

Summarizing Itemset Patterns Using Probabilistic Models*

Chao Wang and Srinivasan Parthasarathy
Dept. of Computer Science and Engineering, The Ohio State University
Columbus, OH, USA
srini@cse.ohio-state.edu

ABSTRACT

In this paper, we propose a novel probabilistic approach to summarize frequent itemset patterns. Such techniques are useful for summarization, post-processing, and end-user interpretation, particularly for problems where the resulting set of patterns are huge. In our approach items in the dataset are modeled as random variables. We then construct a Markov Random Fields (MRF) on these variables based on frequent itemsets and their occurrence statistics. The summarization proceeds in a level-wise iterative fashion. Occurrence statistics of itemsets at the lowest level are used to construct an initial MRF. Statistics of itemsets at the next level can then be inferred from the model. We use those patterns whose occurrence can not be accurately inferred from the model to augment the model in an iterative manner, repeating the procedure until all frequent itemsets can be modeled. The resulting MRF model affords a concise and useful representation of the original collection of itemsets. Extensive empirical study on real datasets show that the new approach can effectively summarize a large number of itemsets and typically significantly outperforms extant approaches.

Categories and Subject Descriptions: H.2.8 [Database Applications]: Data Mining

General Terms: Algorithms

Keywords: Itemset pattern summarization, probabilistic graphical model, Markov Random Field

1. INTRODUCTION

The problem of mining frequent patterns, particularly associations among items in transactional datasets, is an important one with many applications. Efficient algorithms to compute frequent patterns and rules associated with these patterns exist [2, 19, 10, 18, 8, 7]. However, often times in many real-world problems the number of frequent patterns

mined for various parametric settings is extremely large leaving the end-user swamped when it comes to interpreting the results. As a result researchers have turned to various strategies to summarize the patterns the user is asked to examine. This has led to researchers investigating the use of *closed itemsets* [14], *maximal itemsets* [9], *non-derivable itemsets* [4], and more recently *profile patterns* [17], for this purpose.

Closed itemsets and non-derivable itemsets are lossless forms of compressing frequent itemsets, i.e. the full list of frequent itemsets and associated frequency counts (used for computing association rules) can be exactly derived from the compressed representation. Maximal itemsets allow greater compression when compared with closed patterns, but the representation is lossy – the list of frequent itemsets can be exactly computed but the exact frequency counts associated with each frequent itemset cannot be determined. It is important to note that the number of closed, maximal or non-derivable itemsets can still be very large, thus further compression is necessary. Top- k patterns [11], computes the k most frequent closed itemsets and presents it as the approximate summary to the end-user. Choosing an appropriate k for a given domain is usually not easy and there are no theoretical guarantees on the level of approximation for a given k . Afrati *et al.* [1] use K itemsets to recover a collection of frequent itemsets. Like maximal itemsets the method cannot recover the frequency information from the summary. Recently Yan *et al.* [17] proposed *pattern profiles*, an approach to compress closed itemsets. The authors demonstrate that the approach can effectively summarize itemsets, resulting in good compression while retaining high accuracy. However, from an efficiency perspective it is not clear how well this approach will scale to large datasets. The proposed strategy needs to repeatedly scan the original dataset in order to achieve good summarization quality. Obviously, this can become very expensive when dealing with large datasets. Furthermore, the resulting pattern profiles can be quite unbalanced in terms of their size and distribution. There is no easy way to control this problem which can result in poor interpretability.

In this paper we present a new approach to summarizing itemsets. Our approach relies on some well established ideas in graphical models and probabilistic inference. The key idea is to derive a graphical model summary of the data from the set of frequent non-derivable patterns. This serves as the *profile summary* of the dataset. Subsequently the list of frequent itemsets and associated counts can be computed using standard probabilistic inference methods [13, 6]. The resultant profile summary can be thought of as generalized

*This work is supported in part by the following research grants: DOE Award No. DE-FG02-04ER25611; NSF CAREER Grant IIS-0347662;

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

KDD'06, August 20–23, 2006, Philadelphia, Pennsylvania, USA.
Copyright 2006 ACM 1-59593-339-5/06/0008 ...\$5.00.

non-derivable patterns. This is a very nice property considering that in many cases, non-derivable patterns are already able to offer a very condensed representation of a collection of frequent itemsets[4]. Furthermore, there is no need to scan the original dataset during the summarization. This is desirable for truly large datasets where repeated scans can be prohibitive.

Our experimental results on several real datasets show that the proposed approach compares favorably with profile patterns on the axes of accuracy, space and performance. Specifically we find on real datasets that the proposed approach is much more accurate given the same space budget, and more often than not significantly faster than profile patterns.

The rest of the paper is organized as follows. We define the problem of the itemset pattern summarization and briefly go over the related work in probabilistic graphical models in Section 2. In Section 3 we detail our proposed probabilistic model-based itemset summarization approach. We present experimental results in Section 4. Finally, we discuss future work and conclude in Section 5.

2. PROBLEM STATEMENT & BACKGROUND

Let \mathcal{I} be a set of items, i_1, i_2, \dots, i_d . A subset of \mathcal{I} is called an *itemset*. The *size* of an itemset is the number of items it contains. An itemset of size k is a k -itemset. A transactional dataset is a collection of itemsets, $D = \{t_1, t_2, \dots, t_n\}$, where $t_i \subseteq \mathcal{I}$. For any itemset α , we write the transactions that contain α as $D_\alpha = \{t_i | \alpha \subseteq t_i \text{ and } t_i \in D\}$. In the probabilistic model context, each item is modeled as a random variable.

DEFINITION 1. (*(σ -)Frequent itemset*). For a transactional dataset D , an itemset α is (σ -)frequent if $|D_\alpha| \geq \sigma$, where $|D_\alpha|$ is called the support of α in D , denoted as $s(\alpha)$, and σ is a user-specified non-negative threshold. $\frac{|D_\alpha|}{|D|}$ is called the relative support of α in D .

Frequent itemsets satisfy the important *Apriori* property: any subset of a frequent itemset is also frequent [2]. All existing frequent itemset mining algorithms rely on this property to prune the search space. Since the number of subsets of a large frequent itemset is explosive, it is more appropriate to mine closed frequent itemsets or non-derivable frequent itemsets only. We define these below.

DEFINITION 2. (*Closed frequent itemset*). A frequent itemset α is closed if there does not exist an itemset β such that $\alpha \subset \beta$ and $D_\alpha = D_\beta$.

DEFINITION 3. (*(Non-)derivable frequent itemset*). A frequent itemset α is derivable if its support can be exactly inferred from the support of its sub-itemsets based on the inclusion-exclusion principle. Otherwise it is non-derivable. We refer the reader to [4] for more information about non-derivable patterns and the inclusion-exclusion principle.

2.1 Itemset Pattern Summarization

We define the itemset pattern-summarization problem as follows: given a collection of frequent itemsets, we want to find a more concise representation such that the original collection of itemsets and their support information can be reasonably recovered. Additionally, the summarization should be tunable in terms of controlling the trade-off amongst often competing metrics, namely summarization quality, summary size or compactness, and efficiency.

As noted above, closed itemsets and non-derivable itemsets have been shown to be two successful concise representations for a collection of frequent itemsets. In many cases, they can significantly reduce the number of itemsets in the representation without information loss. However, they can still be quite large, which is why new summarization schemes are necessary.

2.2 Pattern Profile Summarization Approach

Recently, Yan *et al.* [17] proposed a pattern summarization approach for frequent itemsets. The key notion is *pattern profile*, which can be viewed as a generalization of closed itemsets. Specifically, a pattern profile is a triple $\langle a, b, c \rangle$ where a is a set of items, b is a distribution vector on these items and c is the relative support of the whole pattern profile. A frequent itemset is a special pattern profile where the distribution vector entirely consists of 1s. Essentially, a pattern profile is a compressed representation of similar itemsets and can be used to summarize them. In the scheme proposed by Yan *et al.* [17], pattern profiles are compared based on the *Kullback-Leibler* (KL) divergence between their distribution vectors. The principle is that the pattern profiles having smaller KL divergence are more similar than those having larger KL divergence. Based on this similarity measure, the traditional k -means clustering algorithm is applied to cluster the itemsets into K groups. Then, a representative profile pattern is identified for each group and used as a compressed representation for that group of itemsets.

2.3 Markov Random Fields

A Markov Random Field (MRF) is an undirected graphical model in which vertices represent variables and edges represent correlations between variables. The joint distribution associated with an undirected graphical model can be factorized as follows:

$$p(X) = \frac{1}{Z(\psi)} \prod_{C_i \in \mathcal{C}} \psi_{C_i}(X_{C_i})$$

where \mathcal{C} is the set of maximal cliques associated with the undirected graph; ψ_{C_i} is a potential function over the variables of clique C_i and $\frac{1}{Z(\psi)}$ is a normalization term. A *clique* is a subset of vertices in the graph that are fully-connected. A *maximal clique* is a clique that cannot have more vertices added and still remain a valid clique. We associate with each maximal clique a non-negative and real-valued potential function.

2.3.1 Using Frequent Itemsets to Construct an MRF

The idea of using frequent itemsets to construct an MRF was first proposed by Pavlov *et al.* [15]. A k -itemset and its support represents a k -way statistic and can be viewed as a constraint on the true underlying distribution that generates the data. Given a set of itemset constraints, a maximum entropy distribution satisfying all these constraints is selected as the estimate for the true underlying distribution. This maximum entropy distribution is essentially equivalent to an MRF. There is a simple algorithm, called *iterative scaling* that one can use to construct an MRF from a given set of itemsets.

While our work is clearly inspired by the work by Pavlov *et al.* [15], there are significant distinctions between the two. Their goal is to estimate query selectivity. When a query Q with variables x_Q is posed in real-time, all itemsets whose variables are subsets of x_Q are selected to construct an MRF on x_Q in an online fashion. Then the selectivity

of Q is estimated from the model. When a new query is posed, a new model must be constructed from scratch. This approach is inherently online and local. In contrast, our goal is to summarize the itemsets. Our MRF is global in that it attempts to integrate the information of all itemsets seen thus far. This *global character* allows for more accurate support estimations.

3. ALGORITHM

Our goal is to use a small number of itemsets to construct an MRF over all involved variables and use it to summarize a much larger collection of frequent itemsets. We rely on the following lemmas (their proofs can be found in our technical report [16]).

LEMMA 1. *Given a transactional dataset D , the MRF M constructed from all of its σ -frequent itemsets is equivalent to M' , the MRF constructed from only its σ -frequent non-derivable itemsets.*

LEMMA 2. *Given an itemset α and all of its non-derivable sub-itemsets and their support estimations, i.e., $\hat{s}_1, \dots, \hat{s}_l$ (true supports are s_1, \dots, s_l , respectively), if these estimations are error bounded by e_1, \dots, e_l , i.e., $|\hat{s}_1 - s_1| \leq e_1, \dots, |\hat{s}_l - s_l| \leq e_l$, then the support estimation for α , $\hat{s}(\alpha)$, derived from these sub-itemsets is error bounded by $e_1 + e_2 + \dots + e_l$.*

Motivated by the above lemmas, we focus on the task of summarizing non-derivable itemsets. These patterns capture the non-redundant distribution information of the data according to Lemma 1. Furthermore, if we summarize these patterns well, the summarization quality for all other derivable patterns will be error-bound according to Lemma 2.

3.1 Summarizing Itemsets Using MRFs

The idea behind our proposed summarization technique is simple. We use statistics of smaller itemsets to construct an MRF and then use this model to infer the supports of larger itemsets. If the estimations are accurate enough (within a user-specified error tolerance), we bypass the corresponding patterns. Otherwise we use the extra information from these itemsets to augment the model. Our summarization proceeds in a level-wise fashion. First, all 1-itemsets are collected and used to construct an MRF. Then, we infer the supports for all 2-itemsets. We bypass those 2-itemsets whose supports are well-estimated from the model and use the information of all skewed 2-itemsets to augment it. We move on to process all 3-itemsets and so on. This process will be repeated level-by-level until we process all the itemsets. At the end of the process, all itemsets remaining in the resulting model provide a concise representation of the original collection of itemsets.

Essentially, we select the skewed itemsets and add their information to the probabilistic model as it is constructed. Thus, we expect that the final resulting model will be able to faithfully capture the most significant dependency information in the data, summarizing the original patterns well. In other words, we try to reduce the original collection of itemsets by eliminating redundancy. The MRF fully specifies the conditional independence in the data, thus if an itemset does not introduce any extra significant dependency information to the current model, it will be pruned. Furthermore, we introduce a parameter δ to tune the granularity of the summarization. δ specifies the error tolerance during the summarization. If the estimation error is within the tolerance, we bypass the corresponding pattern. Otherwise we

label it as skewed. By specifying the error tolerance, the parameter δ provides a mechanism to trade-off summarization accuracy for space.

Itemset_Summarize(C, δ)
Input : C , collection of frequent itemsets;
 δ , error tolerance threshold;
Output : \mathcal{R} , reduced collection of itemsets;
1. Obtain all 1-itemsets in C and their supports, use them to initialize \mathcal{R} ;
2. $k \leftarrow 2$;
3. **while** $k < MAX_LEVEL$
4. Use itemsets in \mathcal{R} to construct an MRF M ;
5. Obtain all k -itemsets in C and their supports;
6. **for** each k -itemset p :
7. Use M to estimate $s(p)$, calculate estimation error e ;
8. **if** $e > \delta$ **then** add p to \mathcal{R} ;
9. $k \leftarrow k + 1$;
10. **return** \mathcal{R} ;

Figure 1: Itemset pattern summarization algorithm

The formal summarization algorithm is presented in Figure 1. Note that this algorithm can be easily extended to a more general summarization scheme that can be applied to summarize any collection of itemsets. We do not pursue this direction in our study, focusing on the summarization of a complete collection of σ -frequent itemsets. The time complexity of the summarization algorithm is dominated by the MRF construction, which we will describe below.

3.2 Constructing MRFs

As was mentioned in Section 2, the iterative scaling algorithm can be used to construct an MRF from a set of itemsets. Figure 2 presents a high-level outline of a computationally efficient version of the algorithm given by Jelinek [12]. During the construction process, we need to iterate over all itemset constraints and repeatedly update the model to force it to satisfy the current itemset constraint. The model update relies on the support estimation for the current itemset constraint. Thus, we need to continuously make inferences on the current model. If the iterative scaling algorithm runs for k iterations and there are m itemset constraints, the time complexity of the algorithm will be $O(k \times m \times t)$, where t is the average inference time over a constraint. Therefore, efficient inference is crucial to the running time of the construction algorithm. In our study, we exploit and evaluate two inference engines, the *Junction Tree* inference algorithm and the *Markov Chain Monte Carlo* (MCMC) inference algorithm.

Iterative_Scaling(C)
Input : C , collection of itemsets;
Output : MRF M ;
1. Obtain all involved variables v and initialize parameters of M ;
 //typically uniform over v ;
2. **while** (Not all constraints are satisfied)
3. **for** (each constraint C_i)
4. Update M to force it to satisfy C_i ;
5. **return** M ;

Figure 2: Iterative scaling algorithm

3.2.1 Junction Tree Inference Algorithm

The junction tree algorithm [13] is a commonly-used exact inference engine for probabilistic models. The general inference problem is to calculate the marginal probability

of a set of variables, given the observed values of another set of variables. In our context, there are no observed variables, and our goal is to calculate the marginal probability associated with the itemsets.

Specifically, the junction tree algorithm decomposes the original model into a *hyper-tree*, in which each tree node consists of a set of variables in the original graphical model. Two sets of variables associated with two tree nodes can overlap, with the overlapped part called their *separator*. Each tree node corresponds to a unique maximum clique in the graph formed by triangulating the original model. The junction tree must satisfy the *running intersection* property, i.e., for every pair of cliques V and W , all cliques on the path between V and W contain $V \cap W$. Each tree node has an expectation on the marginal probability over its associated variables, called its *belief*. The beliefs of all the tree nodes propagate along all distinct paths to achieve a global consistency, which implies that the inference problem within a tree node can be solved independently of the other tree nodes.

The time complexity of the junction tree algorithm is exponential in the maximum number of variables contained in a tree node (also known as the *treewidth* of the underlying graphical model). If the underlying MRF is relatively simple (with a relatively low treewidth), then the junction tree algorithm can yield exact inferences very efficiently. However, when the model becomes complex (the treewidth becomes large), then the exact inference will become slow, sometimes even intractable. In such cases we must resort to approximate inference algorithms.

3.2.2 MCMC Inference Algorithm

MCMC is a method for generating approximate samples from complicated distributions [6]. In our study, we use a type of MCMC algorithm known as *Gibbs sampling* to draw dependent samples from the joint posterior distribution, from which we evaluate the marginal probabilities corresponding to the itemsets. We specify a full conditional distribution $p(x_i | -x_i)$ for each variable x_i in our MRF, where $-x_i$ is the set of variables in the graph not including x_i . Then we draw samples from it. Note that the Markov property indicates that it is sufficient to condition on the neighboring variables of x_i in the MRF. Gibbs sampling proceeds by sampling each x_i from its conditional distribution, given the current values of the other neighboring variables. Marginal probabilities can be estimated by summing over the samples. Note that the Gibbs sampling scheme yields approximate inferences. The quality of the approximate inference is reasonably good when the sample size is large enough. To diminish the effect of the starting distribution, we generally discard a certain number of early iterations, referred to as *burn-in* [6].

3.3 Generalized Non-derivable Itemsets

The probabilistic model-based summarization scheme returns a subset of the original collection of itemsets to the end-user. Similar to the conclusions of Yan *et al.* [17] that pattern profiles can be viewed as generalized closed itemsets, the resulting itemsets in our summarization approach can be viewed as generalized non-derivable itemsets. First, we construct the probabilistic models based on the non-derivable itemsets. As a result, all the itemsets in the final summary are non-derivable. Second, we allow for a certain error tolerance when summarizing the itemsets. If a particular itemset follows the currently-known conditional independence struc-

	k	n	m	d
Chess	75	3196	118252	0.493
Accidents	468	340183	11500870	0.0722
Mushroom	119	8124	186852	0.193
Web	294	32711	98654	0.0102

Table 1: General characteristics of the datasets. k is the number of distinct items, n is the number of records, m is the number of total items and $d = \frac{m}{kn}$ is the density index.

ture specified by the model, we consider its support to be known. Note that there might be cases where we are not able to derive an itemset’s support based solely on the inclusion-exclusion principle. We are able to derive it according to the further conditional independence information, however. Essentially, we relax the requirement for an itemset to be “derivable”, which will significantly increase the number of derivable patterns. Furthermore, the greater relaxation, the greater number of derivable patterns, which will result in a more compressed summarization.

4. EXPERIMENTAL RESULTS

In this section, we examine the performance of our proposed approach on real datasets. We compare our probabilistic model-based summarization (abbreviated as PM) against the state-of-the-art pattern profile summarization scheme (abbreviated as PP). The summarization algorithm is implemented in C++. The junction tree and Gibbs sampling inference algorithms¹ are implemented using Intel’s Open-Source Probabilistic Networks Library (<https://sourceforge.net/projects/openpnl/>). Also, we implement the pattern profile summarization algorithm in C++ and tune it to achieve performance similar to that reported in [17].

4.1 Experimental Setup

Unless otherwise noted, all the experiments were conducted on a Pentium 4 2.66GHz machine with 1GB RAM running Linux 2.6.8. We use the implementation of the apriori algorithm in [3] to collect the frequent itemsets and the corresponding closed itemsets. We use the implementation in [5] to collect frequent non-derivable itemsets. We detail the datasets and performance metrics considered in our evaluation in the text below.

Datasets: We use four publicly available datasets in our experiments. They are the Chess, Accidents, Mushroom and Microsoft Anonymous Web datasets. The first three datasets are publicly available at the FIMI repository (<http://fimi.cs.helsinki.fi/>) and the last Web dataset is publicly available at the UCI KDD archive (<http://kdd.ics.uci.edu/>). The main characteristics of the datasets are summarized in Table 1. The Chess and Mushroom datasets are relatively dense. The Web dataset is quite sparse and the Accidents dataset is somewhere in between.

Summarization accuracy:

DEFINITION 4. Restoration Error. Given a collection of itemsets $\Phi = \{\alpha_1, \alpha_2, \dots, \alpha_l\}$, the quality of a pattern summarization can be evaluated by the following average relative error (called restoration error):

$$E = \sum_{\alpha_k \in \Phi} \frac{|s(\alpha_k) - \hat{s}(\alpha_k)|}{s(\alpha_k)}$$

where s is the true support and \hat{s} is its estimation. Restoration error measures the average relative error between the estimated support of a pattern and its true support.

Summary size: In order to make a fair comparison between the two approaches, we need to consider the sum-

¹The results on Gibbs sampling are omitted for space reasons and can be found in our technical report [16].

Level	Itemsets No.	Skewed Itemsets No. (Varying δ)					
		0.05	0.10	0.15	0.20	0.25	0.30
1	31	31	31	31	31	31	31
2	335	6	0	0	0	0	0
3	653	14	19	2	1	0	0
4	257	2	0	0	0	0	0
Sum	1276	53	50	33	32	31	31

Table 2: Skewed itemset distribution on the Chess dataset when varying error threshold

mary size. The comparison should be made between summarizations which are of the same size. In our study, we use the number of bytes taken by a summarization to quantify its size. Specifically, we assume an item in the summary takes 2 bytes (a short integer) and a floating point support in the summary takes 4 bytes. For example, the following itemset takes 8 bytes, $\{(item_1, item_2), 0.1\}$ and the following pattern profile takes 22 bytes. $\{(item_1, item_2, item_3), (1.0, 0.8, 0.6), (0.1)\}$

Summarization time: We also consider the time taken to summarize the itemsets. Making a fair timing comparison between the two summarization schemes is not easy. Both our approach and pattern profile approach are iterative processes. The running times are highly dependent of the convergence criteria, which can be rather subjective. We report the timing results of those summarizations from which we collect the accuracy results.

4.2 Results on the Chess & Accidents Datasets

First, we report the experimental results on the Chess dataset. For this set of experiments, we set $\sigma = 2000$ to collect the frequent itemsets. As a result, there are 166581 frequent itemsets, from which 1276 itemsets are non-derivable. We also collect all the 68967 closed frequent itemsets at this support level for the pattern profile summarization scheme.

Figure 3a presents the summarization quality as we vary the error tolerance threshold. For our approach we report results on summarizing all itemsets and on summarizing all non-derivable itemsets. For the pattern profile approach, we only report the results on summarizing all itemsets. As a baseline, the results based on a naive independence model are also plotted in the figure. We see that the probabilistic model-based summarization scheme effectively summarizes the itemsets. The restoration error of all frequent patterns is slightly higher than that of non-derivable patterns. This is expected, considering our approach focuses on summarizing non-derivable patterns. It is worth pointing out that the restoration error on all frequent itemsets is also very small, supporting our claim that non-derivable patterns play a key role in representing the whole collection of frequent itemsets. Furthermore, it can be clearly seen that the restoration error increases as we raise the error tolerance threshold. This is due to the fact that we will lose more information with larger error tolerance thresholds. Particularly, the summarization with the threshold above 0.25 becomes equivalent to the naive independence model-based summarization. The advantage of our approach over the pattern profile approach is clearly demonstrated in the figure. For the pattern profiles of the same size, the restoration error is much higher than that of our approach and is actually quite close to that of the naive independence model.

Figure 3b presents the summary sizes with different error tolerance thresholds. The sizes of the original collection of patterns and the naive independence model are also plotted here for reference purpose. As one can see, our summaries use a very small amount of space to represent a much larger

Level	Itemsets No.	Skewed Itemsets No. (Varying δ)					
		0.10	0.20	0.30	0.40	0.50	0.60
1	35	35	35	35	35	35	35
2	207	78	42	25	11	2	1
3	269	31	19	25	46	40	8
4	23	0	0	0	0	0	1
Sum	534	144	96	85	92	77	45

Table 3: Skewed itemset distribution on the Mushroom dataset when varying error threshold

collection of itemsets. For example, the summary takes 398 bytes at an error threshold of 0.05 to summarize itemsets of size 12480 bytes, about a 30-fold reduction.

Figure 3c presents the timing performance of our approach. As one can see, our approach quickly summarizes the itemsets of this dataset. In all cases, the summarization takes less than 5 seconds. In contrast, the pattern profile approach does not finish before it exhausts the memory. When we submit the summarization job to a computer with more memory (4GB RAM) and the same CPU speed at the Ohio Supercomputer Center, the pattern profile approach takes about 40 minutes to finish. Another trend is that our approach takes more time when using a lower error tolerance threshold, since the models with lower thresholds are more complex.

Table 2 presents the distribution of the skewed itemsets at different levels with respect to different error tolerance thresholds. As can be seen from the table, the numbers of skewed itemsets are very small at all the thresholds. For example, at the threshold of 0.05, there are 6, 14 and 2 skewed 2, 3 and 4-itemsets, respectively. As we raise the threshold, the overall number of skewed itemsets decreases.

The results on the Accidents dataset are quite similar to that on the Chess dataset and are omitted in the interest of space. However, the complete results can be found in our technical report[16].

Note that both of the two datasets are relatively dense and largely satisfy the independence assumption. For this kind of dataset, the MRF based summarization scheme works extremely well. Interestingly, we note that on these two datasets, the frequent non-derivable patterns are much fewer than the frequent closed patterns. Next we examine the performance of our approach on the skewed datasets that do not satisfy the independence assumption.

4.3 Results on the Mushroom & Microsoft Web Datasets

First, we report the experimental results on the Mushroom dataset. For this set of experiments, we set $\sigma = 2031$ (a support threshold of 25%) to collect the frequent itemset patterns, resulting in 5545 frequent itemsets, from which 688 are closed and 534 are non-derivable.

Figure 4a-c present the restoration error, summary sizes, and summarization times, respectively. Figure 4a shows that the independence assumption does not hold well on this dataset. The restoration errors for all itemsets and non-derivable itemsets are 20% and 39%, respectively. The restoration errors for all frequent patterns and non-derivable patterns are reasonably low, and are much lower than that of the pattern profile summaries of the same size. Note that both approaches work better than the naive independence model. Figure 4b shows that the summaries take relatively more space, compared with that on the previous two datasets. Figure 4c shows that our approach is faster than the pattern profile approach.

Table 3 presents the distribution of the skewed itemsets.

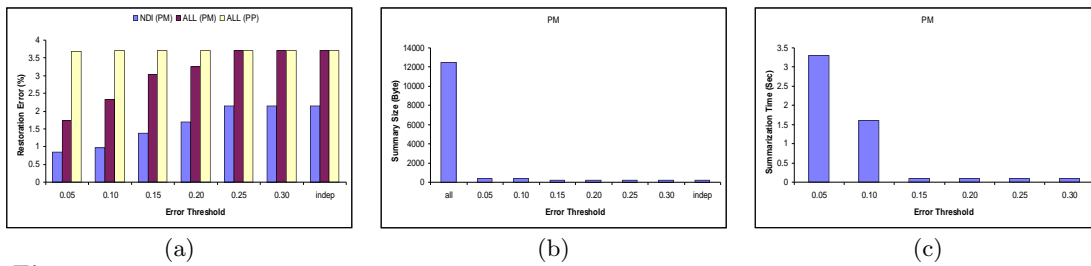


Figure 3: Results on the Chess dataset:(a)Restoration error (b)Summary size (c)Summarization time

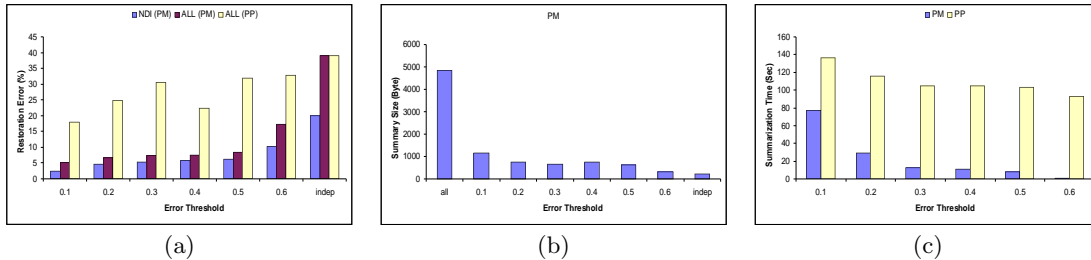


Figure 4: Results on the Mushroom dataset:(a)Restoration error (b)Summary size (c)Summarization time

Compared with the previous two datasets, the proportion of the skewed itemsets is much higher on this dataset, which explains why we need more space when summarizing.

We also report the results on the sparse Microsoft Web dataset. The results overall are quite similar to that on the Mushroom dataset. Our approach consistently outperforms the pattern profile approach in terms of restoration error. The proportion of the skewed itemsets is relatively high, so we have to use more space and time to summarize the patterns. Again, the complete results can be found in [16].

4.4 Result Summary & Discussion

The experimental results have shown that the probabilistic model-based summarization scheme overall is very efficient and effective in summarizing itemsets. In most cases, it outperforms the pattern profile summarization scheme.

When datasets are dense and largely satisfy the conditional independence assumption, there usually exists a large amount of redundancy in the corresponding itemsets. In such cases our approach will be extremely efficient and effective. On the other hand, when datasets become sparse and do not satisfy conditional independence assumption well, the summarization task for our approach becomes more difficult. As a result, we have to spend more space and time on summarizing the corresponding itemsets.

5. CONCLUSIONS

In this paper, we have presented a novel approach for summarizing itemset patterns using MRFs, which exploit conditional independence relations among the items in the transactional data. The success of our approach on all the tested real-world datasets indicates that the conditional independence structure is quite common. This is particularly true when dealing with relatively dense datasets, whose dense structure leads to significant redundancy in mined itemsets. As a result, our approach is a viable option for many real-world datasets. Given the same summary size, our approach can achieve up to a 4-fold improvement in accuracy compared with the pattern profile approach. On certain datasets, it is orders of magnitude faster than the pattern profile approach.

In the future, we would like to examine the scalability of our approach on truly large-scale collections of itemsets

which can result in very complex MRFs. We intend to evaluate and adapt ideas from the approximate probabilistic inference field for this purpose. It would be interesting to evaluate the viability of the proposed approach in a streaming or incremental setting.

6. REFERENCES

- [1] F. N. Afrati, A. Gionis, and H. Mannila. Approximating a collection of frequent sets. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 12–19, 2004.
- [2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499, 1994.
- [3] C. Borgelt. Efficient implementations of apriori and eclat. In *Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations*, 2003.
- [4] T. Calders and B. Goethals. Mining all non-derivable frequent itemsets. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery*, 2002.
- [5] T. Calders and B. Goethals. Depth-first non-derivable itemset mining. In *Proceedings of the SIAM 2005 International Conference on Data Mining*, 2005.
- [6] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian data analysis*. Chapman & Hall/CRC, 2004.
- [7] A. Ghoting, G. Buehrer, S. Parthasarathy, D. Kim, A. Nguyen, Y. K. Chen, and P. Dubey. Cache-conscious frequent pattern mining on a modern processor. In *Proceedings of the 31st International Conference on Very Large Data Bases*, pages 577–588, 2005.
- [8] K. Gouda and M. J. Zaki. Genmax: An efficient algorithm for mining maximal frequent itemsets. *Data Mining and Knowledge Discovery*, 11(3):223–242, November 2005.
- [9] D. Gunopulos, R. Khardon, H. Mannila, and H. Toivonen. Data mining, hypergraph transversals, and machine learning. In *Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 209–216, 1997.
- [10] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 1–12, 2000.
- [11] J. Han, J. Wang, Y. Lu, and P. Tzvetkov. Mining top-k frequent closed patterns without minimum support. In *Proceedings of the 2002 IEEE International Conference on Data Mining*, pages 211–218, 2002.
- [12] F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, Cambridge, MA, 1998.
- [13] S. Lauritzen and D. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B (Methodological)*, 50(2):157224, 1988.
- [14] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *Database Theory - ICDT '99, 7th International Conference, Jerusalem, Israel, January 10-12, 1999, Proceedings*, pages 398–416, 1999.
- [15] D. Pavlov, H. Mannila, and P. Smyth. Beyond independence: probabilistic models for query approximation on binary transaction data. *IEEE Transactions on Knowledge and Data Engineering*, 15(6):1409–1421, November 2003.
- [16] C. Wang and S. Parthasarathy. Summarizing itemset patterns using probabilistic models. In *The Ohio State University, Technical Report*, 2006.
- [17] X. Yan, H. Cheng, J. Han, and D. Xin. Summarizing itemset patterns: a profile-based approach. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 314–323, 2005.
- [18] M. J. Zaki and C.-J. Hsiao. Charm: An efficient algorithm for closed itemset mining. In *Proceedings of the Second SIAM International Conference on Data Mining*, 2002.
- [19] M. J. Zaki, S. Parthasarathy, and W. L. Mitsunori Ogihara. New algorithms for fast discovery of association rules. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, pages 283–286, 1997.