Tensor-based Event Detection

Hadi Fanaee Tork
Doctoral Program in Informatics at
University of Porto, Aveiro and Minho
Department of Computer Science
2015

Supervisor
João Gama, Associate Professor, Faculty of Economics
Hadi Fanaee Tork

Tensor-based Event Detection

Department of Computer Science
Faculty of Science of University of Porto
August 2015
Hadi Fanaee Tork

Tensor-based Event Detection

Submitted in partial fulfillment of the requirements for the Degree of Doctor of Philosophy in Informatics at the Faculty of Science of the University of Porto

Supervisor: Prof. Dr. João Manuel Portela da Gama

Department of Computer Science
Faculty of Science of University of Porto
August 2015
Acknowledgements

This work could not have been accomplished without the support and assistance from many people, institutes, friends and family members.

I acknowledge the financial support of European Regional Development Fund (ERDF) through the "COMPETE Program", North Portugal Regional Operational Program through the "ON.2 O Novo Norte program" and the national funds, through the Portuguese funding agency, Fundacao para a Ciencia e a Tecnologia (FCT) within the projects FCOMP-01-0124-FEDER-022701, NORTE-07-0124-FEDER-000059 and NORTE-07-0124-FEDER-000056. This work is also supported by by the Institute for Systems and Computer Engineering of Porto (INESC TEC) within project TNET, grant number BL/120033/PEst/LIAAD and funds from European Commission through the European FP7 project MAESTRA (Grant number ICT-750 2013-612944).

First and foremost I would like to express my sincerest gratitude to my advisor, Prof. Joao Gama, the continuous supporter of my PhD study for his patience, enthusiasm and knowledge. His guidance helped me in all aspects of my research and writing of this thesis. Without his guidance and constant help this dissertation would not have been possible. I could not have imagined having a better and friendlier advisor for my PhD study.

My special thanks goes to Prof. Luis Barbosa, 2010-2011 director of MAP-i program for facilitating my admission process and his rapid support regarding the issues I encountered in moving to Portugal. I also thank University of Aveiro for providing me the fellowship for my first year of study.

I would like to express my appreciation to my host institution during my PhD, the Laboratory of Artificial Intelligence and Decision Support (LIAAD/INESC TEC), for providing a vibrant and enthusiastic working environment. I deeply thank the help of my advisor Prof. Joao Gama and Prof. Alipio Jorge, the head of LIAAD for providing the financial support for my PhD study. I thank Luis Gama for proofreading of the thesis. I thank my colleague and friend Marcia Oliveira who sincerely shared her great knowledge about tensors in the first days I started my PhD in LIAAD. Without her I never had a rapid start on the topic. I thank Prof. Pavel Brazdil, spiritual father of LIAAD who reputation of today’s LIAAD greatly owes to him. I thank Carlos Ferreira for organizing LIAAD seminars which had a great influence on updating my knowledge during the PhD. I thank my other labmates: Mohammadreza Valizadeh, Zezito Tavares, Salisu Abdul, Ezilda Almeida, Petr Kosina, Hanen Borchani, Inigo
Mendialdua, Luis Matias, Elaine Faria and Jose Paiva for the days we were working together, and for all the fun we have had. I also thank LIAAD secretaries, Pedro Almeida and Ana Paula Silva for their friendly support.

I acknowledge Chris Holben, the Bike sharing Project Manager and Kim Lucas, the Bicycle Program Specialist from District Department of Transportation, Washington D.C, USA for their help and feedback, as well as Capital Bike Sharing for providing data.

I would like to thank my friends, Hamed Akhavan and Ali Azarian for their sincere and friendly support in my first arrival days in Porto and other friends for their great company.

Last but not the least, I would like to thank my parents, for giving birth to me and raising me at the first place and supporting me spiritually throughout my life.
"Everything we see is a perspective, not the truth"
— Marcus Aurelius (Roman Emperor, 121-180 AD)

The design is adapted from work of Trevor Shannon, entitled "MIT Shadow Sculpture"
©2010 Creative Commons - Attribution - Share Alike license
http://www.thingiverse.com/thing:3252
Abstract

An event corresponds to a time interval where we observe an unexpected behavior in the system. Event detection is an automatic procedure for discovering such time instants in the system and alerting the corresponding operators about emerging or occurred events in order to prevent further escalation and losses. Depending on the application domain, the detection process can be carried out prospective or retrospective. The objective of retrospective systems is to discover unusual patterns from an offline set of data as accurate as possible (maximizing true detection while minimizing false discovery). This provides the system operators the opportunity to efficiently perform preventive tasks to avoid similar abnormal situations in the future. On the other side, prospective systems operate on stream of data for prediction of events. The real-time nature of these systems requires that the delay factor to be considered carefully in addition to the accuracy factor. In order to have an accurate and fast event detection system we need to monitor many potential signals along with the existing inner/outer correlations among them which is a challenging problem. Tensors can be one of the powerful models for analysis of complex data. In particular, tensor factorization transforms complex data into a concise representation providing new insights about hidden knowledge and complex relationships between data items. In recent decades, such a property is being widely used in many disciplines with the aim of improving the performance of anomaly detection systems. The main assumption of these approaches is that abnormal states that are invisible in the multiway data can be uncovered if get decomposed in a true order. If data has a tensor (multiway) structure, such anomalies can remain invisible via matrix factorization techniques that flatten data into one of its ways (modes or dimensions). Nowadays, many systems generate multiway data. The simplest scenario is a system that generates a multivariate data with spatial and temporal information. For instance, in syndromic surveillance multiple features are measured in each geographic region over different periods of time. A bit different from this kind is time-changing network data. For instance, traffic dynamic (Origin × Destination × Time) is a multiway data where
relations between origin/destination constitute the first two dimensions and time forms
the third mode.

This thesis addresses the problem of event detection from multidimensional data with
emphasis on tensor-based techniques. We survey the literature and classify the existing
works based on factors such as learning method, decomposition technique. Later, we
extract the important issues and corresponding solutions. According to this knowledge
base we present the thesis contributions which are threefold. In the first contribution
we propose a new online/semi-supervised method for event detection from multi-
variate/spatiotemporal tensors with direct applications for syndromic surveillance.
In the second contribution we propose a new retrospective/unsupervised model for
event detection from traffic tensors. We present a new hybrid model for tensor-based
event detection, which is able to accurately detect meaningful events in traffic data.
In the third contribution, we present a new multi-aspect-streaming tensor analysis
approach and show its application to anomaly and event detection. Additionally, in
two independent works we address two important sub-problems arisen from the first
collection: hotspot detection and event labeling which are important parts of the
event detection procedure in online/semi-supervised scenarios.

The evaluation of the proposed methods and algorithms with benchmark problems,
simulated data and various real-world data sets shows promising results. For instance,
we outperform the detection quality of the state-of-the-art syndromic surveillance
system. Our approach not only is faster but also signals fewer false alarms comparing
the state-of-the-art. Our hotspot detection algorithm outperforms the state-of-the-art
in some circumstances and is much more efficient. The time complexity of our approach
is $O(N^2)$ which is considered a tremendous improvement over $O(N^4)$ of the state-of-
the-art. Our approach for event labeling promises a new type of automatic labeling
systems. We show how ensemble learning and background knowledge sources such as
the Web can come to the aid of labeling task which is in principle a time-consuming
and expensive task. Our hybrid model for event detection from traffic tensors also has
10% better detection power than naive tensor models. Our multi-aspect-streaming
tensor analysis approach outperforms the state-of-the-art both in terms of runtime
and memory usage. The proposed approach also exhibits a promising performance on
the real-life datasets.
Resumo

Um evento corresponde a um intervalo de tempo, durante o qual é observado um comportamento inesperado no sistema. A deteção de eventos é um processo automático usado para descobrir não só os anteriormente referidos intervalos de tempo em sistemas, mas também para alertar os operadores sobre eventos emergentes ou já ocorridos de maneira a prevenir o escalar da situação e outros danos. Dependendo do domínio da aplicação, o processo de deteção pode ser levado proativamente ou retrospectivamente. Os sistemas retrospectivos têm como objetivo a descoberta de padrões inominadas a partir de um set offline de dados, cuja taxa de precisão tem de ser a máxima possível (maximizando a deteção de eventos verdadeiros e minimizando a falsa descoberta de eventos). Isto cria a oportunidade para os operadores eficientemente realizarem tarefas preventivas de maneira a evitar o aparecimentos de situações similares no futuro. Por outro lado, os sistemas prospetivos operam sobre fluxos de dados para a deteção de eventos. Estes sistemas que operam em tempo real precisam de considerar cuidadosamente o fator de atraso em adição ao fator de precisão. Para obtermos um processo rápido e preciso para a deteção de eventos precisamos de monitorizar diversos potenciais sinais tal como as correlações interiores e exteriores entre eles, que é um problema desafiante. Os tensores podem ser um poderoso modelo para a análise de dados complexos. Em particular, a fatorização dos tensores transforma dados complexos numa representação concisa proporcionando outro ponto de vista sobre dados escondidos e relações complexas entre itens de dados. Nas últimas décadas, esta propriedade foi vastamente usada em várias áreas com o intuito de melhorar o rendimento de sistemas de deteção de anomalias. Esta técnica assume que estados anormais que são invisíveis em dados multi-direcionais podem ser descobertos se decompostos na sua ordem correta. Se nos dados há uma estrutura (multi-direcional) de tensores, as anomalias vão permanecer invisíveis via técnicas de fatorização de matrizes, que compressam os dados em uma das direções (modos ou dimensões). Hoje em dia temos muitos sistemas a gerar dados multi-direcionais. O cenário mais simples é o de um sistema que gera dados multi-variados com informação
espaço-temporal. Por exemplo, na vigilância sindromática várias propriedades são medidas em cada localização e em diferentes períodos de tempo. Um pouco diferente deste tipo são os redes de dados que variam com o tempo. Por exemplo, a dinâmica de trâfego \((\text{Origin} \times \text{Destination} \times \text{Time})\) é data multi-direcional cujas relações entre origem e destino constituem as duas primeiras dimensões, a terceira é o tempo.

Esta tese direciona-se ao problema da deteção de eventos em data multi-direcional com foco em técnicas baseadas em tensores. Examinamos a biblioteca existente e classificamos a mesma baseado em fatores como o método de aprendizagem e técnicas de decomposição. Mais tarde, extraímos os problemas mais importantes e respetivas soluções. De acordo com esta base de conhecimento apresentamos as três contribuições desta tese. A primeira contribuição é a proposta de um novo método online e semi-supervisionado para a deteção de eventos em tensores multi-variados/espaço-temporais que tem aplicação direta em vigilância sindromática. A segunda contribuição é um modelo retrospetivo sem supervisão para deteção de eventos em tensores de tráfego. Apresentamos um novo modelo híbrido para a deteção de eventos, baseado em tensores e que encontra com precisão eventos importantes em dados de tráfego. Na terceira contribuição é apresentada uma aproximação analítica a tensores de fluxos de vários aspetos e mostramos a sua aplicação na deteção de eventos. Adicionalmente, em dois trabalhos independentes, foram focados dois sub-problemas nascidos da primeira contribuição: deteção de hotspots e rotulagem de eventos que são componentes importantes para a deteção de eventos em cenários online semi-supervisionados.

A avaliação dos métodos e algoritmos com problemas de referências propostos, dados simulados e vários conjuntos de dados do mundo real mostraram resultados promissores. Por exemplo, ultrapassamos a qualidade de deteção de eventos do melhor sistema de vigilância sindromática. A nossa abordagem não só é mais rápida mas também assinala menos alarmes falsos, comparativamente ao melhor sistema. O nosso algoritmo de deteção de hotspots supera o melhor existente em algumas circunstâncias e é muito mais eficiente. A complexidade temporal da nossa abordagem é \(O(N^2)\) que é considerado um grande avanço face ao do melhor método: \(O(N^4)\). Quanto à rotulagem de eventos, a nossa abordagem promete um novo tipo de um sistema de rotulagem automático. Mostramos que a aprendizagem por conjuntos e fontes de conhecimento profundo tal como a Web podem ser um grande auxílio na tarefa de rotular que é a partida uma tarefa cara e demorada. O nosso modelo híbrido para a deteção de eventos a partir de tensores de tráfego observa uma taxa de 10% melhor poder de deteção do que modelos tensoriais sem fontes de conhecimento. A nossa análise de tensores de fluxos de vários aspetos supera a melhor existente em termos de tempo de
execução e uso de memória. A abordagem proposta também mostra bastante potencial em conjuntos de dados da vida real.
# Contents

Abstract

Resumo

List of Tables

List of Figures

## 1 Introduction

1.1 Motivation ................................................. 1

1.2 Objectives ................................................ 2

1.3 Contributions ............................................. 4

1.3.1 Application ........................................... 5

1.3.2 Learning ............................................... 5

1.3.3 Addressed issues ...................................... 5

1.3.3.1 Tensor rank estimation .............................. 6

1.3.3.2 Processing .......................................... 6

1.3.3.3 Seasonality ......................................... 6

1.3.3.4 Adaptivity .......................................... 6

1.3.3.5 Hybrid model ....................................... 7

1.3.3.6 Hotspot detection .................................. 7
1.3.3.7  Baseline data generation  ............................................. 7
1.4  Publications  ................................................................. 8
1.5  Thesis structure  ............................................................. 9

2  Literature Review  .............................................................. 12

2.1  Chapter introduction  ....................................................... 12
2.2  Preliminary Concepts  ...................................................... 13
  2.2.1  Event and its relation to change, anomaly and outlier  ........ 13
  2.2.2  Event Detection  ....................................................... 14
  2.2.3  Taxonomy of Methods for Event Detection  ..................... 14
  2.2.4  Notations  ............................................................. 16
  2.2.5  Matrix Decomposition  .............................................. 16
  2.2.6  Tensor  .............................................................. 16
  2.2.7  Slice  .............................................................. 16
  2.2.8  Tensor Analysis  ..................................................... 17
  2.2.9  Tucker Decomposition  ............................................. 17
2.3  Tensor-based Anomaly Detection: History and Applications  .... 18
  2.3.1  Process control  ..................................................... 20
  2.3.2  Environmental monitoring  ........................................ 20
  2.3.3  Video surveillance  .................................................. 21
  2.3.4  Network security  ................................................... 21
  2.3.5  Social networks  ..................................................... 22
  2.3.6  Text-based systems  ............................................... 22
  2.3.7  Neuroscience  ........................................................ 23
  2.3.8  Remote sensing  ..................................................... 24
  2.3.9  Sensors  ............................................................ 24
2.3.10 Engineering ................................................. 25
2.3.11 Transportation systems .......................... 25
2.3.12 Medical applications .......................... 26
2.3.13 Other applications ................................... 26
2.4 Tensorial Learning Strategies ......................... 27
  2.4.1 Supervised models .................................. 27
    2.4.1.1 Tensor decomposition for dimensionality reduction 29
    2.4.1.2 Tensor classifiers ................................ 30
    2.4.1.3 Tensor regression ................................ 30
    2.4.1.4 Tensor forecasting .......................... 31
  2.4.2 Semi-supervised models ......................... 32
  2.4.3 Unsupervised models .......................... 32
    2.4.3.1 Score plot-based .......................... 32
    2.4.3.2 Streaming decomposition error-based .......... 33
  2.5 Tensor decomposition methods ..................... 33
    2.5.1 PARAFAC-based ................................ 33
      2.5.1.1 PARAFAC ................................ 33
      2.5.1.2 Non-negative PARAFAC .................. 33
      2.5.1.3 PARAFAC2 ................................ 35
      2.5.1.4 Dynamic PARAFAC .................. 35
      2.5.1.5 Poisson tensor factorization .......... 36
    2.5.2 Tucker-based ................................ 36
      2.5.2.1 Tucker1 ................................ 36
      2.5.2.2 GTucker2 ................................ 36
      2.5.2.3 Tucker3 ................................ 36

XII
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5.2.4</td>
<td>Non-negative Tucker</td>
<td>37</td>
</tr>
<tr>
<td>2.5.2.5</td>
<td>HOSVD</td>
<td>37</td>
</tr>
<tr>
<td>2.5.3</td>
<td>ICA-based</td>
<td>37</td>
</tr>
<tr>
<td>2.5.4</td>
<td>DEDICOM-based</td>
<td>37</td>
</tr>
<tr>
<td>2.5.5</td>
<td>Bayesian methods</td>
<td>38</td>
</tr>
<tr>
<td>2.5.6</td>
<td>Locality preserving based methods</td>
<td>38</td>
</tr>
<tr>
<td>2.5.7</td>
<td>Tensor rank estimation</td>
<td>39</td>
</tr>
<tr>
<td>2.5.7.1</td>
<td>Cumulative sum of the percentage of eigenvalues or explained variance</td>
<td>39</td>
</tr>
<tr>
<td>2.5.7.2</td>
<td>Cross-validation</td>
<td>39</td>
</tr>
<tr>
<td>2.5.7.3</td>
<td>CORCONDIA</td>
<td>41</td>
</tr>
<tr>
<td>2.5.7.4</td>
<td>DIFFIT</td>
<td>41</td>
</tr>
<tr>
<td>2.5.7.5</td>
<td>Multiway scree plot</td>
<td>42</td>
</tr>
<tr>
<td>2.5.7.6</td>
<td>Split-half analysis</td>
<td>42</td>
</tr>
<tr>
<td>2.5.7.7</td>
<td>Other methods</td>
<td>42</td>
</tr>
<tr>
<td>2.6</td>
<td>Issues</td>
<td>43</td>
</tr>
<tr>
<td>2.6.1</td>
<td>Data pre-processing</td>
<td>43</td>
</tr>
<tr>
<td>2.6.2</td>
<td>Processing types : Offline/Online/Streaming</td>
<td>43</td>
</tr>
<tr>
<td>2.6.3</td>
<td>Tensor dimensionality</td>
<td>45</td>
</tr>
<tr>
<td>2.6.4</td>
<td>Nonlinearity</td>
<td>45</td>
</tr>
<tr>
<td>2.6.5</td>
<td>Seasonality</td>
<td>46</td>
</tr>
<tr>
<td>2.6.6</td>
<td>Unequal-length slices</td>
<td>46</td>
</tr>
<tr>
<td>2.6.7</td>
<td>Scalability</td>
<td>47</td>
</tr>
<tr>
<td>2.6.8</td>
<td>Adaptivity</td>
<td>48</td>
</tr>
<tr>
<td>2.6.9</td>
<td>Multi-scale anomalies</td>
<td>49</td>
</tr>
</tbody>
</table>
2.6.10 Hybrid models .......................... 50
2.6.11 Data fusion ............................. 50
2.6.12 Noise removal .......................... 51
2.7 Evaluation .................................. 51
2.8 Tensor software ............................. 51
2.9 Conclusion ................................. 52

3 Online/Semi-supervised Method .............. 53
3.1 Introduction .................................. 53
3.2 Proposed method ............................ 58
  3.2.1 The intuitions ............................ 58
3.2.2 Proposed Algorithm: EigenEvent ............. 60
  3.2.2.1 Data Processing and Decomposition ......... 60
  3.2.2.2 Subspace Matching ...................... 61
  3.2.2.3 Indicator Selection ...................... 62
  3.2.2.4 Dynamic Baseline Tensor ................. 63
  3.2.2.5 Updating Step .......................... 66
3.3 Evaluation ................................. 66
  3.3.1 Data set ................................. 66
  3.3.2 Performance ............................. 67
  3.3.3 Runtime .................................. 70
  3.3.4 Leading Indicators ....................... 71
  3.3.5 Baseline Selection ....................... 72
3.4 Conclusion and future works .................. 73

4 Subproblem: Hotspot Detection ............... 77

XIV
4.1 Introduction .................................................. 78
4.2 The proposed approach ...................................... 80
  4.2.1 The problem .............................................. 80
  4.2.2 The intuitions ............................................ 81
  4.2.3 The EigenSpot Algorithm ............................... 82
  4.2.4 Illustrative Example .................................... 84
4.3 Experimental Evaluation ................................... 85
  4.3.1 Simulation Study ......................................... 86
    4.3.1.1 Data Generation ................................. 86
    4.3.1.2 Performance Evaluation ....................... 87
    4.3.1.3 Effect of hotspot size and impact on the performance . 89
    4.3.1.4 The effect of SVD Implementation ............. 91
  4.3.2 Experiment with real data ............................. 92
4.4 Conclusion and future works .............................. 94

5 Subproblem: Event Labeling .................................. 96
  5.1 Introduction .............................................. 96
  5.2 Proposed Solution .................................... 99
    5.2.1 Definitions ......................................... 99
    5.2.2 Proposed Event Labeling System .................. 100
  5.3 Experimental Evaluation ................................ 101
    5.3.1 Data Set .............................................. 102
    5.3.2 The Proposed Method .............................. 105
      5.3.2.1 Event Detectors .............................. 108
      5.3.2.2 Detectors Settings ............................ 111
      5.3.2.3 Fleiss’ Kappa Test ........................... 111

XV
7.3.2.5 Software ......................................................... 163
7.3.2.6 Evaluation process ............................................. 163

7.4 Result and discussion .............................................. 164
  7.4.1 Correctness of the evaluation methodology .................. 165
  7.4.2 Evaluation on non-evolving modes ............................ 165
    7.4.2.1 ITA algorithms ........................................... 165
    7.4.2.2 MASTA .................................................. 166
  7.4.2.3 Online vs. offline histogram calculation ................. 167
  7.4.2.4 Effect of approximation on the accuracy ................. 167
  7.4.3 Evaluation of the evolving mode ............................. 167
  7.4.4 Sensitivity Analysis ......................................... 168
  7.4.5 Computational complexity ................................... 169
    7.4.5.1 OHTA .................................................. 169
    7.4.5.2 MASTA ................................................ 170
  7.4.6 Empirical efficiency ......................................... 170

7.5 Case studies ..................................................... 172
  7.5.1 Event detection in trade data set ........................... 172
  7.5.2 Event detection in video data set ........................... 173

7.6 Conclusion ....................................................... 174

8 Conclusion and Future Works ......................................... 177

  8.1 EigenEvent ....................................................... 177
  8.2 EigenSpot ........................................................ 178
  8.3 Event Labeling System ......................................... 179
  8.4 HTM ............................................................... 179
  8.5 MASTA ........................................................... 180

XVIII
8.6 Future trends ................................................. 181

References ......................................................... 183
List of Tables

1.1 Comparative summary of contributions .......................... 11

2.1 Applications of Tensors in Anomaly Detection ................. 19
2.2 Learning Strategies for Tensor-based Anomaly Detection .......... 28
2.3 Methods for Tensor Decomposition and Applications to Anomaly Detection 34
2.4 Methods for Tensor Rank Estimation ................................ 40
2.5 Important Issues and Solutions in Tensor-based Anomaly Detection .. 44

3.1 Characteristic of CityBN benchmark data ........................ 67
3.2 Evaluation comparison results ..................................... 70
3.3 Runtime comparison results ....................................... 70
3.4 False alarm results in details ...................................... 75
3.5 Detection delay in details ......................................... 76

4.1 Eigenspot performance evaluation with simulation ............... 87
4.2 The effect of hotspot size and impact on the performance (One-way ANOVA test) .............................................. 89
4.3 Effect of SVD implementation ..................................... 92
4.4 Comparison of STScan versus EigenSpot in detection of real hotspots 93

5.1 Event Detectors Settings ........................................... 112
5.2 Fleiss’es Kappa Interpretation ..................................... 113

XX
5.3 Predictive Models Tested .................................................. 114
5.4 Detected Events after verification phase by background knowledge ... 115
5.5 Ensemble Detectors Retrieval Performance in the absence of background knowledge .................................................. 118
5.6 Individual Detectors Retrieval Performance in the absence of background knowledge .................................................. 118
5.7 Area under ROC Curves for each individual detectors ................. 119

6.1 Evaluation with simulated data sets in presence of different event types 138
6.2 Evaluation with simulated data sets for three data representation scenarios .................................................. 140
6.3 Real-world data sets used in the experiments ................................. 140
6.4 Results for Bike sharing data sets ............................................. 142
6.5 Results for trade data set .................................................... 143
6.6 Results for US flight data set ................................................ 144

7.1 Datasets ........................................................................ 159
7.2 Descriptive statistics of data sets ............................................ 160
7.3 Results ........................................................................... 164
7.4 Space-time complexity ...................................................... 171
7.5 Detection of global financial crisis 2007-2009 ........................... 172
List of Figures

1.1 Structure of Contributions ...................................... 9
2.1 Tucker3 Decomposition ....................................... 17
3.1 Syndromic Surveillance Data Example ..................... 54
3.2 The environmental setting affects the data items .......... 57
3.3 Framework of EigenEvent .................................. 58
3.4 Eigenspace Tracking in EigenEvent ......................... 60
3.5 Illustrative example of dynamic baseline tensor creation process .... 65
3.6 An strategy for evaluation of EigenEvent .................. 68
3.7 AMOC Curve for EigenEvent vs. WSARE .................... 69
3.8 Effect of indicators on the performance .................... 71
3.9 Dynamic baseline vs. Environmental matching baseline .... 72
3.10 Dynamic baseline vs. Historical baseline ................. 73
4.1 Illustrative example of EigenSpot .......................... 84
4.2 Accuracy Box plot: EigenSpot vs. STScan .................. 89
4.3 The effect of hotspot impact and size on the performance of Eigenspot and STScan ......................... 90
4.4 Detected hotspot via STScan and EigenSpot ................ 92
5.1 Event labeling models ..................................... 100

XXII
5.2 Data set in daily scale .................................................. 103
5.3 Data set in hourly scale .................................................. 104
5.4 The proposed event labeling system ................................. 105
5.5 Event understanding with querying Google ....................... 106
5.6 A criterion for estimation of weight of the event ............... 107
5.7 A general architecture of the ensemble detectors ............... 108
5.8 Effect of condition \( z - score \geq 2 \) on false alarm rate .......... 117

6.1 Event detection from traffic tensors: motivational example .... 124
6.2 Four major traffic data analysis techniques ....................... 126
6.3 Illustrative example of proposed method (HTM) .................. 129
6.4 Simulating artificial events ............................................. 136

7.1 Multi-aspect-streaming tensor analysis vs. streaming tensor analysis .. 149
7.2 Offline histogram-based tensor analysis (OHTA) ............... 152
7.3 MASTA Update process example ..................................... 156
7.4 Sensitivity of MASTA and OHTA to the input parameters .... 168
7.5 Computational efficiency of MASTA versus ITA ............... 171
7.6 Event detection from video data ..................................... 173
7.7 A guideline for choosing the appropriate tensor analysis solution . 175
Chapter 1

Introduction

In this chapter we identify the motivation, objectives and contributions of the thesis and highlight the novelties in terms of applications, learning strategy, and addressed issues. We then list the publications produced by this thesis and present an overview of chapters in a comparative scheme.

1.1 Motivation

The simplest scenario for event detection can be a direct observation of the system state. However, if the system state is not observable directly we need to continuously monitor some measurable parameters from the system. The basic approach is to measure a single parameter from the system (e.g. temperature) and signal an alarm when its value exceeds the predefined threshold. However, the main obstacle for this approach is that the monitoring system is highly dependent on the measuring device. If it fails, the monitoring system will fail as well. In such cases, monitor of multiple parameters might be a better approach, because when observing one parameter become impossible, its value can be estimated based on the values of other parameters (known as outer-correlation). For instance, [CDL+04a] demonstrated that a multivariate detector could achieve 1.5 days faster detection comparing to a univariate detector in the disease outbreak detection context.

Now suppose that one of the monitoring parameters is a phenomena that influences the rest of the parameters. For instance, we know that in many systems, geographic location plays an important role in influencing the system parameters. Hence, ge-
CHAPTER 1. INTRODUCTION

Oographic location in such cases should not be treated as a single system parameter. Such phenomena is usually interpreted as a mode (or dimension or way) of the system. If there exist multiple parameters in system that have such an influential role, the data being generated via this system is called multiway or tensor data. Several research has shown that taking into account the system behavior in its natural structure (i.e. tensor) provides better accuracy for identification of anomalies. The reason is that tensors capture both inner and outer correlations between parameters and therefore, provides more accurate modeling mechanism. However, using tensors has its own challenges and issues. The focus of this thesis is to study the capabilities of tensors, their advantages and their corresponding issues related to the event detection problem.

1.2 Objectives

In many domains such as syndromic surveillance (a sub-field in epidemiology) we encounter with multiple data streams from different sectors such as emergency departments, medication sales, etc. These streams which have extra dimensions such as location and time are simultaneously under seasonal effects and need to be monitored rapidly for event detection. One of the serious problems, in particular in disease surveillance is that the majority of the existing algorithms are sensitive to small changes and raise a high rate of false alarms. From our knowledge from tensors, we know that these approaches can be potential for this problem. Because, modeling data in the tensor structure help us to analyze each element of data along with other elements. Therefore, it is anticipated that tensors reduce the false discoveries as the serious issue in this domain. However, adapting tensors for seasonal environment is a challenge. Therefore, two questions should be addressed:

Q1 Can tensor approaches provide a better solution for syndromic surveillance?

Q2 How can we adapt tensor models for seasonal environments?

Presented work in Chapter 3 is an effort in answering the above questions. Nevertheless, right after that we face with two other sub-problems:

Q3 How can we identify the hotspots after we signal an event alarm?

Q4 How can we automatically (or semi-automatically) obtain a high quality baseline data, free of anomalies and outliers?
CHAPTER 1. INTRODUCTION

The baseline solution for the Q3 is space-time scan statistics (STScan) or its multivariate extensions. However, STScan has many problems that makes it inappropriate for real time settings. Firstly, it is a very expensive approach. It should exhaustively search the whole space for finding significant spatiotemporal clusters; Secondly, it is restricted, in the sense that it can be used only when extra dimensions are space and time. Even for spatiotemporal data makes some restrictive assumptions about data distribution, shape of hotspots and the quality of data, which can be unrealistic for some nontraditional data sources [Nei06]. In Chapter 4 we investigate the capability of SVD and eigenvector matching for solving this problem. Therefore, our main question in this chapter will be:

Q31 Whether SVD is helpful for hotspot detection? How is its accuracy and complexity comparing STScan?

Answering to Q4 has same importance as the Q3. Because, accuracy of the proposed approach is highly dependent on the quality of baseline data. Recently, ensemble approaches emerged for labeling anomalies in data. But these methods still require some manual efforts in terms of confirmation of labels. In Chapter 5 we investigate the usefulness of ensemble approaches and background knowledge. Thus, the research question would be:

Q41 Whether ensemble methods perform better than individual detectors? What about if we add some background knowledge to the ensemble detectors? Does this improve the accuracy of event labeling?

Note that in Chapters 4 and 5 due to more focus on the sub-problems and some limitations and requirements in evaluation, we used lower-order data in experiments. However, conclusions are valid for higher-order tensors in both contributions. For instance, in Chapter 4 by replacing SVD with HOSVD we can apply the method on the three-order tensor data. Likewise, in chapter 5 we can apply ensemble tensor models when data has multiway structure.

In chapter 6 we concentrate on the problem of event detection from traffic tensors which has lots of potential applications in network information systems and transportation systems. Our preliminary investigations indicate that tensor decomposition models in their simple scheme are not sensitive as expected, to capture some important topological fluctuations in traffic tensors. Therefore, the first question would be:
Q5 How can we solve the insensitivity of naive tensor models to the topological events? Basically, how can we improve the performance of tensor-based event detection techniques for traffic data?

In addition to this problem, choosing the tensor rank as a main issue is left without any attention in the majority of works. Usually, a constant number is chosen as the tensor rank which is highly subjective. Therefore, we should seek for an efficient and reliable technique for tensor rank estimation. This forms another research question:

Q6 Does automatic tensor rank estimation improve the quality of event detection?

Eventually, in Chapter 7 we address a more fundamental problem: streaming tensor analysis and its application for event and anomaly detection. There exist some streaming frameworks such as Incremental Tensor Analysis (ITA) [STP+08a] which are already applied to the event detection problem. Although ITA is capable to solve many streaming problems, it restricts the tensor growth only in time, which is a huge constraint in scalability and adaptability of other modes. Our objective is to investigate the usefulness of some other low-cost alternative approaches apart from the linear algebra based solutions. We investigate the application of histograms for tensor analysis. Therefore, the main question would be:

Q7 Can we adapt histograms for tensor analysis?

However, histograms are expensive methods which in practice cost even more than linear algebra techniques. Therefore, their use does not have economic justification, unless we use them in a streaming way. For that purpose, we should answer the following question:

Q8 Can we replace online histograms with offline histograms in tensor analysis? In that case, how much accuracy we loose and how much efficiency we obtain in exchange? What are the applications to anomaly and event detection?

1.3 Contributions

In this chapter we highlight the novelty of the thesis in terms of application domain, learning strategy, and open issues.
1.3.1 Application

We focus on two main applications in this thesis: disease outbreak detection and traffic data analysis. The former one has received a great attention by the community due to emerging infectious disease such as Ebola and Influenza. To the best of our knowledge, tensor-based approaches are not yet applied to this application. Hence, Chapter 3 contains novelty in terms of application domain.

The second major application, traffic data analysis is covered in Chapter 6. We study a delicate problem of tensor models in this domain. We show that tensor models are not sensitive as expected, to capture existing topological fluctuations in this data type. Chapter 6 attempts to address this problem for the first time. Therefore, what we address in this chapter is a novel contribution in traffic data analysis domain.

1.3.2 Learning

Tensorial learning as well as classical machine learning algorithms are categorized into three main classes: supervised, semi-supervised and unsupervised (See section 2.4). Our novel learning strategies in this thesis promises three categories in the existing solutions. In Chapter 3 we introduce a new kind of semi-supervised learning (i.e. section 2.4.2) which we call it Eigenspace based method. For the first time we use the properties of eigenvector/eigenvalue matching in tensor-based event detection; in chapter 6 we present a novel category of unsupervised methods (i.e. section 2.4.3) in the class of score-plot based techniques (i.e. section 2.4.3.1); and in Chapter 7 we propose a new streaming method for tensor analysis, which is considered a new category in unsupervised tensor learning.

1.3.3 Addressed issues

We address some important issues that so far have not been concerned. In the following we provide the novelty of our works in terms of addressed issues mentioned (See Table 2.5 for a list of issues).
1.3.3.1 Tensor rank estimation

Tensor rank estimation is barely used in automatic settings, in particular for event detection. But it has lots of potential applications if we manage to automatically determine the number of components for tensor decomposition model. In Chapter 6 we test some capable tensor rank estimation in automatic event detection settings, including FastDIFIT [KK03] and Maximum block improvement (MBI) [CLZ14] and show the promising property of MBI-based method.

1.3.3.2 Processing

In terms of processing, we examine three kinds of processing. In Chapter 3 we propose an online model with updating. This model is the first in its kind in terms of the methodology used. In Chapter 6 an offline model is proposed. This model is also distinctive since for the first time combines two data representation of data into a unified tensor model. Finally, in Chapter 7 a new streaming tensor analysis approach is proposed. The new technique is the first multi-aspect-streaming tensor analysis in the literature that allows tensor to evolve over all its modes.

1.3.3.3 Seasonality

In Chapter 3 we propose a sophisticated method for addressing seasonality issue in tensor-based event detection. We build a separated tensor for each environmental setting (e.g. Day = weekend, Weather = cold, Flu = high, Season = winter) and then these tensor models are used in a real time setting for event detection.

1.3.3.4 Adaptivity

Incremental tensor analysis approaches restrict the tensor growth only in time, which is a huge constraint in scalability and adaptability of other modes. Hence, these approaches are only useful for large, but slender tensors. In Chapter 7 we propose a new approach that relaxes this constraint and allows tensor to concurrently evolve through all modes.
1.3.3.5 Hybrid model

In Chapter 6 we present a new hybrid strategy for tensorial modeling of traffic data. We create two sets of same data set in two different representations and make a parallel model for each and then combine the model output of both for purpose of event detection.

1.3.3.6 Hotspot detection

In Chapter 4 we propose the first eigenspace-based hotspot detection algorithm that discover subgroup of data affected by events in an efficient way.

1.3.3.7 Baseline data generation

Semi-supervised methods like what we propose in Chapter 3 require a high quality baseline data for building the normal condition operation model. In Chapter 5 we propose a new semi-automatic labeling procedure by combining ensemble detectors and background knowledge.


1.4 Publications

The work presented in this thesis has produced the following list of publications and submissions.

A. Peer-reviewed journal papers


B. Peer-reviewed conference and workshop papers


C. Data Sets


1.5 Thesis structure

The hierarchical structure of the thesis contributions is presented in Fig. 1.1. In Chapter 2 an overview of related works is presented. In Chapter 3 we present our online/semi-supervised method named EigenEvent for tensor-based event detection. Chapters 4 and 5 respectively address two important sub-problems arisen by Chapter
3: hotspot detection and event labeling. In Chapter 6 we propose our hybrid tensor model for event detection from traffic tensors. In Chapter 7 we introduce our histogram-based solution for multi-aspect-streaming tensor analysis with applications to anomaly and event detection. Chapter 8 concludes the thesis with final remarks, conclusions and suggestions for future works.

Table 1.1 presents a summarized overview of the thesis contributions in a comparative format. Each chapter is compared with other chapters in terms of the principal issues in scientific research in computer science, such as application domain, learning strategy, processing type, fundamental techniques used in the method development, data sets tested and their dimensions, the compared methods and the final conclusions.
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Developed Algorithm</th>
<th>Application</th>
<th>Learning Type</th>
<th>Detection Techniques</th>
<th>Data dimension</th>
<th>Issues addressed</th>
<th>Data sets</th>
<th>Compared with</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>EigenEvent</td>
<td>Epidemiology</td>
<td>Semi-supervised</td>
<td>HOSVD + SVD</td>
<td>3D</td>
<td>Seasonality</td>
<td>CityNet (Benchmark)</td>
<td>WSARE 1, 2, 3</td>
<td>Less false alarms rate, better runtime</td>
</tr>
<tr>
<td>4</td>
<td>EigenSpot</td>
<td>Epidemiology</td>
<td>Semi-supervised</td>
<td>SVD</td>
<td>2D</td>
<td>Hotspot detection</td>
<td>Simulated and Real (Cancer in New Mexico)</td>
<td>Space-time Scan Statistics, Baseline method</td>
<td>Better time complexity and accuracy</td>
</tr>
<tr>
<td>5</td>
<td>ELS</td>
<td>Transportation</td>
<td>Semi-supervised + Unsupervised</td>
<td>Ensemble</td>
<td>1D</td>
<td>Baseline data labeling</td>
<td>RealBike-sharing Washington D.C.</td>
<td>Baseline method</td>
<td>Better accuracy</td>
</tr>
<tr>
<td>6</td>
<td>HTM</td>
<td>Traffic data</td>
<td>Unsupervised</td>
<td>ACS-Tucker</td>
<td>3D</td>
<td>Hybrid data model + tensor rank estimation</td>
<td>Simulated and RealFlight, Trade, Bike Washington D.C, Bike Boston</td>
<td>DIFFIT+ALS, DTA, PCA</td>
<td>Better accuracy</td>
</tr>
<tr>
<td>7</td>
<td>MASTA</td>
<td>Generic</td>
<td>Unsupervised</td>
<td>Online Histogram</td>
<td>3D, 4D</td>
<td>Adaptivity + Scalability</td>
<td>11 real data sets from 7 domains</td>
<td>DTA, STA, WTA</td>
<td>Better space/time complexity and run time</td>
</tr>
</tbody>
</table>

Table 1.1: Comparative summary of contributions
Chapter 2

Literature Review

Traditional spectral-based methods such as PCA are popular for anomaly detection in a variety of problems and domains. However, if data includes tensor (multiway) structure (e.g. space-time-measurements), some meaningful anomalies and events may remain invisible with these methods. Although tensor-based anomaly detection (TAD) has been applied within different disciplines over the last twenty years, it is not recognized yet as a formal category in anomaly detection. In this chapter, we provide a comprehensive overview of interdisciplinary works in which TAD is reported and characterize the learning strategies, methods and applications; extract the important issues in TAD and provide the current state of solutions according to the state-of-the-art. Through this chapter we highlight the existing gaps and open issues in TAD research which will be the basis of the thesis and focus of the remaining chapters.

2.1 Chapter introduction

Plenty of methods were developed during the last decades for anomaly detection in different domains. One group of methods are spectral techniques that attempt to project high dimensional data onto a lower subspace [CBK09]. The main assumption of these approaches is that normal and abnormal instances appear significantly different in the projected subspace [CBK09]. Spectral methods have been well researched and have traditionally concentrated on matrix-based techniques. However, in many real-world applications we deal with data with tensor (multiway) structure. In such circumstances, anomalies may remain invisible if we lean on the matrix-based modeling.
During the last twenty years, since the work of [NM94], a variety of methods that consider multiway structure of data have been developed in multiple disciplines from chemometrics and environmental monitoring to signal processing and data mining. Despite the popularity of this research area (though with different terminologies), no comprehensive survey on TAD is yet available. The reason, probably, is that TAD belongs to wide scopes and spans across different research fields.

Our main objective in this chapter is to study the literature from all major disciplines where tensors are frequently applied and classify the contributions related to TAD based on some factors such as applications, learning types, methods and evaluation metrics. Moreover, we identify and classify the important issues and proposed solutions in TAD research. We follow a motivational strategy in this chapter, in the sense that we do not limit ourselves to only techniques that are applied for anomaly detection. Rather, we include those methods that are applied in the close applications, such as classification, regression and forecasting that have a great potential for anomaly detection.

The chapter is organized as follows. In section 2.2 we present the preliminary concepts. In Section 2.3, we introduce the history of TAD and its applications. Section 2.4 presents learning methods for TAD. Section 2.5 discusses the techniques for tensor decomposition. Section 2.6 outlines the issues in TAD along with the corresponding solutions. In section 2.7 we discuss the evaluation metrics used in TAD. Section 2.8 lists the available software for tensor decomposition. Section 2.9 concludes the chapter.

2.2 Preliminary Concepts

This chapter contains the necessary materials and definition of preliminary concepts that are required for understanding the rest of thesis. Concepts such as event and event detection are first defined and then a summarized taxonomy of event detection methods will be demonstrated. Thereafter, the terms related to tensors as the main concentration of the thesis will be described.

2.2.1 Event and its relation to change, anomaly and outlier

An event corresponds to a temporary period where we observe an unexpected behavior in the system. This can be due to a natural phenomena or manual system interaction.
CHAPTER 2. LITERATURE REVIEW

Depending on the application it can be referred as terminologies such as fault, failure, damage and break in industrial settings; disease outbreak in epidemiology; traffic jam in transportation systems; intrusion in computer networks; malicious activity in social networks; fraud in financial transactions; crisis in economy; disease in medical applications; tsunami and earthquake in seismology; pollution and forest fires in environmental applications; and bioterrorist activities in national security.

There is not a mutual agreement on the definition of event, change, anomaly and outlier, such that most of the times, these concepts are used alternatively. However, to be concrete in this thesis we provide our interpretation from these concepts based on opinion of the majority.

Change is a time point that the system initiates to exhibit a different behavior comparing its previous state. The border between change and event is that change is a persistent shift in data when system is assumed that never backs to its previous state, at least within a short period of time, while in event we anticipate that system returns back to its normal state after it passes the critical conditions. Event is related to anomaly in the sense that event is a special case of anomaly where the context is defined in time. In other words, anomalies in time are usually called events. Outlier is also very close concept to anomaly, but is more used for static data where time is not involved.

2.2.2 Event Detection

Event detection is an automatic mechanism for uncovering significant and critical facts about the system behavior with the aim of prevention of further escalation and losses. Event detection has been identified as an inevitable function in surveillance systems due to its importance in morality saving and economical values. A recent study indicates that diagnosing and controlling abnormal situations has an economic impact of at least $10 billion annually in the United States [MFM+10].

2.2.3 Taxonomy of Methods for Event Detection

The majority of event detection approaches originate from disciplines such as time series analysis, quality control, signal processing, scan statistic and data mining. Based on the complexity of data, the existing approaches can be classified into three groups: univariate, multivariate and multidimensional which are briefly described in
the following.

1) Univariate methods that can be classified into three groups. The first group is based on Statistical Process Control (SPC) techniques. Examples are Shewhart chart [She31], Exponentially weighted Moving Average (EWMA) [LS90] and Cumulative-Sum Chart (CuSUM) [MPR08]. The second group of methods uses time series models such as regression [TWDC01], Moving Average(MA) [RPM03], Autoregressive moving average (ARIMA) [RM03], Singular Spectrum Analysis (SSA) [SGH11], Hidden Markov Model (HMM) [RCS03] and Exponential smoothing [NTH96]. Finally, the third category of methods uses signal processing techniques such as Wavelet [Shm05] and Kalman Filter [KS03].

2) Multivariate extension of univariate methods also exists so that these are used for event detection from multivariate data. For instance, Multivariate extension of SPC techniques are proposed for event detection. Examples are Hotelling $T^2$ [Dub11], Multivariate EWMA [JWRF08], Multivariate CuSUM [WN85] and Principal component Analysis (PCA) [BSM09]. Time series models are also extended for modeling multivariate data and are applied to anomaly detection problems. Examples are Vector Auto Regression (VAR) [GGF14], Vector Auto Regression Moving Average (VARMA) [DLL10], Multiple regression [BJY99] and Multivariate singular spectral analysis (M-SSA) [APR06].

3) Multidimensional methods are also threefold. The first group of methods is based on associate rule mining. They mine data for finding strange rules. Examples are Emerging Patterns [DL99], Search and Testing for Understandable Consistent Contrasts (STUCCO) [BP99] and What’s Strange About Recent Events (WSARE) [WMCW05]. The second group of methods is based on Scan-statistics and is specifically developed for disease outbreak detection where both dimensions of space and time are highly important. The idea of scan statistics is to exhaustively search the whole geo-temporal space with varying window sizes and compute some statistics inside the window and compare them to the rest of data. Examples are multivariate scan statistics [KMD+07], nonparametric scan statistic (NPSS) [NL07] and Bayesian multivariate scan statistics [NC10]. The third group of methods is tensors which is more generic and has wider applications. These methods are the focus of this thesis and will be discussed comprehensively in this chapter.
2.2.4 Notations

Following [KB09], throughout the thesis, scalars are denoted by non-bold lowercase letters (e.g. \(i\)), vectors are identified by boldface lowercase letters (e.g. \(a\)), matrices are designated by boldface capital letters (e.g. \(A\)) and tensors are denoted by calligraphic letters (e.g. \(\mathcal{X}\)).

2.2.5 Matrix Decomposition

Any given matrix \(A \in \mathbb{R}^{m \times n}\) can be decomposed uniquely via Singular Value Decomposition (SVD) procedure as \(A = UDV^T\) such that \(U\) is an \(m \times n\) matrix with orthogonal columns; \(D\) is a \(n \times n\) diagonal matrix; and \(V^T\) is an \(n \times n\) orthogonal matrix. The diagonal values of \(D\) are called the singular values of \(A\). The column vectors of \(U\) are the left singular vectors of \(A\). The column vectors of \(V\) are the right singular vectors of \(A\).

2.2.6 Tensor

Tensor is a geometric or mathematical object used to extend scalars, vectors and matrices to higher dimensions. A tensor is an array of form \(\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}\) where \(N\) is the number of orders or ways.

2.2.7 Slice

A slice is a \((d-1)\)-dimension partition of tensor where an index is fixed in one mode and the indices vary in the other modes. The horizontal, lateral, and frontal slides of a third-order tensor \(\mathcal{X}\), are denoted by \(X_{i,:}, X_{:,j}, \text{ and } X_{:,k}\), respectively. Each slice in each mode corresponds to an entity (or feature). For instance, in a three-order tensor of \(country \times year \times measurement\), the country "Portugal" is a feature in the first mode. The year 2014 is an entity in the second mode and "population" or "GDP" are the features in the third mode.
2.2.8 Tensor Analysis

Traditional data analysis techniques, such as the PCA, clustering, regression, etc. are only able to model two-way data and they do not consider the interaction between more than two modes. 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. Tensor (Multi-way) data 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 [Tuc66] and CP/PARAFAC [Har70], which are generalized versions of PCA or, more specifically, Singular Value Decomposition (SVD) for higher order matrices.

2.2.9 Tucker Decomposition

Tucker decomposition is an optimization task, through which a large tensor can be approximated by a product of a smaller tensor with predetermined dimensions (called core tensor), multiplied by factor matrices in each dimension (See Fig. 2.1). Formally, the problem can be defined as follows [CLZ14]: 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 $1 \leq r_i \leq n_i$ for $i=1,2,\ldots,d$, that optimizes

$$
\min \left\| \mathcal{X} - \mathcal{G} \times_1 \mathbf{A}^{(1)} \times_2 \mathbf{A}^{(2)} \times_d \mathbf{A}^{(d)} \right\| 
$$

(2.1)
Subject to:
\[ \mathcal{G} \in \mathbb{R}^{r_1 \times r_2 \times \ldots \times r_d}, \]
\[ \mathbf{A}^{(i)} \in \mathbb{R}^{n_i \times r_i}, \quad i = 1, 2, \ldots, d \]

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

The Tucker optimization becomes PARAFAC (or CP) decomposition if we have \( r_1=r_2=\ldots=r_d \) and core tensor \( \mathcal{G} \) is a cubical tensor with ones along the superdiagonal.

Assuming \( r_i \) as fixed constants, many algorithms are developed for solving the above optimization task. The better known algorithm is the alternating least squares (ALS) method which is used for both Tucker [KDL80] and PARAFAC [CC70, Har70]. Two other algorithms that are specifically developed for Tucker decomposition include High-Order SVD (HOSVD) [DLDMV00a] and High-Order Orthogonal Iteration (HOOI) [DLDMV00b]. HOSVD independently performs SVD on every metricized tensor in each mode, while HOOI performs alternating optimization to find better projection matrices iteratively. Therefore, HOSVD can be viewed as a special case of HOOI with one iteration only [Sun07, p. 134].

### 2.3 Tensor-based Anomaly Detection: History and Applications

The origin of the word "tensor" is a Latin tendere "to stretch" which appeared for the first time in anatomy in the seventeenth century to denote muscle’s stretch. It was later used by mathematician William Rowan Hamilton in the middle of the eighteenth century to describe some concepts in quaternion algebra. Tensor calculus, the more close the current meaning, was firstly introduced in 1900 by Italian mathematician Gregorio Ricci-Curbastro and his doctoral student Tullio Levi-Civita. In 1915, tensor was used by Albert Einstein in general relativity theory for explaining causal and geometric structure of space-time and definitions of concepts such as future, past, distance and volume. The first principles of tensor decomposition [KB09a] were founded by American mathematician Frank Hitchcock in 1927. Complex and multiway structure of human behaviors was the first motivation for use of tensors in data analysis. Psychologists such as Raymond Cattell, Ledyard Tucker and Richard
Harshman were pioneers in extending tensor decomposition applications in psychology during three decades from the 1940s to 1970s. In 1981, tensor decomposition was introduced by Appelhof and Davidson to the Chemometrics community. The first applications of tensors in anomaly detection appeared in this community almost a decade later. The work of [NM94] about tensor-based batch monitoring was pioneer in motivating tensor (multiway) methods in the monitoring and fault detection problems. The modern application of tensors in anomaly detection appeared a decade ago in a series of articles from Jimeng Sun [SPP06, STF06, STP+08b] who had a major contribution to the growth of TAD research in data mining community. Nowadays, application of TAD has been widespread to wider areas, including environmental monitoring, video surveillance, network security, social networks, text-based systems, neuroscience, remote sensing, engineering and other domains. In the following, some of these applications are discussed in more detail (See Table 2.1 for summary).
2.3.1 Process control

The first footprint of tensors as earlier mentioned is seen in the monitoring of batch processes. The common objective in operating batch processes is to achieve value-added products of high-quality with competitive prices. The goal of the batch process analysis is to understand the major sources of batch-to-batch variations [NM94], detection of faulty batches and based on that improve the operation policies.

Tensors are very popular monitoring techniques in production of chemicals and other manufacturing applications. Examples are polymerization processes [KPD+94, NM95b, KNM95, CY03], semiconductor etching process [GCL13, WGM01, WGB+99], manufacturing pharmaceutical materials [ZJ07, HY09], wastewater treatment [SBM+09], bioprocesses [ASC+08], fed-batch fermentation process [MY14b, LYL03, YLVL04, HY09], nuclear waste storage tank monitoring [GWS96] and winemaking process [UHR12].

In the majority of these applications, the typical tensor is a three-order tensor of I (batch) \( \times \) J (measurement) \( \times \) K (time) which usually is unfolded in batch or time mode. Therefore, usually the matrix of I \( \times \) JK or J \( \times \) IK is processed which is called respectively batch-wise and time-wise unfolded matrix. The main goal of tensor-based batch processing is to identify the abnormal batches or time instants.

2.3.2 Environmental monitoring

Thanks to the recent advances in sensor technologies, it is feasible to analyze tens of ecological parameters through different locations and times. The need for tensors emerged in the environmental domain, mainly due to existing spatiotemporal variations in such data. Identification of locations or time periods related to abnormal measurement is the main goal of this application. Tensors have been recently applied in water quality monitoring [BAP+02, SMS+06, SMBS07, DZZ+10, EGS+14, SMS+07], air pollution control [LLK+14, SS05] and monitoring of soil quality [SMSS06, AKCP07].

The multi-way data in these applications follow a general scheme of \( \text{variables} \times \text{sampling site} \times \text{sampling time} \) where the first dimension normally includes the chemical (e.g. oxygen rate), physical (e.g. temperature) and biological parameters (e.g. faecal coliforms) measured by the sensors.
2.3.3 Video surveillance

Identification of time instants in video surveillance cameras is of great interest in public security for prevention of terrorist/criminal activities. Tensors are natural data models for video data. Video data can be represented as a 4D tensor of \( RGB \text{ color} \times \text{image row} \times \text{image column} \times \text{time} \) or a 3D tensor of \( \text{image row} \times \text{image column} \times \text{time} \). The most relevant works that use tensor model for anomaly detection are [LHWG11, TNL12] who apply TAD in video surveillance cameras. [ZZA+13] also models 3D video as tensor for human action recognition. A tensor-based approach is proposed in [LWL10] for real-time tracking of moving points from infrared image sequences. Some other techniques [ZSH+11, HLZ+11, LS11] also use tensors for object tracking in video data which has the potential to be extended for anomaly detection purposes. Some methods like [ZZP12, KO09] exploit tensors respectively, for crowd density estimation and motion recognition that can be useful for anomaly detection.

2.3.4 Network security

Computer-based systems are vulnerable to various attacks and malicious activities. Anomaly detection in these networks has been for long years the center of attention by many researchers. Tensors are potential methods for anomaly detection in these networks. The reason is that a tensor can easily model the dynamic of traffic matrices that requires an extra dimension of time. Moreover, in network security application it is very difficult to obtain labels for abnormal situations. Usually only the history of normal operation is available. Therefore semi-supervised and unsupervised tools such as tensor decomposition might be adequate tools.

There is no unique tensor model for TAD in networks, but the majority of works use the \( \text{origin} \times \text{destination} \times \text{time} \) model. This tensor model is used in various network-based systems such as TCP/IP network, email network, phone calls, IP-TV network or World Wide Web (WWW). For instance, in TCP/IP network, two of the most popular models used are \( \text{SourceIP} \times \text{TargetIP} \times \text{Time} \) [APG+14, MWP+14, KLMW09, KLMW09, STF06, MG11] and \( \text{SourceIP} \times \text{TargetIP} \times \text{Port} \times \text{Time} \) [KPFI12, STF06]. In email or phone call networks the tensor models are constructed in the scheme of \( \text{Sender} \times \text{Recipient} \times \text{Time} \) [APG+14, PFS12, KS08, STF06, BHK07]. There exists another type of works that model the interaction of user with the system. Examples are \( \text{IP} \times \text{URL} \times \text{User} \times \text{Time} \) [MY14a] and \( \text{Users} \times \text{URL} \times \text{Time} \) [MSF+12] in web-access log data and \( \text{User} \times \text{TV Program} \times \text{Time} \) in IP-TV system [FTOG+12].
Anomaly detection from Internet networks is also addressed in [GCP10]. The authors propose a method based on tensor decomposition for finding the source of disturbances originated in the network elements in a large Internet network. A three-order tensor model of $VP \times AS \times time$ is introduced where VP denotes the vantage point and AS refers to a network element. The built model is then used to track large earthquakes occurred during the network activity.

### 2.3.5 Social networks

Social networks are a special case of general networks where nodes of networks are mostly human and the edges show the human interactions together. Tensors are normally used for the detection of anomalous people, links and communities which is obtained by taking into account their long term behavior over time. The general tensor model for this task is $person \times person \times time$. One of the popular tensor-based practices is related to the analysis of anomalies in electronic discussion networks such as ENRON data set [PFS12, KS08, STP+08b, STF06, BHK07, PL11, XYQ15]. Tensors are used in analysis of Facebook data [KPF12], phone calls [APG+14], location-based social networks ($user \times location \times time$) [PPF14] and analysis of physical social networks such as face-to-face contacts of individuals [GPC14]. Apart from the traditional model, [OG13] proposed new tensor models such as $nodes \times measures \times time$ and $communities \times measures \times time$ for dynamic social networks where measures such as betweenness, degree closeness are computed from social network in each snapshot.

### 2.3.6 Text-based systems

Tensors are used for modeling the user/topic evolution in text-based systems. The constructed models are later applied to event and anomaly detection or co-clustering. The general tensor model for textual data analysis is $user \times keyword \times time$. Such model is used for anomaly detection from Twitter data [PGQC14] and analysis of chatrooms [ACKY05] and bibliographic data ($author \times keyword \times time$) [KS08, STP+08b, STF06]. The tensor-based topic modeling techniques such as [MSF+12] also have potential for text-based event detection.
Brain is one of the complex systems that produces a rich source of multiway data. The reason is that every activity occurring in the brain is managed via different regions of the brain during a specific period of time. Therefore, brain data is inherently spatiotemporal. The two well-known tools for capturing the brain activities in a machine-readable format are Electroencephalography (EEG) signals or Functional magnetic resonance imaging (fMRI). The data being generated from these tools is analyzed via tensor models to detect abnormal activities in the brain. For instance, tensors are used to find the responsible regions in the brain that generate the abnormal neural activity resulting in the initial seizure discharge [AABB+07, CLW+15]. The information obtained from this analysis is very helpful for the success of an epilepsy surgery. Different from the above-mentioned application, tensors are used for mental workload monitoring of operators in safety-critical applications (e.g. controlling the Unmanned Air Vehicle (UAV) [RTN09]).

The general tensor model for EEG data is $\textit{frequency} \times \textit{channel} \times \textit{time}$ [MVSAV+08, CLW+15, AABB+07, RTN09, MHH+06]. If measurements are recorded across different subjects or conditions, extra dimensions can be added to the simple model. For instance, in [CPA+13, CPZ+12], multi-subject EEG data is modeled as a fourth-order tensor of $\textit{frequency} \times \textit{channel} \times \textit{time} \times \textit{subject}$. Likewise, EEG data is modeled as a fifth-order tensor $\textit{frequency} \times \textit{channel} \times \textit{time} \times \textit{subject} \times \textit{condition}$ [MHH+06]. Note that tensor decomposition does not operate directly on EEG raw signals, rather, a preprocessing step (usually via wavelet transformation) is required to transform the raw EEG signals to EEG tensors [MHH+06].

Tensors are also applied to fMRI data analysis. fMRI images can be used to detect brain regions that have been damaged by various neurodegenerative diseases such as Alzheimer and Parkinson. A typical fMRI scan image may contain $64 \times 64 \times 14$ voxels (3D equivalent of pixels) sampled at different consecutive time instants, producing a single matrix. Multiple scans on a given subject generate a higher-order tensor of $\textit{voxel} \times \textit{time} \times \textit{runs}$ which is usually used in fMRI data analysis [AR04]. Scans can also be performed for multiple subjects resulting in $\textit{voxel} \times \textit{time} \times \textit{subjects}$ [AR04]. The tensor model can have extra dimensions such as trials (e.g. rest, finger tapping, etc.) resulting in a four-order tensor of $\textit{voxel} \times \textit{time} \times \textit{trials} \times \textit{runs}$ [BS05].
2.3.8 Remote sensing

Nowadays, with the aid of hyperspectral imaging technology we are able to capture spectral images with a different range of spectra. We can create multiple images from a scene or object via light from different parts of the spectrum. Furthermore, these hyperspectral images can be used for target and object detection and identifying materials from long distances and of course anomalies.

The simplest tensor model used for hyperspectral images is a three-order tensor of spatial rows $\times$ spatial column $\times$ wavelength that is used for target detection and classification [RB08, BFC10, RB09] or for space object material identification [ZWPP08]. The more advanced tensor models are those used by [ZZTH11] which add two extra dimensions to the hyperspectral tensor. The new model which is called multi-feature-tensor representation is a fifth-order tensor of spatial rows $\times$ spatial column $\times$ wavelength $\times$ scale $\times$ direction which scale and direction are the parameters of the Gabor function which are chosen as constant numbers. The Gabor function is a popular technique for texture representation and discrimination in image processing.

The other dimension that can be added to the simple model is time. Majority of remote sensing techniques are based on the assumption that the spectral signature of objects is persistent and uniform over time which might not be true. Therefore, a new model called multi-temporal hyperspectral tensor, denoted by spatial rows $\times$ spatial column $\times$ wavelength $\times$ time is proposed in [HFSES13] that is reached by combining multiple hyperspectral images obtained at different time points. This model is considered as a new generation of soft sensors for remote sensing.

2.3.9 Sensors

One of the potential applications of tensors is anomaly detection in sensor networks which use the same tensor model as environmental monitoring with this difference that sensors gather data with higher speeds and mostly are used in real time monitoring. The sensor networks are modeled as a three-order tensor of measurements $\times$ space $\times$ time in [SPP06, SGJ14, STP+08b]. In some other circumstances, sensor may gather some information from people. The scheme of tensor in this condition is persons $\times$ measurements $\times$ time. For instance, in [HTS+10] six measurements are gathered from 20 people during a period of 255 hours in an office environment. Then via tensor decomposition, some meaningful events are detected which have been linked to some
regular events such as lunch break or general meeting or a monthly seminar.

2.3.10 Engineering

Tensor decomposition has been used in civil engineering [PTKH12, PDBG12] for detection of abnormal changes in the structure vibration response. Different sensors are employed in different parts of the structures and their vibration responses are measured during a period of time. Therefore, the tensor model is represented as \( \text{space} \times \text{time} \times \text{frequency} \).

Application of tensors in metallurgy engineering can be seen in [CCYH09] where tensor decomposition is used for fault detection in the hot strip mill, specifically damage to the surface of coils. The data generated from ASIS (automatic surface inspection system) is modeled as a third-order tensor of \( \text{Coils} \times \text{PSD} \times \text{frequencies} \) where PSD (power spectrum densities) and frequencies are obtained via autoregressive processes of several signals modeled by Fast Fourier transform (FFT).

An example from the mechanical engineering domain can be observed in [MVR+08] where tensors are applied to detect damage in sensitive artefacts such as aircraft wing flap. The main problem in aircraft wing flap is that the impacts on its surface is barely visible. To solve this problem, the authors propose a new multiway model for detection of damage via monitoring multiple sensors. They suggest a tensor scheme of \( \text{experiment} \times \text{sensor} \times \text{time} \) for this problem.

The electrical engineering community has also used tensors for voltage sag detection in power distribution networks [KMCS08]. The tensor model of \( \text{experiments} \times \text{variables} \times \text{time} \) is proposed and is later unfolded time-wise to detect sag points.

The robotic engineers have also used tensors for predication of fall in walking robots [KW09]. Inspired by the tensor-based batch process monitoring, they model the non-linear trajectory of walking robots and suggest a third-order tensor of \( \text{trajectory slices} \times \text{scaled state variables} (e.g. position, angle) \times \text{time} \) for fault detection.

2.3.11 Transportation systems

Traffic data (\( \text{Origin} \times \text{Destination} \) matrix) is frequently used for traffic planning and management in intelligent transportation systems. Tensor decomposition has been used on the tensor \( \text{Origin} \times \text{Destination} \times \text{Time} \) for discovery of spatiotemporal
traffic structure [WGC+14] that has important applications to urban planning and traffic jam control. Sometimes the collected data might be abnormal due to failures in the collection process and recording systems. This problem is known as outlier recovery and addressed in [TFF+13b] with tensors. Tensors are also used for prediction of missing values in traffic data (known as tensor completion) [TFF+13a].

2.3.12 Medical applications

Tensors are exploited for analysis of electronic medical records. In [HQSH12] a change detection system is developed for pain management decision making. A collection of medical forms completed at various treatment and recovery stages is modeled as a sixth-order tensor of initial pain $\times$ initial infusion $\times$ sex $\times$ surgery site $\times$ pain $\times$ month and based on that some interesting change patterns are detected. Tensor decomposition is also applied to electronic health records (EHR) for prediction of heart failure [HGS+14]. A tensor model of Medication $\times$ Patient $\times$ Diagnosis is used for this purpose. Tensors are also used in bio-informatics for modelling micro-array gene expression tensors (gene $\times$ sample $\times$ time) that can be used for diagnosing diseases [LN10].

2.3.13 Other applications

Many other applications from tensor-based methods have appeared in recent years, in particular during the last five years that are inherently different from the traditional applications of tensors. In [BMG+13] spectral changes of substrates and products are monitored in real time via modeling temporal evolution of enzyme activity with three-order tensor of wavenumber $\times$ time $\times$ activity. Tensor analysis is applied for tracking the analysis of proteins. In [RAL08] authors use tensor analysis to model the deviations of contacts between residues and their environment with respect to each other (i.e., relative behavior) as well as with respect to time (i.e. temporal behavior). The tensor model used in this work follows the scheme of contract matrix $\times$ time where the contact matrix $A_{ij}(t)$ represents the normalized value of the number of heavy atoms in residue $i$ coming in contact with the heavy atoms in residue $j$ at time $t$.

A dynamic pattern of international trades and the asymmetric relations between countries is studied in [BHK07] which has some potential for anomaly detection.
Tensor decomposition has applications in seismology. A third-order tensor of \( space \times time \times frequency \) is built in [BTCC13] for prediction of ground motion after earthquake. Time-frequency components are obtained by the transformation of acceleration records of earthquake ground motions with continuous wavelet transform.

Tensor analysis is used for analysis of climate tensors (\( climate \ indicator \times grid \times time \)) [Lei10, LQD07, UHTJ11]. These models are quite potential for investigations on climate change.

Tensors are used for crime forecasting [MDMT11]. A four order tensor of \( longitude \times latitude \times time \times measurements \) is used for this purpose where measurements refer to criminal activities such as residential burglaries, construction permits, motor vehicle larceny, offender data, etc.

One of the recently emerged topics in anomaly detection is acoustic anomaly detection in which several acoustic sensors are monitored for event detection. Acoustic anomaly detection can be used, for instance, safety monitoring in nuclear power plants [MD14]. Unfortunately, although tensor decomposition has a high potential, is not yet used for this purpose, whereas we can find works that model voice data as a third-order tensor of \( rate \times scale \times frequency \) [MDFS14, MSS06] or \( rate \times time \times frequency \) [PKA10]. These tensor models have the potential for being used in the acoustic anomaly detection.

2.4 Tensorial Learning Strategies

Tensor methods are more known for unsupervised and semi-supervised learning. However, in recent years, many supervised tensor learning methods and tensor time series models are developed. Some of these recent techniques have not yet been used for anomaly detection purposes, but are quite potential. Table 2.2 presents the summary of these methods with corresponding references. In the following, these strategies are described in more detail.

2.4.1 Supervised models

Perhaps we can seek the first footprint of using tensors in supervised anomaly detection in Multiway PLS models [NM95a]. The second important role of tensors is in dimensionality reduction in classification problems. Nowadays, more supervised
Table 2.2: Learning Strategies for Tensor-based Anomaly Detection

<table>
<thead>
<tr>
<th>Model</th>
<th>Category</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Dimensionality reduction</td>
<td>Categorical target [PTKH12, PDBG12]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Numerical target [BTCC13]</td>
</tr>
<tr>
<td>Supervised</td>
<td>Classification based</td>
<td>Support Tensor Machines [ZZTH11]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Supervised Tensor Learning [TLH05]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tensor Least Square [CHH06]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Multilinear discriminant analysis [YXY07]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Factorization Machines [Ren10]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tensor subspace learning [LPV11]</td>
</tr>
<tr>
<td>Regression based</td>
<td></td>
<td>Multiway PLS (N-PLS) [NM95a]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tensor ridge regression [GKP12]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Support Tensor Regression [GKP12]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>H-MOTE [ZHL14]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tensor regression [ZLZ13]</td>
</tr>
<tr>
<td>Time series based</td>
<td></td>
<td>Multilinear Dynamical Systems [RLR13]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Greedy Low-rank Tensor Learning [BYL14]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tensor Hidden Markov Model [Yu12]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tensor time series models [TNHST11, TN11]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tensor Singular Spectrum Analysis [KS13]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TrIMine [MSF12]</td>
</tr>
<tr>
<td>Semi-Supervised</td>
<td></td>
<td>Monitoring of decomposition statistics (SPE, T2, etc.)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[LLK14, TZDC09, ZJ07, MLS+06, LYL03, CY03, LV03, CL02, MTC+11, Gao12, CCYH09, YLV04, YLVL04, SH03, HY09, KW09]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Thesis Chapter 3</td>
</tr>
<tr>
<td>Un-supervised</td>
<td></td>
<td>Analysis of score-plots</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1D [PPF14, KPD+94, GPC14]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2D [WGM01, LLK+14, MWP+14, CPA+13]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3D [MWP+14]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Latent factors time series [BAP+02, PFS12]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Multivariate-SPC on multiple latent factors</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Thesis Chapter 6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Streaming residuals</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dynamic tensor analysis [STF06, HQSH12, RAL08, ZSH+11]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Window-based tensor analysis [SPP06]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Spatio-temporal tensor streams [SGJ14]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Thesis Chapter 7</td>
</tr>
</tbody>
</table>
CHAPTER 2. LITERATURE REVIEW

tensor-based learning methods are developed. Some of these methods, despite their potential for anomaly detection have not yet been applied. The goal of this section is to provide a structured list of existing and potential models.

2.4.1.1 Tensor decomposition for dimensionality reduction

In this category of supervised models, tensor decomposition is used as a dimensionality reduction tool for feature extraction. Depending on the target value, methods can be grouped in two categories.

In the first group of methods [PTKH12, PDBG12, NSSC06], it is assumed that we have two sets, train and test, where the train set contains normal samples. Tensor decomposition is applied on the normal tensor as a dimensionality reduction tool. Then, one of the factor matrices (usually time) is fed to a regular classifier (e.g. k-nearest neighbors or SVM) for making a model from the normal samples.

The goal is to predict the labels of the observations in the test set. Therefore, the built model from train set is used to predict the label (normal or abnormal) of observation in the test factor matrix. For instance, in [PTKH12, PDBG12], a PARAFAC decomposition with \( k \) number of components is applied on the \( space \times time \times frequency \) tensor corresponding to the normal samples and then the derived time factor matrix is trained via k-NN (features are the latent variables). The built model is then used for classification of time points in the arriving data. In other related work, a combination of PARAFAC and self-organizing map (SOP) is used [HFSES13] for classifying signatures of multitemporal-hyperspectral images.

The second group of methods [BTCC13] follows the same procedure as the former one with this difference that instead of binary labels (abnormal/normal) a numeric target is given which is required to be predicted. Therefore, regression models are replaced with categorical classifiers. Targets can be single or multiple variables. For instance, in [BTCC13] authors train a GRNN (generalized regression neural networks) on the tensor subspace latent variables for prediction of multiple seismological variables. The authors in this work used this method for prediction purposes, but this kind of approaches can be easily extended for anomaly detection. A further step, however, is required. For instance, the difference of predicted and actual values can be used along with a threshold to detect anomalies.
2.4.1.2 Tensor classifiers

Tensor classifiers are those that adapt regular classifiers for tensorial data. In these methods, data is trained directly via tensor-based classifier and then the built model is used for prediction. A binary tensor classifier has a great ability for anomaly detection from multiway data. A good example from this category is a method presented by [ZZTH11] where SVM (support vector machines) is extended to STM (support tensor machines). The new tensorial classifier is trained directly with the tensorial data of specific objects and then the built model is used for target detection. In another work [TLH+05] a general framework called Supervised Tensor Learning (STL) is proposed. It adapts many conventional learning machines to take higher order tensors as inputs. This model is successfully tested for binary classification problem which is very useful for anomaly detection as well. In [CHH06] in addition to another version of STM a new method is also presented called Tensor Least Square (TLS) which is the extension of the least square classifier. A new type of STM is also presented in [KGP12] which is applied for gait and action recognition.

Multilinear discriminant analysis (MDA) [YXY+07] is also proposed for tensor-based image classification that is an extension of Linear discriminant analysis (LDA) for tensor data. Factorization machines [Ren10] is another method for tensor-based classification that extends SVM for tensors using PARAFAC which is motivated for SVM difficulty in collaborating filtering problems. Tensor classifiers are also known as supervised multilinear subspace learning in image processing community. The recent survey paper [LPV11] covers the majority of advances for tensor subspace learning.

2.4.1.3 Tensor regression

The first tensor regression models emerged in the 1980s by the Chemometrics community with the traditional name of N-PLS or multiway PLS [WGEÖ87]. In these techniques which are widely used for anomaly detection [NM95a, NM95b, CL02, CY03, MLS+06, KNM95, SMBS07, LWY06] a model is built based on the relationship of the input tensor (X) to some quality measurements (Y). That model is then used for prediction of quality measurements of new tensors. Deviations of predicted target variables from the normal reference are interpreted as abnormal behavior.

Apart from the traditional multiway regression models, some novel techniques have been recently developed in different research communities. One is [GKP12] that proposed two tensor regression models called Tensor ridge regression (TRR) and
support tensor regression (STR) that respectively extend vector regression models such as ridge regression (RR) and support vector regression via some properties of PARAFAC model. The authors applied these methods to facial data for human-age estimation and head/body-pose prediction. However, these methods can be quite interesting for many more applications in TAD.

Another tensor regression model is proposed in [ZLZ13] which is motivated by some problems in brain imaging where observed binary diagnosis status (Y) are required to be modeled based on the fMRI images as an input (X). The proposed tensor model is used to identify regions of interest in brains that are relevant to a clinical response with applications for detection of brain diseases, including Attention Deficit Hyperactivity Disorder and Alzheimer.

Moreover, [ZHL14] proposed a tensor-based regression algorithm called H-MOTE that is capable to incorporate background knowledge into the model. This model is used for prediction of wafer quality in semiconductor manufacturing.

2.4.1.4 Tensor forecasting

Tensor forecasting is an extension of vector time series models for multiway time series. The procedure for anomaly detection is the same as univariate ones. A model is built for tensor time series and then based on that model, future tensors are predicted. In the subsequent moment, if the tensor has a considerable difference with the predicted tensor, it is marked as an anomaly. Different methods are developed for tensor forecasting. In [RLR13] a model called Multilinear Dynamical Systems (MDS) is proposed which is a tensorial extension of linear dynamical system (LDS). Detection of the market collapse and climate change are introduced as the applications of this methodology. Another tensor forecasting method, named Greedy Low-rank Tensor Learning is proposed in [BYL14] that is applied for forecasting tensor time series such as climate tensors.

Some time series analysis tools are also extended for tensors. For instance, a tensor-based Hidden Markov Model (HMM) approach is proposed in [Yu12] and is used for fault detection and prediction. Some ideas in time series analysis, such as weighting and averaging are also extended for tensor analysis in [TNHST11, TN11]. Tensor version of singular spectrum analysis (STA) is also presented in [KS13] by replacing SVD with PARAFAC in regular STA and is applied for a non-stationary source separation of seizure signals.
An innovative approach called TriMine [MSF+12] is also proposed for tensor forecasting in the context of topic modelling. In the proposed methodology, a train tensor data is decomposed with regular tensor decomposition and then based on the obtained time factor matrix, the next time factor matrix is predicted with different scales. Later, the new predicted time factors are multiplied by other two dimensions to construct the tensor forecast in short-term and long-term way. This approach seems promising for multi-scale anomaly detection and prediction.

### 2.4.2 Semi-supervised models

Semi-supervised methods are twofold. The first group is originated in online fault detection in batch processes where a train tensor model corresponding to normal operation condition (NOC) is usually constructed. Then, arriving data is monitored to detect deviations from NOC model using statistics such as Squared Prediction Error (SPE) or Hotelling $T^2$ chart [NM94]. Examples of this category are [NM94, CL02, YLVL04, LV03, YLV04, TZDC09, LYL03].

Semi-supervised models impose less human intervention, consequently they are more desirable comparing supervised methods. In many applications such as process control or network security, usually labeling data for each time instant is infeasible. Therefore, this model presents a better flexibility and simplicity.

### 2.4.3 Unsupervised models

Tensors are better known for unsupervised learning in problems such as co-clustering and anomaly detection. In this section, the popular unsupervised methods are described.

#### 2.4.3.1 Score plot-based

The most traditional use of tensor decomposition in anomaly detection is with score plots obtained from the decomposition that are analyzed manually or automatically for anomaly detection or clustering. Score plots can be 1D (only one factor) [PPF14, KPD+94, GPC14], 2D [WGM01, LLK+14, MWP+14, CPA+13] and 3D [MWP+14]. If the latent factor is time, some factors might be presented as a multivariate time series [BAP+02, PFS12].
2.4.3.2 Streaming decomposition error-based

This group of methods is those streaming decomposition methods that operate on data incrementally without the requiring a train set. They monitor decomposition reconstruction error for each tensor in each time instant. Anomalous time instant is the one that its corresponding reconstruction error goes beyond a pre-defined threshold (e.g. twice standard deviation of errors so far). Examples are [STP+08b, STF06, LHWG11, HLZ+11, SGJ14, ZSH+11].

2.5 Tensor decomposition methods

Tensor decomposition is the central tool in either supervised, semi-supervised and unsupervised models. Many methods have been developed for tensor decomposition, but Tucker [Tuc66] and PARAFAC [Har70] (special case of Tucker) have received more attention from the community. In this section we list the six main categories of methods for tensor decomposition that have potential for anomaly detection, including PARAFAC-based, Tucker-based, DEDICOM-based, Bayesian, Locality Preserving Projection (LPP) based and ICA-based. Table 2.3 demonstrates the summary of existing techniques.

2.5.1 PARAFAC-based

2.5.1.1 PARAFAC

Canonical Polyadic Decomposition (CPD) or Parallel factor analysis (PARAFAC) [Har70] is one of the oldest methods used for multiway data analysis. PARAFAC as PCA requires only one parameter (i.e. number of factors). Probably due to this simplicity, it has been applied extensively in various anomaly detection tasks [MVSAV+08, CAP+11, SMS+06, BMG+13, PTKH12, GCP10].

2.5.1.2 Non-negative PARAFAC

One of the important issues in tensor decomposition is that elements in factor matrices can get negative values. These negative scores cannot be justified with the real background knowledge. Negative elements might not be a problem for automatic anomaly
Table 2.3: Methods for Tensor Decomposition and Applications to Anomaly Detection

<table>
<thead>
<tr>
<th>Family</th>
<th>Method</th>
<th>Anomaly detection example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tucker</td>
<td>Multiway PCA (Tucker1) [GWS96]</td>
<td>[Gao12, ZJ07, VRS⁺08, CCYH09, KMCS08, UHR12, MTC⁺11, CCYH09, KW09, KMCS08, ZJ07, LP06, LYL03, LBJ14, PLH⁺14]</td>
</tr>
<tr>
<td></td>
<td>GTucker2 [LBGY14a]</td>
<td>[LBGY14a]</td>
</tr>
<tr>
<td></td>
<td>Tucker3 [Fuc66]</td>
<td>[DZZ⁺10, CAP⁺11, DZZ⁺10, SMS⁺06, ACKY05, SS05, BAP⁺02, FTOG⁺12]</td>
</tr>
<tr>
<td></td>
<td>Non-negative Tucker [KC07]</td>
<td>[CPA⁺13, WGC⁺14]</td>
</tr>
<tr>
<td></td>
<td>HOSVD [DLDMV00a]</td>
<td>[ZZP12, CSS08, SSC⁺06]</td>
</tr>
<tr>
<td>PARAFAC</td>
<td>PARAFAC [Har70]</td>
<td>[RTN09, AR04, MVSAV⁺08, PPF14, APG⁺14, SZS⁺04, CAP⁺11, SMS⁺06, BMG⁺13, HFSES13, BTCC13, PTKH12, GCP10]</td>
</tr>
<tr>
<td></td>
<td>Non-negative PARAFAC [CDSP89]</td>
<td>[GPC14, CPZ⁺12, KPF12, BBB08, PFS12]</td>
</tr>
<tr>
<td></td>
<td>PARAFAC2 [Kie93]</td>
<td>[WGM01]</td>
</tr>
<tr>
<td></td>
<td>Dynamic PARAFAC [CY03]</td>
<td>[CY03]</td>
</tr>
<tr>
<td></td>
<td>CP-APR [CK12]</td>
<td>[MYT4a]</td>
</tr>
<tr>
<td>DEDICOM</td>
<td>[Har78]</td>
<td>[BHK06, BHK07]</td>
</tr>
<tr>
<td>Bayesian</td>
<td>EM-based (pTucker [CG09], ETF [HTS⁺12], InfTucker [XQY15] )</td>
<td>[HTS⁺12, HTS⁺10, XYQ15]</td>
</tr>
<tr>
<td></td>
<td>MAP-based (ARD [MH09], FBCP [ZC15] )</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Gibbs sampling (Multi-HD [PBW08], BTA [TSL⁺08], BPTF [XCH⁺10], TriMine [MSF⁺12], BMTF [KK14], MGP-CP [RWG⁺14], sp-PARAFAC [ZBHD14] )</td>
<td>[MSF⁺12]</td>
</tr>
<tr>
<td>LPP</td>
<td>TLPP [HCN05, DY06]</td>
<td>[LWL10, HY09]</td>
</tr>
<tr>
<td></td>
<td>TGLPP [LBGY14b]</td>
<td>[LBGY14b]</td>
</tr>
<tr>
<td>ICA</td>
<td>Tucker1-based (MICA [YLVL04], MKICA [ZQ07], FS-MKICA [TZDC09] )</td>
<td>[YLVL04, ZQ07, TZDC09]</td>
</tr>
<tr>
<td></td>
<td>Tucker3-based [UHTJ11]</td>
<td>[UHTJ11]</td>
</tr>
<tr>
<td></td>
<td>PARAFAC-based [BS05]</td>
<td></td>
</tr>
</tbody>
</table>
detection approaches, but might be a constraint in visual inspections. This problem is mostly motivated by the chemometrics and neuroscience community where the output of tensor decomposition requires to be interpreted by a specialist. PARAFAC model with non-negative constraint is called non-negative PARAFAC or non-negative tensor factorization (NTF) which was presented for the first time in [CDSP89]. Nowadays, NTF due to its meaningful and physical interpretation has become popular, especially in manual score-plot based anomaly detection [GPC14, CPZ+12, KPF12, BBB08, PFS12].

### 2.5.1.3 PARAFAC2

In some specific circumstances as occur in batch monitoring, we face with a tensor with uneven-length slices. For instance, in batch monitoring with tensor of batch × measurement × time, the matrix measurement × time can be in different length for each batch due to different elapsed time for the batch. PARAFAC2 [Kie93] which is an extension of PARAFAC provides a solution for such problems. It is used in [WGM01] for fault detection from batch tensors with unequal time axis and its superiority over regular PARAFAC and Tucker is shown.

### 2.5.1.4 Dynamic PARAFAC

A procedure called DPARAFAC (dynamic parallel factor analysis) is introduced in [CY03] for online fault detection in process monitoring. This methodology includes two phases, learning and detection. In the learning phase, we are given the data of normal operation condition (NOC). Each slice of the NOC tensor (matrix measurement × time) is segmented into different equal-length windows in the time axis. Then all the segments together form a new tensor (measurement × window × time). PARAFAC is then applied on this tensor for each batch and loadings are obtained. The average of factor matrices for each window is obtained for all batches. Later, some statistics such as $T^2$ and control limits are computed for each time point. In the detection phase, when new batch data arrives, it is arranged as the previous procedure, and then is projected onto the previous under-control subspace to assess its degree of abnormality.
2.5.1.5 Poisson tensor factorization

Poisson tensor decomposition (PTF) [CK12], also known as CANDECOMP/PARAFAC Alternating Poisson Regression (CP-APR) uses a new fitting algorithm based on Kullback-Leibler (KL) divergence instead of common ALS fitting algorithm in PARAFAC. The idea of such approaches is that count data can be better described by a Poisson distribution rather than Gaussian distribution. This model is exploited for anomaly detection from count data [MY14a].

2.5.2 Tucker-based

2.5.2.1 Tucker1

Tucker1 or Multiway PCA (MPCA) is the first tensor model used for TAD in many applications [Gao12, ZJ07, VRS+08, CCYH09, KMCS08, UHR12, MTC+11, CCYH09, KW09, KMCS08, ZJ07, LPV06, LYL03, LBJ14, PLH+14]. Tucker1 is used when variance is important only in one dimension. Therefore, the tensor is usually unfolded through one dimension and then regular PCA is applied to the unfolded data. For instance, in batch monitoring, Tucker1 model is used on batch-wise or time-wise unfolded matrices.

2.5.2.2 GTucker2

Tucker2 model is barely used for anomaly detection. Only very recently a generalized version of Tucker2 called GTucker2 was proposed [LBGY14a] for fault detection from tensors with unequal slice lengths. GTucker2 is equivalent to PARAFAC2, such that PARAFAC2 can be viewed as a constraint version of GTucker2. In [LBGY14a] the superiority of GTucker2 is shown over Tucker1, PARAFAC, Tucker3 and PARAFAC2.

2.5.2.3 Tucker3

The other model which promises more flexibility is known as Tucker3. This model as is presented in the previous section is normally used when there is multiway variations in all modes [DZZ+10, CAP+11, DZZ+10, SMS+06, ACKY05, SS05, BAP+02, FTOG+12]. For instance, for water quality tensors, we are interested to discover
abnormal locations, time instants and measurements that are more correlated to anomalies. Therefore, Tucker3 is preferred model [DZZ+10].

2.5.2.4 Non-negative Tucker

There are some extensions of NTF for Tucker decomposition, called Nonnegative Tucker decomposition [KC07]. The NTF is used for modeling EEG tensors [CPA+13, WGC+14] and is shown that is superior to NTF in some circumstances.

2.5.2.5 HOSVD

Higher-order singular value decomposition (HOSVD) is a generalization of SVD for higher-order tensors. HSOVD can be viewed as a special case of the Tucker3 model when ALS optimization is not performed, rather, tensor is unfolded across different modes and then regular SVD is applied on the unfolded matrices. HOSVD is applied mostly for processing of video tensors [ZZP12, CSS08, SSC+06].

2.5.3 ICA-based

Independent component analysis (ICA) is a popular method for decomposition of a multivariate signal into additive subcomponents. The basic assumption in ICA is that subcomponents are independent, non-Gaussian signals. Extension of ICA for tensors is available for Tucker1 (MPCA) [TZDC09, ZQ07, YLVL04], Tucker3 [UHTJ11], and PARAFAC [BS05]. All these methods except the latter one are applied for anomaly detection.

2.5.4 DEDICOM-based

DEDICOM (DEcomposition into DIrectional COMponents) [Har78, Kie93] is a generalization of PARAFAC2 for discovering asymmetric relationships between two modes that refer to the same type of object (e.g. transactional data). This model has been found promising in temporal analysis of social networks [BHK06, BHK07]. Therefore, it can be used for for event detection goals in similar scenarios.
2.5.5 Bayesian methods

Traditional tensor decompositions are unable to handle issues such as missing values, outliers, noises and different data types. Recently, probabilistic methods are taken into consideration due to their flexibility and less restrictive assumptions. They are successfully applied to anomaly detection problems [HTS+12, HTS+10, XYQ15, MSF+12] and is expected that the number of their applications be increased day by day.

Bayesian approaches, based on the means of used statistical inference can be divided into three categories. The first group is based on Expectation maximization (EM) algorithm, including pTucker [CG09], Exponential Family Tensor Factorization (ETF) [HTS+12] and Infinite Tucker (InfTucker) [XYQ15]. The second group exploits maximum a posterior (MAP) estimation, such as Automatic Relevance Determination (ARD) [MH09] and Fully Bayesian CP Factorization (FBCP) [ZZC15]. Finally, the third category uses gibbs sampling as an inference engine. Examples are Multi-HD [PBW08], Bayesian tensor analysis (BTA) [TSL+08], Bayesian Probabilistic Tensor Factorization (BPTF) [XCH+10], TriMine [MSF+12], Bayesian Multi-View Tensor Factorization (BMTF) [KK14], multiplicative gamma process based CP decomposition (MGP-CP) [RWG+14] and sp-PARAFAC [ZBHD14].

2.5.6 Locality preserving based methods

Tensor decomposition methods such as Tucker and PARAFAC do not consider the intrinsic local geometric structure of tensors. A recent group of techniques is developed for dealing with this problem on the basis of locality preserving projections (LPP). It has been shown [HY09, HY08] that LPP-based approaches have better performance than conventional PCA-based methods which preserves only the global Euclidean structure. LPP-based approaches are more attractive when two dimensions of tensors are in a pairwise relationship (e.g. image data).

The methods that address these techniques are known as different names such as Tensor subspace analysis [HCN05] or Tensor embedding [DY06]. The most popular method for this family is Tensor Locality Preserving Projection (TLPP) [HCN05, DY06] which is applied to detection problems [LWLT10, HY09]. A more sophisticated version of TLPP is very recently proposed, called Tensor Global-Local Preserving Projections (TGLPP) and is applied for fault detection problem in batch processes [LBGY14b] which is able to capture both global and local structures of tensors simul-
2.5.7 Tensor rank estimation

The quality of the tensor model has a direct relationship with true model selection. Although estimation of tensor rank is an NP hard problem [Häs90], in the majority of cases, an optimal low-rank approximation is desirable. In most of the works discussed in this chapter, it is assumed that the number of components is known in advance via knowledge of the underlying phenomena. However, this might not be the case in many applications. Some approaches are developed for estimation of optimal number of ranks for both tensor decomposition approaches. Some of these approaches are listed in the below subsections (See Table 2.4 for summary).

2.5.7.1 Cumulative sum of the percentage of eigenvalues or explained variance

This is the most basic method for choosing the number of components. It is mostly used for MPCA (Tucker1) models. The number of principal components is chosen based on cumulative percentage of eigenvalues or cumulative percentage of explained variance. If the cumulative percentage of first \( k \) components is over a threshold (e.g. 75%), \( k \) is selected as the adequate number of components. For instance, [MY14b] use eigenvalue criterion and [UHR12, WKM99, NM95b, MBY01] use cumulative variance for anomaly detection in process batch tensors.

Sometimes instead of a threshold cut point, broken stick rule [Jac93] is used. This approach assumes that percentage of explained variance (or eigenvalues) of a random data when is divided randomly amongst \( k \) components follows a broken-stick distribution \( G_k = \frac{1}{p} \sum_{i=k}^{p} \frac{1}{i} \). Therefore, the \( k \)-th principal component is valuable if has a greater value than \( G_k \) (i.e. a random PC). This rule is used for model order estimation of Tucker1 for anomaly detection [MTC+11, NM95b].

2.5.7.2 Cross-validation

A popular method for finding the adequate model order in component analysis is cross validation [Efr83]. This technique is applied for fault detection problem in [NM95b, YLVL04, CY03, Gao12] for estimation of number of components in MPCA model and
Table 2.4: Methods for Tensor Rank Estimation

<table>
<thead>
<tr>
<th>Method</th>
<th>Common use</th>
<th>Fast</th>
<th>Auto</th>
<th>Application to anomaly detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cumulative sum of percentage of eigenvalues [MY14b]</td>
<td>Tucker1</td>
<td>No</td>
<td>Yes</td>
<td>[MY14b]</td>
</tr>
<tr>
<td>Cumulative sum of explained variance [NM95b]</td>
<td>Tucker1</td>
<td>No</td>
<td>Yes</td>
<td>[UHR12, WKM99, NM95b, MBY01]</td>
</tr>
<tr>
<td>Broken stick rule [Jac93]</td>
<td>Tucker1</td>
<td>No</td>
<td>Yes</td>
<td>[MTC+11, NM95b]</td>
</tr>
<tr>
<td>Cross-validation [Efr83]</td>
<td>Tucker1/</td>
<td>No</td>
<td>Yes</td>
<td>[NM95b, YILV04, CY03, Gao12, LSK99, NM95b]</td>
</tr>
<tr>
<td></td>
<td>PARAFAC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CORCONDIA [BK03]</td>
<td>PARAFAC</td>
<td>No</td>
<td>Yes</td>
<td>[LLK+14, GPC14, PTKH12, CAP+11, AABB+07, SMS+06, SZS+04]</td>
</tr>
<tr>
<td>DIFFIT [TK00]</td>
<td>Tucker3</td>
<td>No</td>
<td>Yes</td>
<td>[CPA+13, CPZ+12]</td>
</tr>
<tr>
<td>FastDIFFIT [KK03]</td>
<td>Tucker3</td>
<td>Yes</td>
<td>Yes</td>
<td>Thesis Chapter 6</td>
</tr>
<tr>
<td>Multiway scree plot [AB00]</td>
<td>Tucker3</td>
<td>No</td>
<td>No</td>
<td>[EGS+14, FTOG+12, WGM01, DZZ+10, SMS+06, SS05]</td>
</tr>
<tr>
<td>Split-half analysis [HDS84]</td>
<td>PARAFAC</td>
<td>No</td>
<td>Yes</td>
<td>[CAP+11, SMS+06]</td>
</tr>
<tr>
<td>Maximum block improvement [CLZ14]</td>
<td>Tucker3</td>
<td>Yes+</td>
<td>Yes</td>
<td>Thesis Chapter 6</td>
</tr>
<tr>
<td>Convex hull [CK06]</td>
<td>Generic</td>
<td>No</td>
<td>Yes</td>
<td>Is not yet applied for anomaly detection but is used for tensor rank estimation [MH09].</td>
</tr>
<tr>
<td>Akaike’s information criterion (AIC) [Aka74]</td>
<td>Generic</td>
<td>No</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Bayesian information criterion (BIC) [S+78]</td>
<td>Generic</td>
<td>No</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Automatic relevance determination (ARD) [MH09]</td>
<td>Generic</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Genetic algorithm [KYA10]</td>
<td>Tucker3</td>
<td>No</td>
<td>Yes</td>
<td>Used for noise removal [KYA10]</td>
</tr>
</tbody>
</table>
its extension is presented in [LSK99] for Tucker3 and PARAFAC models. The basic idea of cross-validation is leave-out a single data element [LSK99], a slice [RB03] or random half of a slice [K+05] at a time, perform tensor decomposition and then compute the Predictive Residual Error Sum of Squares (PRESS) \( \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} (\tilde{X}_{ijk} - X_{ijk}) \) for the elements not included in the model building. Finally, the sum of PRESS values for each principal component \((p,q,r)\) is calculated for all eliminated parts to compute \( PRESS_{pqr} \). Those \((p,q,r)\) that gives the minimum PRESS are considered a good model dimension. The more sophisticated cross-validation approaches are developed based on w-statistics [LSK99] which uses F-test strategy to determine whether an additional component is worth to add or not.

2.5.7.3 CORCONDIA

Core consistency test (also known as CORCONDIA) [BK03] is a heuristic method for determination of the number of components in PARAFAC model. It is widely applied in anomaly detection from tensors [LLK+14, GPC14, PTKH12, CAP+11, AABB+07, SMS+06, SZS+04]. If \( P \) be the number of components in PARAFAC model, CORCONDIA checks the superdiagonality of Tucker3 model with a core size of \((P,P,P)\). If all elements in core tensor except those with same indices \((i = j = k)\) become zero, it concludes that the PARAFAC model fits perfectly. The procedure is as follows. First, core consistency criterion is defined as the similarity percentage of Tucker3 core size with superdiagonal array \( T \) of ones and then the PARAFAC model is fitted for a series of model from \( P=1 \) to \( F \) and core consistency is computed for all these models. The last model in these series that its corresponding Tucker3 core is similar to \( T \) is considered as the adequate number of components.

2.5.7.4 DIFFIT

DIFFIT (Difference in Fit) [TK00] is residual-based heuristic procedure for estimation of the number of components in Tucker model. It computes the Tucker decomposition for all sensible combinations of components \((i,j,k)\) and computes the model fit as \( \text{Fit}(m) = 1 - \frac{\|X - \bar{X}\|_F}{\|X\|_F} \) for each potential model where \( \| . \| \) is the Frobenius norm and \( m = i + j + k \). Then the DIF\((m)\) for \( m-th \) model is computed as \( \text{Fit}(m) - \text{Fit}(m-1) \) and accordingly DIFFIT is computed as \( \text{DIFFIT}(m) = \text{DIF}(m) / \text{DIF}(m+1) \). The model with largest DIFFIT value is chosen as the most adequate model. The DIFFIT model has been used for estimation of tensor model dimension in EEG tensors.
[CPA+13, CPZ+12]. DIFFIT requires computing the Tucker fit for all combinations of components which is very time-consuming. [KK03] proposed a faster version of DIFFIT (so called Fast-DIFFIT) that requires performing a single computation of Tucker decomposition. [KK03] provide some evidences that this approach can be sufficient as the exact solution.

2.5.7.5 Multiway scree plot

Multi-way score plot [Kro08] projects Tucker3 model onto the convex hull. The most adequate model is the one on the convex hull with less complexity and better fit. This method is used in [EGS+14, FTOG+12, WGM01, DZZ+10, SMS+06, SS05] for tensor-based monitoring and anomaly detection.

2.5.7.6 Split-half analysis

This technique first time was introduced by [HDS84] for PARAFAC. The procedure is that the tensor is splitted into two (or more) parts and tensor model with the same number of components is built for two parts. The assumption of this method is that if model is valid, both models on two sides should be stable. A criterion called split-half stability coefficients is defined and if its value is lower than a threshold (e.g. 0.1), the model is considered stable. However, the main requirement for use of this method is that tensor be splittable [Kro08] which is restrictive for non-stochastic systems. Limited works such as [CAP+11, SMS+06] use this technique for determination of the number of components in tensors with application to anomaly detection. Extension of this method later was proposed by [KM01].

2.5.7.7 Other methods

Some other approaches are proposed for tensor rank estimation which may not be used for anomaly detection applications but are potential. Some of these methods include convex hull [CK06], Akaike’s information criterion (AIC) [Aka74], Bayesian information criterion (BIC) [S+78] and Automatic relevance determination (ARD) [MH09]. These four approaches are implemented for multiway models and compared in [MH09] and superiority of ARD is concluded against the other three ones. Bayesian-based tensor decompositions also may be a good solution for tensor rank estimation since they automatically find the tensor rank in their inference procedure [RWG+14,
In [KYA10] a different approach named GAHNTD is proposed based on Genetic algorithm for finding the optimal Tucker lower rank, but no comparison is performed against the known approaches. [BPPC13] propose a greedy approach that build the tensor model iteratively, and use BIC criterion to identify the correct number of components. A more efficient method based on maximum block improvement (MBI) is proposed in [CLZ14] that use non-convex block optimization for finding the Tucker3 model rank. It is shown that this method outperforms DIFFIT and ARD (when the sum of dimension is predefined) both in terms of accuracy and runtime.

2.6 Issues

This section outlines some of the most important issues in TAD and the corresponding solutions extracted from the related works. Table 2.5 presents a summary of the issues and the corresponding solutions.

2.6.1 Data pre-processing

Data pre-processing is an important step in TAD. Tensor models are sensitive to the scale of data elements. If data of multiple scale is going to be used in tensor, it first requires to be scaled, such that all columns have a same scale [LSK99]. This is usually done via z-score scaling [KPD+94, LLK+14]. In some cases, the input data is a continuous signal and therefore is required to be transferred to discrete values with tools such as wavelet transform [MHH+06, AABB+07, BTCC13, MVR+08].

2.6.2 Processing types : Offline/Online/Streaming

Based on processing tensors in offline or real time, TAD methods are classified into three categories of offline, online and streaming. Offline processing model [BAP+02, SMSS06, SMBS07] is usually used in score plot based unsupervised detection (section 2.4.3). Online processing is usually referred to semi-supervised methods (section 2.4.2). A tensor model is built from a normal operation condition of the system and then that tensor model is used for matching with new observed data to identify abnormal items. Since the expensive task is performed in offline, the detection part includes processing only a small piece of data. In the majority of cases, the normal model is not updated during the detection process. However, in some works, it is suggested to constantly...
<table>
<thead>
<tr>
<th>Problem</th>
<th>Solutions</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data pre-processing</strong></td>
<td>Scaling</td>
<td>[KPD+94, LLK+14][Kro08, Chapter 6]</td>
</tr>
<tr>
<td></td>
<td>Continues to discrete transformation</td>
<td>[AABB+07], BTCC+13, MVR+08</td>
</tr>
<tr>
<td><strong>Processing</strong></td>
<td>Offline</td>
<td>[BAP+02], SSS06, SMB07</td>
</tr>
<tr>
<td></td>
<td>Online without updating</td>
<td>[HY09, KW09, YL04, YL04, Yu12, UHR12, MTG+11, Gao12, TZDC09, ZI07, MLS+06, LYL03, CY03, CY03, LV03, CL02, WGM01]</td>
</tr>
<tr>
<td></td>
<td>Online with updating</td>
<td>[BAP+02], Thesis Chapter 3</td>
</tr>
<tr>
<td><strong>Tensor dimensionality</strong></td>
<td>Prior Knowledge</td>
<td>The Majority</td>
</tr>
<tr>
<td></td>
<td>Multiway ANOVA</td>
<td>[PTOG+12]</td>
</tr>
<tr>
<td></td>
<td>Compare different ranks (e.g., 2D vs. 3D)</td>
<td>[AI04]</td>
</tr>
<tr>
<td><strong>Tensor rank</strong></td>
<td>See Table 2.4</td>
<td></td>
</tr>
<tr>
<td><strong>Nonlinearity</strong></td>
<td>Eliminate nonlinearity in a preprocessing step</td>
<td>[KW09]</td>
</tr>
<tr>
<td></td>
<td>Kernel tensor decomposition</td>
<td>[MY14, TZDC09, RT12]</td>
</tr>
<tr>
<td><strong>Seasonality</strong></td>
<td>Seasonal segmentation</td>
<td>[LLK+14]</td>
</tr>
<tr>
<td></td>
<td>Separated tensors for each environment setting</td>
<td>Thesis Chapter 3</td>
</tr>
<tr>
<td><strong>Unequal-length slices</strong></td>
<td>Treating the absent part of shorter-length series as missing values</td>
<td>[Krut8]</td>
</tr>
<tr>
<td></td>
<td>Dynamic time warping</td>
<td>[Gao12]</td>
</tr>
<tr>
<td></td>
<td>Phase division</td>
<td>[LGW04]</td>
</tr>
<tr>
<td><strong>Scalability</strong></td>
<td>Sparse-optimized methods</td>
<td>[K08, AL12, BMVL12]</td>
</tr>
<tr>
<td></td>
<td>GPU-based</td>
<td>[CIM+15, ZTCT13]</td>
</tr>
<tr>
<td></td>
<td>Distributed and parallel approaches</td>
<td>[KPHF12, K08, PF12, SPF14, IPKF15, dAK14]</td>
</tr>
<tr>
<td><strong>Adaptivity</strong></td>
<td>Incremental tensor analysis</td>
<td>[SPF06, ZH+11, STF06, ZSH+11]</td>
</tr>
<tr>
<td></td>
<td>Online probabilistic</td>
<td>[HTS+12]</td>
</tr>
<tr>
<td></td>
<td>Multi-aspect-streaming</td>
<td>Thesis Chapter 7</td>
</tr>
<tr>
<td><strong>Temporal scaling</strong></td>
<td>Prior knowledge (single scale)</td>
<td>The majority</td>
</tr>
<tr>
<td></td>
<td>Multi scale in data model</td>
<td>[SMB07, SMS+06]</td>
</tr>
<tr>
<td></td>
<td>Multi-scale approaches</td>
<td>[MSF+12]</td>
</tr>
<tr>
<td><strong>Hybrid models</strong></td>
<td>Decomposition + Classification</td>
<td>[PTKH12, PDBG12, NSC06]</td>
</tr>
<tr>
<td></td>
<td>Decomposition + Clustering</td>
<td>[KRS+08, PTOG+12]</td>
</tr>
<tr>
<td></td>
<td>Decomposition + Case-based reasoning</td>
<td>[MV1+08]</td>
</tr>
<tr>
<td></td>
<td>Combine tensor models on different data represen-</td>
<td>Thesis Chapter 6</td>
</tr>
<tr>
<td></td>
<td>tations</td>
<td></td>
</tr>
<tr>
<td><strong>Data fusion</strong></td>
<td>Multiblock/multiway models</td>
<td>[Krut6]</td>
</tr>
<tr>
<td></td>
<td>Coupled matrix and tensor factorization (CMTF)</td>
<td>[ALRB13, ARS+13, LSC+09, YLS+11]</td>
</tr>
<tr>
<td><strong>Noise removal</strong></td>
<td>Preliminary phase removal</td>
<td>[Kro08, Chapter 12]</td>
</tr>
<tr>
<td></td>
<td>Two-step decomposition</td>
<td>[MY14]</td>
</tr>
</tbody>
</table>
2.6.3 Tensor dimensionality

The dimensionality of tensors is usually chosen based on a prior knowledge. For instance, space and time are inherent modes of the tensor when system behavior is time-changing and data items are subject to change according to their spatial position. Dynamic networks are also in principle three-way tensors, such that the first and second dimensions denote the interactions between nodes and the third mode models the time-changing factor. Due to higher cost of tensors comparing matrix methods, use of tensors is justified if there is at least three-way interactions in data. Multiway ANOVA test is one of the approaches used for discovering multi-way interactions in the data. For instance, Three-way ANOVA test [KM01] is used in [FTOG+12] for determination of tensor dimensionality in anomaly detection application. However, the inefficient solution also exists that compares the fit of models in different orders (e.g. 2D vs. 3D) [AR04].

2.6.4 Nonlinearity

Traditional tensor decompositions are unable to model complex nonlinear interactions between entities in each mode. Nonlinearity problem in TAD is recently reported [MY14b, KW09, Yu12, TZDC09, RY12]. The solution to this problem is yet limited. Some propose to eliminate nonlinearity in a preprocessing step by segmentation of tensors to different linear parts [KW09], while some other such as [MY14b, TZDC09, RY12] propose the kernelized form of existing tensor decomposition methods. The probabilistic non-parametric methods such as [Yu12, XYQ15, CG09] also are suggested for dealing with this issue. A kernel non-negative Tucker Decomposition is also proposed in [KYZA11].
2.6.5 Seasonality

Most of TAD approaches are based on the assumption that the behavior of a system is persistent and uniform over the time. However, in the systems that deal with human activities such as Internet, social networks, public health, etc. this assumption is valid only in a particular temporal period. For instance, we know that in the winter, rate of flu increases. If the model epidemic data with tensors and do not incorporate the seasonality, we probably signal many false alarms for winter season.

When we apply tensors for such data, two objectives are usually pursued. One is to discover periodic patterns of an unknown system. For instance, discover what is the seasonal pattern of water quality or land surface changes [DZZ+10, CAP+11, HFSES13]. The other and more practical target is to monitor the seasonal tensor for more accurate detection of anomalies and deviations. For the latter case, knowledge of periodic patterns is necessary for building a better tensor model. Although this issue is very important, only very few works addressed that. [LLK+14] model the indoor air pollutants into four subsets of fall, winter, spring and summer and make a tensor model for each season. They compared this strategy with global model when all seasons are modeled together and showed that the new strategy is more accurate.

2.6.6 Unequal-length slices

The tensor with uneven slices is a known problem in process monitoring. This problem arises when process duration in each batch is different and thus measurement $\times$ time matrix for each batch is in unequal-length due to different length of time axis. Four scenarios lead to such a problem [Kou03]: 1) Majority of measurement time series have equal length, but a minority of them despite of overlap in common time part, have shorter length; 2) All measurement time series have same length but some of them have small shift due to delay or acceleration in data collection; 3) The measurement time series have the same length, but are appeared in different shape; and 4) The time series of measurements are in different length and shape.

Two different groups of approaches exist for the above problems. The first group of methods suggests to perform a pre-processing step on the data known as trajectory synchronization/alignment before doing the decomposition. In this category, the first problem is solved by treating the absent part of shorter-length series as missing values. For the second problem, synchronization is carried out with a simple shift only on the
minority series. For the third and fourth scenario which are more general cases, the measurements are expressed against the other variable (known as indicator variable) other than time such that when shape of time series overlap for all measurements. Dynamic time warping [Gao12] and phase division [LGYW04] techniques are also used for this purpose. However, these techniques are criticized in the sense that they distort the anomalous patterns and reduce the anomaly detection accuracy [Kou03].

The second group, and more sophisticated methods are those that model uneven-length tensors in its natural way. One of the tensor models that can operate directly on uneven-length tensors is PARAFAC2. This model was utilized for fault detection [WGM01] and its superiority was shown over synchronization techniques. PARAFAC2 is able to directly model the original uneven-length tensor without performing further data unfolding or trajectory synchronization. However, it is inherited the restrictive constraints of its equivalent model, PARAFAC. [LBGY14a] proposed GTucker2, a generalized version of PARAFAC2, that does not have these limitations and at the same time can be used to model both even-length and uneven-length tensors. The authors showed that GTucker2 has a better anomaly detection performance than PARAFAC2 for both even-length and uneven-length batch tensors.

### 2.6.7 Scalability

Scalability of tensor decomposition techniques is a hot and young research area in data mining, machine learning and signal processing community. The important problem is that the decomposition of big tensors is not computationally affordable by traditional techniques. Therefore, it is necessary to extend tensor methods for processing of large data sets. Three major groups of solutions are presented for this purpose, including sparse-optimized methods, GPU-based solutions and parallel and distributed techniques.

The need for sparse-optimized methods arises from this fact that the majority of tensors in data mining applications is in principle sparse. For instance, density of Email, Web and network tensors barely exceed 0.1%. Methods like [KS08, All12, BMVL12] attempt to optimize the traditional tensor decomposition for large sparse tensors, in particular with operating on nonzero elements.

GPU-based techniques try to use new computing paradigms such as GPU instead of CPU for speeding up the decomposition process. The GPU is proven that has a substantially outpaced CPU in dealing with computationally demanding and com-
plex problems. Two examples from this category are G-PARAFAC [CLW+15] and GPUTENSOR [ZLTC15].

Distributed and parallel approaches have received more attention by researchers due to the current progresses in parallel, distributed and cloud computing. The general objective of these methods to reduce the intermediate data explosion problem [KPHF12, KS08] and improvement of the runtime of tensor decomposition by splitting tensors into different sub-tensors and process each smaller sub-tensors in a distributed, parallel or cloud environment (e.g. MapReduce). Examples of this category include GigaTensor [KPHF12], ParCube [PFS12], PARACOMP [SPF14] and HaTen2 [IPKF15].

2.6.8 Adaptivity

Standard tensor decompositions are developed for operation in offline settings. It means that when new data is received they are unable to update the model and therefore they have to rebuild the model from the scratch. Normally, due to the large volume of data in many applications, rebuilding the model is not feasible. There exist some streaming approximation solutions for this problem for either classical tensor decomposition, subspace analysis or probabilistic tensor decomposition.

The most popular framework for incremental tensor analysis is ITA [STP+08b] consisting of three algorithms called Dynamic tensor analysis (DTA) and streaming tensor analysis (STA) [STF06] and window-based tensor analysis (WTA) [SPP06]. DTA decomposes the tensor incrementally by maintaining only the covariance matrix for each arriving tensor. Then, via diagonalization it outputs the principal eigenvectors of the updated covariance matrix as projection matrices. STA attempts to approximate DTA. Instead of maintaining a covariance matrix for all arriving tensors, it directly updates the principal eigenvectors using SPIRIT algorithm [PY06] which does not require diagonalization. The other algorithm, WTA instead of processing individual tensors uses a sliding window strategy for handling time dependency between consecutive tensors. It decomposes the sliding window with a regular Tucker or PARAFAC and then as well as DTA and STA keeps some statistics from the window in the processing of next windows.

ITA restricts the tensor growth only in time, which is a huge constraint in scalability and adaptability of other modes. In fact, ITA is only useful for large, but slender tensors.
Incremental extensions of locality projection based methods (section 2.5.6) are also developed that are typically developed for object tracking in video tensors (i.e. \( \text{spatialrow} \times \text{spatialcolumn} \times \text{frame} \)). The motivation of these methods is to model the appearance changes of objects in video data. A more recent approach from this category is DTAMU [ZSH+11] that extends DTA for subspace learning. The objective of work is to take into account the geometric structure of the image object, which is ignored in DTA. The similar ideas are used in [LHZ+07, HLZ+11].

Incremental version of probabilistic methods (section 2.5.5) is also presented in some works such as [HTS+12].

### 2.6.9 Multi-scale anomalies

In the discrete space, determination of the right scale (or sampling rate) for temporal dimension requires a prior knowledge about the scale of fluctuations. The sampling rate, depending on the application, can be per second [MVSAV+08], minute, [KPD+94, SPP06, GCP10, KPF12, BMG+13, GPC14], hour [MWP+14, LLK+14, SGJ14, FTOG+12, HTS+12, LLK+14, STP+08b], k-hours [ZJ07, MLS+06], day [KPF12, STP+08b], k-days [APG+14], month [HQSH12, Lei10, BPB08, SMBS07, BAP+02] and year [BHK07, STP+08b]. If there is no precise knowledge about scaling, multiple tensor model with different temporal scale may be built from data (e.g. see [BBB08, TFF+13b]). As is demonstrated in [BBB08] the smaller scale (day) may provide a similar interpretation to the bigger scale (month), but with finer resolution. However, this might not be the case for all applications. If multiple scales have different influences on the data, a combination of more than one temporal scale may be used. For instance, in [SMBS07, SMS+06] a multi-scale scheme of \( \text{Sites} \times \text{variables} \times \text{year} \times \text{month} \) is proposed for modeling of soil and water quality data. In this case, year and month, even though both refer to temporal dimensions, affect data in a different manner. Therefore, some meaningful patterns might be hidden if we lean to only-month or only-year scales. Recently, a multi-scale probabilistic tensor analysis framework called TriMine is developed in [MSF+12] that accounts for several time granularities.
2.6.10 Hybrid models

Hybrid models are those that use tensor decomposition as the secondary tool. Usually the output of tensor decomposition is entered to another analysis tool such as classification (see section 2.4.1.1), case-based reasoning [MVR+08] and clustering [VRS+08, FTOG+12].

2.6.11 Data fusion

Coupled matrix and tensor factorization (CMTF) are emerging group of techniques that attempt to formulate a data fusion model based on joint factorization of matrices and higher-order tensor. In many applications, jointly analysis of an ensemble of data sets from multiple sources (also known as multi-block, multi-view, multi-set, multi-source data analysis) results in enhancement of knowledge discovery.

The first use of data fusion based tensor and matrix decomposition in anomaly detection appeared in the work of [Kou03] who proposed the use of multiblock/multiway PLS model for batch processes. The authors proposed that if we incorporate prior knowledge such as initial conditions for batches, such as raw material properties, initial ingredient charges and operation conditions in the original tensor model, the accuracy of anomaly detection will be improved.

Nowadays, the application of CMTF has been extended to wider areas such as location-based recommender systems [ZXY12, EAC15], neuroscience [BCA12, Cic13, SHA+14, KK14], and sensory data analysis [ARS+13]. CMTF also has been used in related applications to anomaly detection such as social networks [LSC+09, YLS+11] and metabolomics [ALRB13, ARS+13]. For instance, in the metabolomics case, many heterogeneous data sets are generated via different analytical techniques for measuring biological fluids (e.g. blood). These complementary data sets if analyzed jointly may improve the understanding of the underlying biological processes corresponding to specific diseases.

A complete list of bibliography related to data fusion based on coupled matrix/tensor factorizations is gathered in [Aca15].
2.6.12 Noise removal

Noise is a disturbing phenomena in data that is not of interest for the analyst and it only negatively affects data analysis task [CBK09]. Sometimes it can be difficult to distinguish anomalies from noises in tensor models due to their similar nature. Noise removal is usually undertaken as a preliminary phase in tensor-based modeling (See [Kro08, Chapter 12]). However, in some works such as [BAP+02, VRS+08, MY14a], a two-step decomposition is proposed for handling this issue. For instance, [MY14a] propose a two-step tensor decomposition framework such that the first decomposition accounts for noise removal and the second decomposition that operates on the first step output takes into account the meaningful anomalies.

2.7 Evaluation

Evaluation of TAD methods usually is similar to classical anomaly detection techniques [CBK09]. The typical metric used include precision/recall [SGJ14, MY14a, GPC14], accuracy [HFSES13, ZZP12, Yu12, Yu12, UHR12, PKA10, KMC08] and area under ROC curve (AUC) [MY14a, PTKH12, HTS+10]. For semi-supervised and unsupervised techniques, true and false positives (or false alarms) are also assessed [MY14b, Yu12, PTKH12, KLMW09]. For regression based tensor models and tensor forecasting methods, predication error metrics such as root mean square error (RMSE) or mean absolute error (MAE) is normally used [HFSES13, MSF+12, ZSH+11, HTS+10, SMBS07, SZS+04, SH03, RLR13]. Detection delay has been also exploited in some works [HY09]. Visual inspection of score plots with visual inspection is another evaluation method used in [LLK+14, MWP+14, PPF14, BMG+13, ZZP12, CPZ+12, PFS12, KPF12].

2.8 Tensor software

Various toolboxes are developed for tensor analysis in the recent years. The most popular ones are Tensor toolbox (http://www.sandia.gov/~tgkolda/TensorToolbox) and N-way toolbox (http://models.life.ku.dk/nwaytoolbox) which are widely used in the many disciplines for tensor analysis. More recently, two toolboxes, TensorBox (http://www.bsp.brain.riken.jp/~phan) and Tensorlab (http://www.esat.kuleuven.be/sista/tensorlab) also are developed. TensorBox is more focused on advanced fitting algorithms for Tucker and PARAFAC, while Tensorlab offers
CHAPTER 2. LITERATURE REVIEW

A wider range of algorithms for more complex tasks in tensor decomposition such as coupled tensor factorization, sparse and incomplete tensor decomposition, and new fitting algorithms such as quasi-Newton and nonlinear-least squares optimization, etc. NFEA toolbox (http://www.bsp.brain.riken.jp/~phan/nfea/nfea.html) is a tensor toolbox specifically developed for processing EEG tensors. CMTF toolbox is also developed for coupled matrix and tensor factorization (http://www.models.life.ku.dk/~acare/CMTF_Toolbox). Hierarchical Tucker toolbox (http://anchp.epfl.ch/htucker) is developed for hierarchical Tucker decomposition. Apart from above MATLAB toolboxes, some R packages also exist for tensor decomposition, including ThreeWay, rTensor and PTAk.

2.9 Conclusion

We provided the conceptual classification of many existing techniques, applications and issues for tensor-based anomaly detection. In the majority of works that we surveyed, the superiority of tensor-based methods has been shown over matrix methods which exhibit the importance of tensors as new category in spectral-based anomaly detection. We classified the tensor-based learning into three categories of supervised, semi-supervised and unsupervised. Supervised methods are less investigated for anomaly detection, but some of them like time tensor series and tensor forecasting have a good potential for TAD. We reviewed the majority of tensor decomposition methods applied to anomaly detection, including populars and the potential ones. We also surveyed the TAD literature in terms of the important problem of tensor rank estimation and provided a list of existing techniques. Finally, we presented some of the important issues related to TAD and the provided the existing solutions for them.
Chapter 3

Online/Semi-supervised Method

Syndromic surveillance systems continuously monitor multiple pre-diagnostic daily streams of indicators from different regions with the aim of early detection of disease outbreaks. The main objective of these systems is to detect outbreaks hours or days before the clinical and laboratory confirmation. The type of data that is being generated via these systems is usually multivariate and seasonal with spatial and temporal dimensions. The algorithm What’s Strange About Recent Events (WSARE) is the state-of-the-art method for such problems. It exhaustively searches for contrast sets in the multivariate data and signals an alarm when it finds statistically significant rules. This bottom-up approach presents a much lower detection delay comparing the existing top-down approaches. However, WSARE is very sensitive to the small-scale changes and subsequently comes with a relatively high rate of false alarms. We propose a new tensor-based approach called EigenEvent that is neither fully top-down nor bottom-up. In this method, we instead of top-down or bottom-up search, track changes in data correlation structure via eigenspace techniques. This new methodology enables us to detect both overall changes (via principal eigenvalue) and dimension-level changes (via principal eigenvectors). Experimental results on hundred sets of benchmark data reveals that EigenEvent presents a better overall performance comparing WSARE, in particular in terms of the false alarm rate.

3.1 Introduction

The goal of syndromic surveillance systems is to enable earlier detection of epidemics and a more timely public health response, hours or days before clinical and laboratory
CHAPTER 3. ONLINE/SEMI-SUPERVISED METHOD

confirmation comes out [Hen04]. Two kinds of events are usually required to be detected: man-made events such as bio-terrorist activities like anthrax attacks [GI003] and natural events such as epidemic diseases like H1N1, avian influenza, SARS, and West Nile Virus, etc. All kinds of events regardless of their type make some changes in the environment. If we somehow manage to identify such changes in the early stages we can save many lives and prevent the potential damages. The early event detection systems were developed for such purposes. In these systems, multiple streams of pre-diagnostic health records [SB10, Hen04] such as daily counts of doctor/hospital/emergency room visits, over-the-counter medication sales, work/school absences, animal illness or deaths, Internet-based health inquiries are being monitored simultaneously to trace the event footprints.

![Figure 3.1: A sample complex system in syndromic surveillance that generates 128 time series for 16 features and 8 spatial regions.](image)

Figure 3.1 demonstrates an example of a complex data stream in syndromic surveillance systems. As it can be seen, this system measures 16 features aggregated daily within 8 different regions. Hence, the system generates 128 time series. Our goal is to monitor this complex system and signal an alarm when something strange occurs. One straightforward approach for monitoring such system is to monitor each individual time series and then apply an anomaly detection technique (e.g. Control chart) on each. This approach, however, imposes a much higher false alarm rate. Because pre-diagnostic streams of indicators are weak and noisy signals [CDL+04b] and applying detectors on each individual signal results in multiple hypothesis testing
problem [WMCW05]. For instance, suppose that we reject null hypothesis when the
$p$-value $< 0.05$, for a single hypothesis test, the probability of making a false discovery
is equal to 0.05. Now assume that we do the test for each of 128 time series. Probability
of false alarm could be as bad as: $1 - (1 - 0.05)^{128} = 1.00 >> 0.05$.

Existing univariate methods (See section 2.2.3) only monitor a single variable, hence
are not proper techniques for handling the complex data in syndromic surveillance.
Because, if we monitor each individual feature independently without taking into ac-
count the correlation between them, we then likely confuse the measurements error and
noises with the events. Also, multivariate methods, despite of their wide application
in many areas, are not well-suited to syndromic surveillance and outbreak detection
problems where geographic dimension is widely involved.

The methods that take into account geographic dimension are twofold: spatial and
spatiotemporal. Spatial methods such as spatial scan statistics [Kul97] do not capture
the temporal fluctuations of the data and only operate on spatial data. Spatiotemporal
methods instead take into account both spatial and temporal dimensions. Space-time
scan statistics (STScan) belongs to this group that can operate both on univariate
count data [Kul97, KN95, Kul99, KAF+98] and multivariate data [KMD+07]. Univa-
riate STScan is not adequate for syndromic surveillance for the same reason mentioned
for univariate temporal methods. Multivariate STScan also has some drawbacks that
make it inappropriate for the introduced problem. On one hand it assumes that the
environment is static and does not consider seasonal effects and on the other hand it
is developed for retrospective and offline analysis. Therefore, this group of techniques
is not appropriate for real-time monitoring purposes.

There is another group of techniques such as PANDA [CDL+04b] that uses a causal
Bayesian network to model spatiotemporal patterns of outbreaks. These methods
not only explicitly compute the probability of events, but are also able to operate in
real time settings through incremental updating of the Bayesian network. However,
the main criticism against these techniques is that tuning the primary parameters
requires a deep prior knowledge that is not available most of the time. Therefore,
these methods are considered domain specific and their applications remain limited.

Among many existing techniques and algorithms, the most suited approach to the
introduced problem is an algorithm called "What’s Strange About Recent Events"
(WARE) [WMCW03, WMCW05] that is able to handle multivariate data along
spatial and temporal dimensions. WARE searches for surprising rules in data streams
given some baseline reference. The baseline creation strategy varies in different ver-
CHAPTER 3. ONLINE/SEMI-SUPERVISED METHOD

sions of the algorithm. WSARe 2.0 uses raw historical data from selected days, WSARe 2.5 uses all historical data that match the environmental attributes and WSARe 3.0 models the baseline distribution using a Bayesian network. Opposed to PANDA where Bayesian network is created manually, WSARe 3.0 learns the Bayesian network from historical set. Therefore, it is not as such domain-dependent as PANDA. WSARe has been successfully applied and merited in many real world problems such as in bioterrorism surveillance for 2002 Winter Olympics [GGT+03] and Israel influenza type B outbreak and Walton outbreak [KCPL+05]. However, the main criticism about WSARe is its high rate of false alarms [BBC+05]. WSARe opposed to other techniques, processes the data from bottom to up. Therefore, instead of overall changes in the whole data, it tracks the changes in subgroup of data. Therefore, it is sensitive to small changes and consequently presents lower detection delay, however, it comes with higher false alarm rate.

The methodological differences between our proposed method and WSARe are as follows. 1) WSARe is a bottom-up rule-based approach while our method is a middle approach between bottom-up and top-down that tracks both high level and dimension-based changes in the data subspace. 2) Our approach takes into account both multi-linear and multi-way correlations in data while WSARe is not able to capture such complexity; and 3) Our method is suitable only for alarming purposes and cannot explain the subgroup of the data that cause the alarm, while WSARe can be used for both purposes. 4) The statistical significance of the alarms in WSARe is computed via Monte Carlo simulation while in our approach is computed by statistical process control techniques.

In overall, the main objective in syndromic surveillance systems is to detect events in a timely manner before they turn into an epidemic. This early detection has important functions in both mortality saving and prevention of economic losses. An estimation by DARPA shows that a two-day improvement in detection time could reduce fatalities by a factor of six [Nei06]. Another study states that improvement of even an hour in detection can reduce the economic impact of by a hundred million of dollars [WeTE+01]. To reach this objective, any capable signal is required to be considered. However, this is somehow problematic, since involving more signals results in more false alarms. In the recent years the emphasis of the developed algorithms in syndromic surveillance has been focused more on the early detection and rate of false alarms is rarely taken into account. This is while the recent studies show that the false alarm rate can have an inverse effect as bad as delay in detection. A recent study concerning the warning system for tornado events [SS09] reveals that tornadoes
occurring in the regions with a high false alarms ratio kill and injure more people. A statistically significant effect of false alarms is identified in this study: A one-standard-deviation increase in the false alarm ratio increases expected fatalities by between 12% and 29% and increases expected injuries by between 14% and 32%.

Besides, opposed to anomaly or outlier detection problems, which assume that the process occurs in an isolated and static environment, in syndromic surveillance systems we deal with dynamic and time-changing environment. In such environments, attributes such as day of the week, holiday, weather, etc. affect the whole or part of the system behavior. Figure 3.2 illustrates an individual time series corresponding to the feature $V1$ and $Zone1$ sampled from Figure 3.1. As it can be seen, in point $A$ due to cold weather and high rate of influenza rate, we have a higher count comparing point $B$. Such effects impose another kind of complexity to the event detection problem in syndromic surveillance which is required to be taken into account along with other issues.

![Figure 3.2: The environmental setting affects the data items](image)

In this chapter, we propose a novel event detection methodology based on tensor and matrix decomposition that considers both data complexity and time-changing environmental issues in syndromic surveillance. The focus of this work is to reduce the false alarm rate of early event detection systems. Our contributions are as follows.

- To the best of our knowledge this is the first time that the tensor decomposition techniques have been applied to the syndromic surveillance problem.
- We use the changes in data dimensions and data correlation structure as an effective criteria for event detection.
• We introduce a novel and effective approach for baseline data creation that can infer baseline for unseen environmental settings.

The rest of the chapter is organized as follows. In section 6.2 we introduce the proposed solution and our proposed algorithm. The section 4.3.1.2 includes experimental evaluation, including the introduction of the data set, performance evaluation and sensitivity analysis. The last section concludes the chapter presenting the final remarks.

3.2 Proposed method

![Diagram of proposed method]

Figure 3.3: Snapshot of the proposed solution at a hypothetical timestamp. We detect events through tracking changes in the subspaces of baseline and recent data.

3.2.1 The intuitions

The fundamental idea that is used to develop the method relies on tracking changes in the subspace. This is impossible unless we can match the recent data with a baseline
reference. However, in streaming settings, data itself is time-changing due to the effect of the dynamic environment on the data items. Therefore, using a static baseline seems to be inappropriate for dynamic environments. We propose a dynamic baseline creation strategy which takes into account both seasonality and non-stationarity. The main novelty of our method is that we not only track changes in the feature subspace, but also in the subspace of other dimensions.

Figure 3.3 demonstrates an illustrative example of our proposed method. Each day we receive a chunk from a complex data stream. In the data stream model this can be translated to the sliding window with fixed size of one day across the data stream. The window here is more complicated than a one-dimensional window in temporal data processing. Each window is a two-dimensional matrix of $Space \times Features$ (top-right matrix). Each cell in the matrix corresponds to the count of a feature in specific regions. With respect to the sliding window environmental setting, we generate a dynamic baseline tensor with order of $Space \times Features \times Time$ (top-left tensor in the figure) which is fed from the historical data. This baseline tensor is built in each step or cycle of the algorithm run. The baseline tensor is composed of some previously arrived sliding windows that are combined in a particular order. We decompose the recent matrix and the baseline tensor to a lower-rank subspace and then match their pairwise principal singular vectors and values. We signal an alarm if we observe any unexpected difference in the match.

Figure 3.4 illustrates the eigenspace of both baseline and recent matrix. The solid vector in this figure corresponds to the baseline tensor. The direction of this vector corresponds to the principal singular vector corresponding to a dimension and the length of the vector corresponds to the principal singular value. When we receive a matrix we decompose it to the singular vectors and values and then match it with the reference one (solid vector). We signal an alarm if the singular vector has a considerable difference in direction (singular vector) or length (singular value). For instance, dashed lines in the figure correspond to those matrices that have close singular vectors to the baseline singular vector and have the close singular value (vector length). Such matrices are considered normal by EigenEvent. Dash-dot lines in the figure on the contrary are related to abnormal matrices that have an unexpected singular vector (unexpected vector direction) or unexpected singular value (unexpected vector length) with respect to the baseline.
Figure 3.4: A simplified example showing how events can be detected by tracking changes in the eigenspace. If the distance between the sliding window’s singular vector and baseline’s singular vector is higher than expected, then the window is marked as abnormal. Similarly, if the ratio of window’s singular value to the baseline singular value is higher than expected, the window is marked as abnormal.

3.2.2 Proposed Algorithm: EigenEvent

In this section we describe our proposed algorithm for event detection from syndromic surveillance tensors. As it is presented in Algorithm 3.1, the inputs are as follows: sliding window $D$ with length of one day; $t$ which is the sequence number; $e$ is a number corresponding to the environmental setting of the day. For instance, the environmental setting 1214 is related to: day=weekend(1), weather=cold(2), flu=high(1), season=winter(4). The algorithm as a result outputs a p-value indicating the statistical significance of the recent data. A very low p-value can be interpreted as an event signal.

3.2.2.1 Data Processing and Decomposition

The first phase is to transform the sliding window to the matrix format of $Space \times Feature$ (line 1). To assess the abnormality of sliding window we need a baseline reference to match with. Two strategies can be utilized, one is to compare the window with the previous data and the other one is to compare the window with previous data that has the same environmental setting. We use a combined strategy that takes both into account seasonality (section 3.2.2.4) and produces the dynamic baseline set according the context corresponding to the window (line 2). As a result, the baseline set is presented as a tensor of $Space \times Feature \times Time$. We then apply SVD [KL80] on window matrix and higher order SVD (HOSVD) [DLDMV00a] on the baseline tensor.
and for each dimension we take the principal singular vector and principal singular value.

Note that EigenEvent does not concern the feature selection (selection of pre-diagnostic signals that are required to be monitored). Feature selection, however, may be performed via standard feature selection techniques or via domain experts or a combined technique. Nevertheless, feature selection is one of the most important steps in a data mining process that is required to be taken into account. Selection of inappropriate signals may result in higher false alarm or more detection delay. The well-known over-fitting problem may happen here as well. Leinweber in an article entitle stupid data miner tricks: over-fitting the S&P 500 [Lei07] outlines some of such problems. He found a strong correlation between butter production in Bangladesh and S&P 500 (stock market index) over a ten year period. This implies that the selection of appropriate signals still is human-dependent.

### 3.2.2.2 Subspace Matching

The next phase is the matching phase. If we denote the principal singular value of baseline with $\lambda_b$, the principal singular value of window with $\lambda_s$, the principal singular vector of baseline with $x_b$ and the principal singular vector of window with $x_w$, we can define the ratio of singular values and Euclidean distance of singular vectors between baseline and the window respectively as:

$$
\begin{align*}
    d_{1,t} &= \frac{\lambda_s}{\lambda_b} \\
    \|d_{2,t}\| &= d(x_w, x_b).
\end{align*}
$$

We keep the historical distances in two vectors of $vd_1$ and $vd_2$ for singular values and singular vectors respectively, such that at time $t$ we have $vd_1 = (d_{1,1}, d_{1,2}, ..., d_{1,t-1})$ and $vd_2 = (d_{2,1}, d_{2,2}, ..., d_{2,t-1})$. Having $d_{1,t}$, $d_{2,t}$, $vd_1$ and $vd_2$ we can compute the z-scores corresponding $d_{1,t}$ and $d_{2,t}$ as follows.

$$
\begin{align*}
    z_1 &= \frac{d_{1,t} - \mu_{vd_1}}{\sigma_{vd_1}} \\
    z_2 &= \frac{d_{2,t} - \mu_{vd_2}}{\sigma_{vd_2}}
\end{align*}
$$
CHAPTER 3. ONLINE/SEMI-SUPERVISED METHOD

Where \( \mu_{vd_1} \) and \( \mu_{vd_2} \) denote the mean and \( \sigma_{vd_1} \) and \( \sigma_{vd_2} \) denote standard deviation of vector \( vd_1 \) and \( vd_2 \) respectively.

Although z-scores alone can be used along with a threshold for alarming purpose, since most related event detection algorithms in the literature output a p-value, we may want to transform z-scores to the corresponding p-value to ease the comparison task. We can use the following equation to derive the p-value from the z-score:

\[
P(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-\frac{t^2}{2}} dt
\]

(3.5)

3.2.2.3 Indicator Selection

As we already explained, HOSVD and SVD decompose the complex data into smaller subspaces (eigenspace). We have three elements in the eigenspace that can be matched: principal singular vector of spatial and feature dimensions and the principal singular value. We may observe three kinds of deviations in the match. The first kind includes an overall change in the system which is more related to the late days of outbreak period when we have both infection and outbreak. This kind of event must be reflected in a significant change in the ratio of eigenvalues \( (d_{1,t}) \). The second kind of change occurs when an event agent (e.g. Virus) begins to spread over the geographical space. This type of event is also reflected on the changes in the spatial eigenvector pairwise distance \( (d_{2,t}) \). The third kind is the change in the feature values. This event type can be reflected in the singular vectors corresponding to feature dimension. However, as we show later due to the noisy properties of the feature dimension, this kind of indicator is not that helpful.

We propose a new strategy that is able to detect both overall and dimension-based changes in the system. We monitor the system using a combination of indicators, including singular values and different singular vectors and the compute the p-value corresponding to each combination for each window. Then we take the minimum p-value as the algorithm output (line 9). Suppose that we have three p-values of 0.01, 0.12 and 0.43 corresponding to the pairwise match between the principal singular vector related to spatial and feature dimensions, and the principal singular value respectively. The EigenEvent algorithm reports the minimum p-value (0.01) as the output. This p-value indicates three facts about the system: 1) No overall change has occurred in the system, because the p-value’s corresponding singular value is considerably high; 2) No significant change is occurring in the feature values; 3) A
significant change is occurring in the spatial dimension. We may infer that data items
despite of showing normal behavior in the features are showing different behavior in
geographical space and hence we probably are in the outbreak phase. The minimum p-
value selection strategy lets us to detect all above kinds of changes and subsequently
make the algorithm sensitive to changes in both overall system behavior and the
dimension level.

3.2.2.4 Dynamic Baseline Tensor

There should be a criterion to estimate the abnormality of the recent data. As is
mentioned before, two types of common criteria include comparing with the previous
data and comparing with only the previous data that matches the current environment
settings. Both of these criteria are vulnerable. The first criteria is vulnerable when
data contains seasonal effects and second one is vulnerable when there is not enough
historical data matching the recent environmental setting. To solve this problem
a typical inference is usually performed, for instance, a causal Bayesian network is
constructed in WSARE 3.0 [WMCW03, WMCW05] so that when there is not enough
historical data, the baseline is inferred from the constructed Bayesian network. This
approach, however, only makes inference about the days that their corresponding
environmental settings cannot be found in the baseline set. In the rest of the time
it compares the recent window with the previous data that matches the current
environment settings. This approach can be vulnerable as well, since the correlation
of the current window with the recent data is ignored. We introduce another way of
baseline set selection which is a combination of both ideas. We assume that the recent
data is not only related to the previous data and data with the same environmental
settings, but also to data with the most repeated environmental settings. In fact, our
baseline tensor is a combination of previous data, data with the same environmental
setting and data from most frequent environmental settings. The main advantage of
this approach is that it is not vulnerable when faced with an unseen environmental
setting.

The function BaselineTensorUpdate in Algorithm 3.1 receives six inputs, including $B$
(current baseline tensor); $H$ (whole data, historical tensor); $t$ (instant number); $e$ (the
recent environmental setting); $ev$ (vector of all environmental settings seen yet); and
$C$ (recent matrix) and outputs the updated baseline tensor $B$. We first check whether
the tensor $B$ is empty. In the case that $B$ remains empty, $C$ is immediately added to $B$.
Then we search in the historical tensor $H$ for data that match the recent environmental
Algorithm 3.1 EigenEvent

// D: Recent data (Right top table in figure 3.3)
// C: Recent data in the format of matrix Space × Features
// t: instant (e.g. t=3 means after 3 days of monitoring started)
// e: Recent data Env. Setting (e.g. 1214={day=weekend, weather=cold, flu=high, season=winter})
// ev: Environmental setting vector (e.g. [1214,1321,3214,1456])
// ℋ: Historical Tensor
// vd₁: vector of principal Eigenvalue distances (d₁₁, d₁₂, ..., d₁ₜ₋₁)
// vd₂: vector of principal spatial Eigenvector distances (d₂₁, d₂₂, ..., d₂ₜ₋₁)
// B: Current generated Baseline tensor (Tensor Space × Time × Features in Figure 3.3)
// P-value: Statistical Significance of the recent data (e.g. Signal an alarm when p-value < 0.05)

Input: D, t, e, ev, B, ℋ
Output: P-value

1. Matrix C ← D
2. B ← BaselineTensorUpdate(B, ℋ, t, e, ev, C)
3. HOSVD(B): xₖ ← principal spatial Eigenvector, λₖ ← principal Eigenvalue
4. SVD(C): xₖ ← principal spatial Eigenvector, λₖ ← principal Eigenvalue
5. d₁ = λₖ
6. ||d₂|| = d(xₖ, xₖ).
7. p1 = p-value of d₁ given vd₁
8. p2 = p-value of d₂ given vd₂
9. P-value ← Min[p1, p2]
10. if e exists in ev then
11.   vd₁ ← d₁
12.   vd₂ ← d₂
13. end if
14. ℋ ← C
15. ev ← e
16. function BaselineTensorUpdate(B, ℋ, t, e, ev, C)
17.   if B is empty then B ← C
18.    else
19.      k=0
20.      for i=1 to t-1 do
21.        if ev(i) == e then
22.          k = k + 1
23.          B(k) ← ℋ(i)
24.        end if
25.      end for
26.    end if
27.    Return B
28. end function
setting. Next, we rewrite the first $k$ matrices of tensor $B$ with the matched items.

An illustrative example of the procedure is demonstrated in Figure 3.5. The figure is a snapshot of the system at four hypothetical days between days 50 to 53. From the figure we can also observe four distinct environmental settings, which are shown with different colors and their corresponding name is demonstrated in the guide table. Each cube in the figure represents a baseline tensor and each plate inside the cubes is a $Space \times Features$ matrix from the historical set. At day 50 the baseline tensor is composed of 20 matrices such that 9 of matrices are from setting $c_4$, 4 matrices from setting $c_2$ and 7 matrices from setting $c_1$. We also assume that the context $c_1$ is the dominant environmental setting with 20 times occurrence. The dominant context is the most frequent setting in all the history. For this reason, all baseline tensors, in Figure 3.5, include 20 matrices, given that the length of the baseline tensor is equal to the number of occurrences of the dominant context.

Now let us explain how a dynamic baseline set is generated. At day 50, we receive a matrix with setting $c_3$. We search in historical tensor $H$ for a match with $c_3$ setting, but we do not find it, so the function $BaselineTensorUpdate$ returns input $B$ unchanged. On day 51, we again receive a matrix with the setting $c_3$. We subsequently search for a match in $H$. This time we find one match, because one day before (day 50) the setting has been $c_3$. Therefore, we rewrite the first $k$ elements of $B$ tensor with $k$ found matrices. In this case since we find only one match, $k$ is equal to 1. At day 52 we receive a matrix corresponding with environmental setting $c_2$. We search in $H$ for a match and suppose that we find 13 matrices. Hence, $k$ will be equal to 13, so we rewrite the first 13 elements of the baseline tensor with the matched 13 matrices. As it can be observed at day 52, setting $c_2$ has been the dominant setting versus $c_3$ and $c_4$ settings, however, still $c_1$ dominates $c_2$ ($c_1$ setting has more repeats comparing $c_2$), therefore, the baseline tensor is composed of matrices with most dominant settings.
with preference to the recent data. Finally, on day 53, we receive a matrix with setting c3. We search in $\mathcal{H}$ for a match and we find 20 matrices (k=20), thus we rewrite first 20 elements of the baseline tensor with matrices corresponding c1 settings. At this moment, the whole baseline tensor is only filled with matrices with setting c1. This procedure repeats itself continuously. However, the size of baseline tensor always stays fixed to the repeat count of the most repeated environmental setting.

3.2.2.5 Updating Step

In this step we update the vector of distances (line 11-12). We add the distances to the vectors if their corresponding contexts have already been seen. If we have a matrix with an unseen environmental setting, we do not add the computed distance to the vector of distances. Because an inference for this setting is approximate and adding the distance obtained from this approximation is not adequate for keeping. We finally update historical tensor and vector of environmental settings.

3.3 Evaluation

3.3.1 Data set

Validation of event detection algorithms is a difficult task basically due to the type of required data [WMCW05, BBC+05, SF06]. To evaluate the algorithms, the event occurrence period is required to be clearly labeled in the data. This requires a knowledge expert to look into the data and specify the event period manually, making this task infeasible. Benchmark data sets that are already used for change detection and anomaly detection are not appropriate for our research purpose, because, on one hand, most of the time they do not have seasonality property and on the other hand do not contain multi-way property.

We use a benchmark data set used in [WMCW03] including 100 data sets of a simulated disease outbreak. These data sets are generated using a Bayesian network simulator namely CityBN which generates temporal fluctuations based on a variety of factors such as weather and food conditions [WMCW05]. The structure and parameter of this Bayesian network are manually adjusted. As is mentioned by the authors, this simulator produces extremely noisy data sets that are a challenge for any detection algorithm. This data set is publicly available online in [WKWCG13].
Table 3.1: Characteristic of CityBN benchmark data

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Cardinality</th>
<th>Sample Record #1</th>
<th>Sample Record #2</th>
</tr>
</thead>
<tbody>
<tr>
<td>XY</td>
<td>Spatial</td>
<td>9</td>
<td>SW</td>
<td>NE</td>
</tr>
<tr>
<td>Daynum</td>
<td>Temporal</td>
<td>730</td>
<td>73779</td>
<td>74508</td>
</tr>
<tr>
<td>Age</td>
<td>Feature</td>
<td>3</td>
<td>senior</td>
<td>child</td>
</tr>
<tr>
<td>Gender</td>
<td>Feature</td>
<td>2</td>
<td>female</td>
<td>male</td>
</tr>
<tr>
<td>Action</td>
<td>Feature</td>
<td>3</td>
<td>purchase</td>
<td>revisit</td>
</tr>
<tr>
<td>Reported symptom</td>
<td>Feature</td>
<td>4</td>
<td>nausea</td>
<td>respiratory</td>
</tr>
<tr>
<td>Drug</td>
<td>Feature</td>
<td>4</td>
<td>nyquil</td>
<td>vomit-b-gone</td>
</tr>
<tr>
<td>Flu Level in season</td>
<td>Environmental</td>
<td>4</td>
<td>high</td>
<td>decline</td>
</tr>
<tr>
<td>Day of week</td>
<td>Environmental</td>
<td>3</td>
<td>weekday</td>
<td>sat</td>
</tr>
<tr>
<td>Weather</td>
<td>Environmental</td>
<td>2</td>
<td>cold</td>
<td>hot</td>
</tr>
<tr>
<td>Season</td>
<td>Environmental</td>
<td>4</td>
<td>winter</td>
<td>summer</td>
</tr>
</tbody>
</table>

Table 3.1 shows the characteristic of the original data sets. As it can be seen, this data is multi-way. It contains two dimensions of space and time and multiple variables. It also contains seasonal effects, because features are under the influence of some environmental settings. Cardinality of each attribute is also specified in the table. As it can be seen, we have 9 distinct spatial regions and 730 temporal instants (days). We also have 16 \((3+2+3+4+4)\) distinct time series and \(4 \times 3 \times 2 \times 4\) possible environmental settings.

### 3.3.2 Performance

Receiver operating characteristic (ROC) curve [HM82] measures the trade-off between sensitivity and specificity. ROC curve is a widely used method for the evaluation of anomaly detection and classification methods. However, ROC curve, even though it summarizes the overall ability of the algorithm, does not evaluate the timeliness of detection which is critical in syndromic surveillance. An algorithm with the lowest false positive and the highest true positive rate that detect outbreaks with heavy delay is inappropriate for syndromic surveillance applications. In fact a system with this characteristic is more helpful for retrospective applications than the prospective applications like what is required in syndromic surveillance. One of the proper metrics for evaluation of online event detection systems is Activity Monitoring Operating Characteristic (AMOC) curve [FP99] that evaluates the trade-off between specificity (false alarms) and timeliness (detection time). AMOC curve is widely used for evaluation of methods in syndromic surveillance [CDL+04b, WMCW05, SP04, JC10]. In this work we use AMOC curve for evaluation of our algorithm.
CHAPTER 3. ONLINE/SEMI-SUPERVISED METHOD

Figure 3.6: Evaluation Strategy: Alarms in days with black color represents false alarms and in white days represents true alarms. Detection delay is also specified for each day inside the plates

We use the same evaluation strategy as [WMCW05]. Assuming that the agent release occurs at timestamp $t$. A true alarm corresponds to a case where the alarm is raised in a period between $t+1$ and $t+14$. The alarms before or after this period are considered false positives. The detection delay is also defined as the temporal difference between the first alarm in the above period and the release time. In reality, the data of each day is processed on the following day. Therefore, it is not possible to detect an event on the day the release. Thus, the optimum detection is on the following day of the release (detection delay=1). This one day delay is also considered in CityBN simulation. Figure 3.6 demonstrates that how we define false alarms and detection delay. If we signal an alarm in a period of 14 days after release it is marked as true alarm and if we signal an alarm before or after this period, it is marked as a false alarm. Detection delay is also specified in the figure as numbers in the plates. If we signal an alarm during the day following the release, we get only one-day delay which is the optimum condition. For any alarm after this period we define detection delay equal to 14 (as [WMCW05]).

The outputs of both WSARE and EigenEvent are $p$-values indicating the statistical significance of recent data. Depending on the desired confidence level, we may signal an alarm. For instance, given a threshold like 0.05 we signal an alarm if the $p$-value corresponding to the recent data goes lower than 0.05. To assess the algorithms performances we use variable $p$-value threshold from 0.020 to 0.250 with the step of 0.001 (totally 231 $p$-values). Each data set has temporal size of 730 days. We use the first 365 days for training the primary baseline and the next 365 days for the evaluation of the algorithms. Baseline set is also incrementally updated whenever a new window arrives after day 365. Note that agent release in all 100 data sets occurs in the second year and it is guaranteed that the first year does not contain any release. A sliding window moves across the data from day 366 to day 730 and matches each window with the baseline. If the match outputs a $p$-value below the threshold, then
an alarm is raised. After reaching to day 730, we compute the number of false alarms and detection delays. We finally average the detection delay and false alarms for all 100 data sets and plot the AMOC Curve. In the AMOC curve, the x-axis indicates the number of false alarms per month and the y-axis measures the detection time in days. The optimal detection is one day detection delay with zero false alarm. The closer to the point (0,1) better the detection algorithm is.

The results are shown in Figure 3.7. Although the curve corresponding EigenEvent seems different comparing to WSARE, if we rotate the AMOC curve 90 degrees anti-clockwise we observe the same pattern similar to WSARE 3.0. The difference is that EigenEvent performs better in terms of false alarm rate and performs worse in terms of detection delay. The intersection between the curves makes the overall comparison difficult. For instance, in a desired false positive rate from 2.8 to 3.3, EigenEvent is the best method both in terms of false alarm rate and detection delay. Nevertheless, to specify which of the algorithms are better in overall we need to compute the area under the AMOC curve [QT08], average delay and average false positive rate (see Table 3.2). Obtained area under AMOC curve implies that EigenEvent outperforms all versions of WSARE. Its average false positive is considerably lower than all versions of WSARE. However, in terms of detection delay as was expected presents one more day delay. To have a separate look on both numbers of false alarms and detection delay, we also compute the average false alarms and detection delay for 231 p-values (from 0.020 to 0.250 with the step of 0.001). The results are presented in Table 3.4 and Table 3.5 respectively. As it can be seen from the first table, EigenEvent in terms of false alarms, beats other methods in the majority of data sets. Regarding the detection
Table 3.2: False positive rate (per month), Detection delay (in days), Area under AMOC Curve and Runtime (in seconds) Averaged for 100 data sets of CityBN

<table>
<thead>
<tr>
<th>Method</th>
<th>False positive</th>
<th>Detection delay</th>
<th>AUAMOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSARE 2.0</td>
<td>4.052439</td>
<td>2.163983</td>
<td>12.859000</td>
</tr>
<tr>
<td>WSARE 2.5</td>
<td>2.739062</td>
<td>2.192338</td>
<td>9.885192</td>
</tr>
<tr>
<td>WSARE 3.0</td>
<td>2.877031</td>
<td><strong>1.929134</strong></td>
<td>8.648379</td>
</tr>
<tr>
<td>EigenEvent</td>
<td><strong>1.866439</strong></td>
<td>2.839827</td>
<td><strong>8.027842</strong></td>
</tr>
</tbody>
</table>

Table 3.3: Runtime (in seconds) Averaged for 100 data sets

<table>
<thead>
<tr>
<th>Method</th>
<th>Runtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSARE 2.0</td>
<td>59.2</td>
</tr>
<tr>
<td>WSARE 2.5</td>
<td>105.3</td>
</tr>
<tr>
<td>WSARE 3.0</td>
<td>838.4</td>
</tr>
<tr>
<td>EigenEvent</td>
<td><strong>16.8</strong></td>
</tr>
</tbody>
</table>

delay even though is not the best, detects events on the tomorrow of release in half of the data sets.

The main reason for the differences in the performance is related to the methodological differences between EigenEvent and WSARE. EigenEvent opposed to WSARE is not a bottom up approach and subsequently is less sensitive to the small-scale changes and consequently, presents less false positive rate. Naturally, EigenEvent due to its less sensitivity to the small-scale changes reacts slower to events. However, EigenEvent has this ability to track changes in the dimensions, for this reason it does not suffer from the high false alarm rate problem of bottom-up approaches and heavy delay problem of the top-bottom approaches.

### 3.3.3 Runtime

Since in syndromic surveillance systems, data is often required to be processed in daily scale, computational efficiency receives less attention. In the unlikely case where data size becomes very huge and processing of data requires a run-time of more than 24 hours (the process scale) then we have to come up with computational efficiency issues. Although, computational efficiency is not the claim in this research work, runtime in Table 3.3 indicates the superiority of EigenEvent over all versions of WSARE. EigenEvent requires only 16.8s to deliver the result. This is three times faster than
CHAPTER 3. ONLINE/SEMI-SUPERVISED METHOD

Figure 3.8: Effect of indicators on the performance

WSARE 2.0, 6 times faster than WSARE 2.5 and 50 times faster than WSARE 3.0. The majority of this difference is related to two factors; WSARE exhaustively searches the whole space while EigenEvent only tracks the changes in the correlation structure. The second factor is related to the method the approaches compute the p-value of alarms. WSARE exploits Monte Carlo simulations for computing the p-value while EigenEvent computes the p-value using statistical process control techniques which is lighter.

3.3.4 Leading Indicators

As we mentioned before, EigenEvent algorithm tracks the deviation of sliding window’s eigenspace from the baseline tensor’s eigenspace for event detection. Now the question is what elements of the eigenspace should we take for the match. Should we opt for singular vectors corresponding to the spatial dimension or to the feature dimension? Should singular value be used along eigenvector or is eigenvector alone enough? We examine five circumstances: the first condition is the default setting in the Algorithm (The optimum selection in line 9 of Algorithm 3.1), and the rest are different combination of singular vectors and singular values. Figure 3.8 illustrates the AMOC curve for these different combinations. As it can be seen, by using only singular vector corresponding to spatial dimension (without considering singular value) we experience the same result but with more half-day average delay. In fact, involving of Eigenvalue in the event detection process provides earlier detection. We also study a condition
where the whole eigenspace is used. In this case we take into account the principal singular vector of spatial and features dimensions along the principal singular value. This leads to an extra half-day delay on the detection, but with 1.5 days more false alarms. Excluding spatial singular vector from the eigenspace matching also leads to lower performance both in terms of delay and false alarms. This result reveals how the spatial dimension is important. In fact, temporal methods that do not take into account the spatial dimension lose a high portion of information. The reason is that feature signals are very noisy and detection of pattern of such noisy data comes with high false discovery. Instead, the spatial dimension is more stable and tracking changes in this dimension can be a better indicator for tracking particular events such as disease outbreaks (our case study), because, one of the key signatures of disease outbreak is the fluctuations in the geographic space. The reason is that outbreak changes the constant patterns in the spatial dimension and subsequently this appears in the principal singular vector corresponding to spatial dimension.

3.3.5 Baseline Selection

We compare three scenarios for baseline creation: 1) from historical set without respect to the environmental setting; 2) from historical set with respect to the environmental setting; and 3) Dynamic baseline tensor (our strategy). In the first scenario we compare the recent data with historical data without considering the environmental setting. For instance, we compare the recent data with data of the previous week or

![AMOC Curve](image)

Figure 3.9: Dynamic baseline vs. Environmental matching baseline
the last eight weeks. In the second scenario we take reference data from the matched environmental setting of the day. For instance, if the environmental setting of recent day is 4112 we search in historical set for those Space × Features matrices whose corresponding environmental setting is 4112. In the third scenario (our method), we create the baseline tensor according to the function BaselineTensorUpdate in Algorithm 3.1.

Figure 3.9 and 3.10 demonstrate the obtained performance through these different strategies. Figure 3.10 illustrates the comparison of the first scenario versus the third scenario and Figure 3.9 compares the performance of the second scenario versus the third scenario. The results reveal that our dynamic tensor creation strategy outperforms the first and second scenarios. The reason of this good performance is related to our approach making a better inference for unseen environmental setting.

3.4 Conclusion and future works

We propose a novel approach based on eigenspace techniques for event detection from complex data streams in syndromic surveillance. The purpose of this work was to reduce the false alarm rate of the state-of-the-art early detection methods. The experimental evaluation results on benchmark data sets show that the proposed approach provides a better overall performance versus the state-of-the-art algorithm for syndromic surveillance. Our approach while maintaining the detection delay in a
reasonable level improves the false alarm rate to a considerable extent. While top-down approaches look for changes in higher level feature space and bottom-up approaches track changes in the low-level feature space, we introduce a novel methodology based on eigenspace and tensor decomposition techniques that track deviations both in high level and the dimension level. The overall fluctuations in the system appear in the principal singular value and fluctuations in the dimensions appear in the singular vectors. Such dimension-based strategy is very helpful in some applications such as disease outbreak where the spatial dimension becomes very important. However, using such methods makes sense when data contains further dimensions (e.g. Space and time). In other words, the competitive part of our approach is its dimension-based change tracking which is valid only for multidimensional (multiway) data.

A challenge to the future research is to utilize EigenEvent in a real-world problem and evaluate its performance in the practice. This was one of our main limitations in this research. Unfortunately, there is no public real-world data available with ground truth for syndromic surveillance research. Most of bio-surveillance programs also correspond to the governmental sections where gaining access to data in most of the cases is impossible. Even if we have access to real data, the period of outbreaks or events is not specified in that. There is a recent developed simulator [LSY07] that simulates multivariate syndromic time series and outbreak signatures. However, this simulator does not support the spatial dimension. Therefore future work can be adapting that for spatiotemporal data sets.
❈❍❆P❚❊❘ ✸✳

✼✺

❖◆▲■◆❊✴❙❊▼■✲❙❯P❊❘❱■❙❊❉ ▼❊❚❍❖❉

❚❛❜❧❡ ✸✳✹✿ ◆✉♠❜❡r ♦❢ ❢❛❧s❡ ❛❧❛r♠s ♣❡r ②❡❛r ❢♦r ✶✵✵ ❞❛t❛s❡ts ❛✈❡r❛❣❡❞ ❢♦r ✷✸✶ ♣✲✈❛❧✉❡s
✭✵✳✵✷✵✱✵✳✵✷✶✱✳✳✳✱ ✵✳✷✺✵✮
❞❛t❛s❡t ❲❙✷✳✵

❲❙✷✳✺

❲❙✸✳✵

❊✐❣❡♥❊✈

❞❛t❛s❡t ❲❙✷✳✵

❲❙✷✳✺

❲❙✸✳✵

❊✐❣❡♥❊✈

✶

✼✵✳✵✶✸

✺✶✳✽✺✼✶

✺✼✳✽✹✽✺

✸✼✳✽✸✾✽

✺✶

✺✹✳✾✽✷✼

✸✻✳✶✻✵✷

✹✶✳✵✶✸

✷✹✳✼✽✸✺

✷

✵

✵

✵

✵

✺✷

✽✾✳✼✾✷✷

✺✶✳✷✾

✺✾✳✾✹✽✶

✺✶✳✻✾✼

✸

✵✳✹✹✶✻

✷✳✷✷✺✶

✶✳✽✶✽✷

✶✳✹✺✹✺

✺✸

✽✾✳✷✵✼✽

✻✷✳✶✾✾✶

✻✶✳✷✼✷✼

✹✻✳✸✵✼✹

✹

✼✵✳✹✼✶✾

✺✹✳✸✽✾✻

✺✸✳✺✸✻✽

✸✵✳✼✶

✺✹

✹✾✳✵✾✺✷

✸✶✳✼✹✵✸

✸✸✳✹✼✻✷

✶✵✳✹✽✵✺

✺

✻✾✳✺✾✼✹

✹✵✳✺✶✵✽

✸✾✳✺✼✺✽

✷✾✳✵✽✷✸

✺✺

✹✼✳✹✶✺✻

✹✷✳✶✾✾✶

✹✽✳✸✽✾✻

✷✾✳✷✸✽✶

✻

✾✸✳✼✼✾✷

✺✾✳✵✼✼✾

✻✷✳✵✸✹✻

✸✼✳✼✵✺✻

✺✻

✶✵✽✳✸✼✻✻ ✼✸✳✺✶✵✽

✼✶✳✸✸✸✸

✹✵✳✺✹✾✽

✼

✼✷✳✺✻✷✽

✺✵✳✺✺✽✹

✻✷✳✶✵✽✷

✸✺✳✸✷✵✸

✺✼

✺✾✳✻✵✶✼

✸✻✳✼✹✽✾

✸✽✳✸✶✶✼

✷✵✳✵✽✻✻

✽

✽✾✳✺✻✷✽

✼✶✳✵✸✾

✼✾✳✷✼✼✶

✺✵✳✵✸✵✸

✺✽

✷✺✳✶✷✾✾

✷✵✳✹✺✽✾

✷✷✳✶✾✹✽

✶✸✳✼✼✵✻

✾

✷✻✳✸✺✵✻

✶✼✳✻✽✹

✶✽✳✹✻✸✷

✽✳✷✹✻✽

✺✾

✾✵✳✹✸✼✷

✹✶✳✽✻✺✽

✹✶✳✻✼✾✼

✹✵✳✵✼✼✾

✶✵

✼✵✳✵✺✻✸

✺✸✳✽✽✸✶

✹✾✳✺✺✽✹

✸✹✳✼✺✸✷

✻✵

✼✾✳✵✾✾✻

✺✶✳✷✷✵✽

✺✷✳✸✵✼✹

✺✵✳✼✵✶✸

✶✶

✽✾✳✻✶✵✹

✻✸✳✷✷✺✶

✺✸✳✷✼✼✶

✹✽✳✸✻✸✻

✻✶

✼✵✳✹✸✷✾

✹✹✳✷✾

✺✵✳✹✷✽✻

✷✾✳✸✶✻

✶✷

✽✾✳✸✽✾✻

✼✶✳✸✸✸✸

✼✺✳✶✸✹✷

✹✽✳✹✽✵✺

✻✷

✸✶✳✺✹✺✺

✶✾✳✶✻✹✺

✶✾✳✼✽✼✾

✾✳✸✹✷

✶✸

✽✷✳✽✹✹✷

✺✶✳✵✾✾✻

✹✾✳✵✸✹✻

✹✵✳✵✸✹✻

✻✸

✺✾✳✸✹✻✸

✹✷✳✻✸✷

✹✺✳✷✺✶✶

✸✸✳✷✺✾✼

✶✹

✸✺✳✼✵✶✸

✷✺✳✷✺✾✼

✷✺✳✹✼✻✷

✶✼✳✻✸✷

✻✹

✼✷✳✻✹✵✼

✹✽✳✹✽✾✷

✺✶✳✾✷✷✶

✸✼✳✷✾

✶✺

✽✵✳✼✻✶✾

✹✹✳✻✾✷✻

✹✸✳✻✹✵✼

✸✹✳✼✽✼✾

✻✺

✷✾✳✷✷✺✶

✶✾✳✻✽✹

✷✵✳✷✽✺✼

✾✳✷✻✽✹

✶✻

✺✽✳✽✼✹✺

✹✷✳✼✵✺✻

✹✷✳✺✶✾✺

✷✺✳✼✷✼✸

✻✻

✸✳✷✼✼✶

✷✳✸✼✷✸

✸✳✾✽✼

✵✳✽✸✶✷

✶✼

✺✳✺✼✶✹

✹✳✷✷✵✽

✻✳✻✼✺✸

✶✳✸✾✸✾

✻✼

✸✾✳✽✸✾✽

✷✷✳✻✺✸✼

✷✷✳✾✸✺✶

✶✵✳✷✹✻✽

✶✽

✻✷✳✵✹✸✸

✹✷✳✵✻✵✻

✹✹✳✵✸✹✻

✶✻✳✷✸✸✽

✻✽

✸✵✳✺✽✹✹

✶✸✳✹✷✽✻

✶✺✳✽✵✺✷

✼✳✹✷✽✻

✶✾

✺✺✳✶✹✼✷

✸✽✳✼✺✸✷

✹✺✳✻✵✶✼

✸✷✳✾✺✻✼

✻✾

✾✳✵✹✼✻

✽✳✵✵✽✼

✶✵✳✺✾✸✶

✷✳✾✵✵✹

✷✵

✸✶✳✽✷✷✺

✸✶✳✸✽✾✻

✸✵✳✹✻✼✺

✶✼✳✸✹✻✸

✼✵

✶✳✵✾✺✷

✵

✵

✵

✷✶

✶✼✳✸✵✼✹

✾✳✾✷✷✶

✶✶✳✽✽✸✶

✽✳✸✶✶✼

✼✶

✼✳✾✻✶

✼✳✼✹✽✾

✾✳✺✷✽✶

✷✳✼✶✽✻

✷✷

✹✷✳✹✺✵✷

✷✷✳✸✺✵✻

✷✵✳✾✵✾✶

✶✶✳✶✼✼✺

✼✷

✾✳✾✼✹

✾✳✻✹✺

✼✳✼✽✸✺

✷✳✾✻✾✼

✷✸

✶✵✳✾✾✶✸

✼✳✷✷✺✶

✽✳✻✻✻✼

✵✳✾✺✷✹

✼✸

✸✺✳✼✼✹✾

✷✽✳✸✾✸✾

✷✼✳✵✹✼✻

✶✻✳✻✻✻✼

✷✹

✷✾✳✹✶✾✾

✷✵✳✷✸✸✽

✷✻✳✷✵✸✺

✶✵✳✶✷✺✺

✼✹

✶✽✳✺✽✵✶

✶✸✳✼✼✵✻

✶✸✳✵✺✻✸

✻✳✸✽✺✸

✷✺

✸✺✳✶✾✵✺

✷✺✳✸✸✼✼

✷✷✳✼✼✵✻

✶✹✳✺✹✺✺

✼✺

✼✽✳✵✹✸✸

✹✹✳✼✸✶✻

✺✻✳✾✶✸✹

✸✺✳✷✹✷✹

✷✻

✽✹✳✷✺✾✼

✺✹✳✼✷✷✾

✺✼✳✹✺✽✾

✺✵✳✵✾✵✾

✼✻

✼✸✳✶✵✽✷

✹✽✳✽✵✾✺

✹✽✳✻✼✺✸

✺✷✳✷✶✻✺

✷✼

✾✳✾✽✼

✾✳✽✵✺✷

✶✶✳✷✻✹✶

✹✳✹✸✼✷

✼✼

✽✳✽✻✶✺

✽✳✷✾✹✹

✽✳✸✾✽✸

✵✳✷✸✽✶

✷✽

✷✵✳✸✽✾✻

✶✻✳✹✻✸✷

✶✹✳✼✸✶✻

✸✳✶✼✸✷

✼✽

✷✾✳✺✷✸✽

✷✶✳✺✶✵✽

✶✷✳✺✹✺✺

✶✵✳✻✾✼

✷✾

✵

✵

✵

✵

✼✾

✼✻✳✹✻✼✺

✺✸✳✶✼✸✷

✺✺✳✾✻✶

✺✵✳✾✹✸✼

✸✵

✾✸✳✷✹✻✽

✻✺✳✵✼✸✻

✻✾✳✹✶✺✻

✹✽✳✹✶✾✾

✽✵

✹✽✳✺✶✺✷

✷✸✳✹✹✺✾

✷✸✳✺✸✷✺

✶✺✳✸✵✼✹

✸✶

✹✼✳✽✼✵✶

✸✺✳✺✶✵✽

✸✻✳✽✸✾✽

✸✷✳✽✺✷✽

✽✶

✹✳✷✸✸✽

✶✵✳✸✵✸

✾✳✶✻✹✺

✹✳✺✽✵✶

✸✷

✼✹✳✼✵✶✸

✻✵✳✼✽✸✺

✼✵✳✼✺✼✻

✹✵✳✸✾✸✾

✽✷

✸✽✳✸✶✻

✷✷✳✵✼✼✾

✷✶✳✾✹✽✶

✶✵✳✸✸✼✼

✸✸

✵

✵

✵✳✻✺✸✼

✵

✽✸

✽✺✳✾✵✵✹

✺✼✳✽✵✺✷

✺✼✳✶✼✼✺

✹✵✳✻✼✾✼

✸✹

✼✶✳✼✷✷✾

✹✵✳✻✻✷✸

✹✵✳✾✹✸✼

✷✵✳✵✽✻✻

✽✹

✽✳✸✵✼✹

✹✳✼✾✻✺

✹✳✺✶✺✷

✵✳✼✼✹✾

✸✺

✽✽✳✼✼✵✻

✺✹✳✼✹✽✾

✹✽✳✺✶✺✷

✺✷✳✶✹✼✷

✽✺

✷✵✳✸✵✸

✶✹✳✽✵✾✺

✶✷✳✹✾✼✽

✸✳✶✷✺✺

✸✻

✻✶✳✷✹✻✽

✺✵✳✹✼✶✾

✹✵✳✻✹✵✼

✸✾✳✽✷✻✽

✽✻

✼✼✳✻✵✻✶

✹✷✳✵✾✾✻

✹✾✳✽✼✹✺

✸✺✳✼✺✸✷

✸✼

✸✶✳✵✽✻✻

✶✺✳✹✼✶✾

✶✷✳✾✻✶

✶✻✳✽✾✶✽

✽✼

✷✷✳✺✻✼✶

✶✽✳✵✶✼✸

✷✷✳✹✹✺✾

✽✳✻✺✽

✸✽

✸✶✳✻✶✵✹

✷✸✳✷✼✼✶

✷✼✳✸✷✹✼

✶✽✳✼✻✻✷

✽✽

✽✳✸✼✷✸

✻✳✺✷✸✽

✼✳✵✸✵✸

✶✳✸✶✻

✸✾

✾✺✳✸✷✵✸

✻✺✳✶✸✽✺

✼✸✳✾✾✶✸

✹✻✳✽✽✸✶

✽✾

✺✼✳✺✶✵✽

✹✵✳✵✻✵✻

✺✶✳✶✷✶✷

✷✶✳✼✷✷✾

✹✵

✸✸✳✸✾✸✾

✶✽✳✾✸✵✼

✷✸✳✾✶✼✼

✶✺✳✽✽✼✹

✾✵

✻✼✳✼✶

✹✻✳✵✹✸✸

✺✽✳✶✸✽✺

✷✹✳✷✸✽✶

✹✶

✻✺✳✵✾✵✾

✸✾✳✽✾✶✽

✸✾✳✷✷✵✽

✸✶✳✷✾

✾✶

✻✼✳✼✼✹✾

✹✽✳✻✼✾✼

✹✸✳✶✶✻✾

✷✵✳✼✵✺✻

✹✷

✼✼✳✼✸✺✾

✹✸✳✼✷✼✸

✹✾✳✽✺✷✽

✷✹

✾✷

✻✽✳✸✶✻

✺✺✳✺✺✹✶

✺✻✳✷✵✸✺

✸✹✳✾✷✷✶

✹✸

✸✵✳✾✷✻✹

✷✵✳✹✼✶✾

✷✵✳✺✽✽✼

✶✵✳✹✼✻✷

✾✸

✻✻✳✾✽✼

✹✽✳✼✵✶✸

✹✼✳✶✵✸✾

✷✽✳✺✵✷✷

✹✹

✻✸✳✵✸✵✸

✹✻✳✵✸✾

✹✽✳✽✾✶✽

✸✻✳✹✹✶✻

✾✹

✷✽✳✻✾✼

✷✹✳✶✻✽✽

✷✼✳✼✵✺✻

✾✳✼✺✼✻

✹✺

✽✶✳✼✽✼✾

✹✼✳✼✼✾✷

✹✾✳✸✺✵✻

✹✺✳✷✼✼✶

✾✺

✻✾✳✹✻✸✷

✹✽✳✶✶✻✾

✹✼✳✷✾✹✹

✸✸✳✷✽✶✹

✹✻

✽✹✳✼✼✹✾

✻✺✳✼✻✻✷

✻✼✳✽✸✺✺

✹✶✳✹✺✽✾

✾✻

✸✹✳✵✷✶✻

✶✻✳✽✺✷✽

✶✽✳✷✶✷✶

✶✷✳✹✺✵✷

✹✼

✽✳✸✸✼✼

✶✶✳✶✻✽✽

✾✳✻✽✹

✵✳✺✶✵✽

✾✼

✹✳✵✾✾✻

✻✳✶✽✶✽

✼✳✺✹✾✽

✶✳✻✶✾

✹✽

✹✸✳✻✷✸✹

✶✾✳✼✽✼✾

✷✾✳✼✹✽✾

✶✺✳✹✷✹✷

✾✽

✸✹✳✵✹✼✻

✶✽✳✹✸✼✷

✶✺✳✵✵✽✼

✶✹✳✼✹✵✸

✹✾

✷✻✳✺✶✾✺

✶✹✳✶✽✻✶

✶✽✳✶✶✻✾

✶✺✳✹✷✽✻

✾✾

✺✻✳✵✾✵✾

✸✵✳✾✽✼

✹✷✳✽✹✽✺

✷✷✳✼✶✽✻

✺✵

✸✸✳✹✸✷✾

✸✶✳✻✼✾✼

✸✺✳✸✶✶✼

✶✻✳✺✻✼✶

✶✵✵

✻✻✳✼✾✷✷

✹✻✳✹✷✹✷

✺✵✳✺✽✹✹

✷✽✳✾✸✾✹


Table 3.5: Detection delay (in days) for 100 datasets averaged for 231 -values (0.020,0.021,..., 0.250)

<table>
<thead>
<tr>
<th>dataset</th>
<th>WS2.0</th>
<th>WS2.5</th>
<th>WS3.0</th>
<th>EigenEv</th>
<th>dataset</th>
<th>WS2.0</th>
<th>WS2.5</th>
<th>WS3.0</th>
<th>EigenEv</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.3463</td>
<td>2</td>
<td>1.3463</td>
<td>2</td>
<td>51</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3.3329</td>
</tr>
<tr>
<td>2</td>
<td>1.0433</td>
<td>1</td>
<td>1.2887</td>
<td>52</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2.0939</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2</td>
<td>12.3934</td>
<td>53</td>
<td>3.7835</td>
<td>8.7532</td>
<td>7.7066</td>
<td>4.8312</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>54</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>2.7273</td>
<td>55</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>4.7056</td>
<td>6.2035</td>
<td>5.2511</td>
<td>6.0346</td>
<td>56</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1.5065</td>
<td>57</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1.2251</td>
<td>1</td>
<td>1</td>
<td>1.6104</td>
<td>58</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>1.5628</td>
<td>1.2771</td>
<td>1.6688</td>
<td>59</td>
<td>5.0909</td>
<td>4.4992</td>
<td>6.1212</td>
<td>11.0606</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>9.1905</td>
<td>1</td>
<td>00</td>
<td>3.0399</td>
<td>2</td>
<td>1.5628</td>
<td>5.2826</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1.5974</td>
<td>61</td>
<td>1.9957</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>62</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>4.2078</td>
<td>1</td>
<td>1</td>
<td>7.8359</td>
<td>63</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>64</td>
<td>4.0665</td>
<td>13.9913</td>
<td>1.7316</td>
<td>2.926</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>2.3033</td>
<td>1</td>
<td>1</td>
<td>1.7316</td>
<td>65</td>
<td>4.0779</td>
<td>7.6364</td>
<td>10.4566</td>
<td>8.4502</td>
</tr>
<tr>
<td>16</td>
<td>1.039</td>
<td>1</td>
<td>1</td>
<td>1.0096</td>
<td>66</td>
<td>2</td>
<td>1.3435</td>
<td>2</td>
<td>1.5758</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.3600</td>
<td>67</td>
<td>2.8095</td>
<td>1</td>
<td>1</td>
<td>1.3506</td>
</tr>
<tr>
<td>18</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.025</td>
<td>68</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>19</td>
<td>1.3074</td>
<td>1</td>
<td>1.2338</td>
<td>70</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3.8139</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>71</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2.9341</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>72</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>73</td>
<td>4</td>
<td>7.5438</td>
<td>6.3463</td>
<td>5.3333</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>74</td>
<td>1.2597</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>75</td>
<td>1</td>
<td>2.1255</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>76</td>
<td>2</td>
<td>1.2121</td>
<td>1</td>
<td>1.2251</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>2.5511</td>
<td>2</td>
<td>1.2035</td>
<td>5.0649</td>
<td>77</td>
<td>1.0901</td>
<td>1</td>
<td>1</td>
<td>2.2468</td>
</tr>
<tr>
<td>27</td>
<td>1.4329</td>
<td>1</td>
<td>1</td>
<td>17</td>
<td>80</td>
<td>1.0779</td>
<td>7.4632</td>
<td>13.7403</td>
<td>11.2294</td>
</tr>
<tr>
<td>28</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>78</td>
<td>8.6883</td>
<td>2.0823</td>
<td>2</td>
<td>1.2338</td>
<td>80</td>
</tr>
<tr>
<td>29</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>79</td>
<td>5.4212</td>
<td>1.2597</td>
<td>1</td>
<td>2.3506</td>
<td>82</td>
</tr>
<tr>
<td>30</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>81</td>
<td>5.4212</td>
<td>1.2597</td>
<td>1</td>
<td>2.3506</td>
<td>82</td>
</tr>
<tr>
<td>31</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>83</td>
<td>6.6753</td>
<td>4.5022</td>
<td>7.3556</td>
<td>5.6147</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>84</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>85</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>86</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>87</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>88</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>89</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>90</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>91</td>
<td>2.1732</td>
<td>1.0866</td>
<td>1</td>
<td>7.3692</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>92</td>
<td>6.6956</td>
<td>3.4762</td>
<td>5.1429</td>
<td>11.1602</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>93</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>2.329</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>94</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>2.329</td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>95</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>2.329</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>96</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>2.329</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>97</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>2.329</td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>98</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>2.329</td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>99</td>
<td>1.3436</td>
<td>1</td>
<td>1</td>
<td>2.329</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>100</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.9567</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 4

Subproblem : Hotspot Detection

The EigenEvent algorithm developed in Chapter 3, as we previously mentioned, accounts only for signaling alarms and does not identify the affected areas. This problem is independently studied in this chapter in the broader sense of the hotspot detection problem. Hotspot detection aims at identifying subgroups in the observations that are unexpected, with respect to some baseline information. For instance, in disease surveillance, the objective is to detect sub-regions in spatiotemporal space, where the count of reported diseases (e.g. Cancer) is higher than expected, with respect to the population. The state-of-the-art method used for this kind of problem is the Space-Time Scan Statistics (STScan), which exhaustively searches the whole space through a sliding window looking for significant spatiotemporal clusters. STScan makes some restrictive assumptions about the distribution of data, the shape of the hotspots and the quality of data, which can be unrealistic for some nontraditional data sources. We propose a novel methodology called EigenSpot that instead of an exhaustive search over the space, tracks the changes in a space-time occurrences structure. Not only does the new approach present much more computational efficiency, but it also makes no assumption about the data distribution, hotspot shape or the data quality. The principal idea is that with the joint combination of abnormal elements in the principal spatial and the temporal singular vectors, the location of hotspots in the spatiotemporal space can be approximated. The experimental evaluation, both on simulated and real data sets, reveals the effectiveness of the proposed method.
CHAPTER 4. SUBPROBLEM : HOTSPOT DETECTION

4.1 Introduction

Hotspot detection which may come with different terminologies, such as outbreak detection, cluster detection or event detection is somehow related to the clustering and anomaly detection, however it is distinct from these two. In clustering, the entire data set is partitioned into some groups, but in the anomaly detection, the anomalous points are searched for and sought after. Hotspot detection addresses the same problem, but with this difference that anomalous instances are recognized given some baseline information. In other words, looking into the dataset, everything might seem normal, however, when the cases along the baseline are considered, some points might be considered unexpected. A realistic scenario of the hotspot detection is in disease surveillance. Suppose that we have the population of different postal codes, during a range of years, as the baseline information and the count of the reported diseases in a range of postal codes, throughout different years as the cases dataset. The goal is to detect those spatiotemporal regions that contain unexpected counts. For instance, an output like zones $S_1, S_2$ and $S_3$ during the years $T_1$ to $T_5$ might be considered a spatiotemporal hotspot. The detection of such hotspots enables the officials to better understand their target of interest for essential medical care and preventive measures.

The current methods for hotspot detection are twofold: clustering-based techniques and the scan statistics-based ones. Clustering-based techniques, such as [Lev06], infer some thresholds from the population data and then apply the thresholds for clustering of data points in the cases set. Their prominent benefit, as opposed to the other methods, is that they provide the exact shape of the clusters. However, handling complex data, such as spatiotemporal data is not straightforward for these techniques. Besides, the clustering methods do not consider chance and randomness issues, which are very important in sensitive applications, such as security and public health. Moreover, there is not a standard clustering method for hotspot detection, which is widely accepted by the community. The methods are mostly outspread and diverse, in terms of the technical details. The clustering methods also require restrictive input parameters, which make their usage limited in automatic settings.

The second group of techniques which relies on scan statistics are widely used and accepted by the epidemiological community. These groups of techniques exhaustively scan the whole space to find interesting spatial and spatiotemporal clusters. A specific statistic is computed for each possible window and then potential clusters are sorted, based on the obtained statistics. Thereafter, the statistical significance of the top-k clusters is simulated via a Monte-Carlo technique. Since these methods scan an
entire space, they are expensive to compute. Spatial scan statistics [Kul97] requires computation time of $O(N^3)$ and space-time scan statistics (STScan) [Kul97, KN95, Kul99, KAF+98] requires $O(N^4)$. Some recent efforts have been made to reduce this complexity. For instance, [AMP+06] proposes a method that requires $O\left(\frac{1}{\varepsilon}N^2\log^2 N\right)$ for spatial scan, which is more efficient than $O(N^3)$. However, the minimum complexity for space-time scan statistics in its best condition has not reached less than $O(N^3)$. This high computational cost practically has restricted their use in real-time applications or large-scale data sets. Besides, scan statistics-based techniques are highly associated with the strong parametric model assumptions (e.g. Poisson or Gaussian counts) [Nei06]. These assumptions mitigate the performance when the models are incorrect for nontraditional data sources. Additionally, scan statistics-based methods are not efficient for detection of irregular shape clusters [TT05, DAa04] apart from the pre-defined shapes such as circle (spatial scan) and cylinders (space-time scan). They also assume that data is presented in a high quality format, hence are vulnerable against the noises [NS07].

Our proposed method is a solution to some of the above mentioned issues, in scan statistics-based methods. An efficient method is proposed (linear with both space and time dimensions) for approximation of hotspots in the spatiotemporal space, without the need for exhaustive search. Instead of looking for deviations in the assumed parametric model, we track changes in the space-time occurrences structure, using the eigenspace techniques. This approach enables us to detect irregular shape hotspots from even noisy data sets, without any prior knowledge about the data nature or hotspot characteristics. To the best of our knowledge this has not already been addressed by other researchers. Our approach also differs from those of ones that focus on the improvement of scan statistics-based methods efficiency (e.g. [AMP+06, NM04, NC10]). We do not improve the efficiency of scan statistics-based methods; rather we propose and examine a new methodology, which follows a different aim. Hence, this is not an "apples to apples" comparison, as both groups of approaches have inherent differences and subsequently their own applications. STScan can be more helpful for retrospective and sensitive applications, when some prior knowledge exists about the nature of hotspot and data. On the other hand, our presented approach focuses more on real-time applications, where neither the nature of data nor the hotspot characteristics is known in advance. In such circumstances, a computationally feasible approximation method that rapidly identifies the alarming areas, without any prior knowledge might be very useful.

The rest of the chapter is organized as follows: Section 4.2 describes the problem,
the proposed solution and algorithm, as well as an illustrative example. Section 4.3 includes an experimental evaluation and results of the simulation study and the real case study. The last section concludes the chapter presenting the final remarks.

4.2 The proposed approach

4.2.1 The problem

Given a spatiotemporal count matrix for the cases, needed for the detection of those spatiotemporal regions (hotspots) that seems unexpected, given the baseline spatiotemporal matrix. Each cell in each matrix represents a count corresponding to a specific region and time. In particular, for disease outbreak detection, each cell in the baseline matrix represents the population corresponding to a region in a specific time period. Each cell in the matrix cases also represents the count of reported disease in a specific region, within a given time period, as well. The purpose is to determine those subgroups of the spatiotemporal space whose reported cases are unexpected.

A baseline method that can be applied to the problem is to compute the ratio of the cases to the population for all possible spatiotemporal regions (each cell in the spatiotemporal matrix) and then compute the z-score of the ratios. Then the null hypothesis $H_0$: there is no hotspot is rejected, in case some spatiotemporal regions with z-score greater than a threshold are found. This approach theoretically and practically, as will be illustrated later, imposes too many false alarms, since for a $n \times m$ matrix, it is required to perform $n \times m$ comparison tasks. In this chapter, an approach that performs only $n+m$ comparisons is proposed. Another approach can use a clustering method on the ratios. However, it suffers from the same problem of the baseline method (it requires $n \times m$ comparisons versus our $n+m$). Besides, the requirement for determination of appropriate cut-point or number of clusters adds more complexity and user intervention to the system. We are interested in developing a system, which has the following characteristics: 1) does not require any input parameter; 2) weighs all the possible hotspots, based on a standard metric like statistical significance (p-value). The benefit is that the output can be compared to relevant systems or methods. The alpha threshold is also easy to estimate (e.g. the common uses are 0.01 or 0.05).
CHAPTER 4. SUBPROBLEM: HOTSPOT DETECTION

4.2.2 The intuitions

In this section, we describe the logic and intuitions behind the proposed method. Assume that we have two identical \( n \times m \) matrices \( B \) (baseline) and \( C \) (cases) such that \( n \) is the number of components in the spatial dimension and \( m \) the number of components in the temporal dimension. As was mentioned in section 2.2.5, The SVD of the \( n \times m \) matrix is a factorization of the form \( A = UDV^T \). The \( n \) columns of \( U \) and the \( m \) columns of \( V \) are called the left-singular vectors and right-singular vectors of the matrix \( A \), respectively. The left-singular vectors correspond to spatial dimension, while the right-singular ones correspond to the temporal dimension. In order to clarify and elaborate, a new terminology spatial singular vector is used along with temporal singular vector that respectively refer to the principal left singular vector and the principal right singular one. Note that we take only the singular vector corresponding to the largest eigenvalue for the comparison, due to the fact that the first principal singular vector accounts for major directions of data. Hence, it explains or extracts the largest part of the inertia of the data [AW10].

Now, let us denote the spatial singular vector of baseline matrix (\( B \)) and cases matrix (\( C \)) respectively with \( \mathbf{s}_B = (s_{b1}, s_{b2}, ..., s_{bn}) \) and \( \mathbf{s}_C = (s_{c1}, s_{c2}, ..., s_{cn}) \). Then let's denote temporal singular vector of \( B \) and \( C \) respectively with \( \mathbf{t}_B = (t_{b1}, t_{b2}, ..., t_{bm}) \) and \( \mathbf{t}_C = (t_{c1}, t_{c2}, ..., t_{cm}) \). If we hypothetically assume that \( B = C \), then \( \mathbf{s}_B = \mathbf{s}_C \) and \( \mathbf{t}_B = \mathbf{t}_C \). In this condition, the angles between \( \mathbf{s}_B \) and \( \mathbf{s}_C \), and between \( \mathbf{t}_B \) and \( \mathbf{t}_C \) would be equal to almost zero. Now assume that some change occurs in the \( C \) and this change corresponds to a specific region and time period. Therefore, the matrices are no longer identical and subsequently the angles between their singular vectors rise up in value. From this angle change, we can only infer that some changes occur, but we do not know what subgroup of data is affected by this change. If we could identify those vector elements from \( \mathbf{s}_C \) and \( \mathbf{t}_C \) that caused this change, we would be able to identify the spatial and temporal components of the affected area. For instance, assume that through a hypothetical method we could identify that \( s_{c1} \) from the \( \mathbf{s}_C \) and \( t_{c1} \) from the \( \mathbf{t}_C \) corresponding to the affected area. If we remove the region corresponding to \( s_{c1} \) and time related to \( t_{c1} \) from both baseline and cases data sets, the matrices should again become identical. Hence, we would have the angles between the pair singular vectors equal to almost zero. Here, \( (s_{c1}, t_{c1}) \) is called hotspot and \( s_{c1} \) and \( t_{c1} \) are respectively called the spatial and temporal components of the hotspot. The process of finding these components is also called hotspot detection. Note that in this work, the angle between the singular vectors is not the criterion used. In the above, the angle concept is only used to explain the rationale behind the proposed method.
CHAPTER 4. SUBPROBLEM : HOTSPOT DETECTION

Some assumptions were made above, which were only for simplification of explanation. In practice, we rarely find two identical baseline and cases matrix. However, we are able to assume that in a normal condition, where no hotspot exist, both baseline and cases set should have a same space-time occurrences structure. In this case, the pair singular vectors of baseline and cases sets, regardless of the data distribution should stay in a constant distance. Now, if a hotspot initiates to grow in the cases set, this change can be directly observed from the changes in the elements of singular vectors. In such cases, some distances between the singular vector elements become abnormal for elements corresponding to the affected areas in both the spatial and temporal dimension. We exploit this idea to develop our algorithm for hotspot detection.

4.2.3 The EigenSpot Algorithm

Algorithm 4.1 EigenSpot

\[
// n: \text{number of items in the spatial dimension} \\
// m: \text{number of items in the temporal dimension} \\
// B: \text{Baseline } n \times m \text{ spatiotemporal matrix} \\
// C: \text{Cases } n \times m \text{ spatiotemporal Matrix} \\
// \alpha: \text{Statistical significance level (e.g. } 0.05) \\
\]

\begin{algorithm}
\begin{algorithmic}
  \State $[\mathbf{sb}, \mathbf{tb}] = 1\text{-rank SVD (B)}$ \Comment{baseline : sb: spatial singular vector, tb: temporal singular vector}
  \State $[\mathbf{sc}, \mathbf{tc}] = 1\text{-rank SVD (C)}$ \Comment{cases: sc: spatial singular vector, tc: temporal singular vector}
  \For{$i = 1:n$}
    \State $\mathbf{ds}_i = \mathbf{sc}_i - \mathbf{sb}_i$ \Comment{ds: subtract vector corresponding spatial dimension}
  \EndFor
  \For{$j = 1:m$}
    \State $\mathbf{dt}_j = \mathbf{tc}_j - \mathbf{tb}_j$ \Comment{dt: subtract vector corresponding temporal dimension}
  \EndFor
  \State \text{Spatial out of control elements $\leftarrow$ Z-score Control Chart ($\mathbf{ds}, \alpha$)}
  \State \text{Temporal out of control elements $\leftarrow$ Z-score Control Chart ($\mathbf{dt}, \alpha$)}
  \State \text{Hotspots $\leftarrow$ All joint combination of out of control elements in spatial and temporal dimensions}
\end{algorithmic}
\end{algorithm}

In this section, the proposed algorithm EigenSpot is described in detail. The inputs of the algorithm 4.1 are matrices for the baseline and cases with dimension of $n \times m$ where $n$ represents the number of regions and $m$ represents the number of temporal instants. We initiate by decomposing both matrices, using one-rank SVD. The one-
rank SVD gives us the principal singular vector corresponding to the spatial and
temporal dimensions (lines 1-2). The reason why the low-rank SVD is applied versus
the full-rank SVD is that our approach requires only the principal singular vector for
each matrix. The full-rank SVD is a more expensive method, because of the fact that
a $N \times N$ matrix requires $O(N^3)$ while the low-rank SVD requires $O(kN^2)$ where in our
case $k=1$ and therefore we require only $O(N^2)$ for each one-rank SVD. The principal
singular value accounts for the major directions of data in both cases and baseline,
therefore, it is appropriate for matching purposes.

In the next step, we subtract each element of the pair singular vectors together (lines
3-8). If we denote the spatial singular vector for baseline with $(sb_1, sb_2, ..., sb_n)$ and
spatial singular vector for cases with $(sc_1, sc_2, ..., sc_n)$, the subtract vector would be
d$s = (ds_1 = sc_1 - sb_1, ds_2 = sc_2 - sb_2, ..., ds_n = sc_n - sb_n)$. Similarly for the temporal
dimension we have d$t = (dt_1 = tc_1 - tb_1, dt_2 = tc_2 - tb_2, ..., dt_m = tc_m - tb_m)$. Subsequently, in order to identify the spatial and temporal components of the hotspot,
a $z$-score control chart is applied on vectors d$s$ and d$t$ with significant level $\alpha$. To do
so, the standardized vector of $z$-scores is first computed for d$s$ and d$t$. Thereafter, we
obtain the equivalent two-tailed p-value for each $z$-score. Finally, those components
of d$s$ and d$t$ that obtain p-value lower than $\alpha$ are considered abnormal. Eventually,
a joint combination of all spatial and temporal components to the original space gives
us the approximation of hotspots.

For instance, assume that $sb = (0.25, 0.10, 0.75, 0.20)$ is the spatial singular vector
of baseline and $sc = (0.30, 0.90, 0.80, 0.15)$ be the spatial singular vector of cases.
Each element in the spatial singular vector corresponds to a specific region. For
instance, 0.30 and 0.25 in the first element correspond to region 1. Similarly, the
second, third and the fourth element correspond to the region 2, 3 and 4, respectively.
The angle between the two singular vectors $sb$ and $sc$ is equal to 68 degrees in this
example. This angle does not tell us what elements of singular vector have contributed
to this difference. However, if in the above example we remove region 2 from the
system, we have two vectors $sb = (0.25, 0.75, 0.20)$ and $sc = (0.30, 0.80, 0.15)$ where
the angle between them is equal to 0.09 which is almost equal to zero. Region 2 in this
example is equivalent to the spatial component of the hotspot. In order to identify
the region 2 in this example, a $z$-score control chart is applied on the subtract vector
d$s = (0.25 - 0.30, 0.10 - 0.90, 0.75 - 0.80, 0.20 - 0.15) = (-0.05, -0.80, -0.05, 0.05)$. Afterwards, we compute the standardized $z$-scores of the subtract vector, which in
this case is $zds = (0.4119, -1.4893, 0.4119, 0.6654)$. $z$-score of -1.4893 is equivalent to
the left-tailed P-value of 0.06. If we define $\alpha = 0.10$, region 2 would be identified as
Figure 4.1: EigenSpot Algorithm; an Illustrative example. The goal of the approach is the identification of the shaded area in the cases matrix. The values c and b in the baseline and cases matrix are counts corresponding to a spatiotemporal window. The process is composed of the following four steps: 1) matrix decomposition; 2) subtraction of pair singular vectors elements; 3) applying the z-score control chart on the subtract vector; and 4) combining the spatial and temporal hotspots components.

**The Algorithm Complexity.** If we assume that we have N regions and N time instants, EigenSpot requires two \(O(N^2)\) for two one-rank SVD for cases and baseline matrices and two \(O(N)\) for elements matching corresponding spatial and temporal dimensions. This makes the EigenSpot require only \(O(2N^2) + O(2N) = O(N^2)\) which is much more efficient than the STScan. Because, the STScan requires \(O(NlogN)\) and \(O(N^2logN)\) for finding the relevant time and space cylinders and \(O(N^4)\) for finding the space-time cylinders as intersections of space and time cylinders [AaTK04]. Therefore, a single execution of the STScan procedure takes \(O(NlogN) + O(N^3logN) + O(N^4) = O(N^4)\).

### 4.2.4 Illustrative Example

Figure 4.1 demonstrates an illustrative example of how a hotspot can be identified by the EigenSpot algorithm. We are given two sets of baseline and cases that encompass three regions within four time windows. Each region can be a postal code or a city.
Each temporal window also can be a time period, such as a year (e.g. $T1 = 2010$). If we represent these two sets as a matrix, we have two sets of $3 \times 4$ matrices such that each cell represents the count. For instance, $b11$ represents the population of region 1 at time window $T1$ and $c32$ represents the count of reported disease in region 3 within the temporal window $T2$. The shaded area in the cases matrix (conjunction of third row with first-second columns) is the hotspot of interest that is required to be detected by the method. As demonstrated, the principal singular vector corresponding to the spatial and temporal dimensions is obtained via one-rank SVD. As a result, we have two singular vectors corresponding to the spatial and temporal dimensions for each set. In the next step, we subtract elements of each singular vectors pairs together. Therefore, we would have two vectors $dt$ and $ds$ which represent subtract vectors for the temporal and spatial dimensions, respectively. As demonstrated, $dt$ has four elements and $ds$ has three elements, each of which corresponds to the original regions and temporal windows (e.g. $dt1$ corresponds to $T1$ and $ds1$ corresponds to region 1). In the next step, we apply a z-score control chart with significance level $\alpha$ (e.g. $\alpha = 0.05$) on both of these vectors to identify their abnormal elements. As it is hypothetically shown in the example, $T1$ and $T2$ are identified as temporal hotspot components, and region 3 is identified as the spatial hotspot component. We only need to combine spatial components with temporal components to approximate the hotspots in the spatiotemporal space. As shown, the identified hotspot of Region 3, $T1,T2$ is equivalent to the shaded area in cases matrix (the target).

4.3 Experimental Evaluation

In this section, the effectiveness of our proposed approach is assessed, through an experimental study. The data sets used in evaluation of hotspot detection techniques are usually threefold [BBC+05]: 1) wholly simulated: both baseline and cases and hotspots are simulated; 2) semi-realistic: baseline is taken from a real population, but cases and hotspots are simulated; and 3) real data: both baseline and cases are real and hotspots are verified by a domain specialist. In this chapter, we evaluate the proposal, using the latter two strategies. We evaluate the algorithm performance, via the simulation study (section 4.3.1) and a real-world data (section 4.3.2). All experiments are conducted on a PC with Intel Core 2 Duo CPU and 3GB Ram. We use MATLAB 7 for the algorithm implementation and experiments and SatSan 9.2 [Kul12b] for experimenting with STScan.
CHAPTER 4. SUBPROBLEM: HOTSPOT DETECTION

Our method is compared with two other techniques, including the STScan and a baseline method. STScan [Kul97, KN95, Kul99, KAF+98] exhaustively moves a varying radius and height cylinder over the whole spatiotemporal space. The height of the cylinder represents the time dimension and the surface corresponds to the space dimension. Furthermore, it scores each possible cylinder, based on likelihood ratio statistics. Next, it sorts cylinders based on an order of the highest to the lowest score. Finally, a randomization test is performed for obtaining the cylinders statistical significance. Those cylinder whose p-value are lower than $\alpha$ (e.g. 0.05) are returned as hotspots. In the baseline method, we compute the ratio of the count of cases to the corresponding population for each matrix cell and then we compute the z-score of the obtained ratios and obtain the p-value from z-score. Afterwards, we signal an alarm, when p-value for a cell goes lower than $\alpha$.

4.3.1 Simulation Study

Here, we describe how the simulated data is generated and subsequently present the obtained result.

4.3.1.1 Data Generation

We generate 1500 sets of semi-real data, based on the extracted baseline data set from [KAF+98]. The baseline set includes the spatiotemporal distribution of population in New Mexico, USA during 1973-1991 ($m=19$). In order to simulate the cases count, we initially obtain the maximum likelihood of the parameter of the Poisson distribution, $\lambda$ from the first year of the baseline set. Let the vector of counts for the first year be $(c_1, c_2, ..., c_i)$ where $1 \leq i \leq n$ ($n$:number of spatial items which in this case is 39). $\lambda$ can simply be obtained by computing the means of the vector. Then, we multiply $\lambda$ by a fixed constant of 1.2% for subsequent years (1.2% is the average population growth rate). Next, we generate random numbers from the Poisson distribution with corresponding estimated parameters for each year. In order to inject the hotspot into the cases, we select a matrix window with size $H \times H$ (hotspot size) and multiply the counts inside the window by a fixed value of $I$ (hotspot impact). We then vary $H$ from 1 to 5 and select $I$ from (1.5, 2, 2.5). Since we generate data sets based on the random numbers, we generate 100 datasets for each setting to reduce the effect of randomness. Next section explores the evaluation results.
4.3.1.2 Performance Evaluation

In any hotspot detection approach, we determine a decision threshold to distinct hotspots from non-hotspots. Determination of this threshold becomes more important in sensitive applications, such as security and public health. For such applications, the evaluation of methods has to be made within different ranges of decision thresholds.

ROC curve [ZC93] is a widely accepted method for such evaluation tasks. However, due to two reasons, ROC curve cannot be used as an appropriate strategy for the evaluation in this simulation study. On one hand, we want to evaluate the method performance on 100 random data sets for each 15 setting, therefore we have 1500 data sets, which require the analysis of 1500 ROC curves, which is infeasible. We also cannot reduce the number of data sets to one, because we are generating random sets and if we rely only on one data set then our results would be highly dependent on the chance and randomness. At the first glance, the Area Under ROC Curve (AUC) seems to be an appropriate choice, as the AUC does not have user-defined parameters. Besides, it is a summarized scalar and seems to be appropriate for mass comparison of the methods. However, the main criticism about the use of AUC in applications, such as hotspot detection and outbreak detection is that AUC considers all thresholds equal, which is not true in many applications. In practice, in sensitive applications, such as epidemiology where we deal with human lives, we are not interested in knowing how a method performs in high alpha values. The used operational and practical p-value is usually a low value. In other words, the alpha of interest is not between 0 and 1, rather is limited to lower values. For this reason, instead of AUC we opt to use
an averaging strategy for operation thresholds [WMCW05]. We compute the average accuracy for a range of operational significance levels, such as alpha from 0.20 to 0.01 for each data set and then the average obtained values for all 100 data sets for each setting. The range of alpha is obtained as follows: We vary z-score from 1.28 to 3 (equivalent to two-tailed p-value of 0.2005 to 0.0027) and then increase z-score 0.1 in each loop.

We compare our method performance against both the STScan and the baseline approach, via control chart on ratios, described in section 4.2.1. The accuracy of methods in the identification of simulated hotspots is used as the criterion for the performance evaluation. The results are presented in Table 4.1. As seen, EigenSpot presents a better performance in almost all settings, except for low-impact hotspots, such that comparing the results in Table 4.1 we can observe that STScan is more sensitive to small hotspots like $1 \times 1$ and $2 \times 2$ with low-impact of 1.5. However, EigenSpot outperforms STScan in the rest of settings which can be due to two reasons. One reason is related to the inherent methodological difference between the EigenSpot and STScan. STScan search the whole space to find some spatiotemporal windows that the data distribution inside them has some deviation to the standard distribution models (e.g. Poisson). This strict assumption makes this approach less effective, when the data in each of the sets does not exactly follow the standard distribution model or some deviation occurs by chance. EigenSpot, instead of putting this strict restriction, searches for changes in the occurrence patterns and therefore is less sensitive to the deviations in data distribution. The second reason could be that EigenSpot is a shape-free method and does not search for a particular shape hotspot, while STScan looks for specific shape hotspots. Some accuracy loss in STScan relates to different shapes of the simulated hotspot. STScan looks for cylinder-shape hotspots, while the hotspots can be in different shapes like cubic.

The results also show the performance of each method in dealing with noises. We intentionally design some low-impact and size settings for evaluating the ability of the methods in handling noises. A low-size and low-impact region like impact of 1.5 and size $1 \times 1$ seem more likely to be a noise, rather than a realistic hotspot. Therefore, we expect that the methods do not detect that region as hotspot and ignore that.

In other words, the detection of such hotspot shows how a method wrongly identifies the noises as hotspots. Hence, the lower accuracy in this setting reveals the better performance of the method in dealing with noises. Since the EigenSpot is a spectral method, it definitely ignores such noises and does not report them as hotspots, while STScan is vulnerable against such circumstances. For this reason, it presents a higher
Figure 4.2: Mean accuracy for 16000 data sets averaged for 173 $\alpha$ from 0.20 to 0.01.

Table 4.2: The effect of hotspot size and impact on the performance (One-way ANOVA test).

<table>
<thead>
<tr>
<th>Factor</th>
<th>STScan</th>
<th>EigenSpot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hotspot size</td>
<td>$p = 1.6826 \times 10^{-10}$</td>
<td>$p = 5.9713 \times 10^{-13}$</td>
</tr>
<tr>
<td>Hotspot Impact</td>
<td>$p=0.1834$ (for impacts $\geq 2.5$)</td>
<td>$p=0.9337$ (for impacts $\geq 1.75$)</td>
</tr>
</tbody>
</table>

performance for low-size and impact regions. In the experiment, we presumed that a hotspot with small size of $1 \times 1$ or $2 \times 2$ and low impact of 1.5 is more a noise and not a real hotspot. However, since the hotspots are simulated, this is only an assumption. We may interpret the result in other way. If we assume that impact 1.5 is not noise and reveals a real hotspot, then we can infer that STScan outperforms EigenSpot for hotspots with low impacts and sizes.

4.3.1.3 Effect of hotspot size and impact on the performance

In the previous section, we evaluated the performance of methods for some limited important settings. In this section, with the same method of data generation and evaluation criterion, we study the stability of two algorithms STScan and EigenSpot for a wider range of hotspot sizes and impacts. We do not study the baseline method at this stage, since it is defeated in all previous cases. Hence, this time the hotspot size (H) varies from 1 to 10 and we test 16 different hotspot impacts (I) from 1.25 to 5 by step of 0.25. As formerly used, for each setting we generate 100 random data sets. Therefore, we generate $16000 = 10 \times 16 \times 100$ data sets. We then apply STScan and EigenSpot on all data sets and measure their average accuracy on 100 data sets for each setting in the operational significance levels ($p$-values) from 0.20 to 0.01. Figure 4.2 shows the result of this comparison. In order to see that whether this
Figure 4.3: Mean accuracy of STScan and EigenSpot corresponding to different settings for $173 \alpha$ from 0.20 to 0.01 averaged for 100 datasets.
improvement is obtained by chance or is statistically significant, we perform a paired student’s t-test between two sets of obtained performances for STScan and EigenSpot. The t-test confirms that the obtained improvement is statistically significant with $p-value = 3.3591 \times 10^{-89} \approx 0$.

Figure 4.3 shows the performance of methods against different hotspot sizes and impacts. The lowest performance for EigenSpot is obtained for impacts of 1.25 and 1.5. However, we can observe that both methods are relatively robust for a hotspot impact greater than a threshold. For instance, EigenSpot is robust for impacts over 1.75 and STScan is robust for impacts over 2.5. Regarding the hotspot size, EigenSpot has a descending trend by increasing the hotspot size. This implies that by increasing the hotspot size, we should expect lower performance from EigenSpot. It makes sense, as by increasing the size of hotspot, the affected areas gradually start to seem normal and are left undetected, via a spectral method like EigenSpot. EigenSpot, however, exhibits more regular behavior comparing STScan in this matter. The variance of performance is almost zero for EigenSpot during different size of hotspots, while it can vary up to 0.20 for STScan. STScan, also opposed to EigenSpot, experiences both ascending and descending trend. For hotspot sizes $1 \times 1$ to $6 \times 6$ has an ascending trend and then tend to decrease for bigger sizes. For hotspot $9 \times 9$ it has relatively the same performance as $1 \times 1$.

To understand whether the hotspot size and impact affect the performance of the methods, we perform an ANOVA test [MMM84] on the obtained performance for different hotspot sizes and impacts. The null hypothesis $H_0$ is that the mean accuracy does not change for different sizes and impacts. The test result (Table 4.2) confirms our initial guess that both STScan and EigenSpot become independent of hotspot impact when impact goes upper than a specific threshold. However, a very low p-values for hotspot size indicates that the performance of both EigenSpot and STScan is dependent on the hotspot size. However, as observed, both methods do not differ in their dependence on hotspot impact.

4.3.1.4 The effect of SVD Implementation

The central technique used in EigenSpot is the SVD. Two kinds of SVD can be used for this purpose: a full-rank SVD and a low-rank SVD. Here, four of SVD implementations are chosen, two from each category and their effect is studied on the EigenSpot performance. Table 4.3 demonstrates the average accuracy for 1500 data sets for the range of p-values from 0.20 to 0.01. As it is seen, the ARPACK implementation
Table 4.3: Average accuracy for 1500 data sets averaged for $173\alpha$ from 0.20 to 0.01

<table>
<thead>
<tr>
<th>Method</th>
<th>Computation Cost</th>
<th>Implementation</th>
<th>Average Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-rank SVD</td>
<td>$O(N^2)$</td>
<td>ARPACK</td>
<td>0.8975</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IncPACK</td>
<td>0.8387</td>
</tr>
<tr>
<td>Full SVD</td>
<td>$O(N^3)$</td>
<td>LAPACK</td>
<td>0.8429</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PROPACK</td>
<td>0.8177</td>
</tr>
</tbody>
</table>

Figure 4.4: Detected hotspot via STScan (left) and EigenSpot (Right)

[Sor97] (the default SVD implementation we use in the experiments) outperforms other methods. However, since both ARPACK and IncPACK [Bra06] have the same computational cost, we perform an ANOVA test [MMM84] to see whether using IncPACK affects the performance or not. The ANOVA test shows that two sets of accuracy obtained from these two implementations are not statistically different (p-value=0.26). Therefore, we can conclude that low-rank SVD implementation used does not affect the EigenSpot performance. Concerning full-rank SVD, we observe that LAPACK [And99] outperforms PROPACK [Lar98], however full-rank SVD, because of its computational cost is not of interest.

4.3.2 Experiment with real data

In this section, we study the performance of EigenSpot on a real data set. The data set which is publicly available [Ku11a] is provided by surveillance, epidemiology and end results (SEER) program of the National Cancer Institute and collected by the New
Mexico Tumor Registry between the years of 1973 to 1991 for 32 sub-regions of the New Mexico State, United States. There are 1175 reported cases of malignant neoplasm of the brain and the nervous system. The goal of the initial study was a response to a serious concern in 1991 in the New Mexico resident community about the correlation of wartime nuclear activities in Los Alamos with the recent brain tumor deaths in the neighborhood. The concern rapidly emerged at the local and national level and therefore became a center of attention by the local health departments. The data set was gathered, via a comprehensive review of the reported brain cancer incidence rates for the year 1973 through 1991 in order to identify the statistically significant spatiotemporal affected areas. STScan is already applied to this data [KAF+98]. The conclusion made from the previous study shows that excess of brain cancer in Los Alamos falls within the realm of chance, which confirms the final conclusion of the New Mexico Health Department.

In order to compare the EigenSpot with STScan, the EigenSpot is applied on the same data set. In addition to the settings used in the initial study [KAF+98] we use the adjusted incidences for temporal trends, age, race and sex. The results obtained via STScan and EigenSpot are shown in Table 4.4. STScan reports only Santa Fe and Los Alamos in the years 1986-1989 with a relative high p-value=0.45 which indicates that there is no significant hotspot. Applying EigenSpot with \( \alpha = 0.05 \) we could find a hotspot, including areas of Santa Fe, Bernalillo, Valencia as spatial components and the years 1981 and 1987 as the temporal components. However, for \( \alpha = 0.01 \) EigenSpot does not find any hotspot. If we look the hotspot spatial positions in Figure 4.4 we can find a meaningful relationship between STScan and EigenSpot results. Both candidate areas are found close the Los Alamos, the region of nuclear activities. The temporal component of 1987 also is appeared in the EigenSpot, which is located in the period detected by STScan. Based on results obtained via EigenSpot, it seems that in addition to the area close to Los Alamos and Santa Fe more areas were affected by the nuclear activities. These areas are Bernalillo and Valencia where the neighbors of Santa Fe and Los Alamos are. The very interesting point about EigenSpot is that EigenSpot opposed to STScan was not aware about the geographic relationship of the

### Table 4.4: Comparison of STScan versus EigenSpot in detection of hotspots. Incidence rates were adjusted for temporal trends, age, race and sex.

<table>
<thead>
<tr>
<th>Method</th>
<th>Affected Regions</th>
<th>Temporal Period</th>
<th>P-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>STScan</td>
<td>Santa Fe and Los Alamos</td>
<td>1986-1989</td>
<td>0.45</td>
</tr>
<tr>
<td>EigenSpot</td>
<td>Santa Fe, Bernalillo, Valencia</td>
<td>1981, 1987</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Mexico Tumor Registry between the years of 1973 to 1991 for 32 sub-regions of the New Mexico State, United States. There are 1175 reported cases of malignant neoplasm of the brain and the nervous system. The goal of the initial study was a response to a serious concern in 1991 in the New Mexico resident community about the correlation of wartime nuclear activities in Los Alamos with the recent brain tumor deaths in the neighborhood. The concern rapidly emerged at the local and national level and therefore became a center of attention by the local health departments. The data set was gathered, via a comprehensive review of the reported brain cancer incidence rates for the year 1973 through 1991 in order to identify the statistically significant spatiotemporal affected areas. STScan is already applied to this data [KAF+98]. The conclusion made from the previous study shows that excess of brain cancer in Los Alamos falls within the realm of chance, which confirms the final conclusion of the New Mexico Health Department.

In order to compare the EigenSpot with STScan, the EigenSpot is applied on the same data set. In addition to the settings used in the initial study [KAF+98] we use the adjusted incidences for temporal trends, age, race and sex. The results obtained via STScan and EigenSpot are shown in Table 4.4. STScan reports only Santa Fe and Los Alamos in the years 1986-1989 with a relative high p-value=0.45 which indicates that there is no significant hotspot. Applying EigenSpot with \( \alpha = 0.05 \) we could find a hotspot, including areas of Santa Fe, Bernalillo, Valencia as spatial components and the years 1981 and 1987 as the temporal components. However, for \( \alpha = 0.01 \) EigenSpot does not find any hotspot. If we look the hotspot spatial positions in Figure 4.4 we can find a meaningful relationship between STScan and EigenSpot results. Both candidate areas are found close the Los Alamos, the region of nuclear activities. The temporal component of 1987 also is appeared in the EigenSpot, which is located in the period detected by STScan. Based on results obtained via EigenSpot, it seems that in addition to the area close to Los Alamos and Santa Fe more areas were affected by the nuclear activities. These areas are Bernalillo and Valencia where the neighbors of Santa Fe and Los Alamos are. The very interesting point about EigenSpot is that EigenSpot opposed to STScan was not aware about the geographic relationship of the
regions. STScan knows in advance that for instance whether Santa Fe and Bernalillo are neighborhood or not, while EigenSpot does not have this prior knowledge. Based on the EigenSpot result, we can infer that the effect of nuclear activities in the neighborhood has experienced two peaks during the years 1981 and 1987. It makes sense, because the initial concerns about the effect of the nuclear activities started in 1991, four years right after 1987 (second detected temporal component). Indeed, EigenSpot has truly approximated the hotspot spatially and temporally very close the nuclear activity event. Most interestingly, no other meaningless hotspots are detected by EigenSpot. However, lack of strong p-value for the recognized area reveals that the neighborhood has been under a low effect of nuclear activities in Los Alamos, but has not had enough support to be considered alarming. If we do not consider $\alpha = 0.05$ significant, as [KAF+98] we can confirm initial conclusion about the random incidence of brain cancers in Los Alamos.

4.4 Conclusion and future works

We propose a new methodology based on matrix factorization for hotspot detection. We evaluate and compare the performance of the algorithm for detection of a single hotspot against the state-of-the art and a baseline method through a comprehensive simulation study. The obtained results indicate a statistically significant improvement over the state-of-the-art method STScan in particular settings. This improvement comes from the inherent methodological differences of the two approaches. The STScan uses the deviation in probability model as the criteria for identification of hotspots, while our approach tracks the changes the correlation patterns in spatial and temporal dimension to approximate the hotspot location. Besides, our approach is a shape-free method and contrary to STScan it is more robust to the noises. Additionally, Our approach is more efficient than the scan statistics based approaches. The main benefit of our approach is that it has linear complexity, in terms of both space and time.

We also study the effect of hotspot size and impact in the methods performance. Based on this result, both the STScan and EigenSpot are found independent of the hotspot impact in some specific ranges. However, both methods are dependent on the hotspot size. Nevertheless, EigenSpot exhibits a more regular trend against changes in hotspot size and impact. We also study the effect of SVD implementation on the Eigenspot performance. The result shows that there is no statistical difference between two low-
rank SVD implementations ARPACK and IncPACK. Therefore, SVD implementation
does not affect the performance of EigenSpot. Finally, we apply EigenSpot to a real
data set and compare its performance to STScan. EigenSpot, as well as the STScan
recognize the affected area close to the nuclear activity area, both in space and time,
however as well as STScan, cannot provide the strong statistical evidence to identify
this area as hotspot.

Although EigenSpot is an ideal solution in terms of, both accuracy and computational
cost for single hotspot detection. There is a doubt that this result is valid when multi-
ple hotspots exist. In this work, we did not evaluate the performance of EigenSpot for
multiple hotspot detection. However, theoretically we expect that STScan performs
better for that purpose. Because, combining the spatial and temporal components
of different hotspots together raise many false positives, which reduces the method
performance. Future research includes investigation of this problem and adapting
EigenSpot for multiple hotspots.
Chapter 5

Subproblem: Event Labeling

Event labeling is the process of marking events in unlabeled data. Traditionally, this is done by involving one or more human experts through an expensive and time-consuming task. In this chapter, we address the event labeling issue motivated by the baseline data creation sub-problem in Chapter 3. We propose an event labeling system relying on an ensemble of detectors and background knowledge. The target data is the usage log of a real bike sharing system. We first label events in the data and then evaluate the performance of the ensemble and individual detectors on the labeled data set in the absence and presence of background knowledge. Our results show that when there is no access to human experts, the proposed approach can be an effective alternative for labeling events. In addition to the main proposal, we conduct a comparative study regarding the various predictive models performance, semi-supervised and unsupervised approaches, train data scale, time series filtering methods, online and offline predictive models, and distance functions in measuring time series similarity.

5.1 Introduction

Event labeling is recognized as a basic function in surveillance and monitoring systems. Labels are essential for the evaluation of the algorithms and for incorporation in real time systems. However, event labeling is an expensive and time-consuming task which requires synergy of one or more human experts. Several solutions have been developed to avoid performing human-based labeling. The first group of methods relies on the creation of artificial and simulated data [LCD04, NSA+08, RNH+09, SLO+07] so that
both normal and abnormal instances are generated via simulation. In the second
group, events are injected on real background data [BBC+05, JBP07]. However, the
ideal scenario is to have access to ground truth data [LHF+00] where both normal and
abnormal instances are labeled without simulation. The first and second solutions
suffer from two issues. Firstly, they do not reflect the reality [RSR08] and secondly it
is extremely difficult to develop a simulator that generates data close to the ground
truth [FP01]. Besides, availability of ground truth data is limited or has been under
some criticisms (e.g. [McH00, TBLG09]).

Regardless of learning methodologies, evaluation of event detectors is still highly
dependent on human efforts. In supervised event detection, both normal and abnormal
instances are required to be labeled by human experts. In semi-supervised approaches
normal instances should be labeled manually. In unsupervised methods, detected
events are required to be verified by human experts. However, in practice, labeling or
verification of events by human expert can be extremely time-consuming and expensive
[RSR08]. In order to solve this problem some efforts have been made to assist users
to label data more efficiently via a graphical user interface [RSR08]. However, such
methodologies still are human-dependent to a great extent.

An automatic event detection system ought to operate without intuitive dependency
on human resources neither in providing labeled data nor in verification of alarms.
One alternative for human knowledge can be computer-based knowledge resources.
Although there is a lot of non-human knowledge sources, still a large number of event
detection systems rely on human-based knowledge and other knowledge sources are
not yet incorporated. For instance, available knowledge inside search engines, meta-
image and meta-video databases, data archives, data warehouses rarely is incorporated
into event detection problems. Only a few research addressed this issue. For instance,
[ATS12] use background knowledge from reliable sources of information (e.g. CNN,
BBC) for matching and validation of detected events on twitter streams. [CDK+08]
use an event ontology, called BioCaster, as background knowledge for detection of
infectious disease outbreaks from linguistic signals on the web. [SMG09] present an
approach for creation of domain ontology as background knowledge for detection of
events in surveillance video. [XZZ+08] use web cast text as background knowledge
source for detection of events in sport videos. However, none of the above works target
the elimination of human role from the detection cycle, rather background knowledge
is used as a supportive role.

The essential tool for elimination of the human role from the detection cycle is
the possessing of a highly reliable detector. However, different event detectors are
not equally capable of detecting events and subsequently different detectors perform differently on different environments [TM04, TM05, WFP99, Agg13]. This is due to the inconsistency of detector performance [GSS99, WFP99]. But how can we overcome this difficulty? When we want to make an important decision in our daily routine, we probably ask for recommendations from different people from different perspectives. The similar idea is already broached in the machine learning which is entitled ensemble learning. Ensemble learning is robust solution for more accurate and relatively domain-independent classification and clustering [Agg13, Die00]. It also can be embedded in parallel computing paradigm to improve the efficiency [SW09]. However, the application of ensemble methods in event detection has received a little attention in the research literature while theoretically it is believed that combining different detectors should provide a better anomalous space coverage [FBAF10, TM04].

There are few works in the literature [Agg13] that adapt ensemble methods for event detection problem. The first work [AJKR10] applies multiple classifiers for anomaly detection from real network traffic data. The authors showed that a few judiciously selected classifiers outperform many diverse classifiers. They propose a method called standard deviation normalized entropy of accuracy as a strategy for combining the classifiers. In another work [FBAF10] authors combine four diverse anomaly detectors for automated event labeling of network traffic data and create a data with ground truth. The strength of their approach relies on the synergy between detectors with different granularity, however, is not specified how data can be considered as ground truth while not being validated by an external knowledge source or human expert. Besides, the role of randomness and chance is not considered in the combination of the outputs of detectors.

In this work, motivated by problem of baseline data creation in Chapter 3 we aim to develop an approach for labeling and detecting of events in unlabeled data by exploiting a combination of both ideas of ensemble learning and background knowledge. This approach has two main applications. Firstly, it can be used for creating benchmark data sets for evaluation of event detection algorithms and secondly can be used in a real world event detection problems when data nature is unknown and there is no access to human experts for the labeling of data or verification of alarms.

Parallel to our main contribution, we perform a comparative study on different important issues in event detection such as learning strategy composed of unsupervised and semi-supervised, scale analysis, multiple denoising approaches, offline and online regression models and distance function in time series similarity estimation.
The rest of the chapter is organized as follows. Next section identifies the main concepts for event detection and introduces the proposed model. Section 5.3 presents a case study using a real data set, discuss the obtained results and presents a sensitivity analysis. The last section concludes the chapter presenting the final remarks.

5.2 Proposed Solution

5.2.1 Definitions

Event labeling is the process of marking events in unlabeled data.

Event Detector is a method or algorithm that discriminates events from non-event instances.

Ensemble Detector is a group of event detection algorithms that assign a score to each instance of data. The score usually represents the chance of that instance not be an event.

Background Knowledge is a sort of knowledge that cannot be used directly in the training phase due to privacy, computational complexity or competitive reasons but can be queried directly or indirectly. Some examples of Background knowledge sources are as follows.

Homogeneous Sources This category may include data archives, data warehouses, domain ontologies and other homogeneous sources. The assumption in this category is that we have computational limitations for dealing with big data sets. However, it is assumed that it is possible to query the higher scale data set through an efficient DBMS gateway.

Heterogeneous Sources Heterogeneous sources differ in nature with train and test sets. The well-known example is the World Wide Web. There is huge heterogeneous information available on the web that cannot be integrated in the learning process because of both volume and competitive issues. But a direct query or query over API is possible over these sources. Our work is concentrated on this type of knowledge sources. We use existing knowledge inside Google™ web and image search and YouTube™ for verification of detected events.
Figure 5.1: Event labeling models. a) Domain expert(s) verify the alarms raised by a single detector. b) Domain expert(s) label the instances by manual inspection. c) Events are labeled by applying an ensemble of detectors. d) Proposed model: extension of previous model with this difference that alarms are verified by background knowledge before labeling.

Confidential Sources Sometimes due to the privacy or security matters is not possible to have access to the whole database. However, the third party provides a secure gateway to perform limited queries over the databases.

5.2.2 Proposed Event Labeling System

There are two classic event labeling models that rely on human-based knowledge. In the first model (Figure 5.1-a) a desired detector is applied to the data and then detected events are verified by one or more domain expert(s) [BDWS09, DFB+07, LCD05, LBC+06]. In this model, checking of all instances is not essential, rather a
limited number of candidates are finally verified by the expert. This model has two main drawbacks: on one hand there is no guarantee that the detector algorithms work well on that particular data set and could detect all potential events and in another hand in the evaluation phase is not possible to measure the accuracy of the detector.

In the second model (Figure 5.1-b), all instances are checked individually by knowledge expert(s) and events are labeled manually [BKPR02]. This model has also three main drawbacks: firstly, it is an infeasible task for large databases to check instances individually. Secondly, the opinion of one expert may not be sufficient and affects the labeling quality. Finally, different experts have different perspectives and therefore is hard to assume that they have the same agreement on the event labels [WMCW05].

A recent automatic model is proposed in [FBAF10] which does not rely on human-based knowledge. As it is depicted in Figure 5.1-c, output of ensemble detectors are combined based on the detectors outputs similarity. The drawback of this model, however is that the result is highly dependent on the detector selection and no knowledge source (human nor machine) validates the output. Therefore, false alarms might be raised, more than expected, due to not considering chance and randomness.

We extend the third model to a new model (Figure 5.1-d) which uses potential knowledge resources for verification of alarms. It also has no dependency on human-based knowledge. Since it is based on ensemble detectors, it is also capable of working in parallel computing framework and thus has potential to be computationally efficient. Based on these explanations we define our research hypotheses as follows.

**Hypothesis 5.1.** Ensemble detectors improve the detection performance comparing individual detectors.

**Hypothesis 5.2.** Background knowledge along with ensemble detectors improves the performance of event detection systems.

In the following we try to examine and validate the above hypotheses through comprehensive experimental study and evaluation tasks.

### 5.3 Experimental Evaluation

There are several public data sets for outlier and anomaly detection. However, it is difficult to find a real data set that for each instance the corresponding environmental data and background knowledge are available. The most challenging part is the
CHAPTER 5. SUBPROBLEM: EVENT LABELING

background knowledge which hardly can be found as an open access source. For this reason many data sets used in the outlier and anomaly detection literature are not such useful for our research goals. We could manage to find a data set that has a potential to be adapted for both above mentioned issues. In the following we first describe this data set and then explore the concepts related to the method.

5.3.1 Data Set

The data set under study is related to a two-year usage log of a bike sharing system namely Capital Bike Sharing (CBS) at Washington D.C., USA. There are three reasons why we think this data set may fit to our research goal. Firstly, it includes at least two of full life-cycle of the system and therefore seems be suitable for supervised and semi-supervised learning. Secondly, there exist some external sources related to historical environmental values such as weather conditions. The weekday and holidays are extractable from other external sources. Finally the alarms are verifiable through open access knowledge sources (search engines, meta-image and meta-video sources).

Bike sharing systems are a new generation of traditional bike rentals where the whole process from membership, rental and return back has become automatic. Through these systems, the user is able to easily rent a bike from a particular position and return it back at another position. Currently, there are over 500 bike-sharing programs around the world which are composed of over 500 thousands bicycles [Pol13]. Today, there exists great interest in these systems due to their important role in traffic, environmental and health issues. Presently, the top three bike-friendly countries are Spain (132 programs), Italy (104 programs) and China (79 programs) [Pol13]. The number of major cities that are becoming bike-friendly is growing day-by-day. It is expected that in a near future, most major cities provide this service along their other public transport services.

Apart from interesting real world applications of bike sharing systems, the characteristics of data being generated by these systems make them attractive for the research. Opposed to other transport services such as bus or subway, the duration of travel, departure and arrival position is explicitly recorded in these systems. This feature turns bike sharing system into a virtual sensor network that can be used for sensing mobility in the city. Hence, it is expected that most of important events in the city could be traceable via monitoring this data. Some few research have already addressed bike sharing data analysis [VGM11, BAF+11] mostly via spatiotemporal analysis to
aid operation-oriented decisions. However, our work differs from such works. In this work our main concentration is not specifically on bike sharing data, rather we use bike sharing data as a supportive source for examining our event labeling model.

In the CBS system when a rental occurs, the operation software collects basic data about the trip such as duration, start date, end date, start station, end station, bike number and member type. The historical data set of such trip transactions is available online via [Cap13]. To avoid trend issues, we select only corresponding data to years 2011 and 2012 consisting of 3,807,587 records. Later, we aggregate the data into two scales: hourly and daily. The hourly time series includes 17,379 hours and the daily time series includes 731 days. Next, we divide both daily and hourly scale time series into two sets of 2011 (train) and 2012 (test). The test set is illustrated in both scales in Figure 5.2 (daily scale) and Figure 5.3 (hourly scale).

As we discuss later, if we apply a regular anomaly detection algorithm on the daily or hourly time series we will not be able to detect all events. We only can detect severe events. Because bike rental process is probably under effect of seasonality and environmental settings such as weekday, holiday, temperature, precipitation, wind speed, humidity, etc. Therefore, event signature cannot be directly observed in these time series. In order to study such effects we need to extract weather data. There exists several weather data sources, however most of them provide only forecasting data

Figure 5.2: The number of rented bikes in 2012 in daily scale
and do not contain historical weather reports. There is another group of forecasting services that contain historical weather records for a specific number of last days (e.g. 14 days). Another group also contains weather historical report but in daily scale. However, we could manage to find a source that provides the hourly historical data [Fre13]. We therefore, extract from this source some attributes such as weather temperature, apparent temperature, wind speed, wind gust, humidity, pressure, dew point and visibility for each hour in the period of 1 January 2011 to 31 December 2012 for Washington D.C., USA. Next, we map each hour in bike rental time series with corresponding weather report. There are some missing weather reports for some hours. Thus, we map the closest report for that hour. The maximum temporal difference is 292 minutes, with mean of 3 minutes and standard deviation of 14 minutes. We also extract the official holidays of Washington D.C. from [Dep13] and map them to the corresponding dates. Afterward, holidays are combined with weekends such that finally each day is classified as working day or non-working day. Additionally, according to weather condition provided in weather data we mark each hour by four weather grades: good, cloudy, bad and very bad.

As a result we create two sets in two scales. In the hourly scale set, each record includes hour, month, working day, season, weather grade, temperature, filling-temperature, humidity and wind speed as variables and hourly aggregated count of rented bikes as

Figure 5.3: The number of rented bikes in 2012 in hourly scale
target value. In the daily scale, each record consists of month, working-day, season, daily average weather-grade, daily average of temperature, daily average of filling-temperature, daily average of humidity and daily average of wind-speed as variables and daily aggregated count of rented bikes as target value.

We then perform a feature selection step on the data set to identify the most significant features. As a result, month, hour, working day and temperature are selected as most important features for hourly scale and month, Working-day and temperature are selected as final features for daily scale. The data set is available online via [FTG13].

5.3.2 The Proposed Method

Our event labeling system is depicted in Figure 5.4. We first apply ensemble detectors (see section 5.3.2.1 for details of detectors) with the highest possible disagreement rate. To assess the degree of disagreement between the detectors, we perform Fleiss’ Kappa test (see section 5.3.2.3). A great disagreement on alarms leads to wider coverage on anomalous space. In addition, more alarms increase the chance of false alarms and thus alarms are required to be validated by an external knowledge source. We run the detectors and combine all alarms to make a candidate list of events. The output list is much limited comparing whole instances in data set and therefore imposes lower cost for verification. Then we combine all outputs together by adding distinctive instances together. In the next step we verify each candidate via Google web and image search and YouTube (we choose Google due to its prominent coverage). The verification
Figure 5.5: A query with format of date+place is submitted to Google web/image search and YouTube for understanding occurred event.

phase work as follows. A spatiotemporal query is submitted to Google (Figure 5.5). If an important event is detected from the result we mark the date with that event. For instance, by querying "2012-10-30 Washington D.C" we notice that the Sandy storm is happened on this date. So this date is marked as "Sandy". If we could not find any significant event from the search result we try Google images and YouTube or try another query this time with including the keyword "weather" (e.g. "2012-10-30 weather Washington D.C"). Note that due to the relatively small volume of our data set we did not perform automatic text processing steps. However, in a fully automatic system, the use of information extraction technique is required.

In the next phase we compute the weight of the event by a method similar to [ZZXY10] where search results count is used as a criterion for extraction of the correlation between two terms (e.g. food+shopping vs. food+drink). The count of search results itself is meaningless, however, it would be considered as an appropriate criterion for comparative purposes. For instance, if query "food+shopping" results in one million and "food+drink" leads to five million pages, it reveals that food is more correlated to drink than shopping. We adapt this idea to measure the weight of candidate events. To this end, as Figure 5.6 shows, we add event title (e.g. "sandy") to the previous query and then extract the count of retrieved results. For instance, as is depicted in the
Figure 5.6: After understanding event, the query with format of date+event+date is submitted to Google to measure the weight of event.

In the figure, 6,920,000 results are returned for this query. This can be used as a criterion to measure the weight of the event. After obtaining this weight for each event candidate, we transform all weights to z-scores. Suppose the vector \( x = (w_1, w_2, ..., w_n) \) be the obtained weights from Google result count. z-score corresponding to each weight is obtained by the following equation.

\[
z - \text{score} = \frac{w - \mu}{\sigma}\tag{5.1}
\]

where \( w \) is the obtained weight, \( \mu \) is the mean and \( \sigma \) is standard deviation of vector \( x \).

Then we remove from the candidate list those events whose z-score is lower than 2. In other words, we keep only those events that there is a low probability of being produced by chance. After this filtering step, the final list contains the event labels.
Figure 5.7: A general architecture of the ensemble detectors. See Table 5.1 for more details

5.3.2.1 Event Detectors

Although at first glance, data looks like a time series, based on our prior knowledge we can argue that it is rare that someone rides a bicycle in some circumstances such as midnight, heavy rains and very cold weathers. Conversely, it is very likely that people rent more bikes in the peak working hours or in a good weather conditions in weekends. In short, it seems that rental count would have a close relationship to environmental settings. To validate this hypothesis, we design our detectors in a way that could support both of these perspectives. In other words, in some detectors we assume that data is a time series (unsupervised detectors) and in other detectors we assume that instances are temporally independent and are correlated to some environmental settings (semi-supervised detectors). However, we give more weight to the latter detectors since they are more reasonable based on our prior knowledge. Note that the term semi-supervised should not be confused with its equivalent in classification or anomaly detection where classes or anomalies labels are specified in the train data. We here deal with the count time series, therefore, when we use the term semi-supervised we refer to a scenario in which we have access to each instance’s environmental settings in the train set and not the class labels.

Ten detectors are designed in this study such that each has its own distinct ability.
Different techniques are involved such as regression trees, control chart, hierarchical clustering [HTF01, p. 520] with two different distance functions of Euclidean and Dynamic Time Warping (DTW) [Sen08]. We also employ Principal Component analysis (PCA) [AW10] and Multi-channel Singular Spectral Analysis (MSSA) [PHHZ11] for filtering in some detectors. Schematic representation of detectors is presented in Figure 5.7. For semi-supervised detectors, we make a predictive model from the train set based on environmental and periodicity setting (To ease the further explanations, from now on, when we refer to environmental setting we mean both environmental and periodicity settings) and then make a forecast on the test set and then compare the predicted bike rental count with the actual bike rental count in test set and then monitor the residuals to detect events. In some detectors we also apply a filter for denoising data. For unsupervised detectors, we monitor test time series irrespective of the environmental settings.

As already mentioned, the data set is made in two scales: hourly and daily. If we perform analysis only on daily scale we would not be able to detect those events that affect the city only in specific hours during the day. Such events are also interesting and need to be detected. For instance, suppose that in 12/05/15 a severe event has happened during 8AM and in the rest of the hours, we witness a normal day. The daily scale analysis probably would not be able to detect such kind of events, because some events manifest themselves in hourly scale.

In order to provide a unit output, alarms in hourly scale are upgraded to daily scale (e.g. detector 10). For instance in the above example, 12/5/15, 8AM is transformed to its corresponding higher scale 12/05/15. In this case all the detectors despite of different scale inputs generate the same output and their outputs can be combined. Each method finally returns the corresponding p-values of each day. This p-value indicates the probability of that day not including an event. So if we determine a threshold like 0.05 then each instant with p-value lower or equal to 0.05 should be reported as an event.

In the following, each individual detector is described in detail. Note that the selected methods for detectors are optional and can be replaced by any other desired methods. However, we take into account two factors in our ensemble detectors architecture (Figure 5.7). The first factor is diversity and the other one is adequacy to automatic settings. It means that detector should be different in terms of used technology and should also be prepared for use in automatic settings.

- **Detector 1**: The predictive model predicts the expected value on the hourly test
set according to the corresponding environmental setting. Then the residuals of both hourly expected value and hourly actual values on test set will be transformed to z-scores. Next, we compute the daily mean of z-scores for each day and again transform the obtained daily means to z-scores and consequently to p-values.

- **Detector 2**: The predictive model predicts the hourly expected value on the hourly test set according to the corresponding environmental setting. Then we compute the mean of hourly residuals for each day. Afterward, the computed daily means are transformed to z-scores and consequently to p-values for each day.

- **Detector 3**: The predictive model makes a forecast for the daily test set according to the corresponding environmental setting. Then the daily residuals are computed as a difference between daily predicted counts and daily actual counts. Then the residuals are transformed to z-scores and consequently p-values.

- **Detector 4**: This method does not need the train data. It operates directly on the daily test set. The count corresponding to each day is transformed to first z-scores and consequently p-values.

- **Detector 5 and 7**: This method operates as follow. First, The predictive model makes a forecast for the hourly test set according to the corresponding environmental setting and then computes the residuals as difference between hourly predicted count and hourly actual count. Next, matrix of $Days \times Hours$ is built such that each cell represents the residuals corresponding that day and hour. In the next step, MSSA (Method 5) or PCA (method 7) is applied on this matrix. The result is a reconstructed matrix. Later, the residual corresponding each day of original and reconstructed matrix is transformed to z-scores and consequently p-values.

- **Detector 6**: This method does not need train data. The hourly test data is converted to matrix of $Days \times Hours$. Afterward, MSSA is applied on this matrix and then residuals corresponding to each day of original and reconstructed matrices are transformed to z-scores and consequently p-values.

- **Detector 8 and 9**: The predictive model predicts the expected value on the hourly test set according to the corresponding environmental setting. Then the residuals of hourly expected value and hourly actual values on test set
are clustered using agglomeration hierarchical clustering algorithm one time with Euclidean distance and one time with Dynamic Time Warping (DTW) distance. Outliers are then chosen using a manual inspection and are reported as events. Note that this kind of approach is not appropriate for automatic detection and is only provided here for comparison to the other approaches.

- **Detector 10**: The predictive model predicts the expected value on the hourly test set according to the corresponding environmental setting. Then the residuals of hourly expected value and hourly actual count on test set are transformed to z-scores. Reported z-scores are still in hourly scale so we select the maximum obtained z-scores for each day. These z-scores correspondent to each day are then transformed to p-values and are reported.

### 5.3.2.2 Detectors Settings

Table 5.1 illustrates the settings used for each detector. All detectors except detector 4 and 6 are semi-supervised. For semi-supervised methods we apply REPTree regression tree as our predictive model (see section 5.3.2.4 for justification). As it can be seen in Table 5.1, even though some detectors receive hourly train set as input, they score events in the daily scale. Three of detectors (5, 6 and 7) also use a filtering strategy such as MSSA and PCA for data denoising. Two of detectors (8 and 9) that are based on agglomerative clustering can only detect events and are not able to score each instant. In detector 8, Euclidean and in detector 9, DTW distance is used. Note that these clustering-based detectors generally are not appropriate choices for automatic settings since need some prior knowledge for parameter setup. However, we include them in ensemble to have more diverse detectors.

### 5.3.2.3 Fleiss’ Kappa Test

The benefit of combining different detectors relies on diversity among detectors ensembles [FBAF10]. Hence, an ideal ensemble detectors is required to include a sort of diverse and different detectors. In order to ensure about the detectors right choice we apply an agreement test on detectors outputs to measure the disagreement rate of the detectors (the more disagreement, the better)

Since we want to evaluate the overall agreement rate between all detectors and not individual agreements between pairs of detectors, we cannot use the common Cohen’s
### Table 5.1: Event Detectors Settings

<table>
<thead>
<tr>
<th>Detector</th>
<th>Type</th>
<th>Predictive Model</th>
<th>Train</th>
<th>Filter</th>
<th>Test</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Semi-supervised</td>
<td>REP Tree</td>
<td>2011,Hour</td>
<td>No</td>
<td>2012,Day</td>
<td>—</td>
</tr>
<tr>
<td>2</td>
<td>Semi-supervised</td>
<td>REP Tree</td>
<td>2011,Hour</td>
<td>No</td>
<td>2012,Day</td>
<td>—</td>
</tr>
<tr>
<td>3</td>
<td>Semi-supervised</td>
<td>REP Tree</td>
<td>2011,Day</td>
<td>No</td>
<td>2012,Day</td>
<td>—</td>
</tr>
<tr>
<td>4</td>
<td>Unsupervised</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>2012,Day</td>
<td>—</td>
</tr>
<tr>
<td>5</td>
<td>Semi-supervised</td>
<td>REP Tree</td>
<td>2011,Hour</td>
<td>MSSA</td>
<td>2012,Day</td>
<td>3 PCs</td>
</tr>
<tr>
<td>6</td>
<td>Unsupervised</td>
<td>—</td>
<td>—</td>
<td>MSSA</td>
<td>2012,Day</td>
<td>3 PCs</td>
</tr>
<tr>
<td>7</td>
<td>Semi-supervised</td>
<td>REP Tree</td>
<td>2011,Hour</td>
<td>PCA</td>
<td>2012,Day</td>
<td>Agglomerative clustering (K=5), Distance—Euclidean</td>
</tr>
<tr>
<td>8</td>
<td>Semi-supervised</td>
<td>REP Tree</td>
<td>2011,Hour</td>
<td>No</td>
<td>2012,Day</td>
<td>Agglomerative clustering (K=5), Distance—DTW</td>
</tr>
<tr>
<td>9</td>
<td>Semi-supervised</td>
<td>REP Tree</td>
<td>2011,Hour</td>
<td>No</td>
<td>2012,Day</td>
<td>—</td>
</tr>
<tr>
<td>10</td>
<td>Semi-supervised</td>
<td>REP Tree</td>
<td>2011,Hour</td>
<td>No</td>
<td>2012,Day</td>
<td>—</td>
</tr>
</tbody>
</table>

kappa [Car96]. Instead, we use Fleiss’s kappa [FO71] which is a statistical tool that measures the level of agreement between multiple raters when assigning categorical ratings to a number of items or classifying items. It is considered as an extension of Cohen’s kappa statistic that works for multiple raters. If a fixed number of raters assign numerical ratings to a number of items then the kappa reveals the consistency of ratings. Table 5.2 shows how $K$ values can be interpreted [LK77].

So far, Fleiss’s kappa has been used in psychology and bio-informatics for measurement of agreement of different human agents on a subject. Here we use it for measuring the rate of agreement between multiple event detectors. Opposed to the psychology and bio-informatics that a $k$ closer to 1 is more desired, we want a closer value to zero. Because we look for a group of non-similar detectors that could detect a wider range of events. If all detectors for instance agree and have $k$ equal to 1 then it means that use of multiple detectors is meaningless and one detector is enough. In contrast, if $k$ could approach zero it means that the idea of using ensemble detector is logical and results in better coverage of discovery of unknown events.

After the experiment we obtain Fleiss’s kappa equal to 0.0034. That means that the ensemble detectors exhibit a very slight agreement. In other words, if we define H0 hypothesis as the observed agreement is accidental we cannot reject the hypothesis due
Table 5.2: Fleiss’es Kappa Interpretation

<table>
<thead>
<tr>
<th>K Range</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K &lt; 0$</td>
<td>Poor agreement</td>
</tr>
<tr>
<td>$0 \leq K \leq 0.2$</td>
<td>Slight agreement</td>
</tr>
<tr>
<td>$0.2 \leq K \leq 0.4$</td>
<td>Fair agreement</td>
</tr>
<tr>
<td>$0.4 \leq K \leq 0.6$</td>
<td>Moderate agreement</td>
</tr>
<tr>
<td>$0.6 \leq K \leq 0.8$</td>
<td>Substantial agreement</td>
</tr>
<tr>
<td>$0.8 \leq K \leq 1$</td>
<td>Perfect agreement</td>
</tr>
</tbody>
</table>

to the observed low agreement. This is the desired result since it reveals that selected detectors exhibit high degree of diversity.

5.3.2.4 Predictive Model Selection

One of the main components of the introduced semi-supervised detectors is the predictive model. We apply some different regression and classifier algorithms in Weka [IHWW11] on the train data to measure the accuracy of the built model. Table 5.3 illustrates the comparison of the models in terms of correlation coefficient, relative absolute error (RAE), root relative squared error (RRSE), train time in seconds and test time in seconds. Train time is the time that takes for the model to be built on the train set and test time is the time it takes for the model to be evaluated through 10-folds cross validation. As it can be seen, among all, REPTree [IHWW11] has a better performance in terms of the trade-off between accuracy, train and test time. It is from the regression tree family and thus presents interpretable model. IBk, and decision table both provide relatively the same accuracy but with higher test time. Therefore, we select REPTree as the predictive model in the detectors. RRSE, RAE and correlation coefficient in Table 5.3 also can be calculated using the following equations [IHWW11, p. 180].

$$RAE = \frac{|p_1 - a_1| + \ldots + |p_n - a_n|}{|\bar{a} - a_1| + \ldots + |\bar{a} - a_n|}$$  \hspace{1cm} (5.2)

$$RRSE = \frac{(p_1 - a_1)^2 + \ldots + (p_n - a_n)^2}{(\bar{a} - a_1)^2 + \ldots + (\bar{a} - a_n)^2}$$  \hspace{1cm} (5.3)

$$Correlation \ Coefficient = \frac{S_{PA}}{\sqrt{S_P S_A}}$$  \hspace{1cm} (5.4)
Table 5.3: Predictive Models Tested

<table>
<thead>
<tr>
<th>Model</th>
<th>Correlation (%)</th>
<th>RAE(%)</th>
<th>RRSE(%)</th>
<th>Train Time(s)</th>
<th>Test Time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>REPTree</td>
<td>91.57</td>
<td>30.56</td>
<td>40.24</td>
<td>0.04</td>
<td>2.20</td>
</tr>
<tr>
<td>Linear Regression</td>
<td>81.12</td>
<td>54.67</td>
<td>58.47</td>
<td>0.69</td>
<td>7.08</td>
</tr>
<tr>
<td>RBF Network</td>
<td>22.77</td>
<td>96.65</td>
<td>97.36</td>
<td>0.25</td>
<td>2.08</td>
</tr>
<tr>
<td>IBk</td>
<td>91.99</td>
<td>29.98</td>
<td>39.27</td>
<td>0.01</td>
<td>7.03</td>
</tr>
<tr>
<td>LWL</td>
<td>65.54</td>
<td>76.47</td>
<td>76.25</td>
<td>0.01</td>
<td>60.00</td>
</tr>
<tr>
<td>Additive Regression</td>
<td>71.57</td>
<td>67.54</td>
<td>69.83</td>
<td>0.10</td>
<td>2.01</td>
</tr>
<tr>
<td>Random SubSpace</td>
<td>84.74</td>
<td>59.47</td>
<td>62.78</td>
<td>0.20</td>
<td>3.10</td>
</tr>
<tr>
<td>RegressionByDisc</td>
<td>90.91</td>
<td>34.02</td>
<td>41.65</td>
<td>0.09</td>
<td>2.00</td>
</tr>
<tr>
<td>Conjunctive Rule</td>
<td>35.59</td>
<td>92.70</td>
<td>93.43</td>
<td>0.05</td>
<td>1.70</td>
</tr>
<tr>
<td>Decision Table</td>
<td>91.69</td>
<td>30.11</td>
<td>39.91</td>
<td>5.95</td>
<td>70.70</td>
</tr>
<tr>
<td>Decision Stump</td>
<td>35.63</td>
<td>92.59</td>
<td>93.42</td>
<td>0.02</td>
<td>2.20</td>
</tr>
<tr>
<td>FIMTDD</td>
<td>68.29</td>
<td>71.89</td>
<td>71.49</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Where:

\[ S_{PA} = \frac{\sum(p_i - \bar{p})(a_i - \bar{a})}{n - 1}, \]  
\[ S_P = \frac{\sum(p_i - \bar{p})^2}{n - 1}, \]  
\[ S_A = \frac{\sum(a_i - \bar{a})^2}{n - 1} \]

In the above equations, \( a \) denotes actual target values, \( p \) denotes predicted target values, \( \bar{a} \) represents the average of actual target value, \( \bar{p} \) denotes the average of predicted target values and \( n \) denotes the sample size.

5.3.3 Results

5.3.3.1 Event Labels

Table 5.4 demonstrates detected events by our system that have passed verification phase by background knowledge. Bold items are those events that meet the condition of \( z-score \geq 2 \). The primary candidates list before verification phase contains 69 events and as it can be seen this number is decreased to 30 events. To validate this result we ask a domain specialist to rate the impact of each detected event corresponding date from 0 to 5. The third column in Table 5.4 indicates the impact rates specified by a specialist for that date. To evaluate the effectiveness of condition
Table 5.4: Detected Events after verification phase by background knowledge. Bold items are verified detected events (Events with \( z - score \geq 2 \)) and non-bold items are those events whose \( z - score < 2 \). The numbers in the third column represent the impact rate (from 0 to 5) given by a human domain specialist for that date indicating the impact of event.

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
<th>Impact</th>
</tr>
</thead>
<tbody>
<tr>
<td>29-10-2012</td>
<td>Sandy</td>
<td>5</td>
</tr>
<tr>
<td>30-10-2012</td>
<td>Sandy</td>
<td>5</td>
</tr>
<tr>
<td>19-10-2012</td>
<td>Storm</td>
<td>5</td>
</tr>
<tr>
<td>04-07-2012</td>
<td>Washington DC Fireworks</td>
<td>5</td>
</tr>
<tr>
<td>23-11-2012</td>
<td>Black Friday</td>
<td>4</td>
</tr>
<tr>
<td>24-12-2012</td>
<td>Christmas day</td>
<td>4</td>
</tr>
<tr>
<td>08-10-2012</td>
<td>Columbus Memorial Celebration</td>
<td>4</td>
</tr>
<tr>
<td>27-05-2012</td>
<td>Memorial Day</td>
<td>4</td>
</tr>
<tr>
<td>22-11-2012</td>
<td>Annual Thanksgiving Day</td>
<td>3</td>
</tr>
<tr>
<td>12-11-2012</td>
<td>Veterans Day</td>
<td>2</td>
</tr>
<tr>
<td>16-04-2012</td>
<td>Tax Day</td>
<td>1</td>
</tr>
<tr>
<td>23-03-2012</td>
<td>National Cherry Blossom Festival</td>
<td>5</td>
</tr>
<tr>
<td>18-09-2012</td>
<td>Heavy Rain</td>
<td>5</td>
</tr>
<tr>
<td>18-07-2012</td>
<td>Severe Thunderstorm</td>
<td>5</td>
</tr>
<tr>
<td>01-06-2012</td>
<td>Tornado</td>
<td>4</td>
</tr>
<tr>
<td>04-12-2012</td>
<td>Warm weather floods</td>
<td>4</td>
</tr>
<tr>
<td>13-05-2012</td>
<td>Bike DC</td>
<td>3</td>
</tr>
<tr>
<td>11-02-2012</td>
<td>Cupid Undie Run 2012</td>
<td>3</td>
</tr>
<tr>
<td>23-01-2012</td>
<td>March for life</td>
<td>3</td>
</tr>
<tr>
<td>29-09-2012</td>
<td>Green Festival Washington DC</td>
<td>3</td>
</tr>
<tr>
<td>25-11-2012</td>
<td>The coldest morning of the season</td>
<td>2</td>
</tr>
<tr>
<td>07-10-2012</td>
<td>Unseasonably cool weather</td>
<td>2</td>
</tr>
<tr>
<td>07-04-2012</td>
<td>D.C. United vs. Seattle Sounders FC</td>
<td>2</td>
</tr>
<tr>
<td>26-05-2012</td>
<td>D.C. United vs. NE Revolution</td>
<td>2</td>
</tr>
<tr>
<td>21-05-2012</td>
<td>Occasional showers and storms</td>
<td>2</td>
</tr>
<tr>
<td>15-09-2012</td>
<td>United vs. NE Revolution</td>
<td>2</td>
</tr>
<tr>
<td>11-10-2012</td>
<td>D.C. Baseball v.s Tigers</td>
<td>2</td>
</tr>
<tr>
<td>12-10-2012</td>
<td>Hockey Capitals vs. NJ Devils</td>
<td>2</td>
</tr>
<tr>
<td>29-01-2012</td>
<td>Occupy DC</td>
<td>1</td>
</tr>
<tr>
<td>19-05-2012</td>
<td>SurviveDC 2012</td>
<td>1</td>
</tr>
</tbody>
</table>
$z - score \geq 2$, we define a cut line on impact rates given by a specialist and then see how detected events match with the events over the cut line. Since the given impact rates are between 1 and 5, we define four cut point of 2,3,4 and 5 and compute the true and false alarm rate one time for all items of Table 5.4 and another time for bold items. The result is shown in Figure 5.8. As it can be seen, applying condition of $z - score \geq 2$ can decrease the false alarm rate up to an average of 20%.

Although, domain specialist implicitly confirmed the accuracy of the detected events, looking to the Table 5.4 we notice that five alarms with impact rates of 5 and 4 that are specified by the specialist do not appear in the verified alarms list. This can be due to some existing problems in our methodology details. For instance, probably due to the naming complexity of the events, our submitted query has not been well enough for measuring the weight of weather-related events (four of the cases). It indicates that the query of 'time-place-weather' might not be a good idea and we should find a more appropriate alternative query. The reason can be that the weather events on these days are entitled with different terms and the weight corresponding to these events is distributed in different terms. For instance, some sources might call "Tornado" with different terms such as high speed wind, storm or severe weather, etc. However, as it can be seen, only one non-weather event is missed in our final event labels. It reveals that the condition of $z - score \geq 2$ is reasonably able to filter non-significant alarms and consequently avoid false alarms more effectively.

### 5.3.3.2 Event Labeling in the absence of Background knowledge

Here we study an event labeling model which relies only on ensemble detectors and have no access to any external knowledge (Figure 5.1-c). To compare this model with our proposed model (Figure 5.1-d), we need to compare the outputs of both models. If we assume model d and its result as reference we can formulate the problem as an information retrieval problem. We run model c and then measure the similarity of retrieved events with the reference detected events. In other words, we want to know how we can reproduce the same result as model d by using model c. We consider our model as reference model because its output is already checked with a human domain expert.

With this end in mind, we use a voting strategy to combine ensemble detectors alarms. We first define a confidence threshold equal to $\alpha = 0.05$ and compute the total votes of detectors corresponding each day. For detector 8 and 9 that are based on clustering and do not return p-value we assume that the detectors are confident enough ($p-value \leq$
0.05) and thus include them in the voting process.

We count the votes of detectors for each instant and then compute the similarities of detected list with Table 5.4. The results are presented in Table 5.5. N in this table denotes the detector votes. $N > 1$ for instance indicates that at least two detectors agree that a hypothetical instant should be recognized as event. The terms $N > 2$, $N > 3$ and $N > 4$ mean that at least three, four or five detectors are respectively suspicious to a particular instant. As it can be seen, event signals corresponding to $N > 1$ are 72% similar to events marked with bold in Table 5.4. It means that if we have the vote of at least two detectors for an instant, the alarms would be over 70% similar to the final output of our proposed model. However, by increasing the required vote, F-measure decreases to 0.58 for $N > 2$ and 0.23 for $N > 3$ and $N > 4$.

We repeat the same procedure for individual detectors to compare their individual performance. Comparing F-measures of the ensemble detectors in Table 5.5 with individual detectors in Table 5.6 reveals the effectiveness of ensemble detectors comparing the individual detectors. As it can be seen the maximum F-measure obtained for individual detector is related to detector 1 which is equal to 0.55. This is almost 20% lower than ensemble detectors with $N > 1$ condition.

Figure 5.8: Effect of condition $z - score \geq 2$ on false alarm rate.
Table 5.5: Ensemble Detectors Retrieval Performance in the absence of background knowledge. Reference events: bold items in Table 5.4

<table>
<thead>
<tr>
<th>Votes</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N &gt; 1$</td>
<td>0.81</td>
<td>0.64</td>
<td>0.72</td>
</tr>
<tr>
<td>$N &gt; 2$</td>
<td>0.63</td>
<td>0.53</td>
<td>0.58</td>
</tr>
<tr>
<td>$N &gt; 3$</td>
<td>0.27</td>
<td>0.20</td>
<td>0.23</td>
</tr>
<tr>
<td>$N &gt; 4$</td>
<td>0.27</td>
<td>0.20</td>
<td>0.23</td>
</tr>
</tbody>
</table>

Table 5.6: Individual Detectors Retrieval Performance in the absence of background knowledge. Reference events: bold items in Table 5.4

<table>
<thead>
<tr>
<th>Detector</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.60</td>
<td>0.50</td>
<td>0.55</td>
</tr>
<tr>
<td>2</td>
<td>0.25</td>
<td>0.22</td>
<td>0.23</td>
</tr>
<tr>
<td>3</td>
<td>0.25</td>
<td>0.28</td>
<td>0.26</td>
</tr>
<tr>
<td>4</td>
<td>0.40</td>
<td>0.11</td>
<td>0.17</td>
</tr>
<tr>
<td>5</td>
<td>0.31</td>
<td>0.28</td>
<td>0.29</td>
</tr>
<tr>
<td>6</td>
<td>0.43</td>
<td>0.17</td>
<td>0.24</td>
</tr>
<tr>
<td>7</td>
<td>0.17</td>
<td>0.11</td>
<td>0.13</td>
</tr>
<tr>
<td>8</td>
<td>1.00</td>
<td>0.22</td>
<td>0.36</td>
</tr>
<tr>
<td>9</td>
<td>0.75</td>
<td>0.17</td>
<td>0.28</td>
</tr>
<tr>
<td>10</td>
<td>0.33</td>
<td>0.33</td>
<td>0.33</td>
</tr>
</tbody>
</table>
Table 5.7: Area under ROC Curves for each individual detectors

<table>
<thead>
<tr>
<th>Detector</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.74</td>
</tr>
<tr>
<td>2</td>
<td>0.70</td>
</tr>
<tr>
<td>3</td>
<td>0.60</td>
</tr>
<tr>
<td>4</td>
<td>0.47</td>
</tr>
<tr>
<td>5</td>
<td>0.70</td>
</tr>
<tr>
<td>6</td>
<td>0.39</td>
</tr>
<tr>
<td>7</td>
<td>0.75</td>
</tr>
<tr>
<td>8</td>
<td>N/A</td>
</tr>
<tr>
<td>9</td>
<td>N/A</td>
</tr>
<tr>
<td>10</td>
<td>0.76</td>
</tr>
</tbody>
</table>

5.3.3.3 Evaluation of individual detectors with AUC

In order to compare the accuracy of individual detectors we compute the area under the obtained ROC curves (AUC). The result is presented in Table 5.7. As it can be seen, detector 10 provides the best accuracy and detectors 7, 1, 2 and 5 are ranked in the subsequent places, which all provides over 70% accuracy. The values in this table do not indicate the strength of the detectors in general; it rather reveals their specific performance on this particular condition. In other words, if we would not consider ensemble detectors and we did not have access to background knowledge, detector 10 could reproduce the same result as our system with accuracy of 75%. There is no guarantee that this detector performs the same on other circumstances.

5.3.4 Discussion

Here we present our comparative study results. Note that the following results are domain specific and may not be generalized to other different problems and settings. Due to the difficulty and limitations on obtaining data with required characteristics we were not able to perform experiments on different domains and applications. Hence, the following specific findings emphasis on bike sharing data and the similar regression problems and domains and are required to be re-validated for other domains.
5.3.4.1 Learning: Semi-supervised vs. Unsupervised

Table 5.7 shows the AUC of the individual detectors. As it can be seen, semi-supervised detectors outperform unsupervised detectors. AUC for unsupervised detectors (detector 4 and 6) is below 0.6. This shows that bike sharing data is highly seasonal and and instants are highly dependent on the environmental settings. Unsupervised detectors are able to detect only severe events while semi-supervised detectors are the ones that are more appropriate for detection of meaningful events.

5.3.4.2 Scale Analysis: Hour vs. Day

We designed the detectors such a way that they are able to compare the performance of analysis on daily scale vs. hourly scale. Among semi-supervised detectors, detector 3 is the only approach that operates on daily scale. That means it makes a model from train set in daily aggregated counts (Figure 5.2) and then make a forecast on test set in daily scale. Other approaches make a model from train set in the hourly scale. AUC of detector 3 comparing to the other methods is presented in Table 5.7. As it can be seen, the low AUC value for this detector reveals that for detecting events in daily scale it is not always a better idea to make a predictive model on the same scale, rather sometimes is better to make a model on smaller scales. The related detector that operates on hourly scale (i.e. detector 2) has a at least 10% better accuracy comparing detector 3. This provides evidence that for event detection in a desired scale, training on smaller scale would also be worth of consideration.

5.3.4.3 PCA vs. MSSA

The idea of MSSA is to adapt PCA for time series. PCA looks to the instances independently while MSSA takes into account the auto-correlation between temporal instances together. Capturing this auto-correlation is not considered in PCA. We compared the performance of PCA (detector 5) vs. MSSA (detector 7). We considered only three Principal components for both and did not check other settings. They might have a different performance in other settings. However, in the same condition as Table 5.7 shows, PCA outperforms MSSA. The most likely reason is that the bike sharing data is non-sequential due to its seasonal effects and it has a poor correlation between consecutive instances. If there was a strong auto-correlation then MSSA would perform better. The point is that PCA and MSSA are required to be chosen
depending on the data nature. PCA is recommended for independent instances and MSSA for auto-correlated instances.

5.3.4.4 Predictive model: Online vs. Offline

In Table 5.3 we compared the performance of 12 different algorithms. The last algorithm in this table is related to FIMT-DD algorithm [IGD11], which is an online regression tree model. This is a streaming algorithm that scans the training data only once, using little computational resources (memory and CPU). As it can be seen, this model includes less accuracy comparing to the REPTree, however if the computation complexity is the problem, can present a relatively reasonable performance. The predicted counts correlation to the original counts is 68.29% vs. REPTree that presents 91.57% correlation. This difference seems reasonable for large data sets or streaming settings where REPTree fails.

5.3.4.5 Distance: Euclidean vs. DTW

A comparison of the performance obtained using Euclidean (detector 8) and DTW (detector 9) distances is presented in Table 5.7. Although it is not a significant difference, DTW exhibits worse results. This is an evidence that environmental attributes play an important role in bike sharing process. Because, if data did not contain seasonality, using DTW distance had to give a better result.

5.4 Conclusion

We proposed a novel event labeling model based on ensemble learning and background knowledge. We provided some evidences about the effectiveness of the proposed model through a set of tasks on a real-world data set.

Our research findings can be summarized as follows: 1) When there is no access to human experts, background knowledge (if available) can be an appropriate alternative; 2) Scale of the train and test data sets should not necessarily be the same for the learning. 3) Regression tree model REPTree is promising on data sets like bike sharing. We believe that this model will probably work well on data sets with same nature dealing with count time series under effect of environmental and periodicity settings 4) MSSA and DTW are reconsigned as robust tools in time series analysis. However
as we demonstrated they can act inverse when data is under seasonal effects. 5) On
the absence of background knowledge, ensembles detectors can present a result 70%
similar to the condition where we have access to background knowledge for verification;
6) We offered evidence that ensemble detectors with at least two votes provide 20%
better result than the best individual detector; 7) We showed that bike rental data
is highly correlated with environmental and periodicity settings such as temperature
and hour of the day, month, work of the day (weekend, weekday, and holiday). A
regression tree model can make a prediction based on these environmental attributes
very close to actual counts. This shows that bike rental count time series should not
be analyzed without taking into account the environmental attributes; 8) Web and
existing knowledge in that can be potential source for aiding event detection systems.

Event detection on bike sharing data also has two potential applications. First, it can
be incorporated in a decision support system for a better planning and management
of system and secondly, can be employed in a recommender system for alarming or
suggestion purposes. For instance, suggesting people not going out due to severe
weather conditions or encourage them to go out for participating in an ongoing event
in the town.

Further research will include the following directions: 1) Verification of the proposed
model performance on other data sets and with other knowledge sources apart from
online sources; 2) Testing different ensemble designs; 3) Studying different combination
techniques in ensemble detectors; 4) Spatiotemporal analysis on the data to discover
localized events; 5) Real time detection; 6) Development of some text processing
methods for automated capturing of knowledge from the Web.
Chapter 6

Retrospective/Unsupervised Method

Traffic tensor or simply $\text{origin} \times \text{destination} \times \text{time}$ is a new data model for conventional origin/destination (O/D) matrices. This new data model is used in emerging traffic data analysis techniques called tensor models. The key superiority of tensors over other models is that both temporal and spatial fluctuations of traffic patterns are taken into account simultaneously, resulting in more natural pattern discovery. Three major types of fluctuations can occur in traffic tensors: mutations in the overall traffic flows, alterations in the network topology and chaotic behaviors. How can we detect events in a system that faces with all types of such fluctuations in its life cycle? Our initial studies reveal that the current form of tensor models have some difficulties in dealing with such a realistic scenario. We propose a new retrospective/unsupervised method that enhances the detection ability of tensor models by parallel tracking of traffic topology. However, tensor decomposition techniques such as Tucker, as the heart of tensor models require a complicated parameter that not only is difficult to choose but also affects the model quality. We address this problem by examination of a recent technique called adjustable core size Tucker decomposition (ACS-Tucker) that promises gains relative to the current situation. Experiments on simulated and real-world data sets from different domains versus several techniques indicate that the proposed model is effective and robust and can be a potential alternative for analysis of traffic tensors.
Figure 6.1: Motivational example: a simplified hypothetical scenario in bike-sharing network. Between t=1 to 94 the system has a stable behavior. At t=95 to 96 a mutation occurs thorough whole network. In t=97 and 98 topology of traffic changes in a part of the network. During t=99 to 100 a chaotic behavior shows up in a part of the network. How can we accurately detect these events via a unified model?

6.1 Introduction

Understanding and characterizing 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 make better decisions about emerging problems and perform some prevention tasks. Intuitive examples are identification of attacks and malicious activities in Internet networks and traffic planning in transportation systems. However, one of the serious problems in event detection from traffic tensors is the complexity and diversity of event types. Fig. 6.1 illustrates a simplified scenario of a hypothetical bike sharing system during the operation period of 100 days. The stations are specified by letters A to F and the connections between them are shown with directed lines. For more simplicity let us suppose that system has 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 only between days 95 to 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 quantity 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 that affects the whole population of the
city. In this example, \( t=95 \) relates to a severe event and \( t=96 \) to a low-scale event. Note that alteration in traffic volume is not necessarily additive. The weather-related events such as rain lead to the same patterns, but in subtractive form. For instance, in a normal working day, a heavy rain may remarkably reduce the requests for bike rental.

The next event type occurs during days 97 and 98 where some new connections are established between the stations in a part of the network. As we see, traffic between main stations remains unchanged as the system’s normal behavior, 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, imagine a scenario in which stations E and F are out of service for a long period and suddenly become available. Or in the other possible scenario users may refer to the next closest stations E and F, if do not find available bikes in the station B. Some local events such as a sports rivalry also can be the reason of this event type. For instance, 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 a way the stadium may establish a new connection from a station close to his neighborhood to the stations close the stadium’s location.

Finally, in 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, at \( t=99 \) although the traffic pattern seems similar to the stable condition, two links A to D and C to D exhibit an odd behavior. In one of them we see a very slight change while in another one we observe a very intensive fluctuation. Likewise, in \( t=100 \) we see 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 mostly involved. An intuitive example of such a chaotic behavior can be an occurrence of various events at the same time such as a football rivalry, along with a blackout event in some stations plus a weather-based event like rain.

As we can see in the above cases, some events are associated with fluctuations in flow (e.g. days 95 and 96), while some are linked to alterations in the network topology (e.g. days 97 and 98) and some others, such as 99 and 100 to a chaotic behavior in a part of the network. These kinds of patterns may be repeated several times and are not necessarily limited to a specific type. Practically, most of these patterns take place in the system during its life cycle. The question is how we can construct a model that simultaneously covers 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 well studied area in network information systems and transportation systems and many methods are developed in these two domains. The techniques, as is illustrated in Fig. 6.2 depending on the data structure and analysis component are classified into four major models.

![Diagram of four major traffic data analysis techniques](image)

**Figure 6.2:** Four major traffic data analysis techniques. Numbers in the figure are derived from the example scenario in Fig. 6.1.

Vector models are the most basic solutions for analysis of traffic data. In this kind of approaches 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) [VDVDW96] or a regression model [ZLS04]. The application of these techniques is limited, because they do not take into account the correlation among links, hence have difficulties in handling noises and missing values in data.

Matrix models instead attempt to model all links simultaneously and, therefore do not have the problem of vector models. With these methods [LPC+04, WHX+12], all links together form a matrix whose rows are links and its columns are time instants (see Fig. 6.2). The next step is to apply a matrix decomposition solution such as Principal Component Analysis (PCA) [Hot33] on the link matrix. PCA is able to explain 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 relaxes the
natural three-dimensional structure of data to a two-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) [GR70]. In [IK04] authors propose a new approach based on SVD that tracks the angle of dominant left singular vector along with principal eigenvalue over time for detection of faults in a simulated web traffic. Although SVD-based methods seem potential for anomaly detection, opposed to PCA are not applicable for modeling and forecasting.

There exists a new branch of matrix-based methods which we call here matrix residual models to differentiate with 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 [SXZF07] a variant of PCA called Compact Matrix Decomposition (CMD) is applied to traffic flow matrices. Then a time series of reconstruction errors is created and tracked for anomaly detection. Similarly, in [STP+08b] authors apply a strategy for anomaly detection from evolving Internet network. These methods, even though they may be 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 some part of temporal information be lost which is essential for event detection. Additionally, they are vulnerable to seasonal effects [STP+08b] and hence are not ideal for modeling human-generated data sets [TR13].

There is another category of methods called graph-based techniques that are developed in social network analysis community and are less relevant for the present study. Because, we are interested in global events rather than anomalous nodes, edges or communities. However, interested readers are referred to recent survey paper [ATK14] covering the majority of advances in this area.

The more sophisticated tools for traffic data modeling are tensor models which do not include the majority of above-mentioned limitations. The need for tensors in traffic data modeling has emerged in recent years in two research communities, including data mining [ADKM11, SJMG12, STF06] for network anomaly detection and transportation systems [TFF+13a, TWF+13, TYF+13, WGC+14, TFC+14] for traffic estimation and imputation.

Tensor decomposition, a central technique used in these models is a powerful tool for analysis of multiway data with many applications in psychology, chemistry, neuroscience, signal processing, bioinformatics, computer vision and data mining [Mor11].
Tensor solutions opposed to other techniques are able to model spatio-temporal fluctuations in traffic data, therefore, are capable to generate a more natural model from data and consequently discover more realistic patterns. Despite of 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, contrary to other kind of tensors, we deal with some complex fluctuations that are required to be carefully considered. Our preliminary analysis indicates that when there is a high variance in quantity of traffic, tensor models become less sensitive to topological fluctuations. Hence, the current form of naïve tensor models, yet may not 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 complicated input parameter which is difficult to choose for the majority of users. This leads to incorrect parameter selection risk [KLR04] 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) [CLZ14] that takes into account core size determination as a part of the decision in the decomposition procedure.

To the best of our knowledge, this is the first work that addresses the problem of topology modeling in traffic tensors. 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. In section 6.2 we describe our hybrid tensor model and its components. Section 6.3 defines the experimental settings. In sections 6.4 and 6.5 we respectively present the result of our simulation and real case studies. We discuss the scalability issues in section 6.6. Finally, the last section concludes the exposition, presenting the final remarks.

6.2 Hybrid tensor model (HTM)

Fig. 6.3 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 to 3 in the figure. In these steps, as opposed to existing approaches that rely on only traffic flow tensor, in parallel, we construct a new tensor called topology which is the Boolean copy of original tensor.
The second basis of the proposed solution concentrates on Tucker optimal core size estimation problem. Like data clustering, where the number of clusters as 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 quality.

Finally, we address the event detection task (numbers 5 to 7 in the figure) by combining both topology and flow models using multivariate statistics and tracking the system behavior in the hybrid subspace.

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

Figure 6.3: An illustrative example of 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 Hoteling’s $T^2$ statistics to $\chi^2$ distribution with 5 degree of freedom is computed, p-value is reported as 1-CDF and those instants with p-value lower than $\alpha = 0.01$ are marked as significant event.
6.2.1 Data model

The original target data set is represented as a graph $G = (V(G), E(G))$ where $V(G)$ are nodes and $E(G)$ are a set of edges. $V(G)$ can be countries, cities or stations and the $E(G)$ indicates the flow volume between the nodes (e.g. number of travels or trade volume). In order to analyze this data in the tensor scheme, we first need to transform it to the adjacency matrix. The $G$ is transformed into a flow adjacency matrix $\text{OD}(N \times M)$. The entries of the matrix take values from 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 $\text{OD}_1, \text{OD}_2, \ldots, \text{OD}_T$ are combined, to generate a tensor $\mathcal{X} \in \mathbb{R}^{N \times M \times T}$. This tensor is called traffic tensor or flow tensor and alternatively naive tensor. Thereafter, A boolean copy of $\mathcal{X}$ is generated, replacing all non-zero elements with 1 and naming it topology tensor.

We keep the topology tensor in parallel to give importance to those events that are associated with structural fluctuations in the 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 lean to the flow tensor. In other words, topology tensor opposed to the original traffic tensor, is more sensitive to topological changes as well as non-sensitive to severe variation in the flow rates.

6.2.2 ACS-Tucker decomposition of flow/topology tensors

We here describe the method we use for decomposition of flow and topology tensors.

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 $A_1 \in \mathbb{R}^{N \times r_1}$, $A_2 \in \mathbb{R}^{M \times r_2}$ and $A_3 \in \mathbb{R}^{T \times r_3}$ such that $\mathcal{X} \approx \mathcal{G} \times_1 A_1 \times_2 A_2 \times_3 A_3$.

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 in nature is not orthogonal constrained as PARAFAC assumes.

In [CLZ14] a new method is proposed called adjustable core size Tucker decomposition (ACS-Tucker) that performs Tucker decomposition with unspecified size of the core
through maximum block improvement [CHLZ12]. The authors apply the method on known-rank tensors (both simulated and real) and show that ACS-Tucker is remarkably accurate in the estimation of correct Tucker core size.

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 the \( r _ { i } \) from being too large, the summation of \( r _ { i } \) is assumed to be equal to \( c \), i.e. \( r _ { 1 } + r _ { 2 } + r _ { 3 } = c \). For the three-dimensional tensor \( \mathcal{X} \), Tucker decomposition optimization is as follows.

\[
\min ||\mathcal{X} - \mathcal{G} \times _ { 1 } \mathbf{A} ^ { ( 1 ) } \times _ { 2 } \mathbf{A} ^ { ( 2 ) } \times _ { 3 } \mathbf{A} ^ { ( 3 ) } ||
\]

(6.1)

Subject to:
\( A ^ { ( i ) } \in \mathbb{R} ^ { m _ { i } \times r _ { i } } , ( A ^ { ( i ) } ) ^ { T } A ^ { ( i ) } = I , i = 1 , 2 , 3 \).

The above minimization function is equivalent to the following maximization problem [CLZ14, KB09b]

\[
\max ||\mathcal{X} \times _ { 1 } ( A ^ { ( 1 ) } ) ^ { T } \times _ { 2 } ( A ^ { ( 2 ) } ) ^ { T } \times _ { 3 } ( A ^ { ( 3 ) } ) ^ { T } ||
\]

(6.2)

Subject to:
\( A ^ { ( i ) } \in \mathbb{R} ^ { m _ { i } \times r _ { i } } , ( A ^ { ( i ) } ) ^ { T } A ^ { ( i ) } = I , i = 1 , 2 , 3 \).

We need \( r _ { i } \) to be determined. 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 \( \mathbf{Y} ^ { ( i ) } = \text{diag}(y ^ { ( i ) } ) , y ^ { ( i ) } \in \{ 0 , 1 \} ^ { m _ { i } } , \sum _ { j = 1 } ^ { m _ { i } } y _ { j } ^ { ( i ) } = r _ { i } \). If we replace the term \( ( A ^ { ( i ) } ) ^ { T } \) with \( \mathbf{A} ^ { i } \) in (6.2) we have:

\[
\max ||\mathcal{X} \times _ { 1 } ( A ^ { ( 1 ) } \mathbf{Y} ^ { ( 1 ) } ) ^ { T } \times _ { 2 } ( A ^ { ( 2 ) } \mathbf{Y} ^ { ( 2 ) } ) ^ { T } \times _ { 3 } ( A ^ { ( 3 ) } \mathbf{Y} ^ { ( 3 ) } ) ^ { T } ||
\]

(6.3)

Subject to:
$A^{(i)} \in \mathbb{R}^{n_i \times r_i}, (A^{(i)})^T A^{(i)} = I, y^{(i)} \in \{0, 1\}^{m_i}, \sum_{j=1}^{m_i} y^{(i)}_j \geq 1, \sum_{i=1}^{3} \sum_{j=1}^{m_i} y^{(i)}_j = c, i = 1, 2, 3.$

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

$$\text{Max } \left\| X \times_1 (A^{(1)}Y^{(1)})^T \times_2 (A^{(2)}Y^{(2)})^T \times_3 (A^{(3)}Y^{(3)})^T \right\|^2 - \lambda \left( \sum_{i=1}^{3} \sum_{j=1}^{m_i} y^{(i)}_j - c \right)^2$$

(6.4)

Subject to:

$$A^{(i)} \in \mathbb{R}^{n_i \times r_i}, (A^{(i)})^T A^{(i)} = I, y^{(i)} \in \{0, 1\}^{m_i}, \sum_{j=1}^{m_i} y^{(i)}_j \geq 1, i = 1, 2, 3.$$  

In the new formulation, each block $(A^{(i)}Y^{(i)})^T, i = 1, 2, 3$ can be separately optimized, while the other blocks are fixed. This new optimization task can be performed, using the Maximum Block Improvement (MBI) [CHLZ12]. As we can see, in the new formulation, Tucker core size is not required to be pre-specified, because is now included in the optimization task. After MBI converges, the number of non-zero entries in $Y^{(1)}$, $Y^{(2)}$ and $Y^{(3)}$ will be equal to $r_1$, $r_2$ and $r_3$. The full description of ACS-Tucker is presented in Algorithm 2 of [CLZ14].

### 6.2.3 Event detection

As is illustrated in Fig. 6.3, after we obtain the factor matrices from decomposition of flow and topology tensors, we combine time factor matrices corresponding to both tensors and form a new hybrid matrix called hybrid time factor (HTF) which has $T$ rows (total time instants) and $k$ columns where $k$ denotes number of columns in flow time factor plus number of factors in the topology time factor. The $k$, however, 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 \( t = t_n (1 \leq n < T) \) when the multivariate series gets out of control.

Hotelling’s \( T^2 \) statistic is a common metric for monitoring multivariate time series, which is computed as follows [YS14].

\[
T^2_t = (X_t - \mu)^T S^{-1} (X_t - \mu)
\]

where \( \mu \) is the mean and \( X_t \) is the multivariate observation at time \( t \), and \( S \) is the covariance matrix.

If we assume that in an under control condition, multivariate time series (HTF) follows a multivariate normal distribution, \( T^2 \) should be explained by \( \chi^2 \) distribution with \( k \) degrees of freedom [MY02, p. 23] where \( k \) is the number of time series. Therefore, if some abnormal event occurs at the specific time instant, either in flow or topology we should witness a deviation from the \( \chi^2 \) distribution at that moment. The Cumulative Distribution Function (CDF) for the \( \chi^2 \) distribution [Thi88, p. 333], which is shown in the Eq. 6.6 computes this deviation. Consequently, the Eq. 6.7 is equivalent to the statistical significance (p-value) for each time instant, which shows the severity of deviation in each time, i.e. null hypothesis: no abnormal event.

\[
CDF = F(x|k) = \int_0^x \frac{t^{(k-2)/2}e^{-t/2}}{2^{k/2}\Gamma(k/2)} \, dt
\]

\[
P = 1 - CDF
\]

P-values closer to zero represent more severe events. In Eq. 6.6, \( \Gamma \) denotes the Gamma function and \( k \) denotes the degree of freedom, equals to the number of columns in HTF matrix and \( x \) refers the obtained Hotelling’s \( T^2 \) statistics in Eq. 6.5.

### 6.3 Experimental settings

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

There exists no hybrid tensor model in the literature like what we propose in this work. Therefore, we develop a similar baseline method to be able to compare our proposed model with. We also compare against two matrix residual models (see Fig. 6.2) including DTA-based, PCA-based and an SVD-based approach. These approaches are described in the following. For simplicity to refer these methods in the paper, we present their abbreviated names in the parenthesis.

**Baseline hybrid tensor model (Baseline)** Basically, we have to compare our proposed model versus a naive Tucker model with manually chosen core size. But, that comparison is not fair enough, because our approach benefits from an adjustable core size technique. We should compare to a baseline that as well as its counterpart is equipped with the ability of automatic parameter tuning. To do so, we apply a residual-based technique such as DIFFIT [KK03] that enables baseline approach automatically choose the numbers of components. Then we feed the obtained parameters to the Tucker-ALS algorithm. The rest of the baseline algorithm 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) [CC70] for solving Tucker decomposition while our proposed model uses Maximum Block Improvement (MBI) [CHLZ12]; the second difference is that baseline method instead of a single-step decomposition procedure, uses a two-step classic strategy of DIFFIT+Tucker.

**DTA-Residual (DTA-R)** Dynamic Tensor Analysis (DTA) is the first reported tensor solution for analysis of network traffic [STF06]. 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-R)** This method is similar to the method proposed in [IK04], with this difference that we apply 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.
Eigenvalue  Similar to [IK04] we apply SVD on the adjacency matrices of traffic flows and then the principal eigenvalue is obtained for each adjacency matrix. The time series of principal eigenvalues is then monitored using z-score control chart.

6.3.2  Software

All the experiments are performed in MATLAB on a PC with Intel Core 2 Duo CPU and 3GB RAM. Two MATLAB toolboxes are also used during the experiments: Tensor toolbox [BK+12a] and ITA toolbox [Sun12].

6.3.3  Used parameters

We set $c$ parameter in ACS-Tucker as $5+5+5=15$ as well as $\lambda = 0.005\|X\|$ as the default value proposed by [CLZ14]. For DIFFIT we set the max core size is set as $(5,5,5)$ as same ceiling we choose for ACS-Tucker. For DTA we use an input parameter of $(2,2)$ and for PCA we set number of principal components as 2.

6.3.4  Evaluation metric

We use Area Under ROC curve ($AUC$) [Bra97] as the main evaluation metric in this work, because of two reasons. Firstly, the area under the curve is not dependent on the chosen decision threshold, hence making the evaluation independent of the p-value threshold. This is an important criterion for 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 alpha threshold is varied from 0 to 1 by the step of 0.0001 in all compared methods. For a labeled data set we are able to measure the ability of methods in detecting events, with respect to the alpha in the loop. After computing false alarms rate and true alarms rate, we plot the ROC curve and then finally compute the AUC.
6.4 Simulation study

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

6.4.1 Simulation of artificial events in real traffic data

Among different simulation strategies, the injection of artificial events into real background data provides more validity than wholly-simulated test sets [BBC+05]. 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 have a conflict with the existing anomalies, resulting in a misleading conclusion.

![Real background data](image)

**Figure 6.4:** Three types of artificial events injected into the real background data.

An ideal data set for evaluation of 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 [BKP09,
BK12b] more suitable according to the above-mentioned requirements. The data set contains the trade flows for pairs of world countries between 1870 to 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 [FTOG^12], which is based on a combination of Tucker decomposition, clustering and visualization. During this process, we remove information about 30 years from the raw data. The final tensor \( \text{country} \times \text{country} \times \text{years} \) is with the size of \( 207 \times 207 \times 110 \).

As examples in Fig. 6.1 we inject three types of artificial events:

- Event type 1 (mutation in overall traffic flow) : a random positive integer from range \((\text{max}/8, \text{max})\) 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 \((1, \text{avg})\) is added to the zero elements in \(a \times a\) area.

- Event type 3 (chaotic behavior in a part of the network) : each cell in \(a \times a\) area is replaced with a random integer from range \((\text{max}/8, \text{max})\)

These three injected event types are illustrated in Fig. 6.4. First, the maximum \((\text{max})\) and the average \((\text{avg})\) of the tensor are computed, then for type 1, a matrix of uniformly distributed random integers is generated from the range \((\text{max}/8, \text{max})\). Subsequently, the generated matrix is added to all the elements within the adjacency matrix. For the second type, a \(a \times a\) area is selected and a random integer from the range \((1, \text{avg})\) 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 \((\text{max}/8, \text{max})\) and it 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 event type 1 in adjacency matrices corresponding to four time instants. In the second scenario, we inject event type 2 in four instants while in the third scenario we inject event type 3 in four matrices. Finally, in the fourth scenario, we inject a mixture of type 1, type 2, and type 3, each of which on four time instants, in total 12 time instants.

We also simulate low-scale events by varying the parameter \(a\) from 2 (1% of nodes) to 30 (15% of nodes) with the step of 2 and large-scale events by varying \(a\) from 50 (quarter of nodes) to 200 (almost all nodes) with the step of 10.
Table 6.1: Average AUC for event injection on the 15 areas from $2 \times 2$ to $30 \times 30$. The total possible injection area is $207 \times 207$ 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.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Hybrid tensor models</th>
<th>Matrix-based models</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Baseline</td>
<td>HTM (proposed)</td>
</tr>
<tr>
<td>1</td>
<td>0.9605</td>
<td>0.9548</td>
</tr>
<tr>
<td>2</td>
<td>0.9293</td>
<td>1.0000</td>
</tr>
<tr>
<td>3</td>
<td>0.4805</td>
<td>0.5775</td>
</tr>
<tr>
<td>Mixture</td>
<td>0.7672</td>
<td>0.8276</td>
</tr>
</tbody>
</table>

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.

6.4.1.1 Evaluation of methods in detection of different event types

Table 6.1 demonstrates the averaged AUC for low-scale events, regarding four simulation scenarios, i.e. event type 1, 2 and 3 and the mixture one.

All methods have a very good performance for detection of type 1 events indicating that detection of this kind is relatively straightforward for both groups of methods. An interesting point, however, is that incorporation of the topology model has a negative effect of almost 5%. 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 event type 2 except matrix residual models (i.e. DTA-R and PCA-R), other methods present a good performance. We recall that event type 2 is a simulation for alteration of topology in a part of the network. The obtained results here reveal that the matrix residual models are unable to detect topological fluctuations. Comparing HTM against the baseline method also reveals that ACS-Tucker has been quite helpful strategy in comparison of the traditional DIFFIT+Tucker model.

Nevertheless, DTA-R contrary to its low performance for type 2 events, as is expected performs better for chaotic fluctuations (i.e. event type 3). As a matter of fact, detection of such patterns is one of the main application of DTA. For instance, we know
that DTA can effectively track new objects in video streams [ZSH+11]. Appearance of new objects in video scene is equivalent to the chaotic fluctuations we define here.

According to the above results, we can infer that if we know, in advance, that the system only faces with event type 3, we may find DTA-R an ideal approach. Likewise, if we deal with only event type 2, probably tracking Eigenvalue alone should be sufficient. However, in reality, we do not have any prior knowledge about the types of events that the system faces with, rather we deal with a mixture of these events in the system 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 in handling such realistic circumstances. The results in Table 6.1 clarify the superiority of our proposed method over other techniques. As we can see, HTM has a better overall performance in dealing with different event types, both over matrix-based models (10-20 % better) and the baseline method (6% improvement).

6.4.1.2 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 tensor improves the naive scheme. Therefore, we focus on the two tensor models including our proposed one and the baseline method described in section 6.3.1. We compute the mean AUC for a mixture event scenario in two situations of low-scale and large-scale events.

The obtained results for low-scale mixture events presented in Table 6.2 indicate that our hybrid tensor strategy considerably improves the detection accuracy up to 13% with the proposed HTM and around 9% with the baseline method. More interesting observation is that when topology tensor is applied alone, it fails to detect events, even worse than a random detector (lower than 50% accuracy) but when it gets combined with flow tensor it leads to performance improvement. This is an evidence for an auxiliary role of the topology tensor in tensor-based event detection.

We also observe that in large-scale events when the majority of the nodes get involved, the events are much easier to discover by either of matrix-based or tensor-based models. 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 so rare that the whole system gets affected by a chaotic or topological mutation. However, we see that incorporating the topology model has more value for low-scale events than large-scale
events.

Table 6.2: Average AUC for event injection with a mixture scenario for low-scale (15 areas from $2 \times 2$ to $30 \times 30$) and large-scale (16 areas from $50 \times 50$ to $200 \times 200$). Flow: only naive tensor model is used. Topology: only topology tensor model is used. Hybrid: our proposed hybrid model that combines both naive and topology models.

<table>
<thead>
<tr>
<th>Used Tensor</th>
<th>low-scale events</th>
<th>large-scale events</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Baseline</td>
<td>HTM(proposed)</td>
</tr>
<tr>
<td>Flow</td>
<td>0.6736</td>
<td>0.6919</td>
</tr>
<tr>
<td>Topology</td>
<td>0.4607</td>
<td>0.4942</td>
</tr>
<tr>
<td>Hybrid</td>
<td>0.7672</td>
<td>0.8276</td>
</tr>
</tbody>
</table>

6.5 Real case studies

Table 6.3: Real-world data sets used in the experiments.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Dimensions</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>World [BK12b, BKP09] Trade Data</td>
<td>$Country \times Country \times Year$</td>
<td>$207 \times 207 \times 140$</td>
</tr>
<tr>
<td>US flight Data [U.S13]</td>
<td>$Airport \times Airport \times Month$</td>
<td>$169 \times 368 \times 195$</td>
</tr>
<tr>
<td>Bike-sharing Washington D.C. [Cap13]</td>
<td>$Station \times Station \times Day$</td>
<td>$157 \times 157 \times 731$</td>
</tr>
<tr>
<td>Bike-sharing Boston [Hub13]</td>
<td>$Station \times Station \times Day$</td>
<td>$95 \times 95 \times 327$</td>
</tr>
</tbody>
</table>

There exist some popular traffic data sets such as KDD Cup 99 [Arc14] that are widely used for analysis of Internet traffic or PeMS [oT14] 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 the real occurred events, and worse, there is no external knowledge source for labeling events [FTG14]. Benchmark data sets such as KDD Cup 99 are also repeatedly criticized in different aspects in terms of the quality of ground truth (e.g. in [TBLG09]). In this study as is listed in Table 6.3 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 have a more comprehensive assessment. In the following subsections, each data set will be first described in detail and then obtained
results will be presented. Note that the experiments settings are the same as section 6.3.

6.5.1 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 at another station. 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, 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 monitoring these data.

Our objective in this case study is the examination of methods in terms of model quality and stability. Our expectation from 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.

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

Washington, D.C. bike-sharing data set [Cap13] includes a historical usage log of all transactions in the bike-sharing network in a two-year window from 2011-01-01 to 2012-12-31 total in 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, such 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 have 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 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. 6.2 results in a traffic tensor by size of 157 × 157 × 731.
6.5.1.2 Boston bike-sharing data set

Boston bike-sharing data set has been extracted from hub-way data challenge 2013 [Hub13]. It includes a historical usage log of all transactions in the network from 2011-07-28 to 2012-10-01, exclusive of the system’s off-days in 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 $95 \times 95 \times 327$.

6.5.1.3 Event labeling and evaluation strategy

It is expected that events in the city affect users functioning, because, bike users are a sample of the city population and their behavior can be considered a sample of the entire city 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 hybrid model. Here, we are interested to evaluate the hybrid model on a real-world data with its complex event patterns. In Chapter 5 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.

6.5.1.4 Results

Table 6.4: AUC for bike sharing data sets. The ability of methods in detection of events from Boston and Washington, D.C. data sets.

<table>
<thead>
<tr>
<th>Method family</th>
<th>Method Name</th>
<th>Boston</th>
<th>Washington, D.C.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid tensor models</td>
<td>Baseline</td>
<td>0.7529</td>
<td>0.6367</td>
</tr>
<tr>
<td></td>
<td>HTM (proposed)</td>
<td><strong>0.8188</strong></td>
<td>0.7414</td>
</tr>
<tr>
<td>Matrix-based models</td>
<td>Eigenvalue</td>
<td>0.5709</td>
<td>0.7227</td>
</tr>
<tr>
<td></td>
<td>PCA-R</td>
<td>0.7575</td>
<td>0.5440</td>
</tr>
<tr>
<td></td>
<td>DTA-R</td>
<td>0.5782</td>
<td><strong>0.7599</strong></td>
</tr>
</tbody>
</table>

We compare the performance of methods with respect to their ability to detect the labeled events. In this case study, we use AUC for the evaluation. The results in Table 6.4 indicate that our proposed model comparing to other methods has a better and robust performance for both data sets. Although, DTA-R gains 2% excellence over
HTM for Washington, D.C. data set, its accuracy drops sharply by 24% for Boston data set. In contrast, PCA-R exhibits a reasonable performance on Boston data set, but performs bad for Washington, D.C. data set. It appears that neither of matrix residual methods are as robust as hybrid tensor models, but are rather dependent on the data characteristics.

Hybrid tensor models, on the other side exhibit a more stable performance on both data sets, though HTM has a relatively better accuracy than the baseline method.

### 6.5.2 World trade data set

Table 6.5: Statistical significance($p$) of events related to the world financial crisis 2007-2009. The events with $p-value \leq 0.05$ are shown boldface.

<table>
<thead>
<tr>
<th>Method family</th>
<th>Method Name</th>
<th>2007</th>
<th>2008</th>
<th>2009</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid tensor</td>
<td>Baseline</td>
<td>$3.06 \times 10^{-6}$</td>
<td>$2.05 \times 10^{-9}$</td>
<td>$4.46 \times 10^{-7}$</td>
</tr>
<tr>
<td></td>
<td>HTM(proposed)</td>
<td>$7.15 \times 10^{-5}$</td>
<td>$4.84 \times 10^{-8}$</td>
<td>$1.42 \times 10^{-6}$</td>
</tr>
<tr>
<td>Matrix-based</td>
<td>Eigenvalue</td>
<td>0.0043</td>
<td>0.0120</td>
<td>0.0372</td>
</tr>
<tr>
<td></td>
<td>PCA-R</td>
<td>0.2100</td>
<td>0.0338</td>
<td>0.0592</td>
</tr>
<tr>
<td></td>
<td>DTA-R</td>
<td>0.4848</td>
<td>0.1302</td>
<td>0.1386</td>
</tr>
</tbody>
</table>

The world trade data set [BK12b, BKP09] includes the bilateral trade flows between countries for the period from 1870-2009. The objective is to investigate the ability of methods in detecting the well-known global financial crisis in the period of 2007-2009 [APRR09].

Transforming the raw data to the $Country \times Country \times Year$ scheme results in a tensor with the 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 specifically can be interpreted as the probability that an abnormal event occurs 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 6.5. The p-values lower than 0.05 are shown boldface. As we can see, matrix-based models fail in rejection of the null hypothesis for crisis years in some years, while both hybrid tensor models reject null hypothesis very strongly. Among matrix-based models, DTA-R is
defeated in all cases, PCR-R fails in adequately detecting of 2007 and 2009. Comparing these two, Eigenvalue approach appears more effective in detection of crisis period, however, except for 2007, in other years its p-value is not strong as the tensor models.

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 with 2007 and 2009. This leads to a new insight about the crisis, apparently its crisis situation has been more severe in 2008.

6.5.3 US international flights data set

Table 6.6: Detection of bankruptcies in the United States airlines. \( \alpha \leq 0.05 \). Boldface values are related to true alarms and non-bold values are associated with false alarms.

<table>
<thead>
<tr>
<th>Method family</th>
<th>Method Name</th>
<th>Alarms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hybrid tensor</td>
<td>Baseline</td>
<td><strong>1991/02, 2002/08, 2006/02</strong></td>
</tr>
<tr>
<td></td>
<td>HTM (proposed)</td>
<td><strong>1991/02, 2002/08</strong></td>
</tr>
<tr>
<td></td>
<td>DTA-R</td>
<td>None</td>
</tr>
</tbody>
</table>

USA international air passenger statistics [U.S13] reports the commercial traffic traveling between international points and U.S. airports from 1990 to 2013. We transform the raw data to a flow tensor \( Airport \times Airport \times Month \) in size of \( 169 \times 368 \times 195 \). Our objective is to evaluate the capability of the methods in the tracking of events related to the most 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 Jan 1991, 18 Jan 1991 and 11 Aug 2002 [Bos11].

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 6.6. The months related to the aforementioned bankruptcies are shown boldface, 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 and PCA-R 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, three methods, including HTM, the baseline one, and Eigenvalue 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 since opposed to the baseline it does not have any false discovery. On the other side, it appears that matrix-based approaches do not present an acceptable performance. PCA-R and Eigenvalue methods report respectively 18 and 15 false alarms. DTA-R also conservatively does not report anything.

6.6 Scalability issues

Tensor models in principle are known to be computationally expensive. For decomposition of a cubic $n \times n \times n$ tensor with conventional Tucker or PARAFAC methods we require $O(n^4)$ time and $O(n^3)$ space. Our hybrid model in this study has the same complexity as other existing 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. Scalability of tensor decomposition methods is recently addressed in a couple of works which are discussed in Section 2.6.7.

6.7 Conclusion

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 separation and inclusion of network topology in the naive tensor model. Our simulation experiments show that the proposed hybrid approach has a better detection power up to 20% in comparison of matrix-based approaches in dealing with mixture events. Furthermore, our experiments on two data sets from a single domain reveals that the proposed approach is more robust-to-domain than the matrix models. We also confirm the
findings in [CLZ14] 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 endorses our findings in the simulation study about the robustness and the adequacy of the proposed model. The data set introduced in this paper is also found useful and can be potential choices for assessment of future traffic analysis approaches. Another interesting challenge 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.
Chapter 7

Multi-aspect-streaming Method

In this chapter we address some fundamental problems in tensor-based event detection such as adaptivity (section 2.6.8) and Scalability (section 2.6.7). Nowadays, the most important challenges in tensor analysis are efficiency and adaptability. Still, the majority of techniques are not scalable or not applicable in streaming settings. One of the promising frameworks that simultaneously addresses these two issues is Incremental Tensor Analysis (ITA) that includes three variants called Dynamic tensor analysis (DTA), Streaming tensor analysis (STA) and Window-based tensor analysis (WTA). However, ITA restricts the tensor growth only in time, which is a huge constraint in scalability and adaptability of other modes. We propose a new approach called multi-aspect-streaming tensor analysis (MASTA) that relaxes this constraint and allows the tensor to concurrently evolve through all modes. The new approach, which is developed for anomaly and event detection purposes, instead of relying on expensive linear algebra techniques is founded on the histogram approximation concept. This consequently brought simplicity, adaptability, efficiency and flexibility to the tensor analysis task. The empirical evaluation on various data sets from several domains reveals that MASTA is a potential technique with a competitive value against ITA algorithms.

7.1 Introduction

The growing interest in tensors is due to their capability of discovering complicated patterns in multiway settings that is impossible via other methods. Existing tensor decomposition models suffer from two major issues. Firstly, they are not scalable to
large size data sets due to their time/space complexity; and secondly, are not updatable when a new stream of data is retrieved.

The scalability issue is already addressed in three major groups of solutions, including sparse-optimized methods, parallel and distributed techniques and GPU-based solutions (see section 2.6.7). However, though the scalable techniques are great tools for dealing with large tensors, they suffer from the non-adaptability problem. This means that when new data is received we have to rebuild the model from scratch. In addition, sparsity-optimized techniques do not have any added value for dense tensors, because they only scale up when there is a considerable amount of zero elements in the tensor. Furthermore, the parallelization of tensor decompositions is not as straightforward and it requires extra hardware and software infrastructures.

The pioneer research studies on this problem are those performed by [STF06, SPP06, STP+08b] who propose some streaming approximation solutions for tensor decomposition in an unified framework called incremental tensor analysis (ITA). The ITA solution, opposed to other scalable decomposition techniques does not need any special infrastructure. It also does not make any restrictive assumption like sparsity. It performs tensor decomposition on each tensor in each time instant, maintains some statistics and then incorporates that for the processing of the next tensor. Therefore, it does not require keeping historical data in the memory. This solution has two advantages. First, tensor model is easily updatable when new data arrives, and second, the space required for decomposition of the tensor becomes independent of stream length.

The merits of ITA and its usefulness to the analysis of time-evolving tensors are investigated in many studies, nowadays, ITA is recognized as the state-of-the-art solution for streaming tensor analysis. However, although ITA allows the tensor to evolve infinitely in time, it makes a restrictive assumption that the dimension of the tensor remains constant during the process. We may not find this limitation annoying for only-time-evolving tensors like network traffic or video streams, when the number of nodes or image frames remain constant during the analysis. But, we may deeply feel this constraint in dealing with multi-aspect-evolving tensors such as social networks, where the number of nodes grows during the evolution of the network. Or in recommender systems when new users are joined to the system, and size of user × profile matrix consistently changes. Aside from that, ITA encounters the intermediate data explosion problem [KPHF12, KS08] as well as its offline counterparts when the size of the tensor is large.
CHAPTER 7. MULTI-ASPECT-STREAMING METHOD

The intermediate data explosion problem corresponds to the heart of the these techniques, i.e. space-inefficient linear algebra computations that operate directly on the input data. Therefore, in these methods, space efficiency is more influenced by the size of input data rather than the method per se. However, we know that a large portion of tensor decomposition applications is related to analysis-only tasks such as anomaly and event detection (e.g. [SGJ14, KLMW09, GCP10, PTKH12]) or simple data analysis (e.g. [LLK+14, MWP+14, BMG+13, HFSES13]). In such applications, computing the exact subspace of the tensor may not seem mandatory, as opposed to other applications such as compression where the reconstruction of tensor is inevitable. Can we find an alternative adaptive solution for tensor analysis that on one hand avoids space-inefficient computations and on the other hand provides the basic analytical power of tensor decomposition?

We know that histograms are central tools for summarization in data mining. They are also the key technique in image retrieval for measuring similarity between images. Is it possible to extend these ideas to tensor analysis problem? We may find a positive answer for this question, but two more questions will be raised in the following: a) how do we deal with the huge space/time complexity of histograms while we actually require an efficient method?; b) is it conceivable to utilize a non-adaptive tool like histogram for solving a streaming problem?

![Multi-Aspect-Streaming Vs. Streaming](image)

Figure 7.1: Comparison of multi-aspect-streaming tensor analysis (proposed) versus streaming tensor analysis (state-of-the-art).

In this research, we tackle these problems by recommending a histogram-based solution that allows the tensor to simultaneously evolve through all modes. We initiate with the description of fundamental concepts such as histograms, tensor segmentation and distribution matching and proceed to develop the first basic approach for histogram-based tensor analysis. Furthermore, we extend the baseline solution to the multi-
aspect-streaming scheme (see Fig. 7.1) by replacing the conventional histogram with a recent incremental approach. To the best of our knowledge, the application of histograms in tensor analysis is not reported elsewhere. This is also the first work that addresses multi-aspect-streaming tensor analysis problem.

The rest of the chapter is organized as follows. In section 7.2 we describe the proposed method. We introduce a new evaluation methodology in section 7.3 and later employ that for assessment of the proposal in section 7.4. Next, in section 7.5 we illustrate the application of the proposed approach on two real case studies. The last section concludes the exposition, presenting the final remarks.

7.2 Histogram-based tensor analysis

Histograms are simple statistical tools that have been applied in a wide range of applications [Loa03]. They are simple, non-parametric and easy to interpret, which make them attractive for summarizing data. Histograms are extensively used in the mining and processing of data streams [Gam10, GKS01, DGIM02] to keep the abstract of past data; in database management systems for cost estimation in query optimization [KW99, PI96]; and in image retrieval [SWS+00, PZ96] for image matching. With some inspirations from these applications, we intend to extend the application of histograms to tensor analysis problem. In the following we explain the logic for this selection.

We know that a $d$-dimensional tensor is composed of multiple $(d-1)$-dimensional slices in each mode. For a 3D tensor as is depicted in Fig. 7.2, slices are 2D matrices. Each slice contains a particular segment of the tensor information, so that if all slices get combined together they rebuild the original tensor. In tensor analysis, we assume that many of the features are correlated with others and they jointly explain the data. From image matching application, we know that histograms are useful for measuring the similarities between the two images. Hence, it is rational to assume:

**Hypothesis 7.1.** If two features are similar, the histogram of their corresponding slices should be similar as well.

However, we may have two slices with totally different correlation patterns that have similar histograms. For instance, in image matching we may find two different images with similar histograms [PZ96]. Therefore, this assumption might be violated in some applications. However, we presume that this is not the case in the majority of applications. Aside from this, measuring distribution distances between all pairs
of slices is an exhaustive task. We believe that if two slices have similar distances
to the tensor distribution (as a reference), they probably are similar because they
explain the same part of tensor information. Therefore, maybe it is better to instead
of performing exhaustive match between all pairs of slices, only compute the similarity
of slice histograms to the tensor histogram. However, this needs to be assumed:

**Hypothesis 7.2.** Two features are similar, if histograms of their corresponding slices
have a similar distance to the tensor histogram.

With the new assumption, anomalous slices (or features) are also easy to discover.
Those slices that have a totally different histogram when comparing to other slices are
considered abnormal, so if we remove them from the tensor, we still can explain the
majority of information using the remaining slices.

Using the above assumptions, we propose the first histogram-based method for tensor
analysis. In the following subsections we first present the detailed presentation of the
basic algorithm and then proceed with the introduction of its multi-aspect-streaming
extension.

### 7.2.1 Offline histogram-based tensor analysis (OHTA)

In this section we introduce OHTA, a baseline algorithm for histogram-based tensor
analysis. This method which is presented in Algorithm 7.1 requires three inputs:
tensor $\mathcal{X}$, number of bins for reference histogram $b_1$ and number of bins for distances
histogram $b_2$. The $b_2$ parameter is somehow equivalent to the number of components
in PARAFAC decomposition in the sense that OHTA finds a $b_2$ group of features in
each mode.

The algorithm initiates with vectorization of the tensor $\mathcal{X}$ where $d$-dimensional tensor
is transformed into a one-dimensional vector $x$ (Fig. 7.2-1) and then its histogram
which is called reference histogram (or simply tensor histogram) is calculated with
$b_1$ numbers of bins. Next, for each slice in each mode (Fig. 7.2-2) we compute its
histogram according to the bins obtained for the reference histogram (Fig. 7.2-3).
Note that, the way we generate histograms for slices is different from when we apply
the histogram directly to the slices. Here, for all slices we use the same bins as the
reference histogram. For instance in Fig 7.2, if the bins for the reference histogram
are $(25, 20, 15, 10, 5)$ and vectorized slice $1,1$ be $(26, 24, 19, 27, 14, 16, 10, 4, 3)$,
the histogram of slice $1,1$ will be $(25:3, 20:1, 15:2, 10:1, 5:2)$. This is obtained as
follows. We first create an empty copy of the reference histogram (25:0, 20:0, 15:0, 10:0, 5:0). Then for each quantity in the slice we add one to the count corresponding to the closest point. For instance, for 26 the closest bin in the reference histogram is 25. For 24 and 27 the closest point is also 25. Therefore, bin 25 gets frequency count of 3. For bin 20, the count is 1 because among all values only one item, i.e., 19 has been the closest point to 20. We do not count 24 for bin 20 because 24 is closer to 25 than 20. Via this procedure, we generate the histograms for all slices.

In the next step, we compute the distance of each slice histogram to the reference histogram (Fig. 7.2-4) and save the distances for each mode in \( d \), vector (Fig. 7.2-5). The distance measure we use here is Earth Mover's distance (EMD)[RTG00] which is
Algorithm 7.1 OHTA

**Input:** Tensor $\mathcal{X}$, bins in reference histogram ($b_1$), bins in distances histogram ($b_2$)

**Output:** $H_i$

1: Transform Tensor $\mathcal{X}$ to vector $\mathbf{x}$
2: Reference Histogram $\leftarrow$ Histogram ($\mathbf{x}, b_1$)
3: for each mode $i$ do
4:   for each slices do
5:     Create histogram of the slice according the bin centers obtained for reference histogram
6:     $d_i \leftarrow$ EMD(slice histogram, reference histogram)
7:   end for
8: $H_i \leftarrow$ Histogram ($d_i, b_2$)
9: end for

widely used in image retrieval to compute distances between the color histograms of two images. We may want to use other distance measures, but EMD is the only one that is suitable for partial match purposes [RTG00], hence it seems more appropriate here.

We finally compute the histogram of EMD distances, $d_i$ for mode $i$ with $b_2$ numbers of bins (Fig. 7.2-6). The constructed histogram, $H_i$ is the output of OHTA. Each bin in $H_i$ represents a cluster of features and the cluster support is equal to frequency count of the histogram.

### 7.2.2 Multi-aspect-streaming tensor analysis (MASTA)

Although histograms are very useful tools for summarization of data, they have two problems that make them impractical for streaming or large-scale data analysis. Firstly, they are computationally very expensive; and secondly it is not possible to update the histogram when new data arrives.

Algorithm 7.2 MASTA Update

**Input:** data chunk $s, b_1$, old reference histogram, old slice histograms for each mode

**Output:** reference histogram, slice histograms for each mode

1: Update reference histogram with respect to elements in $s, b_1$ and old reference histogram
2: Update or add histogram of slices corresponding to modes of $s$, old slice histogram

Fortunately, advances in data stream mining and database management systems has
Algorithm 7.3 MASTA Output

Input: reference histogram, slice histograms for each mode, $b2$
Output: $H_i$
1: for each mode $i$ do
2:   for each slice $j$ do
3:       $R_j \leftarrow$ Compute reconstructed slice histogram using Algorithm 7.4
4:       $d_i^{\text{add}} \leftarrow \text{EMD}(R_j, \text{reference histogram})$
5:       $H_i \leftarrow$ Update Histogram $(d_i, b2)$
6:   end for
7: end for

Algorithm 7.4 MASTA Slice histogram reconstruction

Input: reference histogram, slice histogram, $b1$
Output: $R$
1: $R \leftarrow$ copy reference histogram with frequency counts of zero
2: for each bin in $R$ do
3:   Find the closest point in slice histogram
4:   Add the frequency count corresponding to the found point to the current bin’s count.
5: end for

brought new efficient techniques for histogram approximation. Two of these recent techniques are [GKS06] and [BHTT10] which propose an online approach for histogram approximation respectively, with space consumption of $O(\log n)$ and $O(1)$. In this work we apply the latter approach.

The idea of online histogram approximation in [BHTT10] is very simple. Instead of keeping the whole data, we update the old histogram through procedures such as update, merge, sum and uniform. The idea is that given a determined maximum number of bins when new data item arrives, if its value is close to each of the previous bins it is allocated to that bin and the corresponding bin center is updated accordingly. However, if the new data item is far away from the previous bins, a new bin is created and two most close bins are merged. This process continues until approximation of the whole data histogram.

Employment of an online histogram in OHTA accomplishes two functions: first, promises a huge space efficiency, because space complexity of the online algorithm is $O(1)$ versus the expensive offline approach $O(N)$; and second, we do not face with the intermediate data explosion problem [KPHF12, KS08]. Because, each piece of data upon arrival enters to the model, updates the model and then is removed from the memory. This piece of data necessarily should not be a tensor or matrix as existing
CHAPTER 7. MULTI-ASPECT-STREAMING METHOD

tensor solutions, rather it can be a single element of the tensor.

According to the above description, we develop the multi-aspect-streaming edition of OHTA, called MASTA by replacing the exact histogram calculation in OHTA with the above-mentioned online method. Some other modifications are also required to be taken into account that will be explained in the following. MASTA is composed of three procedures called update (Algorithm 7.2), output (Algorithm 7.3) and slice reconstruction (Algorithm 7.4).

Algorithm 7.2 presents the update procedure. This process is responsible for updating the model upon data arrival. In this algorithm we update the reference histogram and slice histograms in all modes upon new stream arrives. We also create the histogram if it does not exist. A toy example of this process is illustrated in Fig. 7.3.

Algorithm 7.3 is executed when the user asks for the model result. The inputs of this algorithm are the outputs of Algorithm 7.2. In the output procedure, the first step is rebuilding the slice histogram with respect to the reference histogram via Algorithm 7.4. The reconstruction process in Algorithm 7.4 works as follows. We create an empty histogram with same bins as reference histogram and then instead of allocating the original counts in reference histogram we calculate the counts from the slice histograms. Via this procedure, we build histograms for all slices. After that, as OHTA we compute the EMD distance of each slice histogram to the reference histogram. We keep distances for all slices in each mode \( d_i \). Finally, similar to OHTA we compute the online histogram of distance vectors, \( d_i \) and report the slice indices and the corresponding frequency count for each bin.

7.3 Experimental evaluation

An ideal tensor analysis approach is the one that presents the most accurate model while it uses less resources (time and space). Therefore, as well as many learning algorithms, usefulness of any approximation solution should be evaluated based on a trade-off between accuracy and efficiency. In this section, we introduce the datasets, experimental settings, and the evaluation strategy we use for assessment of the methods. Finally, we examine the efficiency of the proposed approaches in terms of both runtime and memory consumption.
Figure 7.3: MASTA Update process. The initial state is shown when MASTA is already applied on tensor $\mathcal{X}(2 \times 2 \times 2)$ (corresponding to elements 1 to 8). At the end of this moment, MASTA outputs one reference histogram plus 6 slice histograms of $(1,1),(1,2),(2,1),(2,2),(3,1),(3,2)$. In the next stage as is highlighted with bold font and gray background, we receive a new stream, such as $S(1,1,3)=9$, $S(1,2,3)=10$, $S(2,1,3)=11$, $S(2,2,3)=12$. We first update the reference histogram using new elements of $(9, 10, 11, 12)$, and then update the histograms correspond to four slices $(1,1),(1,2),(2,1),(2,2)$ that are affected by the new stream. We also generate a new histogram for slice $(3,3)$, because all new elements belong to the third slice of the third dimension. Two histograms of slices $(3,1)$ and $(3,2)$ have remained unchanged, because are not affected by new elements. This example is for a time-evolving tensor (tensor that evolves only in time mode). However, MASTA allows tensor to evolve in all three directions. The only difference will be that if histogram evolves in all modes, we will not see unchanged slices like $(3,1)$ and $(3,2)$.

### 7.3.1 Data sets

As it is demonstrated in Table 7.1 we use several real-life data sets from various domains, including economy, neuroscience, epidemiology, climatology, video-surveillance, psychometric and transportation. Some statistical information about these data sets are presented in Table 7.2 such as tensor size, the optimum Tucker model parameter and its corresponding fit, the number of iterations for Tucker model, the percentage of non-zero values in tensor, class of values ("int" indicates integer and "float-" implies positive/negative float numbers). The mean and standard deviation of the non-zero values in tensors are also presented in the two last columns. In the following the detailed description of data sets are briefly presented. Note that all the used data sets
are publicly available and can be accessed via Internet.

**trade.** This dataset [BKP09] includes the bilateral trade flows (import/export in US dollars) between countries for the period from 1870-2009. We transform the raw data to tensor scheme of \( \text{country} \times \text{country} \times \text{year} \).

**eeg.** BCI III motor imagery dataset (4a) data set is made as a part of brain-computer interface study in [DBCM04]. According to the instructions in [Pha11] we extract trials from EEG continuous signals in this dataset to a four-dimensional tensor of \( \text{features} \times \text{time} \times \text{channel} \times \text{trail} \).

**newmexico.** Brain cancer incidence in New Mexico data set [Kul12a] includes the New Mexico Tumor Registry between the years of 1973 to 1991 for 32 sub-regions of the New Mexico state along with some categorical demographic features that together form 23 features. We generate a three-way tensor with the county in the first mode, year in the second mode and measurements in the third mode.

**spain02.** Spain02 data set [HGA+12] contains the 50-year high-resolution monthly gridded precipitation data over Spain. First and second modes of tensor for this data sets are respectively month and the spatial grid-id and third mode contains precipitation and (maximum and minimum) temperature.

**walk.** This video data set [The14] is extracted from CAVIAR project data sets which is filmed at INRIA Labs at Grenoble, France. The resolution of videos is in half-resolution PAL standard (384 x 288 pixels, 25 frames per second). A number of video clips were recorded in this project acting out the different scenarios of interest. In this work, we use the first 50 frames of CAVIAR /INRIA walking data set (walk1) where one human object appears in the video after a while of recording.

**flight delay.** This data set is a part of Data expo 09 set (airline on-time performance competition) [ASA14] consists of flight arrival and departure delays for all commercial flights within the USA, in 2008. The tensor is in scheme of \( \text{origin} \times \text{destination} \times \text{time} \) and elements in the tensor are the average daily delays measured for corresponding flights.
flight. USA international air passenger statistics [U.S13] that reports the commercial traffic traveling between international points and U.S. airports from 1990 to 2013. We transform the raw data to a flow tensor $airport \times airport \times month$ in this study.

bikeboston. This dataset has been extracted from hub-way data challenge 2013 [Hub13]. It includes a historical usage log of all transactions in the Boston bike sharing system from 2011-07-28 to 2012-10-01, exclusive of the system’s off-days in winter, a total of 327 days.

bikewashington. Washington, D.C. bike-sharing dataset [Cap13] includes a historical usage log of all transactions in the bike-sharing network in a two-year window from 2011-01-01 to 2012-12-31, in total, 731 days. We select top 157 stations that have more frequent trips.

taxi. This data set [YZXS11] contains a one-week trajectories of 10,357 taxis in Beijing, China, in a period between 2008-02-02 to 2008-02-08. We use a grid strategy to divide the spatial space into equal areas. Then in each zone, we calculate the number of existing taxis in each hour (for total 149 hours).

kojimagirls. This data set [Koj75] includes the judgments (in 20 scales) of 153 parents behavior with respect to their own 13-year child on four conditions of daughter-father, daughter-mother, father-father and mother-mother. The output tensor consist of a $subjects \times scales \times condition$.

7.3.2 Evaluation framework

Evaluation of tensor analysis methods is still a difficult challenge. When we want to compare one method versus its counterpart, normally the model fit or error rate is measured. Also in some application domains such as chemometrics, researchers usually use some field knowledge to validate the model.

Normally, when a novel tensor methodology is developed, the computational efficiency receives more attention than accuracy. On the other side, those researchers who develop new algorithms and methods for tensor analysis do not have sufficient access to the domain knowledge for performing a realistic validation. All these issues together
Table 7.1: Datasets

<table>
<thead>
<tr>
<th>tensor dataset</th>
<th>modes</th>
<th>domain</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trade [BKP09]</td>
<td>country × country × time</td>
<td>economy</td>
<td>global trade dynamics</td>
</tr>
<tr>
<td>eeg [DCBM04]</td>
<td>frequency × time × channel × trial</td>
<td>neuroscience</td>
<td>brain-computer interfacing</td>
</tr>
<tr>
<td>newmexico [Kul13a]</td>
<td>location × time × measurements</td>
<td>epidemiology</td>
<td>brain cancer in N.Mexico</td>
</tr>
<tr>
<td>span02 [HGA+12]</td>
<td>time × location × measurements</td>
<td>climatology</td>
<td>climate analysis over Spain</td>
</tr>
<tr>
<td>walk [The14]</td>
<td>image frames × time</td>
<td>video-survey</td>
<td>human motion analysis</td>
</tr>
<tr>
<td>flightdelay [ASA14]</td>
<td>airport × airport × time</td>
<td>transportation</td>
<td>airline on-time modeling</td>
</tr>
<tr>
<td>flight [U.S13]</td>
<td>country × country × time</td>
<td>transportation</td>
<td>air passengers demand modeling</td>
</tr>
<tr>
<td>bikeboston [Hub13]</td>
<td>station × station × time</td>
<td>transportation</td>
<td>bike sharing O/D flows in Boston</td>
</tr>
<tr>
<td>bikewashington [Cap13]</td>
<td>station × station × time</td>
<td>transportation</td>
<td>bike sharing O/D flows in W.D.C.</td>
</tr>
<tr>
<td>taxi [YXZS11]</td>
<td>time × region × region</td>
<td>transportation</td>
<td>taxi count matrix in Beijing</td>
</tr>
<tr>
<td>kojimagirls [Koj17]</td>
<td>sample × condition × measurements</td>
<td>psychometric</td>
<td>behavior analysis</td>
</tr>
</tbody>
</table>

make the evaluation of tensor analysis a difficult challenge. In our case, this difficulty is even greater, because our proposed solution is not decomposition-based like its counterparts.

In this chapter, we propose a new evaluation framework that is capable to solve this issue. This new methodology can be used for assessment of any new tensor analysis approach irrespective of the methodological details. However, we make two assumptions. The first is about the reliability of reference models where we assume:

**Hypothesis 7.3.** Tucker model with optimum model parameter (or PARAFAC with the best number of components) is the best possible model we can generate from the tensor.

The second assumption lies within the main hypothesis of spectral-based anomaly detection techniques mentioned in [CBK09][p. 37]: "data can be embedded into a lower dimensional subspace in which normal instances and anomalies appear significantly different". By extending this for tensor-based approaches, we infer that:

**Hypothesis 7.4.** An optimum Tucker/PARAFAC solution is an effective model for discriminating the normal and abnormal features in complex tensors.

Above assumptions state that if we apply Tucker or PARAFAC models we will be able to identify anomalies in the most effective way. Therefore, if we consider the
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Tensor size</th>
<th>best model</th>
<th>Fit</th>
<th>Itr.</th>
<th>Nuzro.</th>
<th>Type</th>
<th>Mean</th>
<th>STD</th>
</tr>
</thead>
<tbody>
<tr>
<td>trade</td>
<td>$207 \times 207 \times 140$</td>
<td>(3 3 2)</td>
<td>25.7</td>
<td>7</td>
<td>7.7</td>
<td>int+</td>
<td>3434.7</td>
<td>327.6</td>
</tr>
<tr>
<td>eeg</td>
<td>$23 \times 330 \times 7 \times 280$</td>
<td>(1 2 1 2)</td>
<td>53.2</td>
<td>5</td>
<td>100</td>
<td>int+</td>
<td>38.8</td>
<td>43.1</td>
</tr>
<tr>
<td>newmexico</td>
<td>$32 \times 19 \times 23$</td>
<td>(1 2 2)</td>
<td>25.8</td>
<td>6</td>
<td>6.8</td>
<td>int+</td>
<td>0.7</td>
<td>1.3</td>
</tr>
<tr>
<td>spain02</td>
<td>$699 \times 1445 \times 3$</td>
<td>(3 3 2)</td>
<td>61.0</td>
<td>7</td>
<td>64.0</td>
<td>float-</td>
<td>46.0</td>
<td>36.5</td>
</tr>
<tr>
<td>walk</td>
<td>$288 \times 384 \times 50$</td>
<td>(2 2 1)</td>
<td>74.4</td>
<td>5</td>
<td>100</td>
<td>int+</td>
<td>49.3</td>
<td>130.7</td>
</tr>
<tr>
<td>flightdelay</td>
<td>$152 \times 150 \times 305$</td>
<td>(2 2 1)</td>
<td>10.2</td>
<td>6</td>
<td>3.2</td>
<td>int+</td>
<td>66.2</td>
<td>114.8</td>
</tr>
<tr>
<td>flight</td>
<td>$77 \times 211 \times 282$</td>
<td>(3 3 i)</td>
<td>30.2</td>
<td>6</td>
<td>4.4</td>
<td>int+</td>
<td>7500.6</td>
<td>6161.0</td>
</tr>
<tr>
<td>bikeboston</td>
<td>$95 \times 95 \times 327$</td>
<td>(3 3 2)</td>
<td>21.4</td>
<td>5</td>
<td>9.5</td>
<td>int+</td>
<td>1.4</td>
<td>1.8</td>
</tr>
<tr>
<td>bikewashington</td>
<td>$157 \times 157 \times 731$</td>
<td>(2 2 1)</td>
<td>17.9</td>
<td>4</td>
<td>8.5</td>
<td>int+</td>
<td>1.7</td>
<td>1.9</td>
</tr>
<tr>
<td>taxi</td>
<td>$149 \times 275 \times 196$</td>
<td>(1 i 1)</td>
<td>87.5</td>
<td>3</td>
<td>0.2</td>
<td>int+</td>
<td>18.5</td>
<td>0.2</td>
</tr>
<tr>
<td>kojiimagirls</td>
<td>$153 \times 4 \times 20$</td>
<td>(2 1 2)</td>
<td>86.9</td>
<td>4</td>
<td>100</td>
<td>int+</td>
<td>18.7</td>
<td>21.4</td>
</tr>
</tbody>
</table>

Table 7.2: Descriptive statistics of data sets

We test two popular tensor decomposition solutions, Tucker and PARAFAC for selecting the reference model. In order to obtain the best number of components in PARAFAC we perform CONCORDIA test [BK03] and for Tucker we carry out a scree test (Tucktest in N-way toolbox [AB00]) which are popular methods in the literature for tensor model parameter tuning. Our experiments show that the fit corresponding to optimum model for these two decomposition techniques presents almost the same quantity for all data sets. However, the Tucker optimum model performs slightly better, because it has more flexibility in terms of imbalances in the number of components in each mode. Due to this reason, in the further experiments we only focus on the Tucker model as the reference model.

### 7.3.2.1 Choosing reference model

We test two popular tensor decomposition solutions, Tucker and PARAFAC for selecting the reference model. In order to obtain the best number of components in PARAFAC we perform CONCORDIA test [BK03] and for Tucker we carry out a scree test (Tucktest in N-way toolbox [AB00]) which are popular methods in the literature for tensor model parameter tuning. Our experiments show that the fit corresponding to optimum model for these two decomposition techniques presents almost the same quantity for all data sets. However, the Tucker optimum model performs slightly better, because it has more flexibility in terms of imbalances in the number of components in each mode. Due to this reason, in the further experiments we only focus on the Tucker model as the reference model.

### 7.3.2.2 Building reference model

The Tucker scree test operates as follows. A candidate list of model parameters is first generated by choosing a range from a minimum (e.g. 1) to a maximum (e.g. 5). Afterward, the Tucker model is built for all the possible combinations of parameters (e.g. (1 1 1), (2 1 1),… (5 5 5)). Next, we calculate the explained variance for each parameter and then demonstrate every obtained value for each possible parameter
on a plot called scree plot. By looking at this plot we can choose the best model parameter. A good parameter is the one that is simpler and explains more variance. For instance, between parameter (3 4 3) and (2 1 1) that respectively explain 75% and 73% of variance, the preference is to (2 1 1). Because, it has less complexity. Following this procedure, we manually obtain the optimum model parameters for each data set.

After selection of the optimum parameter for the Tucker model we apply Tucker decomposition with the obtained optimum parameter on each data set. The decomposition gives us a core tensor plus $d$ factor matrices for each mode. For a third order tensor $(n_1 \times n_2 \times n_3)$ and Tucker model parameter $(r_1, r_2, r_3)$, the factor matrices will be in dimensions of $n_1 \times r_1$, $n_2 \times r_2$ and $n_3 \times r_3$, respectively for mode 1, 2 and 3. The columns in the factor matrices denote the latent variables (or singular vectors). After decomposition, the features in each mode can be represented with these latent variables in a more compact way. For instance, if $r_1 = 2$, in mode 1 factor matrix is in $\mathbb{R}^2$ space where its x-axis and y-axis are respectively first and second column of the factor matrix.

Although observing anomalies in this two-dimensional space might seem straightforward, in a higher dimensional space (if $r >> 2$) it would not be so easy. In order to detect anomalies from higher dimensional spaces we require a multivariate outlier detection technique. We consider four different methods, including Wilks’s method [Wil63], Hotelling’s $T^2$ [MY02, p. 21], Minimum Covariance Determinant (MCD) [Rou84] and Minimum Volume Enclosing Ellipsoid (MVE) [SF04]. The preliminary assessment of outliers in accordance with some available prior knowledge about some data sets shows that Hotelling’s $T^2$ provides more reasonable results comparing other methods. Therefore, we use that in further steps. Hotelling’s $T^2$ statistics is computed as follows.

$$T^2_i = (X_i - \mu)^T S^{-1}(X_i - \mu)$$ (7.1)

Where $\mu$ is the mean and $X_i$ is the multivariate observation for feature $i$, and $S$ is the covariance matrix. We compute the $T^2_i$ for all factor matrices derived from Tucker decomposition. For instance, in the above example, we compute $T^2_i$ for factor matrices of three modes, $n_1 \times r_1$, $n_2 \times r_2$ and $n_3 \times r_3$.

The output will be $T^2_i$ for feature $i$ in each mode of tensor. Next, in order to identify the anomalous features we assume that $T^2$ follows the $\chi^2$ distribution with $k$ degrees of freedom [MY02, p. 23], where $k$ denotes number of columns in the factor matrices.
(e.g. $k = r_1$ for mode 1). Hence, features that have a significant deviation from this distribution are considered anomalies. The Cumulative Distribution Function (CDF) for the $\chi^2$ distribution [Thi88, p. 333], which is shown in the following equation computes this deviation. Consequently, the Eq. 7.3 is equivalent to the statistical significance (p-value) for each feature, i.e. null hypothesis: the feature is normal.

$$CDF_i = F(T_i^2 | k) = \int_0^{T_i^2} \frac{t^{(k-2)/2}e^{-t/2}}{2^{k/2} \Gamma(k/2)} dt$$  \hspace{1cm} (7.2)

$$P_i = 1 - CDF_i$$  \hspace{1cm} (7.3)

The lower $P$ the more anomalous feature is. Therefore, we sort the features based on the obtained $P$ and retrieve the top 5 percent anomalous features in each mode. We call this set "top-5% reference anomalies".

### 7.3.2.3 Competitors

We use three variants of ITA framework [STP+08b, STF06, SPP06] for comparison: Dynamic Tensor Analysis (DTA), Streaming Tensor Analysis (STA) and Window-based Tensor Analysis (WTA). DTA incrementally decomposes the tensor by maintaining only the covariance matrix for each arriving tensor. Then, via diagonalization it outputs the principal eigenvectors of the updated covariance matrix as projection matrices. STA attempts to approximate DTA. It instead of maintaining a covariance matrix for all arriving tensors, directly updates the principal eigenvectors using SPIRIT algorithm [PY06] which does not require diagonalization. STA runtime can be faster by decreasing the sampling percentage. For instance, STA with sampling percentage of 10% is almost 10 times faster than STA with 100% sampling percentage. The other algorithm, WTA instead of processing individual tensors uses a sliding window strategy for handling time dependency between consecutive tensors. It decomposes the sliding window with a regular Tucker or PARAFAC and then as well as DTA and STA keeps some statistics from the window in the processing of the next window.

The temporal information loss of ITA algorithms is not evaluated in the original papers [STP+08b, STF06, SPP06]. So, in parallel with the evaluation of main proposal, we evaluate this issue as well.
7.3.2.4 Settings

For MASTA and OHTA the input parameters are b1 and b2 which are chosen respectively 50 and 10. The required model parameters DTA, STA and WTA are chosen equal to what is obtained for the optimum number of components in CONCORDIA test corresponding to PARAFAC. The configuration we use for WTA is the independent-window mode (IW) with Tucker decomposition. Also, no forgetting factor is used for DTA and STA. For STA we test three different sampling rates of 100% (no sampling), 50% and 10% and for WTA we examine three window sizes of 4,7 and 10.

For a fare comparison, we proceed a blind evaluation strategy. In other words, we assume that we do not know the optimum parameters for OHTA and MASTA in advance. For this reason we do not use the best b1 and b2 we obtain in Fig. 7.4.

7.3.2.5 Software

We use MATLAB for the experiments along with two other toolboxes, tensor toolbox [BK12a] for computing Tucker and PARAFAC and n-way toolbox [AB00] for CONCORDIA and Tucker scree plot (tucktest). MATLAB implementation of ITA [Sun12] is also used for experimenting with DTA, STA and WTA. Furthermore, we use Golihistogram [Viv14], the Golang implementation of online histogram approximation [BHTT10].

7.3.2.6 Evaluation process

After we generate the reference model and top 5% reference anomalies we start to identify the top 5% anomalies via MASTA, OHTA, DTA, STA and WTA. In terms of MASTA and OHTA we apply them on the tensor data and then retrieve the top 5% low support items in each mode. Concerning DTA, STA and WTA we proceed the same procedure as was explained for Tucker in section 7.3.2.2, i.e. applying Hotelling’s $T^2$ on the non-evolving factor matrices. However, these algorithms operate incrementally on the evolving dimension of tensor which can be time or sample mode. For identification of anomalies in the evolving mode, we perform the same process that was mentioned in [STP08b]. In terms of DTA and STA we incrementally measure reconstruction error (residual) and then retrieve the top 5% highest errors. Regarding WTA we proceed the similar strategy with this difference, that the reconstruction error is considered for
the starting item in the window.

Once we retrieve the top 5% anomalies in each dimension we compute the accuracy of the methods in prediction of the top-5% reference anomalies obtained by the Tucker optimum model. We define accuracy as the number of true predictions divided by the total number of reference anomalies. However, if we rely on the only single point accuracy, our assessment would be highly dependent on the anomaly border point we choose. Therefore, we opt to use the average accuracy. It means that if \( M \) is the 5% of the size of the mode, we compute the accuracy for Top-M, Top-(M-1), ..., and Top-1 and then compute the mean of them. Suppose that by applying one method, Top-1, Top-2 and Top-3 anomalies be detected with accuracy of 0, 0.5 and 0.66. If we rely only on Top-1, we come up with this conclusion that the detector fails. Even if we lean to Top-3 we may conclude that the detector has succeeded, while neither of them are indeed fair. The average accuracy in this case is \((0+0.5+0.66)/3=0.39\). As we see, the averaging strategy fines the detector for not detecting the Top-1 but not much sever as the point-based evaluation. Moreover, it does not reward the detector for high accuracy of Top-3, because has not been such good for Top-1 and Top-2.

### 7.4 Result and discussion

In this section we report the result of the evaluation process explained in previous section and then discuss the results.

<table>
<thead>
<tr>
<th>Data set</th>
<th>MASTA s=100%</th>
<th>OHTA s=50%</th>
<th>DTA s=10%</th>
<th>STA t nt</th>
<th>STA s=10% t nt</th>
<th>STA s=50% t nt</th>
<th>STA s=100% t nt</th>
<th>WTA w=4</th>
<th>WTA w=7</th>
<th>WTA w=10</th>
</tr>
</thead>
<tbody>
<tr>
<td>trade</td>
<td>60 34</td>
<td>37 61</td>
<td>0 72</td>
<td>0 70</td>
<td>0 70</td>
<td>0 71</td>
<td>6 2</td>
<td>0 9</td>
<td>0 9</td>
<td></td>
</tr>
<tr>
<td>eeg</td>
<td>40 15</td>
<td>31 14</td>
<td>17 51</td>
<td>17 46</td>
<td>14 45</td>
<td>14 28</td>
<td>24 47</td>
<td>18 44</td>
<td>18 43</td>
<td></td>
</tr>
<tr>
<td>newmexico</td>
<td>0 50</td>
<td>0 50</td>
<td>0 100</td>
<td>0 50</td>
<td>0 50</td>
<td>0 13</td>
<td>0 0</td>
<td>0 0</td>
<td>0 13</td>
<td></td>
</tr>
<tr>
<td>spain02</td>
<td>12 67</td>
<td>48 72</td>
<td>0 46</td>
<td>0 96</td>
<td>0 45</td>
<td>0 95</td>
<td>0 65</td>
<td>0 65</td>
<td>1 69</td>
<td></td>
</tr>
<tr>
<td>walk</td>
<td>0 25</td>
<td>0 20</td>
<td>28 92</td>
<td>28 34</td>
<td>28 21</td>
<td>28 13</td>
<td>0 5</td>
<td>0 6</td>
<td>0 6</td>
<td></td>
</tr>
<tr>
<td>flightdelay</td>
<td>81 64</td>
<td>81 59</td>
<td>0 85</td>
<td>0 83</td>
<td>0 79</td>
<td>0 52</td>
<td>0 13</td>
<td>0 1</td>
<td>0 16</td>
<td></td>
</tr>
<tr>
<td>flight</td>
<td>0 36</td>
<td>0 33</td>
<td>28 87</td>
<td>31 73</td>
<td>31 65</td>
<td>23 50</td>
<td>0 0</td>
<td>0 0</td>
<td>0 2</td>
<td></td>
</tr>
<tr>
<td>bikeboston</td>
<td>11 64</td>
<td>6 64</td>
<td>12 84</td>
<td>13 82</td>
<td>13 84</td>
<td>13 89</td>
<td>4 0</td>
<td>1 0</td>
<td>0 0</td>
<td></td>
</tr>
<tr>
<td>bikewashington</td>
<td>23 60</td>
<td>0 59</td>
<td>56 90</td>
<td>57 97</td>
<td>49 92</td>
<td>18 39</td>
<td>9 1</td>
<td>5 18</td>
<td>10 2</td>
<td></td>
</tr>
<tr>
<td>taxi</td>
<td>0 41</td>
<td>0 43</td>
<td>48 62</td>
<td>13 56</td>
<td>13 56</td>
<td>13 60</td>
<td>17 62</td>
<td>0 63</td>
<td>0 62</td>
<td></td>
</tr>
<tr>
<td>kojimagirls</td>
<td>2 50</td>
<td>36 100</td>
<td>0 100</td>
<td>0 50</td>
<td>0 100</td>
<td>0 50</td>
<td>21 50</td>
<td>21 50</td>
<td>21 50</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.3: Average accuracy in prediction of top-5% reference anomalies. The columns identified with "t" correspond to the temporal (or sample) mode and columns with "nt" represent the average accuracy for non-temporal modes.
Throughout this section when we use the term accuracy we refer the average accuracy defined in the previous section. Table 7.3 demonstrates the accuracy of each method in detecting the top 5% reference anomalies for each data set. ITA algorithms increment over the time mode, therefore for a fair comparison we separate the evolving mode from non-evolving modes. For this reason, the first columns of Table 7.3 represent the average accuracy for evolving mode (i.e. time or sample modes) and the second columns represent the mean of average accuracy for other non-evolving modes.

7.4.1 Correctness of the evaluation methodology

According to [STP+08b, STF06, SPP06] we expect that DTA has the better accuracy than its approximation counterpart, STA. Moreover, we expect that STA behaves similarly as DTA with no sampling percentage, but loses accuracy when sampling percentage increases. All these anticipations are satisfied and re-confirmed via obtained results in Table 7.3 as well. This is an evidence that our evaluation framework has been effective in assessment of tensor analysis techniques.

7.4.2 Evaluation on non-evolving modes

In the following subsection we discuss the results corresponding to non-evolving modes ("nt" in Table 7.3).

7.4.2.1 ITA algorithms

From Table 7.3 we can observe that the best accuracy is obtained via DTA. Out of 11 data sets, DTA presents accuracy over 50% in 10 data sets and in average 80% accuracy for all data sets. STA with 100% sampling rate also presents accuracy of greater than 50% in 9 data sets. By increasing the sampling rate to 50% we do not see a considerable accuracy loss comparing the 100% sampling rate. However, increasing sampling rate to 10% results in STA failure in two further data sets (newmexico and bikewashington).

WTA is not compared against DTA and STA in the original paper [STP+08b], so Table 7.3 can be considered the first official comparison of these approaches. As we observe, WTA irrespective of window size except three data sets (spain02, taxi, kojimagirls) fails in the rest of data sets. Even increasing or decreasing the window size does
not affect the performance significantly. In some data sets, accuracy gets better by increasing window size and in some declines. However, the low performance of WTA probably relates to the fact that WTA processes tensor in higher scales and assumes that data does not contain important fluctuations inside the window. Therefore, an anomaly that occurs in one day appears normal in a 10-day window. One can infer that WTA does not have enough sensitivity to small fluctuations occurrences in the original temporal scale.

7.4.2.2 MASTA

MASTA, has been able to present an accuracy of more than 50% in six data sets which is a good result for a full approximation technique. Matching Table 7.3 with data sets characteristics in Table 7.2 shows that MASTA is vulnerable in dealing with three kinds of tensors: very dense, very sparse and high-variation tensors. Among the five data sets that MASTA has a low performance on, two sets, i.e. eeg and walk are the two most dense tensors (100% dense) and one (taxi) is the most sparse tensor (99.8% sparse). Two other data sets, flight and trade are also the top-2 in terms of the standard deviation magnitude. The best performance of MASTA is seen on bike data sets (bikeboston and bikewashington) and spain02 which has a reasonable sparsity and variance. The good performance of MASTA on the two bike data sets reveals that MASTA is robust within the domain.

MASTA is weak in facing with very dense tensors because of two reasons, first it is not based on the correlation concept, therefore it loses the existing strong correlations in dense tensors. It also is not an ideal technique for very sparse tensors, because, histograms for these tensors are not informative enough to be used for matching the slices, resulting in misleading judgments. MASTA is also unable to approximate the tensor with high amount of variance, because of the inherent limitation of streaming histogram calculation. As is mentioned in [BHTT10] when data distribution is highly skewed, the accuracy of histogram approximation technique decays. In high variance data sets, such property appears in its extreme way. This is the reason why we see that the offline approach (OHTA) outperforms MASTA up to 30% in terms of trade data set. However, the same justification does not hold for the flight data set. Probably it is because flight data set has a mixture of two factors. Out of 11 data sets, flight data set is the second sparse tensor with the biggest standard deviation. It seems that OHTA has the same vulnerability as MASTA when both of these factors get involved.
7.4.2.3 Online vs. offline histogram calculation

The results in Table 7.3 reveal that OHTA also has a similar performance as MASTA with slight differences in some datasets. The severe difference of these approaches is in some data sets such as trade which can be explained by the weakness of online histogram technique in handling tensors with large dispersion. This reveals that the online histogram strategy has been quite effective, so we lose only 5-6% average accuracy, in exchange of huge efficiency.

7.4.2.4 Effect of approximation on the accuracy

To study the effect of approximation level on the accuracy, we can compare DTA against STA with three different sampling rates. STA is an approach for faster approximation of DTA. Therefore, we can study this effect if we increase the level of approximation in STA. Considering the Table 7.3 along with Table 7.2 shows that that dense data sets such as eeg and walk are very sensitive to approximation. The accuracy for these data sets decays by increasing the level of approximation. For instance, in walk data set, while DTA has accuracy of 92%, the performance of STA sharply decays to 34%, 21% and 13% respectively for sampling rates of 100%, 50% and 10%. Our histogram-based methods which in principle are approximation solutions also present a very low performance for these data sets. Therefore, we can infer that very dense tensors are more vulnerable than the other tensors against approximation solutions.

7.4.3 Evaluation of the evolving mode

Concerning the temporal mode (as is demonstrated in Table 7.3 with "t") we observe that all methods have difficulty in dealing with fluctuations in the time mode. The reason why accuracy for evolving mode is lower than non-evolving mode for all methods is that time or sample mode is the main cause of tensor fluctuations in the evolving data sets. Temporal analysis with lack of knowledge about the system past behavior results in information loss. This is an important issue that requires to be carefully taken into account when we exploit incremental approaches. Incremental approaches focus more on accuracy of non-temporal mode and sacrifice the information of time mode.

Among ITA algorithms, WTA is of the approaches that sacrifices more time infor-
mation than others. It processes a group of tensors in multiple time points together, which results in less sensitivity to low-scale variations, and subsequently much more accuracy loss.

It seems that histogram-based approaches even though do not provide ideal solution for time mode, perform slightly better than ITA algorithms. For instance, MASTA in two of data sets including flight delay and trade provides respectively high accuracy of 81% and 60% while all ITA algorithms have zero accuracy. This reveals that histograms are better tools for keeping the summarized information in time mode than the model residuals in ITA methods.

7.4.4 Sensitivity Analysis

MASTA and OHTA both require two parameters b1 and b2 which are respectively the number of bins in reference/slice histograms and distance histogram. In this section we study the sensitivity of them to these parameters. To do so, we select a range of b1 from 20 to 80 and b2 from 4 to 20; apply MASTA and OHTA with varying parameters; and then compute the mean accuracy over all modes averaged for all data sets.

In order to evaluate the sensitivity, we perform a standard ANOVA test [MMM84] on the obtained averaged accuracies. Out of 34 tests (Ten 3D tensor plus one 4D tensor) and choosing \( \alpha = 0.05 \), the null hypothesis "accuracies drawn from the same distribution" is rejected, respectively in 12 and 8 of the tests for MASTA and OHTA indicating that both methods have a moderate sensitivity to b1 parameter.

![Figure 7.4: Sensitivity of MASTA and OHTA to the input parameters](image)

(a) b1  
(b) b2

Regarding the b2 parameter, out of 34 tests only in one case, the null hypothesis is rejected for both MASTA and OHTA. This reveals that neither of both methods are
sensitive to \( b_2 \) parameter for a range of 4 to 20.

The sensitivity test results presented here are valid only for anomaly detection application. In some other application such as clustering, the parameter \( b_2 \) is equivalent to the number of clusters and the proper determination of that is quite important in the model output. Sensitivity evaluation of this parameter does not make sense in such applications.

Nevertheless, for specific application of anomaly detection, Figure 7.4 might be helpful while choosing the optimum parameters. We can observe that OHTA is more robust to the input parameters and its performance remains relatively constant by changing the parameters. However, fluctuations of MASTA show that perhaps parameters \( b_1 = 30 \) and \( b_2 = 18 \) lead to a slightly better detection accuracy.

### 7.4.5 Computational complexity

Table 7.4 summarizes the complexity of methods for a dense cubic tensor of \( n \times n \times n \). For more simplicity, we demonstrate the dominant cost for all methods and omit all the constants such as input parameters. In the following we present the computational complexities of the proposed methods.

#### 7.4.5.1 OHTA

For a \( n \times n \times n \) tensor the main cost for OHTA relates to the construction of the reference histogram which exploits the sorting algorithm. If we use for instance quicksort algorithm, OHTA would require \( (n^3)^2 = n^6 \) time and \( n^3 \) space for computing the reference histogram. The construction of slice histograms also costs \( 3n(n^2)^2 = 3n^5 \) time and \( n^2 \) space with additional time of \( 3n(b_1^2) \) for EMD distance calculations. Moreover, the calculation of distance histogram \( H_i \) for three modes consumes \( 3n^2 \) time and \( 3n \) space. Thus, the total time complexity of OHTA is \( n^6 + 3n^5 + 3n^2 + 3n(b_1^2) \). Besides, OHTA requires keeping the intermediate data in the memory which adds additional cost of \( n^3 \) to the space complexity. Hence, the total space complexity is equal to \( 2n^3 + n^2 + 3n \).
7.4.5.2 MASTA

MASTA uses the streaming histogram approximation, therefore, for a full tensor $n \times n \times n$ requires $O(n^3)$ time and $O(1)$ space for reference histogram construction. Three more $O(n^3)$ time is required for construction of the slice histograms in each mode, which makes the total time complexity $O(4n^3)$. For keeping the reference and slices histogram we need a space of $(b1)(3n + 1)$. We also do not need any space for keeping intermediate data, because every piece of tensor stream can immediately enter to MASTA for updating histograms and then get removed from the memory. Therefore, no space cost is imposed for keeping intermediate data. Hence, total space complexity of MASTA is what we need for keeping histograms which after removing the constants becomes $O(n)$.

For updating the MASTA model we need $s$ (data chunk size) time for the reference histogram and $3s$ (3 modes) for updating or adding slice histograms. Therefore, update procedure costs $O(4s)$. Thus, MASTA update procedure is not dependent of $n$ and is linear with data chunk size. This is an enormous advantage over ITA algorithms which at least require $O(n^2)$ for updating the model.

One of the interesting properties of MASTA is that the process of model update is separated from the model output procedure. If the user asks for model output, the input of the updated model feeds into Algorithm 7.3 for the final output. This process needs $3nb1^2$ time for slice reconstruction in Algorithm 7.4 and an additional $3nb1^2$ for histogram distance calculation. Finally, distance histograms need to be updated, that costs further $3n$. Therefore, after removing the constants, the time complexity of output procedure becomes $O(n)$. This $O(n)$ computation is not mandatory for the learning part. We only need to perform this procedure when the user asks for the model output.

7.4.6 Empirical efficiency

The empirical assessment for runtime is executed on a PC with Intel Core 2 Duo, 2000 MHz. The memory consumption test is also performed on a computer with 8GB of memory. We pick 50 and 5 respectively, for $b1$ and $b2$ parameters in OHTA and MASTA. We also choose 5 for the number of components in PARAFAC. For Tucker and WTA the model parameter is chosen (5 5 5) and for DTA and STA the parameters are set as (5 5). The window size of 10 is also selected for WTA.
For assessment of runtime, we create five random cubic dense tensors \( n \times n \times n \), \( n \in \{10, 50, 150, 250, 350\} \) and compute the runtime required for generating the model. The result is presented in Fig. 7.5-a. As we can see, MASTA performs slightly faster than all ITA variants.

In order to evaluate the memory consumption, we create random cubic dense tensors \( n \times n \times n \), \( n \in 1000 \times \{1, 2.5, 5, 7.5, 10\} \) and measure the required memory for model generation by four methods including MASTA, STA/DTA and WTA. Fig. 7.5-b shows the result. As we can see, for processing the large dense tensor of \( 10k \times 10k \times 10k \), MASTA requires only 24 MB of memory that is a tremendous improvement over DTA/STA and WTA that respectively use 2.24 GB and 7.46 GB. Roughly speaking, we can say that for three-order cubic tensors, MASTA can solve 100x bigger problems comparing ITA approaches.

<table>
<thead>
<tr>
<th>Type</th>
<th>Method</th>
<th>Time</th>
<th>Update time</th>
<th>Space</th>
<th>Estimation for ( n=10^4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offline</td>
<td>Tucker/PARAFAC</td>
<td>( n^4 )</td>
<td>—</td>
<td>( n^3 )</td>
<td>7453 GB</td>
</tr>
<tr>
<td>ITA</td>
<td>DTA</td>
<td>( n^4 )</td>
<td>( n^2 )</td>
<td>( n^2 )</td>
<td>2.24 GB</td>
</tr>
<tr>
<td></td>
<td>STA</td>
<td>( n^3 )</td>
<td>( n^2 )</td>
<td>( n^2 )</td>
<td>2.24 GB</td>
</tr>
<tr>
<td></td>
<td>WTA</td>
<td>( n^3 )</td>
<td>( n^2 )</td>
<td>( n^2 )</td>
<td>2.46 GB</td>
</tr>
<tr>
<td>Histogram-based</td>
<td>OHTA</td>
<td>( n^6 )</td>
<td>—</td>
<td>( n^3 )</td>
<td>7442 GB</td>
</tr>
<tr>
<td></td>
<td>MASTA</td>
<td>( n^3 )</td>
<td>( s )</td>
<td>( n )</td>
<td>24 MB</td>
</tr>
</tbody>
</table>

Table 7.4: Time and space complexity of tensor analysis approaches for third-order tensor of \( X(n \times n \times n) \) with zero sparsity. All constants are omitted for simplicity. \( s \) denotes the size of the recent data chunk.

![Figure 7.5](image_url)

(a) Runtime (b) Memory consumption

Figure 7.5: Efficiency comparison of the proposed multi-aspect-streaming method (MASTA) against streaming methods for processing of three-order dense cubic \( n \times n \times n \) tensor.
7.5 Case studies

Any kind of interpretation without having access to a precious domain knowledge may result in a misleading conclusion. We cautiously consider two event detection case studies related to two data sets: trade and walk. These two case studies correspond to one success and one failure case of MASTA in handling evolving mode. As is specified in the column "t" of Table 7.3, MASTA provides 60% accuracy for trade set (success) and 0% accuracy for walk data set (failure). The reason why we choose these two data sets is that there exists an evident knowledge about these data sets that can be incorporated for interpretation of results. In the former case we have the knowledge of global financial crisis that happened in 2007-2009 and for the latter we have the visual insight from the video.

7.5.1 Event detection in trade data set

The trade data set contains historical pairwise trade volumes between 207 countries over 140 years. In this case study, we want to see how methods detect the global financial crisis happened in 2007-2009 [APRR09]. Our expectation is that tensor analysis models rank these years as the most anomalous years. So, in this case study, we apply all methods, including Tucker and PARAFAC reference models on the trade tensor data to see how they rank the crisis period of 2007-2009. We then use the same parameters used in section 7.3.2.4. For STA we test the sampling rate of 50% and for WTA we examine the window size of 10.

<table>
<thead>
<tr>
<th>Year</th>
<th>Tucker</th>
<th>PARAFAC</th>
<th>MASTA</th>
<th>OHTA</th>
<th>DTA</th>
<th>STA (s=50%)</th>
<th>WTA (w=10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2007</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>23</td>
<td>73</td>
<td>131</td>
</tr>
<tr>
<td>2008</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>19</td>
<td>66</td>
<td>132</td>
</tr>
<tr>
<td>2009</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>13</td>
<td>18</td>
<td>67</td>
<td>133</td>
</tr>
</tbody>
</table>

Table 7.5: The anomalous ranking of the period related to global financial crisis 2007-2009 in trade data set.

The obtained rankings are presented in Table 7.5. As we can see, Tucker and MASTA rank the crisis years in their top-4 priority. PARAFAC with a little difference ranks the crisis years in its top-6. OHTA ranks the crisis years in the top-13 with similar orders as MASTA. Also, DTA, STA and WTA identify these years, respectively in their top-23, top-67 and top-133.
The rankings we obtain in this study have a meaningful correlation with the reported accuracies in Table 7.3. For instance, from Table 7.3 we anticipate that MASTA and OHTA present a better accuracy for evolving mode (respectively 60% and 37%) in comparison with other methods. Here, similarly we observe that rankings of MASTA and OHTA are close to the reference models. In particular, between MASTA and OHTA, the former one ranks the crisis period better than the latter which makes sense according to its finer accuracy. This can be interpreted as an evidence for validity of reported accuracies in Table 7.3.

### 7.5.2 Event detection in video data set

![Output of MASTA and OHTA on the time mode (3rd mode) of walk tensor $x \times y \times time$. t=1) video record started. t=2) human object starts to enter to scene while he is walking and moving in different directions. t=3) position of object at the end of record.](image)

Figure 7.6: Output of MASTA and OHTA on the time mode (3rd mode) of walk tensor $x \times y \times time$. t=1) video record started. t=2) human object starts to enter to scene while he is walking and moving in different directions. t=3) position of object at the end of record.

Here we analyze the walk data set which includes the appearance of a human object entering the scene while walking. The background of the video is not static, so that includes some small-scale movements in the back.

We transform the color image frames to grayscale and build a three-order tensor of $x \times y \times frame (or \ time)$. Then we apply MASTA and OHTA with $b1=50$ and $b2=3$ on the video tensor to see how they detect the entrance of the human object. The three
important moments in the video and the output of MASTA and OHTA on the evolving mode are illustrated in Fig. 7.6. As we can see, OHTA discriminates the frames very close to the reality happens in the video. It clusters the instants into three relevant moments: a) frames 1-24 that object still has not entered into the scene; b) frames 25-29 which are related to when the object is entering but not still appeared fully in the scene; and c) frames 30 to 50 that correspond to moments when the object has fully entered but still moving in the scene.

MASTA, though is able to detect the entrance event at t=25, is too sensitive to the small-scale changes. In certain moments before object enters the scene, some small-scale changes are being seen in the background. MASTA assumes that these small movements belongs to a new emerging event, so it allocates a new bin for them (t=13 and t=15).

The other difficulty of MASTA relates to, when the object turns around in different directions. MASTA faces with this doubt that maybe those movements belong to a new concept, so creates a new bin for justification of the new movements (e.g. t=27 or t=34-36). However, it is unsure that how to classify the object movements, thus between t=25 to t=50 repeats its mistake several times. The reason of such behavior is that MASTA does not have free access to the historical data, as opposed to its offline counterpart. Hence, it is natural for it to react faster to the small-scale changes.

One interesting point is that either MASTA or OHTA present meaningful outputs which was not expected, according to their zero average accuracy in Table 7.3. The reason is that our evaluation framework in this study is based on quality of anomaly detection and not quality of clusters. Therefore, MASTA and OHTA even though might have less value for anomaly detection on dense tensors, might have potential in applications such as clustering and change detection. Evaluation of MASTA in these applications requires future research and new evaluation methodologies which was out of the scope of this work.

7.6 Conclusion

We study the application of histograms to tensor analysis. We introduce two histogram-based algorithms, namely OHTA and MASTA respectively, for offline and streaming analysis. The streaming solution not only presents a close accuracy to the offline approach, but also reduces the total time complexity from $O(n^6)$ to $O(n^3)$ and space
complexity from $O(n^3)$ to $O(n)$. More importantly, it allows tensor evolution through all modes, which is a major progress. It also has three superiorities over the state-of-the-art incremental tensor analysis techniques: Firstly, MASTA has a significant lower time complexity for updating the model when new data arrives: $O(s)$ vs. $O(n^2)$; Secondly, it is robust against the well-known problem of intermediate data explosion; and finally, it consumes much less space: $O(n)$ vs $O(n^2)$, such that is capable to solve bigger problems.

MASTA, however, suffers from two main issues: First, is not a tensor decomposition approach, hence its application is limited to the analysis-only tasks such as anomaly detection, clustering or change detection; Second, its accuracy is not ideal as other approximation solution such as DTA. Its best accuracy is 67% for non-evolving mode and 81% for evolving mode. As for the last, it has a poor performance in handling some kind of tensors including very dense, very sparse and tensors with large amount of dispersion.

Figure 7.7: A guideline for choosing the appropriate tensor analysis solution
We recommend MASTA as the last alternative solution for three situations (see Fig. 7.7): 1) when scalable tensor decomposition solutions are not applicable due to the shortages in equipments and infrastructures; 2) when tensor size changes over time and the model needs to be updated rapidly and frequently; and 3) when size of tensor is large and is not affordable by ITA. In such cases, MASTA can offer a better solution at least finer than WTA or comparable to STA with 5-10% sampling rate.

Future works include evaluation of MASTA on other problems such as clustering and concept drift detection; investigation of MASTA problems in particular settings such as very dense or very sparse tensors; and development of parallel version of MASTA.
Chapter 8

Conclusion and Future Works

In this thesis we focused on event detection problem using tensor-based techniques. We reviewed the existing methods and solutions and identified the open issues and challenges. We addressed some of these issues and developed new methods and algorithms for solving them. We compared our methods against the state-of-the-art solutions and gained improvement in either accuracy or efficiency. We used tensors as the main tools for this purpose, but for solving sub-problems such as hotspot detection and baseline data labeling we explored other tools such as matrix decomposition and classical machine learning algorithms. Conclusions and feature works are discussed in each chapter separately. In this chapter we outline the most important contributions and point out the directions of future research. The findings are specified with () and future works are shown with (>).

8.1 EigenEvent

In Chapter 3 we proposed an algorithm called EigenEvent for event detection from multidimensional data streams. EigenEvent for the first time uses tensor and matrix decomposition along with seasonal segmentation technique for real-time monitoring of syndromic surveillance data streams. The main contributions of this chapter findings are as follows.

- We reduced the high false alarm rate of existing solutions for syndromic surveillance. Besides, our approach is faster than state-of-the-art methods such as WSARE. Our experimental results shows that EigenEvent is three times faster
than WSARE 2.0, six times faster than WSARE 2.5 and fifty times faster than WSARE 3.0.

- Combination of principal singular vector and leading singular value performs better than each separately.

- Dynamic baseline creation with taking into account seasonal effects leads to much better performance than static approaches in semi-supervised tensor-based event detection.

Future research work includes evaluating EigenEvent in a real syndromic surveillance setting and extending the realistic multivariate simulator [LSY07] for spatiotemporal tensors.

8.2 EigenSpot

In Chapter 4 we proposed a new algorithm called EigenSpot for spatiotemporal hotspot detection which is the first eigenspace solution for this problem. EigenSpot can be utilized alone or along with EigenEvent (chapter 3) in a complete event detection scenario (Alarming via EigenEvent and subgroup discovery via EigenSpot). We exploited low-rank SVD and pairwise singular vectors element matching for hotspot detection. Our findings in this chapter can be listed as follows.

- The existing solution, STScan, can be vulnerable if its main assumptions such as data distribution, hotspot shape and data quality are violated. In some particular settings, Eigenspot can be more accurate and faster than STScan.

- Eigenspot can efficiently approximate the real spatiotemporal hotspot similar to an expensive method like STScan.

- Both EigenSpot and STScan are dependent on the hotspot size. However, EigenSpot exhibits a more stable behavior against changes in hotspot size and impact.

- The assumption of EigenSpot is that there is only one hotspot in data which is its current limitation. Extension of EigenSpot for multiple hotspots needs to be investigated as a future work.
8.3 Event Labeling System

In Chapter 5 we proposed a new event labeling system based on ensemble detectors and background knowledge. The methodology we proposed in this event labeling system can be used for automatically generating high quality baseline data for EigenEvent algorithm (chapter 3). Our findings in this chapter are as follows.

- Ensemble of event detectors with various properties on different time scales performs better than individual detectors. Adding background knowledge to the ensemble system also leads to additional performance improvement.

- Computer-based knowledge sources like world wide web can be potential sources for aiding event and anomaly detection systems.

- Bike sharing data is interesting for research. This is because of three reasons: 1) environment information is available (e.g. weather in daily and hourly scales), and holidays, etc; 2) external knowledge source such as world wide web (text, videos, Images) can uncover what actually occurs in the system; 3) there exists an access to human specialist to confirm the results.

Future work includes testing the event labeling system for tensor data and developing a tensor-based ensemble architecture. For instance, using different family of tensor decomposition models such as Tucker, PARAFAC, ICA, LPP, Bayesian in a unified model.

8.4 HTM

In Chapter 6 we proposed a hybrid tensor model for event detection from traffic tensors. We showed that insensitivity problem of naive tensor models to topological fluctuations can be treated if we keep, in parallel, a topology tensor along with the flow tensor. We also studied the tensor automatic rank estimation problem and compared a recent method with the existing solution and showed the new solution results in improvement of traffic tensor model. The major contributions in this chapter are as follows.

- In traffic tensors, separation of topology makes tensor models more sensitive to topological fluctuations, and therefore leads to a better detection power than naive tensor models.
Tensor-based models are more robust methods for event detection in comparison with matrix residual-based methods.

Adjustable core size Tucker decomposition method presented in [CLZ14] is a potential method for tensor models. It provides better accuracy in comparison to the traditional approach of DIFFIT+Tucker-ALS.

Feature work can be the real-time extension of HTM. However, this is a difficult challenge, because, our findings in this research reveal that incremental approaches like DTA are not reliable for diverse event types. Hence, for solving this problem, other possibilities should be taken into account.

8.5 MASTA

In chapter 7 we proposed new approach for multi-aspect-streaming tensor analysis with applications to anomaly and event detection. We showed that our approach can handle 100x bigger problems comparing state-of-the-art. Our main findings in Chapter 7 can be outlined as follows.

We extended the application of histograms to tensor analysis problem and proposed the first approach for multi-aspect-streaming tensor analysis (MASTA) based on online histograms.

MASTA is space-efficient and fast with constant-time update ability. We identified the strengths and weaknesses of MASTA on eleven real-life data sets from seven various domains.

MASTA has a competitive value against existing solutions in three situations: 1) when scalable tensor decomposition solutions are not applicable due to the shortages in equipments and infrastructures; 2) when tensor size changes over time and the model needs to be updated rapidly and frequently; and 3) when size of tensor is large and is not affordable by incremental approaches.

Evaluation of MASTA on other problems such as clustering and concept drift detection is one of the important future research. The difficulties of MASTA in particular settings such as very dense or very sparse tensors are also required to be investigated. The parallel extension of MASTA will also be helpful when the runtime speed matters.
8.6 Future trends

During our continuously literature review we noticed two powerful but less known tensor approaches, including LPP-based [HCN05, DY06] and DEDICOM [Har78] that have quite a potential for event detection. DEDICOM, for instance, is very potential for analysis of data in social networks and traffic data. LPP-based approaches also are very helpful for video and spatial data since they preserve the geometric structures in the data. Specially, recently proposed method, TGLPP [LBGY14b] seems a promising method for tensor-based event detection, since it captures both local and global structure in data. Bayesian approaches are also emerging techniques with huge potential for anomaly detection. However, their appeal is limited due to their high computational costs. Fortunately, some new scalable methods have been proposed to deal this issue (e.g. [RWG+14]) and it is anticipated that we witness more works in this area in upcoming years.

We identified some important issues in tensor-based event detection and suggested the possible solutions for each category, according to the state-of-the-art. One of the important issues that has not received sufficient attention is tensor rank estimation. We could not find any work studying the effect of tensor rank determination on the quality of event detection or comparison of different automatic tensor rank estimation methods in accuracy of anomalies. Another problem about this issue is that the majority of tensor rank estimation methods are computationally expensive and hence infeasible for automatic purposes. Perhaps, the work of [CLZ14] is the most efficient method of the currently existing solutions, which we tested its application for event detection in Chapter 6. However, still a need for a fast, accurate and adaptive method for tensor rank estimation is deeply felt.

Scalability, which is a quite important problem has received enough attention and is almost a hot topic in tensor literature. On the other side, it seems that less quantity of research is devoted to the adaptivity issue which is as important as the scalability. After the work of [STF06] we have not witnessed a serious contribution from this kind in the literature. Seasonality issue is also less noted in the tensor-based event detection literature. Our work in Chapter 7 was an effort for this gap, but still an accurate and fast method would be great of interest.

In many phenomena we have the prior knowledge of seasonality that can be incorporated for more accurate event detection. Unfortunately, the seasonality
issue is ignored in tensor modeling of human generated data in many works. Our work in Chapter 3 was an effort for responding this issue. But still the literature is poor in handling issues related to seasonality in tensor-based event detection.

Data fusion based tensor approaches [Aca15] is also predicted to be the hot topic in near future due to the increasing number of heterogeneous data sources in modern digital systems. It is anticipated that development in this area directly influences the quality of tensor-based event detection techniques.
References


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES

[CHH06] Deng Cai, Xiaofei He, and Jiawei Han. Learning with tensor representation. Technical Report UIUCDCS-R-2006-2716, Dept. of Computer Science, University of Illinois, 2006.


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


[KMCS08] Abbas Khosravi, Joaquim Melendez, Joan Colomer, and Jorge Sanchez. Multiway principal component analysis (mpca) for upstream/downstream classification of voltage sags gathered in distri-


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


Juan Carlos SanMiguel, José M. Martinez, and Álvaro García. An ontology for event detection and its application in surveillance video. In *Proceedings of the 2009 Sixth IEEE International Conference on*.
REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


REFERENCES


