Efficient Pattern Mining of Uncertain Data with Sampling

Toon Calders[1], Calin Garboni[2], and Bart Goethals[2]

[1] TU Eindhoven, The Netherlands
t.calders@tue.nl
[2] University of Antwerp, Belgium
{calin.garboni,bart.goethals}@ua.ac.be

Abstract. Mining frequent itemsets from transactional datasets is a well known problem with good algorithmic solutions. In the case of uncertain data, however, several new techniques have been proposed. Unfortunately, these proposals often suffer when a lot of items occur with many different probabilities. Here we propose an approach based on sampling by instantiating "possible worlds" of the uncertain data, on which we subsequently run optimized frequent itemset mining algorithms. As such we gain efficiency at a surprisingly low loss in accuracy. These is confirmed by a statistical and an empirical evaluation on real and synthetic data.

1 Introduction

In frequent itemset mining, the transaction dataset is typically represented as a binary matrix where each line represents a transaction and every column corresponds to an item. An element M_{ij} represents the presence or the absence of the item j in transaction i by the value 1 or 0 respectively. For this basic traditional model, where an item is either present or absent in a transaction, many algorithms have been proposed for mining frequent itemsets; i.e., sets of columns of M that have all ones in at least a given number of transactions (see e.g. [5] for an overview on frequent itemset mining).

In many applications, however, an item is not present or absent in a transaction, but rather an existence probability of being in the transaction is given. This is the case, for example, for data collected from experimental measurements or from noisy sensors. Mining frequent patterns from this kind of data is more difficult than mining from traditional transaction datasets. After all, computing the support of an itemset now has to rely on the existence probabilities of the items, which leads to an *expected support* as introduced by Chui et al. [4].

If the binary matrix is transformed into a probabilistic matrix, where each element takes values in the interval $[0, 1]$, we have the so called uncertain data model. Under the assumption of statistical independence of the items in all transactions in the dataset, the support of an itemset in this model, as defined by Chui et al. [4], is based on the possible world interpretation of uncertain data. Basically, for every item x and every transaction t there exist two sets of

possible worlds, one with the worlds in which x is present in t and one with the worlds where x is not present in t. The probability of the first set of worlds is given by the existence probability of x in t ($P(x,t)$) and the probability of the second set of worlds by $1-P(x,t)$. The probability of a single world is then obtained by multiplying the probabilities for all its individual items; i.e., $P(W) = \sum_{t \in W} \prod_{x \in t} P(x,t) \prod_{x \notin t} (1 - P(x,t))$. The expected support of an itemset can be obtained by summing the support of that itemset over all possible worlds, while taking into consideration the probability of each world. There exist $2^{|D| \times |I|}$ worlds, where $|D|$ is the total number of transactions in the probabilistic dataset and $|I|$ is the total number of items. This rather complicated formula can be reduced to:

$$expSup(X) = \sum_{t \in D} \prod_{x \in X} P(x,t)$$

Every transaction thus supports an itemset with the probability given by the product of existence probabilities of all items in the itemset and in that transaction. The expected support of an itemset over the entire dataset is the sum of the existence probabilities of that itemset in every transaction of the dataset.

In the remainder of this paper we revisit the related work, then we present our proposed method based on sampling, followed by theoretical and empirical analysis of the quality of the results.

2 Related Work

The efficient data structures and techniques used in frequent itemset mining such as TID-lists [2], FP-tree, which adopts a prefix tree structure as used in FP-growth [6], and the hyper-linked array based structure as used in H-mine [8] can no longer be used as such directly on the uncertain data. Therefore, recent work on frequent itemset mining in uncertain data that inherits the breadth-first and depth-first approaches from traditional frequent itemset mining adapts the data structures to the probabilistic model.

U-Apriori [4] is based on a level wise algorithm and represents a baseline algorithm for mining frequent itemsets from uncertain datasets. Because of the generate and test strategy, level by level, the method does not scale well.

UCP-Apriori [3] is based on the decremental pruning technique which consists in maintaining an upper bound of the support and decrementing it while scanning the dataset. The itemset is pruned as soon as its most optimistic value falls below the threshold. This approach represents the state of the art for mining frequent patterns from uncertain data with a generate-and-prune strategy.

UF-growth [7] extends the FP-Growth algorithm [6]. It is based on a UF-tree data structure (similar to FP-tree). The difference with the construction of a FP-tree is that a transaction is merged with a child only if the same item and the same expected support exist in the transaction and in the child node, leading to a far lower compression ratio as in the original FP-tree. The improvements consist in discretization of the expected support to avoid the huge number of different

values and in adapting the idea of co-occurrence frequent itemset tree (COFI-tree). The UF-trees are built only for the first two levels. It then enumerates the frequent patterns by traversing the tree and decreasing the occurrence counts.

Aggarwal et al. [1] extended several existing classical frequent itemset mining algorithms for deterministic data sets, and compared their relative performance in terms of efficiency and memory usage. The *UH-mine* algorithm, proposed in their paper, provides the best trade-offs. The algorithm is based on the pattern growth paradigm. The main difference with UF-growth is the data structure used which is a hyperlinked array.

The limitations of these existing methods are the ones inherited from the original methods. The size of the data for the level-wise generate-and-test techniques affects their scalability and the pattern-growth techniques require a lot of memory for accommodating the dataset in the data structures, such as the FP-tree, especially when the transactions do not share many items. In the case of uncertain data, not only the items have to be shared for a better compression but also the existence probabilities, which is often not the case.

3 Sampling the Uncertain Dataset

The first method we propose, called *Concatenating the Samples*, takes the uncertain dataset and samples according to the given existential probabilities. For every transaction t and every item i in transaction t we generate an independent random number $0 \leq r \leq 1$ (coin flip) and we compare it with the probability p associated with the item i. If $p \geq r$ then item i will appear in the currently sampled transaction. For every transaction in the uncertain dataset we repeat the step above n times, for a given n. The result is a dataset which can be mined with any traditional frequent itemset mining algorithm. To obtain the estimated support of an itemset in the uncertain dataset, its support in the sampled dataset still needs to be divided by n.

The difficulty of this method resides in the fact that we physically instantiate and store the sampled "certain" dataset which can be up to n times larger than the original uncertain dataset. Fortunately, for most efficient itemset mining algorithms, we do not actually have to materialize this samples database. After all, most efficient techniques read the database from disk only once, after which their advanced data structures contain the database in the main memory. Therefore, the sample can be generated immediately in memory when the database is being read from disk for the first time. We call this method *Inline Sampling*.

To this end, we made minor modifications of the frequent itemset mining algorithms. We will briefly describe **U-Eclat** and **UFP-growth**, the modified versions of the **ECLAT** and **FP-growth** algorithms.

U-Eclat is an adaptation of the **ECLAT** algorithm [11] with an improvement based on diffsets as described in [10]. In only one scan of the dataset the relevant items are stored into memory together with the list of transactions where the

items appear, called tid-list. The candidates are then generated using a depth-first search strategy and their support is computed by intersecting the tid-list of the subsets. The only adaptation for **U-Eclat** consists in reading the uncertain transactions and instantiating them as described above. More specifically, given the number of iterations n, for every transaction t and every item i in transaction t we generate n independent random numbers $r_1, \ldots, r_n$ betweeen 0 and 1 and we compare them with the probability p associated with the item i. If $p \geq r_j$, for $1 \leq j \leq n$, then $n \cdot t + j$ will appear in the tid-list of item i. From there on, the standard Eclat algorithm is being executed.

UFP-growth extends the initial **FP-growth** algorithm [6]. The FP-tree construction needs two scans of the dataset. The first scan collects the frequent items and their support and in the second scan every transaction is accommodated in the FP-tree structure. The frequent itemsets are then generated recursively from the FP-tree. In order to adapt this algorithm to our method, the first scan computes the expected support of every itemset exactly by computing their support as the sum of existential probabilities in every transaction where it occurs. In the second scan, every transaction is instantiated n times, according to the existential probability of the items in the transaction and then it is inserted in FP-tree structure. The algorithm then extracts the frequent itemsets the same way as the **FP-growth** algorithm.

4 Statistical Bounds on the Quality of the Approximation

As before, D denotes the set of transactions, and I the set of items. $P(x,t)$ denotes the probability assigned to item x by transaction t. We extend this notation to itemsets X; i.e., $P(X,t)$ will denote $\prod_{x \in X} P(x,t)$. Our whole analysis will be based on the numbers $P(X,t)$ only and hence, will not depend on the assumption of independence between the items. Notice that this implies that our sampling-based method, in contrast to the other existing proposals, could also be applied when a more involved probabilistic model is assumed. We first start our analysis for a single itemset X and will extend it later on for the complete collection of itemsets.

Suppose that, for every transaction $t \in D$, we sample n deterministic versions of this tuple, $t_1, \ldots, t_n$. Let X_t^i be the stochastic variable denoting if $X \subseteq t_i$; i.e., $X_t^i = 1$ if $X \subseteq t_i$, and $X_t^i = 0$ otherwise. Notice that the variables X_t^i are statistically independent as they are sampled using independent coin flips. X_t^i follows a Bernoulli distribution with mean $P(X,t)$. It is easy to see that the stochastic variable $X_t = \sum_{i=1}^{n} X_t^i$ follows a binomial distribution with mean $nP(X,t)$ and variance $nP(X,t)(1-P(X,t))$. Consider now the sum: $S(X) := \frac{\sum_{t \in D} X_t}{n}$ The expected value and variance of this sum are as follows:

$$E[S] = expSup(X)$$
$$V[S] = V\left[\frac{\sum_{t \in D} X_t}{n}\right] = \frac{\sum_{t \in D} V[X_t]}{n^2} = \frac{\sum_{t \in D} nP(X,t)(1-P(X,t))}{n^2} \leq \frac{|D|}{4n}.$$

Hence, not surprisingly, the sum S we use to approximate the expected support is an unbiased estimator with a variance that decreases linearly with n. For the relative version, $rS = \frac{S}{|D|}$, we get $V[rS] = V\left[\frac{S}{|D|}\right] = \frac{V[S]}{|D|^2} \leq \frac{1}{4n|D|}$.

We now apply Hoeffding's inequality. This inequality is as follows: given independent (but not necessarily identically distributed) stochastic variables $X_1, \ldots, X_m$ such that for all $i = 1 \ldots n$, $P(a_i \leq X_i - E[X_i] \leq b_i) = 1$, then

$$p\left[\left|\sum_i X_i - E\left[\sum_i X_i\right]\right| \geq m\epsilon\right] \leq 2\exp\left(-\frac{2m^2\epsilon^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right).$$

In our case, for all X_t^i, $X_t^i - E(X_t^i)$ is in the interval $[-1, 1]$, and hence we get:

$$\begin{aligned} p\left[\left|\sum_{t\in D}\sum_{i=1}^{n} X_t^i - E\left[\sum_{t\in D}\sum_{i=1}^{n} X_t^i\right]\right| \geq n|D|\epsilon\right] &\leq 2\exp\left(-\frac{2(n|D|)^2\epsilon^2}{\sum_{i=1}^{n|D|} 2^2}\right) \\ &= 2\exp\left(-\frac{n|D|\epsilon^2}{2}\right). \end{aligned}$$

If we now rewrite in function of $rS(X)$ and $rsupp(X) := \frac{expSup(X)}{|D|}$, we get:

$$p[|rS(X) - rsupp(X)| \geq \epsilon] \leq 2\exp\left(-\frac{n|D|\epsilon^2}{2}\right).$$

Hence, for given $\epsilon, \delta > 0$, we have: If $\delta \geq 2\exp\left(-\frac{n|D|\epsilon^2}{2}\right)$, i.e., $n \geq -\frac{2\ln(\delta/2)}{|D|\epsilon^2}$, then $p[|rS(X) - rsupp(X)| \leq \epsilon] \geq 1 - \delta$.

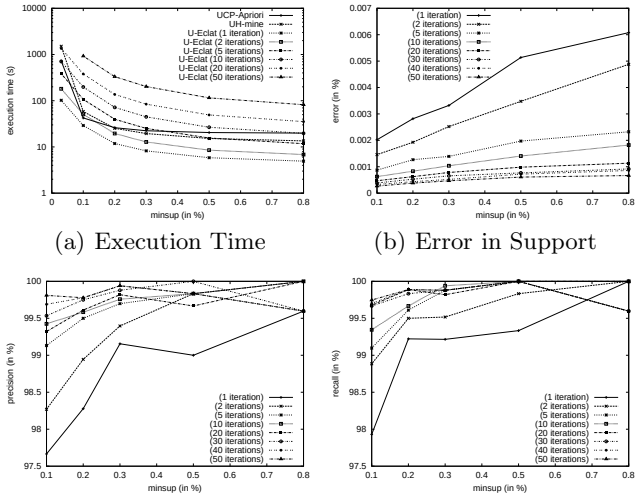
The significance of this result can best be illustrated by an example. Suppose D contains 100 000 probabilistic transactions and X is an itemset. In order to guarantee that the support of X is approximated with 99% probability with less than 1% error, we need to have $n \geq -\frac{2\ln(0.01/2)}{100\,000(0.01)^2} \approx 1$. Hence, we need approximately 1 sample per transaction in D to achieve this result. Furthermore, suppose that we have a collection of 1 000 000 frequent itemsets. In order to guarantee that all these itemsets have less than 1% error with 99% probability, we need to have (using the union rule) $n \geq -\frac{2\ln(1/200\,000\,000)}{100\,000(0.01)^2} \approx 3.8$; i.e., less than 4 samples per transaction.

As a side note, even tighter bounds can be gotten by approximating the distribution of rS with a normal distribution, using a weaker form of the Central Limit Theorem, called *Lyapunov's central limit theorem*. That is, $\frac{S - supp(X)}{nV[S]}$ converges in probability to $N(0, 1)$.

5 Experiments

The experiments were conducted on a GNU/Linux machine with a 2.1GHz CPU with 2 Gb of main memory. We used the datasets and the executables for comparison from [1]. Kosarak contains anonymized click-stream data. It is a very

sparse dataset, with a density of less than 0.02%, about 1 million transactions, 42170 distinct items and an average of 8 item per transaction. The dataset T40I10D100K was generated using the IBM synthetic data generator, having 100K transactions, 942 distinct items and a density of 4.2%. The original datasets were transformed by Aggarwal et al. [1] into uncertain datasets by assigning to every item in every transaction existential probabilities according to the normal distribution $N(\mu, \sigma^2)$, where μ and σ were randomly and independently generated with values between $[0.87, 0.99]$ and $[1/21, 1/12]$ respectively.

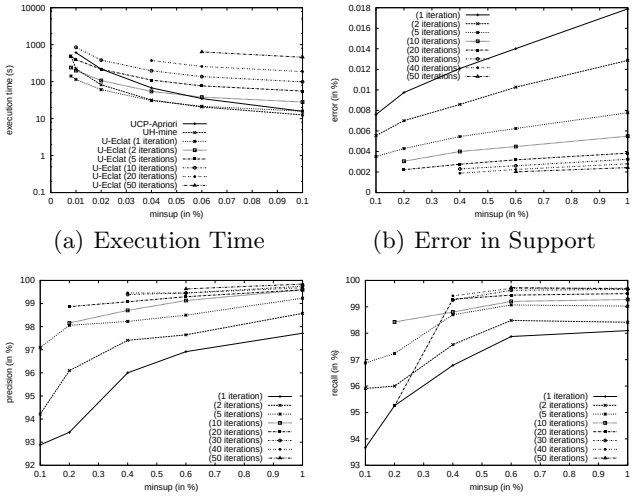


(a) Execution Time (b) Error in Support

(c) Precision (d) Recall

Fig. 1. kosarak Dataset

For different values of the minimum support, we ran our implementations of **U-Eclat** and **UFP-growth**. The number of times we instantiate the uncertain dataset varies between 1 and 50. The higher the number of instantiations, the better the accuracy of the results becomes, at the cost of an increase in execution time. We also experimented with the original **ECLAT** and **FP-growth** algorithms after materializing the sampled datasets. Obviously the size of these datasets become very large for multiple iterations, and thus, those experiments always resulted in a decrease in performance as compared to their inline versions. Experimentally we show that for relatively low number of instantiations we reach highly accurate results. The gain in time motivates the use of our method which outperforms in execution time the existing state of the art methods mentioned in [1]. For every dataset, we plot the execution times we obtained for different values of the minimum support and for some different numbers of iterations. It turns out that **U-Eclat** always outperformed **UFP-growth**. In many cases, the FP-tree simply became too large to handle [5]. In the experiments, for clarity, we thus only show the results for **U-Eclat**. For a fair comparison we also only show the best performing implementations of the algorithms mentioned in [1],



(a) Execution Time (b) Error in Support

(c) Precision (d) Recall

Fig. 2. t40 Dataset

being UCP-Apriori and UH-mine. The execution times are depicted in Figures 1(a) and 2(a).

For low support threshold, our **U-Eclat** outperforms UCP-Aprori and UH-mine for up to 5 sampling iterations of the dataset. Note that more efficient frequent set mining algorithms as can be found in the FIMI repository will perform even better, also for a higher number of iterations. As our theoretical results already indicated that 2 iterations already result in a very accurate approximation of the expected supports of all itemsets. This also shows in practice.

To this end, we compare the collections of frequent patterns and their support obtained using the exact method and our sampling method. First, the collection of frequent itemsets is generated using UCP-Aprori [1]. Based on this, we evaluate the errors in support computed with the sampling method.

	kosarak					
	precision			recall		
Iter	min	avg	max	min	avg	max
1	97.67	98.95	100	97.93	99.28	100
2	98.27	99.41	100	98.88	99.62	100
5	99.13	99.63	100	99.10	99.70	100
10	99.43	99.77	100	99.34	99.74	100
20	99.32	99.74	100	99.60	99.83	100
30	99.53	99.79	100	99.60	99.83	100
40	99.69	99.87	100	99.60	99.84	100
50	99.60	99.83	100	99.75	99.92	100

	t40					
	precision			recall		
Iter	min	avg	max	min	avg	max
1	92.88	96.16	100	93.66	96.95	100
2	94.22	97.25	99.54	95.91	97.73	100
5	97.10	98.52	100	96.88	98.48	100
10	98.16	99.12	100	98.43	99.14	100
20	98.87	99.37	100	95.25	98.70	100
30	99.39	99.65	100	99.26	99.64	100
40	99.46	99.65	100	99.42	99.71	100
50	99.63	99.82	100	99.68	99.80	100

Fig. 3. Summary of Precision and Recall

In terms of support error, we compute the average of the absolute difference between the support of itemsets found by both methods. The error is depicted in Figures 1(b) and 2(b). It can be seen that, as expected and predicted by the statistical evaluation, the higher the number of iterations grows, the lower the error becomes. But even for relatively low number (5 or 10 iterations), the average error in support estimation drops below 1%.

For itemsets having the support close to the minimum support threshold, small variations of support can introduce false positives when the real support is overestimated or false negatives when the real support is underestimated. To evaluate the impact of this, we report Precision and Recall of our method w.r.t. the true collection in terms of patterns found as frequent. We plot in Figures 1(c), 1(d), 2(c) and 2(d) the values of precision and recall for different number of iterations. A summary of these values is reported in Figure 3 as the overall minimum, average and maximum for each dataset and different numbers of iterations. The values confirm the quality of the approximation.

Acknowledgements. We thank to the authors of Aggarwal et al. [1] for making available the executables and datasets. This research was partially founded by FWO project "Foundations for Inductive Databases".

References

1. Charu C. Aggarwal, Yan Li, Jianyong Wang, and Jing Wang. Frequent pattern mining with uncertain data. In *Proc. of KDD '09*, pages 29–38, New York, NY, USA, 2009. ACM.
2. Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proc. of VLDB '94*, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.
3. Chun K. Chui and Ben Kao. A decremental approach for mining frequent itemsets from uncertain data. In Washio et al. [9], pages 64–75.
4. Chun K. Chui, Ben Kao, and Edward Hung. Mining frequent itemsets from uncertain data. In Zhi-Hua Zhou, Hang Li, and Qiang Yang, editors, *PAKDD*, volume 4426 of *Lecture Notes in Computer Science*, pages 47–58. Springer, 2007.
5. Bart Goethals. Frequent set mining. In *In The Data Mining and Knowledge Discovery Handbook*, chapter 17, pages 377–397. Springer, 2005.
6. Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Min. Knowl. Discov.*, 8(1):53–87, 2004.
7. Carson Kai-Sang Leung, Mark Anthony F. Mateo, and Dale A. Brajczuk. A tree-based approach for frequent pattern mining from uncertain data. In Washio et al. [9], pages 653–661.
8. Jian Pei, Jiawei Han, Hongjun Lu, Shojiro Nishio, Shiwei Tang, and Dongqing Yang. H-mine: Hyper-structure mining of frequent patterns in large databases. In *Proc. of ICDM '01*, pages 441–448, Washington, DC, USA, 2001. IEEE Computer Society.
9. Takashi Washio, Einoshin Suzuki, Kai Ming Ting, and Akihiro Inokuchi, editors. *Advances in Knowledge Discovery and Data Mining, 12th Pacific-Asia Conference, PAKDD 2008, Osaka, Japan, May 20-23, 2008 Proceedings*, volume 5012 of *Lecture Notes in Computer Science*. Springer, 2008.
10. Mohammed J. Zaki and Karam Gouda. Fast vertical mining using diffsets. In Lise Getoor, Ted E. Senator, Pedro Domingos, and Christos Faloutsos, editors, *Proc. of KDD '03*, pages 326–335. ACM, 2003.
11. Mohammed J. Zaki, Srinivasan Parthasarathy, Mitsunori Ogihara, and Wei Li. New algorithms for fast discovery of association rules. In *Proc. of KDD '97*, pages 283–286, 1997.