduce the complexity, making the UPC score practically feasible as we will show in our experiments in Section 6. Using such a vastly smaller set induces, of course, the risk that we miss anomalies. However in other research we have seen that the pattern sets chosen by KRIMP and SLIM are highly characteristic for the data. The experiments in Section 6 bear out that this is also the case here: all anomalies we inject in synthetic data are discovered using these small sets only.

4 How to use our scores

In the process of explorative data mining, one has to consider that all 3 classes of anomalies we identify give different insight, i.e., one should instantiate all 3 scores and investigate the top-ranked anomalies for each class. Here, our focus is of course on class 2 anomalies.

To determine which of the UPC top-ranked transactions to investigate, as well as to verify the significance of the scores, we propose two bootstrap methods. Recall that bootstrap methods consider the given data as a sample, and generate a number of pseudo-samples from it; for each pseudo-sample calculate the statistic of interest, and use the distribution of this statistic across pseudo-samples to infer the distribution of the original sample statistic [7].

4.1 Significance test For a synthetic dataset it is easy to test the significance of anomaly scores, as we can generate data with and without anomalies for which the resulting scores must clearly differ. For real world data this is unfortunately not the case as we do not know which and how much (negative) patterns the data comprises. Nevertheless, to give a measure of significance we use the following bootstrap approach. We randomly sample transactions from our original dataset (with replacement) to retrieve an equally sized new dataset. We repeat this a thousand times and save the highest anomaly score for each dataset. Then we repeat this process, but we first remove the transaction with the highest UPC score from the sample set. That is, the top-ranked anomaly is definitely not present in the bootstrap samples of the second kind and may or may not be present in the bootstrap samples of the first kind. The bigger the difference between the distributions of scores with and without the top-ranked transaction, the more significant the top-ranked anomaly.

4.2 Which transactions to investigate Choosing the right parameter value is never easy in explorative data mining. As the UPC score produces a ranking of all transactions, where higher scores indicate a higher chance on being anomalous, it does not need any parameters. To determine which transactions to investigate based on this ranking we employ Cantelli's inequality to identify the transactions that significantly differ from the norm.

THEOREM 4.1. ***Cantelli's inequality*** *[10]. Let X be a random variable with expectation μ_X and standard deviation σ_X. Then for any $k \in \mathcal{R}^+$,*

$$P(X - \mu_X \geq k\sigma_X) \leq \frac{1}{1+k^2} \quad .$$

Smets and Vreeken [23] proposed a well-founded way to determine threshold values to distinguish between 'normal' and anomalous transactions. The positive class comprises anomaly scores for 'normal' transactions and based on the distribution of these scores we can choose a threshold by choosing an upper bound on the false-negative rate (FNR). For example, if we choose a confidence level of 10%, Cantelli's inequality tells us that this corresponds to a threshold θ at 3 standard deviations from the mean, given by $\theta = \mu + k\sigma$, with $k = 3$ in this case. This means that the chance on a future transaction with an anomaly score above the threshold is less than 10%, see Figure 1.

To compute these thresholds we need the distribution of the positive class, i.e. the anomaly scores for all 'normal' transactions. Because we have only one dataset available which can contain both transactions from the positive and negative (actual anomalies) class, we use again a bootstrap approach. We generate bootstrap datasets by randomly sampling transactions (with replacement) from the original dataset. We then use all anomaly scores from all bootstrap datasets to estimate the distribution.

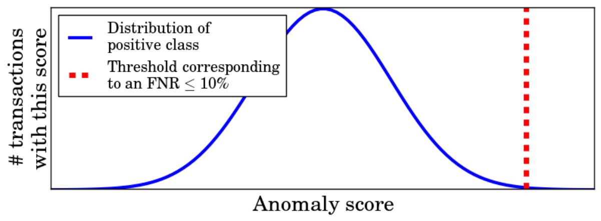

Figure 1: **Example of setting a threshold using Cantelli's inequality.** Based on the positive class we compute a threshold corresponding to a false-negative rate of 10%.

5 Related Work

In this paper we study anomaly detection in binary transaction data. As anomalies are referred to in many different ways, mostly with slightly different definitions, we refer to [17] and [1] for in-depth overviews on this field of research. In general, most anomaly detection methods rely on distances. Here we focus on discrete data, nominal attributes, for which meaningful distance measures are typically not available.

Of the methods that are applicable on transaction data, that of Smets and Vreeken [23] is perhaps the most relevant. They propose to identify anomalies as those transactions that cannot be described well by the model of the data, where as models they use small descriptive pattern sets. Their method OC^3 works very well for one-class classification, however it is not able to identify unexpected co-occurrences in the data. Akoglu et al. [3] proposed COMPREX which takes a similar approach in that they also rank transactions based on their encoded length. The difference is that they do not use a single code table, but a code table for each partition of correlated features. Although this method achieves very good results it is only suitable for categorical data and not for transaction data in general. Note that, following our generalised anomaly score for class 1 anomalies, any method that provides a probability for a transaction can be used. Examples based on pattern sets are those of Wang and Parthasarathy [28] and Mampaey et al. [16].

He et al. [13] rank transactions based on the number of frequent patterns they contain given only the top-k frequent patterns, and Narita et al. [19] rank transactions based on confidence of association rules but need a minimum confidence level as parameter. All these methods have no means to identify class 2 anomalies.

In the Introduction we already mentioned the relation between our score and lift [5]. As stated there, the difference is that we score transactions rather than rules and we give an algorithm to quickly discover the highest scoring transactions. Our notion of anomaly is also related to the conditional anomalies introduced in [25]. In our running example, $\mathbb{C}$ could be seen as the context that makes a purchase of $\mathbb{P}$ unexpected in their terminology. The difference is that we do not expect the user to define such contexts, they are discovered automatically. Moreover, we use a small set of patterns to discover all the class-2 anomalies rather than probabilistic models on context and other attributes.

To compute the UPC score we need the characteristic patterns of the data. In general, we can use the result of standard frequent pattern mining [2, 20] although this incurs a high computational cost. Instead, we can resort to pattern sampling techniques [12, 4], yet then we have to choose the number of patterns to be sampled. Alternatively, we [22] proposed to mine such pattern sets by the Minimum Description Length principle [11]. That is, they identify the best set of patterns as the set of patterns that together most succinctly describe the data. By definition this set is not redundant and does not contain noise. KRIMP [27] and SLIM [24] are two deterministic algorithms that heuristically optimise this score. Other pattern set mining techniques, especially those that mine patterns characteristic for the data such as [16, 9, 28], are also meaningful choices to be used with UPC.

6 Experiments

In this section we evaluate the power of the UPC score to identify class 2 anomalies. Firstly, we show how we generated synthetic data needed for some of the experiments. Secondly, we provide a baseline comparison where we show that the size of the input set of patterns is of great importance. Next we show the performance of UPC on synthetic data and show its statistical power. Lastly, we show some nice results of class 2 anomalies on a wide variety of real world datasets.

We implemented our algorithms in C++ and generated our synthetic data using Python. Our code is available for research purposes.[1] All experiments were conducted on a 2.6 GHz system with 64 GB of memory.

6.1 Generating Synthetic Data Here we describe how we generated both transaction and categorical synthetic data.

Transaction Data To generate synthetic datasets we first choose the number of transactions $|D|$ and the size of the alphabet $|\Omega|$. We generate a set $\mathcal{P}$ of random patterns, choosing a random cardinality from 3 to 6 items, and a random support in the range of [5-10%]. In addition we generate 2 patterns with a support of 20%, which we call the anomaly generators, and which we add to the data such that they only occur together in a single transaction; that is the anomaly. In addition, each singleton from Ω is added to each transaction similarly with a probability of 10%.

Categorical Data To generate categorical data we take a similar approach. Firstly, we choose the number of transactions $|D|$, the number of attributes $|A|$ and the alphabet size per attributes $|\Omega_i|$. We generate random patterns with the same settings as for transaction data and again first add the anomaly generators to the dataset. We then try to add the other patterns as long as they fit and do not interfere with the anomaly. Then we fill the unspecified attributes for each transaction with random singletons.

[1] http://eda.mmci.uni-saarland.de/upc/

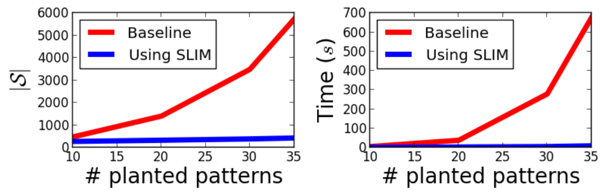

Figure 2: **SLIM pattern set versus closed frequent patterns (baseline).** We computed the UPC scores on 5 000 transactions using both input sets with a minimum support of 1 for SLIM and at 5% for the baseline. Using both input sets $\mathcal{S}$ the anomaly is always ranked first, but for the baseline both $|\mathcal{S}|$ and the runtime quickly explode when we increase the number of planted patterns in the data. The runtimes include the time needed to compute the used input set.

6.2 Baseline Comparison Before investigating the reliability of UPC, we first show its efficiency. To emphasise the necessity for using small pattern sets as input, we compare the use of all closed frequent patterns with a minimum support at 5% to the use of SLIM [24] pattern sets with a minimum support of 1. We generated random transaction data with $|D|$ = 5 000, $|\Omega|$ = 50 and we let $|\mathcal{P}|$ range from 10 to 35 patterns. We then ran our method on both input sets keeping track of the runtimes and the size of the input set $\mathcal{S}$ for which we have to consider all $|\mathcal{S}| \times |\mathcal{S}|$ possible combinations. Both approaches always rank the anomaly highest, therefore further we can focus on the runtime and the number of patterns that were considered. The runtimes include the time needed to compute the pattern sets, which are negligible in light of the exponential time the baseline approach takes in the size of $\mathcal{S}$. Figure 2 shows the results which are averages over 5 runs per setting. For higher minimum support thresholds the baseline approach starts missing important patterns and it cannot identify the anomaly. Other settings for generating synthetic data lead to a similar figure. Since using a SLIM pattern set as input set for UPC we attain similar results compared to the baseline approach, that is we correctly identify the planted anomaly, in the remainder of this paper we always use the SLIM pattern set to compute the UPC score.

6.3 Performance on Synthetic Transaction Data The goal of this experiment is twofold. Firstly, we show that our method is able to identify class 2 anomalies in transaction data. Secondly, we justify the definition of the different

Table 1: **The performance of UpC on transaction data.** The number of generated transactions is represented by $|D|$, the alphabet size by $|\Omega|$, and the number of synthetic patterns by $|\mathcal{P}|$. All experiments were performed 10 times and the average ranks and runtimes (in seconds) are reported.

Generated Data			**UpC**		**OC^3**	
$\|D\|$	$\|\Omega\|$	$\|\mathcal{P}\|$	rank	time (s)	rank	time (s)
5 000	50	100	1	4	2 420	1
5 000	100	100	1	6	2 757	2
5 000	100	200	1	18	2 433	4
10 000	100	100	1	11	5 464	4
20 000	50	100	1	18	8 281	4

Table 2: **The performance of UpC on categorical data.** Each dataset contains 5 000 transactions over $|A|$ attributes, each with an alphabet size of $|\Omega_i|$. $|\mathcal{P}|$ refers to the number of synthetic patterns. All experiments were performed 10 times and average ranks and runtimes (in seconds) are reported.

Generated Data			**UpC**		**CompreX**	
$\|A\|$	$\|\Omega_i\|$	$\|\mathcal{P}\|$	rank	time (s)	rank	time (s)
20	5	100	1	1	3 119	221
50	5	100	1	6	2 028	885
100	5	100	1	29	3 121	5 477
20	10	100	1	1	2 244	429
50	10	200	1	5	2 714	1 978

classes of anomalies as we show that the class 2 anomalies are not identified by the state of the art class 1 anomaly detector, which is OC^3 [23]. We emphasise again that as a result both scores should not be further compared as they are complementary to each other.

We generated random datasets as described in Section 6.1. The results in Table 1 show that UpC always ranks the anomaly highest and that OC^3 does not identify them.

6.4 Performance on Synthetic Categorical Data Knowing that UpC correctly identifies class 2 anomalies for transaction data, here we compared it to the state of the art on categorical data, which is CompreX [3]. Again we note that we only compare these methods to show that class 2 anomalies are different from class 1 anomalies and that these two methods thus should be used complementary to each other.

We generated random datasets as described in Section 6.1 with various settings. The results in Table 2 show that UpC always ranks the anomalous transaction first and CompreX is not able to identify it (gives it a much lower rank).

6.5 Statistical Power Our aim here is to examine the power of the UpC score for identifying class 2 anomalies.

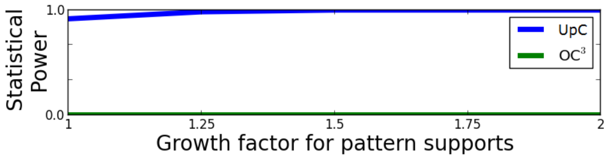

Figure 3: [Higher is better] **Statistical power of UpC.** Whereas OC^3 does not identify any class 2 anomalies, UpC does perfectly with large enough supports. We observe the same behaviour for categorical data comparing UpC and CompreX (not shown in this plot). The growth factor on the x-axis describes the increase of the pattern supports.

For this purpose, we perform statistical tests using synthetic data. To this end, the null hypothesis is that the data contains no class 2 anomalies. To determine the cutoff for testing the null hypothesis, we first generate 100 transaction datasets without the single co-occurrence between the 2 anomaly generators, whereafter we generate another 100 datasets with this co-occurrence included. For all datasets we choose $|D|$ = 5 000, $|\Omega|$ = 25 items and $|\mathcal{P}|$ = 100. Next, we report the highest UpC score for all 100 datasets without anomaly. Subsequently, we set the cutoff according to the significance level $\alpha = 0.05$. The power of the UpC score is the proportion of the highest scores from the 100 datasets with anomaly that exceed the cutoff. Note that, we only look at the highest score for each dataset as we know that this must be the anomaly for the datasets containing it. We show the results in Figure 3 while varying the range from which we randomly choose the supports for the patterns in $\mathcal{P}$ from [4-8%] to [8-16%] and the support for the anomaly generators from 16% to 32%. In Figure 3 we label these linearly growing supports with their growth factor from 1 to 2. With other settings to generate the data we observe the same trend. Again, only to emphasise that methods to identify class 1 anomalies are not suitable to discover class 2 anomalies, in Figure 3 we also plotted the statistical power of OC^3 regarding class 2 anomalies. As CompreX is not applicable to transaction data we performed a similar experiment on categorical data. This resulted in a similar plot with UpC at the top and CompreX at the bottom.

In Figure 4 we show the distribution of the highest scores for both the datasets with and without an anomaly and with pattern supports in range [7-14%] and an anomaly generator support at 28%. We can see a clear distinction between the scores for 'normal' and anomalous transactions.

6.6 Real World Data To show that class 2 anomalies actually exist, are not identified by the state of the art in anomaly detection, and can give much insight we performed multiple experiments on real world datasets from various domains. We used the *Adult* and *Zoo* datasets from the UCI repository, together with the *Mammals* [18] and *ICDM*

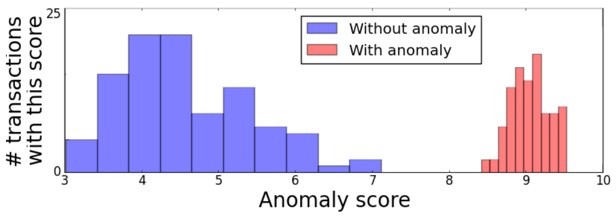

Figure 4: **Significance of UPC scores.** The plot shows a clear separation between the highest UPC scores for random synthetic datasets with and without class 2 anomalies.

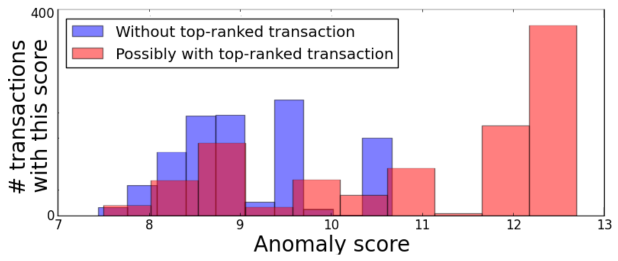

Figure 5: **Significance test on *Adult* dataset.** This plot shows the difference in the distribution of highest scores for bootstrap samples without (blue) and possibly with (red) the highest ranked transaction from the original dataset.

Abstracts [8] datasets. Note that, we cannot provide the reader with the accuracy of our method because anomalies can manifest themselves in many different forms, e.g. unexpected behaviour or mistakes in the data, and no ground truth for these datasets is available. However, we aim to give insightful examples instead.

Adult The *Adult* dataset contains information about 48 842 people such as age, education, and marital-status and is used to predict whether someone's income exceeds $50K a year.

We computed a ranking based on the UPC score and found some interesting anomalies. The top-ranked transaction contains the very unexpected co-occurrence of someone for which the attribute sex is female yet for whom the relationship status has the value of husband. The following 3 anomalies are persons with a similar situation but with the patterns reversed. That is, the dataset contains 3 persons whose sex is male and whose relationship is wife. The OC^3 rankings of these first 4 people are 115, 148, 89 and 4 090, respectively. These examples show that class-2 anomalies indeed exist in real datasets, and that UPC is effective at identifying these. Clearly, each of these four anomalies is an error, but that does not make them less of an outlier. In fact, one of the goals of outlier detection is to find errors.

To get an idea of the significance of the results we performed the significance test as described in Section 4.1. Figure 5 shows the difference in the distribution of highest scores for bootstrap samples from data without (blue), resp. from data including (red) the top-ranked transaction from the original dataset. Figure 5 gives insight in how much this transaction deviates from the norm, as the difference between the two distributions can only be caused by this transaction.

Zoo The *Zoo* data contains 17 attributes describing 101 different animals. We performed the bootstrap method described in Section 4.2 to determine which transactions are worth investigating. To this end, we generated 1 000 bootstrap samples for which we computed the anomaly scores for all transactions. In Figure 6 we show the distribution of all these scores with a histogram. Further, we show the anomaly scores of the 5 highest ranked transactions in the original dataset together with the FNR corresponding to a θ equal to their score. That is, with an FNR not higher than 8% only the top-ranked transaction scores above θ. This transaction contains information about the platypus (duck bill) and from our results we found that the co-occurrence causing this high score is that the platypus is the only oviparous mammal in the dataset. Further, we see that the chance that the second ranked animal belongs to the positive class is not more than 11%. This is the scorpion for which UPC found that it is the only animal without teeth that is not oviparous.

Clearly, both these anomalies are well-known as somewhat weird species and, so, these finds may not seem that interesting. However, the algorithm does not know much biology and yet it finds both anomalous species as well as the explanation for why they are seen as somewhat weird.

ICDM Abstracts Next, we ran our algorithm on a dataset comprising the abstracts from the ICDM conference, after stemming, and removing stopwords.

For this data we expect co-occurrences of terms used in different research fields to rank highly. The abstract with the highest UPC rank contains both the frequently used words 'pattern mining' and 'training'. Given that (frequent)

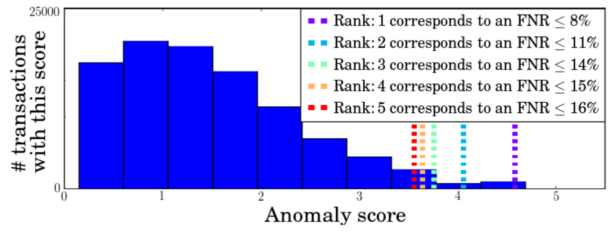

Figure 6: **Anomalies in the *Zoo* dataset.** The histogram shows the estimated distribution of anomaly scores. The vertical lines show the scores of the top-5 ranked anomalies in the original dataset, together with the false-negative rates corresponding to the decision threshold for their score.

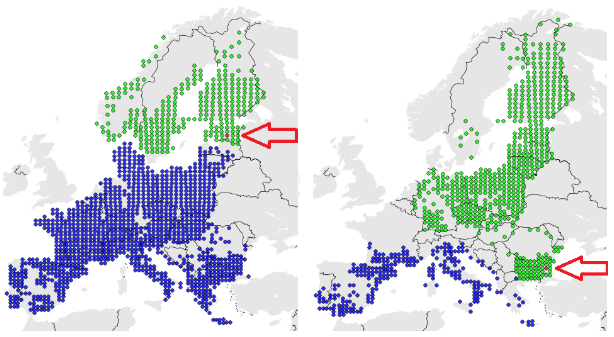

Figure 7: **UpC in action; top-ranked anomalies on the *Mammals* dataset.** The explanations for these highly ranked area's are as follows. On the left we see that the habitat of the beech marten (blue) only intersects with that of the moose, European hedgehog and mountain hare (green) at the (red) area pointed to by the arrow. On the right we see the habitat of the Etruscan shrew (blue) only intersects with that of the raccoon dog (green) at the (red) area pointed to by the arrow.

pattern mining is unsupervised while methods like "train and test" are usually applied in a supervised setting, their combination is genuinely surprising. From the corresponding abstract it transpires that the term 'training' is used to refer to physical exercise rather than that of an algorithm. Hence, the discovered anomaly points to an unusual application rather than to an unexpected combination of techniques.

Other highly ranked abstracts show similar unexpected co-occurrences, for example 'learning' on one side and 'frequent pattern mining' on the other or 'frequent pattern mining' and 'compare', which suggest that exploratory algorithms are difficult to compare.

Mammals The *Mammals* dataset consists of presence/absence records of 121 European mammals within 2 183 geographical areas of 50×50 kilometres. In this dataset an anomaly constitutes two large territories of (groups of) animals which only overlap in a small region.

Figure 7 shows two top-ranked area's (in red and pointed to by arrows) and readily explains why these are anomalous. For each of these two area's two groups of animals share this territory where the rest of their territory is completely separated. On the left in Figure 7 we see that the large habitat of the beech marten intersects with that of the moose, the European hedgehog and the mountain hare only in this single area. On the right in Figure 7 we see a similar phenomenon for the Etruscan shrew on one side and the raccoon dog on the other. The ranks of these two areas using OC^3 are 591 and 294 out of the 2 183, respectively. There are also top-ranked areas that are explained by two groups of animals which habitat intersects in multiple areas (of course including the area that has received this score).

7 Discussion

The experiments show that although the state of the art in anomaly detection is not able to identify the newly defined class 2 anomalies, we can identify them using our new UpC score. We demonstrated that a naive baseline approach using closed frequent items as input set quickly becomes infeasible when the number of patterns present in the data grows. Using a Slim pattern set to compute our UpC score, however, we attain similar results in a fraction of the time. We showed the statistical power of our method which scores transactions containing planted class 2 anomalies significantly higher than 'normal' transactions. Moreover, both on transaction and categorical synthetic data we showed that UpC always ranked the planted anomaly at the top.

From our experiments on real world datasets we find that the class 2 anomalies do actually exist and can provide useful insights. That is, because next to identifying interesting transactions the UpC score also readily explains which co-occurrence of patterns is responsible for the transaction's anomaly score. For example, in the *Adult* dataset we found a very unexpected individual who is described as being a female husband. Further we showed how a UpC ranking can be used to study the significance of identified anomalies using a bootstrap approach. For example, in the *Zoo* dataset we found that the platypus, which is special because its the only oviparous mammal, has a less than 8% chance on being 'normal' given the data. Each of these class 2 anomalies were not identified, i.e. ranked low, using OC^3 or CompreX.

8 Conclusion

The recognition of anomalies provides useful application-specific insights [1]. More specifically, the field of anomaly detection focusses on the identification of data that significantly differ from the rest of the dataset. There are many reasons for anomalies – ranging from errors to outliers to simply highly unexpected data points – however, whatever the reason, the anomalies should be brought to the attention of the data miner.

There are also many ways in which a data point (or a subset of the data) can differ from the rest of the data set. That is, there are many types of anomalies [1]. In this paper we introduced a new class of anomalies which consist of unexpected co-occurrences of patterns. In a world where the vast part of the population drinks either soft drink $\mathbb{C}$ or soft drink $\mathbb{P}$ but not both, it is surprising to find someone who apparently drinks both.

We introduced the UpC score which intuitively scores a transactions based on its most unexpected co-occurrence of patterns. Next we introduced an algorithm that discovers and ranks transactions with a high UpC score. Moreover, it does so efficiently by relying on a small set of characteristic patterns [24] rather than on the full set of low support patterns (which would make an algorithmic approach intractable quickly). Finally we introduced a statistical test to decide whether or not an anomaly is significantly anomalous.

We tested our methods firstly on synthetic data. These experiments show that we are able to reliably discover the anomalous patterns we planted in a wide range of different settings and circumstances. Moreover, these experiments show that the anomalies identified by state of the art methods for anomaly detection are of a different class and that these methods are not able to identify planted anomalies.

That we can discover a new class of anomalies does not make them into an interesting class of anomalies. To illustrate that our anomalies indeed provide interesting and useful information we also did experiments on four real world data sets, viz., *Adult*, *Zoo*, *ICDM Abstracts*, and *Mammals*. In all cases the identified transactions with a high UPC score were truly anomalous. None of these examples were discovered using the state of the art anomaly detection algorithms

In some cases – such as on the *Adult* data set in which we discovered a female husband – the identified anomalies are very likely errors. In other cases – such as on the *Zoo* data set where we discovered the platypus – the discovered anomalies are not an error but simply a highly surprising combination of patterns: whereas laying eggs is quite normal, so is being a mammal, but being an egg-laying mammal is truly special.

Whatever the reason, these anomalies provide useful information to the analyst; whether they point to errors that probably should be corrected, such as female husbands, or to genuinely new information, such as the existence of egg-laying mammals.

While we tested our approach against state of the art, this does not mean that we claim that our methods are *better*. Rather, the experiments were performed to show that our methods are *complementary* to those methods. When looking for anomalies, one should not use one method, but many.

Acknowledgments

Roel Bertens and Arno Siebes are supported by the Dutch national program COMMIT and Jilles Vreeken by the Cluster of Excellence "Multimodal Computing and Interaction" within the Excellence Initiative of the German Federal Government.

References

[1] C. C. Aggarwal, editor. *Outlier Analysis*. Springer, 2013.

[2] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. Fast discovery of association rules. In *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI/MIT Press, 1996.

[3] L. Akoglu, H. Tong, J. Vreeken, and C. Faloutsos. Fast and reliable anomaly detection in categorical data. In *CIKM*, pages 415–424, 2012.

[4] M. Boley, C. Lucchese, D. Paurat, and T. Gärtner. Direct local pattern sampling by efficient two-step random procedures. In *SIGKDD*, pages 582–590, 2011.

[5] S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: Generalizing association rules to correlations. In *SIGMOD*, pages 265–276. ACM, 1997.

[6] T. Calders and B. Goethals. Non-derivable itemset mining. *DMKD*, 14(1):171–206, 2007.

[7] A. C. Cameron, J. B. Gelbach, and D. L. Miller. Bootstrap-based improvements for inference with clustered errors. *Rev. Econ. Stat.*, 90:414–427, 2008.

[8] T. De Bie. Maximum entropy models and subjective interestingness: an application to tiles in binary databases. *DMKD*, 23(3):407–446, 2011.

[9] F. Geerts, B. Goethals, and T. Mielikäinen. Tiling databases. In *DS*, pages 278–289, 2004.

[10] G. Grimmett and D. Stirzaker. *Probability and Random Processes*. Oxford University Press, 2001.

[11] P. Grünwald. *The Minimum Description Length Principle*. MIT Press, 2007.

[12] M. A. Hasan and M. Zaki. Musk: Uniform sampling of k maximal patterns. In *SDM*, pages 650–661, 2009.

[13] Z. He, X. Xu, J. Z. Huang, and S. Deng. Fp-outlier: Frequent pattern based outlier detection. *ComSIS*, 2(1):103–118, 2005.

[14] Y. S. Koh and S. D. Ravana. Unsupervised rare pattern mining: A survey. *TKDD*, (4), 2016.

[15] N. Lavrač, P. Flach, and B. Zupan. Rule evaluation measures: A unifying view. In *ILP*, pages 174–185, 1999.

[16] M. Mampaey, J. Vreeken, and N. Tatti. Summarizing data succinctly with the most informative itemsets. *TKDD*, 6:1–44, 2012.

[17] M. Markou and S. Singh. Novelty detection: a review. *Signal processing*, 83(12), 2003.

[18] T. Mitchell-Jones. Societas europaea mammalogica. http://www.european-mammals.org, 1999.

[19] K. Narita and H. Kitagawa. Outlier detection for transaction databases using association rules. In *WAIM*, pages 373–380, 2008.

[20] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *ICDT*, pages 398–416, 1999.

[21] G. Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. In G. Piatetsky-Shapiro and W. Frawley, editors, *Knowledge Discovery in Databases*. AAAI/MIT Press, 1991.

[22] A. Siebes, J. Vreeken, and M. van Leeuwen. Item sets that compress. In *SDM*, pages 393–404, 2006.

[23] K. Smets and J. Vreeken. The odd one out: Identifying and characterising anomalies. In *SDM*, pages 804–815, 2011.

[24] K. Smets and J. Vreeken. SLIM: Directly mining descriptive patterns. In *SDM*, pages 236–247, 2012.

[25] X. Song, M. Wu, C. Jermaine, and S. Ranka. Conditional anomaly detection. *TKDE*, 19(5):631–645, 2007.

[26] P. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. In *KDD*, pages 32–41, 2002.

[27] J. Vreeken, M. van Leeuwen, and A. Siebes. KRIMP: Mining itemsets that compress. *DMKD*, 23(1):169–214, 2011.

[28] C. Wang and S. Parthasarathy. Summarizing itemset patterns using probabilistic models. In *SIGKDD*, pages 730–735, 2006.