Neurocomputing 203 (2016) 22–33

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Event detection from traffic tensors: A hybrid model

Hadi Fanaee-T [a,*], João Gama [b]

[a] Laboratory of Artificial Intelligence and Decision Support, FCUP/University of Porto, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal
[b] Laboratory of Artificial Intelligence and Decision Support, FEP/University of Porto, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal

ARTICLE INFO

Article history:
Received 31 December 2014
Received in revised form
24 February 2016
Accepted 17 April 2016
Communicated by Jianbin Qiu
Available online 6 May 2016

Keywords:
Traffic data
Origin/destination matrix
Tensor decomposition
Tucker
Core size

ABSTRACT

A traffic tensor or simply *origin* × *destination* × *time* is a new data model for conventional origin/destination (O/D) matrices. Tensor models are traffic data analysis techniques which use this new data model to improve performance. Tensors outperform other models because both temporal and spatial fluctuations of traffic patterns are simultaneously taken into account, obtaining results that follow a more natural pattern. Three major types of fluctuations can occur in traffic tensors: mutations to the overall traffic flows, alterations to the network topology and chaotic behaviors. How can we detect events in a system that is faced with all types of fluctuations during its life cycle? Our initial studies reveal that the current design of tensor models face some difficulties in dealing with such a realistic scenario. We propose a new hybrid tensor model called HTM that enhances the detection ability of tensor models by using a parallel tracking technique on the traffic's topology. However, tensor decomposition techniques such as Tucker, a key step for tensor models, require a complicated parameter that not only is difficult to choose but also affects the model's quality. We address this problem examining a recent technique called adjustable core size Tucker decomposition (ACS-Tucker). Experiments on simulated and real-world data sets from different domains versus several techniques indicate that the proposed model is effective and robust, therefore it constitutes a viable alternative for analysis of the traffic tensors.

1. Introduction

The understanding and characterization of traffic tensors has many applications in network information systems, transportation systems and many other areas. In particular, event detection in these systems enables operators to make better decisions about emerging problems and perform some prevention tasks. Some intuitive examples include the identification of attacks and malicious activities inside networks and traffic jams in transportation systems. However, one of the most serious problems in event detection from traffic tensors is the complexity and diversity of event types. Fig. 1 illustrates a simplified scenario of a hypothetical bike sharing system during the operational period of 100 days. The stations are specified by letters A to F and the connections between them are represented by directed lines. For extra simplicity let us assume that the system shows a normal behavior with stable traffic among four stations of A, B, C and D and no traffic from nodes E and F until day 95. Let us also presume that events take place in the system exclusively during days 95–100.

The first event type occurs on days 95 and 96 when the system experiences an increase in traffic flow throughout the network. As we can see, a constant amount is added to the volume of traffic between all stations. This is similar to what happens in a large impact citywide event such as a big festival which affects the city's whole population. For this example, $t=95$ relates to a severe event and $t=96$ to a low-scale event. Note that the alteration in traffic volume is not necessarily additive. Weather-related events such as rain lead to the same patterns, but in a subtractive form. For instance, in a normal working day, heavy rain may remarkably reduce the requests for bike rental.

The next event type occurs during days 97 and 98 when some new connections are established between stations in a part of the network. As we see, traffic between main stations remains unchanged as the system behaves normally, but moderate traffic shows up from stations E and F to C. This kind of event can appear due to operational changes or occurrence of some regional events. For instance, let us imagine a scenario in which stations E and F are out of service for a long period and suddenly become available. Or in another possible scenario, if users do not find available bikes in station B they may refer to the closer stations E and F. Some local events such as a sports rivalry can also account for this type of events. We know that during a football rivalry, people come from different regions to the event's location. Hence, many rare links with zero traffic might be connected by these users. For instance, a user who lives far away from the stadium may establish a new

* Corresponding author.
E-mail address: hadi.fanaee@fe.up.pt (H. Fanaee-T).

http://dx.doi.org/10.1016/j.neucom.2016.04.006
0925-2312/

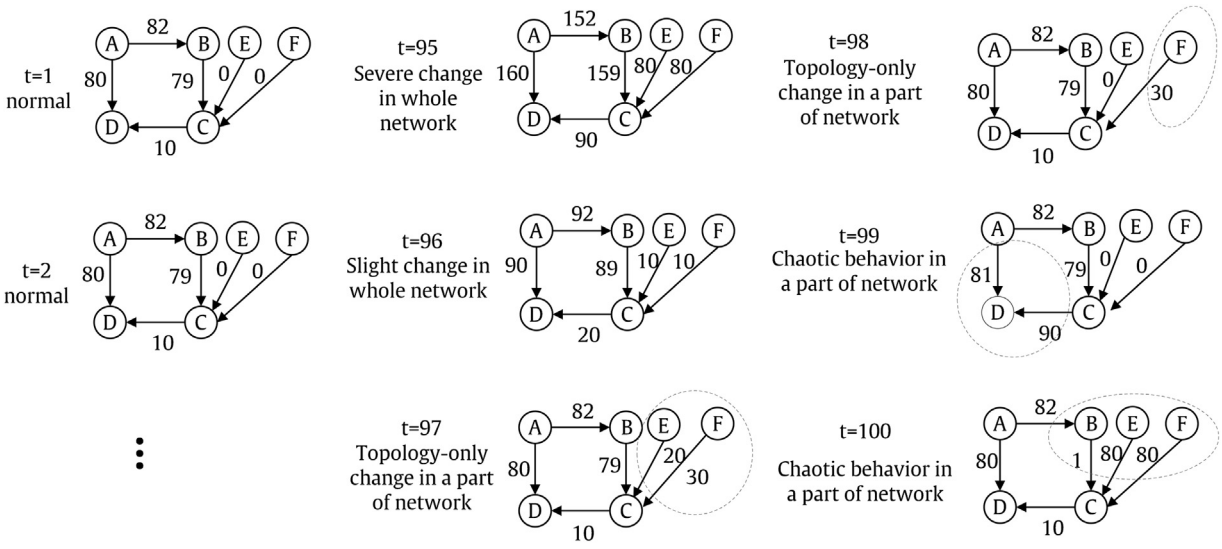

Fig. 1. Motivational example: a simplified hypothetical scenario in bike-sharing network. Between $t=1$ and 94 the system has a stable behavior. At $t=95$–96 a mutation occurs through whole network. In $t=97$ and 98 topology of traffic changes in a part of the network. During $t=99$–100 a chaotic behavior shows up in a part of the network. How can we accurately detect these events via a unified model?.

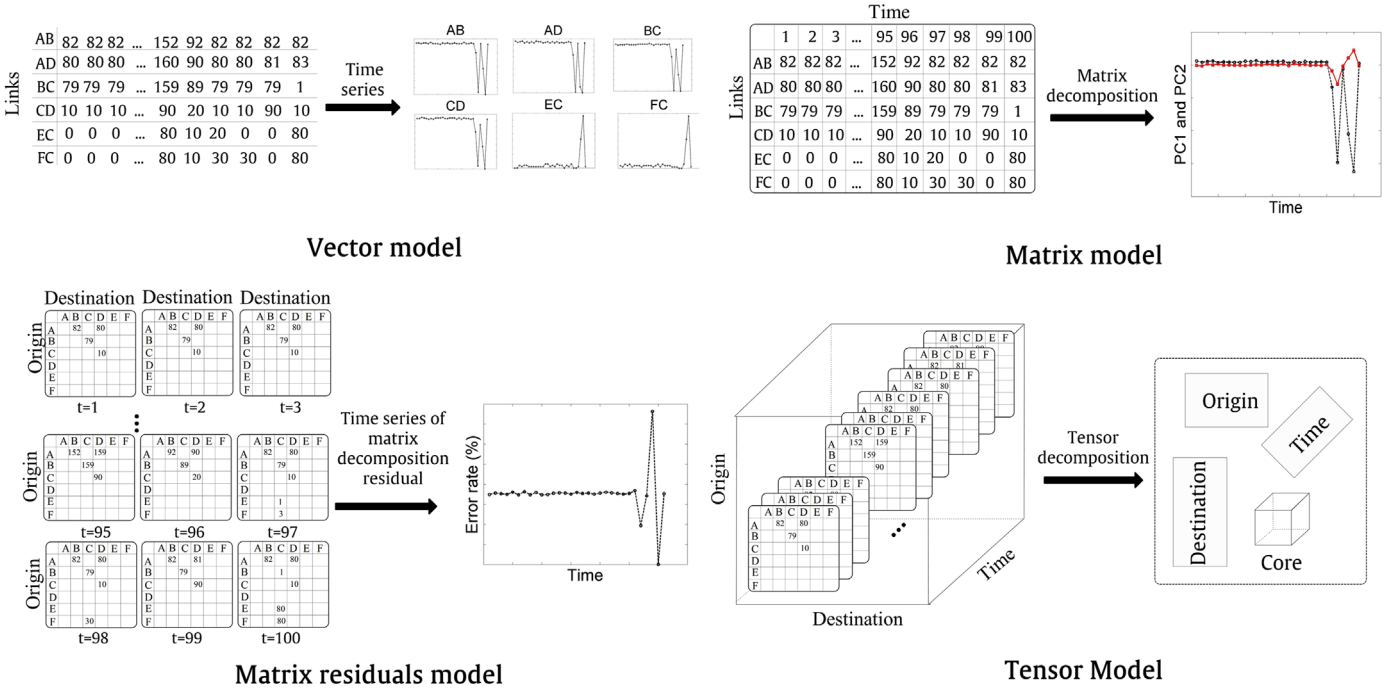

Fig. 2. Four major traffic data analysis techniques. Numbers in the figure are derived from the example scenario in Fig. 1.

connection from a station close to his neighborhood to the stations close to the stadium's location.

Finally, during the last two days, i.e. $t=99$, 100 we observe a chaotic behavior in a part of the network where events do not follow any regular pattern. For instance, even though the traffic pattern at $t=99$ seems similar to the expected for stable conditions, links A to D and C to D exhibit an odd behavior. In one of them we observe a very slight change while in the other one the fluctuation is very intense. Likewise, in $t=100$ we perceive a mutation in the network topology as well as an irregular inconstancy in the flow volume. The reason for this type of events is not evident, because multiple factors are usually involved in their occurrence. An intuitive example of such a chaotic behavior can be an occurrence of various events at the same time such as a football rivalry, alongside a blackout event in some stations added to a weather-based event like rain.

As we can notice in the above cases, some events are associated with fluctuations in flow (e.g. days 95 and 96), while some are linked with alterations to the network topology (e.g. days 97 and 98) and others, such as 99 and 100 to a chaotic behavior in part of the network. This kind of patterns may be repeated several times and are not necessarily limited to a specific type. In a practical manner, most of these patterns take place in the system during its life cycle. The question is how to construct a model that simultaneously detects all these types of events. The answer to this question is the matter of this research and will be discussed in the following, but before that let us briefly review the existing solutions for this problem.

Traffic data analysis is a commonly studied area in network information systems and transportation systems, many methods have been developed in these two domains. The techniques, as illustrated in Fig. 2 depend on the data's structure and the analyzed components are classified into four major models.

Vector models are the most basic solutions for analyzing traffic data. For this kind of approach we generate a flow time series for each link (or O/D pair) and then apply a time series method such as Autoregressive Integrated Moving Average (ARIMA) [1] or a regression model [2]. The application of these techniques is limited, because they do not take into account the correlation among links, hence it is not ideal for handling noises and missing values in data.

Alternatively matrix models attempt to model all links simultaneously and, therefore do not face the problem of vector models. With these methods [3,4], a matrix is constructed from the complete set of links, rows are links and its columns are time instants (see Fig. 2). The next step is to apply a matrix decomposition solution such as Principal Component Analysis (PCA) [5] on the link matrix. PCA is able to interpret data in terms of a small number of independent variables (or components) which consequently enables us to identify anomalies and irregular patterns. Although PCA provides a better model quality against vector-based methods, it suffers from two major issues. To begin with, regular PCA performs poorly on traffic matrices that are polluted by the large volume of outliers [4]. Secondly, PCA relaxes the natural three-dimensional structure of data into a bi-dimensional form, hence, it is not able to capture existing spatio-temporal fluctuations in traffic data.

There is another class of matrix-based methods which rely on Singular value decomposition (SVD) [6]. In [7] authors propose a new approach based on SVD that tracks the angle of the dominant left singular vector along with the principal eigenvalue over time, aiming at the detection of faults in a simulated web traffic data set. Although SVD-based methods seem fitting for anomaly detection, in contrast to PCA, are not applicable for modeling and forecasting.

There exists a new branch of matrix-based methods which we will name matrix residual models to differentiate from previous matrix-based models. These models incrementally apply a matrix decomposition method on each origin/destination matrix in each instant and then construct a time series of model errors. For instance, in [8] a variant of PCA called Compact Matrix Decomposition (CMD) is applied to traffic flow matrices, followed by the creation of a time series of the reconstruction's errors which is then analyzed for anomaly detection. Similarly, in [9] authors apply a strategy for anomaly detection from evolving Internet networks. These methods, even though they are good choices for the detection of anomalous traffic links have a limited application for event detection. The main reason is that their incrementalization over time mode causes the loss of some temporal information, which is essential for event detection. Additionally, they are vulnerable to seasonal effects [9] and as a consequence they are not ideal for modeling human-generated data sets [10].

There is another category of methods called graph-based techniques [11] that are being developed in the social network analysis community. For instance, Markov chains can be used to model the evolution of (social) networks [12,13]. However, these methods are less relevant to the present study, as we are more interested in global events rather than anomalous nodes, edges or communities.

Tensor models are sophisticated tools used for traffic data modeling, these do not include the majority of the above-mentioned limitations. The need for tensors in traffic data modeling has emerged in recent years in two research communities, including data mining [14–17] for network anomaly detection and transportation systems [18–22] for traffic estimation and imputation. The majority of related works are covered in the recent survey paper of [23].

Tensor decomposition, an essential technique used in these models is a powerful tool for the analysis of multiway data with many applications [24] in psychometrics, chemometrics, neuroscience, signal processing, bioinformatics, computer vision and data mining. Tensor solutions, opposed to other techniques, are able to model spatio-temporal fluctuations [25]. Therefore, they are capable to generate a more natural model from data and consequently discover more realistic patterns. Despite great flexibility and quality of tensor models, the current form of these techniques still have some difficulties in detecting complex event patterns, especially in traffic tensors.

In traffic tensors, as opposed to other kind of tensors, we deal with some complex fluctuations that require a more thorough consideration. Our preliminary analysis indicates that a higher variance of the models's quality, is related to a lower sensitivity to topological fluctuations by the tensor models. Hence, the current form of naive tensor models, may not yet be sufficiently prepared for traffic tensors.

In this work we address this delicate problem and propose a hybrid tensor model that simultaneously takes into account both flow rates and network topology in a unified model. Another problem about tensor models is that they require a complicated input parameter which is usually very difficult to choose. This leads to the incorrect parameter selection risk [26] which directly influences the model quality. In order to solve this problem we examine the application of a novel adjustable core size Tucker decomposition technique (ACS-Tucker) [27] that takes into account core size determination as a part of the solution in the optimization process.

To the best of our knowledge, this is the first work that addresses the problem of mixed event detection in traffic tensors. Moreover, we propose for the first time a statistical framework for the combined flow and topology tensors in a unified detection mechanism. It is also the first research that studies the application of tensor rank estimation techniques for event detection.

The rest of the paper is organized as follows: Section 2 outlines the preliminary concepts. In Section 3 we describe our hybrid tensor model and its components. Section 4 defines the experimental settings. In Sections 5 and 6 we respectively present the result of our simulation and some real case studies. We discuss scalability issues in Section 7. Finally, the last section concludes the exposition, presenting the final remarks.

2. Preliminary concepts

In this section we elucidate some necessary notations and concepts required for further description of the proposed method.

2.1. Notations

Following [28], throughout the paper, scalars are denoted by non-bold lowercase letters (e.g. i), vectors are denoted by boldface lowercase letters (e.g. $\mathbf{a}$), matrices are denoted by boldface capital letters (e.g. $\mathbf{A}$) and tensors are denoted by calligraphic letters (e.g. $\mathcal{X}$). A tensor is a multi-dimensional array and its order is its number of dimensions, also known as ways or modes. Vectors, matrices and tensors are respectively equivalent to the first, second and dth order tensor where $d \geq 3$.

2.2. Tensor decomposition

Traditional data analysis techniques, such as the PCA, clustering, regression, etc. are only able to model bidimensional data and

they do not consider interactions between more than two dimensions. However, in several real-world phenomena, there is a mutual relationship between more than two dimensions, in particular, when the time dimension is added to the problem. Multi-way analysis considers all mutual dependencies between the different dimensions and provides a compact representation of the original data in lower dimensional spaces. The most common multi-way analysis techniques are Tucker [29] and CP/PARAFAC [30], which are generalized versions of PCA or, in particular, Singular Value Decomposition (SVD) for higher order matrices.

2.3. Tucker decomposition

Tucker decomposition is an optimization task, through which a large tensor can be estimated as the product of a smaller tensor with predetermined dimensions (called core tensor), multiplied by factor matrices in each dimension. Formally, the problem can be defined as follows: Given a tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times \ldots \times n_d}$, find a core tensor $\mathcal{G} \in \mathbb{R}^{r_1 \times r_2 \times \ldots \times r_d}$ with pre-defined integers r_i with $1 \le r_i \le n_i$ for $i = 1,2,\ldots,d$, that optimizes:

$$\begin{aligned} \min \quad & \| \mathcal{X} - \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \ldots \times_d \mathbf{A}^{(d)} \| \\ \text{s.t.} \quad & \mathcal{G} \in \mathbb{R}^{r_1 \times r_2 \times \ldots \times r_d}, \\ & \mathbf{A}^{(i)} \in \mathbb{R}^{n_i \times r_i}, \quad \mathbf{A}^{(i)}\Big)^T \mathbf{A}^{(i)} = I, \quad i = 1, 2, \ldots, d. \end{aligned} \tag{1}$$

In the above model, d represents the dimension of the tensor (e.g. for three-dimensional tensor, $d=3$) and r_1, r_2,..., r_d ($i = 1, 2, \ldots, d$) are model input parameters (core size).

3. Hybrid tensor model (HTM)

Fig. 3 illustrates an example of our proposed hybrid tensor model. The procedure consists of three major stages. The first is data modeling, which is specified with numbers 1–3 in the figure. In these steps, as opposed to existing approaches that rely on only traffic flow tensor, we construct a new tensor named topology which is the boolean copy of the original tensor.

The second step of the proposed solution focuses on Tucker optimal core size estimation problem. Like data clustering, the number of clusters as an input parameter plays a significant role in the quality of the clustering model, in Tucker model, core size is quite important, because it directly influences the model's quality.

Finally, we address the event detection task (numbers 5–7 in the figure) by combining both topology and flow models using multivariate statistics and tracking the system's behavior in the hybrid subspace. This part is the major contribution of this research. We provide a statistical framework for combining the subspace of flow and topology tensors and detect events in the hybrid subspace.

In the following sections, each of the above components is described in more detail.

3.1. Data model

The original target data set is represented as a graph $G = (V(G), E(G))$ where $V(G)$ are nodes and $E(G)$ is a set of edges. $V(G)$ can be countries, cities or stations and $E(G)$ indicates the flow volume between the nodes (e.g. number of travels or trade volume). In order to analyse the data in the tensor scheme, we first need to transform it into an adjacency matrix. G is transformed into a flow adjacency matrix $\mathbf{OD}(N \times M)$. The entries of the matrix take values from the interval $[0, \max(w)]$, where w represents the volume of flow between the two corresponding nodes.

In the next step, finite t consecutive adjacency matrices $\mathbf{OD}_1$, $\mathbf{OD}_2$, ..., $\mathbf{OD}_t$ are combined, in order to generate a tensor $\mathcal{X} \in \mathbb{R}^{N \times M \times t}$. This tensor can be called *traffic tensor* or *flow tensor* or alternatively *naive tensor*. Thereafter, a boolean copy of $\mathcal{X}$ is generated, replacing all non-zero elements with 1 resulting in what is named a *topology tensor*.

We keep the topology tensor in parallel to highlight those events that are associated with structural fluctuations in the

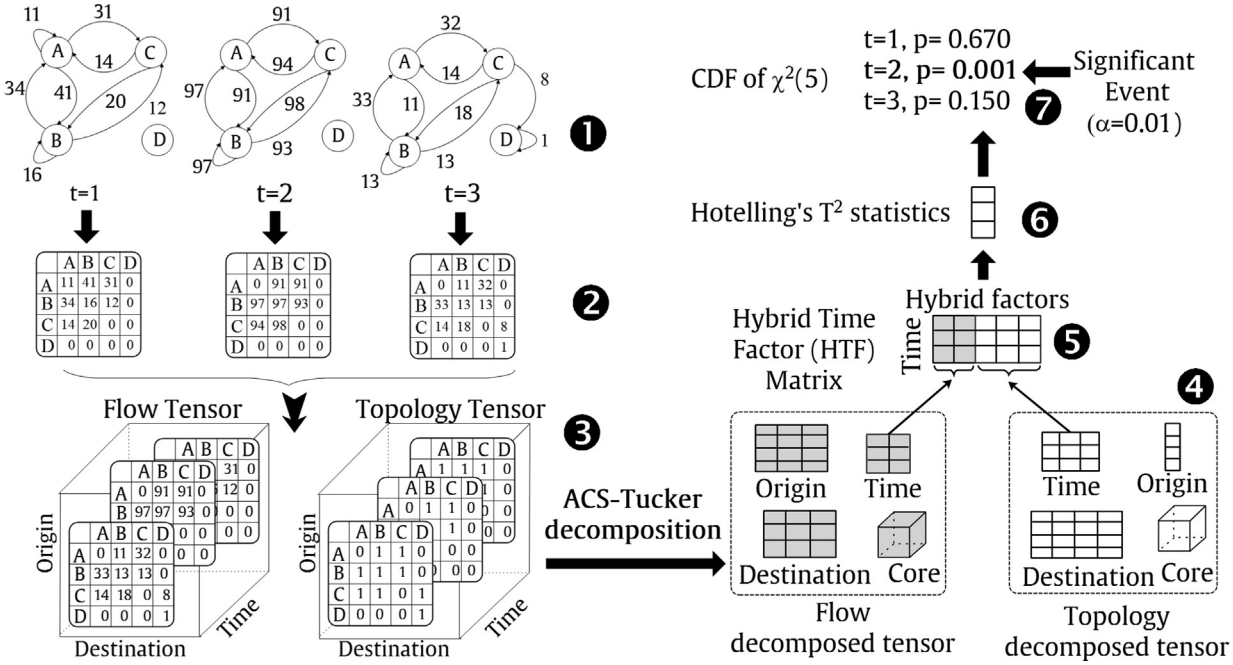

Fig. 3. An illustrative example of the proposed method (HTM) for event detection from traffic tensor; (1) Traffic data in graph structure; (2) data is transferred to adjacency matrix; (3) adjacency tensor is built by combining adjacency matrices; (4) ACS-Tucker decomposition is applied on both flow and topology tensors. ACS-Tucker automatically determines core size of (3,3,2) for flow tensor and (1,4,3) for topology tensor; (5) hybrid time factor (HTF) matrix is generated by combining time factor matrices of flow and topology; (6) Hotelling's T^2 statistics is computed from HTF matrix; (7) cumulative distribution function (CDF) of the Hotelling's T^2 statistics to χ^2 distribution with 5 degrees of freedom is computed, p-value is reported as 1-CDF and those instants with α lower than $\alpha = 0.01$ are marked as significant events.

topology of the network. Due to the high rate of variance in the flow rate, these types of changes can remain invisible if we only focus on the flow tensor. In other words, the topology as opposed to the original traffic tensor, is more sensitive to topological changes and on the other end is non-sensitive to severe variation in the flow rates.

3.2. ACS-tucker decomposition of flow/topology tensors

In this section we describe the ACS-Tucker decomposition method [27] which we use for decomposition of flow and topology tensors. Some notations are borrowed from [27].

Given the traffic tensor $\mathcal{X} \in \mathbb{R}^{n\times m\times t}$ and core sizes (r_1, r_2, r_3), Tucker model decomposes the original tensor to an abstract subspace, including core tensor $\mathcal{G} \in \mathbb{R}^{r_1\times r_2\times r_3}$ and factor matrices $\mathbf{A}_1 \in \mathbb{R}^{n\times r_1}$, $\mathbf{A}_2 \in \mathbb{R}^{m\times r_2}$ and $\mathbf{A}_3 \in \mathbb{R}^{t\times r_3}$ such that $\mathcal{X} \approx \mathcal{G}\times_1\mathbf{A}_1\times_2\mathbf{A}_2\times_3\mathbf{A}_3$.

Determination of Tucker model parameters (r_1, r_2, r_3) in the above equation is a difficult task. Instead of Tucker we may want to use more specific models such as PARAFAC that require only one parameter, but in that case we loose the model generality and flexibility. The situation gets worse when the tensor is not orthogonally constrained as PARAFAC assumes.

In [27] a new method called adjustable core size Tucker decomposition (ACS-Tucker) is proposed. It performs Tucker decomposition with an unspecified size of the core through *maximum block improvement* [31]. The authors apply the method on known-rank tensors (both simulated and real) and show that ACS-Tucker is remarkably accurate compared to existing methods.

Given $\mathcal{X} \in \mathbb{R}^{n\times m\times t}$, the goal of ACS-Tucker is to find the best approximation of $\mathcal{X}$, as a product of a smaller core tensor and factor matrices. Here, the dimensions of the core tensor r_i are no longer pre-specified and need to be determined. However, to prevent r_i from being too large, the sum of r_i is assumed to be equal to c, i.e. $r_1+r_2+r_3=c$. As we already know for the third-order tensor $\mathcal{X}$, Tucker decomposition in optimization (1) is equivalent to the following maximization problem for third-order tensors [27,28]:

$$\begin{aligned} \max \quad & \|\mathcal{X}\times_1(\mathbf{A}^{(1)})^T\times_2(\mathbf{A}^{(2)})^T\times_3(\mathbf{A}^{(3)})^T\| \\ \text{s.t.} \quad & \mathbf{A}^{(i)} \in \mathbb{R}^{n_i\times r_i}, \quad (\mathbf{A}^{(i)})^T\mathbf{A}^{(i)} = I, \quad i=1,2,3. \end{aligned} \tag{2}$$

We need to determine r_i. However, these two constraints make the objective function incompatible to a direct solution. In order to combine the block variables $\mathbf{A}^i$ and r_i variables, a new block variable $\mathbf{Y}^{(i)} \in \mathbb{R}^{m_i\times m_i}$ is defined such that m_1:=min$\{n, c\}$, m_2:=min$\{m, c\}$, m_3:=min$\{t, c\}$ and $\mathbf{Y}^{(i)} = diag(y^{(i)})$, $y^{(i)} \in \{0,1\}^{m_i}$, $\sum_{j=1}^{m_i} y_j^{(i)} = r_i$. If we replace the term $(\mathbf{A}^{(i)}\mathbf{Y}^{(i)})^T$ with $\mathbf{A}^i$ in (2) we have:

$$\begin{aligned} \max \quad & \|\mathcal{X}\times_1(\mathbf{A}^{(1)}\mathbf{Y}^{(1)})^T\times_2(\mathbf{A}^{(2)}\mathbf{Y}^{(2)})^T\times_3(\mathbf{A}^{(3)}\mathbf{Y}^{(3)})^T\| \\ \text{s.t.} \quad & \mathbf{A}^{(i)} \in \mathbb{R}^{n_i\times r_i}, \quad (\mathbf{A}^{(i)})^T\mathbf{A}^{(i)} = \mathbf{I}, \quad y^{(i)} \in \{0,1\}^{m_i}, \\ & \sum_{j=1}^{m_i} y_j^{(i)} \geq 1, \quad \sum_{i=1}^{3}\sum_{j=1}^{m_i} y_j^{(i)} = c, \quad i=1,2,3. \end{aligned} \tag{3}$$

If the nontransferable constraint $\sum_{i=1}^{3}\sum_{j=1}^{m_i} y_j^{(i)} = c$ is replaced and a λ is defined, as a penalty parameter, the objective function is then reformulated to the following maximization problem:

$$\begin{aligned} \max \quad & \|\mathcal{X}\times_1(\mathbf{A}^{(1)}\mathbf{Y}^{(1)})^T\times_2(\mathbf{A}^{(2)}\mathbf{Y}^{(2)})^T\times_3(\mathbf{A}^{(3)}\mathbf{Y}^{(3)})^T\| \\ \text{s.t.} \quad & \mathbf{A}^{(i)} \in \mathbb{R}^{n_i\times r_i}, \quad (\mathbf{A}^{(i)})^T\mathbf{A}^{(i)} = \mathbf{I}, \quad y^{(i)} \in \{0,1\}^{m_i}, \\ & \sum_{j=1}^{m_i} y_j^{(i)} \geq 1, \sum_{i=1}^{3}\sum_{j=1}^{m_i} y_j^{(i)} = c, \quad i=1,2,3. \end{aligned} \tag{4}$$

In the new formula, each block $(\mathbf{A}^{(i)}\mathbf{Y}^{(i)})^T, i=1,2,3.$ can be separately optimized, while the other blocks are fixed. This new optimization task can be performed, using the Maximum Block Improvement (MBI) [31]. As we can see, Tucker core size is not required to be pre-specified, because it is now included in the optimization task. After MBI converges, the number of non-zero entries in $\mathbf{Y}^{(1)}$, $\mathbf{Y}^{(2)}$ and $\mathbf{Y}^{(3)}$ will be equal to r_1, r_2 and r_3. The full description of ACS-Tucker is presented in Algorithm 2 of [27].

3.3. Event detection

As illustrated in Fig. 3, after we obtain the factor matrices from decomposition of flow and topology tensors, we combine time factor matrices corresponding to both tensors to form a new hybrid matrix called *hybrid time factor* (**HTF**). This matrix has t rows (total time instants) and k columns where k denotes the number of columns in flow time factor plus the number of factors in the topology time factor, however k depends on the ACS-Tucker output. For instance, in the presented example, **HTF** matrix includes 2+3=5 columns, because ACS-Tucker outputs core size of (3,3,2) for flow tensor and (1,4,3) for topology. As we can see, **HTF** matrix is equivalent to a multivariate time series. Therefore, we reformulate the problem to multivariate time series monitoring, and define event as a time instant when the multivariate series falls out of control.

Hotelling's T^2 statistic is a common metric for monitoring multivariate time series, which is computed as follows [32]:

$$T_t^2 = (X_t-\mu)^T S^{-1}(X_t-\mu) \tag{5}$$

where μ is the mean, X_t is the multivariate observation at time t and S is the covariance matrix.

If we assume that under controlled conditions, multivariate time series (**HTF**) follows a multivariate normal distribution, T^2 should be explained by χ^2 distribution with k degrees of freedom [33, p. 23] where k is the number of time series. Therefore, if some abnormal event occurs at the specific time instant we should witness a deviation, either in flow or topology, from the χ^2 distribution at that moment. The Cumulative Distribution Function (CDF) for the χ^2 distribution [34, p. 333], which is shown in Eq. (6) computes this deviation. Consequently, Eq. (7) is equivalent to the statistical significance (p-value) for each time instant, which shows the severity of the deviation in each time, i.e. null hypothesis: no abnormal event:

$$CDF = F(x|k) = \int_0^x \frac{t^{(k-2)/2}e^{-t/2}}{2^{k/2}\Gamma(k/2)}dt \tag{6}$$

$$P = 1 - CDF \tag{7}$$

P-values closer to zero represent more severe events. In Eq. (6), k denotes the degree of freedom, which is equal to the number of columns in **HTF** matrix, x refers to the obtained Hotelling's T^2 statistics in Eq. (5) and Γ refers to the Gamma function.

4. Experimental settings

In this section we briefly describe the methods that are compared in this paper. Next we define the tools, parameters and evaluation metrics we use in the experiments.

4.1. Compared methods

There exists no hybrid tensor model in the literature like the one we propose in this work. Therefore, we develop a similar baseline method to which we will be able to compare our proposed model. To have an evaluation against naive tensor models we also compare it against two non-hybrid models. Furthermore we compare it against two matrix residual models (see Fig. 2) including DTA-based and PCA-based approaches. These approaches are described in the following. For simplicity, when referring

to these methods we present their abbreviated names between parenthesis.

DIFFIT-ALS: In this approach a residual-based tensor rank estimation technique such as DIFFIT [35] is employed on both flow and topology tensors for estimation of these tensors' rank. Then we feed the obtained ranks to the Tucker-ALS algorithm. The rest of the method (event detection part) is similar to our proposed hybrid model. It means that the baseline approach applies this process for both flow and topology tensors.

We can mention two main differences between this baseline method and the proposed model. The first is that the baseline approach exploits the Alternating Least Square (ALS) [36] for solving Tucker decomposition while our proposed model uses Maximum Block Improvement (MBI) [31]; the second difference is that the baseline method instead of a single-step decomposition procedure, uses a two-step classic strategy of rank estimation plus tensor decomposition.

CP-APR (*CPR*): In this method we apply CP-APR (nonnegative CP with alternating Poisson regression) [37] on the flow tensor only. The detection part is similar to the proposed method, however Hotelling's T^2 is applied only on the time's latent factors.

MBI-log: Borrowing the idea from [38] we apply ACS-Tucker decomposition only on log non-zero counts of the flow tensor. This is a solution to reduce the effect of large values when the majority of cells are of smaller values. The detection part is similar to the previous method.

DTA-Residual (*DTA*): Dynamic Tensor Analysis (DTA) is the first reported tensor-based solution for anomaly detection from temporal network data sets [16]. In this method we apply DTA on the traffic tensor and then construct the DTA error time series. Finally, we apply a *z*-score control chart in this time series for event detection.

PCA-Residual (*PCA*): This method is similar to the method proposed in [8], differing with the application of PCA instead of CMD algorithm. We apply PCA on traffic matrices and then construct the time series of PCA error. Afterward, we apply a *z*-score control chart in the time series for event detection.

4.2. Software

All the experiments are performed in MATLAB on a PC with Intel Core 2 Duo CPU and 3 GB RAM. Two MATLAB toolboxes are also used during the experiments: Tensor toolbox 2.5 [39] and ITA toolbox [40].

4.3. Used parameters

We set c parameter in ACS-Tucker and MBI-log as $5+5+5=15$ as well as $\lambda = 0.005\|\mathcal{X}\|$ as the default value proposed by [27]. Note that since we only address temporal event detection it is expected that the approach should not be very sensitive to parameter c. Because, with $c=15$ we have 13 degrees of freedom for the time dimension which is much greater than the usual optimum number of components (normally lower than 10 for traffic tensors). For DIFFIT we set the max core size as (5,5,5) the same maximum value we choose for ACS-Tucker. For CPR, PCA and DTA we try the models with $p=1$ to $p=13$ and then obtain the best and worst accuracies (p is the number of components). In terms of experiments with real datasets we use n-way toolbox pftest for estimation of the number of components for the CPR method [41].

4.4. Evaluation metric

We use the Area Under ROC curve (*AUC*) [42] as the main evaluation metric in this work, due to two reasons. Firstly, the area under the curve is not dependent of the chosen decision threshold, hence making the evaluation disregard the p-value threshold. This is an important criterion for an event detection application where only a single threshold discriminates events from non-events. Secondly, the ROC curve takes into account the trade-off between false alarms and true alarms. Therefore, it is more appropriate than metrics such as false alarms-only or true alarms-only. The output of all methods is p-value for each time instant. So, to compute the *AUC*, the p-value threshold varies from 0 to 1 with increments of 0.0001 in all compared methods. For a labeled data set we are able to measure the ability of methods in the detection of events, with respect to the p-value in the loop. After computing false and true alarms rates, we plot the ROC curve and only then do we compute the AUC.

5. Simulation study

In this section we describe our methodology for the creation of simulated data sets and subsequently report the obtained results.

5.1. Simulation of artificial events in real traffic data

Among different simulation strategies, the injection of artificial events into real background data provides more accuracy than wholly-simulated test sets [43]. Therefore, in our simulation study, instead of simulating both background data and events, we create artificial fluctuations with different properties in the real background data. The critical part, however, is that the background data should be free of any events or outliers. Otherwise, the created events will conflict with the existing anomalies, resulting in a misleading conclusion.

An ideal data set for evaluating the proposed method is required to meet two conditions. The first is that it should contain a very low amount of outliers, anomalies, or other events. And second, some amount of prior knowledge should be available via external sources. Among different data sets, we found world trade data [44,45] more suitable according to the above-mentioned requirements. The data set contains trade flows for pairs of world countries between 1870 and 2009. We first remove the years corresponding to the global economic crisis (2007–2009) based on our prior knowledge. Then the severe outliers and anomalies in the data set are removed, based on a proposed approach in [46], which is based on a combination of Tucker decomposition, clustering and visualization. During this process, we remove information for about 30 years of the raw data. The final tensor *country* × *country* × *years* is produced with a size of $207 \times 207 \times 110$.

As depicted examples in Fig. 1 we inject three types of artificial events:

- Event type 1 (mutation in overall traffic flow) : a random positive integer from range $(\max/8, \max)$ is added to all elements of the matrix.
- Event type 2 (alteration of only topology in part of the network) : a random positive integer from range $(1, avg)$ is added to the zero elements in $a \times a$ area.
- Event type 3 (chaotic behavior in part of the network) : each cell in a $a \times a$ area is replaced with a random integer from the range $(\max/8, \max)$

These three injected event types are illustrated in Fig. 4. First, the maximum (max) and the average (avg) of the tensor are computed, then for type 1, a matrix of uniformly distributed random integers is generated from the range $(\max/8, \max)$. Subsequently, the generated matrix is added to all elements within the adjacency matrix. For the second type, a $a \times a$ area is selected and

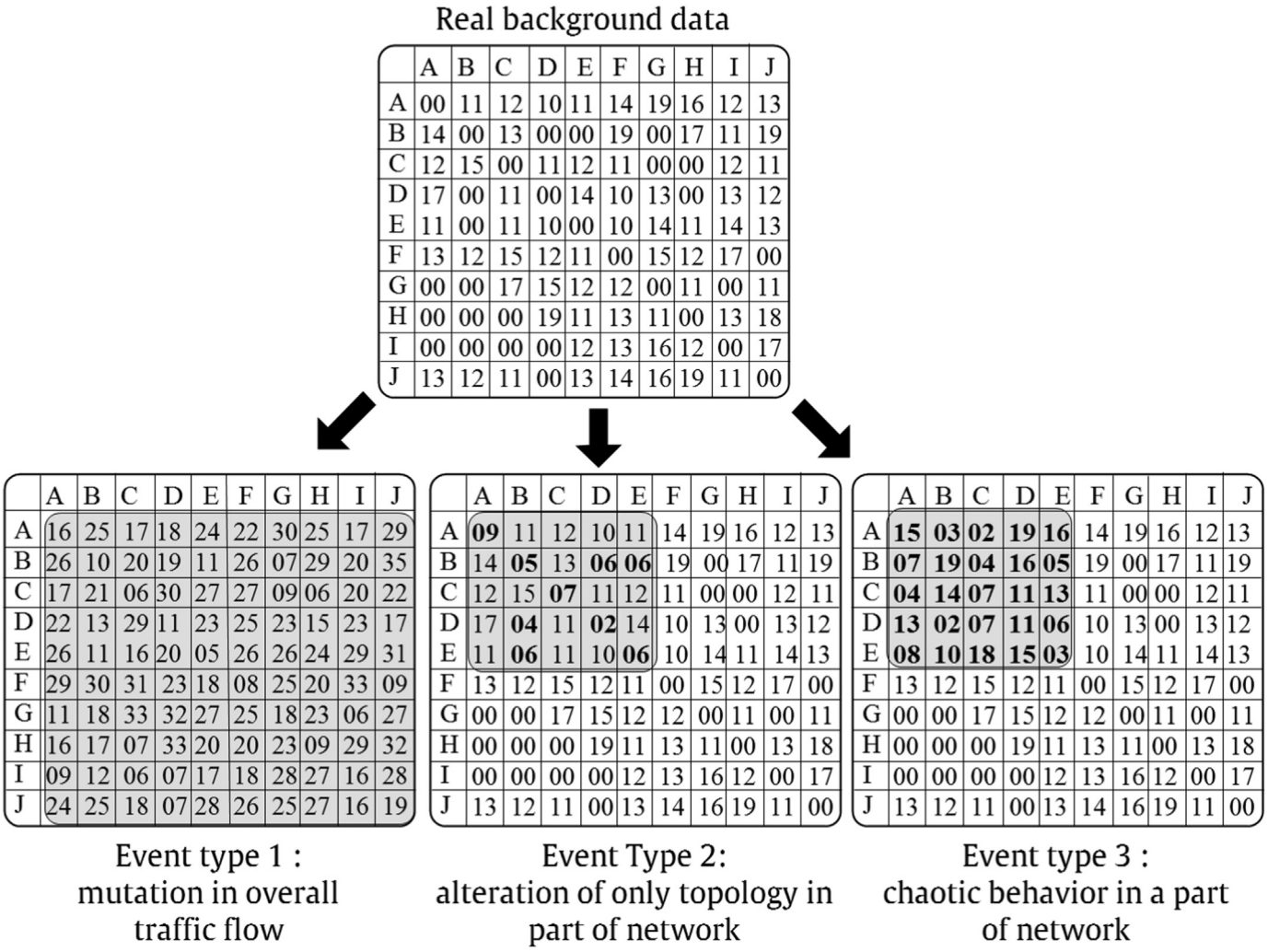

Fig. 4. Illustrative example of injection of artificial events.

Table 1
Average AUC for event injection on the 15 areas from 2×2 to 30×30. The total possible injection area is 207×207 and total number of time instants (matrices) is 110. Scenario 1: four matrices are injected with event type 1. Scenario 2: four matrices are injected with event type 2. Scenario 3: four matrices are injected with event type 3. Mixture scenario: four matrices are injected with event type 1, four with type 2 and four with type 3, in total 12 matrices.

Scenario	Hybrid tensor models		Naive tensor models			Matrix models			
	DIFFIT-ALS	HTM (proposed)	MBI-log	CPR_{Best}	CPR_{Worst}	PCA_{Best}	PCA_{Worst}	DTA_{Best}	DTA_{Worst}
1	0.9602	0.9644	0.9906	0.9992	0.9297	**0.9996**	0.5057	0.9910	0.8857
2	0.9243	**1.0000**	**1.0000**	**1.0000**	0.9971	**1.0000**	0.3895	**1.0000**	0.2084
3	0.4804	0.7666	0.4682	**0.8431**	0.4677	0.6142	0.4977	0.7666	0.4386
Mixture	0.7606	**0.8835**	0.8111	0.8478	0.6561	0.7838	0.6537	0.6821	0.4648

a random integer from the range $(1, avg)$ is added to the existing zero elements in the area. For the third type, a $a \times a$ matrix of uniformly distributed random integers is generated from the range $(\max/8, \max)$ and is replaced with a selected area of $a \times a$ in the adjacency matrix.

Datasets are generated using four scenarios: In the first scenario, we only create type 1 events in adjacency matrices corresponding to four time instants. In the second scenario, we inject type 2 events in four instants while in the third scenario we inject type 3 events in four matrices. Finally, in the fourth scenario, we inject a mixture of events type 1, type 2, and type 3, each type is injected on four time instants, for a total of 12 time instants.

We also simulate low-scale events by varying the a parameter from 2 (1% of nodes) to 30 (15% of nodes) with steps of 2 and large-scale events by varying a from 50 (quarter of nodes) to 200 (almost all nodes) with steps of 10.

The events in these data sets are labeled for the injected instants. Hence, we only need to apply the methods on the created data sets and evaluate the performance in detecting artificial events.

5.1.1. Evaluation of methods in detection of different event types

Table 1 demonstrates the average AUC for low-scale events, regarding four simulation scenarios, i.e. event types 1, 2 and 3 and

their mixture. To have a fair comparison we perform experiments for $p=1$ to $p=13$ for CPR, PCA and DTA (p is the number of components) and present their worst and best results.

All methods except PCA have a very good performance for the detection of type 1 events indicating that detection of this kind (overall mutation in traffic flows) is relatively straightforward. Although the performance of PCA_{best} is the best for the type 1, it is the worst for PCA_{worst} which makes PCA very sensitive to the k parameter and thus unreliable. Another interesting point that we can observe is that the incorporation of the topology model has a negative effect of almost 5% (comparing hybrid models versus naive models for type 1 and type 2 events). The main reason is that hybrid models manifest hypersensitivity to the small topological fluctuations in this scenario and therefore raise more false alarms and consequently lower AUC.

For type 2 event matrix residual models (i.e. DTA-R and PCA-R) seem very sensitive to the k parameter. Therefore, they are considered very risky to use. We recall that event type 2 events simulate topological alterations in part of the network. The obtained results reveal that the matrix residual models are unable to detect topological fluctuations. Comparing HTM against the other hybrid model also reveals that ACS-Tucker has been a quite helpful strategy in comparison of the traditional DIFFIT+Tucker model.

In terms of the type 3 events, HTM outperforms DIFFIT-ALS demonstrating the effectiveness of the MBI-based Tucker decomposition. Although CPR beats HTM in its best accuracy, it seems risky, because in its worst case can result in a 0.30 lower accuracy. DTA also in its best case can have a performance equal to HTM and in its worst case presents almost 0.3 less accuracy.

If we know, in advance, that the system only faces a specific event type, we may use either of the compared methods when they seem more successful. However, in reality, we do not have any prior knowledge about the types of events that the system is faced with, rather we deal with a mixture of these events in the system's life cycle. In such cases, we need to rely on a method that performs reasonably in handling all types of events. The mixture event simulation is designed to evaluate methods that handle such realistic circumstances. The results in Table 1 clarify the superiority of our proposed method over other techniques. As we can see, HTM has a better overall performance when dealing with mixture events: over matrix-based models (10–20% better), the baseline hybrid method (12% improvement) and baseline naive methods (4–13%).

5.1.2. Effectiveness of hybrid model

Here we study the effect of our hybrid strategy in the presence of low-scale and large-scale mixture events. In particular, we are interested to know how topology tensors improve the naive model. Therefore, we focus on the two tensor models including our proposed one and the baseline method described in Section 4.1. We compute the mean AUC for a mixture event scenario for two situations: low-scale and large-scale events.

The obtained results for low-scale mixture events presented in Table 2 indicate that the hybrid tensor strategy considerably improves the detection accuracy for both low-scale and large-scale events. In other words, if we rely only on the flow tensor, depending on the scale of events we will have 7–16% lower performance comparing to the hybrid model.

We also observe that in large-scale events when the majority of the nodes get involved, the events are much easier to discover. Our conclusion for low-scale events is valid for large-scale events as well. However, in reality we barely experience large-scale events, because it is very rare for the whole system to get affected by a chaotic or topological mutation. Despite this, we observe that incorporating the topology model works out better for large-scale events than for low-scale events.

Table 2
Average AUC for event injection with a mixture scenario for low-scale (15 areas from 2×2 to 30×30) and large-scale (16 areas from 50×50 to 200×200). Flow: only the naive tensor model is used. Topology: only the topology tensor model is used. Hybrid: our proposed hybrid model that combines both naive and topology models is used.

Used tensor	Low-scale events		Large-scale events	
	DIFFIT-ALS	HTM (proposed)	DIFFIT-ALS	HTM (proposed)
Flow	0.6706	0.8111	0.6872	0.8490
Topology	0.5032	0.6388	0.7762	0.8216
Hybrid	**0.7606**	**0.8835**	**0.8449**	**1.0000**

6. Real case studies

There exist some popular traffic data sets such as KDD Cup 99 [50] that are widely used for analysis of Internet traffic or PeMS [51] that is used for experimenting with transportation data. However, two problems exist about these data sets. The first is that in some data sets such as PeMS we do not have any knowledge about real events, and worse, there is no external knowledge source for labeling events [52]. Benchmark data sets such as KDD Cup 99 are also repeatedly criticized in different aspects related to the quality of ground truth (e.g. in [53]). In this study as listed in Table 3 we consider four publicly available real-world traffic data sets from three domains with various temporal scales and diverse mobility structures. The fact that makes these data sets interesting is that we can partially label these data sets via existing external knowledge sources.

We evaluate each case study with various metrics such as AUC, hypothesis testing (p-value) and false/true alarms to gain a more comprehensive assessment. In the following subsections, each data set will first be described in detail and then we will present the obtained results. Note that the experiments' settings are the same as in Section 4. However, we skip matrix-based methods for the real experiments since they were beaten by tensor-based methods in the simulation study. For CP-APR we select the number of components via pftest in N-way toolbox [41].

6.1. Bike-sharing data set

Bike sharing systems are a new generation of traditional bike rentals, where the whole process from membership, rental and return has become automatic. These systems enable users to easily rent a bike from a particular position and return it at another. There exists a great interest in bike-sharing systems due to their important role in transportation and environment. What makes bike sharing data more attractive is that opposed to other transportation services, such as bus or subway, the duration of travel,

Table 3
Real-world data sets used in the experiments.

Data set	Dimensions	Size
World trade data [45,44]	*Country* × *Country* × *Year*	207 × 207 × 140
US flight data [47]	*Airport* × *Airport* × *Month*	169 × 368 × 195
Bike sharing, Washington D.C. [48]	*Station* × *Station* × *Day*	157 × 157 × 731
Bike sharing, Boston [49]	*Station* × *Station* × *Day*	95 × 95 × 327

Table 4
AUC for bike sharing data sets. The ability of methods in detection of events from Boston and Washington, D.C. data sets. For CPR $p=6$ and $p=4$ are obtained by running a pftest for respectively Washington D.C. and Boston datasets.

Method family	Method name	Boston data set	Washington, D.C. data set
Hybrid tensor models	DIFFIT-ALS	0.7529	0.6367
	HTM (proposed)	**0.8188**	**0.7414**
Naive tensor models	CPR	0.7145	0.6767
	MBI-log	0.6771	0.7153

Table 5
Statistical significance (p-value) of events related to the world financial crisis 2007–2009. The events with $p-\text{value} \leq 0.05$ are shown boldface. For CPR the number of components is selected as $p=5$ based on pftest.

Method family	Method name	2007	2008	2009
Hybrid tensor models	DIFFIT+ALS	$\mathbf{3.06 \times 10^{-6}}$	$\mathbf{2.05 \times 10^{-9}}$	$\mathbf{4.46 \times 10^{-7}}$
	HTM (proposed)	$\mathbf{7.15} \times 10^{-5}$	$\mathbf{4.84} \times 10^{-8}$	$\mathbf{1.42} \times 10^{-6}$
Naive tensor models	MBI-log	0.72	0.58	0.68
	CPR	$\mathbf{4.90} \times 10^{-11}$	$\mathbf{2.09} \times 10^{-13}$	$\mathbf{6.66} \times 10^{-9}$

departure and arrival position is explicitly recorded in these systems. This feature turns the bike sharing system into a virtual sensor network that can be used for sensing mobility in the city. Hence, it is anticipated that most of the major events in the city could be detected via the monitoring of this data [52].

Our objective in this case study is the examination of methods in terms of model quality and stability. Our expectation of a good model is that it not only presents a good performance, but also behaves stably on two data sets from same domain. To evaluate this, two data sets are extracted from the bike-sharing networks in two different cities Washington, D.C. and Boston, both in the United States. Characteristics of these data sets are described in the following.

6.1.1. Washington, D.C. bike-sharing data set

The Washington, D.C. bike-sharing data set [48] includes a historical usage log of all transactions in the bike-sharing network from a two-year window from 01 January 2011 to 31 December 2012 for a total of 731 days. There are 207 bike stations in the network, however, some stations have zero or rare connections. From the whole set we select the top 157 stations that have more frequent trips. Hence, for each day, we create an O/D matrix with a size of 157×157, so that each element in the matrix represents the number of passengers who traveled from one station to another on that day. The network data is directed and weighted. Thus, we account for the number of passengers in the opposite direction as well (i.e. A to B and B to A). Each user can also rent a bike from one station and return it to the same station. Therefore, the nodes can have a relationship with themselves, as well. Combining all O/D matrices as the tensor model shown in Fig. 2 results in a traffic tensor sizing $157 \times 157 \times 731$.

6.1.2. Boston bike-sharing data set

Boston bike-sharing data set has been extracted from hub-way data challenge 2013 [49]. It includes a historical usage log of all transactions in the network from 28 July 2011 to 01 October 2012, exclusive to the system's off-days in the winter, a total of 327 days. There are also 95 stations in total. After creating adjacency matrices for each day, the generated traffic tensor will be in size of $95 \times 95 \times 327$.

6.1.3. Event labeling and evaluation strategy

It is expected that events in the city affect users' behavior, because bike users are a sample of the city population and their behavior can be considered a sample of the entire city's population. For instance, sports rivalries events, festivals, demonstrations, blackouts, natural hazards, such as hurricane and storms, etc. all affect these sample populations.

In the simulation study, we demonstrated the merits of the hybrid model. Here, we are interested to evaluate the hybrid model on a real-world data with its complex event patterns. In [52] some of the most significant events are reported in Washington, D.C. in 2012. With a similar strategy introduced in this work we extract the significant events in Boston as well. Finally, we label the Washington, D.C. and Boston data sets with respectively top-10 and top-7 most significant events

6.1.4. Results

We compare the performance of naive models versus the hybrid methods focusing on their ability to detect the labeled events. In this case study, we use AUC for the evaluation. The results in Table 4 indicate that our proposed model has a better and more robust performance for both data sets. However, it is interesting to observe that the naive models have a better performance than a hybrid model like DIFFIT-ALS on the Washington D.C. data set. Therefore DIFFIT and Tucker-ALS might not be very robust tools for being incorporated in our proposed hybrid model. MBI-log also has a relatively good performance on the Washington D.C. data set but is the worst method for the Boston data set.

6.2. World trade data set

The world trade data set [45,44] includes the bilateral trade flows between countries for the period from 1870 to 2009. The objective is to investigate the ability of the methods in detecting the well-known global financial crisis in the period of 2007–2009 [54].

Transforming the raw data to the *Country* × *Country* × *Year* scheme results in a tensor with a size of $207 \times 207 \times 140$. We apply the event detection methods and measure their corresponding p-value for the 2007–2009 crisis period. The p-value in this case study can be specifically interpreted as the probability of an abnormal event occurring in years 2007–2009. In other words, our null hypothesis is "no event occurrence" and it will be rejected if p-value is sufficiently low. Informally, $p \leq 0.01$ and $0.01 < p \leq 0.05$ are interpreted respectively very strong and strong presumptions against the null hypothesis. It is anticipated that a good model rejects the null hypothesis as strongly as possible for these years.

The p-values obtained for the years 2007–2009 are reported in Table 5. The p-values lower than 0.05 are shown boldfaced. As we can see, the MBI-log method totally fails in detecting the crisis period, while other methods are able to detect the crisis years.

Also a comparison of p-values for the years 2007, 2008 and 2009 shows that all methods report lower p-values for the year 2008 in comparison to 2007 and 2009. This leads to a new insight about the crisis, apparently the most severe situation was in 2008.

6.3. US international flights data set

USA international air passenger statistics [47] reports the commercial traffic between international points and U.S. airports from 1990 to 2013. We transform the raw data to a flow tensor *Airport* × *Airport* × *Month* in size of $169 \times 368 \times 195$. Our objective

Table 6
Detection of bankruptcies in the United States airlines. Boldfaced values are related to true alarms and the other values are associated with false alarms ($\alpha = 0.05$). For CPR the number of components is selected as $p=2$ based on the pftest.

Method family	Method Name	Alarms
Hybrid tensor models	DIFFIT-ALS	**1991/02**, **2002/08**, 2006/02
	HTM (proposed)	**1991/02**, **2002/08**
Naive tensor models	CPR	1996/01 1997/01, 1997/02, 2000/08, 2004/07, 2005/07
	MBI-log	None

is to evaluate the capability of the methods in the tracking of events related to the major airline's bankruptcy records in USA.

We consider three of most influential airline bankruptcies that are related to three major airlines: Pan Am World Airways, Eastern Air Lines and US Airways who filed bankruptcy, respectively in 8 January 1991, 18 January 1991 and 11 August 2002 [55].

In this case study, we want to have a better look at both true and false alarms. Hence, we report all alarms raised by the methods for $\alpha = 0.05$.

The results are demonstrated in Table 6. The months related to the aforementioned bankruptcies are shown boldfaced, while false alarms are specified with a regular font.

The first significant month is February of 1991 that coincides to the bankruptcy of two major airlines *Pan Am World Airways* and *Eastern Air Lines*. Among methods, only the hybrid tensor models have been able to track this event. The second most influential month is August 2002 which is related to the bankruptcy of another major airline *US Airways*. As it can be seen, only hybrid methods, including HTM and DIFFIT-ALS have been able to identify this event. In fact, the hybrid tensor models are the only approaches that detect both of these two major events. Even HTM performs better than the baseline method. On the other hand, it appears that naive tensor models do not present an acceptable performance. CPR signals 6 false alarms and MBI-log is unable to detect any event.

7. Scalability issues

Tensor models in principle are known to be computationally expensive. For decomposition of a cubic $n \times n \times n$ tensor with HOSVD we require $O(n^4)$ time and $O(n^3)$ space. Our hybrid model in this study has the same complexity as the naive tensor models, both in terms of time and space. However, since we require two tensor models, the execution time is doubled with no change in required memory. Even though such complexity can become a serious issue when we deal with a large-scale data.

Fortunately, traffic tensors are, to a great extent, sparse. It means that the majority of elements in traffic tensors are zero. For instance, over 90% of elements in four of the data sets we consider in this work are filled with zero elements. If we do not optimize tensor decomposition for such sparse tensors, we will be faced with the *intermediate data explosion* problem [56,14]. Many solutions have been developed for coping with this problem in the recent years. For instance, in [14] a new extension of Tucker decomposition named Memory-efficient Tucker (MET) is proposed which space complexity is linear with the number of non-zero elements in tensor (i.e. $O(nz)$), this is very helpful for sparse traffic tensors. Similarly, [56] proposes a distributed version of PARAFAC implemented in MapReduce [57] that scales up to 100 times in terms of space complexity for sparse tensors and nearly linear to the number of machines in time complexity.

In a more recent work [58], a different distributed framework is proposed for PARAFAC that divides the tensors into some small sub-tensors and solve sub-tensors problems in different machines. Similar to this work, [59] proposes a parallelized version of PARAFAC called ParCube which is optimized for sparse tensors and provides an acceleration of 14 times in runtime. In [60] authors propose a new PARAFAC-based method based on general-purpose computing, on the graphics processing unit(GPGPU) which results in a runtime acceleration of 360 times comparing to conventional CPU-based methods. There exist many more algorithms and methods for scalability of tensor decompositions which are out of the scope of this paper. However, interested readers can refer to recent works such as [61–63].

8. Conclusion and future works

We address the problem of event detection in traffic tensors. We provide some evidences that detection accuracy can be increased up to almost 10% by the separation and inclusion of network's topology in the naive tensor model. Our simulation experiments show that the proposed hybrid approach has increased detection performance up to 10–20% in comparison to matrix-based approaches when dealing with mixture events. Furthermore, our experiments on two data sets from a single domain reveal that the proposed approach is more robust-to-domain than the matrix models. We also confirm the findings in [27] that the MBI-based adjustable core size Tucker decomposition is more powerful than the traditional strategy of DIFFIT+Tucker.

Additionally, we present some intuitive examples from real-life data sets that endorse our findings in the simulation study about the robustness and the adequacy of the proposed model. As observed, HTM exhibits a successful performance on all real data sets which is in accordance with the results obtained for the simulated data sets. Although CPR appeared as a good competitor for hybrid models, it is probably not as robust as the proposed hybrid model. The data sets introduced in this paper are also found useful and can be potential choices for assessment of future traffic analysis approaches.

Some challenges will remain for future work. One is the real-time extension of the proposed model which is basically a difficult problem. Our findings in this research reveal that an incremental approach like DTA is not reliable for every event type. Hence, for solving this problem, other possibilities should be taken into account. One other direction would be testing different tensor factorization techniques for the HTM model.

Acknowledgement

This research was supported by European Commission through the project MAESTRA (Grant Number ICT-750 2013-612944). The authors also acknowledge the support of the Projects NORTE-07-0124-FEDER-000059/000056 which are financed by the North Portugal Regional Operational Program (ON.2 O Novo Norte), under the National Strategic Reference Framework (NSRF), through the European Regional Development Fund (ERDF), and by national funds, through the Portuguese funding agency, Fundação para a Ciência e a Tecnologia (FCT). The authors would like to thank Dr. Bilian Chen for providing the code and help. The authors are also grateful to the Editor-in-Chief, the Associate Editor, and anonymous reviewers for their constructive comments and Luis Gama for proofreading based on which the presentation of this paper has been greatly improved.

References

[1] M. Van Der Voort, M. Dougherty, S. Watson, Combining kohonen maps with arima time series models to forecast traffic flow, Transp. Res. Part C: Emerg. Technol. 4 (5) (1996) 307–318.

[2] M. Zhong, P. Lingras, S. Sharma, Estimation of missing traffic counts using factor, genetic, neural, and regression techniques, Transp. Res. Part C: Emerg. Technol. 12 (2) (2004) 139–166.

[3] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E.D. Kolaczyk, N. Taft, Structural analysis of network traffic flows, SIGMETRICS Perform. Eval. Rev. 32 (1) (2004) 61–72, http://dx.doi.org/10.1145/1012888.1005697.

[4] Z. Wang, K. Hu, K. Xu, B. Yin, X. Dong, Structural analysis of network traffic matrix via relaxed principal component pursuit, Comput. Netw. 56 (7) (2012) 2049–2067.

[5] H. Hotelling, Analysis of a complex of statistical variables into principal components, J. Educ. Psychol. 24 (6) (1933) 417.

[6] G.H. Golub, C. Reinsch, Singular value decomposition and least squares solutions, Numer. Math. 14 (5) (1970) 403–420.

[7] T. Ide, H. Kashima, Eigenspace-based anomaly detection in computer systems, in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2004, pp. 440–449.

[8] J. Sun, Y. Xie, H. Zhang, C. Faloutsos, Less is more: compact matrix decomposition for large sparse graphs, in: SDM, SIAM, Philadelphia, PA, USA, 2007, pp. 366–377.

[9] J. Sun, D. Tao, S. Papadimitriou, P.S. Yu, C. Faloutsos, Incremental tensor analysis: theory and applications, ACM Trans. Knowl. Discov. Data (TKDD) 2 (3) (2008) 11.

[10] P. Tune, M. Roughan, Internet traffic matrices: a primer, Recent Adv. Netw. 1 (2013).

[11] L. Akoglu, H. Tong, D. Koutra, Graph based anomaly detection and description: a survey, Data Min. Knowl. Discov. (2014) 1–63, http://dx.doi.org/10.1007/s10618-014-0365-y.

[12] M. Hamdi, V. Krishnamurthy, G. Yin, Tracking a Markov-modulated stationary degree distribution of a dynamic random graph, IEEE Trans. Inf. Theory 60 (10) (2014) 6609–6625.

[13] Y. Wei, X. Peng, J. Qiu, S. Jia, H_∞ filtering for two-dimensional continuous-time Markovian jump systems with deficient transition descriptions, Neurocomputing 167 (2015) 406–417.

[14] E. Acar, D.M. Dunlavy, T.G. Kolda, M. Mørup, Scalable tensor factorizations for incomplete data, Chemom. Intell. Lab. Syst. 106 (1) (2011) 41–56.

[15] Y. Sun, V. P. Janeja, M. P. McGuire, A. Gangopadhyay, Tnet: tensor-based neighborhood discovery in traffic networks, in: 2012 IEEE 28th International Conference on Data Engineering Workshops (ICDEW), IEEE, Los Alamitos, CA, USA, 2012, pp. 331–336.

[16] J. Sun, D. Tao, C. Faloutsos, Beyond streams and graphs: dynamic tensor analysis, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, Philadelphia, PA, USA, 2006, pp. 374–383.

[17] H. Fanaee-T, J. Gama, Multi-aspect-streaming tensor analysis, Knowl.-Based Syst. 89 (2015) 332–345.

[18] H. Tan, G. Feng, J. Feng, W. Wang, Y.-J. Zhang, F. Li, A tensor-based method for missing traffic data completion, Transp. Res. Part C: Emerg. Technol. 28 (2013) 15–27.

[19] H. Tan, Y. Wu, G. Feng, W. Wang, B. Ran, A new traffic prediction method based on dynamic tensor completion, Procedia—Soc. Behav. Sci. 96 (2013) 2431–2442.

[20] H. Tan, Z. Yang, G. Feng, W. Wang, B. Ran, Correlation analysis for tensor-based traffic data imputation method, Procedia—Soc. Behav. Sci. 96 (2013) 2611–2620.

[21] J. Wang, F. Gao, P. Cui, C. Li, Z. Xiong, Discovering urban spatio-temporal structure from time-evolving traffic networks, in: Web Technologies and Applications, Springer, Cham, Switzerland, 2014, pp. 93–104.

[22] H. Tan, J. Feng, Z. Chen, F. Yang, W. Wang, Low multilinear rank approximation of tensors and application in missing traffic data, Adv. Mech. Eng. (2014).

[23] Fanaee-T, Hadi, and João Gama. Tensor-based anomaly detection: An interdisciplinary survey. Knowl.-Based Syst. 98, 2016, 130–147. http://dx.doi.org/10.1016/j.knosys.2016.01.027.

[24] M. Mørup, Applications of tensor (multiway array) factorizations and decompositions in data mining, Wiley Interdiscip. Rev.: Data Min. Knowl. Discov. 1 (1) (2011) 24–40.

[25] H. Fanaee-T, J. Gama, Eigenevent: an algorithm for event detection from complex data streams in syndromic surveillance, Intell. Data Anal. 19 (3), http://dx.doi.org/10.3233/IDA-150734.

[26] E. Keogh, S. Lonardi, C. A. Ratanamahatana, Towards parameter-free data mining, in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2004, pp. 206–215.

[27] B. Chen, Z. Li, S. Zhang, On optimal low rank tucker approximation for tensors: the case for an adjustable core size, J. Glob. Optim. 62 (4) (2015) 811–832, http://dx.doi.org/10.1007/s10898-014-0231-x.

[28] T.G. Kolda, B.W. Bader, Tensor decompositions and applications, SIAM Rev. 51 (3) (2009) 455–500, http://dx.doi.org/10.1137/07070111X.

[29] L.R. Tucker, Some mathematical notes on three-mode factor analysis, Psychometrika 31 (3) (1966) 279–311.

[30] R. A. Harshman, Foundations of the parafac procedure: models and conditions for an explanatory multimodal factor analysis, UCLA Working Papers in Phonetics 16 (1970) 1–84.

[31] B. Chen, S. He, Z. Li, S. Zhang, Maximum block improvement and polynomial optimization, SIAM J. Optim. 22 (1) (2012) 87–107.

[32] I. Yahav, G. Shmueli, Directionally sensitive multivariate control charts in practice: application to biosurveillance, Qual. Reliab. Eng. Int. 30 (2) (2014) 159–179.

[33] R.L. Mason, J.C. Young, Multivariate Statistical Process Control with Industrial Applications, vol. 9, SIAM, London, UK, 2002.

[34] R.A. Thisted, Elements of Statistical Computing: Numerical Computation, vol. 1, CRC Press, London, UK, 1988.

[35] H.A. Kiers, A. Kinderen, A fast method for choosing the numbers of components in Tucker3 analysis, Br. J. Math. Stat. Psychol. 56 (1) (2003) 119–125.

[36] J.D. Carroll, J.-J. Chang, Analysis of individual differences in multidimensional scaling via an n-way generalization of Eckart–Young decomposition, Psychometrika 35 (3) (1970) 283–319.

[37] E.C. Chi, T.G. Kolda, On tensors, sparsity, and nonnegative factorizations, SIAM J. Matrix Anal. Appl. 33 (4) (2012) 1272–1299.

[38] B.W. Bader, R. Harshman, T.G. Kolda, et al., Temporal analysis of semantic graphs using ASALSAN, in: 2007 Seventh IEEE International Conference on Data Mining, ICDM 2007, IEEE, Los Alamitos, CA, USA, 2007, pp. 33–42.

[39] B.W. Bader, T. Kolda, et al., MATLAB tensor toolbox version 2.5, ⟨http://www.sandia.gov/~tgkolda/TensorToolbox⟩, 2012 (accessed: December 2012).

[40] J. Sun, Incremental tensor analysis, ⟨http://www.dasfa.net/wiki/index.php?title=Jimeng_Sun⟩, 2012 (accessed: December 2012).

[41] C.A. Andersson, R. Bro, The n-way toolbox for MATLAB, Chemom. Intell. Lab. Syst. 52 (1) (2000) 1–4.

[42] A.P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, Pattern Recognit. 30 (7) (1997) 1145–1159.

[43] D.L. Buckeridge, H. Burkom, M. Campbell, W.R. Hogan, A.W. Moore, Algorithms for rapid outbreak detection: a research synthesis, J. Biomed. Inf. 38 (2) (2005) 99–113.

[44] K. Barbieri, O.M. Keshk, B.M. Pollins, Trading data evaluating our assumptions and coding rules, Confl. Manag. Peace Sci. 26 (5) (2009) 471–491.

[45] K. Barbieri, O. Keshk, Correlates of war project trade data set codebook, version 3.0., ⟨http://correlatesofwar.org⟩, March 2012.

[46] H. Fanaee-T, M. Oliveira, J. Gama, S. Malinowski, R. Morla, Event and anomaly detection using tucker3 decomposition, in: European Conference on Artificial Intelligence—Ubiquitous Data Mining Workshop (UDM 2012), 2012, pp. 8–12.

[47] U.D. of Transportation, U.S. international air passenger and freight statistics report, June 2013, ⟨http://www.dot.gov/policy/aviation-policy/us-international-air-passenger-and-freight-statistics-report⟩.

[48] CapitalBikeShare, Capital bikeshare trip history data, ⟨http://capitalbikeshare.com/trip-history-data⟩, March 2013.

[49] Hubway, Hubway data visualization challenge, ⟨http://hubwaydatachallenge.org⟩, June 2013.

[50] T.U.K. Archive, Kdd cup 99, ⟨http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html⟩, December 2014.

[51] C.D. of Transportation, Pems traffic volume datasets, ⟨http://pems.dot.ca.gov⟩, December 2014.

[52] H. Fanaee-T, J. Gama, Event labeling combining ensemble detectors and background knowledge, Prog. Artif. Intell. 2 (2–3) (2014) 113–127.

[53] M. Tavallaee, E. Bagheri, W. Lu, A.-A. Ghorbani, A detailed analysis of the kdd cup 99 data set, in: 2009 Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications, 2009.

[54] V. Acharya, T. Philippon, M. Richardson, N. Roubini, The financial crisis of 2007–2009: causes and remedies, Financ. Mark. Inst. Instrum. 18 (2) (2009) 89–137.

[55] Boston.com, American joins long list of airline bankruptcies, ⟨http://www.boston.com/business/articles/2011/11/29/american_joins_long_list_of_airline_bankruptcies⟩, November 2011.

[56] U. Kang, E.E. Papalexakis, A. Harpale, C. Faloutsos, Gigatensor: scaling tensor analysis up by 100 times—algorithms and discoveries, in: The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, 12–16 August 2012, 2012, pp. 316–324, http://dx.doi.org/10.1145/2339530.2339583.

[57] J. Dean, S. Ghemawat, Mapreduce: simplified data processing on large clusters, Commun. ACM 51 (1) (2008) 107–113.

[58] A.L.F. de Almeida, A. Y. Kibangou, Distributed large-scale tensor decomposition, in: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, 4–9 May 2014, 2014, pp. 26–30, http://dx.doi.org/10.1109/ICASSP.2014.6853551.

[59] E.E. Papalexakis, C. Faloutsos, N.D. Sidiropoulos, Parcube: sparse parallelizable tensor decompositions, in: Machine Learning and Knowledge Discovery in Databases, Springer, Heidelberg, Germany, 2012, pp. 521–536.

[60] D. Chen, X. Li, L. Wang, S. Khan, J. Wang, K. Zeng, C. Cai, Fast and scalable multi-way analysis of neural data, IEEE Trans. Comput. PP (99) (2014) 1, http://dx.doi.org/10.1109/TC.2013.2295806.

[61] A.H. Phan, A. Cichocki, PARAFAC algorithms for large-scale problems, Neurocomputing 74 (11) (2011) 1970–1984, http://dx.doi.org/10.1016/j.neucom.2010.06.030.

[62] A. Cichocki, Tensor networks for big data analytics and large-scale optimization problems, CoRR abs/1407.3124, URL ⟨http://arxiv.org/abs/1407.3124⟩.

[63] N. Lee, A. Cichocki, Very large-scale singular value decomposition using tensor train networks, CoRR abs/1410.6895, URL ⟨http://arxiv.org/abs/1410.6895⟩.

Hadi Fanaee-T received his Ph.D. degree from University of Porto, Portugal in 2015 and M.Sc. degree from Shiraz University, Iran in 2010. Currently he is a Postdoc researcher in European FP7 project "MAESTRA" at Laboratory of Artificial Intelligence and Decision Support, University of Porto. His main research interests are data mining using tensor decompositions and application of tensor-based methods in real-world problems. He has served as Senior PC member for IJCAI2015, PC Member for ECML-PKDD 2013–2015 and also reviewer for DAMI, MACH, ITKD, KBS, IDA, CSUR, FSS and many more scientific venues.

Joao Gama is an associate professor at the University of Porto and a senior researcher at LIAAD/INESC TEC. He received his Ph.D. degree in computer science from the University of Porto, Portugal. His main interests are in machine learning and data mining, mainly in the context of data streams. He published more than 200 papers in major international conferences and journals and served as chair at ECML05, DS09, ADMA09, IDA11, and ECMLPKDD 2015. He co-organized a series of workshops on learning from data streams in conjunction with ECML-PKDD, KDD, SAC, and ICML. He is a member of the editorial board of MLJ, DAMI, NGC, and PAI; and he is the author of a recent book in Knowledge Discovery from Data Streams.