The Hunt of Best Algorithm for Event Sequence Data

A comparison Essay

1 Abstract

This Essay is written for the comparison of the following papers.
The Long and the Short of It: Summarizing Event Sequences with Serial Episodes (2012). [1]
Keeping it Short and Simple: Summarizing Complex Event Sequences with Multivariate Patterns (2016). [2]
In both papers, authors address the problem of mining interesting patterns from Event sequence data.
In this essay, we will compare the difference between covering a string, Experiments, and the results of these experiments.

2 Introduction

Both papers have the implementation of the algorithms based on the MDL principle [3].
Both papers take pattern set mining approach to solve the problem of mining interesting patterns for event sequence data set. Although paper [2] deals with the multivariate data, the procedure followed is the same as the paper [1]. Both papers make use of some key concepts of data mining. In this section, we discuss the Common key concepts used in both the papers [1] and [2].
Pattern mining: Pattern mining is the branch of data mining where we focus on finding important clusters of patterns that occur together in a data sequence. The ideal output of pattern mining is a small subset of data that gives us a brief idea about the whole data. In this process of pattern mining, we also want to get rid of non-useful data usually referred as noise.
Event sequence data: Event sequence type of data is where multiple events follow each other. Data coming from sensor arrays can be a good example of this. Sensor arrays are constantly reporting the data to the main processing unit and with a large amount of data, it is important to summarize it. But creating a good summary can be a challenging task. Due to highly redundant patterns usual mining techniques soon end up having a ”pattern explosion” problem as discussed in the paper[1]. Algorithms in Paper [1] are mainly focused on univariate data. While paper [2] is about a multivariate data set.
Pattern explosion: Pattern explosion is a common problem whenever we have a very large set of data. The larger data set has too many reoccurring patterns. And it becomes very challenging to summarize the data once it shows overly redundant

patterns. Hence the frequency of patterns cannot be a good measure to summarize the data.

MDL: Both the papers use the MDL principle. The Minimum Description Length principle is a way of finding out the best model for compressing the data set. Repeating patterns in the data can be used to compress its size. Compression is directly proportional to the regularity in the data set. The final goal is to reduce

$$L(M) + L(D|M)$$

Where [L (M)] is the length of the description of model M, [L (D | M)] is the length of the description of the data given model M.

3 Inspiration of the papers

Authors of both the papers have mentioned all the related work done in this area of event sequence mining.

Both SQS algorithms proposed and the DITTO algorithm draw the inspiration from the KRIMP [4] and SLIM [5] algorithms.

KRIMP uses MDL in the best way possible to identify interesting pattern sets. With SQS algorithms authors propose a method that is similar to SLIM [5], but it considers joining patterns.

As there was no solid method to summarize the multivariate data till 2016, authors of paper [2] propose DITTO algorithm.

4 Differences

Brief comparison: In the paper [1] authors Nikolaj Tatti and Jilles Vreeken discuss implementing algorithms based on the MDL principle which will summarize the single variate data well. Two algorithms are discussed in this paper named SQS search and SQS candidates. These two algorithms are based on the covering only single variate event sequence data. In the paper [2] authors Roel Bertens, Jilles Vreeken and Arno Siebes solve the problem of finding interesting and important patterns from a given multivariate data set. In this paper, the algorithm proposed is called the DITTO algorithm. This algorithm is also for the event sequence data type but in this case, we are considering a multivariate data set.

4.1 Similarities in characteristics of algorithms –

The key idea authors of papers [1] and [2] have in mind is to find models that generalize the data.

Hence these models need to have some kind of numeric value or score which increments if the model finds real good structure or decrements if the model comes across redundancy or noise.

Both papers take the pattern set mining approach.

Hence to help with the score for the model, authors have used MDL.

4.2 Difference between covering a string

To explain and compare the covering process in the best way possible we will compare the reconstruction of the original data set from given column Cp from the code table.
As figure 1 shows covering a single variate event sequence string is fairly simple

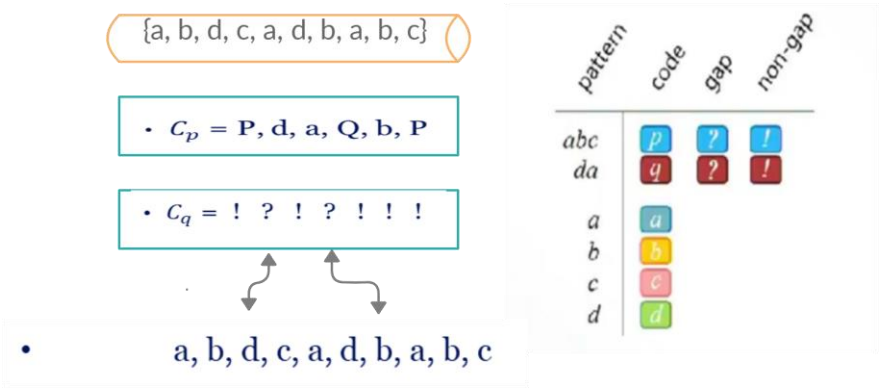

Fig.1. covering a single variate event sequence string

We consider two patterns in this particular example ”a, b, c” as p and pattern ”d, a” as q.
As we can see we get **Cp as P, d, a, Q, b, P** and **Cg = ! ? ! ? ! ! !**
To reconstruct our original data set we start with ’p’ as a is the first element we can safely put a in the reconstructed data set.
Next we check Cg we have no gap hence we put ’b’.
Next we check cg again now we have a gap denoted by ’?’, we check Cp again it has ’d’ hence we put ’d’ in between ’b’ and ’c’.
In the similar way we can reconstruct the full data set and follow same process for pattern ’q’.

Figure 2 explains the covering algorithm for the multivariate event sequence strings. Here we consider two strings having sequence of events.
We have Cp as a,Y,X,a,Y,a and Cg = gap, gap, no gap, no gap, no gap. On the first line - For reconstruction of our original data set we start with the singleton ’a’ from Cp On the second line - Next we have pattern ’Y’ which is c and e with gap. We put these two events on the second line with a gap. On the first line - next we have is pattern ’X’ which is a,b with gap and d on the second line. We put a, b with gap on the first line and put d in the gap of previous pattern. On the first line - Next we have again singleton ’a’. On the second line - Now we have pattern Y again with no

gap hence we can put both the events together on the second line. On the first line - last occurrence of singleton 'a' is added to the first line.

Cover 2.png

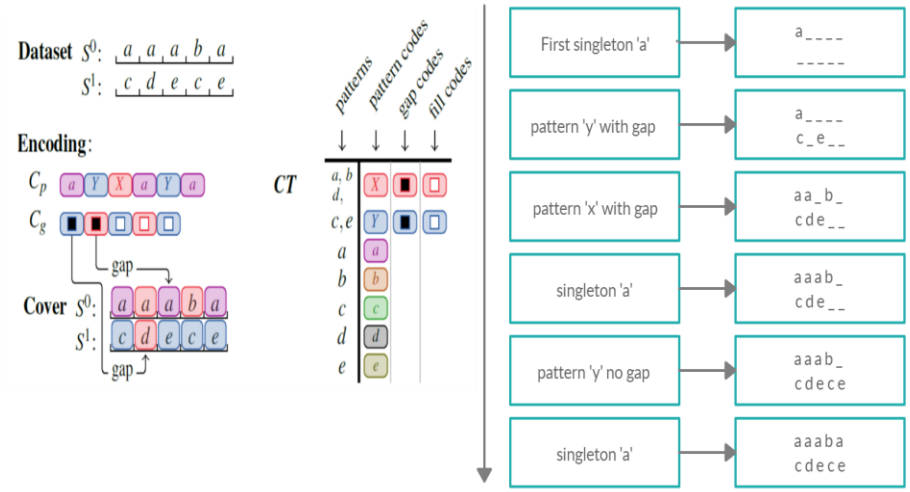

Fig.2. covering a Multi variate event sequence string

Both of the above methods show us way of covering event sequence strings, but the first type discussed in paper [1] can be applied only to the single variate event sequence string.

Example of such a data set can be seen in figure 1. However type 2 allows us to cover multivariate event sequence strings. Figure 2 shows us one of the possibility for this where we consider two sequences. Even though the data types vary in both the examples the basic idea of considering gaps while encoding or reconstructing the data set remains almost identical. Type 2 of covering string mentioned in paper [2] is more complex than method one. But this also allows to cover multiple possibilities.

4.3 How MDL changes?

Given above methods above which have encoding of gaps the MDL principle needs to consider the gaps while calculating final size. For calculating pattern size of the encoded stream

$$L(Cp|CT) = \sum_{X \epsilon CT} usage(X) L(code_p(X))$$

For calculating size the of encoded gaps

$$L(Cg|CT) = \sum_{X \epsilon CT_{|X|>1}} gaps(X)L(code_g(X)) + fills(X)L(code_n(X))$$

Together combining these two we can sum up the equation as follows

$$L(D|CT) = L_N(|D|) + \sum_{S \epsilon D} L(|S|) + L(C_p|CT) + L(C_g|CT)$$

This is the equation used to calculate the compression score in both papers [1] and [2].

4.4 DITTO algorithm

DITTO is very similar to SQS search. It also starts with singletons. DITTO greedily considers the eligible pattern sets in the fixed order which is size dependent. **As DITTO deals with multivariate data there is also possibility of combining two pattern sets. By changing the combination of the gaps and the patterns itself, DITTO is eligible of creating different alignments of eligible combination of patterns.**

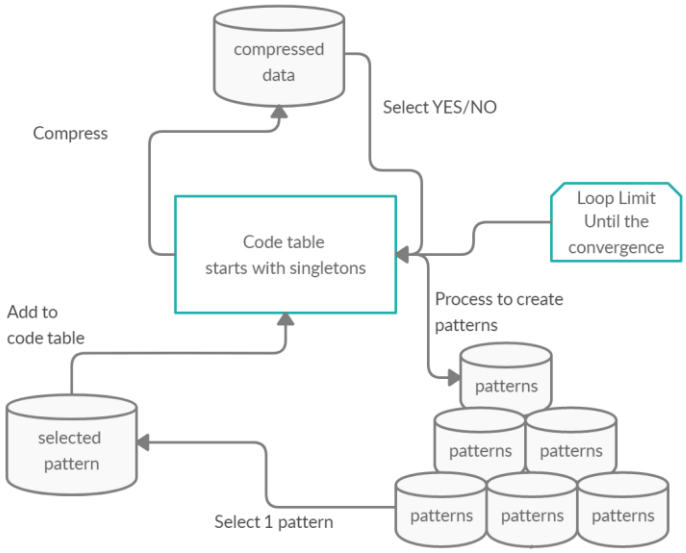

Fig.3. SQS search and DITTO algorithm flowchart

4.5 SQS search

SQS search has no particular need of pre mined patterns. **It simply starts with the singleton events and combines the event occurring together checks the MDL score**. If the pattern created helps in the compression gain then SQS search keeps it or else discards it. Multiple combinations are tried to achieve the best gain in compression. This process is followed until the convergence.

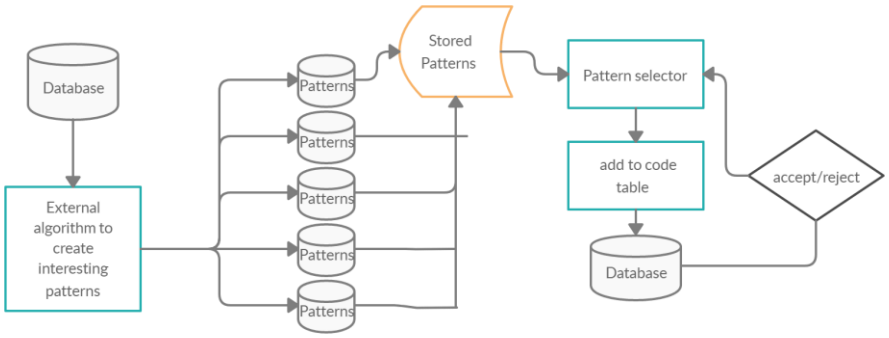

Fig.4. SQS candidates algorithm flowchart

4.6 SQS candidates

In paper [1] the authors propose SQS candidates algorithm which starts with the **pre mined patterns. These patterns are usually made with the help of external algorithm.** We can also provide patterns in which we are interested as an input to this algorithm. First the patterns are sorted and a pattern is selected to check the score.

5 Experiments

Experiments in both papers were done using C++. Out of the three algorithms SQS candidates requires pre mined patterns. For providing these pre mined patterns authors of paper [1] used disjoint minimal windows of maximal length 15, with minimal support thresholds as low as possible. Both papers consider synthetic and real world data. Data sets used in the papers:

5.1 Synthetic Data

Both papers [1] and [2] consider synthetic data sets. In paper [1] the authors consider the synthetic Indep, Plants10, and Plants50 data sets. Each consists of a single sequence of 10 000 events over an alphabet of 1000. In Indep, events considered are independent events. But in Plants10 and Plants50 authors have put in 10 and 50 patterns of 5 events 10 times each respectively, with low probability of having a gap between two consecutive events. However in paper [2] the authors create random data in which they put multiple randomly generated patterns of multiple independent properties. Number of experiments were performed by the authors by changing number of events, attributes and alphabet size, and for the created patterns number frequency and size.

5.2 Real world Data

For the experiments on real data, in order to interpret the patterns, authors of paper [1] consider text data. Stem or the base of every word is considered as an event in this case and basic preprocessing like removal of stop words was done for this text data. JMLR - The journal of the machine learning research papers and other research Pres Addresses - Data from news sources from year 2012 USA. Moby dataset is from the novel 'Moby Dick' written by Herman Melville. While authors of paper [2] also make use of the 'Moby Dick' dataset they also consider an ECG sensor dataset, a structural integrity sensor data from a Dutch bridge, and data European Parliament resolution which is multilingual in structure.

6 Summary of the comparison

- With SQS-CANDIDATES authors of paper [1] **allow the user to give custom set of patterns as an input. This freedom is only available with SQS-candidates.** Lowering the thresholds creates possibility of more eligible patterns, increased number of eligible patterns result in larger search space, and final result is better model of SQS candidates.
  - SEARCH and DITTO other hand are parameter-free algorithms as they generate and test patterns and can result in a better score. These algorithms are more suited when we don't have much information about our dataset.

- In paper [2] the authors have compared SQS and DITTO algorithm in the result section. Comparing these results authors found that DITTO recovers all created patterns. Because **of the limitations of working with univariate data, SQS algorithms recovered very small fraction** of the pattern sets.

- **SQS algorithms which are specifically for univariate data sets work faster than the DITTO.**

- 'Moby Dick' was the common data set tested in both the papers**. Compression gain achieved by SQS algorithms was 1.7% while DITTO achieved compression gain of 14.7%**

- We can summarize the Key properties of algorithms as follows

	SQS candidates	SQS search	DITTO
Need of Pre-mined patterns	YES	NO	NO
Custom input of patterns	YES	NO	NO
Datatype Required	single variate	Single variate	Multivariate or Single variate
Dependency on external algorithm	YES	NO	NO
Time complexity	4 min - External algorithm 15 min - order filter candidates	JMLR - 10 min PRES - 18 min Moby - 91 min	Usually more than SQS
Gain in compression	Depends on support threshold but usually least	Moderate gain	Best gain compared to other 2

All the images in this essay are created using www.creately.com
To create flowcharts C++ codes for SQS were downloaded.

References

1. Nikolaj Tatti and Jilles Vreeken, "The long and the short of it: Summarising eventsequences with serial episodes," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2012, KDD '12, pp. 462–470, ACM.

2. Roel Bertens, Jilles Vreeken, and Arno Siebes, "Keeping it short and simple: Summarising complex event sequences with multivariate patterns," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, 2016, KDD '16, pp. 735–744, ACM.
3. Peter Gru¨nwald, *The Minimum Description Length Principle*, 01 2007.
4. Jilles Vreeken, Matthijs van Leeuwen, and Arno Siebes, "Krimp: mining itemsetsthat compress," *Data Mining and Knowledge Discovery*, vol. 23, no. 1, pp. 169–214, Jul 2011.
5. Koen Smets Jilles Vreeken, "Slim: Directly mining descriptive patterns," 01 2007.