Available online at www.sciencedirect.com
ScienceDirect
Expert Systems with Applications 36 (2009) 72–80
Expert Systems with Applications
www.elsevier.com/locate/eswa

An improved data mining approach using predictive itemsets

Tzung-Pei Hong [a,*], Chyan-Yuan Horng [b], Chih-Hung Wu [c], Shyue-Liang Wang [d]

[a] *Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan, ROC*
[b] *Institute of Information Engineering, I-Shou University, Kaohsiung 840, Taiwan, ROC*
[c] *Department of Electrical Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan, ROC*
[d] *Department of Information Management, National University of Kaohsiung, Kaohsiung 811, Taiwan, ROC*

Abstract

In this paper, we present a mining algorithm to improve the efficiency of finding large itemsets. Based on the concept of prediction proposed in the (n,p) algorithm, our method considers the data dependency in the given transactions to predict promising and non-promising candidate itemsets. Our method estimates for each level a different support threshold that is derived from a data dependency parameter and determines whether an item should be included in a promising candidate itemset directly. In this way, we maintain the efficiency of finding large itemsets by reducing the number of scanning the input dataset and the number candidate items. Experimental results show our method has a better efficiency than the apriori and the (n,p) algorithms when the minimum support value is small.

Keywords: Data mining; Association rule; Predictive itemset; Data dependency; Predicting minimum support

1. Introduction

Years of effort in data mining have produced a variety of efficient techniques (Chen, Han, & Yu, 1996). Depending on the types of datasets processed, mining approaches may be classified as working on transaction datasets, temporal datasets, relational datasets, or multimedia datasets, among others. On the other hand, depending on the classes of knowledge derived, mining approaches may be classified as finding association rules, classification rules, clustering rules, or sequential patterns (Agrawal & Srikant, 1995), etc. Among these techniques, finding association rules from transaction datasets is usually an essential task (Agrawal, Imielinksi, & Swami, 1993b; Agrawal & Srikant, 1994; Agrawal, Srikant, & Vu, 1997; Ezeife, 2002; Han & Fu, 1995; Mannila, Toivonen, & Verkamo, 1994; Park, Chen, & Yu, 1997; Srikant & Agrawal, 1995, 1996; Wojciechowski & Zakrzewicz, 2002).

Many algorithms for mining association rules from transactions are proposed, most of which are executed in level-wise processes. That is, itemsets containing single items are processed first, then itemsets with two items are processed. The process was repeated, continuously adding one more item each time, until some criteria are met. The famous apriori mining algorithm was proposed by Agrawal et al. (1993a, 1993b). The apriori iterates two phases, the phase of candidate generation and the phase of verification. Possible large itemsets are produced in the first phase and verified in the second phase by scanning the input dataset. Since itemsets are processed level by level and datasets had to be scanned in each level, the verification phase thus dominates the performance. Han, Pei, and Yin (2000) then proposed the Frequent-Pattern-tree (FP-tree) structure for efficiently mining association rules without generation of candidate itemsets. The FP-tree was used to compress a database into a tree structure which stored only large items. Several other algorithms based on the FP-tree structure have also been proposed. For example, Qiu, Lan, and Xie (2004) proposed the QFP-growth mining approach to mine association rules. Zaiane and Mohammed (2003) proposed the COFI-tree structure to replace the conditional

[*] Corresponding author.
E-mail addresses: tphong@nuk.edu.tw (T.-P. Hong), hcy@ms4.url.com.tw (C.-Y. Horng), johnw@nuk.edu.tw (C.-H. Wu), slwang@nuk.edu.tw (S.-L. Wang).

0957-4174/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2007.09.009

FP-tree. Ezeife (2002) constructed a generalized FP-tree, which stored all the large and non-large items, for incremental mining without rescanning databases. Koh and Shieh (2004) adjusted FP-trees also based on two support thresholds, but with a more complex adjusting procedure and spending more computation time than the one proposed in this paper.

Denwattana and Getta (2001) proposed an algorithm (referred to as the (n,p) *algorithm*) to reduce the numbers of scanning input datasets for finding large itemsets. The (n,p) algorithm also iterates two phases, the phase of prediction and the phase of verification. Unlike the apriori, the (n,p) algorithm predicts large itemsets for p-levels in the first phase and verifies all these p-level itemsets in the second phase. A heuristic estimation method is presented to predict the possibly large itemsets. If the prediction was valid, then the approach is efficient in finding the actually large itemsets.

In this paper, we propose a mining algorithm to improve the efficiency of finding large itemsets. Our approach is based on the concept of prediction presented in the (n,p) algorithm and considers the data dependency among transactions. As the (n,p) algorithm does, our method iterates the same two phases but uses a new estimation method to predict promising and non-promising candidate itemsets flexibly. The estimation mechanism computes for each level a different support threshold derived from a data dependency parameter and determines whether an item should be included in a promising candidate itemset directly by the support values of items. Since we reduce the number of candidate itemsets to be verified by the new estimation mechanism and the number of scanning of the input dataset by the concept of prediction of the (n,p) algorithm, the performance of finding large itemsets can be improved.

The rest of this paper is organized as follows. Section 2 presents the related works of finding large itemsets. The apriori algorithm and the (n,p) algorithm are reviewed. In Section 3, we describe our motivation and the theoretical foundation of our method. Detailed description of our algorithm is given in Section 4 together with a simple example. Experimental results and the comparison on the performance of the apriori, the (n,p) algorithm, and our method are shown in Section 5. Conclusions are given in Section 6.

2. Related works

One application of data mining is to induce association rules from transaction data, such that the presence of certain items in a transaction will imply the presence of certain other items. Below we briefly reviewed the apriori and the (n,p) algorithm.

2.1. The apriori algorithm

Agrawal and Srikant (1994) and Agrawal et al. (1993b, 1997) propose a famous mining algorithm, the apriori, based on the concept of large itemsets to find association rules in transaction data. The apriori iterates two phases, the phase of candidate generation and the phase of verification, at each level. At the ith level, $i > 1$, itemsets consisting of i-items are processed. In candidate generation, all possible large i-itemsets are produced by combining the unrepeated elements of $(i-1)$-itemsets. In the verification phase, the input dataset is scanned and if the number of an i-itemset appearing in the transactions is larger than a pre-defined threshold (called the minimum support, or *minsup*), the itemset is considered as large. After that, these two phases iterate for $(i+1)$-level until all large itemsets are found.

Suppose that we have a dataset containing six transactions, as shown in Table 1, which has two features, transaction identification (ID) and transaction description (Items). There are eight items in the dataset. Assume that minsup = 30%. The transaction dataset is first scanned to count the candidate 1-itemsets. Since the counts of the items $a(4)$, $b(5)$, $c(6)$, $d(3)$, and $e(4)$ are larger than $6 * 30\% = 1.8$, they are thus put into the set of large 1-itemsets. Candidate 2-itemsets are then formed from these large 1-itemsets by taking any two items in the large 1-itemsets and counting if their occurrences are large than or equal to 1.8. Therefore, *ab*, *ac*, *ae*, *bc*, *bd*, *be*, *cd*, *ce*, and *de* then form the set of large 2-itemsets. In a similar way, *abc*, *abe*, *ace*, *bce*, *bcd*, *bde*, and *cde* form the set of large 3-itemsets, *abce* and *bcde* form the set of large 4-itemsets.

2.2. Denwattana and Getta's approach

In Denwattana and Getta (2001), the (n,p) algorithm tries to reduce the number of scanning datasets and improves the efficiency of finding large itemsets by "guessing" what items should be considered. The approach partitions candidate itemsets into two parts: positive candidate itemsets and negative candidate itemsets, where the former contains itemsets guessed to be large and the later contains itemsets guessed to be small. Initially, the (n,p) algorithm scans the dataset to find large itemsets of 1-items. Two parameters, called n-element transaction threshold t_t and frequency threshold t_f, are used to judge whether an item could compose a positive candidate itemset. According to the n-element transaction threshold t_t, only the transactions with item numbers (lengths) less than or equal to t_t are considered. The frequency of each item appearing in transac-

Table 1
A sample dataset of transactions

ID	Items
1	*abc*
2	*bcdef*
3	*abceg*
4	*acd*
5	*bcde*
6	*abce*