BREADTH ANALYSIS OF ONLINE SOCIAL NETWORKS

by

Luis F. Chiroque

A dissertation submitted by in partial fulfillment of the requirements for the degree of Doctor of Philosophy in

Mathematical Engineering

UNIVERSIDAD CARLOS III DE MADRID

Advisor(s):
Antonio Fernández Anta

Tutor:
Ángel Sánchez Sánchez

September 2021

To those who are
(but specially those who are not anymore)
around us.

Acknowledgements

This work completes a big important chapter in my life. Not even in my best dreams I could imagine I pursue a Ph.D.

It has been a long trip since I started working at IMDEA Networks, back in 2012. Now, it is hard to see myself on that young boy who started this trip. I can only say, we really enjoy every single adventure along the way.

I think a Ph.D. in sciences needs 4 fundamental skills: (i) theoretical knowledge, (ii) practical knowledge, (iii) communications and (iv) writing, and the Ph.D. student needs to dominate (to master, desirably) the 4 of them. Of course, you also need motivation, a background in science, novel ideas, constancy, etc., but I assume you do not start a Ph.D. without those skills. Then, the journey of a Ph.D. student should take into account these 4 skills, and some of them you have to learn them by yourself. Luckily, this it not a journey you made by yourself alone, of course not. I am happy to having had so many great and professional people around me during all these years that help me, encourage me, support me, teach me. I am specially grateful to Antonio and Anxo, as my thesis director and tutor, respectively. Thanks for all your atenttion, patience and advice. My gratitude for José Luis, who I have to acknowledge the fact that I started this long trip. Thanks to my first colleagues at IMDEA Newtorks, when I started working as Research Engineer in the SOCAM Project: Dr. Agustín Santos, Dr. Luis López, Raúl, Radu, Jorge, Hector and Rafa. Thanks to my Ph.D. student mates (most of them already doctors) from IMDEA and from other institutions I had the pleasure to meet: Philippe, Elli, Foivos, Jordi, Lin, Christian, Roderick, Aymen, Roberto, Dario, Ander, Maurizio, Carlo, Demetris, Oché, Nacho, Jorge, Juan Carlos, Sergio, Evgenia, Edgar, Patricia, Ignacio, Arash, Sim, Quim, Juan Camilo, Qing, Syed, Angelos, Nicola, Hany, Guillermo, Carlos, Yonas. Thanks to all IMDEA Networks stuff, who are great professionals; special mention to Dr. José Felix, Rebeca, Brian, Flora, Ana, Dr. Vincenzo Mancuso, Dr. Sergey Gorinsky, Dr. Arturo Azcorra, Dr. Paolo Casari, Dr. Nikolas Nikolaou. Thanks to all great UC3M professors I have got the chance to work with: Dr. Rubén Cuevas, Dra. Rosa E. Lillo, Dr. Henry Laniado, Dr. Juan Romo. Thanks to all great researchers/professors I have got the chance to meet and discuss with: Dr. Christopher Thraves, Dr. George Pallis, Dr. Marios D. Dikaiakos, Dr. Eduardo Castelló, Dr. Esteban Moro, Dr. Souneil Park, Dra. Nuria Oliver, Dr. Pablo Rodriguez. Thanks to my best friends, Rubén and Borja who, somehow, worked at IMDEA Networks also, at least for short time. Thanks to my sister Noelia and to my

parents for their inifinite love and belief. Thanks to my wife and sons for their unconditional love, understanding, patience and support. Thanks to my family. Thanks to those who were at the begining and passed away during this long trip. Thank you all. This would not have been possible without you.

This work completes a big important chapter in my life. Just wait for the next one(s).

Contents published

The materials from the following sources are included in the thesis. Their inclusion is not indicated by typographical means or references, since they are fully embedded in each chapter indicated below.

Paper 1 Anta, A. F., Chiroque, L. N., Morere, P., & Santos, A. (2013). Sentiment analysis and topic detection of Spanish tweets: A comparative study of of NLP techniques. *Procesamiento del lenguaje natural*, 50, 45-52.

- Included in Chapter 2 (FULL)
- url: `http://journal.sepln.org/sepln/ojs/ojs/index.php/pln/article/download/4658/2760`

Paper 2 Cordobés, H., Fernández Anta, A., Chiroque, L. F., Pérez, F., Redondo, T., & Santos, A. (2014). Graph-based techniques for topic classification of tweets in Spanish. *International Journal of Interactive Multimedia and Artificial Intelligence* (IJIMAI), 2(5), 31-37.

- Included in Chapter 3 (FULL)
- url: `http://dx.doi.org/10.9781/ijimai.2014.254`

Paper 3 Azcorra, A., Chiroque, L. F., Cuevas, R., Anta, A. F., Laniado, H., Lillo, R. E., Romo J. & Sguera, C. (2018). Unsupervised scalable statistical method for identifying influential users in online social networks. *Scientific reports*, 8(1), 1-7.

- Included in Chapter 4 (FULL)
- url: `https://doi.org/10.1038/s41598-018-24874-2`

Paper 4 Trihinas, D., Chiroque, L. F., Pallis, G., Anta, A. F., & Dikaiakos, M. D. (2018, July). ATMoN: Adapting the" Temporality" in Large-Scale Dynamic Networks. In *2018 IEEE 38th International Conference on Distributed Computing Systems* (ICDCS) (pp. 400-410). IEEE.

- Included in Chapter 5 (FULL)
- url: `https://doi.org/10.1109/ICDCS.2018.00047`

Further Research Achievments

Papers

- López-Presa, J. L., Chiroque, L. F., & Fernández Anta, A. (2014). Novel techniques to speed up the computation of the automorphism group of a graph. *Journal of Applied Mathematics*, 2014. https://doi.org/10.1155/2014/934637

Software

- MUOD.outliers - R Package in GitHub. Author. https://github.com/luisfo/muod.outliers

Abstract

This thesis is mainly motivated by the analysis, understanding, and prediction of human behaviour by means of the study of their digital fingeprints. Unlike a classical PhD thesis, where you choose a topic and go further on a deep analysis on a research topic, we carried out a breadth analysis on the research topic of complex networks, such as those that humans create themselves with their relationships and interactions. These kinds of digital communities where humans interact and create relationships are commonly called *Online Social Networks*. Then, (i) we have collected their interactions, as text messages they share among each other, in order to analyze the sentiment and topic of such messages. We have basically applied the state-of-the-art techniques for Natural Language Processing, widely developed and tested on English texts, in a collection of Spanish Tweets and we compare the results. Next, (ii) we focused on Topic Detection, creating our own classifier and applying it to the former Tweets dataset. The breakthroughs are two: our classifier relies on text-graphs from the input text and we achieved a figure of 70% accuracy, outperforming previous results. After that, (iii) we moved to analyze the network structure (or topology) and their data values to detect outliers. We hypothesize that in social networks there is a large mass of users that behaves similarly, while a reduced set of them behave in a different way. However, specially among this last group, we try to separate those with high activity, or low activity, or any other paramater/feature that make them belong to different kind of outliers. We aim to detect influential users in one of these outliers set. We propose a new unsupervised method, Massive Unsupervised Outlier Detection (MUOD), labeling the outliers detected os of shape, magnitude, amplitude or combination of those. We applied this method to a subset of roughly 400 million Google+ users, identifying and discriminating automatically sets of outlier users. Finally, (iv) we find interesting to address the monitorization of real complex networks. We created a framework to dynamically adapt the temporality of large-scale dynamic networks, reducing compute overhead by at least 76%, data volume by 60% and overall cloud costs by at least 54%, while always maintaining accuracy above 88%.

Table of Contents

Acknowledgements VII

Contents Published IX

Further Research Achievments XI

Abstract XIII

Table of Contents XV

List of Tables XX

List of Figures XXIII

1. Introduction 1

1.1. Graph Theory Genesis: The Seven Bridges of Könisberg 1

1.2. Zachary's Karate Club 2

1.3. Internet, Social Media and Online Social Networks 3

1.4. Publications and Conclusions 4

1.4.1. Further work 5

2. Paper 1: Sentiment Analysis and Topic Detection of Spanish Tweets: A Comparative Study of NLP Techniques 9

2.1. Introduction 10

2.1.1. Related Work 10

2.1.2. Contributions 12

2.2. Methodology 13

2.2.1. Attributes Definition and Preprocessing 13

2.2.2. Classification Methods 16

2.3. Experimental Results 16

2.3.1. Data Sets and Experiments Configurations 16

2.3.2. Topic Estimation Results 17

2.3.3. Sentiment Estimation Results 19
2.4. Conclusions 19

3. Paper 2: Graph-based Techniques for Topic Classification of Tweets in Spanish 25
3.1. Introduction 26
3.2. State of the Art 26
3.3. Basic Graph-based Classification Techniques 27
3.4. Implementing the Classifier 29
3.4.1. Preprocessing of the Text 29
3.4.2. Reference Graphs 30
3.4.3. Text Classification 31
3.5. Results and Discussion 33

4. Paper 3: Unsupervised Scalable Statistical Method for Identifying Influential Users in Online Social Networks 39
4.1. Introduction 40
4.2. Contributions 41
4.2.1. Identification of Influential Users in OSNs 42
4.2.2. Testing MUOD in Google+ 44
4.3. Conclusions 47

Appendices 51
Appendix 4.A. Introduction 51
Appendix 4.B. A New Outlier Detection Method Based on Indices of Magnitude, Amplitude and Shape 54
Appendix 4.C. A New Outlier Detection Method: Implementation 57
4.C.1. *Shape* Indices 59
4.C.2. *Magnitude* & *Amplitude* Indices 59
4.C.3. Three indices simultaneously 59
Appendix 4.D. Outlier Detection in Functional Data 60
Appendix 4.E. Simulation Study 64
4.E.1. Mixture Models 1, 2, and 3 64
4.E.2. Mixture Model 4 67
4.E.3. Mixture Models 5, 6, and 7 74
Appendix 4.F. Outlier Detection in our Real Data Application 76

5. Paper 4: ATMoN: Adapting the "Temporality" in Large-Scale Dynamic Networks 89
5.1. Introduction 90
5.2. Related Work 92
5.3. Problem Statement 93

5.4. The ATMoN Library 95
5.5. Temporal Granularity Adaptation 96
5.6. Evaluation 103
5.6.1. Adaptive Technique Estimation Accuracy 104
5.6.2. Adaptive Technique Efficiency and Overhead 105
5.7. Conclusions and Future Work 107

6. Conclusions and Future Work **111**
6.1. Paper 1: Sentiment Analysis and Topic Detection of Spanish Tweets: A Comparative Study of NLP Techniques 111
6.2. Paper 2: Graph-based Techniques for Topic Classification of Tweets in Spanish . 112
6.3. Paper 3: Unsupervised Scalable Statistical Method for Identifying Influential Users in Online Social Networks 112
6.4. Paper 4: ATMoN: Adapting the "Temporality" in Large-Scale Dynamic Networks 114

List of Tables

2.1. Detail of Configuration 2 of topic detection with Complement Naive Bayes. . . . 17

2.2. Detail of Configuration 13 of sentiment analysis with Naive Bayes Multinomial. . 18

3.1. Results from the experiments 33

3.2. Tweets per topic in TASS2013 corpus 34

3.3. Results per topic 34

4.E.1. Correct outlier detection percentages (c), false outlier detection percentages (f), F-measures (F) and F-measure-based rankings of the methods (r) in mixture models 1, 2 and 3 which allow for magnitude (mag), amplitude (amp) and shape (sha) outliers, respectively. 67

4.E.2. Correct outlier detection percentages (c), false outlier detection percentages (f), F-measures (F) and F-measure-based rankings of the methods (r) in the mixture model 4 which allows simultaneously for magnitude, amplitude and shape outliers. 68

4.E.3. Decomposed correct outlier detection percentages (c), false outlier detection percentages (f), F-measures (F) and F-measure-based rankings of the methods (r) in mixture model 4 allowing simultaneously for magnitude (mag), amplitude (amp) and shape (sha) outliers. 69

4.E.4. Information about FBPLOT, MUOD$_{mag}$, MUOD$_{amp}$ and MUOD$_{sha}$ when used on a data set with 6,000,000 observations generated by mixture model 4. 70

4.E.5. Correct outlier detection percentages (c), false outlier detection percentages (f) and F-measures (F) in the mixture model 4 which allows simultaneously for magnitude, amplitude and shape outliers. Performances of FBPLOT, MUOD, MUOD$_{mag}$, MUOD$_{amp}$ and MUOD$_{sha}$ on a data set with 6,000,000 observations. 71

4.E.6. Decomposed correct outlier detection percentages (c), false outlier detection percentages (f) and F-measures (F) in mixture model 4 allowing simultaneously for magnitude (mag), amplitude (amp) and shape (sha) outliers. Performances of FBPLOT, MUOD, MUOD$_{mag}$, MUOD$_{amp}$ and MUOD$_{sha}$ on a data set with 6,000,000 observations. 71

4.E.7. Correct outlier detection percentages (c), false outlier detection percentages (f), F-measures (F) and F-measure-based rankings of the methods (r) in mixture models 5, 6 and 7. 75
4.F.1. Dataset variables description. 77
4.F.2. Information about FBPLOT, MUOD$_{mag}$, MUOD$_{amp}$ and MUOD$_{sha}$ when used on our real data application. 77
4.F.3. Distribution of the observations of our real data application in the classes *no MUOD*, *only sha*, *only mag+sha*, *only amp+sha* and *mag+amp+sha*. Absolute (abs) and relative (rel) frequencies. 78
4.F.4. Distribution of the observations of our real data application in the classes *no MUOD*, *only sha*, *only mag+sha*, *only amp+sha* and *mag+amp+sha* against the classes *no FBPLOT* and *yes FBPLOT*. Absolute frequencies. 79
4.F.5. Distribution of the observations of our real data application in the classes*no MUOD*, *only sha*, *only mag+sha*, *only amp+sha* and *mag+amp+sha* against the classes *no FBPLOT* and *yes FBPLOT*. Relative frequencies inside the classes *no MUOD*, *only sha*, *only mag+sha*, *only amp+sha* and *mag+amp+sha*. 80

5.1. Datasets Used to Compare the Frameworks Under Evaluation 98
5.2. Data Volume Reduction (%) in Respect to Max Inaccuracy (η) 106
5.3. MAPE (%) in Respect to Max Inaccuracy (η) 106

List of Figures

1.1. *CC BY-SA 3.0 as in ´Wikipedia, 2021.* The Bridges of Könisberg on the left and its abstract simplification on the right, made by vertices and connections, or edges. 2

1.2. *As originally used in ´Dhaou et al., 2017.* Visualizing the social relationships among the 34 individuals in the karate club, grouped by colour according to their final decision. 3

2.1. Accuracy (%) of different configurations for topic detection in the small data set. 16

2.2. Accuracy (%) of different configurations for sentiment analysis in the small data set. 18

3.1. **Reference graph building process.** From a set of tweets belonging to a specific topic, a set of per-sentence graph is constructed. The reference graph is obtained joining each of these graphs. 28

3.2. **Text classification.** Similarities among the text graph and reference graphs are calculated in order to classify a text. 29

4.1. Example of representation of users' characteristics in the form of a signal (A); Example of signals associated to magnitude outliers (B); Example of signals associated to amplitude outliers (C); Example of signals associated to shape outliers (D). 42

4.2. Illustration of the criterion to determine which users are flagged as shape outliers by MUOD. The horizontal x axis represents sample percentiles based on the shape index. The vertical y axis represents shape index values. 44

4.3. Venn's diagram that describes the relationship between the different sets of outliers identified by MUOD (and the FBPLOT algorithm). 46

4.4. FBPLOT outliers, MUOD outliers (four types) and sample of non-outlying users: parallel coordinates representation of their (log) medians. 47

4.5. Disease propagation simulations for the different outlier classes. Each line is the result of 10 SI (susceptible-infected) simulations using the centroid user/node of each outlier class as infection root. The simulations were carried out using the largest connected component of the network of followers (around 170M nodes) and an infection rate of 0.2. 48
4.A.1 Illustrative Example 1. 53
4.A.2 A modification of Example 1 allowing magnitude outliers. 53
4.A.3 A modification of Example 1 allowing amplitude outliers. 53
4.A.4 A modification of Example 1 allowing shape outliers. 54
4.A.5 A modification of Example 1 allowing magnitude, amplitude and shape outliers. . 54
4.B.1 Scatter plot of the $I_S(x, \mathcal{X})$-based ranks (horizontal axis) and the $I_S(x, \mathcal{X})$ values (vertical axis) for the curves from Figure 4.A.4. 55
4.B.2 Scatter plot of the $I_M(x, \mathcal{X})$-based ranks (horizontal axis) and the $I_M(x, \mathcal{X})$ values (vertical axis) for the curves from Figure 4.A.2. 56
4.B.3 Scatter plot of the $I_A(x, \mathcal{X})$-based ranks (horizontal axis) and the $I_A(x, \mathcal{X})$ values (vertical axis) for the curves from Figure 4.A.3. 57
4.B.4 Tangent method applied to shape indices for our real dataset. The horizontal axis represents sample percentiles based on the shape index. The vertical axis represents shape index values. The green dot cuts the x-axis at $x^* = 0.9546$, yielding a 4.538% of outliers (the higher values at the right-side). $I^*(\mathcal{X}) = I(0.9546, \mathcal{X})$. 58
4.D.1 Time performance for the different algorithms computing different size of data (log-log scale). FBPLOThas the best performance by far as we increase the data size. MUOD and MUOD 10-way are the second best performance lines. Observe that, FBPLOTand MUOD are the only ones performing in less than 100 seconds for a data size of $3 \cdot 10^4$ users. FBAGand FHDRhave a similiar behaviour as the data size grows and performs similar to MUOD, but they are not able to process a data size of $3 \cdot 10^5$. OutGRAM have a similar behaviour than the former algorithms but is not able to process dataset bigger than $3 \cdot 10^4$ users. Then we found the KSFD family and the B family with a poor performance and a dataset size limit of 10^3 users. 63
4.E.1 Simulated observations. 64
4.E.2 Mixture model 1: magnitude outliers. 65
4.E.3 Mixture model 2: amplitude outliers. 65
4.E.4 Mixture model 3: shape outliers. 66
4.E.5 Mixture model 4: magnitude, amplitude and shape outliers. 68
4.E.6 Venn's diagram for outliers and algorithm's detected sets. This plot visually summarizes the results in Tables 4.E.4 and 4.E.5. 71

4.E.7.Accuracy for the considered algorithms when sampling. The plot shows average accuracy for 100 simulations using random samples of 100K observations. Each bar shows the accuracy as the intersection ratio between the outliers detected on a sample and the outliers on the total population (6M observations) for an algorithm. The bars show that shape, magnitude and amplitude are not affected when sampling (accuracy $\sim 90\%$) while FBPlot seems to be sensitive to observation samples. 72
4.E.8.Precision and recall for FBPlot against shape, magnitude and amplitude when sampling. The plot shows average precision and recall for 100 simulations using random samples of 100K observations. Each bar shows that there is a considerable overlap between FBPlot and shape, magnitude and amplitude, yet in different ways. While FBPlot and shape overlap in each sample, FBPlot becomes a small subset of shape. On the other hand, FBPlot overlaps with magnitude and amplitude too, but in this case, magnitude and amplitude become small subsets of FBPlot. To sum up, these plots extend Figures 4.E.6 and 4.E.7, and show the FBPlot behavior as comparison with MUOD (shape, magnitude, and amplitude) when sampling. 73
4.E.9.Mixture models 5 (top), 6 (bottom left) and 7 (bottom right). 75
4.F.1.1% *only sha*, *only mag+sha*, *only amp+sha* and *mag+amp+sha* observations (from top left, in clockwise order). 79
4.F.2.5000 *no MUOD* observations. 80
4.F.3.1% *(only sha, no FBPLOT)* observations (left); 25% *(only sha, yes FBPLOT)* observations (right). Dashed line in both figures: upper bound of the non-outlying region defined by FBPLOT. 81
4.F.4.*(only mag+sha, no FBPLOT)* observations (left); *(only mag+sha, yes FBPLOT)* observations (right). Dashed line in both figures: upper bound of the non-outlying region defined by *FBPLOT*. 82
4.F.5.*(mag+amp+sha, no FBPLOT)* observations (left); *(mag+amp+sha, yes FBPLOT)* observations (right). Dashed line in both figures: upper bound of the non-outlying region defined by FBPLOT. 83

5.1. Graph Metric and Topology Structure Volatility for a Mobile Network with Fixed Temporal Granularity 91
5.1. High-Level Abstract Overview of the ATMoN Framework 95
5.1. The Metric Streams from the Datasets Used to Compare the Frameworks Under Evaluation 101
5.2. Dataset Graph Topology Structure Volatility based on the D-measure: $D(G_i, D_{i-1}) \in [0, 1]$ 102
5.1. Accuracy, Overhead and Cost Comparison of the Techniques Under Evaluation . 104

Chapter 1

Introduction

The human behaviour has been studied for thousands of years from different disciplines (e.g., philosophy, anthropology, psychology, economics, etc.) for different purposes and it is usually aggregated into the field of *sociology*. Since we are social beings, we tend to create extensive groups named societies. Understanding human behaviour might help us to take better collective decisions and protect us from external threats.

Then, early in the 20th century new studies specially focused on human relationships form a different new perspective. Investigating human social structures at different scales using networks and graph theory makes us see ourselves as a complex (social) system to study; Social Network Analysis was born as a field within sociology.

We want to make a journey through Social Network Analysis, from the first studies of social relationships to modern Online Social Networks.

1.1. Graph Theory Genesis: The Seven Bridges of Könisberg

The great Leonhard Eurler laid the foundations of Graph Theory, in 1736, when he faced a historical problem of the city of Könisberg in Prussia (now Kaliningrade, Russia). It really seemed a notable mathematical problem, and its resolution also prefigured the idea of topology. Könisberg was set on both sides of the Pregel River, with two large islands between both sides, which were connected to each other and to the mainland portions of the city by seven bridges.

The problem was stated as follows. *Is it possible to walk through the city crossing each of those bridges once and only once?* The answer is "NO". Euler proved it not to be possible, i.e., there is no solution to such question. As we can observe in Figure 1.1, Euler first pointed out that the path followed within each land mass is irrelevant. Thus, it is simplified as a *vertex* or node. Then, the only important feature of any route is the sequence of bridges crossed. So, each bridge is represented as an abstract connection, and *edge*, which connected two land masses (vertices). The resulting structure is a graph. In a mathematical way, we denote a graph $G = (V, E)$ as the combination of the set of n vertices $V = \{v_1, ..., v_n\}$ and their relationships $E = \{(v_i, v_j), ...\}$.

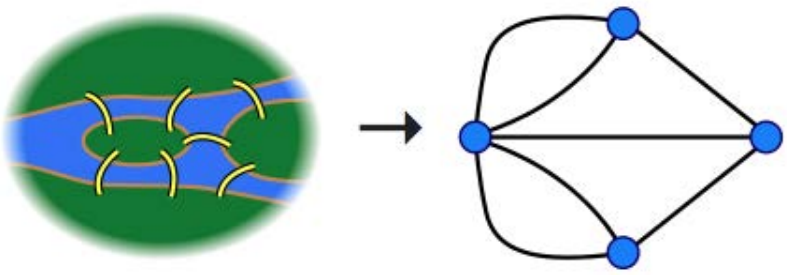


Fig. 1.1: *CC BY-SA 3.0 as in Wikipedia, 2021.* The Bridges of Könisberg on the left and its abstract simplification on the right, made by vertices and connections, or edges.

Euler solved the question observing that, during any walk in the graph, the number of times one enters in a non-departure/non-terminal vertex must be even. Thus, the number of edges from every land mass must to be even, except for those chosen as start and end. Observe that this denotes the concept of vertex degree. To conclude, in our Könisberg bridges graph, all the vertices have an odd number of edges, resulting impossible to follow the initial proposition of walking through the city traversing each bridge only once.

1.2. Zachary's Karate Club

Since Euler laid the foundations of graph theory, this mathematical field has been extended for hundreds of years and its uses are multiple. Now, we are going to introduce a famous graph from 1970, which was popularized by Newman in 2002 (Girvan and Newman, 2002), as an example of community structure. Wayne W. Zachary was studing the social network of a karate club for a period of three years (1970-72) (Zachary, 1977). The club was formed by 34 individuals and Zachary captured their relationships as edges for those who interacted outside the club. During the study there was a conflict between the administrator and the instructor, leading to a split of two clubs. Half of the members formed a new club with their original instructor. The remaining found another instructor or gave up karate. The interesting fact about this is that Zachary was able to correctly predict each member's decision except one (#9). This highlights the importance of the communication flow between individuals and the techniques applied to social networks to obtain insights, resulting in a field known as *Network Science*. Figure 1.2 shows the Karate Club's network, coloured by their final decision after the split.

This is a first example of applying a classic concept of clustering to a network. Then, we can start applying statistical techniques adapted for networks. This is very useful to understand human behaviour and make predictions in case of epidemics, spreding disease, adoption of new products (marketing), etc. What is more, we could enhance the network information adding weights/distance or probabilities to the relationships in a network. Networks offer us a great tool to model social communities to study and understand the human being. Last but not least, we could even study the evolution of a network over time, as a network stream, modeled as a list of

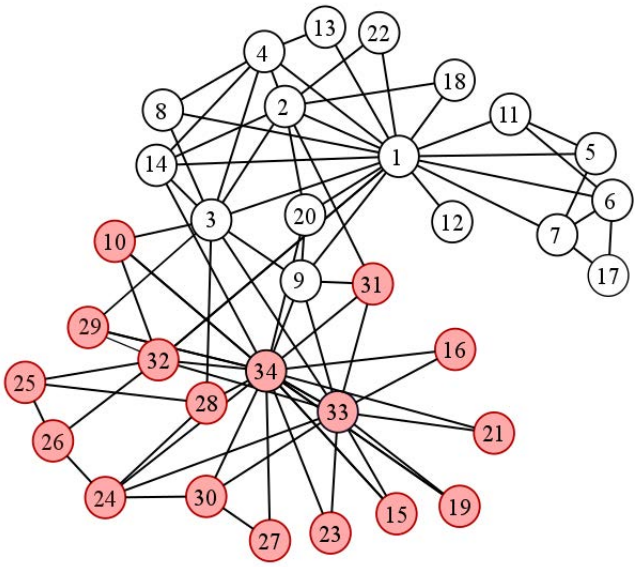


Fig. 1.2: *As originally used in Dhaou et al., 2017.* Visualizing the social relationships among the 34 individuals in the karate club, grouped by colour according to their final decision.

graphs at every time step. The options are multiple.

1.3. Internet, Social Media and Online Social Networks

There is a before and an after in the society with the creation of the internet. It has not only changed the speed at which we communicate with each other but also the way we do it. We have instant information of any natural disaster happening at the other side of the world and we can contact our loved ones to check they are safe. We have instant access to the latest news around the globe as soon as someone upload them. Observe we say 'someone', not exclusively big media groups anymore. That's the power of the internet. Anyone can upload their photos, videos or opinions on the internet to be accessed by anyone. First it was the personal websites, but it quickly evolved to social networks. There are websites where people can connect with each other, anonymously, or create a digital profile to make a wall of content to be shared with anybody else within the social network. Famous examples of social networks nowadays are Twitter, Instagram, Facebook, Whatsapp. Each one provides a different layout of connectivity. One is more focused on sharing short text messages, one more in pictures, one is a mixture, and one is focused on private chats. Online Social Networks (OSN's) has been very popular since the begining of the internet, but the arrival of smartphones make them more than popular: ubiquitous. They hosted the concept of *influencer*, as a Social Network celebrity, and it is believed these individuals can create and change trends within the network. Thus, they are usually paid with marketing purposes to promote new products, and it awakes the eagerness of normal people to became an *influencer*. OSN's have also changed the formal definition of networks since the relationships are not recip-

rocal anymore. Now, the set of edges E of the network became *directed*. This means, an edge $e \in E = (v_i, v_j) \neq (v_j, v_i)$. Thus, influencers accumulate a large amount of people who are subscribed to the content they publish, but they are not necessarily subscribed to them.

1.4. Publications and Conclusions

In this dissertation we explore Online Social Networks in breadth.

Paper 1 *Sentiment Analysis and Topic Detection of Spanish Tweets: A Comparative Study of NLP Techniques*

This dissertation journey starts analyzing the content the users generate and publish in Twitter. Specially, we are going to focus on the text messages they share either with their followers or just open to the world. The wealth of information gathered here is huge and the possibilities are wide. We can perform simple statistics about the most frequent letters, the most used words or even more complex analyses such as the sentiment of a sentence/message, classifying the text topic among a given set of topics or even detecting what are they talking about. Thus, it is when we can use Natural Language Process (NLP) techniques, and here we are going to focus on Sentiment Analysis and Topic Classification.

A significant amount of effort is been invested in constructing effective solutions for sentiment analysis and topic classification, but mostly for English texts. Leveraging our participation on a workshop on the SEPLN 2012 (it stands for Natural Language Processing Spanish Society, in Spanish), we use a corpus of Spanish tweets and present a comparative analysis of different approaches and classification techniques for these problems.

Paper 2 *Graph-based Techniques for Topic Classification of Tweets in Spanish*

Linking with the previous paper, now we are going to take Topic Classification one step further. Topic classification of texts is one of the most interesting challenges in Natural Language Processing (NLP). Back in 2013 (when this work was published), topic classifiers commonly use a bag-of-words approach, as in Paper 1, in which the classifier uses (and is trained with) selected terms from the input texts. In this paper we present techniques based on graph similarity to classify short texts by topic. In our classifier we build graphs from the input texts, and then use properties of these graphs to classify them. We have tested the resulting algorithm by classifying Twitter messages in Spanish among a predefined set of topics, achieving more than 70% accuracy.

Paper 3 *Unsupervised Scalable Statistical Method for Identifying Influential Users in Online Social Networks*

The next step in our journey through Online Social Networks (OSN's) is related with their network structures, relationships, and data values. Assuming a big mass of the OSN's users behaving indistinctly, we will try to identify those who deflect from the big mass (for better or

worse), if any; namely outliers. We hypothesize that influential users may be in one of the detected outliers groups, and developed and implemented a novel technique to detect them. We make use of functional data analysis to develop our own method, providing good time-memory performance and scalability while dealing with large amount of data (Big Data) and data visualization. For our purpose, we have used a dataset of 400M users from the Google+ Social Network and worked with a sample network of 6M nodes (users).

Paper 4 *ATMoN: Adapting the "Temporality" in Large-Scale Dynamic Networks*

In this , we present a framework which dynamically adapts the sampling rate for a monitored metric based on the current evolution and variability of the graph stream. By accomplishing this, data volume and resource utilization are reduced providing timely approximate answers. Experiments on real-world data from face-to-face, contact, social and vehicular networks show that our framework achieves a balance between efficiency and accuracy. Specifically, our framework is capable of reducing data volume by 59%, so does resource utilization, while preserving a greater than 88% accuracy.

1.4.1. Further work

Besides the work published above, it is relevant to mention another related work performed during the disertation duration. We literally copy their abstracts to provide a global idea of the content.

Master Thesis *New Methods for Ranking Influence in Social Networks*; Chiroque, 2015

In this work, propagation dynamics on social networks are studied in order to identify the most influential users. For this purpose, diffusion data has been collected during 4 weeks from a microblogging OSN (online social network) called Tumblr. Then, the propagation graph has been built and studied using the first 2 weeks data (period T1). Subsequently, this graph has been used to predict the influencers during the last 2 weeks (period T2). A ranking of influential nodes is obtained for T2, set as the ground truth. The aim is to predict this ranking using the data from T1. Based on the average spread of users' posts, rankings obtained with several techniques are tested and compared. These techniques include classical centrality measures used in the literature, the T1 ranking itself, and new alternatives based on effective degree using local (network) information. Whilst all methods perform similarly when considering whole global ranking, differences among them appear when ranking the top influencers. For those, in general, the methods proposed here outperform the classical centrality measures.

Conauto *Novel Techniques to Speed Up the Computation of the Automorphism Group of a Graph*; López-Presa, Chiroque, and Fernández Anta, 2014

Graph automorphism (GA) is a classical problem, in which the objective is to compute the automorphism group of an input graph. Most GA algorithms explore a search tree using the

individualization-refinement procedure. Four novel techniques are proposed which increase the performance of any algorithm of this type by reducing the depth of the search tree and by effectively pruning it. We formally prove that a GA algorithm that uses these techniques correctly computes the automorphism group of an input graph. Then, we describe how these techniques have been incorporated into the GA algorithm conauto, as conauto-2.03, with at most an additive polynomial increase in its asymptotic time complexity. Using a benchmark of different graph families, we have evaluated the impact of these techniques on the size of the search tree, observing a significant reduction both when they are applied individually and when all of them are applied together. This is also reflected in a reduction of the running time, which is substantial for some graph families. Finally, we have compared the search tree size of conauto-2.03 against those of other popular GA algorithms, observing that, in most cases, conauto explores less nodes than these algorithms.

References

Chiroque, Luis F et al. (2015). *New Methods for Ranking Influence in Social Networks*. M.Sc. thesis.

Dhaou, Salma et al. (2017). «The advantage of evidential attributes in social networks». In: pp. 1–8. DOI: 10.23919/ICIF.2017.8009758.

Girvan, Michelle and Mark EJ Newman (2002). «Community structure in social and biological networks». In: *Proceedings of the national academy of sciences* 99.12, pp. 7821–7826.

López-Presa, José Luis, Luis F Chiroque, and Antonio Fernández Anta (2014). «Novel techniques to speed up the computation of the automorphism group of a graph». In: *Journal of Applied Mathematics* 2014.

Wikipedia (2021). *Seven Bridges of Königsberg — Wikipedia, The Free Encyclopedia*. http://en.wikipedia.org/w/index.php?title=Seven%20Bridges%20of%20K%C3%B6nigsberg&oldid=1032247857. [Online; accessed 09-September-2021].

Zachary, Wayne W (1977). «An information flow model for conflict and fission in small groups». In: *Journal of anthropological research* 33.4, pp. 452–473.

Chapter 2

Paper 1: Sentiment Analysis and Topic Detection of Spanish Tweets: A Comparative Study of NLP Techniques

Published in: *Procesamiento del Lenguaje Natural, no. 50, 2013.*

Antonio Fernández Anta[1] **Luis Núñez Chiroque**[1]
Philippe Morere[2(*)] **Agustín Santos**[1]

[1] Institute IMDEA Networks, Madrid, Spain
[2] ENSEIRB-MATMECA, Bordeaux, France
{antonio.fernandez, luisfelipe.nunez, philippe.morere, agustin.santos}@imdea.es

Spanish Abstract Se está invirtiendo mucho esfuerzo en la construcción de soluciones efectivas para el análisis de sentimientos y detección de asunto, pero principalmente para textos en inglés. Usando un corpus de tweets en español, presentamos aquí un análisis comparativo de diversas aproximaciones y técnicas de clasificación para estos problemas.

keywords Sentiment analysis, topic detection.

Abstract A significant amount of effort is been invested in constructing effective solutions for sentiment analysis and topic detection, but mostly for English texts. Using a corpus of Spanish tweets, we present a comparative analysis of different approaches and classification techniques for these problems.

(*)Partially done while at Institute IMDEA Networks.

Acknowledgments Partially funded by Factory Holding Company 25, S.L. The authors want to thank Fernando Pérez for useful discussions.

2.1. Introduction

With the proliferation of online reviews, ratings, recommendations, and other forms of online opinion expression, there is a growing interest in techniques for automatically extracting the information they embody. Two of the problems that have been posed to achieve this are *sentiment analysis* and *topic detection*, which are at the intersection of natural language processing (NLP) and data mining. Research in both problems is very active, and a number of methods and techniques have been proposed in the literature to solve them. Most of these techniques focus on English texts and study large documents. In our work, we are interested in languages different from English and micro-texts. In particular, we are interested in sentiment and topic classification applied to Spanish Twitter micro-blogs. Spanish is increasingly present over the Internet, and Twitter has become a popular method to publish thoughts and information with its own characteristics. For instance, publications in Twitter take the form of *tweets* (i.e., Twitter messages), which are micro-texts with a maximum of 140 characters. In Spanish tweets, it is common to find specific Spanish elements (SMS abbreviations, hashtags, slang). The combination of these two aspects makes this a distinctive research topic, with potentially deep industrial applications.

The motivation of our research is twofold. On the one hand, we would like to know whether usual approaches that have been proved to be effective with English text are also so with Spanish tweets. On the other hand, we would like to identify the best (or at least a good) technique for processing Spanish tweets. For this second question, we would like to evaluate those techniques proposed in the literature, and possibly propose new ad hoc techniques for our specific context. In our study, we try to sketch out a comparative study of several schemes on term weighting, linguistic preprocessing (stemming and lemmatization), term definition (e.g., based on uni-grams or n-grams), the combination of several dictionaries (sentiment, SMS abbreviations, emoticons, spell, etc.) and the use of several classification methods.

2.1.1. Related Work

Sentiment analysis is a challenging NLP problem. Due to its tremendous value for practical applications, it has experienced a lot of attention, and it is perhaps one of the most widely studied topic in the NLP field. Pang and L. Lee, 2008 have a comprehensive survey of sentiment analysis and opinion mining research. Liu, 2010, on his hand, reviews and discusses a wide collection of related works. Although most of the research conducted focuses on English texts, the number of papers on the treatment of other languages is increasing every day. Examples of research papers on Spanish texts are Brooke, Tofiloski, and Taboada, 2009; Martínez-Cámara, Martín-Valdivia, and Ureña-López, 2011; Sidorov et al., 2012.

Most of the algorithms for sentiment analysis and topic detection use a collection of data to train a classifier, which is later used to process the real data. Data is preprocessed before being used in the classifier in order to correct errors and extract the main features. Many different techniques have been proposed for these two phases. For instance, different classification methods have been proposed, like Naive Bayes, Maximum Entropy, Support Vector Machines (SVM), BBR, KNN, or C4.5. In fact, there is no final agreement on which of these classifiers is the best. For instance, Go, Bhayani, and Huang, 2009 report similar accuracies with classifiers based on Naive Bayes, Maximum Entropy, and SVM.

Regarding preprocessing the data, Laboreiro et al., 2010 explore tweets tokenization (or symbol segmentation) as the first key task for text processing. Once single words or terms are available, typical choices are using uni-grams, bi-grams, n-gram, or parts-of-speech (POS) as basic terms. Again, there is no clear conclusion on which is the best option, since Pak and Paroubek, 2010 report the best performance with bi-grams, while Go, Bhayani, and Huang, 2009 present better results with unigrams. The preprocessing phase may also involve word processing the input texts: stemming, spelling and/or semantic analysis. Tweets are usually very short, having emoticons like :) or :-), or abbreviated (SMS) words like *Bss* for *Besos* (*kisses*). Agarwal et al., 2011 propose the use of several dictionaries: an emoticon dictionary and an acronym dictionary. Other preprocessing tasks that have been proposed are contextual spell-checking and name normalization Kukich, 1992.

One important question is whether the algorithms and techniques proposed for other types of data can be directly applied to tweets. Twitter data poses new and different challenges, as discussed by Agarwal et al., 2011 when reviewing some early and recent results on sentiment analysis of Twitter data (e.g., Go, Bhayani, and Huang, 2009; Bermingham and Smeaton, 2010; Pak and Paroubek, 2010). Engström, 2004 has also shown that the bag-of-features approach is topic-dependent and Read, 2005 demonstrated how models are also domain-dependent.

These papers, as expected, use a broad spectrum of tools for the extraction and classification processes. For feature extraction, *FreeLing* Padró et al., 2010 has been proposed, which is a powerful open-source language processing software. We use it as analyzer and for lemmatization. For classification, Justin et al., 2010 report very good results using WEKA, which is one of the most widely used tools for the classification phase. Other authors proposed the use of additional libraries like LibSVM Chang and Lin, 2011.

Most of the references above have to do with sentiment analysis, due to its popularity. However, the problem of topic detection is becoming also popular Sriram et al., 2010, among other reasons, to identify trending topics Allan, 2002; Bermingham and Smeaton, 2010; K. Lee et al., 2011. Due to the the real time nature of Twitter data, most works Mathioudakis and Koudas, 2010; Vakali, Giatsoglou, and Antaris, 2012 are interested in breaking news detection and tracking. They propose methods for the classification of tweets in an open (dynamic) set of topics. Instead, we are interested in a closed (fixed) set of topics. However, we explore all the indexing and clustering techniques proposed, since most of them could also be applied to sentiment

analysis.

2.1.2. Contributions

In this paper we have explored the performance of several preprocessing, feature extraction, and classification methods in a corpus of Spanish tweets, both for sentiment analysis and for topic detection. The different methods considered can be classified into almost orthogonal families, so that a different method can be selected from each family to form a different configuration. In particular, we have explored the following families of methods.

Term definition and counting. In this family it is decided what constitutes a basic term to be considered by the classification algorithm. The different alternatives are using single words (uni-grams), or groups of words (bi-grams, tri-grams, n-grams) as basic terms.

Stemming and lemmatization. One of the main difference between Spanish and English is that the latter is a weakly inflected language in contrast to Spanish, a highly inflected one. One interesting questions is to compare how well the usual stemming and lemmatization processes perform with Spanish words.

Word processing and correction. We have used several dictionaries in order to correct the words and replace emoticons, SMS abbreviations, and slang terms by their meaning in correct Spanish. Finally, it is possible to use a morphological analyzer to determine the type of each word. Thus, a word-type filter can be applied to tweets.

Valence shifters. An alternative to the usual direct term-counting method is the processing of valence shifters and negative words (*not, neither, very, little*, etc). We think that those words are useful for sentiment classification since they change and/or revert the strength of a neighboring term.

Tweet semantics. The above approaches can be improved by processing specific tweet artifacts such as author tags, or hashtags and URLs (links), provided in the text. The author tags act like a history of the tweets of a specific person. Additionaly, the hashtags are a great indicator of the topic of a tweet, whereas retrieving keywords from the web-page linked within a tweet allows to overpass the limit of the 140 characters and thus improves the efficiency of the estimation. Another way to overpass this limit is to investigate the keywords of a tweet in a search-engine to retrieve other words of the same context.

Classification methods. In addition to these variants, we have explored the full spectrum of classification methods provided by WEKA.

The rest of the paper is structured as follows. In Section 2.2 we describe in detail the different techniques that we have implemented or used. In Section 2.3 we describe our evaluation scenario and the results we have obtained. Finally, in Section 2.4 we present some conclusions and open problems.

2.2. Methodology

In this section we give the details of how the different methods considered have been implemented in our system.

2.2.1. Attributes Definition and Preprocessing

n-grams As we mentioned, classifiers will consider sets of n words (n-grams), with unigrams as a special case. The value of n could be defined in our algorithm. When using n-grams, n is a parameter that highly influences performance. We found that, in practice, having n larger than 3 did not improve the results, so we limit n by that value.

Of course, it is possible to combine n-grams with several values of n. The drawback of this is the high number of entries in the final attribute list. Hence, when doing this, a threshold is used to remove all the attributes that appear too few times in the data set, as they are considered as noise. We force that the attribute appears at least 5 times in the data set to be considered. Also, a second threshold is used to remove ambiguous attributes. This threshold has been set to 85%, which means that more than 85% of the occurrences of an attribute have to be for a specific topic or sentiment.

Processing Terms The processing of terms involves first building the list of attributes, which is the list of different terms that appear in the data set of interest. In principle, the data set used to identify attributes is formed at least by all the tweets that are provided as input to the algorithm, but there are cases in which we do not use them. For instance, when using an affective dictionary (see below) we may not use the input data. Moreover, even if the input data is processes, we may filter it and only keep some of it. For instance, we may decide to use only nouns. In summary, the list of attributes is built from the input data (if so decided) preprocessed as determined and, potentially, by additional data (like the affective dictionary). Once this process is completed, the list of attributes and the list of vectors obtained from the tweets are passed to the classifier.

Stemming and Lemmatization When creating the list of attributes, typically only the root of the words is used in the attribute list. The root can take the form of the lemma or the stem of the word (lemmatization or stemming, respectively). We have used the FreeLing software to perform the lemmatization process. The Snowball software stemmer has been used in our experiments. We have decided to always use one of the two processes.

Word Processing and Correction As mentioned above, one of the possible preprocessing steps before extracting attributes and vectors is to correct spelling errors. If correction is done, the algorithm uses the Hunspell dictionary to perform it.

Another optional preprocessing step expands the emoticons, shorthand notations, and slang commonly used in SMS messages which is not understandable by the Hunspell dictionary. The

use of these abbreviations is common in tweets, given the limitation to 140 characters. An SMS dictionary is used to do the preprocessing. It transforms the SMS notations into words understandable by the main dictionary. Also, the emoticons are replaced by words that describe their meaning. For example :-) is replaced by *feliz* (*happy*).

We have observed that the information of a sentence is mainly located in a few keywords. These keywords have a different type according to the information we are interested in. For topic estimation, the keywords are mainly nouns and verbs, whereas for sentiment analysis they are adjectives and verbs. For example, in the sentence *La pelicula es buena* (*The movie is good*), the only word that is carrying the topic information is the noun *pelicula*, which is very specific to the cinema topic. Besides, the word that best reflects the sentiment of the sentence is the adjective *buena*, which is positive. Also, in the sentence *El equipo ganó el partido* (*The team won the match*), the verb *ganó* is carrying information for both topic and sentiment analysis: the verb *ganar* is used very often in the soccer and sport topics, and has a positive sentiment. We allow to filter the words of the input data using their type. The filtering is done using the FreeLing software, which is used to extract the type of each word.

When performing sentiment analysis, we have found useful to have an *affective dictionary*. This dictionary consist of a list of words that have a positive or negative meaning, expanded by their polarity "P" or "N" and their strength "+" or "-". For example, the words *bueno* (*good*) and *malo* (*bad*) are respectively positive and negative with no strength whereas the words *mejor* (*best*) and *peor* (*worse*) are respectively positive and negative with a positive strength. As a first approach, we have not intensively used the polarity and the strength of the affective words in the dictionary. Its use only forces the words that contain it to be added as attributes. This has the advantage of drastically reducing the size of the attribute list, specially if the input data is filtered. Observe that the use of this dictionary for sentiment analysis is very pertinent, since the affective words carry the tweet polarity information. In a more advanced future approach, the characteristics of the words could be used to compute weights. Since not all the words in our affective dictionary may appear in the corpus we have used, we have built *artificial* vectors for the learning machine. There is one artificial vector per sentiment analysis category (positive+, positive, negative, negative+, none), which has been built counting one occurrence of those words whose polarity and strength match with the appropriate category.

Valence Shifters There are two different aspects of valence shifting that are used in our methods. First, we may take into account negations that can invert the sentiment of positive and negative terms in a tweet. Second, we may take weighted words, which are intensifiers or weakeners, into account.

Negations are words that reverse the sentiment of other words. For example, in the sentence *La pelicula* ***no*** *es buena* (*The movie is* ***not*** *good*), the word *buena* is positive whereas it should be negative because of the negation *no*. The way we process negations is as follows. Whenever a negative word is found, the sign of the 3 terms that follow it is reversed. This allows us to

differentiate a positive *buena* from a negative *buena*. The area of effect of the negation is restricted to avoid false negative words in more sophisticated sentences.

Other valence shifters are words that change the degree of the expressed sentiment. Examples of these are, for instance *muy* (*very*), which increases the degree, or *poco* (*little*), which decreases it. If the valence shifter is positive, the weight is multiplied by 3, while if it is negative by 0.5.

Twitter Artifacts It has been noticed that with the previous methods, not all the potential data contained in the tweets is used. There are several frequent element in tweets that carry a significant amount of information. Among others we have the following:
Hashtags(any word which starts with "#"). They are used for identify messages about the same topic since some of them may carry more topic information than the rest of the tweet. For example, if a tweet contains `#BAR`, which is the hashtag of the Barcelona soccer team, it can almost doubtlessly be classified in a soccer tweet.
References (a "@" followed by the username of the referenced user). References are interesting because some users appear more frequently in certain topics and will more likely tweet about them. A similar behaviour can be found for sentiment.
Links (a URL). Because of the character limitation of the tweets, users often include URLs of webpages where more details about the message can be found. This may help obtaining more context, specially for topic detection.

In our algorithms, we have the possibility of including hashtags and references as attributes. We believe that these options are just a complement to previous methods and cannot be used alone, because we have found that the number of hashtags and references in the tweets is too small. We also provide the possibility of adding to the terms of a tweet the terms obtained from the web pages linked from the tweet. A first approach could have been retrieving the whole source code of the linked page, get all the terms it contains, and keep the ones that match the attribute list. Unfortunately, there are too many terms, and the menus in the pages induce an unexpected noise which degrades the results. The approach we have chosen is to keep only the keywords of the pages. We chose to retrieve only the text within the HTML tags `h1`, `h2`, `h3` and `title`. The results with this second method are much better since the keywords are directly related to the topic.

Because of the short length of the tweets, our estimations often suffer from a lack of words. We found a solution to this problem in several papers Banerjee, Ramanathan, and Gupta, 2007; Gabrilovich and Markovitch, 2005; Rahimtoroghi and Shakery, 2011 that use web sources (like Wikipedia or the Open Directory) to complete tweets. The web is a mine of information and search-engines can be used to retrieve it. We have used this technique to obtain many keywords and a context from just a few words taken from the tweets. For implementation reasons, Bing was chosen for the process. The title and description of the 10 first results of the search are kept and processed in the same way as the words of the tweet. We found out that we have better results by searching in Bing with only the nouns contained in the tweet; therefore, this is the option we

Configuration number	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Parameters														
n-gram	1	1	1	1	1	1	1	2	1	1	1	1	2	1
Lemma/Stem (**L**/**S**)	L	L	S	L	L	L	L	L	L	L	L	L	L	L
SMS					X						X		X	
Word types (Nouns C&P)	X		X	X	X	X	X	X	X	X			X	X
Correct words				X										
Hashtags	X	X	X	X	X		X	X	X	X	X	X		X
Author Tags	X	X	X	X	X	X		X	X		X	X	X	X
Links									X	X				
Classifiers (Accuracy)														
Ibk	36.62	30.54	36.37	36.62	36.77	31.17	**37.97**	32.64	**38.57**	32.47	30.49	30.54	33.83	36.62
ComplementNaiveBayes	56.75	**58.45**	56.25	56.75	**57**	55.75	53.66	53.56	53.56	51.67	**58.25**	**58.45**	52.02	56.75
NaiveBayesMultinomial	56.35	**57.1**	55.61	56.35	56.25	55.46	53.71	55.61	54.11	53.26	**56.95**	**57.1**	56	56.35
RandomCommittee	53.56	52.47	52.62	53.56	**53.91**	**53.66**	52.52	**55.06**	52.72	52.27	51.92	52.47	38.15	53.56
SMO	56.3	55.06	55.95	56.3	56.55	55.51	55.26	55.9	55.16	54.21	42.38	55.06	54.81	56.3

Fig. 2.1: Accuracy (%) of different configurations for topic detection in the small data set.

chose.

2.2.2. Classification Methods

For classification, we use WEKA[(**)], which is a collection of machine learning algorithms that can be used for classification and clustering. It includes algorithms for classification, regression, clustering attribute selection and association rule mining. Almost all popular classification algorithms are included (Bayesian methods, decision tree learners, random trees and forests, etc.).

For each experiment we set up a configuration that tells our algorithm which attributes to choose and how to create vectors of attributes. The output of this algorithm is a WEKA file for a specific configuration and the input data. Once this file is available, we are able to run all the available classification algorithms that WEKA provides. However, due to space limit we will below concentrate on only a few.

2.3. Experimental Results

2.3.1. Data Sets and Experiments Configurations

In our experiments we have used a corpus of tweets provided for the TASS workshop at the SEPLN 2012 conference as input data set. This set contains about 70,000 tweets. Additionaly, over 7,000 of the tweets were given as a small training set with both topic and sentiment classification. The data set was shuffled for the topics and sentiments to be randomly distributed. Due to the large time taken by the experiments with the large data set, most of the experiments presented have used the small data set, using 5,000 tweets for training and 2,000 for evaluation.

For the TASS workshop we tested multiple configurations with all the WEKA classifiers to choose the one with the highest accuracy. Different configurations gave the best results for

[(**)]http://www.cs.waikato.ac.nz/ml/weka, accessed August 2012.

Precision	Recall	F-Measure	Class
0.468	0.619	0.533	música
0.316	0.318	0.317	economía
0.565	0.503	0.532	entretenimiento
0.721	0.814	0.765	política
0.386	0.354	0.37	cine
0.175	0.241	0.203	literatura
0.551	0.442	0.491	otros
0.194	0.162	0.176	tecnología
0.419	0.5	0.456	deportes
0.5	0.409	0.45	fútbol
0.579	0.584	0.578	Weighted Avg.

Table 2.1: Detail of Configuration 2 of topic detection with Complement Naive Bayes.

sentiment analysis and topic detection. As described, our initial approach was to compare every possible configuration and all classification methods of WEKA. Unfortunately, it was unfeasible to execute all possible configurations with all possible classification methods. Hence, we made some decisions to limit the number of experiments.

In this paper, we have chosen to present only five classification algorithms from those provided by WEKA. In particular, we have chosen the methods Ibk, Complement Naive Bayes, Naive Bayes Multinomial, Random Committee, and SMO. This set tries to cover the most popular classification techniques. Then, we have chosen for each of the two problems (topic and sentiment) a basic configuration similar to the one submitted to the TASS workshop. Starting from this basic configuration a sequence of derived configurations are tested. In each derived configuration, one of the parameters of the basic configuration was changed, in order to explore the effect of that parameter in the performance. Finally, for each classification method a new configuration is created and tested with the parameter settings that maximized the accuracy.

The accuracy values computed in each of the configurations with the five methods with the small data set are presented in Figures 2.1 and 2.2. In both figures, Configuration 1 is the basic configuration. The derived configurations are numbered 2 to 9. (Observe that each accuracy value that improves over the accuracy with the basic configuration is shown on boldface.) Finally, the last 5 configurations of each figure correspond to the parameters settings that gave highest accuracy in the prior configurations for a method (in the order Ibk, Complement Naive Bayes, Naive Bayes Multinomial, Random Committee, and SMO).

2.3.2. Topic Estimation Results

As mentioned, Figure 2.1 presents the accuracy results for topic detection on the small data set, under the basic configuration (Configuration 1), configurations derived from this one by toggling one by one every parameter (Configurations 2 to 9), and the seemingly best parameter settings for each classification method (Configurations 10 to 14). Observe that no configuration uses a search engine. This is because we found that the ARFF file generated iafter searching

Configuration number	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Parameters														
N-gram	1	1	1	1	1	1	1	1	2	2	1	1	2	1
Lemma/Stem (**L**/**S**)	L	L	L	S	L	L	L	L	L	L	L	S	S	L
Affective dictionary	X		X	X	X	X	X	X	X	X	X			X
SMS	X	X	X	X	X	X		X	X		X	X		X
Word types (Adj, Verb)	X	X		X	X	X	X	X	X	X	X			
Correct words					X								X	
Weight						X					X	X		
Negation	X	X	X	X	X	X	X		X	X	X	X		X
Classifiers (Accuracy)														
Ibk	31.32	31.32	29.78	31.32	31.32	31.32	**32.47**	31.32	**31.52**	**32.47**	31.32	28.78	29.08	29.78
ComplementNaiveBayes	30.18	29.88	17.93	28.74	30.13	**30.23**	28.49	30.18	28.74	28.49	**30.23**	16.88	**39.49**	17.93
NaiveBayesMultinomial	32.82	**32.97**	**32.97**	**33.37**	32.77	**32.87**	32.52	32.82	32.87	32.52	**32.87**	32.52	**42.38**	**32.97**
RandomCommittee	33.72	**34.16**	**38.24**	**34.61**	**34.31**	33.67	**34.41**	**34.36**	**34.01**	**34.41**	33.67	**38.34**	**38.14**	**38.24**
SMO	39.79	39.64	**41.93**	38.94	39.59	39.6	29.24	39.74	38.3	39.24	39.6	**41.38**	**41.43**	**41.93**

Fig. 2.2: Accuracy (%) of different configurations for sentiment analysis in the small data set.

Precision	Recall	F-Measure	Class
0.368	0.285	0.321	negative+
0.354	0.43	0.389	negative
0.145	0.064	0.089	neutral
0.317	0.14	0.194	positive
0.461	0.715	0.561	positive+
0.525	0.469	0.495	none
0.404	0.424	0.4	Weighted Avg.

Table 2.2: Detail of Configuration 13 of sentiment analysis with Naive Bayes Multinomial.

the web as described above (even for the small data set) was extremely large and the experiment could not be completed

The first fact to be observed in Figure 2.1 is that Configuration 1, which is supposed to be similar to the one submitted to TASS, seems to have a better accuracy with some methods (more than 56% versus 45.24%). However, it must be noted that this accuracy has been computed with the small data set (while the value of 45.24% was obtained with the large one). A second observation is that in the derived configurations there is no parameter that by changing its setting drastically improves the accuracy. This also applies to the rightmost configurations, that combine the best collection of parameter settings. Finally, it can be observed that the largest accuracy is obtained by Configuration 2 with Complement Naive Bayes. This configuration is obtained from the basic one by simply removing the word filter that allows only nouns. Looking more closely at this combination of parameter configuration and method, we can obtain other performance parameters, presented in Table 2.1. The meaning of these can be found in the WEKA documentation. This combination has a 58.45% of correctly classified instances, and a relative absolute error of 54.07%.

2.3.3. Sentiment Estimation Results

Figure 2.2, on its turn, shows the accuracy computed for the basic configuration (Configuration 1), the derived configurations (2 to 9), and the best settings per classification method (10 to 14) for sentiment analysis with the small data set. As before, it can be observed that the accuracy of Configuration 1 with SMO is better than the reported accuracy of the results submitted (39.79% versus 36.04%). It also holds that no parameter seems to make a huge difference. However in this case the combination of parameters seem to have some impact, since the best combination, formed by Configuration 13 and method Naive Bayes Multinomial, has significant better accuracy than any other configuration with the same method. However, other methods (e.g., SMO) has a more homogenous set of values.

As before, we take a closer look at the best combination in Table 2.2. This combination is able to classify correctly 851 instances (and incorrectly 1157), with an accuracy of 42.38%, and relative absolute error of 77.29%.

2.4. Conclusions

We have presented a comprehensive set of experiments classifying Spanish tweets according to sentiment and topic. In these experiments we have evaluated the use of stemmers and lemmatizers, n-grams, word types, negations, valence shifters, link processing, search engines, special Twitter semantics (hashtags), and different classification methods. This collection of techniques represent a thorough study.

The first conclusion of our study is that none of the techniques explored is the silver bullet for Spanish tweet classification. None made a clear difference when introduced in the algorithm. The second conclusion is that tweets are very hard to deal with, mostly due to their brevity and lack of context. The results of our experiments are encouraging though, since they show that it is possible to use classical methods for analyzing Spanish texts. The largest accuracy obtained (58% for topics and 42% for sentiment) are not too far from other values reported in the TASS workshop. However, these values reflect that there is still a lot of room for improvement, justifying further efforts.

References

Agarwal, Apoorv et al. (2011). «Sentiment analysis of Twitter data». In: *Proceedings of the Workshop on Languages in Social Media*. LSM '11. Portland, Oregon: Association for Computational Linguistics, pp. 30–38. ISBN: 978-1-932432-96-1. URL: http://dl.acm.org/citation.cfm?id=2021109.2021114.

Allan, James (2002). «Introduction to topic detection and tracking». In: *Topic detection and tracking*. Springer, pp. 1–16. ISBN: 0-7923-7664-1. URL: http://dl.acm.org/citation.cfm?id=772260.772262.

Banerjee, Somnath, Krishnan Ramanathan, and Ajay Gupta (2007). «Clustering short texts using wikipedia». In: *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '07. Amsterdam, The Netherlands: ACM, pp. 787–788. ISBN: 978-1-59593-597-7. DOI: 10.1145/1277741.1277909. URL: http://doi.acm.org/10.1145/1277741.1277909.

Bermingham, Adam and Alan F. Smeaton (2010). «Classifying sentiment in microblogs: is brevity an advantage?» In: *CIKM*. Ed. by Jimmy Huang et al. ACM, pp. 1833–1836. ISBN: 978-1-4503-0099-5. URL: http://dblp.uni-trier.de/db/conf/cikm/cikm2010.html#BerminghamS10.

Brooke, Julian, Milan Tofiloski, and Maite Taboada (2009). «Cross-Linguistic Sentiment Analysis: From English to Spanish». In: *Proc. International Conference on Recent Advances in NLP*. URL: http://www.sfu.ca/%5C~%7B%7Dmtaboada/docs/Brooke_et_al_RANLP_2009.pdf.

Chang, Chih-Chung and Chih-Jen Lin (2011). «LIBSVM: A library for support vector machines». In: *ACM Trans. Intell. Syst. Technol.* 2.3, pp. 1–27. ISSN: 2157-6904. DOI: 10.1145/1961189.1961199. URL: http://doi.acm.org/10.1145/1961189.1961199.

Engström, Charlotta (2004). «Topic dependence in sentiment classification». MA thesis. University of Cambridge.

Gabrilovich, Evgeniy and Shaul Markovitch (2005). «Feature generation for text categorization using world knowledge». In: *Proceedings of the 19th international joint conference on Artificial intelligence*. IJCAI'05. Edinburgh, Scotland: Morgan Kaufmann Publishers Inc., pp. 1048–1053. URL: http://dl.acm.org/citation.cfm?id=1642293.1642461.

Go, Alec, Richa Bhayani, and Lei Huang (2009). «Twitter Sentiment Classification using Distant Supervision». In: *Processing*, pp. 1–6. URL: http://www.stanford.edu/~alecmgo/papers/TwitterDistantSupervision09.pdf.

Justin, T. et al. (2010). «Comparison of different classification methods for emotion recognition». In: *MIPRO, 2010 Proceedings of the 33rd International Convention*, pp. 700–703.

Kukich, Karen (1992). «Techniques for automatically correcting words in text». In: *ACM Comput. Surv.* 24.4, pp. 377–439. ISSN: 0360-0300. DOI: 10.1145/146370.146380. URL: http://doi.acm.org/10.1145/146370.146380.

Laboreiro, Gustavo et al. (2010). «Tokenizing micro-blogging messages using a text classification approach». In: *Proceedings of the fourth workshop on Analytics for noisy unstructured text data*. AND '10. Toronto, ON, Canada: ACM, pp. 81–88. ISBN: 978-1-4503-0376-7. DOI: 10.1145/1871840.1871853. URL: http://doi.acm.org/10.1145/1871840.1871853.

Lee, K. et al. (2011). «Twitter Trending Topic Classification». In: *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pp. 251–258. DOI: 10.1109/ICDMW.2011.171.

Liu, Bing (2010). «Sentiment analysis and subjectivity». In: *Handbook of Natural Language Processing, Second Edition. Taylor and Francis Group, Boca*.

Martínez-Cámara, Eugenio, M. Martín-Valdivia, and L. Ureña-López (2011). «Opinion Classification Techniques Applied to a Spanish Corpus». In: *Natural Language Processing and Information Systems*. Ed. by Rafael Muñoz, Andrés Montoyo, and Elisabeth Métais. Vol. 6716. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 169–176. ISBN: 978-3-642-22326-6.

Mathioudakis, Michael and Nick Koudas (2010). «TwitterMonitor: trend detection over the twitter stream». In: *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. SIGMOD '10. Indianapolis, Indiana, USA: ACM, pp. 1155–1158. ISBN: 978-1-4503-0032-2. DOI: 10.1145/1807167.1807306. URL: http://doi.acm.org/10.1145/1807167.1807306.

Padró, Lluís et al. (2010). «Semantic Services in FreeLing 2.1: WordNet and UKB». In: *Principles, Construction, and Application of Multilingual Wordnets*. Ed. by Pushpak Bhattacharyya, Christiane Fellbaum, and Piek Vossen. Global Wordnet Conference 2010. Mumbai, India: Narosa Publishing House, pp. 99–105.

Pak, Alexander and Patrick Paroubek (2010). «Twitter as a Corpus for Sentiment Analysis and Opinion Mining». In: *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. Ed. by Nicoletta Calzolari (Conference Chair) et al. Valletta, Malta: European Language Resources Association (ELRA). ISBN: 2-9517408-6-7.

Pang, Bo and Lillian Lee (2008). «Opinion Mining and Sentiment Analysis». In: *Found. Trends Inf. Retr.* 2.1-2, pp. 1–135. ISSN: 1554-0669. DOI: 10.1561/1500000011. URL: http://dx.doi.org/10.1561/1500000011.

Rahimtoroghi, Elahe and Azadeh Shakery (2011). «Wikipedia-based smoothing for enhancing text clustering». In: *Proceedings of the 7th Asia conference on Information Retrieval Technology*. AIRS'11. Dubai, United Arab Emirates: Springer-Verlag, pp. 327–339. ISBN: 978-3-642-25630-1. DOI: 10.1007/978-3-642-25631-8_30. URL: http://dx.doi.org/10.1007/978-3-642-25631-8_30.

Read, Jonathon (2005). «Using emoticons to reduce dependency in machine learning techniques for sentiment classification». In: *Proceedings of the ACL Student Research Workshop*. ACLstudent '05. Ann Arbor, Michigan: Association for Computational Linguistics, pp. 43–48. URL: http://dl.acm.org/citation.cfm?id=1628960.1628969.

Sidorov, Grigori et al. (2012). «Empirical Study of Machine Learning Based Approach for Opinion Mining in Tweets». In: *MICAI*.

Sriram, Bharath et al. (2010). «Short text classification in twitter to improve information filtering». In: *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. SIGIR '10. Geneva, Switzerland: ACM, pp. 841–842. ISBN: 978-1-4503-0153-4. DOI: 10.1145/1835449.1835643. URL: http://doi.acm.org/10.1145/1835449.1835643.

Vakali, Athena, Maria Giatsoglou, and Stefanos Antaris (2012). «Social networking trends and dynamics detection via a cloud-based framework design». In: *Proceedings of the 21st international conference companion on World Wide Web*. WWW '12 Companion. Lyon, France: ACM, pp. 1213–1220. ISBN: 978-1-4503-1230-1. DOI: 10.1145/2187980.2188263. URL: http://doi.acm.org/10.1145/2187980.2188263.

Chapter 3

Paper 2: Graph-based Techniques for Topic Classification of Tweets in Spanish

Published in: *IJIMAI 2, no. 5, 2014*

Héctor Cordobés[1] **Antonio Fernández Anta**[1] **Luis F. Chiroque**[1]
Fernando Pérez[2] **Teófilo Redondo**[3] **Agustín Santos**[1]

[1] IMDEA Networks Institute, Madrid, Spain
[2] U-tad, Madrid, Spain
[3] Factory Holding Company 25, Madrid, Spain

keywords Topic classification, text classification, graphs, natural language processing

Abstract Topic classification of texts is one of the most interesting challenges in Natural Language Processing (NLP). Topic classifiers commonly use a bag-of-words approach, in which the classifier uses (and is trained with) selected terms from the input texts. In this work we present techniques based on graph similarity to classify short texts by topic. In our classifier we build graphs from the input texts, and then use properties of these graphs to classify them. We have tested the resulting algorithm by classifying Twitter messages in Spanish among a predefined set of topics, achieving more than 70% accuracy.

Acknowledgments Partially funded by the SOCAM research project, Spanish Ministry of Industry, Energy and Turism.

3.1. Introduction

Topic classification of texts is one of the most interesting challenges in Natural Language Processing (NLP). The problem is to assign to every input text to be classified one topic chosen from a collection of predefined topics. Topic classifiers have commonly used a bag-of-words approach, in which the classifier uses (and is trained with) selected terms from the input texts. In these types of approaches the biggest issue is that the set of potential terms used is huge, and has to be reduced to have a practical classifier. Hence, the preprocessing of the texts and the selection of the most important terms to be used becomes fundamental.

In this work, we present classification techniques that are not based on the bag-of-words paradigm. Instead, they generate graphs from the texts, and use graph similarity to classify them by topic. The resulting classifier uses much fewer attributes than bag-of-words classical classifiers.

A prototype classifier was developed using the techniques proposed here, and was used to participate in the topic classification challenge of the Workshop on Sentiment Analysis at SEPLN - 2013, known as TASS 2013 (Taller de Análisis de Sentimientos en SEPLN 2013). As in previous years, the challenge organizers prepared and made available a data set for evaluation. For topic classification, a set of Twitter messages (tweets) in Spanish were provided. Some of these tweets had been previously classified among predefined categories (politics, economy, music, sports, etc.), and the rest was to be classified by the systems developed by the challenge participants. The classifier we developed ended in 3rd position (with respect to the F1 characteristic), very close to the systems that ended first and second, which used classical techniques.

Additionally, we have also tested different configurations of our classifier using the whole data set of tweets provided by the TASS organizers (including the ones used for evaluation), and found that our classifier achieves accuracies above 70%, using very few attributes. In the classifier developed and tested in this work, we have also explored pre-processing alternatives, such as simple Named-Entity Recognition, Thesauri and specific dictionaries (e.g., SMS abbreviations) to account for the special medium Twitter is. We believe that thorough work on this pre-made knowledge data bases could greatly improve the results of the classification.

The rest of the paper is structured as follows. We revise graph-based approaches for NLP in Section 3.2. In Section 3.3 we describe the basic techniques used by our classifier, while in Section 3.4 we describe how these techniques have been transformed into an operational system. In Section 3.5 we present the evaluation results that have been obtained and discuss their significance and implications.

3.2. State of the Art

The great representational power of graphs, in terms of element relationships, and the extensive mathematical work in graph theory, have been useful for text processing. Graph techniques

have been successfully exploited for many tasks such as text summarization and information retrieval.

In fact, a number of scientific works use graph techniques for text summarization of big documents, such as Blanco and Lioma, 2012 or Thakkar, Dharaskar, and Chandak, 2010. Similarly, the TextRank method R. Mihalcea and Tarau, 2004, which is the application of the well-known PageRank metric Brin and Page, 1998 to text graphs, has been used with remarkable success Hassan, Rada Mihalcea, and Banea, 2007 to extract good representatives in text-related graphs by using a random-walk approach. The method is based on the assumption that well-connected nodes (e.g., terms or sentences), would be good representatives of a graph. These works also use an additional set of techniques in order to exploit the relation between sentences in the same document. For this matter, methods such as tf-idf Salton and McGill, 1983, combined with mutual information, information gain, Helmholtz principle Dadachev et al., 2012, and other weighting mechanisms, have been developed to fine-tune the importance of the terms, mainly towards a subsequent bag-of-words scheme. For example, for classification tasks, it is common to describe documents within a Vector Space Model (VSM), and classify them with Rocchio or SMO classifiers, in which each feature is a weighted term. These methods relay in calculating centroid representatives of the text to summarize. Unfortunately, they may sometimes fall in a multi-centroid problem, for which good decision borders determination can be difficult to solve.

In this work, we propose a system where very short text classification is possible by using a vector classification model for which the features are not terms, but graph metrics, thus significantly reducing the training and exploitation computational requirements, while retaining reasonable accuracy. As mentioned, this work makes use of the TASS2013 corpus, managed by SEPLN (Spanish Society for Natural Language Processing) for its TASS sentiment classification challenge. This corpus is in Spanish, which prevents us from using well-known baselines for the English language, such as Reuters-21578 Lewis, n.d. Instead, we will compare ourselves with other participants in the same task.

Nevertheless, this work is a first step in the application of graph techniques to topic classification of short texts, so it must be taken as a proof of concept. More advanced techniques can be used in conjunction with this classification scheme, such as PoS tagging and dependency trees Vilares, Alonso, and Gómez-Rodríguez, 2013, or sophisticated text normalization Porta and Sancho, 2013.

3.3. Basic Graph-based Classification Techniques

The basic principle for all our techniques is that every piece of text (tweets in this case, and in general a sentence) can be represented as a graph. Essentially, for a given text our proposal uses the words in the text as graph vertices (we usually work only with the word lemmas, and optionally with named entities), and creates weighted edges between the words. We have considered different ways of assessing weights on the edges. A simple option is that the weight represents

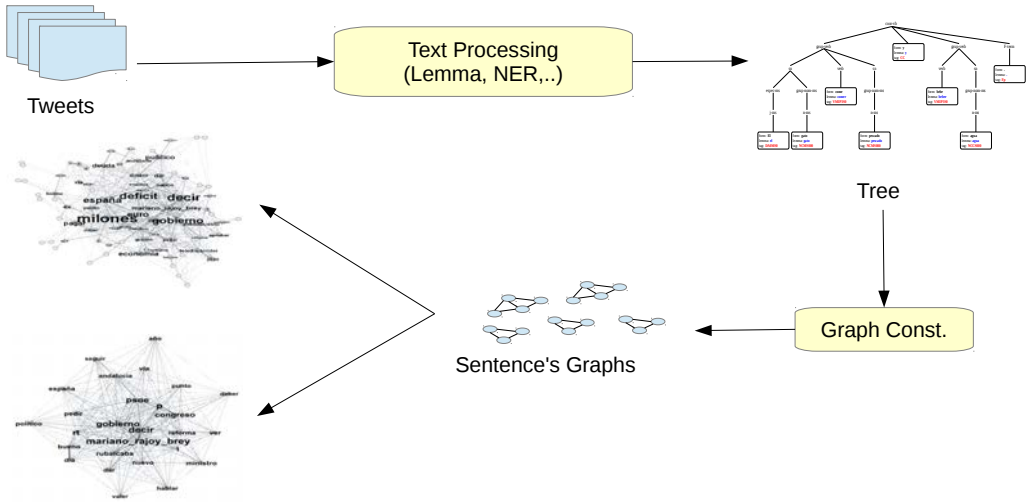


Fig. 3.1: **Reference graph building process.** From a set of tweets belonging to a specific topic, a set of per-sentence graph is constructed. The reference graph is obtained joining each of these graphs.

the frequency with which both words occur together in the text. Another more sophisticated (and complex) choice is that this frequency is weighted by the distance between the words in the syntactic tree of the text. There are other alternatives for building the graph that we deem of great interest in future work (especially those based in directed graphs).

Knowing how to build a graph for each tweet, the first hypothesis for our system is that graphs belonging to the same topic have a common representative structure (topic reference graph). For the text classification, we look for the similarities between the graph generated for a given text and different topic reference graphs. Hence, our work uses a technique of graph similarity in order to detect the topic of a piece of text.

Hence, for our experiments, we have built a reference graph for each topic. This graph is the union of all the graphs generated from all the texts of the same topic. In the resulting reference graph, the weights of the same edge in different graphs are added. This decision is based on the second hypothesis of our work, that is, all words relate to each other with different intensities depending on the topic. For instance, when the topic is *Politics,* the words *Presidencia* and *Congreso* will show a strong relationship. These same words may not appear or have a weak relationship in other topics (e.g., *Football*). Therefore, the reference (union) graph created for every topic is expected to be very different. The overall process of building the reference graphs is shown in Figure 3.1.

Hence, using a pre-classified set of tweets for training, our system builds the reference graph for each of the different topics. When a new tweet needs to be classified, its graph is generated. Then, we search for the reference graph with the highest similarity with the tweet graph we want to classify. Figure 3.2 shows this process.

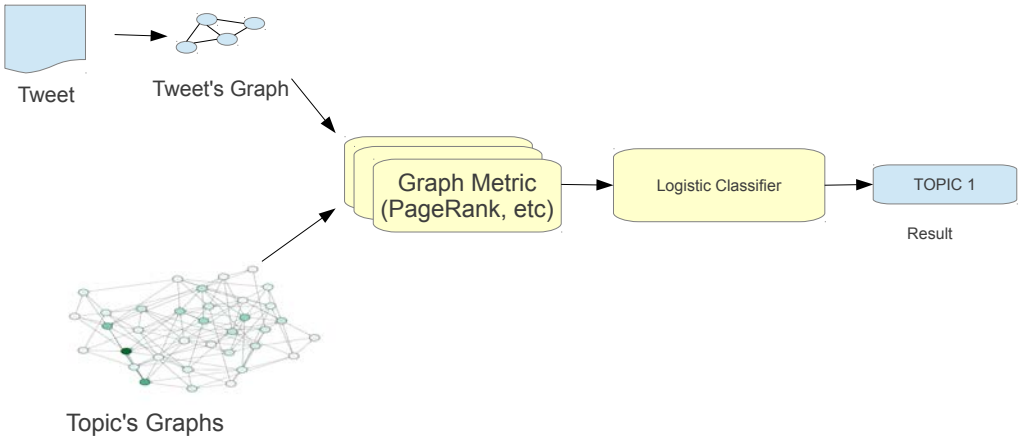


Fig. 3.2: **Text classification.** Similarities among the text graph and reference graphs are calculated in order to classify a text.

The basic mechanism previously described opens up a wide spectrum of choices and approaches that can be combined in multiple ways. The first step in the mechanism is to build the graph for the tweet. As we have already mentioned, in our work we have explored several options for selecting nodes and assigning weights to the edges. Similarly, we have used several criteria to measure the similarity of a given graph to a reference graph. In the following sections we go into greater detail about the methods we have employed.

3.4. Implementing the Classifier

In this section we describe how the classifier has been developed, and particularly how the techniques described in the previous section have been implemented. In Section 3.4.1 we describe the preprocessing that all the tweets go through before using them to build the associated graphs. Section 3.4.2 describes how the reference graphs get built. Finally, in Section 3.4.3 we describe how the topic of a new tweet is identified.

3.4.1. Preprocessing of the Text

As a step prior to building and analyzing the graphs, we run a preprocessing phase on the texts. This is a typical step in many natural language processing techniques. In this phase, the text is corrected, analyzed, and separated into simple elements. In our work we have used the Hunspell dictionaries to obtain an orthographically correct text. We have also used the dictionary of SMS abbreviations and symbols (SMS dictionary) that we already used in the system we developed for TASS 2012 Fernández Anta et al., 2013. In addition, we have used Freeling Padró et al., 2010

for word lemmatization, taking always into account the automatic disambiguation of lemmas according to the syntactic function. Freeling is also used to parse the syntactic tree of the tweets, which is used for calculating the distances between words. These distances will be used in the following sections.

Another step in the preprocessing phase has been identifying the Named Entities (Named Entity Recognition or NER process). The objective in this step has been to have mechanisms available in order to unify in a single term collections of words that refer to the same concept (e.g., *Real Madrid*, *Real Madrid C.F.*). To this end, and as a proof of concept, we have used a small manually-created catalog of slightly less than 100 entity names, with several variations for each one. For the creation of the catalog, the texts in the training set have been separated into n-grams, with no limit as to their length, using the technique described in Nagao and Mori, 1994. After the extraction of statistically significant n-grams, the catalog was manually extended both in similar concepts (for instance, the name of a media provider) and in the different ways these concepts may be present.

For the NER we have used a search in the catalog for every single occurrence of the n-gram in the text in order to verify if it refers to one of the entities in the catalog. If so, the n-gram gets substituted by a given canonical name. For instance, the bigram *Mariano Rajoy* has been considered as one such entity, in this case with canonical name *mariano_rajoy_brey*. The whole process has been executed as an experiment, and we believe that broadening its use and having a more complete catalog could improve significantly the quality of the results.

In summary, the preprocessing of each tweet goes through the following phases: first, all URLs are deleted from the tweet; second, using the SMS dictionary, the abbreviations and symbols present are replaced by their textual equivalent; third, orthography is corrected using the Hunspell dictionaries; fourth, the tweet language is detected using Cybozy Labs Language Detection Library Shuyo, 2010 and, if it is not Spanish, it is discarded; fifth, NER is applied, substituting the entities found for their canonical name (this phase can be removed at will to check how effective it is in the overall result); sixth, lemmatization is performed using Freeling; seventh and last, all the stop words are removed.

3.4.2. Reference Graphs

The key process to build a reference graph per topic is the process of building a graph for each text, since the reference graph is the union of these graphs. We have tried several options to build text graphs, described below, some of them very involved. The differences are on the set of nodes included in the graph or the way weights are assigned to the edges of the graph.

The simplest option considered for building text graphs has been using as nodes of the graph the words of the text (or the named entities, if used). Then, two nodes are connected with an edge whose weight is the product of their respective number of appearances in the text. (For instance, if in a text the word *concierto* appears twice and the word *guitarra* appears three times, the nodes of these two words are connected with a link of weight 6.) The reference graph obtained with

this option has as nodes all the words that appear in the tweets of the topic, and the weight of a link between two words is the number of instances of both words occurring together in the same tweet.

A second option explored assigns to the link between two words a weight that is inversely proportional to the distance between the two words in the text. The intuition is that two words occurring together in a text have larger affinity, and hence should have a stronger link, than words occurring at opposite ends in a sentence. This distance is derived from the syntactic parsed tree as produced by Freeling. To calculate the distance between two words we count the number of jumps in the parsed tree from one word to the other. Our experiments revealed that the results obtained with this option are similar to those with the previous one. Hence, this option was discarded, due to the additional complexity.

Another option that has been explored is using as node set not only the words that appear in the text but also all its synonyms provided by a thesaurus. The intuition is that this will increase the information of the resulting graph. In order to introduce a difference, the weight of the links involving synonyms was slightly below one, while the links connecting words in the text had weight one. In the tests run, the use of synonyms decreased the quality of the results, possibly because they interfered with the use of centrality measures for graph topic. We also tested the use of synonyms when trying to benefit from the graph information (not at the time of creating it). In this case we did not detect any significant improvement either.

As mentioned, none of the options explored was sensibly better that the first option, which is also the simplest one. Hence, this is the type of text graph that is considered in the rest of the paper. However, we think that the use of weights based on distance and synonyms must be addressed in future work since we expect that the augmented graph obtained can improve the reference graphs, and consequently yield a higher rate of successful classifications. In fact, other works such as Aseervatham, 2007 have already benefited from using thesaurus information.

3.4.3. Text Classification

We describe now how the classification of an input text has been done. One of the main questions in our approach is related with the problem of detecting graph similarity. Electing the measure of similarity is a complex decision since there are a great variety of measures and it is not clear which one would be the most appropriate for our problem. In our work we have used several measures, but all of them use the subgraphs of the reference graphs obtained after filtering out the words that do not occur in the text to be classified. That is, for each reference graph we have extracted the words occurring in the text, and we keep the links between them (i.e., we obtain the subgraph induced by the words of the text). Thus, for each topic we obtain a topic subgraph that can even be empty if no word in the text is found within the reference graph.

The following step is to determine one or several topology measures that, when applied to the topic subgraphs, would allow us to choose the topic(s) of the text. We have used two large types of measures: those based in node metrics and those based in relations metrics. The node metrics

have mainly been just two: PageRank Brin and Page, 1998 and HITS Kleinberg, 1999. For the computation of these metrics we have used the variants for undirected graphs with weighted links, and applied them to the topic reference graphs. As a result, each node of the reference graph is assigned a measure (its PageRank or HITS values).

Unfortunately, the size of the reference graphs is heavily influenced (biased) by the training set (i.e., number of tweets for each topic), and the centrality measure assigned to the nodes are influenced by the size of the graph. Hence, we attempt to compensate this deviation by means of a normalization of the centrality measures. Following a simple hypothesis, we assume that, given equal representation, the values for the centrality measures would decrease according to the number of graph nodes. Hence, we have normalized the number depending on the size of the reference graph of a given topic. On the other hand, since these values are also dependent on the graph topology in an unpredictable way, we have tried using non-linear operations (particularly, powers like 0.5 or 2), in order to give more representation capability to the system.

Then, once the topic subgraph has been extracted for a text, the topic is assigned a value that is the sum of the measures of the nodes of the subgraph (for instance, the normalized sum of PageRank for all the nodes in the subgraph). Computing this value is fast and simple from the precomputed reference graphs. These centrality measures (PageRank and HITS) have been very useful in determining the text topic, as we show later in Section 3.5 (see Table 3.1). We observed no big differences between using PageRank and HITS.

As a first approach the value assigned to each topic could be directly used for classification. After adding up the centrality measures for each word in a topic subgraph, the text is classified to the topic with the highest value. With this methodology we achieve nearly a 60% of correct classifications. However, using more sophisticated classifiers (provided in Weka) we achieve a higher rate of accuracy, as we show below.

In addition to the centrality measures, our work has also contemplated links measures. Since every link has a weight, we can compute metrics using those values. We have tried several techniques, but all of them are based on the density of the topic subgraphs (a weighted sum of the links weights). This technique by itself has not rendered better results, but during the evaluation with the training set the technique has proved to be fundamental when combined with the other techniques described before.

In order to combine all the measures described, we have used classifiers included in the Weka system Hall et al., 2009. Each tweet was represented by a vector formed by all the available metrics (PageRank's sum, HITS' sum, graph density, etc.) for every topic reference graph. All in all we have a vector with up to 70 numeric values at our disposal. Of all the classification methods available in Weka, we found that the family of *Logistic* produced a higher rate of correct classifications. Especially the *Logistic MultiClass Classifier* method, appeared to give better results in a consistent way over the training set. Hence, all the results shown below use this classifier.

Experiment	Configuration	NER	Accuracy (%)
1	$PR^{0.5}$, PR, PR^2, $HITS^{0.5}$, HITS, $HITS^2$, GD	Yes	71.90
2	$PR^{0.5}$, PR, PR^2, $HITS^{0.5}$, HITS, $HITS^2$	Yes	71.62
3	$PR^{0.5}$, PR, PR^2, $HITS^{0.5}$, HITS, $HITS^2$	No	71.38
4	PR	Yes	69.78
5	$PR^{0.5}$	No	69.45
6	$PR^{0.5}$	Yes	71.64
7	$PR^{1/3}$	Yes	71.58
8	$PR^{0.1}$	Yes	69.04
9	HITS	Yes	69.75
10	$HITS^{0.5}$	Yes	71.32
11	$HITS^{1/3}$	Yes	71.35
12	$HITS^{0.1}$	Yes	68.88

Table 3.1: Results from the experiments

3.5. Results and Discussion

We have evaluated our system with different configurations. In all the runs we have trained Weka with the full training set of TASS 2013 (approximately 7,000 tweets) and we have assessed the resulting model with slightly less than the 60,000 tweets of the test set (leaving out some tweets we could not obtain). Weka's algorithm in use has always been SimpleLogistic, as mentioned above.

In Table 3.1 we show the results in all the runs. The column "Configuration" shows the text attributes used: PageRank (PR), HITS, graph density (GD), and the modifications applied. These attributes have been generated for every single tweet both during training and evaluation. The column NER shows whether entity recognition has been used or not. As mentioned before, we have disabled this feature in some runs to measure the variation in results. The column "Accuracy" shows in percentage how the system identifies a tweet as belonging to one given topic, according to the evaluation data supplied. Experiment 1 shows the configuration submitted to the TASS 2013 contest.

Tables 3.2 and 3.3 show information about the distribution of the categories, both for the entry tweets and the results of the classifier used in experiment 1. Note that some tweets belong to more than one category, so for the sake of clarity we have expressed both the occurrence rate, and a normalized occurrence rate. This latter is intended to express the occurrence rate as though the sum of occurrences was 100%.

We present the results by category instead of showing a confusion matrix, because the possibility of finding several categories for one tweet would make the latter large and unintuitive. In Table 3.3 the success rate must be interpreted as the proportion of the tagged predictions within the category whose tweet belongs, at least, to that category.

From the results presented we think that the centrality metric used (PageRank or HITS) does

Topic	Tweets	Occurrence (%)	Normalized occurrence (%)
movies	596	1.0	0.9
sports	135	0.2	0.2
economy	2549	4.2	3.7
entertainment	5421	8.9	7.8
football	823	1.4	1.2
literature	93	0.2	0.2
music	1498	2.5	2.1
other	28191	46.4	40.5
politics	30067	49.5	43.2
technology	287	0.5	0.4

Table 3.2: Tweets per topic in TASS2013 corpus

Topic	Predictions	Ratio vs. total	Accuracy rate
movies	460	0.77	43.26
sports	67	0.11	47.76
economy	612	1.03	50.16
entertainment	6919	11.66	38.98
football	420	0.71	52.62
literature	60	0.10	25.00
music	1095	1.84	51.60
other	19753	33.29	77.00
politics	29890	50.38	78.27
technology	58	0.10	32.76

Table 3.3: Results per topic

not incur significant difference. On the contrary, the use of a specific normalization may represent a significant improvement (around 2%, for instance, between Experiments 4 and 6). This, together with the good results achieved by using centrality metrics, leads us to believe that choosing an appropriate normalization is of paramount importance for the improvement of results, or in any case, using a metric capable of taking all the factors (size, topology, etc.) into account. We believe that this is an interesting area for future research.

During the execution of the experiments we have detected sensitivity to the available vocabulary. Topics with very few tweets tended to be ignored, such as the case of *Technology*, because the generated reference graphs are not representative enough. One possible future work could focus on evaluating the sensitivity with larger training sets, and thus determining and measuring how important this effect may be.

In a similar way, this sensitivity could be tested enlarging the NER collection dictionary, so that it can represent in greater detail the topics that the system handles. Maybe given the very limited size of the dictionary used (less than 100 entity names), the impact in the results is not

very significant, although consistent (around 0.3%). We should also consider that the NER rate is about 18.3% (occurring rate per tweet) and, as the corpus tweets have been selected, not many different NE recognitions have occurred. Thus we may hypothesize that the impact could be greater within more heterogeneous corpora and bigger dictionaries. This topic is worth to be explored further.

Additionally, the use of the Graph Density in Experiment 1 combines well and was able to improve another 0.3% over the already complex combination of PageRank and HITS in Experiment 2. Nevertheless, it has to be noted that it is not worth increasing unnecessarily the number of characteristics, because as it is shown on Experiment 6, some well chosen metric may be very significant by itself.

The automatic evaluation of the predictive models in Weka is limited because it cannot take more than one prediction per vector, whereas the tweet labelling may include more than one topic per tweet. It is quite possible that a classifier that allows more than one topic per tweet would achieve better results.

Concerning the results for individual categories (Table 3.3), the system appears quite biased towards the main categories (*politics* and *other*), as they account for 46.4% and 49.5% respectively of the original tweets. In these cases the system achieves roughly a 78% of correct classification. However, the remaining categories show a rather poor behaviour, many below a mere 50%. Of particular note is the case of *entertainment*, with a success rate of only 38%, even though it is the third category in the total number of tweets.

We think that an additional experiment with more accurate training could reveal if this behaviour is due to an unbalanced training or to the actual design of the system. Since the number of training texts in some categories (for instance, *literature*) is rather scarce, we think that a far more complete training set than that currently available would be needed.

References

Aseervatham, Sujeevan (2007). «Apprentissage à base de Noyaux Sémantiques pour le traitement de données textuelles». PhD thesis. Université Paris-Nord-Paris XIII.

Blanco, Roi and Christina Lioma (2012). «Graph-based term weighting for information retrieval». In: *Information retrieval* 15.1, pp. 54–92.

Brin, Sergey and Lawrence Page (1998). «The anatomy of a large-scale hypertextual Web search engine». In: *Comput. Netw. ISDN Syst.* 30.1-7, pp. 107–117. ISSN: 0169-7552. DOI: 10.1016/S0169-7552(98)00110-X. URL: http://dx.doi.org/10.1016/S0169-7552(98)00110-X.

Dadachev, Boris et al. (2012). «On the Helmholtz Principle for Data Mining». In: *Emerging Security Technologies (EST), 2012 Third International Conference on*. IEEE, pp. 99–102.

Fernández Anta, Antonio et al. (2013). «Sentiment Analysis and Topic Detection of Spanish Tweets: A Comparative Study of of NLP Techniques». In: *Procesamiento del Lenguaje Natural* 50, pp. 45–52.

Hall, Mark et al. (2009). «The WEKA data mining software: an update». In: *SIGKDD Explorations* 11.1, pp. 10–18.

Hassan, Samer, Rada Mihalcea, and Carmen Banea (2007). «Random walk term weighting for improved text classification». In: *International Journal of Semantic Computing* 1.04, pp. 421–439.

Kleinberg, Jon M. (1999). «Authoritative sources in a hyperlinked environment». In: *J. ACM* 46.5, pp. 604–632. ISSN: 0004-5411. DOI: 10.1145/324133.324140. URL: http://doi.acm.org/10.1145/324133.324140.

Lewis, David D. *Reuters-21578*.

Mihalcea, R. and P. Tarau (2004). «TextRank: Bringing Order into Texts». In: *Proceedings of EMNLP-04and the 2004 Conference on Empirical Methods in Natural Language Processing*. Barcelonaand Spain.

Nagao, Makoto and Shinsuke Mori (1994). «A new method of n-gram statistics for large number of n and automatic extraction of words and phrases from large text data of Japanese». In: *Proceedings of the 15th conference on Computational Linguistics, COLING 1994, Volume 1*. Association for Computational Linguistics, pp. 611–615.

Padró, Lluís et al. (2010). «Semantic Services in FreeLing 2.1: WordNet and UKB». In: *Principles, Construction, and Application of Multilingual Wordnets*. Ed. by Pushpak Bhattacharyya, Christiane Fellbaum, and Piek Vossen. Global Wordnet Conference 2010. Mumbai, India: Narosa Publishing House, pp. 99–105.

Porta, Jordi and José-Luis Sancho (2013). «Word Normalization in Twitter Using Finite-state Transducers.» In: *Tweet-Norm@ SEPLN* 1086, pp. 49–53.

Salton, Gerard and Michael J McGill (1983). *Introduction to Moderm Information Retrieval*.

Shuyo, Nakatani (2010). *Language Detection Library for Java*. http://code.google.com/p/language-detection/. URL: `http://code.google.com/p/language-detection/`.

Thakkar, Khushboo S, Rajiv V Dharaskar, and MB Chandak (2010). «Graph-Based Algorithms for Text Summarization». In: *Emerging Trends in Engineering and Technology (ICETET), 2010 3rd International Conference on*. IEEE, pp. 516–519.

Vilares, David, Miguel A. Alonso, and Carlos Gómez-Rodríguez (2013). «Una aproximación supervisada para la minería de opiniones sobre tuits en español en base a conocimiento lingüístico». In: *Procesamiento del Lenguaje Natural* 51, pp. 127–134.

Chapter 4

Paper 3: Unsupervised Scalable Statistical Method for Identifying Influential Users in Online Social Networks

Published in: *Scientific Reports 8, Article number: 6955 (2018)*

A. Azcorra[1,2] **LF Chiroque**[1,2] **R. Cuevas**[1] **A. Fernández Anta**[2]
H. Laniado[3] **RE Lillo**[4,5] **J. Romo**[4,5] **C. Sguera**[5]

[1] Universidad Carlos III de Madrid, Leganés, Madrid, Spain.
[2] IMDEA Networks Institute, Leganés, Madrid, Spain.
[3] Department of Mathematical Sciences, Universidad EAFIT, Universidad Nacional de Colombia, Medellín, Colombia.
[4] Department of Statistics, Universidad Carlos III de Madrid, Madrid, Spain.
[5] UC3M-BS Institute of Financial Big Data, Universidad Carlos III de Madrid, Madrid, Spain.

Abstract Billions of users interact intensively every day via Online Social Networks (OSNs) such as Facebook, Twitter, or Google+. This makes OSNs an invaluable source of information, and channel of actuation, for sectors like advertising, marketing, or politics. To get the most of OSNs, analysts need to identify influential users that can be leveraged for promoting products, distributing messages, or improving the image of companies. In this report we propose a new unsupervised method, Massive Unsupervised Outlier Detection (MUOD), based on outliers detection, for providing support in the identification of influential users. MUOD is scalable, and can hence be used in large OSNs. Moreover, it labels the outliers as of shape, magnitude, or amplitude, depending of their features. This allows classifying the outlier users in multiple dif-

ferent classes, which are likely to include different types of influential users. Applying MUOD to a subset of roughly 400 million Google+ users, it has allowed identifying and discriminating automatically sets of outlier users, which present features associated to different definitions of influential users, like capacity to attract engagement, capacity to attract a large number of followers, or high infection capacity.

Acknowledgments Partially supported by Ministerio de Economia y Competitividad grant ECO2015-66593-P, Regional Government of Madrid (CM) grant Cloud4BigData (S2013/ICE-2894, co-funded by FSE & FEDER), the European Union through the ReCRED (653417) project and MIT MISTI Global Seed Funds through the MyBubble project. Henry Laniado has been partially supported by Departamento Administrativo de Ciencia y Tecnología, COLCIENCIAS, under Convocatoria 656, 2014, Es Tiempo de Volver; and wants to express his thanks to the hospitality of El Departamento de Ingeniería de la Organización, Universidad Nacional de Colombia sede Medellín and to the research group, Modelamiento y Análisis Energía Ambiente Economía (MAEAE) led by Professor Sergio Botero-Botero. The authors would like to thank Manuel Cebrián for useful discussions and reviewing the manuscript.

Author contributions Azcorra, A., Chiroque, L.F., Cuevas, R., Fernández Anta, A., Laniado, H., Lillo, R. E., Romo, J., and Sguera, C. have contributed to (1) the formulation of theory and prediction, (2) the experimental conception and design, (3) the acquisition, analysis or interpretation of data, and (4) drafting the article or revising it critically for important intellectual content.

4.1. Introduction

Online Social Networks (OSNs) such as Facebook, Twitter, or Google+ have rapidly become the most used online services, through which billions of users intensively interact every day Jin et al., 2013. This makes OSNs an invaluable resource for sectors like advertising, marketing, or politics, which can use them for collecting information and launching campaigns. A challenging important problem is the identification of influential OSNs users, which can be leveraged by the abovementioned actors for, e.g., advertising a product, propagating a message, or improving the image of a company. The research community has devoted significant effort in characterizing influential OSNs users Arruda et al., 2014; Kitsak et al., 2010; Morone and Makse, 2015; Kempe, Kleinberg, and Tardos, 2015; Domingos and Richardson, 2001; D'Agostino et al., 2015. However, most existing works define a priori the properties that identify influential users, and then use mechanisms based on that definition to find them Bakshy et al., 2011; Basaras, Katsaros, and Tassiulas, 2013; Cha et al., 2010; Morone and Makse, 2015; Simmie, Vigliotti, and Hankin, 2014. These supervised techniques have two main drawbacks. First, they require considerable manual analysis of the problem and the data for the definition of properties. Second, their effectiveness is fully tied to the definition: if such definition is inaccurate or unsuitable in a given context, the

results would be likewise inaccurate or unsuitable. Therefore, effective unsupervised methods to assist in the detection of influential users would be greatly advantageous. Recently proposed methods for outlier detection in the area of functional data analysis (henceforth FDA) Hubert, Rousseeuw, and Segaert, 2015 could be applied to this problem as a form to identify different classes of outliers, which are likely to meet the requirements of different influential user's definitions. Unfortunately, their outlier detection performances are poor with respect to MUOD, or their computational efficiency does not allow applying them to current OSNs (billions of users, each, characterized by tens of variables). (See supplementary material for more details.)

4.2. Contributions

In this report we present a new unsupervised method, that we call Massive Unsupervised Outlier Detection (MOUD), for supporting the identification of influential users in OSNs. MOUD is based on outlier detection in the area of FDA, and it scales to its application in OSNs with millions of users. MUOD considers the characteristics of a user in the form of signal as shown in the example of Figure 1. Each point in the x-axis is one of the user's characteristics and the correspondent value for each characteristic is represented in the y-axis. Figure 1.A shows the signal associated to a set of users with similar characteristics. MUOD identifies three types of outliers: (a) Magnitude outliers, whose associated signals present a magnitude significantly different from the mass of users; (b) Amplitude outliers, whose associated signals present an amplitude significantly different from the mass of users; (c) Shape outliers, whose associated signals present a shape significantly different from the mass of users. Figures 1.B, 1.C and 1.D show a graphical example of signals associated to Magnitude, Amplitude and Shape, respectively. Finally, by considering the intersection of these three sets of outliers, MUOD provides a total of 7 differentiated outlier classes.

Our trials with real data sets (see supplementary material) prove that MOUD is as effective in the identification of outliers as the best state-of-the-art methods, while its much higher computational efficiency allows to apply it to much larger scale problems, including the large data scale of current OSNs. We have applied MOUD to a dataset including a complete snapshot of the social graph of Google+ (400 million nodes) as well as the overall public activity of this OSN in its two first years of operation (more than 1 billion interactions). In particular, our goal is to test the ability of MOUD as a support algorithm in the identification of influential users without pre-defining a target profile. The obtained results confirm the applicability of our methodology in practice; since it is capable of finding separate sets of outliers that include different types of influential users based on their capacity to generate engagement, attract followers or their infection capabilities, in the considered large-scale OSN. Hence our proposed method offers unsupervised support to identifying influential users in OSN in those cases where there is not a predefined type of outlier. In turn, as described later, MOUD opens alternative paths for the exploration of interesting entities in other online systems, like Social Media or Online Advertising. Additionally,

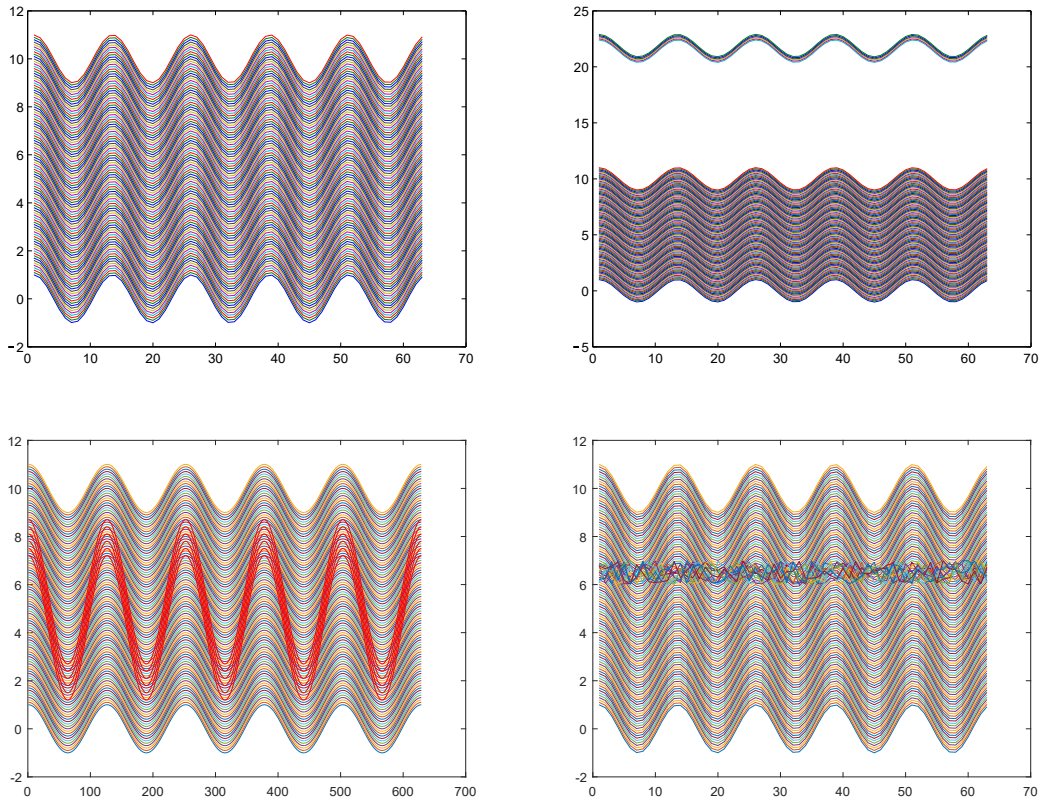


Fig. 4.1: Example of representation of users' characteristics in the form of a signal (A); Example of signals associated to magnitude outliers (B); Example of signals associated to amplitude outliers (C); Example of signals associated to shape outliers (D).

MOUD could also be applied to the identification of relevant nodes in big data problems from other disciplines (neuroscience, immune interactions, $\cdots$).

4.2.1. Identification of Influential Users in OSNs

Users in OSNs can be profiled by a set of parameters that quantify their connectivity, activity, and other relevant characteristics:

- Connectivity parameters include: in- and out-node degree (friendship or follower relationships), different centrality metrics, clustering coefficient, and others.

- Activity parameters are usually divided into two groups. One group covers the actions directly performed by a user. These usually include published posts and likes (plusones in Google+) on other users' posts. The other group includes the reactions that a user's post generates, including likes from other users, comments from other users, and reshares or reposts (i.e., retweets in the case of Twitter).

- Each user has some profiling data in an OSN. The information available in the profile

varies from one OSN to another, but typical information relates to the user's name, location (e.g., city where she lives), job, education, gender, and related data.

Parallel coordinates allow representing users as real functions, and adapt FDA techniques to this problem. Thus, outliers' observations detected in a functional data set are likely to relate to different definitions of influence in Online Social Networks, such as the capacity of creating engagement among other users, the capacity of attracting a large number of followers, or the capacity of spreading messages to a large number of other users. The first goal of MUOD is identifying outlier users that present significant differences from the mass of users. In FDA, an outlier is defined as an observation generated by a functional random variable with a distribution that is different from the one generating the observations of a functional sample Febrero, P. Galeano, and González-Manteiga, 2008. Hubert et al. Hubert, Rousseeuw, and Segaert, 2015 set up a taxonomy of functional outliers that any procedure should detect: shift/magnitude outliers, which have the same shape of the majority, but are shifted away; amplitude outliers, curves that may have the same shape as the majority but their scales differ; and shape outliers, curves whose shape differs from the majority. These qualitative definitions can be a useful tool to isolate users with different behavior but without making any a priori assumption about them. In the literature of FDA, there are recent methods that search for such outliers; however we have found that many of them are not able to distinguish between different types of outliers, nor are computationally efficient when dealing with a large number of observations (see supplementary material). These facts motivate the new outlier detection method that we have developed here. The main contribution of this work is a scalable procedure that can identify the three types of outliers separately. MUOD assigns three values to each OSN user, which defines how different the user is from the mass of users in shape, magnitude, and amplitude (note that a user can be an outlier of several types). The calculation of the three values for a given user is based on the different elements that appear when a linear regression (correlation coefficient, constant, and slope) is fitted between two curves evaluated in the same finite number of points. The larger a value, the "more outlier" the user is. In the supplementary material we have shown that MUOD, when compared with outlier detection approaches recently introduced in the literature, is always among the best options in presence of any type of outlier and thanks to its scalability does not entail the important computational limitations of its best competitors. Hence, outliers are the users with a "relatively high" value in the shape, magnitude or amplitude value. The second step in our methodology is to establish an appropriate criterion to determine which users are considered outliers, and which users are considered as part of the mass. In MUOD, we have selected the technique proposed by Louail et al. Louail et al., 2014. The approach is to first sort the users from the lowest to the highest value in the corresponding variable (e.g., shape, see Fig. 2). Then, we can adjust a continuous monotonically non-decreasing curve from the sorted values. Finally, we find the point x* in which the line tangent to the rightmost point of that curve intersects the x-axis. A user whose position is at least x* is considered outlier of that type (shape in our case). Once the outliers of the three types have been obtained, in order to classify them further, we split them according to the intersection of the

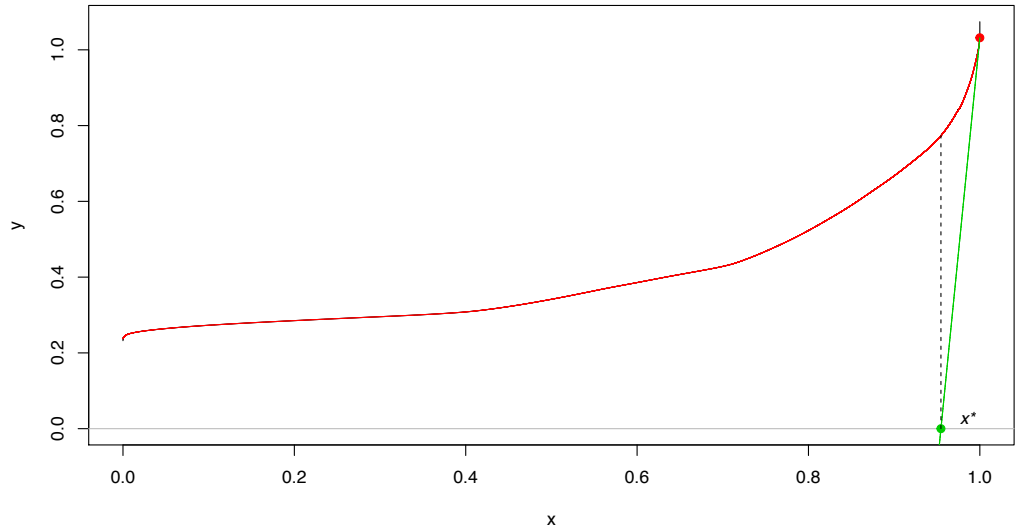


Fig. 4.2: Illustration of the criterion to determine which users are flagged as shape outliers by MUOD. The horizontal x axis represents sample percentiles based on the shape index. The vertical y axis represents shape index values.

three sets. Thus, we obtain 7 different subsets of different users: 1 group of users outliers of the three types simultaneously, 3 groups of users outliers of exactly two types, and 3 groups of users outliers of only one type. Therefore, MUOD automatically identifies outlier users, and classifies them in 7 different classes. As discussed above, these users are likely to be influential users. We will illustrate this in the next section, where we apply our method to Google+, one of the most popular OSNs in the current Social Media market.

4.2.2. Testing MUOD in Google+

Google+ was released in June 2011 and, with the support of Google accounts, it has officially more than 2.5 billion registered users. This would make Google+ the largest OSN in number of users, followed by Facebook and Sina Weibo (a OSN that operates in China). In a previous study Gonzalez et al., 2016, we collected a large-scale dataset including: (i) the connectivity graph of the Largest Connected Component of Google+, including around 400 million users; and (ii) the public activity registered in Google+ during a period of two years, since its release in June 2011 until July 2013, accounting with a total of 541 million posts, 1 billion likes, 140 million reshares, and 408 million comments. At the time of the dataset collection, Google+ had a reportedly number of registered users over half a billion. We leverage this dataset to validate the performance of the proposed method in pre-filtering users in different outlier classes, which likely include users meeting the criteria of different definition of influence within Google+. To this end, we consider n = 21 parameters for each user, covering connectivity, activity, and user profile information. For instance, parameters related to the influence of a user are:

- Number of friends and followers: characterize the popularity of a user

- Number of published posts: characterizes the level of activity of a user in the network
- Number of received likes (plusones), reshares, and comments to the users' posts: characterize the influence capacity of a user to create engagement.
- Pagerank: characterizes the topological importance of the user in the (unweighted) network.

The full list of parameters used and their description can be found in the supplementary material. As in any OSN, the distribution of the value of activity parameters (e.g., number of published posts) is heavily skewed Guo et al., 2009; Leskovec et al., 2007. This implies that there is a huge portion of the population that presents almost no measurable activity (i.e., they typically consume posts published by others but never publish their own posts nor react to posts made by others). Therefore, we have pre-processed the data to remove users of low interest. To this end, we have removed all users with less than 10 public posts in our 2 years activity dataset, since they do not have a substantial and sustained activity. After applying this filtering, we obtain a dataset of 5,619,786 users. The application of the method described in the previous section to them yields a total of 302,345, 6,178, and 6,103 outliers of shape, amplitude, and magnitude, respectively. However, the set of shape outliers completely contains the other two sets. Hence, MUOD produces in this case four different sets of outliers (see Fig. 2):

- A set MAS (mag+amp+sha) of 4,036 outliers of the three types simultaneously.
- A set MS (mag+sha) of 2,067 outliers of magnitude and shape simultaneously.
- A set AS (amp+sha) of 2,142 outliers of amplitude and shape simultaneously.
- A set SHA (sha) of 294,100 outliers of only shape.

Out of the methods for outlier detection in the current state of the art of FDA only one is computationally efficient to be able to process the millions of users we consider: FBPLOT Sun and Genton, 2011; López-Pintado and Romo, 2009. This method extends the standard boxplot to the FDA framework and allows the identification of outliers. However, as shown in the supplementary material, for FBPLOT it is hard to detect correctly amplitude and shape outliers. In Figures 3-5 we also report the results of this method, while in Figure S16 in the supplementary material we show results about a synthetic large data set.

The first observation to be made is that each of these sets is qualitatively different from the others, and from the "mass" (i.e., the users not identified as outliers, see Fig. 4). For instance, it can be seen that while the users in the mass have low values in all the parameters, the users in the above 4 outlier classes have relatively higher values in some of them. Moreover, we also observe significant differences across the different outlier classes (note that the y-axis represents log scale). If we consider the influence issue, we observe some of the identified outlier classes seem to meet the requirements of different influence concepts. For instance, we can consider influencers

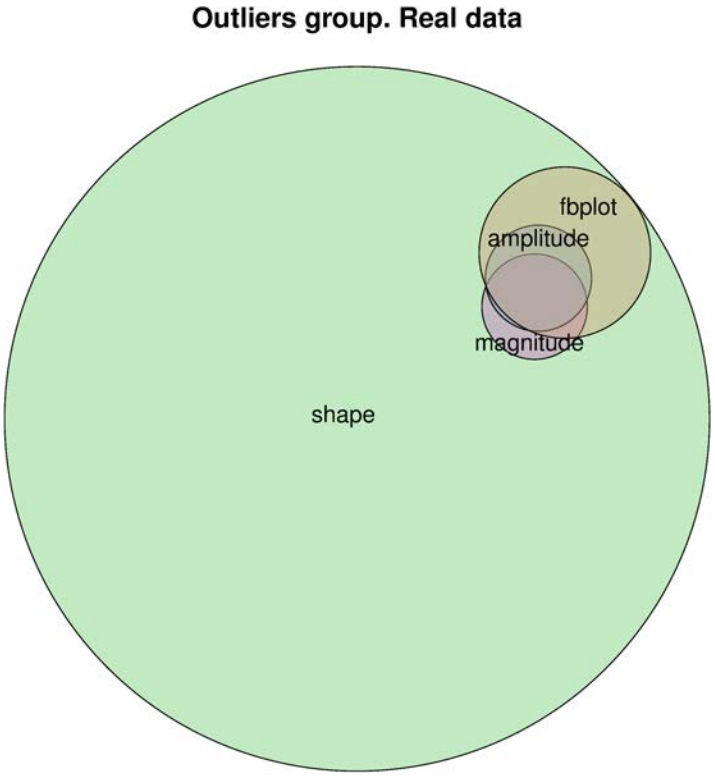


Fig. 4.3: Venn's diagram that describes the relationship between the different sets of outliers identified by MUOD (and the FBPLOT algorithm).

those users able to create engagement among other users through different types of reactions6,7 (likes/plusones, replies and reshares). Figure 4 shows that all the outlier classes identified, and in particular MAS and AS, present a volume of reactions (and hence engagement) that is orders of magnitude larger than the one of regular users (represented by the mass). Similarly, if we consider as influencers those users having a large number of followers (or higherdegree) Cha et al., 2010; Pastor-Satorras and Vespignani, 2001; Cohen et al., 2001, the MAS and MS classes show orders of magnitude more followers than the users in the mass, suggesting that this group may include influencers meeting this criterion. Finally, we may consider influential users those able to propagate messages to a larger number of other users emulating an infection process Kitsak et al., 2010; Morone and Makse, 2015; Kempe, Kleinberg, and Tardos, 2015. Then, Figure 5 shows that the MS and SHA outlier classes derived from our Google+ dataset have a significantly higher infection capacity than regular users from the mass.

FBPLOT identifies 16,140 outliers, all of them contained in the set of shape outliers, and with non-empty intersection with the other sets. The outliers identified by FBPLOT are in fact different from the mass, and seem to have properties similar to the users in AS (see Figures 4 and 5). However, simply comparing the relative sizes, one can conclude that FBPLOT leaves out many users that MUOD has detected as outliers (especially of shape). This can be seen clearly when comparing the sets of users that belong to SHA and are included (9,618) or not (284,482)

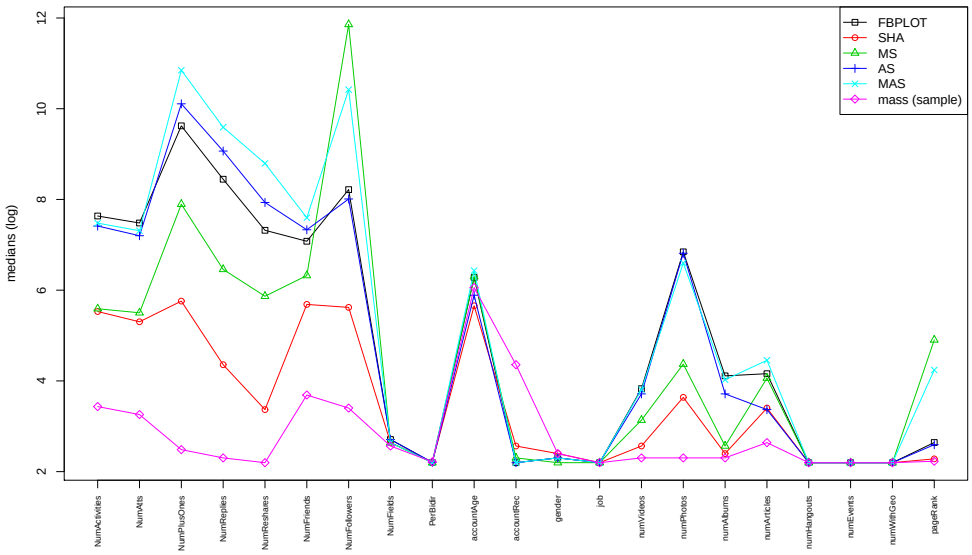


Fig. 4.4: FBPLOT outliers, MUOD outliers (four types) and sample of non-outlying users: parallel coordinates representation of their (log) medians.

in the set of FBPLOT outliers. The qualitative difference between these users is very small, and the reason for not being considered outliers by FBPLOT is rather weak.

4.3. Conclusions

In this paper, we have introduced MUOD, a novel unsupervised outlier detection algorithm based on FDA theory. MUOD outperforms other FDA-based outlier detection algorithms while offering a high scalability that allows to apply it in large scale multivariable datasets. We have tested the practical utility of MUOD in a specific problem, the detection of influencers in OSNs. The application of MUOD in a large-scale Google+ dataset including detailed information for more than 400M users and billions of activities (posts, likes/plusones, reshares, etc) reveals that the different outlier classes identified by MUOD include users that respond to different definitions of influence previously used in the literature. Hence, the results show strong evidences of the utility of MUOD as an algorithm support the unsupervised identification of influencers when a pre-defined type of influential user does not exist. MUOD algorithm can be applied to a myriad of problems, in which the nodes/users/entities can be defined by a set of properties mapped into a signal. As future work we will explore the utilization of MUOD to address the following issues: (i) Fake News detection in Social Media. News can be characterized by a large set of properties including (source of the new, timestamp, geographical origin, topic, number of nodes forwarding the news in the OSN, etc). Our hypothesis is that (at least) some types of Fake News will present specific properties that make them different from the rest. If such hypothesis is correct, MUOD should identify them as a certain types of outliers. (ii) Fraud Detection in Online Advertising.

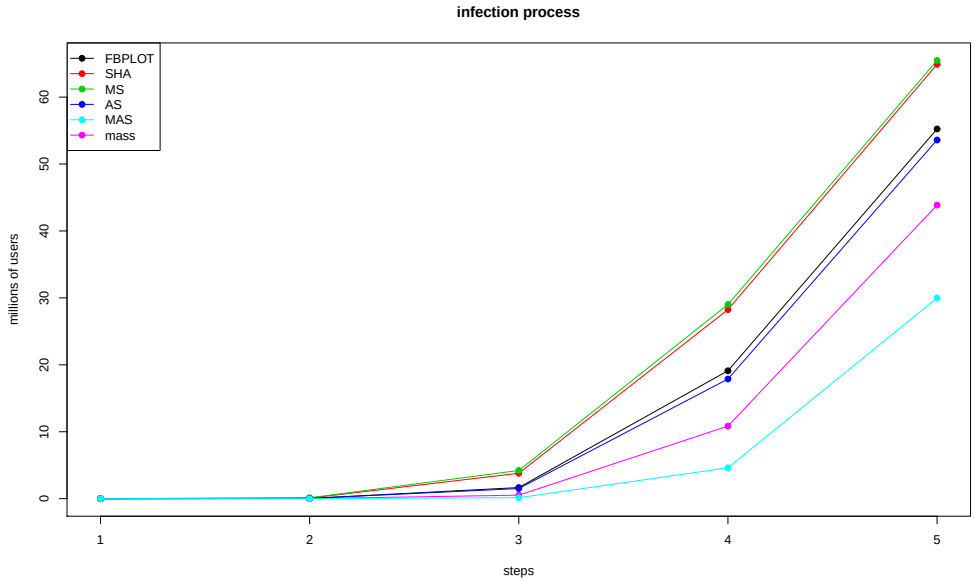


Fig. 4.5: Disease propagation simulations for the different outlier classes. Each line is the result of 10 SI (susceptible-infected) simulations using the centroid user/node of each outlier class as infection root. The simulations were carried out using the largest connected component of the network of followers (around 170M nodes) and an infection rate of 0.2.

Online Advertising is a multibillion dollar business that represents the main source of revenue for Internet. One of the main problems that online advertising faces is fraud, e.g., when a robot instead of a human watches an ad. We can create a signal associated to different websites in order to identify whether the ads presented in that website are actually viewed by human beings. This signal would include input data such IP address visiting the website, browsers visiting the website, number of ads served per hour, average time spent by users in the website, average mouse movement pattern, etc. Our hypothesis is that some websites committing ad fraud will present some specific properties that make them different from the mass. If this hypothesis hold, MUOD is a good candidate algorithm to be applied in this context. Beyond online systems, there are several real scenarios where the number of variables and the number of observations are high in which outlier detection is very important. For example, medical imaging datasets often contain deviant observations due to acquisition or pre-processing artifacts or resulting from large intrinsic inter-subject variability. Specifically in Neuro-imaging, various kinds of acquisition artifacts may be present in fMRI (functional Magnetic Resonance Images) data. Even small movements of the head may produce large artifacts in the signals, and also heartbeat and breathing both induce pulsatile motion in the brain, which creates physiological noise artifacts directly in the data Lazar, 2008; Lindquist, 2008; Monti, 2011; Poline and Brett, 2012. Complexity and massive amount of this kind of data, and the presence of different types of noises, makes the fMRI data analysis a challenging one; that demands robust and computationally efficient statistical analysis methods as MUOD. High-dimensional data are increasingly encountered in other applications

of statistics, e.g. in biological and financial studies Chen et al., 2009; Zeng et al., 2015. Also, in geochemical data, because of their complex nature Reimann and Filzmoser, 2000, regional geochemical datasets practically always contain outliers. In fact, finding data outliers that may be indicative of mineralization (in exploration geochemistry) or contamination (in environmental geochemistry) is one of the major aims of geochemical surveys Templ, Filzmoser, and Reimann, 2008.

Appendix

4.A. Introduction

In this work we deal with real multivariate data that describes $d = 21$ features of Google+ users. See Section 4.F for a detailed description of the features. Moreover, note that, after a preliminary analysis of the data, we run our study using only 21 features.

When multivariate data have more than three dimensions is practically impossible to graphically visualize the observations using Cartesian orthogonal axes. A convenient alternative is given by a visualization tool known as parallel coordinates (Wegman, 1990). Parallel coordinates substitute Cartesian orthogonal axes with parallel axes, represent a multivariate point on these parallel coordinates by a series of points and then connect each pair of adjacent points by a line, forming at the end a broken line.

As suggested by López-Pintado and Romo, 2009, once represented by means of parallel coordinates, observations $\mathbf{x} \in \mathbb{R}^d$ can be seen as real functions defined on an arbitrary set of equally spaced domain points, e.g., $\{1, \ldots, d\}$, and $\mathbf{x}$ can be expressed as $x = \{x(1), \ldots, x(d)\}$. Note that observations that can be represented as curves then can be analyzed using the tools provided by an area of statistics known as functional data analysis (FDA*). We use the same approach in this work: first, we represent multivariate data using parallel coordinates; second, we treat the transformed multivariate data as functional. Note that this approach provides a unified framework where both visualization and analysis tools become available.

In the univariate or multivariate framework we assume that observations are generated by an univariate or multivariate random variable, accordingly. Similarly, in the FDA framework it is common to assume that observations are generated by a functional random variable $X \in \mathbb{F}$, where $\mathbb{F}$ is a functional space. An alternative way to interpret X is as a stochastic process $\{X(t), t \in I\}$, where I is an interval in $\mathbb{R}$.

The accurate identification of outliers is an important aspect in any statistical data analysis and scientists have interest in identifying such atypical observations because:

- a statistical analysis performed on a sample containing outliers may lead to misleading results;

*For overviews on FDA, see Ramsay and Silverman, 2005, Ferraty and Vieu, 2006, Horváth and Kokoszka, 2012 or Antonio Cuevas, 2014

- a non-statistical analysis of the observations detected as outliers done by experts in the field may reveal interesting aspects of the phenomenon under study.

Outlier detection is an issue also in FDA, where an outlier can be defined as an observation generated by a functional random variable with a different distribution than the one generating the observations of a functional sample (Febrero, P. Galeano, and González-Manteiga, 2008). Note that the previous definition covers the case of isolated outliers, which exhibit outlying behavior during a very short domain interval, and the case of persistent outliers, which are outlying on a large part of the domain (Hubert, Rousseeuw, and Segaert, 2015). In this work we focus on persistent outliers, and on the three types of persistent outliers defined by Hubert, Rousseeuw, and Segaert, 2015: *shift/magnitude* outliers, which have the same shape of the majority, but are moved away; *amplitude* outliers, curves that may have the same shape as the majority but their scale differs; *shape* outliers, curves whose shape differs from the majority.

We propose a statistical methodology for identifying outliers and classifying them as magnitude, amplitude and shape outliers, and we argue that influential users of Online Social Networks can be seen as outliers since they usually show behaviors and patterns that are different from those of non-influential users. Therefore, the new methodology can be used to search for potentially relevant Google+ users, whose identification will be based on a statistical criterion instead of directed arguments Cha et al., 2010; Bakshy et al., 2011; Simmie, Vigliotti, and Hankin, 2014; Basaras, Katsaros, and Tassiulas, 2013; Morone and Makse, 2015.

Detection of influencers may have a special relevance in digital marketing in order to maximize profits using social networks data Bapna and Umyarov, 2015; Liu et al., 2015. Moreover, since the new procedure searches for three different types of outliers, it may reveal different types of relevant Google+ users. We refer to the proposed method as Massive Unsupervised Outlier Detection (MUOD).

Before formally defining MUOD, let us consider the illustrative example in Figure 4.A.1, where observations are curves having the following characteristic: at different but relatively common levels, they all share the same behavior in terms of magnitude, amplitude and shape. More precisely, the curves in Figure 4.A.1 represent the same equation, but at different levels defined by different constants, and we can state that data set in Figure 4.A.1 does not contain any outlier.

Figure 4.A.2 is a modification of Figure 4.A.1, where, using now relatively large constants, we obtain several curves that can be seen as magnitude outliers, without being amplitude and shape outliers. One of the goal of MUOD is to detect this kind of outliers and identify them as magnitude outliers.

Figures 4.A.3 and 4.A.4 are two additional modifications of Figure 4.A.1, and they serve to illustrate examples of amplitude and shape contamination of a data set. Our proposal will also deal with these types of scenarios of contamination, trying to solve their associated outlier detection problems.

Finally, note that we also aim to satisfactorily solve outlier detection problems where magnitude, amplitude and shape outliers simultaneously contaminate a data set as in Figure 4.A.5, or

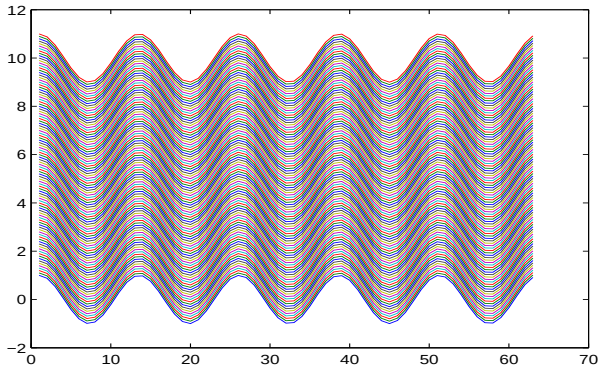


Fig. 4.A.1: Illustrative Example 1.

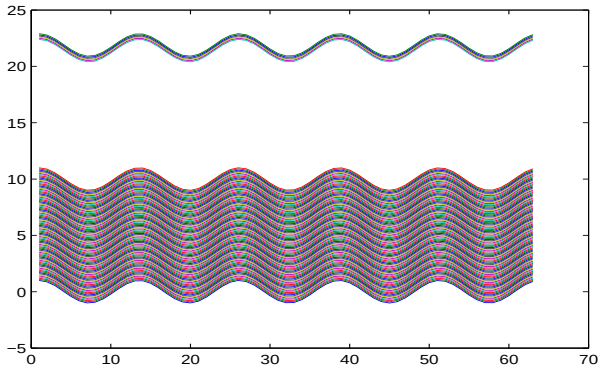


Fig. 4.A.2: A modification of Example 1 allowing magnitude outliers.

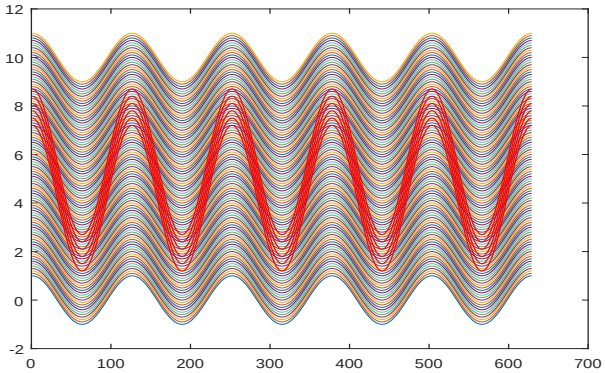


Fig. 4.A.3: A modification of Example 1 allowing amplitude outliers.

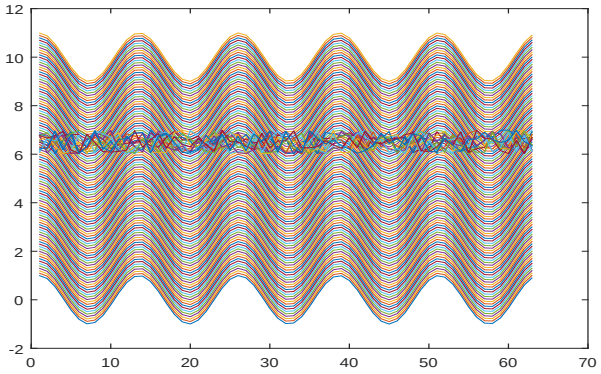


Fig. 4.A.4: A modification of Example 1 allowing shape outliers.

even where an outlier can belong to several classes.

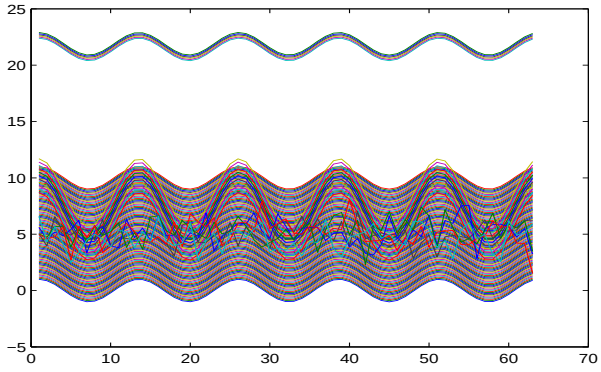


Fig. 4.A.5: A modification of Example 1 allowing magnitude, amplitude and shape outliers.

4.B. A New Outlier Detection Method Based on Indices of Magnitude, Amplitude and Shape

In this section we present three indices that can be used to analyze the type of data sets that we are considering in this work, i.e., multivariate data sets represented by means of a parallel coordinates system. These indices can be interpreted as similarity measures of an observation with respect to a sample, and each one of them focuses on a different feature of the data: magnitude, amplitude or shape. We start introducing the shape index, which is based on a well known statistical coefficient such as the Pearson correlation coefficient.

Let $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$ be a set of n curves whose common discretized form is defined on a given set of d equally-spaced domain points, and let x be another curve defined on the same set. We define the shape index of x with respect to $\mathcal{X}$ as

$$I_S(x, \mathcal{X}) = \left| \frac{1}{n} \sum_{j=1}^{n} \rho(x, x_j) - 1 \right|, \tag{4.1}$$

where $\rho(x, x_j)$ is the Pearson correlation coefficient between of x and x_j, and it is the measure responsible to capture the similarity between x and x_j in terms of shape. Note that, if we compute (4.1) for each curve in Figure 4.A.1 with respect to the whole data set, then we always obtain an I_S equal to 0. The same happens with the data sets in Figures 4.A.2 and 4.A.3. However, if we compute (4.1) for one of the shape outliers from Figure 4.A.4, say x_s, with respect to the whole data set, then we obtain $I_S(x_s, \mathcal{X})$ significantly greater than 0, while $I_S(x_c, \mathcal{X}) \simeq 0$, where x_c is any of the "common" curves in Figure 4.A.4. Indeed, we have computed $I_S(x, \mathcal{X})$ for all the curves in Figure 4.A.4 and sorted them from the lowest to the highest indices, that is, we have ranked the curves using $I_S(x, \mathcal{X})$ as criterion. Figure 4.B.1 is a scatter plot of the $I_S(x, \mathcal{X})$-based ranks (horizontal axis) and the $I_S(x, \mathcal{X})$ values (vertical axis): there are 12 points that stand out because of their $I_S(x, \mathcal{X})$ value, and they correspond exactly to the 12 shape outliers that we have generated in Figure 4.A.4.

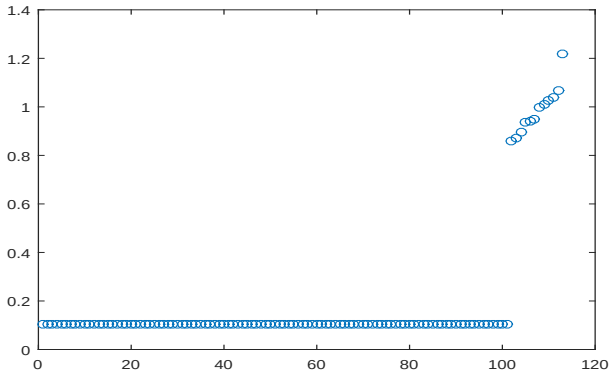


Fig. 4.B.1: Scatter plot of the $I_S(x, \mathcal{X})$-based ranks (horizontal axis) and the $I_S(x, \mathcal{X})$ values (vertical axis) for the curves from Figure 4.A.4.

Therefore, I_S seems to help in detecting shape outliers. Clearly, in real outlier detection problems the behavior of both non-outlying and outlying curves is less "regular" than the one in our illustrative examples. Hence, the corresponding scatter plots will show more differences among the index values of non-outlying curves, and likely less differences between the index values of non-outlying and outlying curves. However, we hold our statement: I_S is a useful tool to identify shape outliers.

The magnitude and amplitude indices are also based on some well-known concepts in statistics. Let $\hat{\alpha}_j$ and $\hat{\beta}_j$ be the estimated intercept and slope of a simple linear regression model where the discretized version of x represents the observed values of the dependent variable and the discretized version of x_j represents the observed values of the regressor. More formally,

$$\hat{\beta}_j = \frac{Cov(x, x_j)}{Var(x_j)}, \qquad \hat{\alpha}_j = \overline{x} - \hat{\beta}_j \overline{x}_j. \tag{4.2}$$

Then, we define the magnitude index of x with respect to $\mathcal{X}$ as

$$I_M(x, \mathcal{X}) = \left| \frac{1}{n} \sum_{j=1}^{n} \hat{\alpha}_j \right|, \tag{4.3}$$

and the amplitude index of x with respect to $\mathcal{X}$ as

$$I_A(x, \mathcal{X}) = \left| \frac{1}{n} \sum_{j=1}^{n} \hat{\beta}_j - 1 \right|. \tag{4.4}$$

As for I_S, using I_M we compute $I_M(x, \mathcal{X})$ for all the curves in Figure 4.A.2 and obtain the scatter plot of Figure 4.B.2. Also in this case we observe points that stand out because of their index's values, which correspond to the magnitude outliers from Figure 4.A.2.

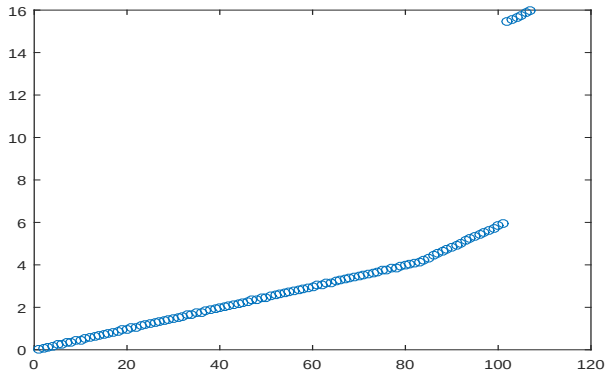


Fig. 4.B.2: Scatter plot of the $I_M(x, \mathcal{X})$-based ranks (horizontal axis) and the $I_M(x, \mathcal{X})$ values (vertical axis) for the curves from Figure 4.A.2.

Similarly, using I_A we compute $I_A(x, \mathcal{X})$ for all the curves in Figure 4.A.3 and obtain the scatter plot shown in Figure 4.B.3. Once more, the true outliers, i.e., the amplitude outliers from Figure 4.A.3, stand out in the scatter plot.

Therefore, also I_M and I_A seem to help in detecting the type of outliers for which they were designed, i.e., magnitude and amplitude outliers, respectively.

Now, we have to be able to detect and filter those values which stand out in the scatter plots. To do so, we can use a heuristic such as the first derivative of the indices curve (if necessary, normalized) and filtering out those fulfilling a certain condition (e.g. slope ≥ 2). Also, we can apply the *tangent method* as described below. Let $I_S(\mathcal{X}) = \{I_S(x_1, \mathcal{X}), \cdots, I_S(x_n, \mathcal{X})\}$ be the set of shape indices. Analogously, let $I_M(\mathcal{X})$ and $I_A(\mathcal{X})$ be the set of magnitude and amplitude indices respectively. Hereafter, we will use $I(\mathcal{X})$ for any of these three sets of indices indistinctly.

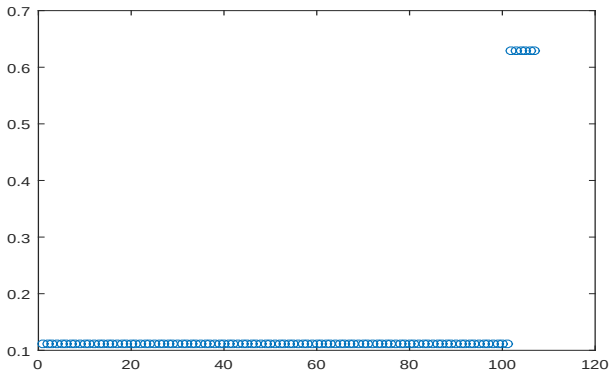


Fig. 4.B.3: Scatter plot of the $I_A(x, \mathcal{X})$-based ranks (horizontal axis) and the $I_A(x, \mathcal{X})$ values (vertical axis) for the curves from Figure 4.A.3.

Then, we sort (ascending order) and plot the set $I(\mathcal{X})$ as in, e.g, Figures 4.B.1 to 4.B.3. Thus, the *tangent method* consists in taking, as boundary between regular curves and outliers, the value $I^*(\mathcal{X}) = I(x^*, \mathcal{X})$ at the intersection point x^* between the tangent at the furthest right point and the horizontal axis. Figure 4.B.4 illustrate this method. The tangent method is inspired on the classical scale determination assuming you have an exponential decay, as described in Louail et al., 2014. Besides, the authors compare this method with the first derivative one, being the latter outperformed by the tangent method.

Finally, we use the value $I^*(\mathcal{X})$ to identify the outstanding indices. The threshold $I^*(\mathcal{X})$ represents the maximum index value to be considered a non-outlying value. Hence, any index value above this threshold is considered an outlier. Thus, we obtain the set of outliers as $I_{out}(\mathcal{X}) = \{I(x_i, \mathcal{X}) : I(x_i, \mathcal{X}) > I^*(\mathcal{X})\}$.

4.C. A New Outlier Detection Method: Implementation

We have built a scalable algorithm that implements the method. Now, we are going to outline the implementation details of our algorithm and how it scales. Since we have the set $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$, where $|x_i| = d$, we build the row-matrix $\mathbf{M}_{\mathcal{X}} = [x_1\ x_2 \ldots x_{n-1}\ x_n] \in \mathbb{R}^{d \times n}$, where $x_i \in \mathbb{R}^d$ is a curve representing each user in our dataset. For scalability, we assume $d \ll n$.

Partitioning A key step to make our algorithm *parallel* and, consequently, *scalable* is to partition the matrix $\mathbf{M}_{\mathcal{X}}$. This way, we can simultaneously compute the indices leveraging the current multi-core machines. For this purpose, we define a partition $\Pi = \{\pi_j\}$ of users, where $\pi_j \subseteq \mathcal{X}$ are non empty, mutually disjoint and $\bigcup_j \pi_j = \mathcal{X}$. The number of classes is $k = |\Pi|$. That will allow us to separately compute the indices in each class. Note that different partitions may exist and they range from the discrete partition ($k = n : |\pi_j| = 1, \forall j$) to the unit partition ($k = 1 : |\pi_j| = n, j = \{1\}$). The partitions considered hereinafter are equal sized and al-

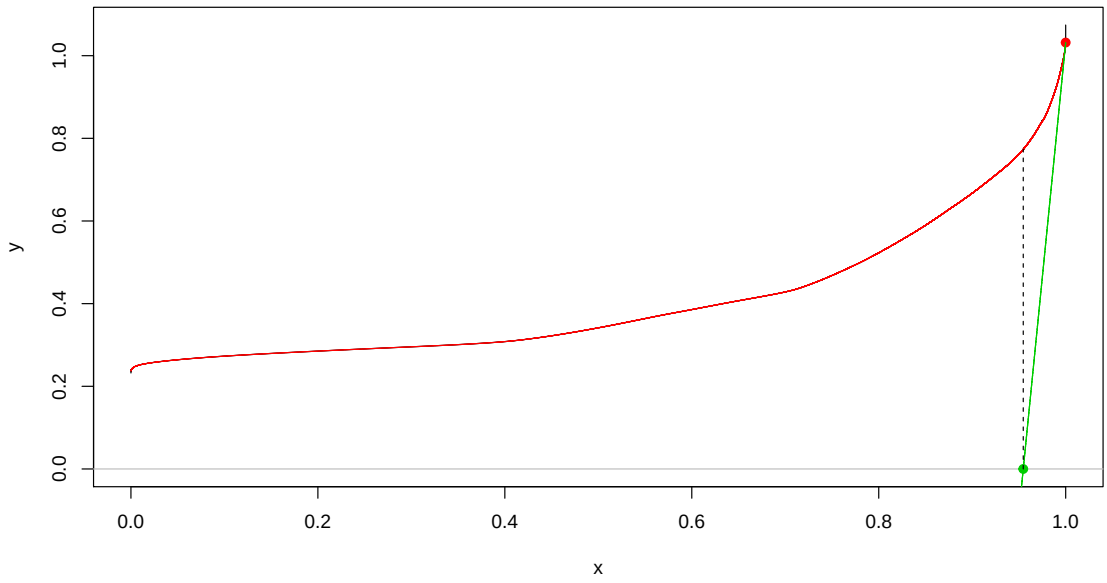


Fig. 4.B.4: Tangent method applied to shape indices for our real dataset. The horizontal axis represents sample percentiles based on the shape index. The vertical axis represents shape index values. The green dot cuts the x-axis at $x^* = 0.9546$, yielding a 4.538% of outliers (the higher values at the right-side). $I^*(\mathcal{X}) = I(0.9546, \mathcal{X})$.

ways consecutive elements (users), according to the matrix $\mathbf{M}_{\mathcal{X}}$. Our algorithm has the matrix $\mathbf{M}_{\mathcal{X}}$ and the correspondant set of users π_j as input. Note that, we can see $\pi_j \in \Pi$ as a matrix $\pi_j = [x_i \dots x_{i+|\pi_j|}] \in \mathbb{R}^{d\times|\pi_j|}$ for every j according to the partition Π.

We describe now the different sub-algorithms for the computation of the indices. Note that the algorithms will use any set π_j as input, regardless of the partition chosen.

4.C.1. *Shape* Indices

We present the algorithm to compute $I_S(\pi_j, \mathcal{X})$. Algorithm 1 shows a possible implementation of Equation 4.1. The *mean* function in Line 3 is a column-wise mean on the matrix *cor*.

Algorithm 1 Get Shape Indices

1: **function** GET_SHAPE_INDICES($\mathbf{M}_{\mathcal{X}}$, π_j)
2: $\quad$ **cor** $\leftarrow \rho(\mathbf{M}_{\mathcal{X}}, \pi_j) : \mathbb{R}^{d\times n} \times \mathbb{R}^{d\times|\pi_j|} \to \mathbb{R}^{n\times|\pi_j|}$
3: $\quad$ Return $|mean(\mathbf{cor}) - \mathbf{1}| \equiv I_S(\pi_j, \mathcal{X}) : \mathbb{R}^{n\times|\pi_j|} \to \mathbb{R}^{|\pi_j|}$
4: **end function**

Time Complexity The algorithm's complexity is dominated by computing the correlation (ρ) matrix **cor** whose complexity is $\mathcal{O}(|\pi_j|dn)$. Since this function will be run for every set π_j it will be computed for the n curves/users at the end. However, it may be computed in parallel in up to p cores, which yields a final time complexity of $\mathcal{O}(\frac{n^2d}{p})$

4.C.2. *Magnitude* & *Amplitude* Indices

These two indices are jointly computed since they are estimated together from a linear regression model as explained in Section 4.B (see equation (4.2)).

Algorithm 2 shows a possible implementation for computing $I_M(\pi_j, \mathcal{X})$ and $I_A(\pi_j, \mathcal{X})$. The $mean(\cdot)$ functions in lines 3, 8, 10 and 12 are column-wise. Finally, the division and product operations in lines 7 and 9 are both column and element-wise, while the operation in line 11 is row-wise.

Time Complexity The algorithm's complexity is dominated by computing the covariance matrix **cov** whose complexity is $\mathcal{O}(|\pi_j|dn)$. Similarly as in the previous algorithm, this function will be run for every set π_j and hence, it will be computed for the n curves/users at the end. However, it may be computed in parallel in up to p cores, which again yields a final time complexity of $\mathcal{O}(\frac{n^2d}{p})$

4.C.3. Three indices simultaneously

We have already shown how to compute the three indices with a total complexity of $\mathcal{O}(\frac{n^2d}{p})$ in the two cases, which would give us the same total time complexity. However, we would like

Algorithm 2 Get Magnitude&Amplitude Indices

1: **function** GET_MAGAMP_INDICES($\mathbf{M}_{\mathcal{X}}$, π_j, *means*, *vars*)
2: *Paramaters*:
3: $means \leftarrow mean(\mathbf{M}_{\mathcal{X}}) : \mathbb{R}^{d\times n} \rightarrow \mathbb{R}^n$
4: $vars \leftarrow Var(\mathbf{M}_{\mathcal{X}}) \equiv \{Var(x_1), \ldots, Var(x_n)\} : \mathbb{R}^{d\times n} \rightarrow \mathbb{R}^n$
5: *algorithm*:
6: $\mathbf{cov} \leftarrow Cov(\mathbf{M}_{\mathcal{X}}, \pi_j) : \mathbb{R}^{d\times n} \times \mathbb{R}^{d\times|\pi_j|} \rightarrow \mathbb{R}^{n\times|\pi_j|}$
7: $\boldsymbol{\beta} \leftarrow \mathbf{cov}/vars : \mathbb{R}^{n\times|\pi_j|} \times \mathbb{R}^n \rightarrow \mathbb{R}^{n\times|\pi_j|}$
8: $amp \leftarrow |mean(\boldsymbol{\beta}) - \mathbf{1}| \equiv I_A(\pi_j, \mathcal{X}) : \mathbb{R}^{n\times|\pi_j|} \rightarrow \mathbb{R}^{|\pi_j|}$
9: $\boldsymbol{\beta}\mathbf{x} \leftarrow \boldsymbol{\beta} \cdot means : \mathbb{R}^{n\times|\pi_j|} \times \mathbb{R}^n \rightarrow \mathbb{R}^{n\times|\pi_j|}$
10: $\overline{\pi_j} \leftarrow mean(\pi_j) : \mathbb{R}^{d\times|\pi_j|} \rightarrow \mathbb{R}^{|\pi_j|}$
11: $\boldsymbol{\alpha} \leftarrow -\boldsymbol{\beta}\mathbf{x} + \overline{\pi_j} : \mathbb{R}^{n\times|\pi_j|} \times \mathbb{R}^{|\pi_j|} \rightarrow \mathbb{R}^{n\times|\pi_j|}$
12: $mag \leftarrow |mean(\boldsymbol{\alpha})| \equiv I_M(\pi_j, \mathcal{X}) : \mathbb{R}^{n\times|\pi_j|} \rightarrow \mathbb{R}^{|\pi_j|}$
13: Return $[mag\ amp] \equiv [I_M(\pi_j, \mathcal{X})\ I_A(\pi_j, \mathcal{X})]$
14: **end function**

to compute the three indices simultaneously in order to optimize the time spent. For this purpose, we are going to leverage the following result.

$$\rho(\mathbf{x}, \mathbf{y}) = \frac{Cov(\mathbf{x}, \mathbf{y})}{\sigma_x \sigma_y} \tag{4.5}$$

Thus, we can compute the correlation matrix used in Algorithm 1 from the covariance matrix used in Algorithm 2. Basically, the resulting algorithm is similar to the previous one for magnitude and amplitude indices, but with one more parameter and one more line. This extra Line 8 applies the Equation 4.5, first dividing (column-wise) by the standard deviations of $\mathbf{M}_{\mathcal{X}}$, and second dividing (row-wise) by the standard deviations from the corresponding curves in π_j. Algorithm 3 below outlines our proposed method.

Time Complexity This final algorithm's complexity does not increase from the previous one, $\mathcal{O}(\frac{n^2 d}{p})$. In spite of not decreasing the time complexity, this last algorithm computes one more index adding a few more instructions, saving the whole computation of the correlation matrix (from scratch) in Algorithm 1. Additionally, each algorithm may be computed in parallel for each partition π_j and then concatenating the results. Observe that, we can reduce the time complexity depending on how many partitions we use and how much computing power we had, i.e., $p = 10$ leads to one order of magnitude reduction, while $p = 100$ allows two orders of magnitude reduction, and so on.

4.D. Outlier Detection in Functional Data

The aim of this section is to present the results of a simulation study where we compare MUOD with some competitors recently used in the literature of functional data. Since some of them are based on the use of measures known as functional depths, we briefly introduce a general

Algorithm 3 Get Shape, Magnitude and Amplitude Indices

function GET_SMA_INDICES($\mathbf{M}_{\mathcal{X}}$, π_j, *means*, *vars*, *sds*)
Paramaters:
    $means \leftarrow mean(\mathbf{M}_{\mathcal{X}}) : \mathbb{R}^{d\times n} \rightarrow \mathbb{R}^n$
    $vars \leftarrow Var(\mathbf{M}_{\mathcal{X}}) \equiv \{Var(x_1), \ldots, Var(x_n)\} : \mathbb{R}^{d\times n} \rightarrow \mathbb{R}^n$
    $sds \leftarrow \sigma(\mathbf{M}_{\mathcal{X}}) \equiv \{\sigma(x_1), \ldots, \sigma(x_n)\} : \mathbb{R}^{d\times n} \rightarrow \mathbb{R}^n$
algorithm:
    $\mathbf{cov} \leftarrow Cov(\mathbf{M}_{\mathcal{X}}, \pi_j) : \mathbb{R}^{d\times n} \times \mathbb{R}^{d\times|\pi_j|} \rightarrow \mathbb{R}^{n\times|\pi_j|}$
    $\mathbf{cor} \leftarrow \mathbf{cov}/sds/sds[j] : \mathbb{R}^{n\times|\pi_j|} \times \mathbb{R}^n \times \mathbb{R}^{|\pi_j|} \rightarrow \mathbb{R}^{n\times|\pi_j|}$
    $sha \leftarrow |mean(\mathbf{cor}) - \mathbf{1}| \equiv I_S(\pi_j, \mathcal{X}) : \mathbb{R}^{n\times|\pi_j|} \rightarrow \mathbb{R}^{|\pi_j|}$
    $\boldsymbol{\beta} \leftarrow \mathbf{cov}/vars : \mathbb{R}^{n\times|\pi_j|} \times \mathbb{R}^n \rightarrow \mathbb{R}^{n\times|\pi_j|}$
    $amp \leftarrow |mean(\boldsymbol{\beta}) - \mathbf{1}| \equiv I_A(\pi_j, \mathcal{X}) : \mathbb{R}^{n\times|\pi_j|} \rightarrow \mathbb{R}^{|\pi_j|}$
    $\boldsymbol{\beta}\mathbf{x} \leftarrow \boldsymbol{\beta} \cdot means : \mathbb{R}^{n\times|\pi_j|} \times \mathbb{R}^n \rightarrow \mathbb{R}^{n\times|\pi_j|}$
    $\overline{\pi_j} \leftarrow mean(\pi_j) : \mathbb{R}^{d\times|\pi_j|} \rightarrow \mathbb{R}^{|\pi_j|}$
    $\boldsymbol{\alpha} \leftarrow -\boldsymbol{\beta}\mathbf{x} + \overline{\pi_j} : \mathbb{R}^{n\times|\pi_j|} \times \mathbb{R}^{|\pi_j|} \rightarrow \mathbb{R}^{n\times|\pi_j|}$
    $mag \leftarrow |mean(\boldsymbol{\alpha})| \equiv I_M(\pi_j, \mathcal{X}) : \mathbb{R}^{n\times|\pi_j|} \rightarrow \mathbb{R}^{|\pi_j|}$
    Return $[sha\ mag\ amp] \equiv [I_S(\pi_j, \mathcal{X})\ I_M(\pi_j, \mathcal{X})\ I_A(\pi_j, \mathcal{X})]$
end function

definition. A functional depth is a measure that allows to order and rank the observations in a functional sample from the most to the least central, and it gives high values to central observations and low values to non-central observations. In FDA, unlike in univariate statistics, where $\mathbb{R}$ provides a natural order criterion for observations, several criteria have been employed to order functional data, and therefore there exist different implementations of the notion of functional depth (see C. Sguera, P. Galeano, and R. Lillo, 2014, where different functional depths have been reviewed and used in classification problems).

Recall that we propose three type-oriented outlier detection methods to find alternatively magnitude (MUOD$_{mag}$), amplitude (MUOD$_{amp}$) and shape (MUOD$_{sha}$) outliers, and note that we refer to their simultaneous use on a single data set simply as MUOD. We will compare MUOD$_{mag}$, MUOD$_{amp}$, MUOD$_{sha}$ and MUOD with a battery of competitors given by the outlier detection methods considered in Carlo Sguera, Pedro Galeano, and R. E. Lillo, 2016. In addition, we also consider a method proposed in Arribas-Gil and Romo, 2014. Next, we briefly describe these competitors (for more details, see Carlo Sguera, Pedro Galeano, and R. E. Lillo, 2016 and references therein, and Arribas-Gil and Romo, 2014):

- B$_{tri}$ and B$_{wei}$ (Febrero, P. Galeano, and González-Manteiga, 2008): depth-based outlier detection methods which select the threshold of a functional depth by means of two alternative procedures based on bootstrap techniques. B$_{tri}$ and B$_{wei}$ differ in their resampling steps: in B$_{tri}$ the resampling is done on a trimmed version of the original sample, that is, after deleting from the sample a given proportion of least deep curves (trimmed resampling); in B$_{wei}$ the resampling is done giving weights to sample observations that are proportional to their depth values (weighted resampling). B$_{tri}$ and B$_{wei}$ work with any func-

tional depth. Following their authors, we use B_{tri} and B_{wei} together with the h-modal depth (A. Cuevas, Febrero, and Fraiman, 2006).

- FBAG (Hyndman and Shang, 2010): procedure that reduces the outlier detection problem from functional to multivariate by means of the functional principal component analysis technique. Once obtained the first two functional principal components scores, FBAG orders the scores using the multivariate halfspace depth (Tukey, 1975) and builds a non-outlying region. FBAG detects as outliers those observations whose scores are outside the non-outlying region.

- FHDR (Hyndman and Shang, 2010): procedure that differs from FBAG after obtaining the first two functional principal components scores. FHDR performs a bivariate kernel density estimation on the scores and defines a high density region. FHDR detects as outliers those observations whose scores are outside the high density region.

- FBPLOT (Sun and Genton, 2011): procedure that extends the standard boxplot to the functional framework, allowing the identification of a non-outlying band defined by a specified percentage of the deepest curves, and therefore the detection of outliers. FBPLOT works with any functional depth. Following its authors, we use FBPLOT together with the modified band depth (López-Pintado and Romo, 2009).

- OG (Arribas-Gil and Romo, 2014): depth-based outlier detection method based on a visualization tool known as outliergram. The outliergram exploits the relation between the modified band depth (López-Pintado and Romo, 2009) and modified epigraph index (López-Pintado and Romo, 2011) to help understanding shape features of observations, and it expressly detects shape outliers.

- $KFSD_{smo}$, $KFSD_{tri}$ and $KFSD_{wei}$ (Carlo Sguera, Pedro Galeano, and R. E. Lillo, 2016): depth-based outlier detection methods which select the threshold of the kernelized functional spatial depth (KFSD, C. Sguera, P. Galeano, and R. Lillo, 2014) by means of a probabilistic procedure based on three alternative resampling techniques. $KFSD_{smo}$, $KFSD_{tri}$ and $KFSD_{wei}$ differ in their resampling steps: in $KFSD_{smo}$ the resampling is simple and smoothed, that is, once an observation is sampled, a small perturbation is added to the observation to avoid the presence of repeated observations; in $KFSD_{tri}$ the resampling is trimmed and smoothed; in $KFSD_{wei}$ the resampling is weighted and smoothed.

We have tested all these methods with random samples of our dataset of Google+ users in order to observe the time performance. Experiments have been carried out in an AMD Opteron 6276 multicore (x64) @ 2.3GHz with 512GiB of RAM under Debian 7.9. Figure 4.D.1 shows the time performance for all these methods including ours. We have run our method with one single partition (MUOD), and using 10 partitions (MUOD 10-way) in order to check the scalability and verify that the running time decreases one order of magnitude (better time performance). It is

observable that the parallel version needs a setup time which is amortized as long as the data size increases.

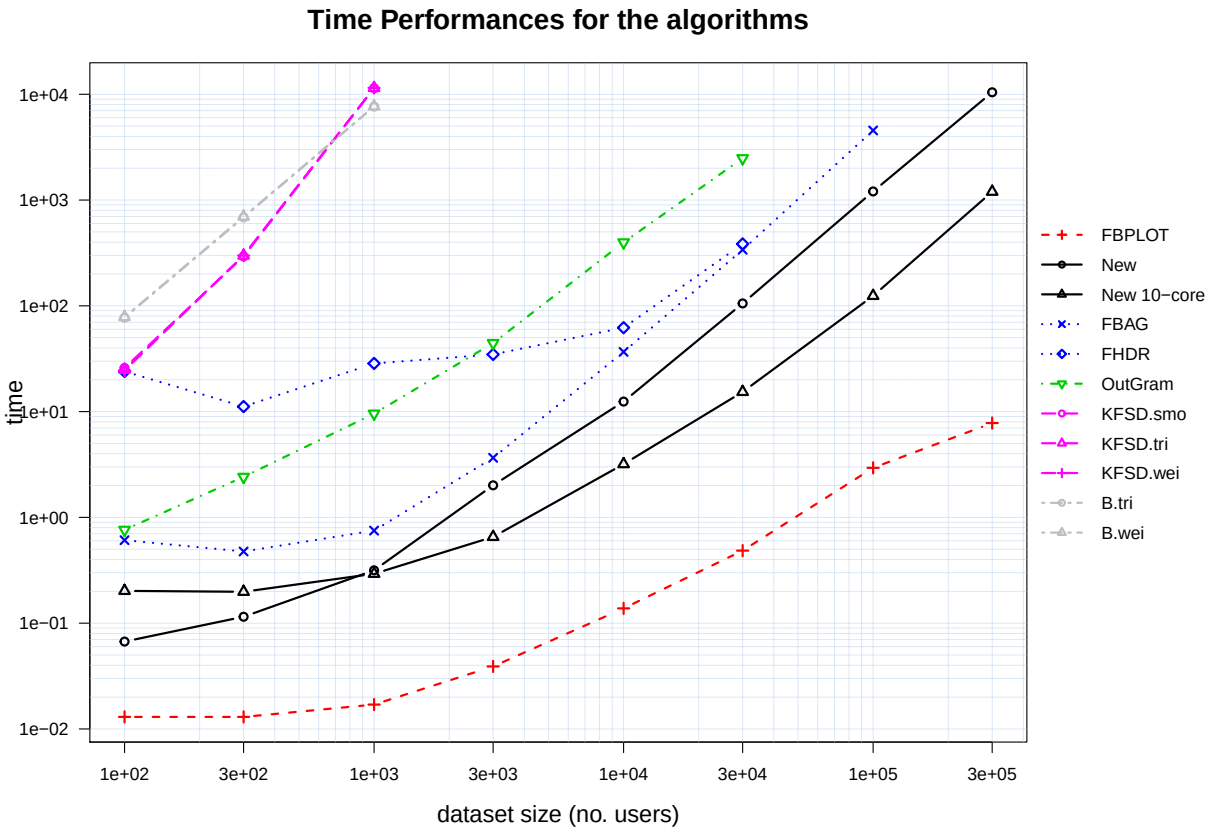


Fig. 4.D.1: Time performance for the different algorithms computing different size of data (log-log scale). FBPLOThas the best performance by far as we increase the data size. MUOD and MUOD 10-way are the second best performance lines. Observe that, FBPLOTand MUOD are the only ones performing in less than 100 seconds for a data size of $3 \cdot 10^4$ users. FBAGand FHDRhave a similiar behaviour as the data size grows and performs similar to MUOD, but they are not able to process a data size of $3 \cdot 10^5$. OutGRAM have a similar behaviour than the former algorithms but is not able to process dataset bigger than $3 \cdot 10^4$ users. Then we found the KSFD family and the B family with a poor performance and a dataset size limit of 10^3 users.

We conclude that, only FBPLOT and MUOD are able to scale out for datasets bigger than 10^5 users, which is far from our real dataset from Google+ ($5,619,786$ users). For this reason, we are only going to use FBPLOT and MUOD in our real data study case (Section 4.F), where both algorithms are evaluated and compared.

4.E. Simulation Study

In this section we present the results of a simulation study where we compare our methods with the competitors described in the previous section. The goal is to show that our procedures are competitive in the usual framework used in the FDA literature.

4.E.1. Mixture Models 1, 2, and 3

In the first part of our simulation study we generate observations using the following model:

$$X(t) = a_1 \sin t + a_2 \cos t, \tag{4.6}$$

where $t \in \{t_j\}$, $\{t_j\}$ is a sequence of d equidistant points between 0 and 2π, and a_1 and a_2 are independent realizations of a continuous uniform random variable between 0.75 and 1.25. In Figure 4.E.1 we show a simulated data set obtained using (4.6), $d = 25$ and $n = 100$, where n is the size of the data set.

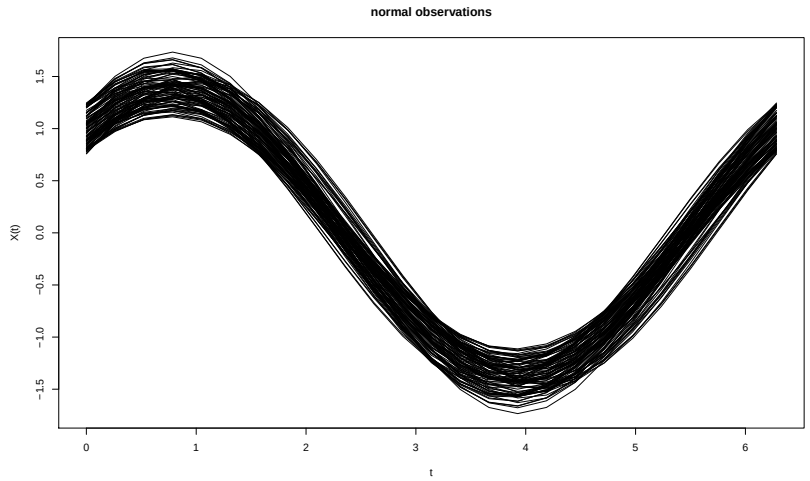


Fig. 4.E.1: Simulated observations.

To generate outliers with respect to model (4.6) we use three different models. They generate magnitude, amplitude, and shape outliers, respectively. The magnitude outliers' model is given by

$$X(t) = a_1 \sin t + a_2 \cos t + k0.5, \tag{4.7}$$

where k is a realization of a discrete random variable with $P(k = -1) = P(k = 1) = 0.5$. In Figure 4.E.2 we show a simulated data set obtained using a mixture of models (4.6) and (4.7) (mixture model 1), with $\alpha = 0.05$ being the probability that each curve is a magnitude outlier.

The amplitude outliers' model is given by

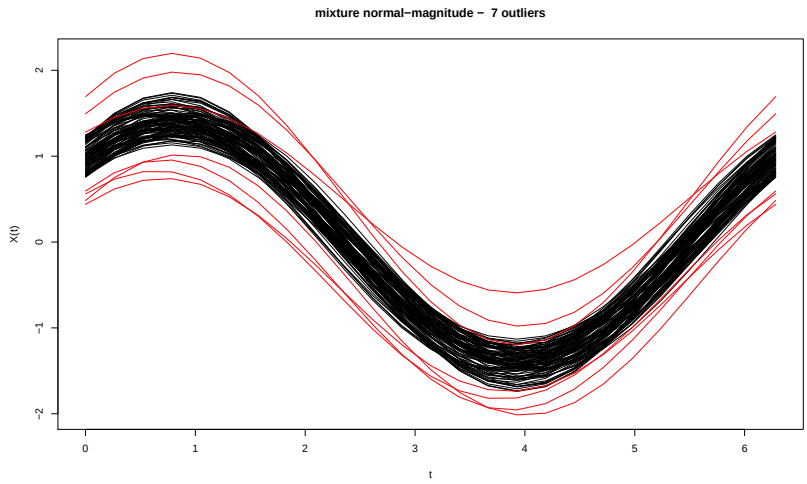


Fig. 4.E.2: Mixture model 1: magnitude outliers.

$$X(t) = b_1 \sin t + b_2 \cos t, \tag{4.8}$$

where b_1 and b_2 are independent realizations of a continuous uniform random variable between 1.30 and 1.50. In Figure 4.E.3 we show a simulated data set obtained using a mixture of models (4.6) and (4.8) (mixture model 2), with $\alpha = 0.05$ being the probability that each curve is an amplitude outlier.

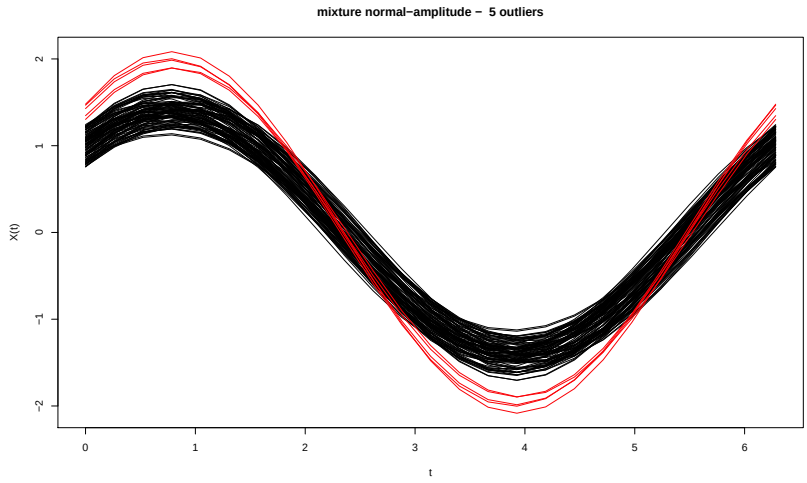


Fig. 4.E.3: Mixture model 2: amplitude outliers.

The shape outliers' model is given by

$$X(t) = a_1 \sin t + a_2 \cos t + \epsilon(t), \tag{4.9}$$

where $\epsilon(t) \in \{\epsilon(t_j)\}$ and $\{\epsilon(t_j)\}$ is a d-dimensional sample of independent realizations of a

normal random variable having mean equal to 0 and variance equal to $1/9$. In Figure 4.E.4 we show a simulated data set obtained using a mixture of models (4.6) and (4.9) (mixture model 3), with $\alpha = 0.05$ being the probability that each curve is a shape outlier.

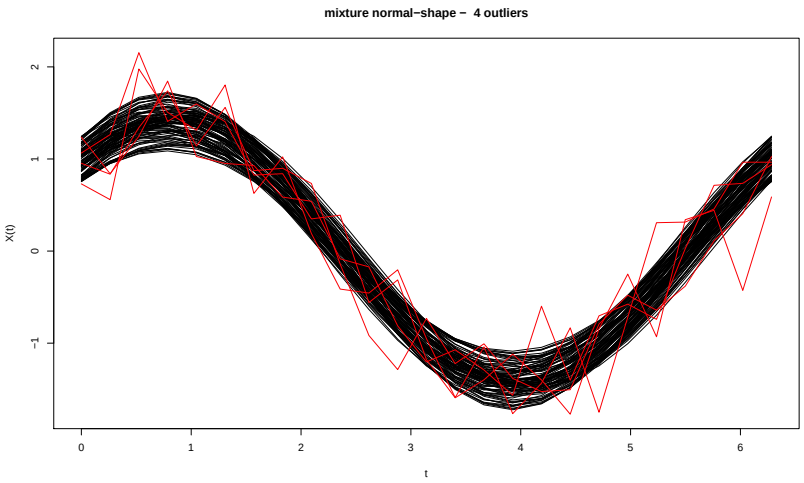


Fig. 4.E.4: Mixture model 3: shape outliers.

For each mixture model, the comparison among methods is based on 100 generated data sets and is performed in terms of correct outlier detection percentages, false outlier detection percentages and F-measures. F-measures are given by

$$F = \frac{2RP}{R+P}, \tag{4.10}$$

where $R = \frac{TP}{(TP+FN)}$ is known as recall measure, $P = \frac{TP}{TP+FP}$ is known as precision measure and TP, FN, and FP are the number of true positives, false negatives and false positives, respectively.

The F-measure is the harmonic mean of precision and recall and it is widely used instead of accuracy when the data set is unbalanced as in our seting (Rijsbergen, 1979). It gives a good trade-off for precision while preserving a high recall rate.

The results obtained by our methods and their competitors are shown in Table 4.E.1, where we also report the F-measure-based rankings of the methods for each mixture model.

Focusing on summary indices such as the F-measures, the results in Table 4.E.1 show that:

- The proposed type-oriented methods are always among the four best methods. More in detail, MUOD$_{mag}$ is the third best method in presence of magnitude outliers, MUOD$_{amp}$ is the best method in presence of amplitude outliers, which are probably the most challenging outliers to detect, and MUOD$_{sha}$ is the fourth best method in presence of shape outliers. Therefore, one the main novelties of our methods, that is, their type-orientation, it is supported by good and stable results for the considered mixture models.

Table 4.E.1: Correct outlier detection percentages (c), false outlier detection percentages (f), F-measures (F) and F-measure-based rankings of the methods (r) in mixture models 1, 2 and 3 which allow for magnitude (mag), amplitude (amp) and shape (sha) outliers, respectively.

	mag				amp				sha			
	c	f	F	r	c	f	F	r	c	f	F	r
B_{tri}	54.55	0.00	0.71	6	16.67	0.01	0.29	9	83.82	0.00	0.91	2
B_{wei}	98.42	0.05	0.98	1	25.00	0.01	0.40	8	100.00	0.00	1.00	1
FBAG	3.16	0.27	0.06	10	91.67	0.46	0.91	2	8.29	0.24	0.14	11
FHDR	15.61	4.43	0.16	9	75.97	1.14	0.77	6	24.08	3.96	0.24	10
FBPLOT	39.13	0.00	0.56	8	0.39	0.00	0.00	11	64.55	0.00	0.79	9
OG	0.00	0.00	-	-	0.78	0.00	0.02	10	0.00	0.00	-	-
$KFSD_{smo}$	98.81	0.09	0.98	1	82.17	0.11	0.89	3	84.39	0.13	0.90	3
$KFSD_{tri}$	99.60	2.51	0.81	4	96.90	2.35	0.81	5	99.23	2.45	0.81	6
$KFSD_{wei}$	100.00	2.71	0.80	5	97.48	2.13	0.82	4	99.81	2.66	0.80	7
MUOD	96.05	5.84	0.63	7	96.71	6.54	0.61	7	95.18	1.60	0.84	5
$MUOD_{mag}$	95.85	0.50	0.93	3	0.00	2.21	-	-	68.98	0.16	0.80	7
$MUOD_{amp}$	0.59	0.93	0.01	12	96.71	0.62	0.93	1	4.62	0.98	0.08	12
$MUOD_{sha}$	4.94	4.79	0.05	11	0.00	5.62	-	-	83.04	0.50	0.86	4

- MUOD, which is the method obtained by the simultaneous use of $MUOD_{mag}$, $MUOD_{amp}$ and $MUOD_{sha}$ on a single data set, has a remarkably good performance since it is always among the seven best methods. Clearly, for the considered mixture models, MUOD slightly suffers in terms of false outlier detection percentages since it searches for all type of outliers while all the simulated data sets contains (eventually) only one type of outliers.

- Concerning our competitors: B_{wei} and $KFSD_{smo}$ perform better than $MUOD_{mag}$ in presence of magnitude outliers, however these are methods that entail important computational limitations; the same methods and B_{tri} perform better than $MUOD_{sha}$ in presence of shape outliers, but the remark on the computational limitations still holds; FBAG is the best competitor of $MUOD_{amp}$ in presence amplitude outliers, however it shows poor performances in presence of magnitude and shape outliers.

4.E.2. Mixture Model 4

In Figure 4.E.5 we show a simulated data set obtained using a mixture of models (4.6), (4.7), (4.8) and (4.9) (mixture model 4) with $\alpha = 0.05$ being the probability that each curve is an outlier and $\frac{1}{3}$ being the probability that each outlier is a magnitude, amplitude and shape outlier, respectively.

For mixture model 4 the comparison among methods is based on 300 generated data sets and is performed as in Table 4.E.1. The results are shown in Table 4.E.2.

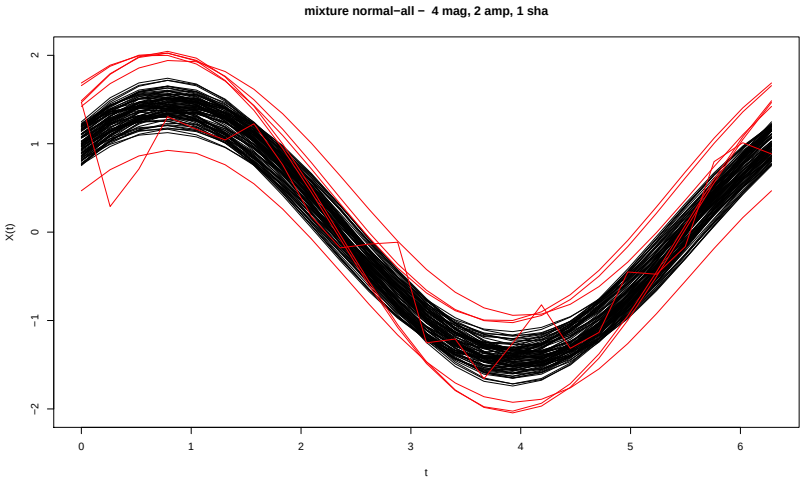


Fig. 4.E.5: Mixture model 4: magnitude, amplitude and shape outliers.

Table 4.E.2: Correct outlier detection percentages (c), false outlier detection percentages (f), F-measures (F) and F-measure-based rankings of the methods (r) in the mixture model 4 which allows simultaneously for magnitude, amplitude and shape outliers.

	$mag-amp-sha$			
	c	f	F	r
B_{tri}	61.81	0.00	0.77	6
B_{wei}	96.21	0.00	0.98	1
FBAG	35.32	0.26	0.50	9
FHDR	42.32	3.00	0.42	11
FBPLOT	34.92	0.00	0.52	8
OG	0.52	0.00	0.02	13
$KFSD_{smo}$	82.67	0.14	0.89	2
$KFSD_{tri}$	99.35	2.34	0.82	4
$KFSD_{wei}$	99.80	2.51	0.81	5
MUOD	97.58	2.01	0.83	3
$MUOD_{mag}$	41.33	0.08	0.57	7
$MUOD_{amp}$	34.01	0.37	0.48	10
$MUOD_{sha}$	30.22	1.61	0.37	12

Note that for the type-oriented methods, i.e., OG, MUOD_{mag}, MUOD_{amp} and MUOD_{sha}, the correct outlier detection percentages of Table 4.E.2 are difficult to interpret because these methods search for a specific type of outliers. Therefore, to interpret the performances of OG, MUOD_{mag}, MUOD_{amp} and MUOD_{sha} that we observe in Table 4.E.2, it is necessary to decompose the outlier detection problem into three parts. This decomposition is presented in Table 4.E.3. Note that this table has the same structure of Table 4.E.1, but it is associated to a fairly more complex outlier detection problem: in this case the observations and the three types of outliers are generated from a unique mixture model. Clearly, since the non-type-oriented methods search for any type of outliers, their false outlier detection percentages of Table 4.E.3 are also difficult to interpret.

Table 4.E.3: Decomposed correct outlier detection percentages (c), false outlier detection percentages (f), F-measures (F) and F-measure-based rankings of the methods (r) in mixture model 4 allowing simultaneously for magnitude (mag), amplitude (amp) and shape (sha) outliers.

	mag				amp				sha			
	c	f	F	r	c	f	F	r	c	f	F	r
B_{tri}	73.08	1.92	0.52	3	21.40	2.84	0.14	9	89.98	1.65	0.63	1
B_{wei}	100.00	3.23	0.52	3	88.60	3.49	0.45	4	99.80	3.27	0.52	4
FBAG	0.77	2.07	0.01	10	98.40	0.42	0.88	2	8.64	1.94	0.08	11
FHDR	8.65	4.94	0.04	9	98.00	3.42	0.49	3	22.00	4.71	0.11	10
FBPLOT	40.19	1.10	0.39	6	1.00	1.79	0.01	11	62.87	0.73	0.61	3
OG	0.00	0.03	-	-	1.60	0.00	0.04	10	0.00	0.03	-	-
KFSD_{smo}	100.00	2.66	0.57	2	69.80	3.24	0.39	5	77.60	3.09	0.43	5
KFSD_{tri}	100.00	5.64	0.39	6	98.40	5.74	0.37	7	99.61	5.69	0.37	6
KFSD_{wei}	100.00	5.84	0.37	8	99.80	5.91	0.36	8	99.61	5.88	0.37	6
MUOD	100.00	5.23	0.40	5	100.00	5.30	0.39	5	92.73	5.39	0.37	6
MUOD_{mag}	100.00	0.46	0.88	1	1.80	2.19	0.01	11	20.24	1.87	0.18	9
MUOD_{amp}	0.00	2.12	-	-	100.00	0.43	0.89	1	3.93	2.05	0.03	12
MUOD_{sha}	1.35	3.10	0.01	10	0.00	3.12	-	-	89.39	1.58	0.63	1

The results in Tables 4.E.2 and 4.E.3 show that:

- MUOD has a good performance in terms of the F-measures of Table 4.E.2. It is the third best method, and it is slightly outperformed by two computational intensive and non-type-oriented methods such as B_{wei} and KFSD_{smo}.

- The good performance of MUOD is due to the good and complementary performances of MUOD_{mag}, MUOD_{amp} and MUOD_{sha} shown in Table 4.E.3: especially MUOD_{mag} and MUOD_{amp} detect very satisfactorily the outliers for which they have been proposed, whereas the performance of MUOD_{sha} is slightly less efficient in terms of its F-measure.

Since mixture model 4 simultaneously allows magnitude, amplitude and shape outliers, it represents a model able to generate data sets that might resemble our real data application. For

this reason, we use mixture model 4 to generate a data set having a sample size comparable to the sample size that we observe in our real data application and we perform outlier detection on it. We generate a data set having 6,000,000 observations and we compare the performances of FBPLOT with the ones of the proposed procedures. We restrict this comparison to these four procedures for computational reasons (see Figure 4.D.1).

The simulated data set contains 300,305 outliers (roughly 5%) divided in this way: 99,787 magnitude outliers, 100,166 amplitude outliers and 100,352 shape outliers. In Table 4.E.4 we report how many observations are detected by FBPLOT, MUOD$_{mag}$, MUOD$_{amp}$ and MUOD$_{sha}$ as outliers (in the diagonal and in bold), and we also describe the intersection among the outputs of these procedures (outside the diagonal).

Table 4.E.4: Information about FBPLOT, MUOD$_{mag}$, MUOD$_{amp}$ and MUOD$_{sha}$ when used on a data set with 6,000,000 observations generated by mixture model 4.

	FBPLOT	MUOD$_{mag}$	MUOD$_{amp}$	MUOD$_{sha}$
FBPLOT	**6934**	1536	65	3785
MUOD$_{mag}$	1536	**118021**	57	4822
MUOD$_{amp}$	65	57	**92336**	135
MUOD$_{sha}$	3785	4822	135	**26708**

According to Table 4.E.4, MUOD$_{mag}$ and MUOD$_{amp}$ are the two procedures detecting the most outliers, and their outputs in practice do not overlap. MUOD$_{sha}$ detects less outliers and it shows an appreciable overlap with MUOD$_{mag}$ (18.05% of the observations detected as outliers by MUOD$_{sha}$ are also detected as outliers by MUOD$_{mag}$). FBPLOT is the procedure detecting the least outliers and it shows appreciable overlaps with MUOD$_{mag}$ and MUOD$_{sha}$ (22.15% and 54.59% of the observations detected as outliers by $FBPLOT$ are also detected as outliers by MUOD$_{mag}$ and MUOD$_{sha}$, respectively). A graphical illustration of such results is given in Figure 4.E.6.

Tables 4.E.5 and 4.E.6 replicate Tables 4.E.2 and 4.E.3, respectively, but now we only consider FBPLOT, MUOD, MUOD$_{mag}$, MUOD$_{amp}$ and MUOD$_{sha}$ as procedures and their performances are evaluated on a unique big data set (with 6,000,000 observations) instead of on many small data sets (300 data sets with 100 observations). In these new tables we omit the F-measure-based rankings.

The results in Tables 4.E.5 and 4.E.6 when compared to the results in Tables 4.E.2 and 4.E.3 show that:

- MUOD improves its performance in terms of F-measures when we compare Table 4.E.5 with Table 4.E.2. On the contrary, FBPLOT shows an important deterioration of its performance.

- Also MUOD$_{mag}$, MUOD$_{amp}$ and MUOD$_{sha}$ improve their performances in terms of F-measures when we compare Table 4.E.6 with Table 4.E.3.

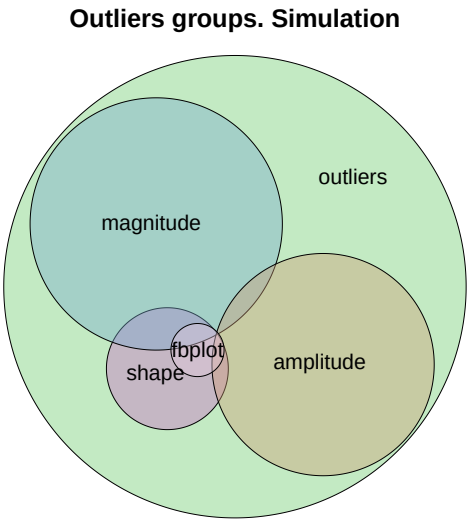


Fig. 4.E.6: Venn's diagram for outliers and algorithm's detected sets. This plot visually summarizes the results in Tables 4.E.4 and 4.E.5.

Table 4.E.5: Correct outlier detection percentages (c), false outlier detection percentages (f) and F-measures (F) in the mixture model 4 which allows simultaneously for magnitude, amplitude and shape outliers. Performances of FBPLOT, MUOD, MUOD$_{mag}$, MUOD$_{amp}$ and MUOD$_{sha}$ on a data set with 6,000,000 observations.

	$mag-amp-sha$		
	c	f	F
FBPLOT	2.31	0.00	0.05
MUOD	77.28	0.00	0.87
MUOD$_{mag}$	39.30	0.00	0.56
MUOD$_{amp}$	30.75	0.00	0.47
MUOD$_{sha}$	8.89	0.00	0.16

Table 4.E.6: Decomposed correct outlier detection percentages (c), false outlier detection percentages (f) and F-measures (F) in mixture model 4 allowing simultaneously for magnitude (mag), amplitude (amp) and shape (sha) outliers. Performances of FBPLOT, MUOD, MUOD$_{mag}$, MUOD$_{amp}$ and MUOD$_{sha}$ on a data set with 6,000,000 observations.

	mag			amp			$shape$		
	c	f	F	c	f	F	c	f	F
FBPLOT	0.06	0.12	0.00	0.00	0.12	-	6.85	0.00	0.13
MUOD	100.00	2.24	0.60	91.92	2.37	0.55	40.08	3.25	0.24
MUOD$_{mag}$	100.00	0.31	0.92	0.00	2.00	-	18.17	1.69	0.17
MUOD$_{amp}$	0.00	1.56	-	91.92	0.00	0.96	0.26	1.56	0.00
MUOD$_{sha}$	0.00	0.45	-	0.00	0.45	-	26.61	0.00	0.42

Therefore, MUOD, MUOD$_{mag}$, MUOD$_{amp}$ and MUOD$_{sha}$ have stable behaviors under mixture model 4 in both small and big sample size scenarios. On the contrary, FBPLOT, experiences a very important deterioration in its outlier detection performance in a big sample size scenario.

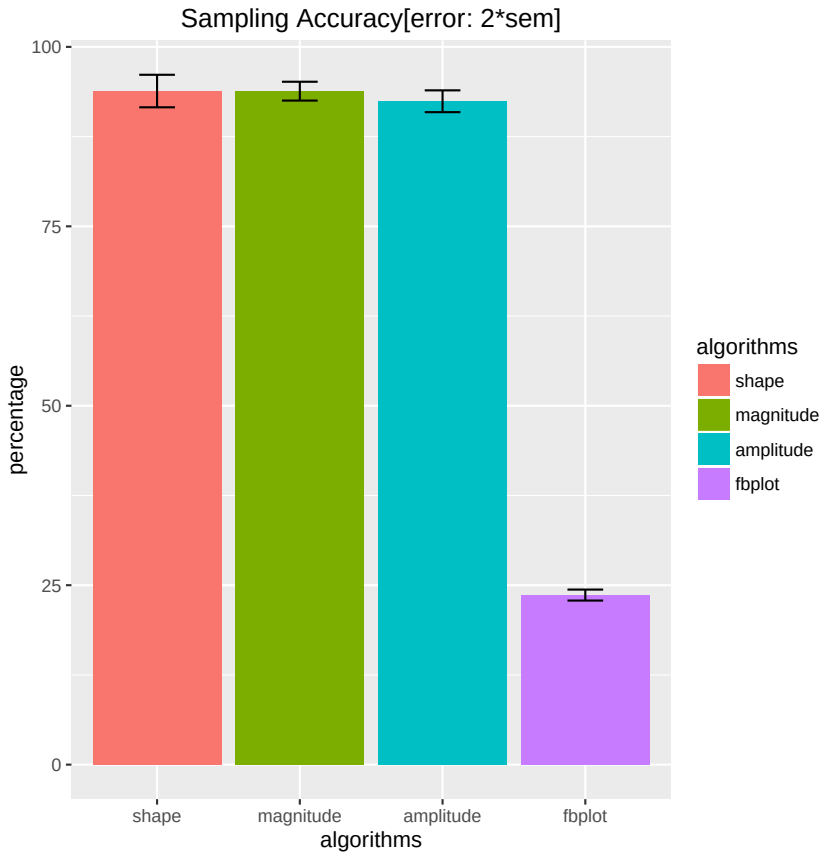


Fig. 4.E.7: Accuracy for the considered algorithms when sampling. The plot shows average accuracy for 100 simulations using random samples of 100K observations. Each bar shows the accuracy as the intersection ratio between the outliers detected on a sample and the outliers on the total population (6M observations) for an algorithm. The bars show that shape, magnitude and amplitude are not affected when sampling (accuracy $\sim$ 90%) while FBPlot seems to be sensitive to observation samples.

We explore now the behavior of the different algorithms applied over samples of a large population instead of over the whole population. For that, we use the population of 6,000,000 observations described above, and take random samples of 100,000 observations. Figures 4.E.7 and 4.E.8 present the accuracy, precision, and recall observed when performing this sampling. This experiment shows that the outliers identified by MUOD in the samples matches in a large degree those identified in the whole population. This is not the case with FBPlot.

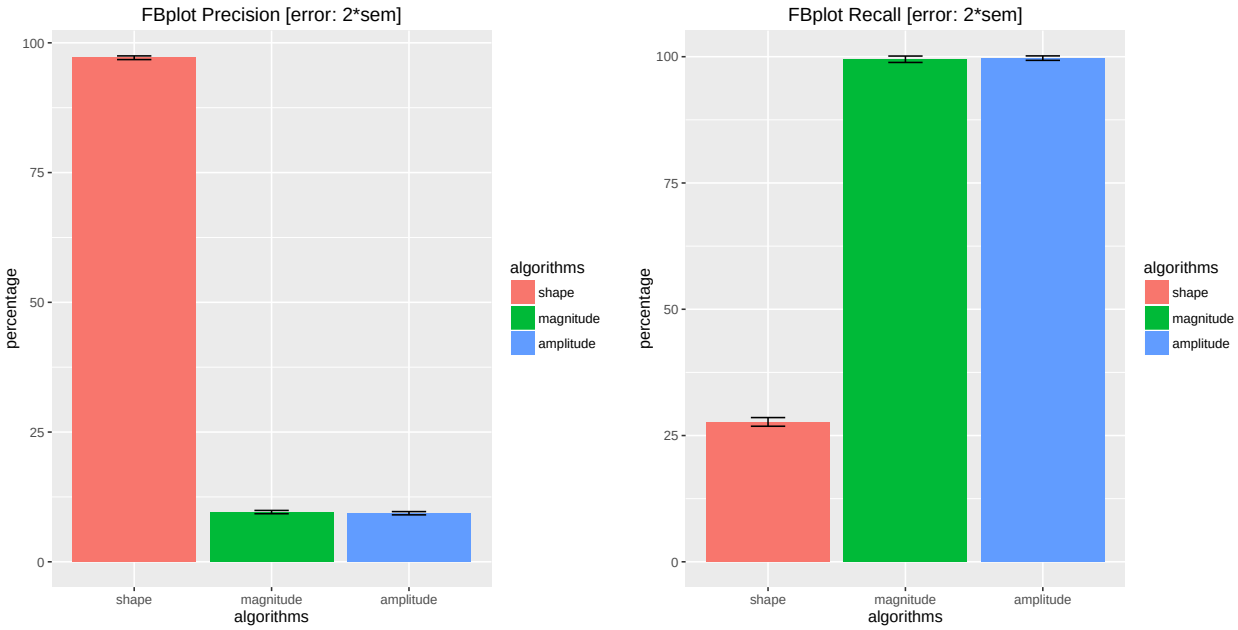


Fig. 4.E.8: Precision and recall for FBPlot against shape, magnitude and amplitude when sampling. The plot shows average precision and recall for 100 simulations using random samples of 100K observations. Each bar shows that there is a considerable overlap between FBPlot and shape, magnitude and amplitude, yet in different ways. While FBPlot and shape overlap in each sample, FBPlot becomes a small subset of shape. On the other hand, FBPlot overlaps with magnitude and amplitude too, but in this case, magnitude and amplitude become small subsets of FBPlot. To sum up, these plots extend Figures 4.E.6 and 4.E.7, and show the FBPlot behavior as comparison with MUOD (shape, magnitude, and amplitude) when sampling.

4.E.3. Mixture Models 5, 6, and 7

We expand our simulation study considering three additional mixture models that were used in Arribas-Gil and Romo, 2014 to evaluate OG. We refer to them as mixture model 5, 6 and 7, respectively. Since OG was proposed to expressly detect shape outliers, these mixture models generate different types of shape outliers, which are also the type of outliers that until now our procedures seem to have the most problems to detect.

- In mixture model 5 the main model is given by

$$X(t) = 30t(1-t)^{3/2} + \epsilon(t),$$

where $t \in [0,1]$ and $\epsilon(t)$ is a Gaussian process with zero mean and covariance function $\gamma(s,t) = 0.3\exp\{-|s-t|/0.3\}$, while the contamination model is given by

$$X(t) = 30t^{3/2}(1-t) + \epsilon(t).$$

- In mixture model 6 the main model is given by

$$X(t) = 4t + \epsilon(t), \tag{4.11}$$

where $t \in [0,1]$ and $\epsilon(t)$ is a Gaussian process with zero mean and covariance function $\gamma(s,t) = \exp\{-|s-t|\}$, while the contamination model is given by

$$X(t) = 4t + (-1)^u 1.8 + \frac{1}{\sqrt{2\pi 0.01}}\exp\left\{-(t-\mu)^2/0.02\right\} + \epsilon(t),$$

where u follows a Bernoulli distribution with probability $1/2$ and μ is uniformly distributed in $[0.25, 0.75]$.

- In mixture model 7 the main model is given by (4.11), with t and $\epsilon(t)$ defined as in mixture model 6, while the contamination model is given by

$$X(t) = 4t + 2\sin(4(t+\mu)\pi) + \epsilon(t),$$

with μ defined as in mixture model 6.

In Figure 4.E.9 we show three simulated data sets obtained using mixture models 5, 6 and 7 with $d = 50$ equidistant points between 0 and 1, $n = 100$ and $\alpha = 0.05$ being the probability that each curve is an outlier in each model.

The comparison among methods for mixture models 5, 6 and 7 is performed similarly as for the previous four models, and it is reported in Table 4.E.7.

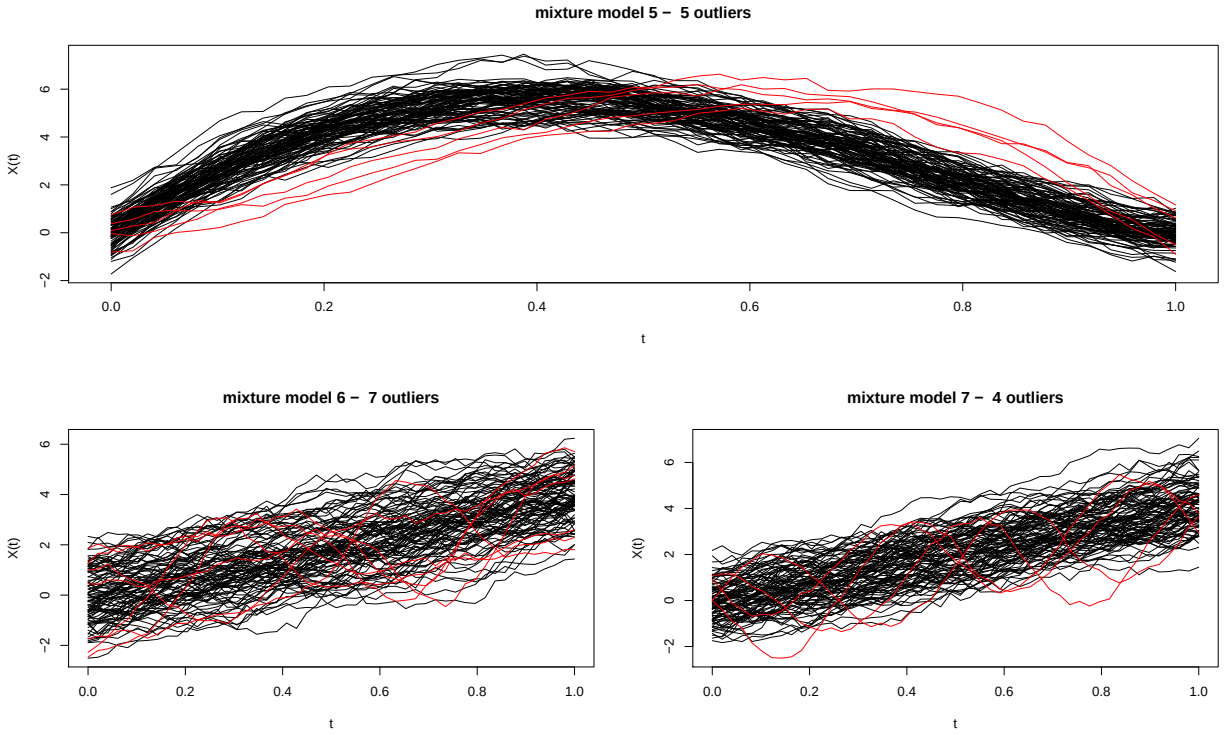


Fig. 4.E.9: Mixture models 5 (top), 6 (bottom left) and 7 (bottom right).

Table 4.E.7: Correct outlier detection percentages (c), false outlier detection percentages (f), F-measures (F) and F-measure-based rankings of the methods (r) in mixture models 5, 6 and 7.

	mix mod 5				*mix mod 6*				*mix mod 7*			
	c	f	F	r	c	f	F	r	c	f	F	r
B_{tri}	48.23	0.03	0.65	10	26.20	0.59	0.38	8	23.55	0.79	0.34	8
B_{wei}	88.08	0.41	0.90	2	30.59	0.71	0.43	7	23.55	0.76	0.35	7
FBAG	99.63	6.70	0.63	11	36.14	7.00	0.27	9	8.58	7.86	0.06	10
FHDR	65.74	1.55	0.68	8	23.71	3.97	0.24	10	5.59	4.97	0.06	10
FBPLOT	26.44	0.01	0.41	13	0.19	0.00	0.00	13	0.40	0.02	0.00	13
OG	97.95	2.31	0.82	4	98.85	3.36	0.76	1	100.00	3.92	0.73	1
$KFSD_{smo}$	84.92	0.55	0.87	3	49.52	3.05	0.48	5	45.71	3.67	0.43	5
$KFSD_{tri}$	98.14	4.36	0.71	7	79.54	6.29	0.54	4	82.04	6.33	0.55	3
$KFSD_{wei}$	99.26	5.27	0.68	8	86.81	7.02	0.56	3	88.22	7.22	0.54	4
MUOD	95.53	3.42	0.75	5	67.30	6.91	0.46	6	67.66	7.47	0.44	5
$MUOD_{mag}$	47.49	1.65	0.53	12	9.37	3.17	0.11	12	1.00	3.92	0.01	12
$MUOD_{amp}$	72.63	1.60	0.72	6	14.53	3.49	0.17	11	6.19	3.75	0.07	9
$MUOD_{sha}$	92.74	0.53	0.92	1	60.04	2.24	0.60	2	66.07	2.15	0.64	2

The results in Table 4.E.7 show that:

- As expected, for the considered mixture models, MUOD_{sha} is the appropriate method among the proposed methods. Moreover, it is always among the two best methods for these mixture models generating shape outliers.

- MUOD has also a good performance since it is always among the six best methods and this is due to the fact that both MUOD_{mag} and MUOD_{amp} do not detect many false outliers.

- The best competitors for MUOD_{sha} and MUOD are B_{wei}, the KFSD-based techniques and OG. For B_{wei} and the KFSD-based techniques hold the remarks made above. For OG, its good performances in mixture models 5, 6 and 7 contrast with its poor performances in mixture models 1, 2 , 3 and 4, and therefore OG does not represent a real competitor for our proposals.

After comparing our methods with a wide set of competitors in standard FDA simulated scenarios and showing their good performances, in the next section we present some supplementary details on our real data application. As for the 6,000,000 observations run of mixture model 4, also in this case we restrict the analysis to the proposed procedures and FBPLOT.

4.F. Outlier Detection in our Real Data Application

In this section we present some remarks on the outlier detection in our real data application. Recall that our real data set describes 21 features of 5,619,786 users of Google+ and that, after discarding 2 features, we treat it as a functional data set.

Table 4.F.2 replicates Table 4.E.4: first, we report how many observations are detected by FBPLOT, MUOD_{mag}, MUOD_{amp} and MUOD_{sha} as outliers (in the diagonal and in bold); second, we describe the interaction among the outputs of these procedures (outside the diagonal).

Note that according to FBPLOT, MUOD_{mag}, MUOD_{amp} and MUOD_{sha} the 0.29%, 0.11%, 0.11% and 5.38% of the observations are outliers, respectively. Moreover,

- All the observations that are detected as outliers by FBPLOT, except 18, are also detected as outliers by MUOD_{sha}.

- All the observations that are detected as outliers by MUOD_{mag} and MUOD_{amp} are also detected as outliers by MUOD_{sha}.

Therefore, according to the proposed procedures in our real data application if an observation is detected as an outlier, it is at least a shape outlier. Due to this result, let us define the following types of observations:

Table 4.F.1: Dataset variables description.

Variable	Description
id	The user identifier in Google+
NumActivities	The total number of activities (posts) of a user
NumAttachments	Total number of attachments included in the posts made by the user
NumPlusOnes	The total number of '+1' the user's posts have received
NumReplies	The total number of replies the user's posts have received
NumReshares	The total number of reshares the user's posts have been done
NumFriends	The number of friends the user has
NumFollowers	The number of followers the user has
NumFields	Number of fields the user made public in the profile
PercentageBidirectional	Percentage of friends which are following each other
accountAge	Number of days since first time we say the account active (i.e., first post), measured from the data collection instant
accountRecency	Number of days since last post, measured from the data collection instant
gender	The gender declared by the user
job	The job declared by the user
numVideos	The number of videos the user has published
numPhotos	The number of photos the user has published
numAlbums	The number of albums the user has
numArticles	The number of articles the user has published
numHangouts	Number of hangouts made by the user
numEvents	Number of events the user participated in
numWithGeo	Number of posts with geographical information
pageRank	The topological pageRank value of the user considering the whole network (unweighted)

Table 4.F.2: Information about FBPLOT, MUOD_{mag}, MUOD_{amp} and MUOD_{sha} when used on our real data application.

	FBPLOT	MUOD_{mag}	MUOD_{amp}	MUOD_{sha}
FBPLOT	**16140**	4366	6074	16122
MUOD_{mag}	4366	**6103**	4036	6103
MUOD_{amp}	6074	4036	**6178**	6178
MUOD_{sha}	16122	6103	6178	**302345**

1. *no MUOD*: observations that are normal according to MUOD_{mag}, MUOD_{amp} and MUOD_{sha}.

2. *only sha*: observations that are detected as outliers by MUOD_{sha} and are normal observations according to MUOD_{mag} and MUOD_{amp}.

3. *only mag+sha*: observations that are detected as outliers by MUOD_{sha} and MUOD_{mag} and are normal observations according MUOD_{amp}.

4. *only amp+sha*: observations that are detected as outliers by MUOD_{sha} and MUOD_{amp} and

are normal observations according MUOD_{amp}.

5. *mag+amp+sha*: observations that are detected as outliers by MUOD_{sha}, MUOD_{mag} and MUOD_{amp}.

Note that the all the observations in our real data set belong exclusively to one of the classes of above. In Table 4.F.3 we describe how the observations are distributed in these classes (in absolute and relative terms).

Table 4.F.3: Distribution of the observations of our real data application in the classes *no MUOD*, *only sha*, *only mag+sha*, *only amp+sha* and *mag+amp+sha*. Absolute (abs) and relative (rel) frequencies.

	abs	rel
no MUOD	5317441	0.9462
only sha	294100	0.0523
only mag+sha	2067	0.0004
only amp+sha	2142	0.0004
mag+amp+sha	4036	0.0007
totals	5619786	1.0000

In Figure 4.F.1 we represent the observations as functional data[**]:

1. a 1% of the 294100 *only sha* observations (top left).
2. the 2067 *only mag+sha* observations (top right).
3. the 2142 *only amp+sha* observations (bottom left).
4. the 4036 *only mag+amp+sha* observations (bottom right).

Observing Figure 4.F.1 we notice that the four classes of observations detected as outliers by the proposed procedures would make functional data sets different among them. Such a result seems to justify the detection of four different types of outliers. Moreover, thanks to Figure 4.F.2, where we represent 5000 *no MUOD* observations as functional data, it is possible to appreciate another important fact, i.e., there are differences between the observations detected as outliers and the observations that are normal according to the proposed procedures. For more qualitative details on these differences, see the main article.

Once defined the classes *no MUOD*, *only sha*, *only mag+sha*, *only amp+sha* and *mag+amp+sha*, and before closing this section, in Tables 4.F.4 and 4.F.5 we show their distributions against two classes related with the FBPLOT procedure: *no FBPLOT* and *yes FBPLOT*, the observations that are normal and outlying according to FBPLOT, respectively.

According to Tables 4.F.4 and 4.F.5, FBPLOT in practice does not detect as outliers those observations that were not detected as outliers by the proposed procedures. Moreover, the

[**]We use a logarithmic transformation that avoids the argument of the logarithm to be non-positive.

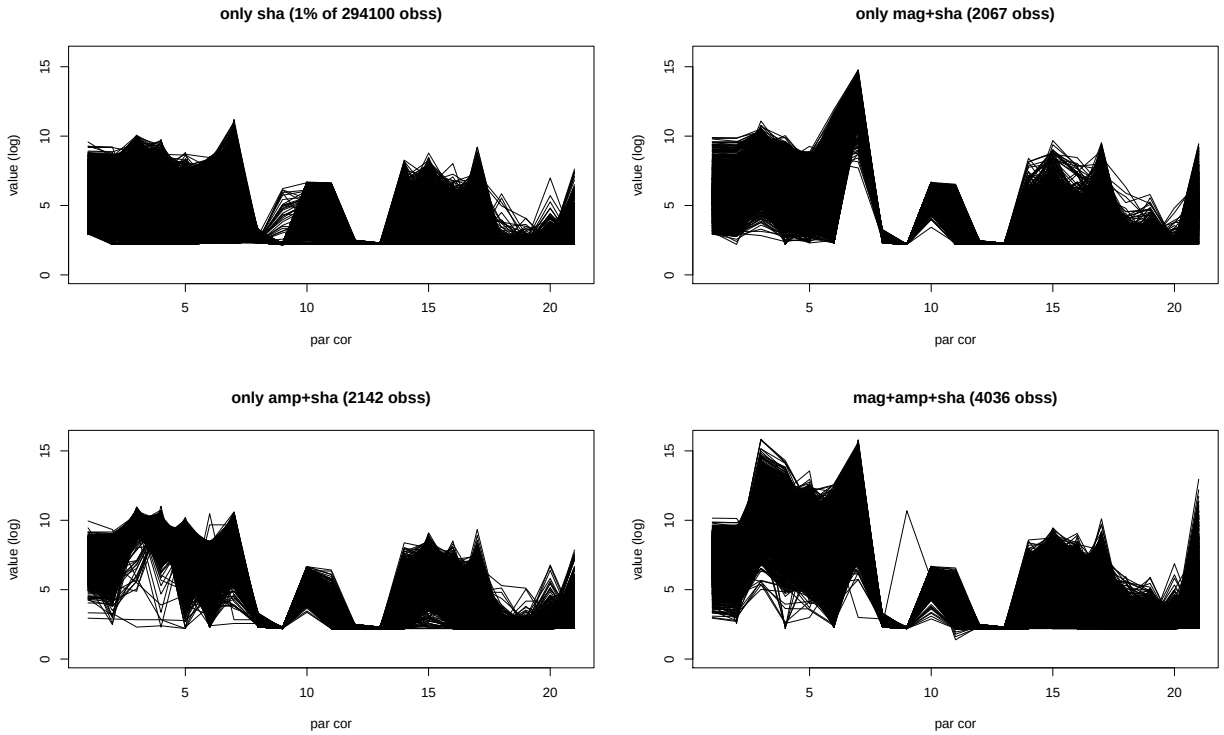


Fig. 4.F.1: 1% *only sha*, *only mag+sha*, *only amp+sha* and *mag+amp+sha* observations (from top left, in clockwise order).

Table 4.F.4: Distribution of the observations of our real data application in the classes *no MUOD*, *only sha*, *only mag+sha*, *only amp+sha* and *mag+amp+sha* against the classes *no FBPLOT* and *yes FBPLOT*. Absolute frequencies.

	no FBPLOT	*yes FBPLOT*	*totals*
no MUOD	5317423	18	5317441
only sha	284482	9618	294100
only mag+sha	1637	430	2067
only amp+sha	4	2138	2142
mag+amp+sha	100	3936	4036
totals	5603646	16140	5619786

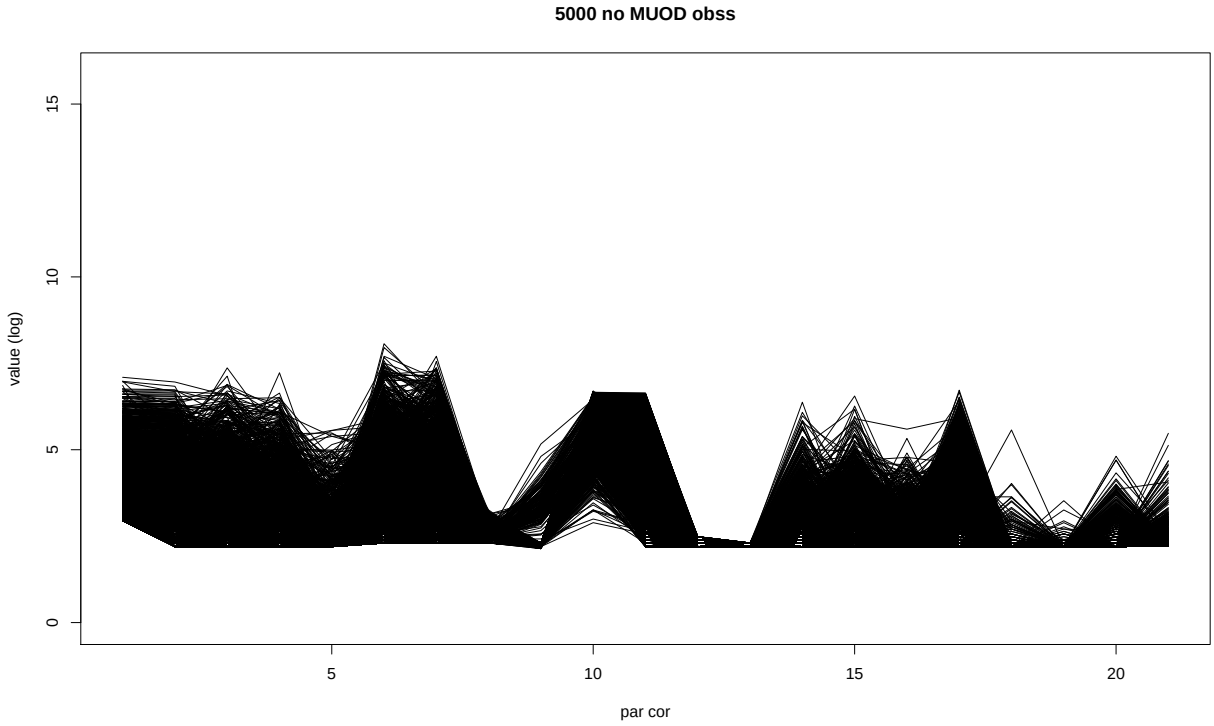


Fig. 4.F.2: 5000 *no MUOD* observations.

Table 4.F.5: Distribution of the observations of our real data application in the classes*no MUOD*, *only sha*, *only mag+sha*, *only amp+sha* and *mag+amp+sha* against the classes *no FBPLOT* and *yes FBPLOT*. Relative frequencies inside the classes *no MUOD*, *only sha*, *only mag+sha*, *only amp+sha* and *mag+amp+sha*.

	no FBPLOT	*yes FBPLOT*	totals
no MUOD	1.0000	0.0000	1
only sha	0.9673	0.0327	1
only mag+sha	0.7920	0.2080	1
only amp+sha	0.0019	0.9981	1
mag+amp+sha	0.0248	0.9752	1

99.81% and 97.52% of *only amp+sha* and *mag+amp+sha* observations are outlying also according to FBPLOT, respectively. The degree of concordance between the proposed procedures and FBPLOT falls to 20.80% and 3.27% in the classes *only mag+sha* and *only sha*, respectively. Therefore, the main differences between proposed procedures and FBPLOT can be observed in the classes *only mag+sha* and especially *only sha*.

In Figures 4.F.3-4.F.5 we represent as functional data:

1. Figure 4.F.3: a 1% of the 284482 *(only sha, no FBPLOT)* observations (left) and a 25% of the *(only sha, yes FBPLOT)* observations (right).

2. Figure 4.F.4: the 1637 *(only mag+sha, no FBPLOT)* observations (left) and the 430 *(only mag+sha, yes FBPLOT)* observations (right).

3. Figure 4.F.5: the 100 *(mag+amp+sha, no FBPLOT)* observations (left) and the 3936 *(mag+amp+sha, yes FBPLOT)* observations (right).

In Figures 4.F.3-4.F.5 we also draw the upper bound of the non-outlying region defined by FBPLOT.

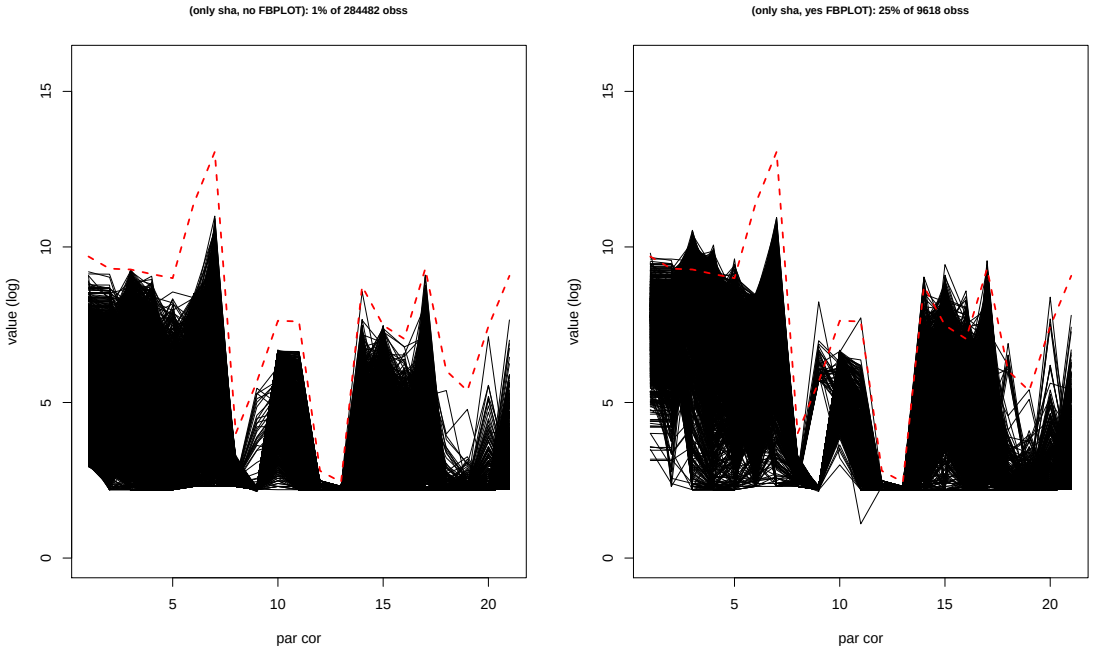


Fig. 4.F.3: 1% *(only sha, no FBPLOT)* observations (left); 25% *(only sha, yes FBPLOT)* observations (right). Dashed line in both figures: upper bound of the non-outlying region defined by FBPLOT.

Observing Figures 4.F.3-4.F.5 we notice that the proposed procedures detect as outliers more observations than FBPLOT, especially as *only sha* outliers. These observations, when compared

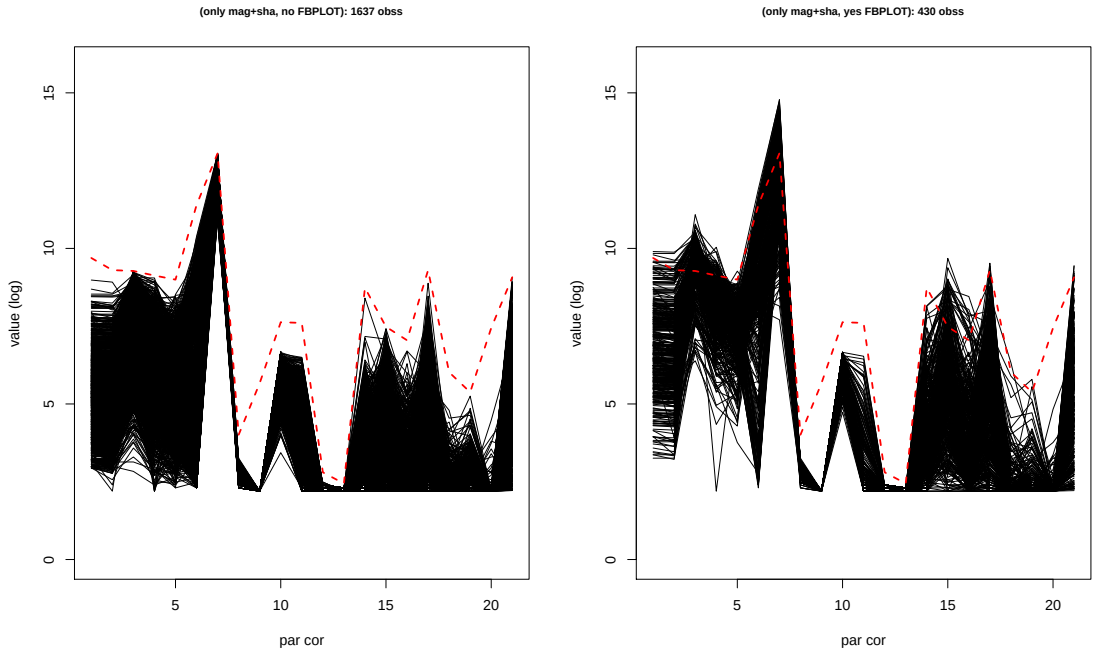


Fig. 4.F.4: *(only mag+sha, no FBPLOT)* observations (left); *(only mag+sha, yes FBPLOT)* observations (right). Dashed line in both figures: upper bound of the non-outlying region defined by *FBPLOT*.

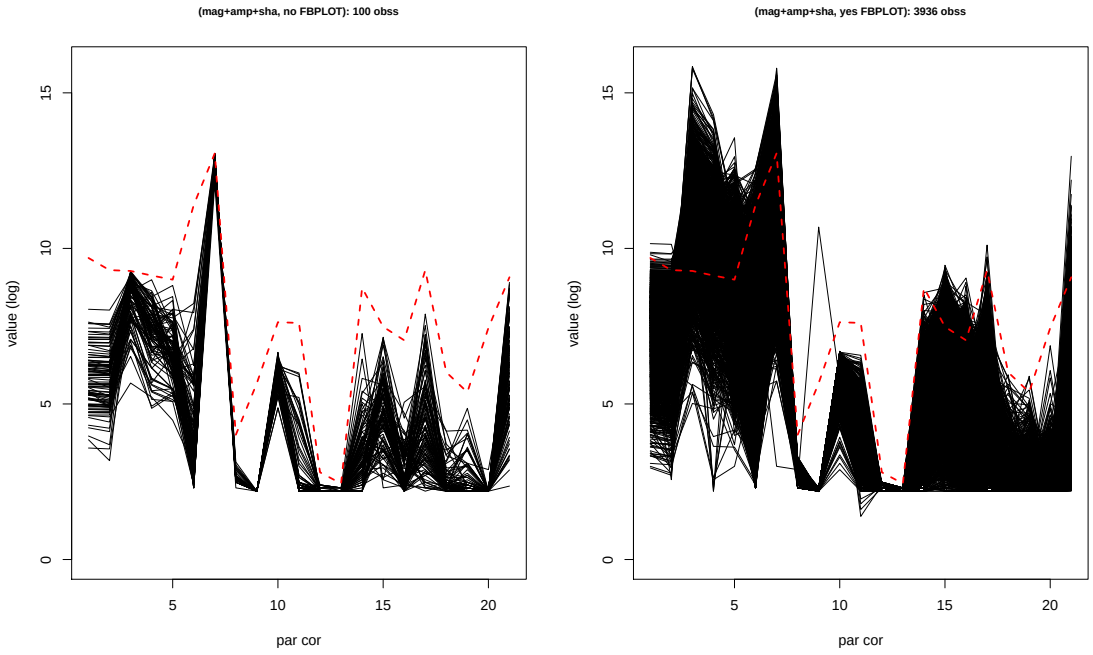


Fig. 4.F.5: *(mag+amp+sha, no FBPLOT)* observations (left); *(mag+amp+sha, yes FBPLOT)* observations (right). Dashed line in both figures: upper bound of the non-outlying region defined by FBPLOT.

to some of the observations that are normal according to the proposed procedures (Figure 4.F.2), show a behavior that seems to justify their detection as outliers. For more qualitative details on the differences between MUOD and FBPLOT, see the main article.

References

Arribas-Gil, A. and J. Romo (2014). «Shape outlier detection and visualization for functional data: the outliergram». In: *Biostatistics* 15, pp. 603–619.

Arruda, Guilherme Ferraz de et al. (2014). «Role of centrality for the identification of influential spreaders in complex networks». In: *Physical Review E* 90.3, p. 032812.

Bakshy, Eytan et al. (2011). «Identifying influencers on twitter». In: *Fourth ACM International Conference on Web Seach and Data Mining (WSDM)*.

Bapna, Ravi and Akhmed Umyarov (2015). «Do your online friends make you pay? A randomized field experiment on peer influence in online social networks». In: *Management Science* 61.8, pp. 1902–1920.

Basaras, Pavlos, Dimitrios Katsaros, and Leandros Tassiulas (2013). «Detecting Influential Spreaders in Complex, Dynamic Networks». In: *Computer* 46.4, pp. 24–29. ISSN: 0018-9162. DOI: http://doi.ieeecomputersociety.org/10.1109/MC.2013.75.

Cha, Meeyoung et al. (2010). «Measuring User Influence in Twitter: The Million Follower Fallacy.» In: *ICWSM* 10.10-17, p. 30.

Chen, Yixin et al. (2009). «Outlier detection with the kernelized spatial depth function». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.2, pp. 288–305.

Cohen, Reuven et al. (2001). «Breakdown of the Internet under intentional attack». In: *Physical review letters* 86.16, p. 3682.

Cuevas, A., M. Febrero, and R. Fraiman (2006). «On the Use of the Bootstrap for Estimating Functions With Functional Data». In: *Computational Statistics and Data Analysis* 51, pp. 1063–1074.

Cuevas, Antonio (2014). «A Partial Overview of the Theory of Statistics With Functional Data». In: *Journal of Statistical Planning and Inference* 147, pp. 1–23.

D'Agostino, Gregorio et al. (2015). «Interests diffusion in social networks». In: *Physica A: Statistical Mechanics and its Applications* 436, pp. 443–461.

Domingos, Pedro M. and Matthew Richardson (2001). «Mining the network value of customers». In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, San Francisco, CA, USA, August 26-29, 2001*, pp. 57–66. URL: http://portal.acm.org/citation.cfm?id=502512.502525.

Febrero, M., P. Galeano, and W. González-Manteiga (2008). «Outlier Detection in Functional Data by Depth Measures, With Application to Identify Abnormal NOx Levels». In: *Environmetrics* 19, pp. 331–345.

Ferraty, F. and P. Vieu (2006). *Nonparametric Functional Data Analysis : Theory and Practice*. New York: Springer.

Gonzalez, Roberto et al. (2016). «Assessing the Evolution of Google+ in Its First Two Years». In: *IEEE/ACM Trans. Netw.* 24.3, pp. 1813–1826. DOI: 10.1109/TNET.2015.2433792. URL: http://dx.doi.org/10.1109/TNET.2015.2433792.

Guo, Lei et al. (2009). «Analyzing patterns of user content generation in online social networks». In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pp. 369–378.

Horváth, L. and P. Kokoszka (2012). *Inference for Functional Data With Applications*. New York: Springer.

Hubert, M., P. J. Rousseeuw, and P. Segaert (2015). «Multivariate Functional Outlier Detection». In: *Statistical Methods and Applications* 24, pp. 177–202.

Hyndman, R. J. and H. L. Shang (2010). «Rainbow Plots, Bagplots, and Boxplots for Functional Data». In: *Journal of Computational and Graphical Statistics* 19, pp. 29–45.

Jin, Long et al. (2013). «Understanding user behavior in online social networks: A survey». In: *IEEE Communications Magazine* 51.9, pp. 144–150.

Kempe, David, Jon M. Kleinberg, and Éva Tardos (2015). «Maximizing the Spread of Influence through a Social Network». In: *Theory of Computing* 11, pp. 105–147. DOI: 10.4086/toc.2015.v011a004. URL: https://doi.org/10.4086/toc.2015.v011a004.

Kitsak, Maksim et al. (2010). «Identification of influential spreaders in complex networks». In: *Nature Physics* 6.11, pp. 888–893.

Lazar, Nicole (2008). *The statistical analysis of functional MRI data*. Springer Science & Business Media.

Leskovec, Jure et al. (2007). «Patterns of cascading behavior in large blog graphs». In: *Proceedings of the 2007 SIAM international conference on data mining*. SIAM, pp. 551–556.

Lindquist, Martin A (2008). «The statistical analysis of fMRI data». In: *Statistical Science*, pp. 439–464.

Liu, Shixi et al. (2015). «Identifying effective influencers based on trust for electronic word-of-mouth marketing: A domain-aware approach». In: *Information Sciences* 306, pp. 34–52.

López-Pintado, S. and J. Romo (2009). «On the Concept of Depth for Functional Data». In: *Journal of the American Statistical Association* 104, pp. 718–734.

— (2011). «A Half-Region Depth for Functional Data». In: *Computational Statistics and Data Analysis* 55, pp. 1679–1695.

Louail, Thomas et al. (2014). «From mobile phone data to the spatial structure of cities». In: *Scientific Reports*. DOI: 10.1038/srep05276.

Monti, Martin M (2011). «Statistical analysis of fMRI time-series: a critical review of the GLM approach». In: *Frontiers in human neuroscience* 5.

Morone, Flaviano and Hernán A Makse (2015). «Influence maximization in complex networks through optimal percolation». In: *Nature* 524.7563, pp. 65–68.

Pastor-Satorras, Romualdo and Alessandro Vespignani (2001). «Epidemic spreading in scale-free networks». In: *Physical review letters* 86.14, p. 3200.

Poline, Jean-Baptiste and Matthew Brett (2012). «The general linear model and fMRI: does love last forever?» In: *Neuroimage* 62.2, pp. 871–880.

Ramsay, J. O. and B. W. Silverman (2005). *Functional Data Analysis*. New York: Springer.

Reimann, Clemens and Peter Filzmoser (2000). «Normal and lognormal data distribution in geochemistry: death of a myth. Consequences for the statistical treatment of geochemical and environmental data». In: *Environmental geology* 39.9, pp. 1001–1014.

Rijsbergen, C. J. Van (1979). *Information Retrieval*. 2nd. Newton, MA, USA: Butterworth-Heinemann. ISBN: 0408709294.

Sguera, C., P. Galeano, and R. Lillo (2014). «Spatial Depth-Based Classification for Functional Data». In: *TEST* 23, pp. 725–750.

Sguera, Carlo, Pedro Galeano, and Rosa E Lillo (2016). «Functional outlier detection by a local depth with application to NOx levels». In: *Stochastic environmental research and risk assessment* 30.4, pp. 1115–1130.

Simmie, Donal, Maria Grazia Vigliotti, and Chris Hankin (2014). «Ranking twitter influence by combining network centrality and influence observables in an evolutionary model». In: *Journal of Complex Networks* 2.4, pp. 495–517.

Sun, Y. and M. G. Genton (2011). «Functional Boxplots». In: *Journal of Computational and Graphical Statistics* 20, pp. 316–334.

Templ, Matthias, Peter Filzmoser, and Clemens Reimann (2008). «Cluster analysis applied to regional geochemical data: problems and possibilities». In: *Applied Geochemistry* 23.8, pp. 2198–2213.

Tukey, J. W. (1975). «Mathematics and the Picturing of Data». In: *Proceedings of the International Congress of Mathematicians*. Vol. 2, pp. 523–531.

Wegman, E. J. (1990). «Hyperdimensional Data Analysis Using Parallel Coordinates». In: *Journal of American Statistical Association* 85, pp. 664–675.

Zeng, Yong et al. (2015). «Aberrant gene expression in humans». In: *PLoS genetics* 11.1, e1004942.

Chapter 5

Paper 4: ATMoN: Adapting the "Temporality" in Large-Scale Dynamic Networks

Pubished in: *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*

Demetris Trihinas[1] **Luis F. Chiroque**[2] **George Pallis**[1] **Antonio Fernández Anta**[2] **Marios D. Dikaiakos**[1]

[1] Department of Computer Science, University of Cyprus
Email: { trihinas, gpallis, mdd }@cs.ucy.ac.cy
[2] IMDEA Networks Institute, 28918 Leganés, Spain
Email: { lf.chiroque, antonio.fernandez }@imdea.org

Abstract With the widespread adoption of temporal graphs to study fast evolving interactions in dynamic networks, attention is needed to provide graph metrics in time and at scale. In this paper, we introduce ATMoN, an open-source library developed to computationally offload graph processing engines and ease the communication overhead in dynamic networks over an unprecedented wealth of data. This is achieved, by efficiently adapting, in place and inexpensively, the temporal granularity at which graph metrics are computed based on runtime knowledge captured by a low-cost probabilistic learning model capable of approximating both the metric stream evolution and the volatility of the graph topology. After a thorough evaluation with real-world data from mobile, face-to-face and vehicular networks, results show that ATMoN is able to reduce the compute overhead by at least 76%, data volume by 60% and overall cloud costs by at least 54%, while always maintaining accuracy above 88%.

Acknowledgement This work is partially supported by the Regional Government of Madrid (CM) grant Cloud4BigData (S2013/ICE-2894) co-funded by FSE & FEDER, the EU Commission in terms of the H2020 projects Unicorn (IA 731846) and RECAP (RIA 732667), and the NSF of China grant 61520106005.

Keywords Dynamic Networks, Edge Computing, Temporal Graphs, Adaptive Monitoring

5.1. Introduction

For a diverse set of applications, graphs have been used extensively to model links among entities in dynamic networks Leskovec, Kleinberg, and Faloutsos, 2005. The network structure describing how the graph is wired allows us to study, predict and optimize the behavior of dynamic systems Holme and Saramaki, 2012. However, with the widespread adoption of social networks to depict digital interactions among “friendship" links Efstathiades et al., 2016, neural networks to chain metabolic reactions Masuda, Klemm, and Eguíluz, 2013 and the prevalence of the Internet of Things to monitor the physical world Traub et al., 2017, the dynamics shaping network evolution yield the need to appraise “time" in graph modeled networks.

Dynamic networks are modeled as time-ordered sequences of graphs in which links are short-lived and span only through the duration of the interaction between the nodes (e.g., online chat, email, phone call) Kostakos, 2009. In such graphs, the concepts of node adjacency and reachability crucially depend on the exact temporal ordering with the adjacency matrix describing the current graph snapshot for a fixed duration of time Wu, Cheng, et al., 2014. In the literature, these graphs are frequently mentioned as time-evolving graphs, time-varying graphs or, simply, temporal graphs A. Li et al., 2017. Hence, temporal graphs have become very popular, featuring the analytic benefits of static graphs, while also improving graph exploration and navigation by retaining all temporal information and interactions.

Monitoring temporal graphs is widely used in a variety of services, such as high-frequency algorithmic stock trading, social network analysis, targeted advertising, and in intelligent transportation services. However, computing complex graph metrics, such as community and distance metrics, is a challenging task Iyer et al., 2016 Ching et al., 2015. In turn, if the nodes of the graph represent data sources distributed across the edges of an actual network, then these sources must timely disseminate and receive processed monitoring data Trihinas, Pallis, and M. Dikaiakos, 2016. As an example, consider a vehicular network where the expectations are that 1GB of telemetry data will be generated by self-driving vehicles every second L. Mearian, n.d. This data must be processed instantly to keep vehicles safe on the road. It may not seem a challenge for one vehicle, but multiply this number by millions of inter-connected vehicles in an urban environment and even powerful organizations equipped with large-scale graph processing engines reach their limits as the velocity of data keeps increasing Wu, Cheng, et al., 2014. Thus, by the time the data reaches the cloud, it will be too late. Instead, data needs to be processed as close to the data

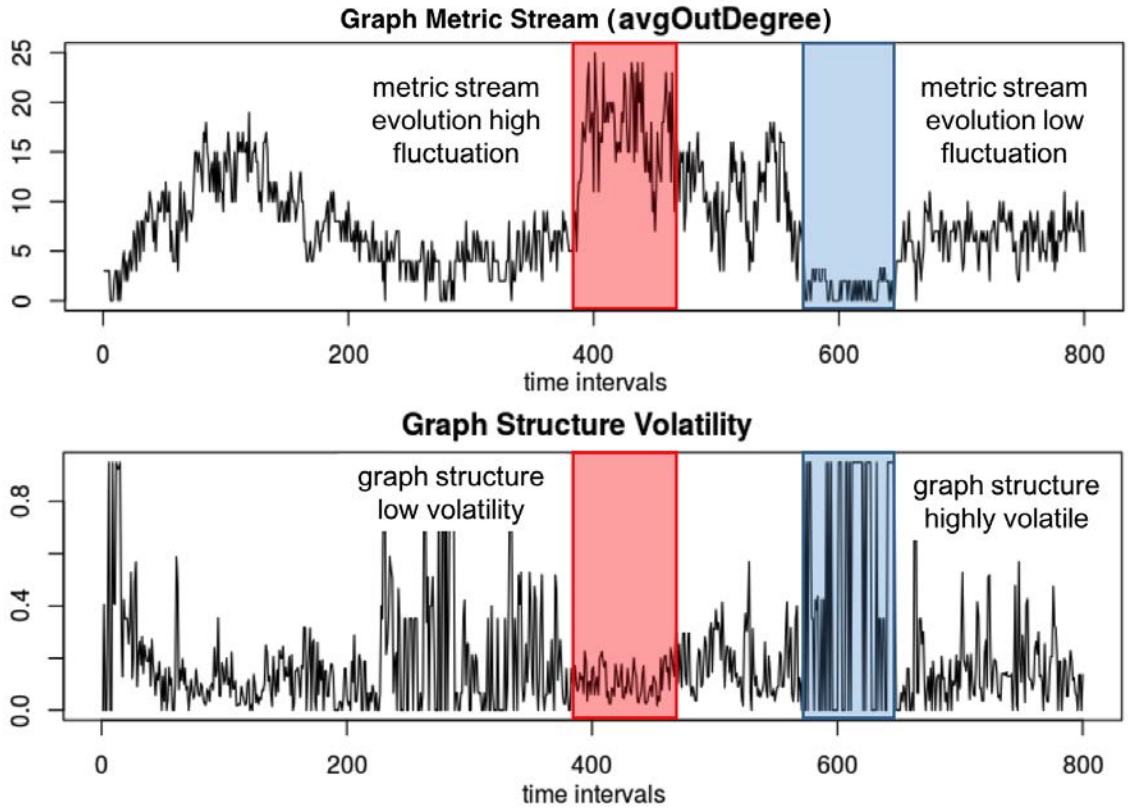


Fig. 5.1: Graph Metric and Topology Structure Volatility for a Mobile Network with Fixed Temporal Granularity

sources as possible, directly at the edge of the network Shi and Dustdar, 2016.

The remedy to the above challenges is to suppress large-scale temporal graphs with approximation techniques Trihinas, Pallis, and M. D. Dikaiakos, 2017. Ideally, an approximation technique dynamically adjusts the rate at which data are processed based on the current data stream evolution, such that when stable phases are detected, the data processing rate is reduced. However, current approximation techniques developed for edge and streaming settings are not suitable for temporal graphs since they do not take into account the dynamicity of the graph topology Fan and Xiong, 2014 Trihinas, Pallis, and M. D. Dikaiakos, 2015. For instance, although the graph metric stream may introduce phases of low variability, the graph topology structure can still be extremely volatile. This is highlighted in Figure 5.1, where although the outgoing connections per node of a mobile network (average out degree) are relatively stable in certain phases, it is actually a different set of nodes interacting in the network Isella et al., 2011. In light of this, ignoring that the particular connections actually span across different locations of the network, can severely affect capacity planning and service provisioning Trihinas, Pallis, and M. D. Dikaiakos, 2017. Thus, while reducing the metric processing rate preserves resources by computationally offloading graph processing engines, it hinders the challenge of missing structural changes in the graph topology which might capture and reveal significant insights.

The focal point of our work is to deliver an adaptive monitoring framework for dynamic networks which adjusts the network temporal granularity given an estimation model capturing both the metric stream runtime evolution and the volatility of the graph topology. Thus, our ultimate goal is to significantly ease processing on graph engines, and costs in general, while respecting at all times user-given accuracy guarantees.
The main contributions of our paper are:

- We introduce Adaptive and Topology-aware Monitoring for dynamic Networks. ATMoN is an open-source framework(*) developed to dynamically adjust the temporal granularity graph metrics are computed based on runtime knowledge captured by a low-cost probabilistic learning model approximating both the metric stream evolution and the runtime volatility of the graph topology. This computationally offloads graph processing engines and eases the communication overhead in edge computing networks where the wealth of data dissemination is plentiful.
- We present a thorough experimentation study to evaluate both the efficiency and accuracy of our solution. All testbeds utilize real-world and publically available datasets from mobile, face-to-face and vehicular networks. Results show that when graph processing engines embrace ATMoN, they can reduce the compute overhead by at least 76%, data volume by 60% and overall cloud costs by at least 54%, while always maintaining accuracy above 88% in comparison to other adaptive frameworks.
- We extend the open-source graph ecosystem of the R programming language R igraph, n.d. by enabling developers to model dynamic networks as adaptive temporal graphs. This extension creates new perspectives by facilitating developers to implement novel algorithms (i.e, ATMoN) that efficiently manage time-evolving graph-structured data streams.

The rest of the paper is structured as follows: Section 2 presents the related work, while Section 3 the problem statement. Sections 4-5 introduce ATMoN, while Section 6 presents the evaluation. Section 7 concludes the paper.

5.2. Related Work

Graph processing frameworks such as Giraph Ching et al., 2015 and Powergraph Gonzalez et al., 2012, enable graph modeling and computation at scale. However, they assume the underlying graph structure is static. On the other hand, Iyer et al. Iyer et al., 2016 present GraphTau, a temporal graph processing framework built on top of Apache Spark, which efficiently unifies data and temporal graph processing. In turn, Han et al. Han et al., 2014 introduce Chronos, a graph engine designed and optimized specifically for running in-memory iterative graph computation on temporal graphs. However, both assume that the network temporal granularity is fixed and predefined. Graph-structured data is on the rise; from social and face-to-face networks to vehicular networks, applications that generate temporal graph structured data are ubiquitous. As IoT con-

(*)https://github.com/dtrihinas/ATMoN

tinues to spread across almost all industries it triggers a massive influx of big data. Undoubtedly, other than temporal graph processing frameworks, we also need algorithms to efficiently provision, manage and monitor IoT services Villari et al., 2016. In light of this, several efforts have been made to process and output complex network metrics. In particular, graph measurement techniques now embrace time as another dimension to graphs and develop “temporal" metrics to describe the network reachability Wu, Huang, et al., 2016, connectivity Mucha et al., 2010 and shortest paths Pan and Saramäki, 2011. However, some graph metrics are, in fact, intractable in practice, and cannot scale at will or be efficiently computed A. Li et al., 2017 Nicholas Loulloudes, 2015.

Another approach successfully demonstrated in edge computing and streaming settings, is to provide approximate metrics values by adapting the monitoring intensity to scale and timely answer queries within certain accuracy guarantees when exact answers are not needed. This is achieved via a runtime estimation model capable of following the metric stream evolution, such that when phases of low variability exist, the metric computation rate is reduced to ease processing while also reducing the data volume. Fan et al. introduce FAST Fan and Xiong, 2014, an adaptive framework for differential privacy which estimates the metric computation rate based on an adjustment given by a PID controller fed with the current estimation error, the time intervals between previously collected metric values and a given inaccuracy budget. In turn, AdaM Trihinas, Pallis, and M. D. Dikaiakos, 2015 is an adaptive monitoring framework for IoT devices, which utilizes a probabilistic moving average as its estimation model and dynamically adjusts the temporal granularity of a metric stream based on the confidence of the algorithmic model to correctly estimate what will happen next in the metric stream. AdaM has been shown to achieve a balance between efficiency and accuracy even for metric streams featuring highly abrupt and transient changes. Nonetheless, while interesting solutions, FAST and AdaM are not suitable for dynamic networks as they limit their scope to adjusting the temporal granularity by only taking into account the metric stream runtime evolution and completely ignore the topology structure.

5.3. Problem Statement

Let $G := (V, E)$ be a graph representing a network, with $|V| = n$ a set of nodes, and $|E| = m$ a set of binary relationships depicting the active interactions among nodes. As the network evolves in time, it is modeled as a graph stream $\mathcal{G}$ indexed by $I \subseteq \mathbb{Z}^+$, with $\mathcal{G}$ denoting a succession of graph instances and formally defined as follows:

$$\mathcal{G} := \{G_i : i \in I\} = \{(V_i, E_i) : i \in I\} = (\mathcal{V}, \mathcal{E}) \tag{5.1}$$

In turn, let $\mu(G_i)$ denote a *graph metric* computed over the i^{th} instance G_i with each obtained measurement, described at the minimum, as a tuple (G_i, t_i, v_i) also comprised by a timestamp t_i

and a value v_i. A measurement may include a set of other attributes, although for brevity, when describing a graph metric we will omit these attributes without loss of generality. Hence, a series of measurements over the graph stream is denoted as a *metric stream* and can be formally defined as follows:

$$\mu(\mathcal{G}) := \{\mu_i : i \in I\} = \{\mu(G_i) : i \in I\} \tag{5.2}$$

Fundamentally, measurements for a temporal graph are obtained periodically over a fixed period of time, denoted as Δ where $\Delta = t_i - t_{i-1}$. With this, Δ is pre-defined and known as the *temporal granularity* of the network so that the i^{th} graph instance is always modeled and processed at known time intervals with the i^{th} measurement obtained at time $t_i = i \cdot \Delta$. Due to its simplicity, this approach is widely adopted by graph processing engines Iyer et al., 2016 R igraph, n.d. In this work, we argue that using a fixed and pre-defined Δ, over large-scale and highly temporal graph streams, features a number of constraints, especially when consecutive measurements do not vary. For example, consider the metric stream depicted in Figure 5.1. If a small Δ is used to process the metric stream, the graph engine is computationally stressed in order to output and disseminate timely measurements to nodes of the monitored network. As the graph grows, more resources (e.g., compute, memory, storage, bandwidth) are required to timely process the graph stream even if there are phases of low variability in the metric stream evolution. Instead, if a large Δ is used, then sudden events or significant insights might remain undetected. In general, because *monitoring depends on the evolution of the data in time, we argue that a fixed periodicity is not effective, as metrics and insights are only useful if collected in meaningful time intervals*.

To accommodate the above challenges, at any given time interval t_i we can dynamically adjust the temporal granularity Δ_i, based on some estimation model, denoted as $\rho(\mu(\mathcal{G}))$, capturing runtime knowledge of the metric stream evolution. Assume μ_i to be the latest computed measurement over $\mathcal{G}$ and that Δ_i accepts discrete integer values in the range $[\Delta_{min}, \Delta_{max}] \subseteq \mathbb{Z}^+$ without loss of generality. Now, suppose the temporal granularity of $\mathcal{G}$ is dynamically adjusted at runtime, so that when the metric stream evolution has not "changed" since last reported, Δ_i should be increased and when the evolution fluctuates, it should be decreased or restored to a minimum value. However, *how much "change" is "tolerable" depends on some evaluation metric (err), upper bounded by the maximum tolerable inaccuracy, denoted as η, given by the user*.

While this argument seems sound, and has been recently explored to adapt the periodicity of monitored nodes at the edge of a network Trihinas, Pallis, and M. D. Dikaiakos, 2015 Fan and Xiong, 2014, in practice this approach is intractable for networks modeled as temporal graphs. This is due to the fact that current adaptive techniques completely ignore the structure of the network and adjust the metric computation rate solely based on the temporal evolution of the metric stream. Hence, although the metric stream evolution may undergo phases of low fluctuation, the network structure can still be extremely volatile. Therefore, an adaptive technique must extend the estimation model $\rho(\cdot)$ to also capture and acknowledge the graph structure volatility. Thus, in a similar fashion, let $\tau(\mathcal{G} \mapsto \mathbb{R})$ denote how the graph stream *structure volatility* evolves based on

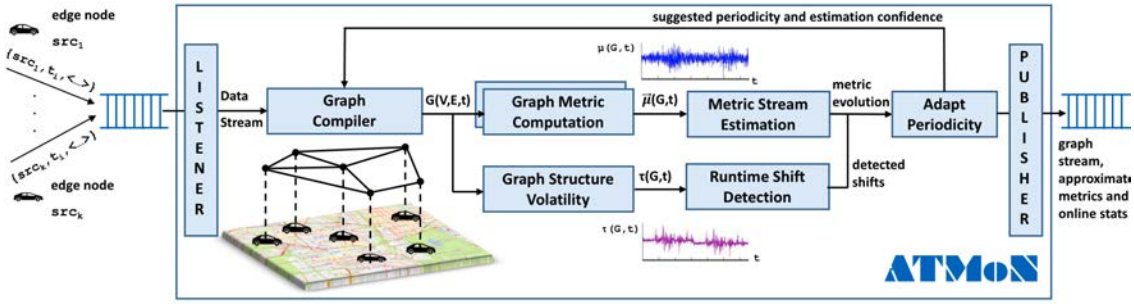


Fig. 5.1: High-Level Abstract Overview of the ATMoN Framework

some metric capable of capturing the dissimilarity of consecutive graph instances Schieber et al., 2017.

$$\tau(\mathcal{G}) := \{\tau_i : i \in I\} = \{\tau(G_i) : i \in I\} \tag{5.3}$$

Our goal is to develop an estimation model $\rho(\mu, \tau)$ capturing at runtime the metric stream evolution and the graph structure volatility, and provide an adaptive function $f(\cdot)$, that outputs the maximum Δ to delay modeling G_{i+1} that will be used to obtain the next set of measurements $\vec{\mu}_{i+1}$. This must be achieved while respecting user-given accuracy guarantees based on some evaluation metric, denoted as $err(\eta)$. Thus, the following equation summarizes the problem:

$$\Delta^* = \arg\max_{\Delta} \{f(\vec{\mu}, \Delta, \rho(\mu, \tau), err(\eta)) \mid \Delta \in [\Delta_{min}, \Delta_{max}]\} \tag{5.4}$$

5.4. The ATMoN Library

To address the above challenges, we have designed and developed the ATMoN framework. ATMoN is a lightweight and open-source library developed to adapt the temporal granularity when computing graph metrics, thus computationally offloading graph processing engines and easing the communication overhead in dynamic networks. The current ATMoN prototype is developed in R and supports the underlying igraph engine, which is developed for C, R and Python. The igraph engine was selected for its open-source nature, allowing the code-base to be extended to provide the ability to adapt the graph temporal granularity, while igraph is also proven to be capable of handling and scaling large networks efficiently R igraph, n.d. Nonetheless, while ATMoN supports the igraph engine, as it features no other external dependencies it can be ported to other popular graph processing frameworks.

Figure 5.1 depicts a high-level and abstract architectural overview of the ATMoN framework. The *Listener* module, continuously digests and queues incoming data updates from the nodes distributed across the monitored network. We consider an example drawn from the domain of vehicular networks. In such a setting, vehicles correspond to nodes and communication links to edges. Digested data is parsed by the *Graph Compiler* module, and depending on the current temporal granularity, the modeled graph stream is updated with the current network graph. Next,

the current graph instance is passed to the *Graph Metric Computation* module so that the metrics of interest are updated. For our vehicular network example this could be the effective diameter, degree distribution and the number of graph components. A set of more 25 metrics are currently available by ATMoN, while users are able to define their own graph metrics by extending the ATMoN metric interface.

After processing the graph instance to obtain the current measurements, the *Metric Stream Estimation* module will update a local reference estimation module capable of capturing the current metric stream evolution, trend and variability. In parallel, the *Graph Structure Volatility* module will compute and update the graph topology volatility by detecting dissimilarities between the consecutive graph instances. The output of this module will be given to the *Runtime Shift Detection* module, which according to the error tolerance provided as input to ATMoN, will render if the network volatility is “tolerable" or not. Having updated the metric stream evolution and the network topology volatility, the *Adaptive Periodicity* module will return a new estimation for the graph stream temporal granularity and a confidence interval for the estimation. The Graph Compiler will then acknowledge the new temporal granularity to ease graph metric computation until this delay expires, thus, allowing the network to “breathe" by reducing resource consumption and the volume of generated data and, subsequently, cloud expenses. Finally, interested users and entities can access through the ATMoN *Publisher* API the approximated graph metric and topology volatility stream.

5.5. Temporal Granularity Adaptation

This section provides an in depth overview of ATMoN with focus on the algorithmic process (Algorithm 4) that updates the temporal granularity of dynamic networks modeled as temporal graphs.

Graph Compiler and Metric Computation. These modules directly map incoming monitoring updates from data sources to graph instances. To support this functionality we have extended the underling graph engine to support adaptive temporal graphs where users are required to provide: (i) the graph model; (ii) the algorithm that will be used to adapt the temporal granularity; (iii) the acceptable range for the temporal granularity; (iv) the maximum tolerable error; and (v) the list of metrics that will be computed by the graph engine. The graph model can be any model supported by the igraph engine, while in the case where igraph is not used, users must provide a direct mapping of how incoming data are modeled to graph instances and provide the respective interface.

```
gstream ← adaptive_graph<graphModel(V,E,...),     algo=ATMoN,
delta=(init,min,max), maxError,     metric_list=list(...)>
```

Metric Stream Estimation. When a new measurement value v_i over the current graph instance

Algorithm 4 Adaptive Graph Temporal Granularity

Input: **init**: user-defined max tolerable error η,
per iteration: current graph G_i from the $\mathcal{G}$ stream
Output: Δ_{i+1} and estimation confidence c_i
Ensure: $\{\Delta_{i+1} \mid \Delta_{i+1} \in \mathbb{Z}^+ \text{ and } \Delta_{i+1} \in [\Delta_{min}, \Delta_{max}]\}$

$\vec{r} \leftarrow \{\}$ //result set
for each m **in** `metric_list` **do**
  $v_i \leftarrow$ `computeMetricUpdate`(G_i, m)
  $\mu_i, \sigma_i \leftarrow$ `updateMetricStreamEvolution`(v_i)
  $c_i \leftarrow$ `updateConfidence`$(\sigma_i, \hat{\sigma}_i)$
  $\vec{r} \leftarrow \vec{r} \cup (\mu_i, c_i)$
end for
$\delta_i \leftarrow$ `computeGraphDissimilarity`(G_i, G_{i-1})
$\tau_i \leftarrow$ `updateTopologyVolatility`(δ_i)
if `changeDetected`(τ_i) **then**
  `change` $\leftarrow$ `true`
else
  `change` $\leftarrow$ `false`
end if
$\Delta_i \leftarrow$ `updatePeriodicity`$(\vec{r}, \eta,$ `change`$)$
return Δ_{i+1}, c_i

G_i is available, this module updates the current metric stream evolution μ_i and variability σ_i, by using a moving average. This will give an initial estimation for the next measurement value, denoted as $\hat{v}_{i+1}$. Moving averages are easy to compute, though many types exist, and can be calculated on the fly with only previous value knowledge. A cumulative moving average for streaming data is the Exponential Weighted Moving Average (EWMA), $\mu_i = \alpha\mu_{i-1} + (1-\alpha)v_i$, where a weighting parameter α, is introduced to decrease exponentially the effect of older values. However, the EWMA features a significant drawback; it is volatile to abrupt transient changes Trihinas, Pallis, and M. D. Dikaiakos, 2015. To overcome this drawback, we adopt a double Probabilistic EWMA (dPEWMA), which dynamically adjusts the weighting based on the probability density of the given observation evolution μ_i and trend x_i. The dPEWMA acknowledges sufficiently abrupt transient changes, adjusting quickly to long-term shifts in the metric evolution and when incorporated in our algorithmic estimation process, it requires no parameterization, scaling to numerous measurements. Equation 5.5 presents the dPEWMA where instead of a fixed weighting factor, we introduce a probabilistically adaptable weighting factor $\tilde{a}_i = \alpha(1 - P_i)$. In this equation, the p-value, is the probability of the current v_i to follow the modeled distribution of the metric stream evolution.

$$\begin{aligned} \mu_i &= \tilde{a}_i(\mu_{i-1} + x_{i-1}) + (1 - \tilde{a}_i)v_i \\ x_i &= \xi\ (\mu_i - \mu_{i-1}) + (1 - \xi)\ x_{i-1} \\ \mu_1 &= v_1,\ x_1 = 0,\ x_2 = v_2 - v_1 \end{aligned} \tag{5.5}$$

The logic behind probabilistic reasoning is that the current v_i depending on its p-value will contribute respectively to the estimation process. Therefore, we update the weighting by $1 - P_i$ so

Name	Description	Metrics	Vertices	Granularity	Duration
mit	A reality mining network obtained from an experimental study conducted at MIT to demonstrate how social structures are developed via online, mobile and direct conversations Eagle and (Sandy) Pentland, 2006	Max Clique	∼1000	5min	9 months
sg09	A face-to-face proximity network obtained from the Dublin Science Gallery in 2009 with data extracted from RFID tags worn by visitors to study human mobility and interactions Isella et al., 2011	Diameter Component Degree Distribution	∼14000	20s	3 months
vanet	A vehicular network from the city of Shanghai exploring wireless communication exchange among vehicles in short-lived proximity-based formed communities Nicholas Loulloudes, 2015	Effective Diameter Giant Comp. Ratio	∼75570	10s	1 day

Table 5.1: Datasets Used to Compare the Frameworks Under Evaluation

that sudden "unexpected" spikes contribute less in the estimation process. This allows the model to refrain from overestimating subsequent v_i's. In turn, if an "unexpected" value turns out to be a shift in the metric stream evolution, as the probability kernel shifts, subsequent "unexpected" values are awarded with greater p-values, allowing them to contribute more to the estimation process. Nonetheless, while adaptive weighting refrains the model from *overestimation at bursty time intervals*, it does not account for monotonic phases of upward and downward trends which often introduce *time lagging effects* in the estimation process. Hence, Equation 5.5 also features a trend component x_i, updated at each time interval, where ξ is a smoothing weight in the range $[0, 1]$ with values near 1 denoting a preference to favor recent trends. Thus, any lagging effects in the estimation process are reduced by boosting the moving average to the appropriate value base with the dPEWMA for $\hat{v}_{i+1}$ now also incorporating an additive trend component.

Assuming, a stochastic and i.i.d distribution as the bare minimum for a metric stream, we adopt a Gaussian kernel $N(\mu, \sigma^2)$, which satisfies the above requirements, with P_i the probability of v_i evaluated under a Gaussian distribution and computed by Eq. 5.6. Nonetheless, while a

Gaussian distribution is assumed, if prior knowledge of the distribution is made available then only the p-value computation must change in the estimation process.

$$\begin{aligned} P_i &= \frac{1}{\sqrt{2\pi}} \exp(-\frac{z_i^2}{2}) \\ Z_i &= \frac{\delta_i - \hat{\delta}_i}{\hat{\sigma}_i} \end{aligned} \tag{5.6}$$

At this point, the metric stream evolution and variability, encapsulated by $\hat{v}_{i+1}$ and $\hat{\sigma}_{i+1}$, can be efficiently updated with only previous value knowledge and without repeatedly scanning the entire stream ($N \to \infty$):

$$\begin{aligned} \tilde{a_i} &\leftarrow \alpha(1 - P_i) \\ \theta_1 &\leftarrow \tilde{a_i} \cdot (\theta_1 + x_i) + (1 - \tilde{a_i}) \cdot v_i \\ \theta_2 &\leftarrow \tilde{a_i} \cdot \theta_2 + (1 - \tilde{a_i}) \cdot v_i^2 \\ \hat{v}_{i+1} &\leftarrow \theta_1 \\ \hat{\sigma}_{i+1} &\leftarrow \sqrt{\theta_2 - \theta_1^2} \end{aligned} \tag{5.7}$$

Having updated the metric stream variability, we proceed to compute the current metric confidence, denoted as c_i. The confidence is a ratio ($c_i \leq 1$) computed from the difference between the estimated and observed standard deviation, and is used as our error evaluation metric. The semantics behind the confidence is that: *the more "confident" the algorithm is, the larger the estimated temporal granularity can be for the reference metric stream*. This supports our framework to "reward" larger adjustments when estimations satisfy the accuracy requested by the user or rollback to a fixed approach when satisfactory estimations cannot be made. Hence, as $\hat{\sigma}_i \to \sigma_i$ the confidence $c_i \to 1$.

$$c_i = 1 - \frac{|\hat{\sigma}_i - \sigma_i|}{\sigma_i} \tag{5.8}$$

Graph Structure Volatility. This module anticipates and models the topology structure volatility for temporal graphs. Hence, after the graph stream is updated, we compute the dissimilarity of the current and previous graph instance, based on a metric capable of quantifying topological differences. However, identifying and quantifying dissimilarities between graphs is a fundamental challenge with several metrics proposed, although most are limited to either extracting only partial information or are computationally demanding Iyer et al., 2016 Ching et al., 2015.

Therefore, to measure the dissimilarity δ_i between two consecutive graph instances, we adopt the D-measure Schieber et al., 2017, denoted as $D(G_i, G_{i-1}) \mapsto \mathbb{R}$ and $D \in [0, 1]$, which associates to each graph instance a set of probability distribution functions, representing node connectivity distances, and compares them, in terms of three graph metrics. The first captures global differences by comparing each graph average node distance distribution based on the Jensen-Shannon distance ($\mathcal{J}$), which is a method for measuring the similarity between probability dis-

tributions Endres and Schindelin, 2003. The second compares the connectivity of each node by looking at the local node dispersion ratio (NDr). The last term analyses the differences in the way this connectivity occurs, through the graph $\alpha-$centrality. We deem the D-measure an appropriate metric as it is capable of comparing graphs efficiently and with high precision, especially for networks with volatile connectivity, which is the case for real-world networks where nodes are not fixed and (dis-)appear in time (e.g., mobile, vehicular networks). We also note that the third term of the D-measure which is the computationally demanding term, can be ignored in scale-free settings, yielding a small imprecision penalty and reducing the complexity of the D-measure to $O(n \log n)$. Thus, the complexity is comparable to other topology metrics (e.g., assortativity, transitivity) but achieves more accurate results due to being able to detect dissimilarities even in the connectivity of graph instances Schieber et al., 2017.

$$D(G_i, G_{i-1}) \cong \sqrt{\frac{\mathcal{J}(dist(G_i), dist(G_{i-1}))}{\log 2}} + |\sqrt{NDr(G_i)} - \sqrt{NDr(G_{i-1})}| \tag{5.9}$$

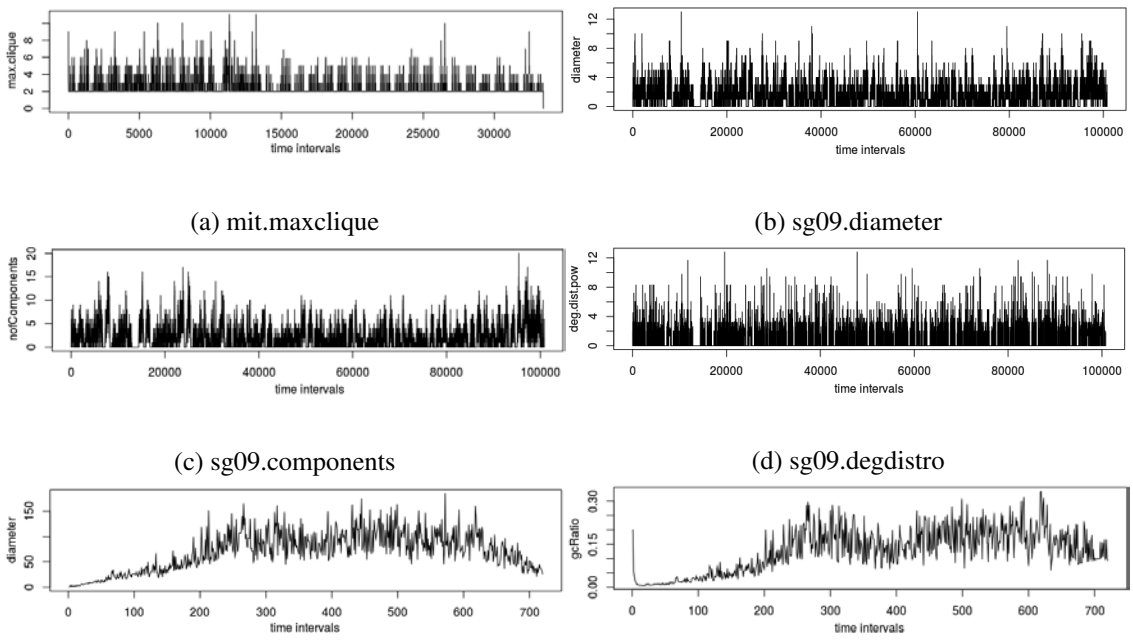
At this point, although a threshold-based approach could be used to determine a runtime change in the graph structure (e.g., $\delta_i > \texttt{thres}$), this is error-prone due to its sensitivity to short-lived spikes. Thus, having computed the dissimilarity between the consecutive graph instances, we then proceed to update the graph topology structure volatility evolution, denoted as τ_i, which is also modeled as a moving average adopting a dPEWMA and apply runtime shift detection to determine if the current topology structure volatility is "tolerable".

Runtime Shift Detection. This is based on the CUSUM test, denoted as C_i, and is a hypothesis test for detecting shifts in the statistical properties of i.i.d timeseries Luo, Z. Li, and Wang, 2009. Specifically, there are two hypothesis θ' and θ'' with probabilities $P(\theta')$ and $P(\theta'')$, where the first corresponds to the statistical distribution of the timeseries prior to a shift and the second to the distribution after a shift ($i > t_s$) with t_s denoting the time interval the shift occurs. The CUSUM is computed with sequential probability testing on the instantaneous log-likelihood ratio given for a metric stream at the i^{th} time interval, as follows:

$$\begin{aligned} c_i &= \ln \frac{P(\theta'')}{P(\theta')} \\ C_{i,\{low,\ high\}} &= C_{i-1,\{low,\ high\}} + c_i \end{aligned} \tag{5.10}$$

where low and $high$ denote the separation of the CUSUM to identify positive and negative shifts respectively.

The typical behavior of the log-likelihood ratio includes a negative drift before a shift and a positive drift after the shift. Thus, the relevant information for detecting a shift in how the volatility of the graph structure evolves, lays in the difference between the value of the log-likelihood



(a) mit.maxclique (b) sg09.diameter

(c) sg09.components (d) sg09.degdistro

(e) vanet.effdiameter (f) vanet.gcRatio

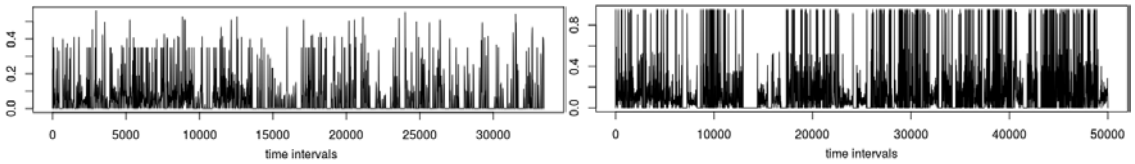
Fig. 5.1: The Metric Streams from the Datasets Used to Compare the Frameworks Under Evaluation

ratio and the current minimum value. A decision function, denoted as L_i, is then used to determine a shift in the graph volatility evolution when its outcome surpasses a threshold ($L_i > h$) determined by the number of standard deviations respecting the user-given maximum tolerable inaccuracy.

$$\begin{aligned} L_{i,\{low,\ high\}} &= \{L_{i-1,\{low,\ high\}} + c_i\}^+ \\ t_s &= \underset{j \leq s \leq i}{\arg\min} \left(C_{s-1}\right) \end{aligned} \tag{5.11}$$

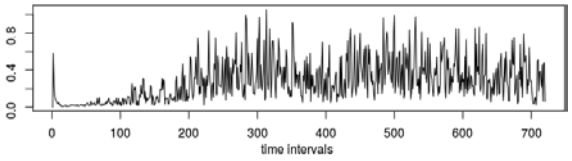
where $L^+ = sup(L, 0)$. Now, let us consider the particular case where the graph structure volatility τ undergoes possible shifts in its evolution modeled by a moving average. Thus, θ' and θ'' can be rewritten as τ' and τ'' respectively, with τ' representing the current evolution, while τ'' the estimated output of the dPEWMA with $\tau'' = \tau' + \epsilon$, and ϵ denoting the estimated magnitude of change in the graph structure volatility. As the structure volatility evolution is used to provide an estimation for $\hat{\delta}_i$, the magnitude of change is equal to $\epsilon = \hat{\delta}_i - \delta_i$. In turn, let $P(\tau')$ and $P(\tau'')$ be modeled and computed from Equation 5.6 when adopting a Gaussian kernel. With some calculations (omitted due to limited space), c_i is rewritten, to perform the decision-making with only previous value knowledge:

$$c_{i,\{low,\ high\}} = \pm \frac{|\epsilon|}{\sigma_\tau^2} \left(\delta_i - \tau' \mp \frac{|\epsilon|}{2}\right) \tag{5.12}$$



(a) mit

(b) sg09



(c) vanet

Fig. 5.2: Dataset Graph Topology Structure Volatility based on the D-measure: $D(G_i, D_{i-1}) \in [0, 1]$

Update Temporal Granularity. Having updated the metric evolution and graph structure volatility we proceed to dynamically adjust the graph stream temporal granularity. To update the temporal granularity we adopt the approach proposed in Trihinas, Pallis, and M. D. Dikaiakos, 2015 and extend it to acknowledge multiple metrics over the graph instance and the runtime volatility of the graph topology. Thus, in Equation 5.13, the estimated temporal granularity Δ_{i+1} is dependent to the current periodicity Δ_i, increasing if variability of the load decreases, and, in turn, decreasing if variability increases. The decision about how large of an adjustment is required, is dependent to the graph structural volatility and the “confidence" of the algorithmic process to (correctly) estimate and follow the runtime evolution of the monitored metric streams. Therefore, when the estimation model is “confident" of its estimations, and the graph volatility does not indicate a change in its evolution, the algorithm will award a larger periodicity.

Intuitively, if $\eta \rightarrow 0$ then the algorithm converges to a fixed periodicity approach (unless an “exact" estimation is made). In turn, if $\eta \rightarrow 1$ an adjustment will take place on each interval even if a confident estimation cannot be made. In turn, if the algorithm process cannot provide a confident estimation within the user-given accuracy guarantees, even for a single metric stream, then the algorithm will rollback to the default periodicity Δ_{min}, in order to preserve accuracy at all times. The complexity of our approach is $O(n \log n)$, with time bounded by the computation of the dissimilarity metric (D-measure). All other calculations (e.g., dPEWMA, CUSUM) feature a constant complexity and are based on previous value knowledge. Moreover, the imprecision η, is the only parameter which is user-defined in the estimation process. Nonetheless, users are free to change: (i) λ which is an optional multiplicity factor (e.g. default $\lambda = 1$) to be used for a more aggressive approach; and (ii) the dPEWMA weights α and ξ, although due to the adaptive weighting process, these may take a wide range of values and can be left to default values for a

small imprecision penalty.

$$\Delta_{i+1} = \begin{cases} \Delta_i + \tau_i \cdot \texttt{min}[\lambda(1 + \frac{c_i^m - \eta}{c_i^m})], & \forall m \mid c_i \geq 1 - \eta \text{ and } change = \texttt{true} \\ \Delta_i + \texttt{min}[\lambda(1 + \frac{c_i^m - \eta}{c_i^m})], & \forall m \mid c_i \geq 1 - \eta \text{ and } change = \texttt{false} \\ \Delta_{min}, & else \end{cases} \tag{5.13}$$

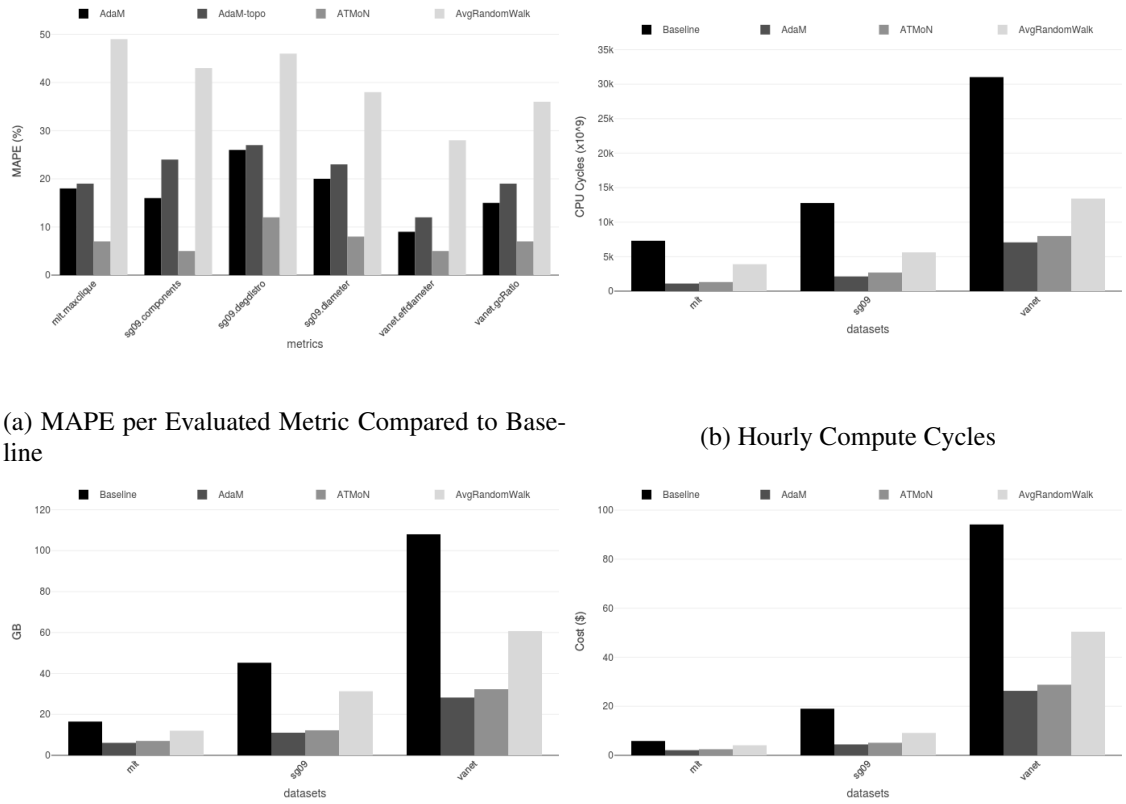
5.6. Evaluation

In this section we present a thorough experimentation study based on real-world datasets to compare the overall accuracy and performance of ATMoN towards:

- **Baseline**, where the temporal granularity of the network is fixed and set to the dataset actual granularity while the graph engine is deployed without ATMoN;
- **AdaM**, a low-cost adaptive monitoring framework which embraces a PEWMA as its estimation model and dynamically adjusts the temporal granularity of the metric stream based on the confidence of the algorithmic model to correctly estimate what will happen next in the metric stream. Hence, AdaM adapts the graph temporal granularity solely based on the predicaments of the metric stream evolution.
- **AdaM-topo**, which is AdaM configured to dynamically adjust the network temporal granularity based on the graph topology structure volatility instead of the actual metric stream evolution. To make this possible we feed the structural volatility as a metric stream to the estimation module of AdaM.
- **AvgRandomWalk**, where a heuristic-based random walk is applied over the graph stream to select the runtime temporal granularity by taking the average of the 90th percentile over a series of 100 runs.

We include both AdaM and AdaM-topo in our evaluation to show that solely adjusting the temporal granularity based on either the metric stream evolution or the graph topology structure is restrictive and will yield high errors, especially for networks with highly volatile connectivity. We conduct each experiment with a tight inaccuracy budget configured to $\eta = 0.1$, unless otherwise stated, meaning that accuracy is expected to be at least 90%. For the frameworks using a moving average, we set the smoothing parameter to $\alpha = 0.45$ and the trend parameter to $\xi = 0.85$ which, after testing, is the best configuration for the AdaM framework.

Table 5.1 presents an overview of the datasets used to evaluate the approaches under-comparison. Instead of opting for trivial or simulated graphs datasets, we introduce graphs that model real-world and highly volatile dynamic networks where the nodes are actual data sources and are remotely distant from the graph engine. These datasets are available online so that our findings can be easily reproduced. Also, all datasets have been widely used in publications referring to dynamic networks with the metrics selected being metrics used in these publications



(a) MAPE per Evaluated Metric Compared to Baseline

(b) Hourly Compute Cycles

(c) Data Volume

(d) Daily Cloud Cost

Fig. 5.1: Accuracy, Overhead and Cost Comparison of the Techniques Under Evaluation

and are of actual interest. Figure 5.1 depicts the graph metrics extracted from each dataset, while Figure 5.2 depicts for each dataset the volatility of the graph topology based on the D-measure.

To emulate the behavior of the monitored sources comprising each network in an edge computing environment, a node emulator was developed. Upon instantiation, each emulator receives a unique ID directly mapping to a node of the graph and, from there on, it emulates the exact behavior of the respected node while also disseminating data updates to ATMoN. We deploy ATMoN and the graph engine in Docker containers on Google AppEngine with the testbed comprised of 8VCPUs and 8GB RAM. AppEngine was selected as a suitable cloud platform due to flexible pricing as compute resources are charged per actual cpu-time. Hence, services benefit in terms of cost when less processing load is applied. A similar scheme is applied for ingress and egress network traffic which is of significant importance in edge computing settings where data sources are distant from the cloud.

5.6.1. Adaptive Technique Estimation Accuracy

In the first set of experiments, we evaluate each of the adaptive techniques towards their estimation accuracy by measuring the mean absolute percentage error (MAPE) towards the dataset

ground truth for each evaluated graph metric. Equation 5.14 depicts how the MAPE is calculated for each evaluated graph metric, where A_i is the actual metric value for the i^{th} datapoint and E_i is the estimated value. For each adaptive technique, when a datapoint is not present, E_i is considered the last reported value.

$$MAPE_n = \frac{1}{n}\sum_{i=1}^{n}|\frac{A_i - E_i}{A_i}| \cdot 100\% \tag{5.14}$$

Figure 5.1a depicts the results of this experiment run. First, we observe that by applying a random walk, it is not feasible to approximate a graph metric stream, especially for metrics featuring highly abrupt and transient phases. Second, we observe that AdaM-topo yields higher errors than ATMoN and AdaM which shows that one cannot use only the graph structure volatility to adapt the temporal granularity of a metric stream. Third, we observe that ATMoN is the only adaptive technique able to satisfy the given accuracy guarantees ($> 90\%$), even for such volatile metric streams. The only exception is `sg09.degdistro`, which is the most complex and unpredictable metric, where the error is slightly above the desired, at 11%. Most importantly, the difference in error of ATMoN from both AdaM and AdaM-topo is significant. Specifically, the difference in error is, at all times, well above 10% for each experiment run. For the `sg09` and `vanet` networks the error difference even exceeds 15%. Thus, in contrast to adaptive techniques which only base the estimation process on the evolution of the metric stream, *ATMoN maintains user-given accuracy guarantees by acknowledging the volatility of the graph topology and adjusts the network temporal granularity based on knowledge of both the metric and graph evolution*.

5.6.2. Adaptive Technique Efficiency and Overhead

In the next set of experiments, we evaluate *efficiency* by measuring the overall overhead imposed to the underlying graph engine which must process the datasets. We measure: (i) *CPU Cycles* consumed to model the graph stream and process the load imposed by each dataset to output the depicted metric streams at runtime; (ii) *Data Volume* generated to store the modeled graph instances and the processed metric streams that must be disseminated at runtime to interested entities and the data sources comprising the network; and (iii) *Cloud Costs* incurred by running the testbed on the cloud based on the Google AppEngine pricing scheme. We note that, when calculating and depicting cloud costs, discounts and compute hours offered by Google are ignored. Also, for figure visualization clarity, in this experiment run we do not depict AdaM-topo which yields significantly higher errors than AdaM and also incurs higher overheads.

Figures 5.1b-5.1d depict the results of the experimentation run. First, we observe that ATMoN is able to significantly reduce both the compute overhead and the overall volume of generated data, when compared to the Baseline. Specifically, *ATMoN is able to reduce the computation overhead by at least 76% and data volume by at least 60%*. These numbers improve even more as the network grows and the graph engine is overwhelmed with data. In particular, for the `sg09`

Trace \ η	**0.01**	**0.05**	**0.1**	**0.15**	**0.2**
mit	5.3	24.6	57.4	69.3	74.2
sg09	12.7	41.3	72.9	80.8	83.1
vanet	9.6	30.3	70.1	74.5	79.8

Table 5.2: Data Volume Reduction (%) in Respect to Max Inaccuracy (η)

Trace \ η	**0.01**	**0.05**	**0.1**	**0.15**	**0.2**
mit	0.01	0.04	8.0	0.13	0.19
sg09	0.01	0.05	9.1	0.15	0.20
vanet	0.01	0.05	9.8	0.14	0.19

Table 5.3: MAPE (%) in Respect to Max Inaccuracy (η)

network, the compute overhead is reduced by 80%. For the `vanet` network, which is comprised by 75000 nodes, the overall data reduction is 71%. This shows that by not embracing adaptivity in an edge computing environment, the overall system is significantly overwhelmed by the volume of data and the required processing which it incurs. In turn, due to constant communication between edge nodes and the cloud service, significant energy is consumed to preserve the wireless link. However, by reducing the compute and network requirements, not only is the network able to achieve greater scalability but the cloud costs are significantly reduced. Specifically, *with ATMoN utilized at the graph engine, an edge computing environment can reduce its daily costs by at least 54% with this number increasing as the network grows larger*.

When comparing ATMoN to AdaM, the former presents a slightly lower compute and data volume footprint which also results in lower cloud costs. The difference in the compute and data volume overhead between ATMoN and AdaM is at most 5% and 7% respectively. This is primarily due to the constant complexity of AdaM, in contrast, to ATMoN where complexity is time-bounded by the dissimilarity metric computation to capture the graph topology volatility. However, for this slight overhead sacrifice, ATMoN yields significantly lower estimation errors. Specifically, ATMoN overall yields 1.5-2x less error compared to AdaM. Hence, by dynamically adjusting the temporal granularity of large-scale dynamic networks, *ATMoN is able to computationally offload graph metric computation, significantly reduce data volume and cloud costs, while maintaining, at all times, given accuracy guarantees*.

Finally, we experiment with the overall data volume reduction in respect to different settings of the maximum tolerable inaccuracy when the graph engine is integrated with ATMoN. Table 2 depicts the results of this analysis where, as expected, relaxing accuracy guarantees provides dynamic and latency-sensitive networks "breathing space" by requiring less compute effort to output graph insights. Nonetheless, the most interesting findings for this experiment are presented in Table 3. In particular, we observe that *at no point does ATMoN violate the accuracy requested*

by the user even for tight error bounds. This is due to the fact that the ATMoN estimation process will not output an adjustment for the graph metric computation rate when a “confident" estimation cannot be made to ensure that the accuracy guarantees given by the user are obeyed at all times.

5.7. Conclusions and Future Work

In this paper, we have presented ATMoN, a novel adaptive framework for monitoring applications that are modeled as temporal graphs. ATMoN provides a flexible architecture to inexpensively and dynamically adapt the temporal granularity graph metrics are computed. This significantly eases graph processing and also reduces data volume and cloud costs. To achieve this, ATMoN uses a low-cost probabilistic learning model that approximates both the metric stream evolution and the volatility of the graph topology structure, while respecting user-given accuracy guarantees. ATMoN is available as open-source and can be served as the vehicle for a number of ongoing research efforts, such as monitoring network dynamics (e.g., fake news epidemiology), community structure and density (e.g., capacity provisioning), abnormal behavior detection (e.g., malicious attacks) and identifying recurring events (e.g.,trends and seasonal patterns).

In the imminent future, ATMoN will be extended to support multivariate graph metric streams. This will allow ATMoN to acknowledge structural changes not only at a graph scale but also in graph communities to dynamically adjust the network temporal granularity at a finer granularity. Consequently, graph engines will be able to allocate processing time only to truly volatile segments of the graph.

References

Ching, Avery et al. (2015). «One trillion edges: Graph processing at facebook-scale». In: *Proceedings of the VLDB Endowment* 8.12, pp. 1804–1815.

Eagle, Nathan and Alex (Sandy) Pentland (2006). «Reality Mining: Sensing Complex Social Systems». In: *Personal Ubiquitous Comput.* 10.4, pp. 255–268. ISSN: 1617-4909.

Efstathiades, Hariton et al. (2016). «Online Social Network Evolution: Revisiting the Twitter Graph.» In: *2016 IEEE International Conference on Big Data*. Washington, DC, USA.

Endres, D. M. and J. E. Schindelin (2003). «A new metric for probability distributions». In: *IEEE Transactions on Information Theory* 49.7, pp. 1858–1860.

Fan, L. and L. Xiong (2014). «An Adaptive Approach to Real-Time Aggregate Monitoring With Differential Privacy». In: *IEEE Transactions on Knowledge and Data Engineering* 26.9, pp. 2094–2106. ISSN: 1041-4347. DOI: 10.1109/TKDE.2013.96.

Gonzalez, Joseph E. et al. (2012). «PowerGraph: Distributed Graph-Parallel Computation on Natural Graphs». In: *Presented as part of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)*. Hollywood, CA, pp. 17–30.

Han, Wentao et al. (2014). «Chronos: A Graph Engine for Temporal Graph Analysis». In: *Proceedings of the Ninth European Conference on Computer Systems*. EuroSys '14. Amsterdam, The Netherlands: ACM, 1:1–1:14. ISBN: 978-1-4503-2704-6.

Holme, Petter and Jari Saramaki (2012). «Temporal networks». In: *Physics Reports* 519.3. Temporal Networks, pp. 97–125.

Isella, Lorenzo et al. (2011). «What's in a crowd? Analysis of face-to-face behavioral networks». In: *Journal of Theoretical Biology* 271.1, pp. 166–180.

Iyer, Anand Padmanabha et al. (2016). «Time-evolving Graph Processing at Scale». In: *Proceedings of the Fourth International Workshop on Graph Data Management Experiences and Systems*. GRADES '16. New York, NY, USA: ACM. ISBN: 978-1-4503-4780-8.

Kostakos, Vassilis (2009). «Temporal graphs». In: *Physica A: Statistical Mechanics and its Applications* 388.6, pp. 1007–1023.

L. Mearian. *Self-driving cars could create 1GB of data a second*. https://www.computerworld.com/article/2484219/.

Leskovec, Jure, Jon Kleinberg, and Christos Faloutsos (2005). «Graphs over time: densification laws, shrinking diameters and possible explanations». In: *ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, pp. 177–187.

Li, A. et al. (2017). «The fundamental advantages of temporal networks». In: *Science* 358.6366, pp. 1042–1046.

Luo, Yunzhao, Zhonghua Li, and Zhaojun Wang (2009). «Adaptive CUSUM control chart with variable sampling intervals». In: *Computational Statistics & Data Analysis* 53.7, pp. 2693–2701. ISSN: 0167-9473.

Masuda, Naoki, Konstantin Klemm, and Víctor M. Eguíluz (2013). «Temporal Networks: Slowing Down Diffusion by Long Lasting Interactions». In: *Phys. Rev. Lett.* 111 (18), p. 188701.

Mucha, Peter J et al. (2010). «Community structure in time-dependent, multiscale, and multiplex networks». In: *science* 328.5980, pp. 876–878.

Nicholas Loulloudes George Pallis, Marios Dikaiakos (2015). «The Dynamics of Vehicular Networks in Large-Scale Urban Environments». In: *1st IEEE International Conference on Collaboration and Internet Computing*. IEEE CIC 2015. Hangzhou, China.

Pan, Raj Kumar and Jari Saramäki (2011). «Path lengths, correlations, and centrality in temporal networks». In: *Physical Review* 84.1, p. 016105.

R igraph. *http://igraph.org/r/*.

Schieber, Tiago A et al. (2017). «Quantification of network structural dissimilarities». In: *Nature communications* 8, p. 13928.

Shi, W. and S. Dustdar (2016). «The Promise of Edge Computing». In: *Computer* 49.5, pp. 78–81. ISSN: 0018-9162.

Traub, Jonas et al. (2017). «Optimized On-demand Data Streaming from Sensor Nodes». In: *Proceedings of the 2017 Symposium on Cloud Computing*. SoCC '17. Santa Clara, California: ACM, pp. 586–597. ISBN: 978-1-4503-5028-0. DOI: 10.1145/3127479.3131621.

Trihinas, D., G. Pallis, and M. D. Dikaiakos (2015). «AdaM: an Adaptive Monitoring Framework for Sampling and Filtering on IoT Devices». In: *IEEE International Conference on Big Data*. Santa Clara, CA, USA, pp. 717–726.

— (2017). «ADMin: Adaptive Monitoring Dissemination for the Internet of Things». In: *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, pp. 1–9.

Trihinas, D., G. Pallis, and M.D. Dikaiakos (2016). «Monitoring Elastically Adaptive Multi-Cloud Services». In: *IEEE Transactions on Cloud Computing*. IEEE 4.

Villari, M. et al. (2016). «Osmotic Computing: A New Paradigm for Edge/Cloud Integration». In: *IEEE Cloud Computing* 3.6, pp. 76–83.

Wu, Huanhuan, James Cheng, et al. (2014). «Path Problems in Temporal Graphs». In: *Proc. VLDB Endow.* 7.9, pp. 721–732. ISSN: 2150-8097.

Wu, Huanhuan, Yuzhen Huang, et al. (2016). «Reachability and time-based path queries in temporal graphs». In: *Data Engineering (ICDE), 2016 IEEE 32nd International Conference on*. IEEE, pp. 145–156.

Chapter 6

Conclusions and Future Work

The main goal in this thesis was carrying out a breadth analysis of Online Social Networks, from differents points of view, different perspectives, and different approaches, aiming to understand better the human being and their behaviour. We have analyzed and classified the sentiment and topic of text content users generate in Twitter. Then, we have developed a tool that can scale to clusterize outliers in large datasets and we test it for Online Social Networks. And, at the end, we created a framework that can dynamically adapt the temporality of large-scale dynamic networks, optimizing computational resources.

In this last chapter, we collect the main conclusions of every paper and we list the next steps to be followed, are been followed or relevant breakthroughs in the literature since the papers were published.

6.1. Paper 1: Sentiment Analysis and Topic Detection of Spanish Tweets: A Comparative Study of NLP Techniques

In this paper we have presented a comprehensive set of experiments classifying Spanish tweets according to sentiment and topic. In these experiments we have evaluated the use of stemmers and lemmatizers, n-grams, word types, negations, valence shifters, link processing, search engines, special Twitter semantics (hashtags), and different classification methods. This collection of techniques represent a thorough study.

The first conclusion of our study is that none of the techniques explored is the silver bullet for Spanish tweet classification. None made a clear difference when introduced in the algorithm. The second conclusion is that tweets are very hard to deal with, mostly due to their brevity and lack of context. The results of our experiments are encouraging though, since they show that it is possible to use classical methods for analyzing Spanish texts. The largest accuracy obtained (58% for topics and 42% for sentiment) are not too far from other values reported in the TASS workshop. However, these values reflect that there is still a lot of room for improvement, justifying further efforts. These results provide a compelling argument to explore even richer linguistic analysis

and compare those results with larger Spanish text.

The study exposed in this chapter was performed during 2012-2013 years. Natural Language Processing has been extensively studied in the last decade. It has evolved with the outbreak of Deep Learning and the use of Neural Newtorks (e.g., Lopez and Kalita, 2017; Alshemali and Kalita, 2020) and new ways of word representation such as *word2vec* (Mikolov et al., 2013).

6.2. Paper 2: Graph-based Techniques for Topic Classification of Tweets in Spanish

In this second paper, as a continuation of the first, we have focused on Topic Detection and we have developed our own method for topic classification, building graphs from the input texts. Although aggregate performance achieved is higher than 70%, we observe our dataset is very unbalanced (more than 80% of the texts are either *politics* or *other* category). We think that an additional experiment with more accurate training could reveal if this behaviour is due to an unbalanced training or to the actual design of the system. Since the number of training texts in some categories (for instance, *literature*) is rather scarce, we think that a far more complete training set than that currently available would be needed. Additionally, the promising results invite us to apply this work for sentiment analysis, as done in the previous chapter.

The study presented in this chapter was performed in 2015 and we believe it is still a novel and interesting approach for NLP. Deep Learning has powered Natural Language Processing to the next level in the last years. However, graph-based techniques remains an interesting approach and it has been recently studied in tandem with Neural Networks (Deep Learning) as in Schlichtkrull, De Cao, and Titov, 2020.

6.3. Paper 3: Unsupervised Scalable Statistical Method for Identifying Influential Users in Online Social Networks

In this paper, we have introduced MUOD, a novel unsupervised outlier detection algorithm based on FDA. MUOD outperforms other FDA-based outlier detection algorithms while offering a high scalability that allows to apply it in large scale multivariable datasets.

We have tested the practical utility of MUOD in a specific problem, the detection of influencers in OSNs. The application of MUOD in a large-scale Google+ dataset including detailed information for more than 400M users and billions of activities (posts, likes/plusones, reshares, etc) reveals that the different outlier classes identified by MUOD include users that respond to different definitions of influence previously used in the literature. Hence, the results show strong evidences of the utility of MUOD as an algorithm support the unsupervised identification of influencers when a pre-defined type of influential user does not exist.

MUOD algorithm can be applied to a myriad of problems, in which the nodes/users/entities

can be defined by a set of properties mapped into a signal. As future work we will explore the utilization of MUOD to address the following issues:

- Fake News detection in Social Media. News can be characterized by a large set of properties including (source of the new, timestamp, geographical origin, topic, number of nodes forwarding the news in the OSN, etc). Our hypothesis is that (at least) some types of Fake News will present specific properties that make them different from the rest. If such hypothesis is correct, MUOD should identify them as a certain types of outliers.

- Fraud Detection in Online Advertising. Online Advertising is a multibillion dollar business that represents the main source of revenue for Internet. One of the main problems that online advertising faces is fraud, e.g., when a robot instead of a human watches an ad. We can create a signal associated to different websites in order to identify whether the ads presented in that website are actually viewed by human beings. This signal would include input data such IP address visiting the website, browsers visiting the website, number of ads served per hour, average time spent by users in the website, average mouse movement pattern, etc. Our hypothesis is that some websites committing ad fraud will present some specific properties that make them different from the mass. If this hypothesis hold, MUOD is a good candidate algorithm to be applied in this context.

Beyond online systems, there are several real scenarios where the number of variables and the number of observations are high in which outlier detection is very important. For example, medical imaging datasets often contain deviant observations due to acquisition or pre-processing artifacts or resulting from large intrinsic inter-subject variability. Specifically in Neuro-imaging, various kinds of acquisition artifacts may be present in fMRI (functional Magnetic Resonance Images) data. Even small movements of the head may produce large artifacts in the signals, and also heartbeat and breathing both induce pulsatile motion in the brain, which creates physiological noise artifacts directly in the data Lazar, 2008; Lindquist, 2008; Monti, 2011; Poline and Brett, 2012. Complexity and massive amount of this kind of data, and the presence of different types of noises, makes the fMRI data analysis a challenging one; that demands robust and computationally efficient statistical analysis methods as MUOD.

High-dimensional data are increasingly encountered in other applications of statistics, e.g. in biological and financial studies Chen et al., 2009; Zeng et al., 2015. Also, in geochemical data, because of their complex nature Reimann and Filzmoser, 2000, regional geochemical datasets practically always contain outliers. In fact, finding data outliers that may be indicative of mineralization (in exploration geochemistry) or contamination (in environmental geochemistry) is one of the major aims of geochemical surveys Templ, Filzmoser, and Reimann, 2008. Since it is an interesting topic with multiple applications, this work is being continued as we can find in Ojo, Lillo, and Anta, 2021.

6.4. Paper 4: ATMoN: Adapting the "Temporality" in Large-Scale Dynamic Networks

Last but not least, in this last paper we expose our framework for monitoring graph streams in an efficient way, saving computational resources with a cost of an affordable loss of accuracy. Our novel technique remains on monitoring an additional graph metric which acts as a sensor of changes on the dynamic network/graph. Besides, these metrics we propose having a linear (assortativity) or quasi-linear cost (transitivity). Such cost implies an additional effort that yields computational savings. This contribution has been included in a state-of-the-art adaptive monitoring tool for data streams, namely AdaM, outperforming its standalone performance.

On the future, it would be desirable to test our framework on the long run and/or with larger graph streams, measuring the resources optimization (computational savings) as the size of the dynamic networks grows.

References

Alshemali, Basemah and Jugal Kalita (2020). «Improving the Reliability of Deep Neural Networks in NLP: A Review». In: *Knowledge-Based Systems* 191, p. 105210. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2019.105210. URL: https://www.sciencedirect.com/science/article/pii/S0950705119305428.

Chen, Y. et al. (2009). «Outlier detection with the kernelized spatial depth function». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31. DOI: 10.1109/TPAMI.2008.72. URL: https://doi.org/10.1109/TPAMI.2008.72.

Lazar, Nicole (2008). *The statistical analysis of functional MRI data*. Springer Science & Business Media.

Lindquist, Martin A (2008). «The statistical analysis of fMRI data». In: *Statistical Science*, pp. 439–464.

Lopez, Marc Moreno and Jugal Kalita (2017). «Deep Learning applied to NLP». In: *arXiv preprint arXiv:1703.03091*.

Mikolov, Tomas et al. (2013). «Distributed representations of words and phrases and their compositionality». In: *Advances in neural information processing systems*, pp. 3111–3119.

Monti, Martin M (2011). «Statistical analysis of fMRI time-series: a critical review of the GLM approach». In: *Frontiers in human neuroscience* 5.

Ojo, Oluwasegun, Rosa E Lillo, and Antonio Fernández Anta (2021). «Outlier Detection for Functional Data with R Package fdaoutlier». In: *arXiv preprint arXiv:2105.05213*.

Poline, Jean-Baptiste and Matthew Brett (2012). «The general linear model and fMRI: does love last forever?» In: *Neuroimage* 62.2, pp. 871–880.

Reimann, Clemens and Peter Filzmoser (2000). «Normal and lognormal data distribution in geochemistry: death of a myth. Consequences for the statistical treatment of geochemical and environmental data». In: *Environmental geology* 39.9, pp. 1001–1014.

Schlichtkrull, Michael Sejr, Nicola De Cao, and Ivan Titov (2020). «Interpreting graph neural networks for nlp with differentiable edge masking». In: *arXiv preprint arXiv:2010.00577*.

Templ, Matthias, Peter Filzmoser, and Clemens Reimann (2008). «Cluster analysis applied to regional geochemical data: problems and possibilities». In: *Applied Geochemistry* 23.8, pp. 2198–2213.

Zeng, Yong et al. (2015). «Aberrant gene expression in humans». In: *PLoS genetics* 11.1, e1004942.