International Journal of Engineering Research and General Science Volume 2, Issue 6, October-November, 2014
ISSN 2091-2730

A Review on Frequent Pattern Mining

Vivek B. Satpute
M.E. Computer-II
Vidya Pratisthan`s College Of Engineering-Baramati.

Abstract - Data mining is the technique that discovers hidden pattern in data sets and association between the patterns. In this association rule mining one of the techniques used to achieve the objective of data mining. These rules are effectively used to uncover unknown relationships producing result that can give us a basis for forecasting and decision making. To discover these rules we have to find out the frequent item sets because these item sets are the building blocks to obtain Association rules with a given confidence and support.

Keywords – Data mining, association rule mining, frequent item sets, pattern, support, closed pattern, representative patterns,

INTRODUCTION

Frequent itemsets has a necessary part in a lot of data mining tasks that attempt to find interesting patterns from databases, like association rules, correlations, sequences, episodes, classifiers, clusters and lot of which the mining of association rules is one of the majority well-known problems. The novel inspiration for investigating association rules came from the required analyze so called supermarket transaction data, i.e., to study customer activities in the form of the purchased products. Association rules illustrate how frequently items are bayed together. E.g., an association rule ‘‘beer => chips (80%)’’ tells that four out of five customers that bought beer, they bought chips too. Such rules may become useful for decisions regarding product pricing, promotions, store outline and many others. As their introduction is in 1993 by Argawal et al. [1], the frequent itemset and association rule mining problems got a huge part of consideration. In the past two decades, more than hundreds of research papers are published giving novel algorithms or expansion on existing algorithms to resolve these mining problems much proficiently. Here, agenda is to explain some etchings of frequent itemset mining and give a comprehensive survey of the most influential algorithms that were proposed during the last two decade.

High Utility Itemset Mining

Vincent S. et al [2] focused on mining the high utility itemsets from the transactional database that refers to the detection of itemsets with high utility like profits. They utilizes a compact tree structure, named UP-Tree to make easy the mining performance and keep away from scanning original database frequently and to retain the information of transactions and high utility itemsets. To reduce the overestimated utilities stored in the nodes of global UP-Tree two different approaches are used, UP-growth (Utility Pattern Growth) and UP-Growth+

After building the structure of UP-Tree, for producing PHUIs(potential high utility itemsets) a fundamental thing is mine UP-Tree by using the FP-Growth. But in that, lot of candidates gets generated. So, they proposed an algorithm UP-Growth by adding two more strategies i.e. DLU (Discarding Local Unpromising) and DLN (Decreasing Local Node Utilities) into the structure of FP-Growth. By the strategies, utilities of itemsets that are overestimated can be reduced and therefore the number of PHUIs can be extra reduced.

Performance of UP-Growth is better than performance of FP-Growth by using DLU and DLN to decrease overestimated utilities of itemsets. However, they proposed an enhanced method, named UP-Growth+, for reducing overestimated utilities more effectively. In UP-Growth, minimum item utility table is used to reduce the overestimated utilities. In UP-Growth+, minimal node utilities in each path are used to make the estimated pruning values closer to real utility values of the pruned items in database.

Normally, UP-Growth+ do better than UP-Growth even though they have trade-offs on memory utilization. The cause behind that is UP-Growth+ makes use of minimal node utilities for further lessening miscalculated utilities of itemsets. Even if it uses time and memory to verify and store minimal node utilities, they are more efficient particularly when there are numerous longer transactions in databases. In difference, UP-Growth do better just when min-util is small. This is for the reason that when amount of candidates of the two algorithms is similar, UP-Growth+ takes added calculations and is therefore slower. Then, high utility itemsets are proficiently recognized from the set of PHUIs which is a lot lesser than HTWUIs (high-transaction weighted utility itemset) produced by IHUP (Incremental High Utility Pattern). So due to this, UP-Growth and UP-Growth+ complete improved performance than IHUP algorithm.

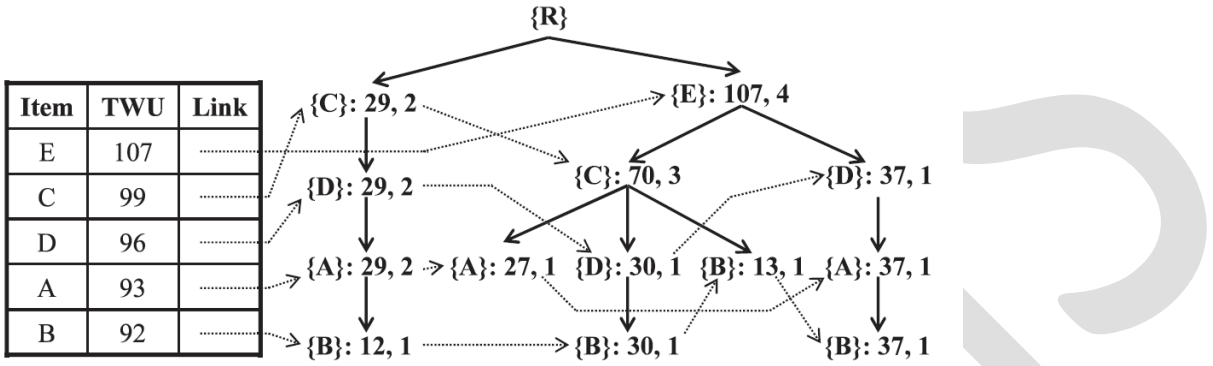

Figure 1. An IHUP-Tree when min_util=40.

Advantages:

4.1. More effective especially when there are a lot of longer transactions present in databases.

4.2. Performance is high over IHUP.

Disadvantages:

1) UP-growth and UP-Growth+ produce greatly less candidates than FP-growth.
2) Requires more time and memory.

Top-k Patterns

Han et al. [3] (Han, Wang, Lu TzVetkov, 2002) proposed most primitive algorithm of this type. Their top-k patterns are k most frequent closed patterns with a user-specified minimum-length, min_l. Restriction of minimum-length was necessary, because without it only length-1patterns (or their corresponding closed super-pattern) will be reported, as they often have the peak frequency. They projected their implementation by using FP-Tree. This method is a support-aware summarization as support is a measure for choosing summary patterns;

Afrati et al. [4] (Afrati, Gionis Mannila, 2004) proposed another top-k summarization algorithm. For this algorithm, pattern support is not a summarization criterion; rather high compressibility with maximal coverage is its main objective. To achieve this, it reports maximal patterns and also, allows few false positive in the summary pattern set. E.g. if the frequent patterns containing the sets ABC, ABD, ACD, AE, BE, and all their subsets, a specific setting of their algorithm may report ABCD and ABE as the summarized frequent patterns. Here point is, this covers all the original sets, and there are only two false positive (BCD, ABCD). Afrati et al. proved that, the problem of finding the best approximation of a frequent itemsets that can be spanned by k set is NP-Hard. They also used a greedy algorithm for keeping least approximation ratio. The interesting thing about this algorithm is that it gives high compressibility; authors reported that they could cover 70% of the whole frequent set collection by using only 20 sets (7.5% of the maximal patterns), with only 10% false-positive. In such a high compression, there is not much redundancy in the reported patterns; but they certainly are very dissimilar from each other.

Disadvantages:

[1] It is a post-processing algorithm, i.e., all maximal patterns first need to be obtained to be used as an input to this algorithm.

[2] The algorithm will not generalize well for complex patterns, like tree or graph, as the number of false-positives will be much higher and the coverage will be very low. So the approximation ratio mentioned above will not be achieved.

[3] It will not work if the application scenario does not allow false-positive.

Xin et al. [5](Xin, Cheng, Yan Han, 2006) proposed another top-k algorithm. For being most effective, the algorithms struggle for the set of patterns that collectively offer the best significance with minimal redundancy. Importance of a pattern is measured by a real value that encodes the level of usefulness of it and redundancy between a pair of patterns is measured by incorporating pattern likeness. Xin et al. proved that finding of a set of top-k patterns is NP-Hard even for itemset, which is the easiest kind of pattern. They also proposed a greedy algorithm that approximates the optimal solution with O(ln k) performance bound.

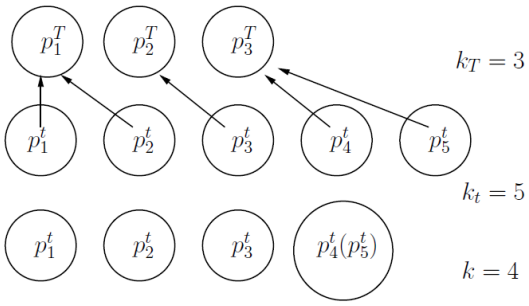

Figure 2: Find k patterns from (u , l) - pair

Disadvantages:

[1] The users need to find all the frequent patterns and evaluate their significances before the process of finding top-k patterns is initiated.

[2] For itemset pattern the distance calculation is very straightforward, this might be very costly for complex patterns.

A Profile Based Approach

Yan et al.[6] (Yan, Cheng, Han, Xin, 2005) proposed profile-based pattern summarization for itemset patterns. A profile-based summarization discovers k representative patterns, here they called it Master patterns and builds a model under which the sup-port of the rest of patterns can be easily recovered. A Master pattern is the combination of a set of awfully alike patterns. To cover the span of the whole frequent patterns, a Master pattern is very different from another Master pattern. Similarity considers both the pattern space and their support, so that they can be representative in the sense of Frequent Pattern Mining. The authors also built a profile for each Master pattern. Using the profile, the support of each frequent pattern that the corresponding Master pattern represents can be approximated with no consulting the original dataset. Profile based summarization is graceful due to its use of probabilistic model.

Advantages:

[1] This is effective to solve the interpretability matter caused by the vast amount of frequent patterns.

[2] This profile model is capable for recovering frequent patterns plus their supports.

Disadvantages:

[1] It makes opposing assumptions. On one side, the patterns represented by the identical profile are supposed to be similar. Whereas on the other side, based on how the supports of patterns are intended from a profile, the items in the identical profile

are expecting to be independent. It is tough for balancing two opposing necessities.

[2] The suggested algorithm for generating profiles is awfully time-consuming as it requires to scan the novel dataset frequently.

[3] The periphery between frequent patterns and infrequent patterns cannot be resolute by profiles.

FP-Growth

Han et al. (2004) [7] invented an FP-growth method that mines the whole set of frequent itemsets with no candidate generation. FP-growth is founded on theory of the divide and-conquer. The first scan of the database gives a list of frequent items. In that list the items are arranged by descending order of frequency. According to that list of descending frequency, the database is placed in a frequent pattern tree, or it can be called as FP-tree, which preserves the itemset association information. Starting from the every frequent length-1 pattern (as initial suffix pattern), the FP-tree is mined, building its conditional pattern base (a sub-database, that contents the set of prefix paths in the Frequent Pattern-tree co-occurring with the suffix pattern), then building its conditional FP-tree and performing mining recursively on such a structure of tree. The pattern expansion is attained by the concatenation of the suffix pattern with the frequent patterns produced from a conditional FP-tree. The major difficulty in FP-tree is that the creation of the frequent pattern tree is lengthy process, it is very time consuming process. Next, FP-tree based approach never gives flexibility and reusability of computation throughout process of mining. The FP-growth algorithm transforms the trouble of finding lengthy frequent patterns to hunting for the shorter ones recursively and then after concatenating the suffix. It utilizes the least frequent items as a suffix, offering excellent selectivity. Performance studies show that the system significantly decrease search time. Lot of alternatives and extensions are there to the FP-growth approach, together with depth-first generation of frequent itemsets by the Agarwal et al. (2001) H-Mine and Pei et al. (2001a) which discover a hyper-structure mining of the frequent patterns; constructing alternative trees; exploring the top-down and the bottom-up traversal of such trees in pattern-growth mining by Liu et al. (2002, 2003) and an array-based implementation of prefix-tree- structure for efficient pattern growth mining by Grahne and Zhu (2003)

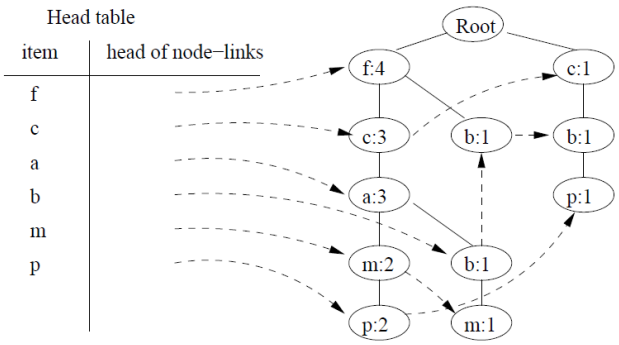

Figure 3: A Sample FP- Tree

Advantage:

1. It requires only 2 passes over data-set.
2. Efficiently it compresses data-set.
3. There is no candidate generation.

Disadvantage:

1. Sometimes tree structure becomes so big that FP-Tree may not fit in memory.

2. FP-Tree is expensive to build

Cluster-Based Representative Pattern-Set

Xin et al. [8] put forward the idea of δ-covered to simplify the concept of frequent closed pattern. The aim is to discover a minimum set of representative patterns that can δ-cover all frequent patterns. They conclude that the set cover problem can be relate to the main problem, and they build up two algorithms, RPglobal and RPlocal. RPglobal first generates the set of patterns that can be δ-covered by each pattern, and then employs the well-known greedy algorithm for the set cover problem to discover representative patterns. First, both RPglobal and RPlocal are clever to discover a subset of representative patterns; second, even if RPlocal gives extra patterns than RPglobal, the feat of RPlocal is awfully close to RPglobal. Nearly all the outputs of RPlocal are contained by two times of RPglobal. The outcomes of RPglobal are partial as minimum support becomes near to the ground, the number of closed patterns grows awfully speedy, and the running times of RPglobal go beyond the time limit (30 minutes). RPglobal does not size in good health w.r.t. the number of patterns, and is a lot slower than RPlocal However, RPglobal is extremely slow and space consuming. It is practicable as only the number of frequent patterns is not big. RPlocal is developed based on FPclose [12]. It integrates frequent pattern mining with representative pattern finding. RPlocal is very eficient, but it produces more representative patterns than RPglobal.

Advantage:

1. Combining both algorithms RPcombin gives balanced quality and efficiency.

Disadvantage:

1. Any single algorithms do not give adequate performance.

Probabilistic Model

Wang et al. [9] formulate generalization on different brief depiction of frequent patterns non-derivable patterns. In their work, probabilistic graphical models are utilized for the summarization job. More expressly, items are acquired like arbitrary variables and Markov Random Field (MRF) models on these variables are build founded on frequent itemset patterns and their support information. The summarization carried on in a level-wise style. Statistics of smaller itemsets are utilized for building an MRF model, and then supports of larger itemsets are concluded from the model. If the estimation error is within a user-specified patience, then there can option like sidestepping these itemsets, or else utilization of these itemsets to enhance the MRF model. Towards the end of the process, every itemset patterns in the resultant model bear a brief depiction of the original set of itemset patterns. wide experiential study on real datasets proved that the new approach is able to efficiently summarize great number of frequent itemset patterns.

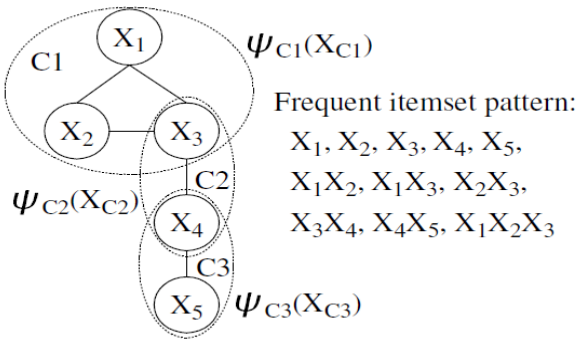

Figure 4: An MRF Example

Advantage:

1. When datasets are dense and mostly assure the conditional independence supposition, there typically exists a great quantity of redundancy in the consequent itemset patterns in which case this approach will be tremendously efficient and effective.

Disadvantages:

1. Markov Random Field model is not as easy as profiles; moreover it is also tough to understand.
2. When datasets happen to sparser and do not assure conditional independence assumption fine, and the summarization job will turn into more hard. So there will require more space and time on summarizing the consequent itemset patterns.

CONCLUSION

Pattern mining in recent times achieved major importance in the data mining community for the reason of its ability of being used as very important tool for the knowledge discovery and its applicability in the other data mining jobs like classification and clustering. Association rules are always of interest to the both database community as well as data mining users. Here a survey have provided of previous studies made in this area and recognize some vital gaps available in the current knowledge.

REFERENCES:

[1] R. Agrawal, T. Imielinski, and A. N. Swami, "Mining association rules between sets of items in large databases" in Proc. SIGMOD, Washington, DC, USA. , 1993, 207-216.

[2] Vincent S. Tseng, Bai-En Shie, Cheng-Wei Wu, and Philip S. Yu, Fellow, IEEE, "Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases," IEEE transactions on knowledge and data engineering, VOL. 25, NO. 8, AUGUST 2013.

[3] Han, J., Wang, J., Lu, Y., Tzvetkov, P. "Mining top-k frequent closed patterns without minimum support," In ICDM., 2002, 211-218.

[4] Afrati, F.N., A. Gionis and H. Mannila,. "Approximating a collection of frequent sets," Proceedings of the 2004 ACM SIGKDD International Conference Knowledge Discovery in Databases, Aug. 22-25, Seattle, WA., USA., 2004, 12-19.

[5] Xin D, Cheng H, Yan X, Han J "Extracting redundancy-aware top-k patterns," In: Eliassi-Rad et al 2006, 444-453 .

[6] X. Yan, H. Cheng, J. Han, and D. Xin, "Summarizing itemset patterns: A profile-based approach," in Proc. KDD, Chicago, IL, USA., 2005, 314-323.

[7] Han J, Pei J, Yin Y, Mao R "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," Data Min Knowl Discov., 2004 ,8(1):53-87

[8] D. Xin, J. Han, X. Yan, and H. Cheng, "Mining compressed frequent-pattern sets," in Proc. 31st Int. Conf. VLDB, Trondheim, Norway., 2005, 709-720.

C. Wang and S. Parthasarathy, "Summarizing itemset patterns using probabilistic models," in Proc. KDD, Philadelphia, PA, USA., 2006, 730-735