

Summarization in Pattern Mining

Mohammad Al Hasan

Rensselaer Polytechnic Institute, USA

INTRODUCTION

The research on mining interesting patterns from transactions or scientific datasets has matured over the last two decades. At present, numerous algorithms exist to mine patterns of variable complexities, such as set, sequence, tree, graph, etc. Collectively, they are referred as **Frequent Pattern Mining** (FPM) algorithms. FPM is useful in most of the prominent knowledge discovery tasks, like classification, clustering, outlier detection, etc. They can be further used, in database tasks, like indexing and hashing while storing a large collection of patterns. But, the usage of FPM in real-life knowledge discovery systems is considerably low in comparison to their potential. The prime reason is the lack of interpretability caused from the enormity of the output-set size. For instance, a moderate size graph dataset with merely thousand graphs can produce millions of frequent graph patterns with a reasonable support value. This is expected due to the combinatorial search space of pattern mining. However, classification, clustering, and other similar Knowledge discovery tasks should not use that many patterns as their knowledge nuggets (features), as it would increase the time and memory complexity of the system. Moreover, it can cause a deterioration of the task quality because of the popular “*curse of dimensionality*” effect. So, in recent years, researchers felt the need to summarize the output set of FPM algorithms, so that the summary-set is *small, non-redundant* and *discriminative*. There are different summarization techniques: lossless, profile-based, cluster-based, statistical, etc. In this article, we like to overview the main concept of these summarization techniques, with a comparative discussion of their strength, weakness, applicability and computation cost.

BACKGROUND

FPM had been the core research topic in the field of data mining for the last decade. Since, its inception with the seminal paper of mining association rules

by Agrawal *et al* (Agrawal & Srikant, 1994), it has matured enormously. Currently, we have very efficient algorithms for mining patterns with higher complexity, like sequence (Zaki, 2001), tree (Zaki, 2005) and graph (Yan & Han, 2002; Hasan, Chaoji, Salem & Zaki, 2005). The objective of FPM is as follows: Given a database D , of a collection of events (an event can be as simple as a set or as complex as a graph) and a user defined support threshold π^{\min} ; return all patterns (patterns can be set, tree, graph, etc. depending on D) that are frequent with respect to π^{\min} . Sometimes, additional constraints can be imposed besides the minimum support criteria. For details on FPM, see data mining textbooks (Han & Kamber, 2001).

FPM algorithms search for patterns in a combinatorial search space, which is generally very large. But, the *anti-monotone* property allows fast pruning: which states, “*If a pattern is frequent, so is all its sub-pattern; if a pattern is infrequent, so is all its super-pattern.*” Efficient data structure and algorithmic techniques on top of this basic principle enable FPM algorithms to work efficiently on database of millions events. However, the usage of frequent patterns in knowledge discovery tasks requires the analysts to set a reasonable support value for the mining algorithms to obtain interesting patterns, which, unfortunately, is not that straightforward. Experience suggests that if the support value is set too high, only few common-sense patterns are obtained. For example, if the database events are recent movies watched by different subscribers, using a very high support will only return the set of super-hit movies which are liked by anybody. On the other hand, setting low support value returns enormously large number of frequent patterns that are difficult to interpret; many of those are redundant too. So, ideally one would like to set the support value at a comparably lower threshold and then adopt a summarization or compression technique to obtain a smaller *FP*-set, comprising interesting, non-redundant, and representative patterns.

Figure 1: An itemset database of 6 transactions (left). Frequent, Maximal and Closed patterns mined from the dataset in 50% support (right)

Transaction Database, D		Frequent Patterns in D (Minimum Support = 3)			
		Support	Frequent Pattern	Maximal Pattern	Closed Pattern
1.	A C T W	6	C		C
2.	C D W	5	W, CW		CW
3.	ACTW	4	A, D, T, AC, AW, CD, CT, ACW		CD, CT, ACW
4.	ACDW	3	AT, DW, TW, ACT, ATW, CDW, CTW, ACTW	CDW, ACTW	CDW, ACTW
5.	ACDTW				
6.	CDT				

MAIN FOCUS

The earliest attempt to compress the FPM result-set was to mine Maximal Patterns (Bayardo, 1998). A frequent pattern is called maximal, if it is not a sub-pattern of any other frequent pattern (see the example in figure 1). Depending on the dataset, maximal pattern can reduce the result-set substantially; especially for dense dataset, the compression ratio can be very high. And, maximal patterns can be mined in the algorithmic framework of FPM processes without a post-processing step. The limitation of maximal pattern mining is that the compression also loses the support information of the non-maximal patterns; for example, in figure 1, from the list of maximal patterns we can deduce that the pattern *CD* is also frequent (since *CDW* is frequent), but its support value is lost (which is 4, instead of 3). Support information is critical if the patterns are used for rule generation, as it is essential for confidence computation. To circumvent that, Closed Frequent Pattern Mining was proposed by Zaki (Zaki, 2000). A pattern is called closed, if it has no super-pattern with the same support. The compressibility of closed frequent mining is smaller than the maximal pattern mining, but for the earlier, all frequent patterns and also, their support information can be immediately retrieved (without further scan of the database). Closed frequent pattern can also be mined within the FPM process.

Pattern compression offered by maximal or closed mining framework is not sufficient, as the result-set

size is still too large for human interpretation. So, many pattern summarization techniques have been proposed lately, each with different objective preference. It is difficult and sometimes, not fair, to compare them. For instance, in some cases, the algorithms try to preserve the support value of the frequent patterns; whereas, in other cases the support value is completely ignored and more emphasis is given in controlling the redundancy in patterns. In the following few paragraphs we discuss the main ideas of some of the major compression approaches, with their benefits and limitations. At the end of this section (see table 1), we show the benefits/limitations of these algorithms in tabular form for quick references.

Top-k Patterns

If the analyst has a predefined number of patterns in mind that (s)he wants to employ in the knowledge discovery tasks, *top-k patterns* is one of the best summarization technique. Han et al. (Han, Wang, Lu & TzVetkov, 2002) proposed one of the earliest algorithms that falls in this category. Their *top-k* patterns are *k* most frequent closed patterns with a user-specified minimum-length, *min_l*. Note that, minimum-length constraint is essential, since without it only length-1 patterns (or their corresponding closed super-pattern) will be reported, since they always have the highest frequency. The authors proposed efficient implementation of their proposed algorithm using FP-Tree; un-

fortunately, this framework works on itemset patterns only. Nevertheless, the summarization approach can be extended to other kind of patterns as well. Since, support is a criterion for choosing summary patterns; this technique is a support-aware summarization.

Another top- k summarization algorithm is proposed by Afrati et al. (Afrati, Gionis & Mannila, 2004). Pattern support is not a summarization criterion for this algorithm, rather high compressibility with maximal coverage is its primary goal. To achieve this, it reports maximal patterns and also, allows some false positive in the summary pattern set. A simple example from their paper is as follows: 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. Note that, this covers all the original sets, and there are only two false positive (BCD , $ABCD$). Afrati *et al.* showed that, the problem of finding the best approximation of a frequent itemsets that can be spanned by k set is NP-Hard. They also provided a greedy algorithm that guarantees an approximation ratio of at least $(1 - 1/e)$. The beauty of this algorithm is its high compressibility; authors reported that by using only 20 sets (7.5% of the maximal patterns), they could cover 70% of the total frequent set collection with only 10% false-positive. Also note, in such a high compression, there won't be much redundancy in the reported patterns; they, indeed, will be very different from each other. The drawbacks are: firstly, 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. Secondly, 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. Thirdly, it will not work if the application scenario does not allow false-positive.

Xin *et al.* (Xin, Cheng, Yan & Han, 2006) proposed another top- k algorithm. To be most effective, the algorithms strive for the set of patterns that collectively offer the best significance with minimal redundancy. Significance of a pattern is measured by a real value that encodes the degree of interestingness or usefulness of it and redundancy between a pair of patterns is measured by incorporating pattern similarity. Xin and his colleagues showed that to find such a set of top- k patterns is NP-Hard even for itemset, which

is the simplest kind of pattern. They also proposed a greedy algorithm that approximates the optimal solution with $O(\ln k)$ performance bound. The major drawback of this approach is that the users need to find all the frequent patterns and evaluate their significances before the process of finding top- k patterns is initiated. Again, for itemset pattern the distance calculation is very straightforward, this might be very costly for complex patterns.

Support-Preserving Pattern Compression

We already discuss ‘‘Closed pattern mining’’, which is one approach of support preserving pattern compression. However, there are other support-preserving pattern mining algorithms that apply to itemset only. One of the most elegant among these is *Non-Derivable Frequent Itemsets* (Calders & Goethals, 2007). The main idea is based on *inclusion-exclusion principle*, which enables us to find a lower bound and upper bound on the support of an itemset based on the support of its subsets. These bounds can be represented by a set of derivable rules, which can be used to derive the support of the super itemset from its sub itemset. If the support can be derived exactly, the itemset are called *derivable itemsets* and need not be listed explicitly. For example, for the database in Figure 1, the itemset CTW is derivable, because the following two rules hold:

$$\begin{aligned} \text{sup}(CTW) &\geq \text{sup}(TC) + \text{sup}(TW) - \text{sup}(T) \text{ and} \\ \text{sup}(CTW) &\leq \text{sup}(TW). \end{aligned}$$

From these, the support of CTW can be deduced exactly to be 3. Thus, a concise representation of frequent itemset consists of the collection of only *non-derivable itemsets*. Of course, the main limitation of this compression approach is that it applies only for itemset patterns; since, the inclusion-exclusion principle does not generalize for patterns with higher order structures. The compressibility of this approach is not that good either. In fact, Calders and Goethals proved that for some datasets, number of closed frequent patterns can be smaller in size than that of non-derivable frequent patterns. Moreover, there is no guarantee regarding redundancy in output set considering many of the frequent patterns are very similar to each other.

Cluster-Based Representative Pattern-Set

Pattern compression problem can be perceived as a clustering problem, if a suitable distance function between frequent patterns can be adopted. Then the summarization task is to obtain a set of representative patterns; each elements of the set resembles a cluster centroid. As typical clustering problem, this approach also leads to a formulation which is NP-Hard. So, sub-optimal solutions with known bounds are sought. Xin et al. (Xin, Han, Yan & Cheng, 2005) proposed greedy algorithms, named **RPglobal** and **RPlocal** to summarize itemset patterns. The distance function that they used considers both pattern object and its support. The representative element, P_r of a cluster containing elements, $P_1, P_2 \dots P_k$, satisfies

$$\bigcup_{i=1}^k P_i \subseteq P_r.$$

And the distance is computed in transaction space. For instance, if two patterns P_1 and P_2 occur in transaction $\{1, 3, 5\}$ and $\{1, 4, 5\}$, respectively, distance between them can be computed as,

$$1 - \frac{|\{1,3,5\} \cap \{1,4,5\}|}{|\{1,3,5\} \cup \{1,4,5\}|} = 0.5.$$

To realize a pattern cluster, a user defined δ is chosen as distance threshold. If a pattern has a distance less than δ from the representative, the pattern is said to be covered by the representative. **RPglobal** algorithm greedily selects the representative based on the remaining patterns to be covered. **RPlocal** finds the best cover set of a pattern by a sequential scan of the output set. For both the algorithms, the authors provided a bound with respect to the optimal number of representative patterns. For details, see the original paper (Xin, Han, Yan & Cheng, 2005). The attraction of cluster-based representative is that the algorithm is general and can easily be extended to different kinds of patterns. We just need to define a distance metric for that kind of pattern. The drawback is that user need to choose the right value of δ , which is not obvious.

Profile-Based Pattern Summarization

Yan *et al.* (Yan, Cheng, Han, & Xin, 2005) proposed profile-based pattern summarization, again, for item-set patterns. A profile-based summarization finds k representative patterns (named as Master patterns by Yan *et al.*) and builds a generative model under which the support of the remaining patterns can be easily recovered. A Master pattern is the union of a set of very similar patterns. To cover the span of the whole frequent patterns, a Master pattern is very different from another Master pattern. Note that, similarity considers both the pattern space and their support, so that they can be representative in the sense of FPM. 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 without consulting the original dataset. The definition of Pattern profile is as follows: Firstly, for any itemset α , let D_α be the transaction-set that contains α ; D be the entire dataset, and $\alpha_1, \alpha_2, \dots, \alpha_l$ be a set of similar patterns. Now, define

$$D' = \bigcup_{i=1}^l D_{\alpha_i}.$$

A profile M over $\alpha_1, \alpha_2, \dots, \alpha_l$ is a triple

$$\langle P, \phi, \rho \rangle,$$

where, P is a probability distribution vector of the items $\alpha_1, \alpha_2, \dots, \alpha_l$ learned from the set D' ,

$$\phi = \bigcup_{i=1}^l \alpha_i$$

is taken as the Master pattern of $\alpha_1, \alpha_2, \dots, \alpha_l$ and

$$\rho = \frac{|D'|}{|D|}$$

is regarded as the support of the profile. For example, if itemset $CT (D_{CT} = \{1,3,5,6\})$ and $CW (D_{CW} = \{1,2,3,4,5\})$ of table 1 are to be merged in a profile, the Master pattern is CTW , its support is

$$\frac{|D'|}{|D|} = \frac{6}{6} = 100\%$$

(note that, original pattern CTW has a support of 83% and $P = (4/6, 5/6)$). Profile based summarization is elegant due to its use of probabilistic generative model, but there are drawbacks. Authors did not proof any bound on the quality of results, so depending on the dataset, the summarization quality can vary arbitrarily. It is a post-processing algorithm, so all frequent patterns need to be obtained first. Finally, clustering is a sub-task of their algorithm; since, clustering itself is an NP-Hard problem, any solution adopting clustering algorithms, like k-means, hierarchical and etc. can only generate a local optimal solution.

Very recently, Wang and Parthasarathy (Wang & Parthasarathy, 2006) used probabilistic profile to summarize frequent itemsets. They used Markov Random Field (MRF) to model the underlying distribution that generates the frequent patterns. To make the model more concise, the authors considered only the non-derivable frequent itemsets instead of the whole set. Once the model is built, the list of frequent itemsets and associated count can be recovered using standard probabilistic inference methods. Probabilistic methods provide the most compact representation with very rough support estimation.

FUTURE TRENDS

Despite the numerous efforts in recent years, pattern summarization is not yet totally solved. Firstly, all the summarization algorithms were mainly designed to compress only the itemset patterns. But, unfortunately the problem of large output size of FPM algorithms is more severe for patterns, like trees, graphs, etc. Out of the existing algorithms, very few can be generalized to these kinds of patterns, although no attempt has been reported yet. Very recently, Hasan *et al.* proposed an algorithm, called ORIGAMI (Hasan, Chaoji, Salem & Zaki, 2007), which summarize graph patterns by following the idea of representative patterns. Since, finding representative patterns is NP-Hard, they adopted a local optimal algorithm to obtain the representatives. Their algorithm is also a post-processing one; but, they avoided the generation of entire frequent pattern set by finding a smaller collection of maximal patterns through a random walk along the frequent graph lattice. The walk terminates when a user chosen convergence condition is satisfied. This work is very different from all the previous works and it directs the future trends of pattern

summarization. Since, generating entire pattern-set is not feasible, an ideal pattern summarization algorithm should avoid the post processing all-together and should find the representative patterns in an online fashion as the patterns are being generated. Some streaming data model can be adopted to solve this kind of problem. This data model also has the added benefit that all pairwise distances between the frequent patterns need not be computed, which, sometimes, is very costly.

CONCLUSION

Pattern summarization is a fascinating research direction with numerous attentions in recent years. To circumvent the problem of enormous output size of FPM algorithms that impede its application, summarization offers an elegant solution. With the invention of new summarization algorithms, many new concepts regarding pattern summarization has also emerged, like similarity, redundancy, significance and etc. These concepts enable to formalize the summarization techniques and many of these, also, provide criteria to evaluate the quality of summarization. This paper provides a gentle overview of different summarization algorithms and these concepts.

REFERENCES

- Afrati F., Gionis. A., & Mannila H. (2004). Approximating a Collection of Frequent Sets, *Proceedings of the 10th ACM SIGKDD international conference of Knowledge discovery and Data mining*, 12-19
- Agrawal R. & Srikant R. (1994). Fast Algorithms for Mining Association Rules in Large Databases, *Proceedings of the 20th International Conference on Very Large Data Bases*, 487-499
- Bayardo, R (1998). Efficiently mining long patterns from databases, *In the Proceedings of ACM SIGMOD Conference*.
- Calders T., & Goethals, B. (2007). Non-Derivable Itemset Mining, *Data mining and Knowledge Discovery*, 14(1), 171-206.
- Han, J. & Kamber, M. (2001). Data Mining: Concepts and Techniques, 2nd Edition, *Morgan Kaufmann Publications*.

Han, J., Wang, J., Lu, Y. & Tzvetkov, P. (2002). Mining Top-K Frequent Closed Patterns without Minimum Support, *Proceedings of IEE International Conference of Data Mining*, 211-218.

Hasan, M., Chaoji, V., Salem, S., & Zaki, M. (2005). DMTL: A Generic Data Mining Template Library, in *Workshop on Library-Centric Software Design (LCSD '05), with OOPSLA'05 conference, 2005*.

Hasan, M., Chaoji, V., Salem, S., & Zaki, M. (2007). ORIGAMI: Mining Representative Orthogonal Graph Patterns, *Proceedings of IEE International Conference of Data Mining*, 153-162.

Pei J., Dong, G, Zou W, & Han J. (2002). On Computing Condensed Frequent Pattern Bases, *Proceedings of IEE International Conference of Data Mining*, 378-385.

Wang C., & Parthasarathy S. (2006). Summarizing Itemset Patterns Using Probabilistic Models, *Proceedings of the 12th ACM SIGKDD international conference of Knowledge discovery and Data mining*, 730-735.

Xin D., Han J., Yan X., & Cheng H. (2005) Mining Compressed Frequent-Pattern Sets. *Proceedings of the 31st international conference on Very large data Bases*, 709-720.

Xin D., Cheng H., Yan X., & Han J. (2006). Extracting Redundancy-Aware Top-K Patterns, *Proceedings of the 12th ACM SIGKDD international conference of Knowledge discovery and Data mining*, 444-453.

Yan, X. & Han. J. (2002). gSpan: Graph-Based Substructure Pattern Mining, *Proceedings of IEEE International Conference of Data Mining*, 721-724.

Yan X., Cheng H., Han J., & Xin D. (2005). Summarizing Itemset Patterns: A Profile-Based Approach, *Proceedings of the 11th ACM SIGKDD international conference of Knowledge discovery and Data mining*, 314-323.

Zaki, M (2000). Generating Non-Redundant Association Rules, *Proceedings of 6th International Conference of Knowledge discovery and Data mining*, 34-43.

Zaki, M. (2005). Efficiently Mining Frequent Trees in a Forest: Algorithms and Applications, *IEEE Transaction on Knowledge and Data Engineering*, 17(8), 1021-1035.

Zaki, M. (2001). SPADE: An Efficient Algorithm for Mining Frequent Sequences, *Proceedings of Machine Learning Journal, Special Issue on Unsupervised Learning* 42(1/2), 31-60.

KEY TERMS

Closed Pattern: A pattern α is called closed if no pattern β exists, such that $\alpha \subset \beta$ and $transaction_set(\alpha) = transaction_set(\beta)$. It is a lossless pattern summarization technique.

Maximal Pattern: A pattern α is called maximal pattern, if no pattern β exists, such that $\alpha \subset \beta$ and β is frequent. It is a lossy summarization technique.

Non-Derivable Pattern: A pattern α is called derivable if its support can be exactly inferred from the support of its sub-patterns based on the inclusion-exclusion principle. Since, inclusion-exclusion principle works only for set, the non-derivable definitions is also applicable only for itemset pattern.

Pattern Profile: Ideally, pattern profile is a compressed representation of a set of very similar patterns, from where the original patterns and their support can be recovered. However, profiles are built using some probabilistic approaches, so the compression is always lossy; therefore, the recovered pattern and their support are usually approximations.

Pattern Significance: Pattern significance measures the importance of a specific pattern in respect to the application domain. For example, in case of itemset pattern which are, usually use to obtain association rule, the pattern significance can be measured in terms of the corresponding rule interestingness. For classification or clustering task, significance of a pattern is the effectiveness of that pattern as a feature to discriminate among different cluster or classes.

Pattern Similarity: Pattern similarity measures the closeness between two patterns. If α and β are two patterns, similarity can be computed as below:

$$sim(\alpha, \beta) = \frac{|\alpha \cap \beta|}{|\alpha \cup \beta|}$$

Table 1. Comparison of different summarization algorithms, features that have an (\uparrow) symbol are desirable and those that have (\downarrow) are not desirable

Summarization Algorithms	Feature lists							
	Compress ratio \uparrow	Post processing \downarrow	Support preserving \uparrow	Support aware \uparrow	Pattern Redundancy \downarrow	Fault tolerant \downarrow	Coverage considered \uparrow	Generalized \uparrow
Maximal	Moderate	No	No	Yes	High	No	Yes	Yes
Closed	Moderate	No	Yes	Yes	Very High	No	Yes	Yes
Top-k with min length	User defined	No	No	Yes	Low	No	No	Yes
Fault-tolerant Top-k	User defined	Yes	No	No	Very Low	Yes	Yes	No
Redundancy-aware top-k	User defined	Yes	No	No	Very Low	No	Yes	Yes
Non-derivable	Moderate	No	Yes	Yes	High	No	Yes	No
Cluster based representatives	User defined	Yes	No	Yes	Low	No	Yes	Yes
Pattern profile	High	Yes	No	Yes	Very Low	Yes	No	No
Probabilistic profile	High	Yes	No	No	Very Low	Yes	No	No

The set union and intersection in the above definition can be extended naturally for complex pattern like, tree or graph. For instance, for graph pattern, intersection between two patterns can be the maximal common sub-graph between them. Generally, pattern similarity ranges from $[0, 1]$. Pattern distance is computed as $1 - \text{Pattern Similarity}$. Distance obeys the metric properties, like, $\text{distance}(\alpha, \alpha) = 0$ and $\text{distance}(\alpha, \beta) + \text{distance}(\beta, \gamma) \geq \text{distance}(\alpha, \gamma)$.

Pattern Summarization: FPM algorithms produce large results which is difficult for the end user to interpret. To circumvent the problem, frequent patterns are compressed to a smaller set, where the patterns are non-redundant, discriminative and representative. Effective summarizations are, generally, lossy *i.e.* the support information of the compressed patterns can not be recovered exactly. Different flavors of summarization techniques exist. Pattern profile is one of the summarization techniques.