Stream Data Processing

Jakub ŠEVCECH*

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
jakub.sevcech@stuba.sk

When processing streams of data, we often encounter data composed of repeating sequences of similar values. Such repeating patterns are present for example in highly seasonal data such as counting metrics running on production or consumption data. As these repeating patterns sketch the outline of the data, we could use them as an approximate representation of the raw data [4]. Such representation could be used for significant dimensionality reduction and could allow application of methods and algorithms not directly applicable on the original form of the data.

However, if we transform evolving data streams into sequence of repeating patterns, the number of patterns could grow indefinitely as more and more data is transformed. This could lead to a rapid growth of the transformed data size (disrupting the effort for dimensionality reduction) and to inability to apply various algorithms relying on limited size of the processed alphabet. In such cases, we would need methods for reduction of the alphabet and to guarantee the maximal size of the alphabet.

In our work, we focus on methods for alphabet size control for Incremental Subsequence Clustering (ISC) [4] time series representation, we proposed in our previous work. However, we propose an alphabet size control approach to be applicable to other methods using subsequence clustering or repeating patterns for data representation as well.

The time series representation, we build on, transforms time series incrementally into a sequence of shapes. Shapes are formed as clusters of subsequences of defined length and overlap. These clusters are defined by an identifier, a time series subsequence in its centre and a limit distance, forming a border for other subsequences to be associated with the cluster.

We proposed the alphabet size control approach as a result of a sequence of improvements of a single basic idea of removing symbols from the alphabet. We only forget old and rarely used symbols to minimize the impact of their loss on reconstruction error. Forgetting of any symbols results in the inability to restore the original form of the time series from the transformed representation. However, since in general in stream

[*] Supervisor: Mária Bieliková, Institute of Informatics, Information Systems and Software Engineering

Spring 2016 PeWe Workshop, April 2, 2016, pp. 27–28.

data processing we are often most interested in recent parts of the data streams, this can be an acceptable limitation in some domains.

By introducing the symbol occurrence frequency into the alphabet management process, we introduce the requirement for a method for frequent symbol mining. We propose the alphabet size control approaches to be straightforwardly adaptable to any frequent item mining method used as symbol occurrence count estimation. An arbitrary counter based and most sketch based [2] frequent item mining methods could be used, but for the purpose of our experiments, we use the Frequent algorithm [3] due to its simplicity and good results even when relatively small number of counters is used to find frequent items.

To evaluate the effectiveness of the proposed alphabet reduction scheme, we use electricity consumption dataset [1] from Belgian electricity transmission operator. The dataset comprises grid load values sampled in 15 minutes intervals between years 2005 and 2015 forming a time series of more than 370 000 data points. We chose this dataset as representative for very long time series, where multiple seasonal effects are present (daily, weekly and yearly) and where multiple frequent patterns are repeating.

The proposed approach is able to maintain stable size of the alphabet and acceptable reconstruction error, but it also has several limitations. The most important one is slow response to changes in the processed data. As the evaluated approach was not able to immediately replace old symbols with newly appearing symbols, in the transitional part of the dataset. An algorithm using some kind of decay would be appreciated for the symbol count estimation as it might be faster to respond to significant changes in the course of the transformed data. As we proposed this approach to be independent from the method used for frequent item mining, as long as it uses a set of counters, this change should be straightforward.

Extended version was published in Proc. of the 12th Student Research Conference in Informatics and Information Technologies (IIT.SRC 2016), STU Bratislava.

Acknowledgement. This work was partially supported by the Scientific Grant Agency of Slovak Republic, grant No. VG 1/0774/16.

References

[1] Elia - Grid data download. http://www.elia.be/en/grid-data/data-download, 2015, Accessed: 2015-09-07.

[2] Cormode, G., Hadjieleftheriou, M.: Finding frequent items in data streams. In: Proceedings of the VLDB Endowment, 2008, pp. 1530-1541.

[3] Misra, J., Gries, D.: Finding repeated elements. Science of computer programming, 1982, vol. 2, no. 2, pp. 143-152.

[4] Sevcech, J., Bielikova, M.: Symbolic Time Series Representation for Stream Data Processing. In: Trustcom/BigDataSE/ISPA, 2015 IEEE. Volume 2., IEEE, 2015, pp. 217-222.