



Master Science de la matière
École Normale Supérieure de Lyon
Université Claude Bernard Lyon I

Internship 2020–2021
Alessandro Chiappori
M2 Complex systems

Stability and Fidelity of Time-window Aggregated Temporal Networks

Abstract

The analysis of many complex systems leverages recent developments in network science. Static networks are already well characterized, at least from a theoretical point of view, while networks evolving in time — temporal networks — still lack a widely-accepted formalization. With our investigation we contribute to bridge this gap, by working on the time-window — or "snapshot" — representation of temporal networks. Our contribution is twofold. Firstly, we define two scores, named stability and fidelity, that can be measured to evaluate quantitatively a given time-window aggregated network and compare different aggregation techniques between them. Secondly, we develop methods to do a preliminary filtering of temporal networks, with the idea to prepare further analysis and visualization. We thus provide a general framework to both simplify large temporal networks, and evaluate the resulting object. Finally, we perform experiments on representative datasets in the literature and then discuss the potential and limits of our framework.

Supervisor

Rémy Cazabet
remy.cazabet@gmail.com
Data Mining and Machine Learning (DM2L) team, in the LIRIS lab (CNRS, INSA, UCBL)
20, Avenue Albert Einstein, 69621 Villeurbanne Cedex
https://liris.cnrs.fr/

July 15, 2021

Acknowledgments

First of all, I would like to express my gratitude to Rémy Cazabet, who supervised my six-month internship. I owe my introduction to the fascinating field of complex networks to him, starting from the class that he taught to me and my colleagues during the M2. Either in presence or in remote, exchanges with him have been fundamental throughout the entire period of my internship, to guide the orientation and content of this report. I also thank him a lot for encouraging me to explore and suggest new research directions by myself: this led me to a fuller and more valuable internship experience.

I sincerely thank all the professors, students and administrative staff of the École Normale Supérieure de Lyon, for the wonderful environment that I have found since the day of my arrival, back in August 2019. In particular, I am extremely grateful for the opportunity that has been offered me through the Ampère scholarship of excellence, which allowed me to join the ENSL and live the past two years in the wonderful city of Lyon. During this period, I exceeded all my expectation from both an academic and personal point of view, despite the COVID-19 pandemic.

Many special mentions are now deserved by all the friends with whom I shared the period of my master, from the newer that I met in Lyon (Teresa, Dora, Eugenio, Serena, Fabio, Andrea, Konrad, Ozan, Luca, Matteo, Camilo, Orestis, Ludovico, Tommaso and many more) to the older buddies from my bachelor in Trieste (Fabio, Jacopo, Marco, Martina, Francesco, all my other fellows from Collegio Fonda, Luca, Yousef, the members of my Fantacalcio league and many others).

Finally, my greatest thank you of all goes to my mother, because my academic career would have never even started without her constant efforts to promote my curiosity, my perseverance and my willingness to put myself out there. She has always been and she still is my first inspiration in life and my accomplishments will never be great enough to pay her back.

Contents

Chapter 1

Introduction

From the beginning of this century, advancements in information technology and computational engines made it possible to increase exponentially the amount of collected data. Data gathering and the analysis of the resulting datasets are crucial for the decision-making, strategy development and activity planning of today innovative companies as well as public institutions. The possibility to efficiently exploit the information that comes from collected datasets is driven by and also drives in turn, new developments in fundamental and applied research. Depending on the objective and background of the researcher, the focus can change from systems modeling (mathematics, physics, complex systems) to the development and optimization of data mining algorithms (computer science, data science, software engineering).

The number and size of existing datasets is growing so fast that the development of computational tools to analyze them can hardly follow. With the limitation imposed by the computational power, time and costs available, it is necessary to have models to simplify the datasets in order to analyze them. Sometimes, known characteristics of the underlying system can be exploited to do so, but it is often more interesting — or even necessary — to model and analyze a dataset without any a priori knowledge.

One way to efficiently model and analyze most of nowadays datasets is with the network formalism. It comes from graph theory, a branch of mathematics that has been adapted to the new needs of data analysis, during the last two decades. The key idea is to introduce a topological structure by connecting the basic units of the system with links, if and only if they are related according to a certain definition of mutual interaction. As a result, we go from a data table to a web of mutual interactions, where much of the detail about each single link and unit is often neglected. This aspect is extremely important from a system modeling point of view: the essence of the system lays on the structure of its relations itself, not much on the specific properties of every single actor.

The next key question from a complex systems science viewpoint is whether the network structure is constant in time, so that systems dynamics occurs *on* it, or whether its web of relations have a temporal evolution. In this second case, it is necessary to add a time component to the network representation. Otherwise, a single *static* network will only capture interactions at a given instant. On the contrary, *temporal* networks integrate the time component in a non-trivial manner and are thus more suitable for systems whose structure of interactions changes consistently in time.

There exist many possible ways to represent and handle temporal networks, depending on the specific dataset, needs and even personal taste. The more intuitive way to picture a temporal network is perhaps as a sequence of static networks obtained by grouping close interactions in time under time windows. This representation is usually referred in literature as a "snapshot graph". In this way, temporal networks are seen as motion pictures, where each single frame is made by a static network. The process to obtain a snapshot graph from the original dataset is non-unique: time windows size has to be chosen and adjacent windows can either be disjoint or partially superpose, for instance. The goal of our investigation is to develop a framework to evaluate and quantify the effect of this type of choices, with the ultimate purpose of improving the representation and analysis of temporal networks.

Chapter 2

Preliminaries

2.1 Background

2.1.1 Static networks

Many complex systems and datasets can be represented as networks. The basic components — that we call *nodes* (or *vertices*) — are connected pairwise according to a certain property of the system — through the so-called *links* (or *edges*). The introduction of this formalism into the analysis of complex systems has led to the development of the entire new field of network science, at the beginning of this century [1, 2, 3]. The term *complex networks* is often used to refer to complex systems that are efficiently studied with the network formalism.

Today, typical approaches, measures — called *centralities* — and methodologies conceived for the network formalism are successfully employed for systems modeling in applied research fields, e.g., human communication, brain science, ecology and transportation. Pioneering papers were [4], where the small-world property was first formalized; [5], introducing the Barabási-Albert generative model for scale-free networks; [6], starting the characterization of community structure in real networks.

One of the biggest strengths of modeling datsets as static networks is in their intuitive and powerful visualization (Figure 2.1). A simple look at a static network is often self-explanatory and quite easily readable even for non-experts, because key aspects of the structure of the underlying dataset can be directly appreciated by eye. And in fact, properties as the node degree — the number of links including that node — and the community structure come mainly from the study of social networks, which means that they are based on dynamics that we know from our every-day life.

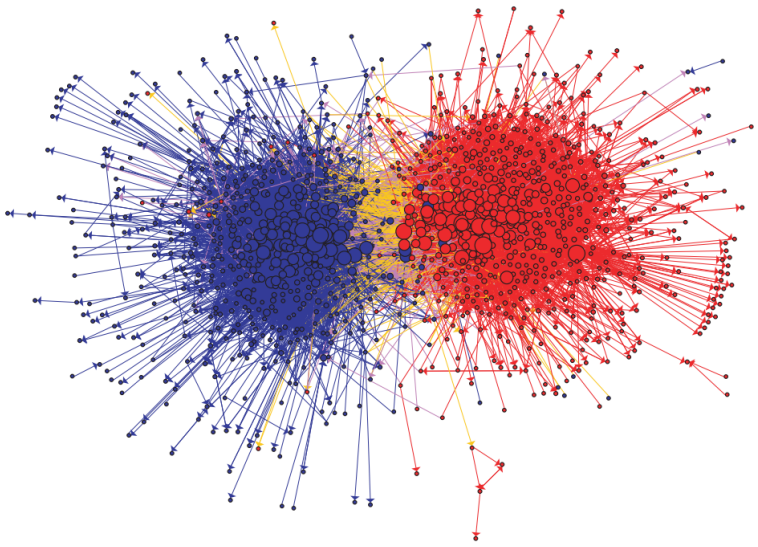

Figure 2.1: Community structure of political blogs in the U.S.A. before 2004 Presidential Election. Nodes are blogs with a political orientation (red for conservatives, blue for liberals), connected on the basis of mutual citations. Adapted from Adamic and Glance, 2005 [7].

2.1.2 Temporal networks

In this report, we are interested in networks that change their structure over time, that is, networks with a time component. In this case, links between nodes also have a time label, that we call a *timestamp*. From now on, we will refer to single links with a timestamp as *contacts* [8] and we will reserve the term *links* itself for node pairs having at least one contact; we will use the term *intervals* to refer to contacts with duration, i.e., lasting for a longer time interval than a single time step dt. We can consider temporal networks to be made up of contacts or of intervals. Temporal networks based on the second approach are usually labeled as "interval graphs" (Figure 2.2). It is still an open question to determine which type of representation is more suitable for a given dataset. The answer could still have to be found or the question above could even be ill-posed, so that choosing the representation should be simply a matter of computational ease or personal taste.

More generally, temporal networks are far from being a simple generalization of static networks; instead, they make an entire line of research on their own. A widely-accepted comprehensive view is still missing [9], starting from the name of the field: temporal networks, dynamic networks, time-varying networks, temporal graphs, evolving graphs etc. Not only the direct transposition of tools that are used for the data mining of static networks often turns out to be ineffective, but even the data representation and visualization itself of temporal networks is non-trivial.

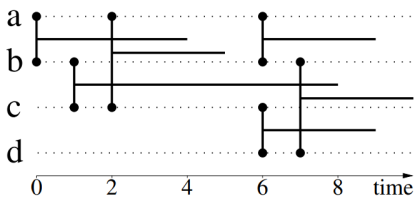

Figure 2.2: Toy example of an interval graph. It is displayed as a link stream, one of the most popular ways to visualize temporal networks made by intervals. Picture adapted from Latapy et al., 2018 [10].

2.1.3 Representation of static and temporal networks

We use the term "representation" as in the reviews of Holme and Saramäki [8, 9]: it has the general meaning of one possible way to look at the network. Usually, a given representation directly is, or matches with, a data structure. Sometimes it also corresponds to a graphical depiction, so that representation and visualization coincide. But this is not the general case, so that network representations and network visualization are two distinct tasks.

For static networks three possibilities are usually considered and they directly correspond to data structures: the edge list — a list of all the edges in the network in the form (i, j) where i and j are nodes; the adjacency list — for each node, a list is stored of the other nodes that are connected to it through links; the adjacency matrix — a binary square matrix with ones at the entries corresponding to links: nodes i and j are connected if and only if the entry ij equals 1 (they are not connected if the entry is 0).

On the contrary, the representation of time-varying networks is more tricky, as many choices are possible to account for the time dependence, including the *contact sequence* — a list of triads on the form (node 1, node 2, timestamp) — the *link stream* (Figure 2.2) and the *snapshot graph* — a temporal sequence of static networks. The latter is the type of representation that is obtained by performing time-window aggregation, as we detail in Section 2.2. It is the most human-friendly way to picture the evolution in time of a complex network, since the handiness of static networks is recovered snapshot by snapshot. With respect to more "conservative" representations as the contact sequence and the link stream, which are closer to the original dataset structure, snapshot graphs have the drawback that the exact time ordering of any two links inside of the same snapshot is lost, as well as the number of times that a given link occurred (this type of information is partially conserved through link weights).

In our investigation we focused on the representation of temporal networks, not on their visualization. Nevertheless, the representation via time-window aggregation can be easily translated to a graphical visualization, as a time sequence of static networks. Visualization algorithms and software

products born for static networks (as ForceAtlas2 [11] and Gephi [12], respectively) can be exploited snapshot-by-snapshot to get a picture of the temporal networks' evolution. More generally, the wide choice of representations brings to many possibilities for visualization: the time coordinate can be included as a spatial axis (as in link streams, Figure 2.2) or by making a movie.

2.2 Definition of the time-window aggregation

An alternative to time-window aggregation of temporal networks is given by many existing methods of network "summarization" — a general term used in data mining to refer to the simplification and compression of various data structures. A review of the existing methods for the summarization of temporal networks, up to 2018, is given in the 4th chapter of [13]. The goal of those summarization methods is to facilitate the analysis of temporal networks; in this view, it is the same goal of time-window aggregation. But we have to keep in mind the following important difference: in the first case, the output is given by a time series of the most important patterns, actors or processes [14, 15, 16, 17, 18, 19], but a global view of the underlying network is lost. On the opposite, the output of a time-window aggregation is given by a sequence of snapshots. Hence, despite being further from an explicit analysis of the dynamic processes going on, time-window aggregation usually allows to keep a richer representation of the system, without breaking the network representation to pieces. It should thus be clear that summarization methods are not the right path to take if the goal is to find a "networky" representation of the evolution of complex systems.

The general idea of time-window aggregation is to convert a contact sequence into a sequence of static networks: the *snapshots*. First, the whole time period has to be segmented into a certain number of time intervals — that we call *windows*; adjacent windows can also partially overlap. We refer to this segmentation process as the *windowing*. For each window, all the links with at least one contact included in the corresponding time interval are "projected" onto a single snapshot. For each link, we store its label (the two nodes involved) and a link weight (Section 3.2.1) and we consider as if the link was an interval covering the entire time window. That is why we call it an "aggregation" method, because links that are close in time end up belonging to a same static network and the exact time ordering is left out.

The first problem that aggregation is expected to solve concerns *degenerate networks*, i.e., temporal networks that are ill-defined at the time resolution of the time step. A classic example is given by email datasets [20], where emails define links between sender and receiver(s): the recording time step is smaller than the average waiting time between consecutive events (the sending of an email), so that most of the timestamps will correspond to very few or even no events at all. Therefore, a time-window aggregation with window sizes close to the finest granularity would be meaningless, resulting in empty snapshots or with very few and usually disconnected links. Aggregating with a window size larger than the measurement resolution, we can go from degenerate networks to aggregated networks that are meaningful from a network science viewpoint, snapshot by snapshot.

In the face-to-face datasets that we considered, from SocioPatterns project, the original time step already comes from the aggregation of measurements taken with a finer granularity (more details are in Section 6.1.1 of the Annex). But even if a time-window aggregation is already performed by the authors to avoid degeneracy, snapshots with a small window size still have very few nodes and are much disconnected. That is why it can be useful to perform a second windowing prior to the analysis of the datasets.

Usually, three types of windowing are found in the literature (Figure 2.3): (i) non-overlapping windows, where the whole period is subdivided into side-by-side windows that do not superpose; (ii) overlapping — or sliding — windows, whose time intervals are partially superposed; (iii) growing windows, where the starting timestamp is in common among all windows and the size is progressively increased, so that each snapshot incorporates all the previous ones. An application of all the three possibilities to the analysis of community evolution can be found in [21]. In our investigation, we only considered windows with fixed size w, either non-overlapping or sliding by one time step at a time.

Known methodologies for the analysis of static networks can be applied to the single snapshots that result from the time-window aggregation. Hence, each snapshot should be "good" from a network science point of view. That is, they should lay between the two following limit cases: i) a too sparse

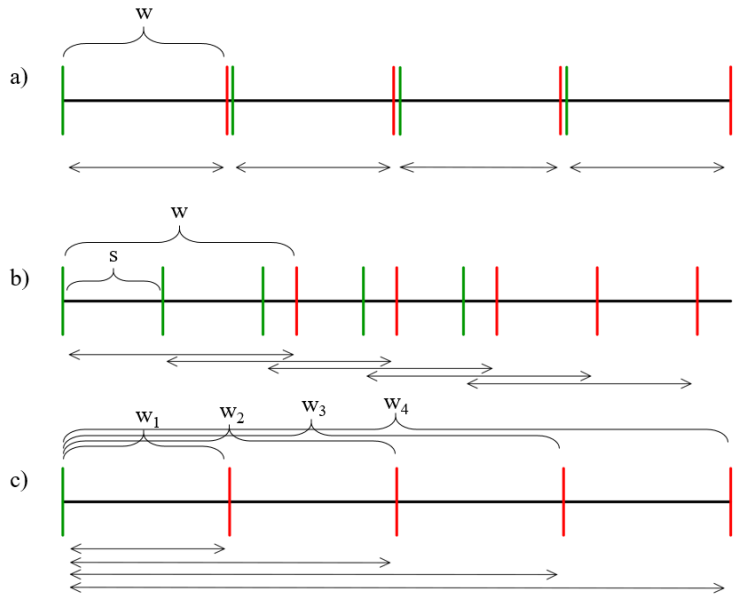

Figure 2.3: Schema of the three main types of windowing. Time is on the horizontal axis; vertical lines mark the beginning (in green) and the end (in red) of the windows. The corresponding time interval for every window is indicated by the arrows below each picture. a) Non-overlapping with window size w, fixed or varying snapshot by snapshot. b) Overlapping with fixed window size w and fixed shift s between two consecutive windows. When not specified, s is taken equal to the measurement resolution, i.e., the smallest possible time step dt. Note that non-overlapping windowing can be considered as the limit case where $s = w$. c) Growing windows with fixed starting time and increasing window size $w_1, w_2, w_3, \ldots$

graph, with too many isolated nodes, ii) a too dense graph, close to a clique, where almost any node is connected to any other. As a consequence, the window size should not be too short, nor too large, respectively (Figure 2.4).

Some articles published in the early '10s addressed the problem of formalizing the aggregation process, to assess the reliability of a subsequent data analysis. In particular, [23, 24] investigated and quantified the influence of the choice of the window size, in the case of non-overlapping windows. For networks with periodic activity, a natural choice is to perform a non-overlapping windowing with fixed window size corresponding to the intrinsic time scale of the dynamics. For instance, communication networks usually follow a circadian cycle, so that aggregation by windows of 24 hours is often an appropriate choice. But in other cases, the characteristic time scale is unknown, or it doesn't even exist [8]. In all these cases, the choice of the window size necessarily biases the analysis. That is why researchers began to propose methods to discover automatically the best parameters for time-window aggregation, focusing mainly on reducing biases and limiting the number of free parameters of their algorithms.

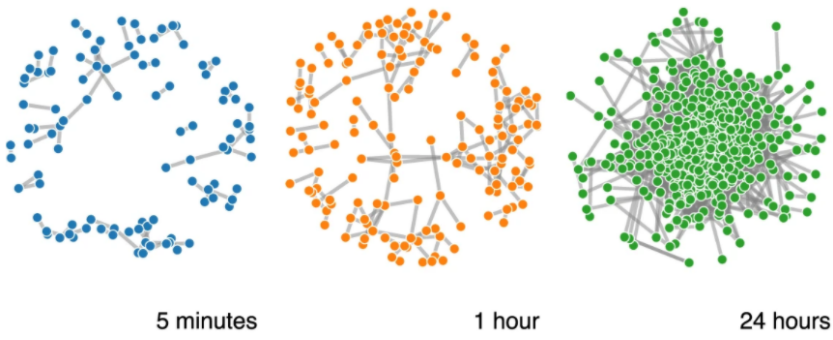

Figure 2.4: Snapshots with different increasing window sizes, obtained by non-overlapping windowing of a Bluetooth proximity network. As we can easily appreciate, at 5 minutes we get a too disconnected network, with many small connected components, while at 24 hours the network has a very high density, close to a full clique. In both cases, the visualization is rather useless. Picture from Sapiezynski et al., 2019 [22].

2.3 Overview of windowing methods

A common goal of windowing methods available in the literature is to automatically find a characteristic time scale of the system, that is, of the evolution of the underlying temporal network structure. But as many analysis suggest, the proper window size depends on the property that we wish to analyze [23, 25, 26, 27, 28]. Hence, it is often inappropriate to look for *the* characteristic time scale, but any possible choice to optimize the aggregation can bring to *a* characteristic time scale of the network. We agree with this view and it was not our main purpose to find a scale with a new method from us. Nevertheless, we are able to suggest for which values of the window size the derived representation is unstable or too distant with respect to the original temporal network. Given a time-window aggregated network, our framework allows to determine how much it is stable and faithful to the original data set.

Windowing algorithms in the literature usually consider non-overlapping windows and they can be split into 2 main groups. In the first one, all the windows have the same size w [25, 27, 28, 29]. So, many values of w are tested and the best one is selected according to a certain criterion. We refer to this group as the "fixed" windowing approach. In the second one, the "dynamic" windowing approach, windows have time-varying sizes [26, 30, 31, 32]. Potentially, a varying w is a valuable generalization, allowing to sample each period with the most appropriate frequency (higher frequency when the network evolves more rapidly). Dynamic windowing becomes particularly interesting for networks with non-periodic evolution, where any characteristic time scale cannot be valid for the whole time period. On the contrary, for temporal networks with an intrinsic periodicity, a fixed windowing is expected to be the more suitable approach. The optimized window size would indeed reflect *a* characteristic time scale of network's evolution.

In this report, we focus on the fixed windowing approach. We take [25] as the representative of this class: it is one of the oldest and it is much cited by successive studies. In this article, to choose the best value w the authors balance noisiness (too small windows), with the loss of information that comes from aggregating data (too large windows). We appreciate this idea of having a balance between two extremes, but we disagree on some aspects of the implementation. First of all, we have to choose a certain network measure F a priori and then track its evolution snapshot-by-snapshot. This passage is quite common across the different methods: since we are often interested in applying ready-made tools for the analysis and visualization of static networks to the single snapshots, many address the issue by directly using static network measures to guide the aggregation [23, 26, 27].

But with this approach, in different ways, available windowing methods go beyond our goal of a general preliminary treatment (the filtering, Section 3.2): they end up already starting or directly performing a data analysis, skipping the more fundamental question of quantifying the stability of the aggregated network that is returned. As a result, the objectiveness and generality that we wish to maintain are compromised by the introduction of biases. In our view, network measures should be considered only *after* the aggregation process. Indeed, different network measures bring to different window sizes and there is no choice that ensures to always obtain a representation that is stable in itself, i.e. with smooth transitions from snapshot to snapshot.

To quantify the information loss, authors of [25] compute a compression ratio: they take the time series $\{F(G_1), F(G_2), \dots, F(G_N)\}$, where $G_1, G_2, \dots, G_N$ are the snapshots, and they apply a compression algorithm on it; then, they return the ratio between the bit length of the uncompressed series and the bit length of the compressed version of it. We believe that, as other methods based on compression algorithms [31, 32], it as the disadvantage of being a sort of black box, lacking intuitiveness and interpretability. We believe that this same downside is shared by ML-driven approaches, as [28].

To conclude this section, we briefly comment [30], a representative of the class of dynamic windowing methods. There, windowing aggregation is driven by a maximization in similarity (Equation 3.1) between every two subsequent snapshots. The algorithm works in an online fashion: at a given time during the execution there is a growing window with the beginning anchored to the end of the previous window: at each step the end of the growing window is moved forward and similarity of the corresponding snapshot to the immediately previous one is computed. The growing window is cut, to begin with the next one, at the size for which similarity with the previous window is maximum.

In the paper, authors present their algorithm and code as almost automatic and parameter-free. We carried out some experiments by running their code on "our" datasets (details on the datasets are in Section 6.1.1) and we discovered that there are actually many parameters left to the user, who can

tune and greatly modify the resulting aggregation.

We bring this example here to point out that most methods that are presented as objective and fully automatic, actually require customization from the final user. This is not necessarily a downside, since in this way methods can adapt to different needs, but it means that: (i) a perfect and autonomous method doesn't exist and (ii) evaluation frameworks as the one we propose are extremely useful to examine time-window aggregated networks produced through different algorithms or sets of parameters. Finally, we remark that often the codes are not directly available to the reader and reproducing exactly authors results by coding from scratch can be very difficult.

2.4 Evaluation of time-window aggregated networks

Every paper presenting aggregation methods includes a certain self-evaluation. Often, network datasets with well-known documented dynamics are considered, as the Enron emails corpus [20]. In this way, authors can test whether the snapshot graphs produced through their method are able to replicate or even outperforms previous ones. On the contrary, very few studies define general evaluation frameworks to compare different windowing methods between them. Moreover, we think that the more fundamental questions about which properties characterize a proper time-window network representation are never directly and comprehensively addressed.

We now review some papers that developed general evaluation frameworks for aggregated temporal networks and that we find worth mentioning. We begin with the supervised approach by Fish and Caceres, 2017 [28]. They start with a segmentation of the original network into large non-overlapping windows (on the order of 5 for a whole dataset). Then, they train a certain prediction task — link prediction, attribute prediction or change-point detection — considering the large segments by couples, with the older segment as the training set and the newer segment as the test set. In this way, they make a quantitative comparison between different windowing methods.

The point of this interesting approach that we find unsatisfactory is that they do not directly compare the aggregated network with the original network. This is precisely what we do through our fidelity score, defined in Section 3.1.2. Also, the rating of different methods depends on the choice of the prediction task. Despite this being an advantage if we are interested from the start on a specific type of analysis — let's say change-point detection — this task-dependence does not allow to establish whether the snapshot series is stable and reliable *in general*, i.e., also for other types of analysis and for visualization.

Mining temporal networks evolution, snapshots should respect the original time ordering of the contacts as much as possible. That is, their fidelity to the original contact structure should be maximized. If we are interested in diffusion processes, one of the most important topics of temporal networks, then it is crucial to directly control the causality of temporal chains of contacts, or *cascades*. As an example from epidemiology, let's consider the following case: a contact between A and B occurs at the timestamp t_1; then, a contact between A and C occurs at the timestamp $t_2 > t_1$. Now, if C is infected and the two contacts are projected onto the same snapshot, either we let the infection to spread across a same snapshot — so that B gets mistakenly infected — or we avoid spreading across a same snapshot — so that A remains mistakenly *not* infected. In general, the aggregation process modifies causal propagation between the nodes, with the risk of introducing artifacts onto the spreading cascades. Hence, a good windowing method should, at the same time, aggregate enough to have meaningful snapshots and control the loss of causal information.

The effect of time-window aggregation on diffusion processes has been quantified in [24] through the occupation probability of simple random walks. Another interesting paper is [29], where authors define and compute the occupancy of time-preserving paths between any two nodes; this quantity is used to find a threshold size for the window size beyond which it becomes risky to trust results coming from the analysis of the aggregated network. Both studies consider non-overlapping windows and warn against too large window sizes.

Our purpose is slightly different: taking causality into account requires a consistent computational effort, which can be imitating in the case of large and/or dense networks. Therefore, defining our property of fidelity to the original graph (Section 3.1.2), we did not directly relate to diffusion properties. Clearly, this also limits the potential of our score, as discussed in Sections 4.1.3 and 5.1.

Chapter 3

Proposed framework

3.1 Evaluation scores

3.1.1 Stability

In this section, we formally define the first of the two properties that we considered for the evaluation of snapshot graphs. Behind this property there is the idea to have smooth transitions between subsequent snapshots. Hence, we want a time-window aggregated network to be labeled as unstable if snapshots that are close in time differ too much from one another.

To define stability, we first have to choose a measure of how much a given snapshot is akin to another one, that we call *similarity* score. The one we mostly worked with is Jaccard coefficient. Given the two static networks $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ (we use the standard graph theory formalism where G is the graph, V the set of vertices — nodes — and E the set of edges — links) the Jaccard similarity between the two is:

$$J(G_1, G_2) = \frac{|E_1 \cap E_2|}{|E_1 \cup E_2|} \tag{3.1}$$

where $|E|$ is the cardinality of the set E, i.e., the number of elements in E. We easily observe that J is normalized: it equals 1 if the two sets of links are identical; 0 if none of the links is in common between the two. During our experiments, we also considered a *weighted* Jaccard coefficient. In this case, graphs are in the form $G = (V, E, W)$ (a real number w_e, the weight, is assigned to every edge e in E and stored in the set W) and the weighted Jaccard coefficient J_w becomes:

$$J_w(G_1, G_2) = \frac{\sum_{e \in E_1 \cup E_2} \min(w_{1,e}, w_{2,e})}{\sum_{e \in E_1 \cup E_2} \max(w_{1,e}, w_{2,e})} \tag{3.2}$$

where $w_{1,e}$ is the weight of edge e with respect to graph G_1 if $e \in E_1$; we fix $w_{1,e} = 0$ if $e \notin E_1$, that is, if $e \in E_2$. The same holds for $w_{2,e}$, mutatis mutandis.

Other choices are possible for the similarity score, but the Jaccard coefficient is among the most diffused and we believe that it intuitively represents the idea of a smooth transition between adjacent snapshots. A recent review of network similarity scores can be found in [33], including more sophisticated and thus less intuitive alternatives to the Jaccard score (based on entropy measures or graph Laplacian matrices, for instance).

Then, stability $S(A)$ of the aggregated network A is defined as a weighted average of snapshot-to-snapshot similarity:

$$S(A) = \frac{\sum_{G_1, G_2} J(G_1, G_2) \cdot \min(|E_1|, |E_2|)}{\sum_{G_1, G_2} \min(|E_1|, |E_2|)} \tag{3.3}$$

where the summations are over all the consecutive snapshots G_1 and G_2 in A. We weight similarities via the smallest cardinality of the smallest snapshot between the two. This allows to normalize with respect to the size and also to cut out degenerate situations where one of the two snapshots has very few links, like at the beginning or at the end of a school day in the datasets that we analyzed.

3.1.2 Fidelity

For this second property we did not find comparable definitions in the literature. As we reported in Section 2.4, very few articles proposed general frameworks to compare aggregation methods and we did not find any that matches well with our objectives. The idea behind this second score is that a time-window aggregated network has a low fidelity if it differs much from the original contact structure.

First of all, we chose to quantify the distance between a time-window aggregated temporal network and the original dataset with a generalized graph edit distance. Namely, we count how many contacts differ between the two networks. In terms of generalized adjacency matrices (tensors with time as the third component), we compute the distance $D(A)$ between the time-window aggregated network A and the original network O through an entry-wise L_1 norm:

$$D(A) = \|A - O\|_1 = \sum_{i,j,t} |a_{i,j,t} - o_{i,j,t}| \tag{3.4}$$

where entry $a_{i,j,t}$ equals 1 if at least one contact from node i to node j exist within the time window that contains timestamp t; while $o_{i,j,t}$ equals 1 if a contact was detected at the timestamp t. We recall that for each time window $[t_1, t_2]$ we replace each link (i, j) by an interval $(i, j, [t_1, t_2])$, if and only if there exist a contact $(i, j, t), t \in [t_1, t_2]$. $D(A)$ counts the number of "artificial" contacts introduced in this way: for each interval $(i, j, [t_1, t_2])$ we introduce as many false positives (FPs) as the number of times that there wasn't a contact (i, j, t) at the timestamp $t \in [t_1, t_2]$.

As we discuss in the next Section, we considered different ways to filter the data, in order to neglect those contacts that are not crucial for the structure of the temporal network, and that we can consider as noise. Filtering data in this way, we can reduce the number of FPs. As an example, let's consider the extreme case where a single contact occurs in a certain snapshot of a temporal network aggregated trough non-overlapping windows. Removing that contact we reduce $D(A)$ by the number of timesteps dt that fit in a time window of size w: w/dt (minus 1, for the only contact that was actually there). In other words, we get rid of all the FPs that the contact caused, at the cost of introducing a single false negative (FN) for the contact itself which is filtered out. In general, increasing the number of filtered contacts we have that FPs decrease and FNs increase, their sum being $D(A)$.

Distance is inversely correlated to what we want to call fidelity. The limit cases are $D(A) = 0$ — if A and O are equal, so that $D(A) = \text{size}(A) = \text{size}(O)$ (where the function "size" counts the total number of entries) — and $D(A) = 1$ — if all the entries are different. To have a fidelity score of 1 in the first case and 0 in the second case, we define $F(A)$ as:

$$F(A) = 1 - \frac{D(A)}{\text{size}(A)} \tag{3.5}$$

3.2 Filtering

To explain the principle of our filtering procedure, we make an analogy with astrophotography: while taking a long-exposure picture, we could want to capture a planet or a galaxy; but during the exposure, closer objects as a shooting star could rapidly pass throw the field of view, and a background noise (from the atmosphere and electronics) causes pixel to stochastically blink. Processing the photograph, we could remove the faster objects and smooth out the noise, to only keep the actual subject. Links lasting for only a small interval or blinking in the form of isolated contacts are considered respectively as fast shooting stars and as a background noise with respect to the contact structure. That is, they are secondary actors that we want to filter out because they do not really take part in the object that we want to capture.

We point out that the purpose of our filtering procedure is similar to backbone extraction for the reduction of static networks' density and noisiness [34]. In analogy with this process, we also consider temporal networks as made of a core structure of the most important links, surrounded by a cloud of less meaningful links — the noise.

3.2.1 Link weights

The time-window aggregation process returns a single interval that covers the whole duration of the window $[t_1, t_2]$, for every link that presents at least one contact with timestamp $t \in [t_1, t_2]$. Every

detail on the time ordering inside a same snapshot is lost, but to choose which links to keep and which to filter out, a link weight is computed while projecting onto the snapshots.

The simplest choice possible is to count the number of contacts (NOC). In this first case, the distribution of the timestamps is not considered: link A with five consecutive contacts and link B with five contacts isolated from one another will have the same weight. The filtering procedure is to fix a threshold value θ and to only keep the links for which the link weight is greater than θ, which are expected to be the most important in determining the contact structure. The value of θ can be fixed for all the windows or window-dependent: in Chapter 4 we fix a percentage N and compute θ in order to filter out the N% of the links with the lowest weight.

The other weights that we consider take the intervals structure into account. Therefore, a list of intervals is considered for each link on a given window, instead of a list of contacts. In other words, in this case we consider the temporal network as an interval graph. The idea to perform the windowing becomes to keep the stabler links, that is those for which contacts are organized into few relatively long intervals. We thus filter out links with many small intervals, lasting even one single contact (20 s), because they are less likely to represent a real conversation.

The first possibility is to simply count the number of intervals (NOI). In this case, "high-quality" links passing the filter are those with smaller weight. Another possibility to directly penalize links with short intervals is to combine the NOC with the NOI with the ratio:

$$\frac{\text{NOC}}{\text{NOI}} \tag{3.6}$$

which is equivalent to the average intervals' duration. In this way, a link A with a single interval lasting for half of the window duration and a link B with a single contact over the whole window have different weights. As for the simple NOC, important links for temporal network structure are those with a high value.

All the link weights proposed until now present a decreasing exponential distribution across the links population. That is, most of the links have small weight and very few links have a much greater weight (see Section 6.1.1). Instead, another measure that we tried as link weight does not have this same distribution: it is broadly distributed in the range $[0, 1]$ (Figure 6.1). This last measure is taken in the following way: for every window and for every link we count how many times a contact is anticipated by another contact in the previous timestep. Then, we normalize this value link-wise by the NOC. This reduced to calculating:

$$1 - \frac{\text{NOI}}{\text{NOC}} \tag{3.7}$$

3.2.2 Filtering windows

In the previous section it is taken for granted that the windows on which the filtering is performed directly superpose with the aggregating windows. But in general, we can also take two different window sizes: we thus distinguish between a filtering windows' size w_f and an aggregating windows' size w_a. The idea to consider this general case came to our mind while filtering snapshots with small window size w, i.e., only few times the resolution timestep dt. In that case, only a few number of values for the weight are possible (possible values for the NOC are between 1 and w/dt). Hence, filtering can become rather meaningless and stochastic.

A valuable configuration to perform the filtering is with overlapping windows both for aggregation and filtering, eventually with different window sizes. To give an example, one of the configuration that we discuss in Chapter 4 has the values $w_f = 1\,\text{h}$ and $w_f = 10\,\text{min}$. We also tested the general configuration $w_a \neq w_f$ with non-overlapping windows, both in the case $w_f > w_a$ and $w_a > w_f$. Here, w_f/w_a (or w_a/w_f if $w_a > w_f$) has to be an integer, to have a finite number of windows of the smaller size inside of the window with greater size.

Finally, with overlapping windows we also consider a sort of convolution between two different weights: the link weights w_l as defined in Section 3.2.1 and a *time weight* depending on the position of contacts' timestamps within the window. The computation of the second weight requires a function $f(t)$, that we call a *mask*. For every link there is a list $t_1, t_2, t_3, \ldots$ of the timestamps for which it presents contacts within the window; the global weight of the link will thus be $[f(t_1) + f(t_2) + f(t_3) + \ldots] \cdot w_l$. In this view, we can see the previous filtering methods as the trivial case with $f(t) = 1$.

Chapter 4

Results

We tested our framework on different SocioPatterns datasets (Section 6.1.1), but given that the results are quite similar, we decided to present and discuss the results only for one of them. We chose a dataset of contacts in a high school [35]. Students from 9 classes were recorded during 5 days in December 2013. Hence, it had a global duration that was not too short — poor statistics — nor too long — heavy computations.

The datasets that we considered all belong to a particular category of temporal network, sometimes reported as proximity networks. As most of the measured temporal networks in human dynamics, our high-school dataset presents the characteristics of burstiness of the time structure and scale-free distributed quantities. We briefly recall these concepts in the Annex (Section 6.1.1). It is important to remember these two properties while looking at the results presented in this chapter, because even if the framework that we define could be useful for many other types of temporal networks, the results that we present here clearly depend on the specific dataset category that we chose.

The resolution of the dataset is $dt = 20\,\mathrm{s}$; it does not correspond to the sampling frequency, but it comes from a time-window aggregation at the source (as explained in Section 6.1.1). So, we can consider single rows on the data table either as instantaneous contacts happening at the timestamp t or as intervals on the form $[t - dt, t]$, depending on whether we are using the contact sequence representation or the interval graph representation, respectively. We used the first when working with non-overlapping windows and the second with overlapping windows, mainly for computational ease (we leveraged the tnetwork library to work with intervals).

4.1 Non-overlapping windows

We begin with the following configuration: non-overlapping windows for both filtering and aggregation. We tested different combinations of the window sizes w_f and w_a, respectively, and the most interesting results are discussed below. We start with the simplest case for link weights, which is the number of contacts (NOC). Before showing the results of our filtering procedures, we define a stochastic baseline to compare with. After, we show the behavior of FPs and FNs with respect to window sizes and threshold values. At the end, we discuss the effect of the choice of the link weights (Section 3.2.1). Quantitative comparisons are done in terms of the two scores of stability and fidelity (Section 3.1).

When not specified, the values of the window sizes for the case $w_f = w_a$ are the following:

$$w_a \in [1\,\mathrm{min},\ 2\,\mathrm{min},\ 5\,\mathrm{min},\ 10\,\mathrm{min},\ 20\,\mathrm{min},\ 30\,\mathrm{min},\ 1\,\mathrm{h},\ 2\,\mathrm{h},\ 24\,\mathrm{h}] \tag{4.1}$$

We usually skipped sizes smaller than $w = 1\,\mathrm{min}$ (namely $20\,\mathrm{s}$ and $40\,\mathrm{s}$), because link weights become meaningless if the aggregation window is only few times larger than the resolution time scale dt. For the case $w_f > w_a$ instead, we display the results for $w_f = 1\,\mathrm{h}$ combined with smaller values for w_a, again taken from the values above (Equation 4.1).

4.1.1 Filtering baseline

In this section, we consider a stochastic filtering of the contacts: for every snapshot, a portion of the links with at least one contact is selected, and each of their corresponding contacts within the snapshot

is removed. As a parameter, we fix the percentage value for the number of links to be removed among all the links in the snapshot.

We begin with the case $w_f = w_a = w$ (Figure 4.1) and take for w all the values in Equation 4.1, plus the timestep $dt = 20\,\text{s}$. A heat map of stability, computed with the unweighted Jaccard coefficient (Equation 3.1), is displayed in Figure 4.1. We notice a drop of the score while increasing the number of links removed (vertically, from top to bottom). This behavior is observed for all values of w, i.e., for every column. Hence, filtering stochastically we break the network structure and make consecutive snapshots less similar to each other.

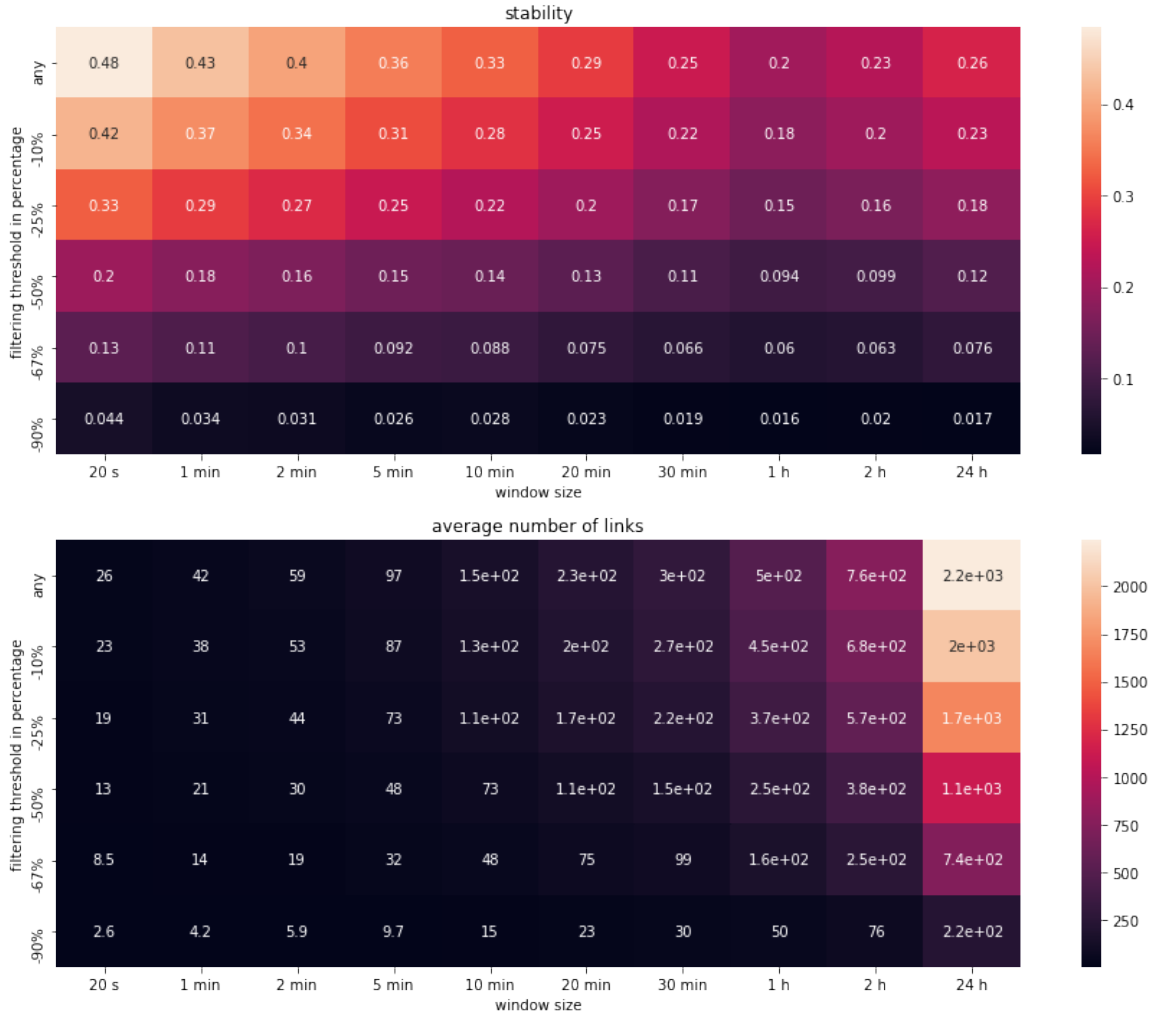

Figure 4.1: Stochastic filtering baseline for the case $w_f = w_a$. *Above*: heat map with stability score (inside boxes) against window size (horizontal axis) and filtering threshold in percentage (vertical axis, $-$N% means that N% of the links are removed). *Below*: heat map with the average number of links (not the NOC, which is higher) within each snapshot. Values can correspond *not* exactly to the percentages on the left because weights have integer values; therefore, all the links with weights equal to the threshold value have to be either included or excluded from the links to remove. In both cases there is a detachment from the actual percentage value.

Looking at the first row — without filtering — we see that the highest stability is at the smallest size, and the lowest is at the intermediate sizes. The behavior is thus non trivial, with a non-monotonic dependence with respect to the window size. The fact of having a very high stability at the resolution size dt depends on how the single contacts were defined, starting from the measurement: as explained in the Annex (Section 6.1.1) they are sort of already aggregated. This does not mean that it is only a matter of convention, but that data comes already aggregated at the smallest scale possible that is meaningful for the dynamics that is recorded. Relatively high scores are also found by aggregating over full days: this reflects the quite general circadian structure of human (and other living systems) dynamics [23].

A crucial aspect to notice is that absolute values are relatively low. Even at the resolution size the average Jaccard similarity is lower than 50% which means that more than half of the contacts

change from one snapshot to the other, in average. We are thus quite far from the idea of a motion picture with smooth transitions between frames. The situation gets even worse if we consider that at the smallest time scales the snapshots are "ugly" from a network science viewpoint (few nodes, too sparse), while at higher time scales, where networks are better structured, the score drops to 25-33%.

This is a key point for the orientation of our study. The idea to perform a filtering of the contacts came exactly from this finding. We were quite shocked to realize how far aggregated networks are from the ideal situation where single snapshots are well defined networks that also evolve smoothly in time. For the same reason we began to consider also overlapping windows, instead of only non-overlapping ones: to increase similarity between adjacent windows.

We now display the same type of heat map for the case $w_f \neq w_a$ (Figure 4.2). The stochastic filtering is done at $w_f = 1\,\mathrm{h}$: the links selected by chance are removed in every aggregation window — with smaller size w_a — that fits inside of the corresponding filtering window.

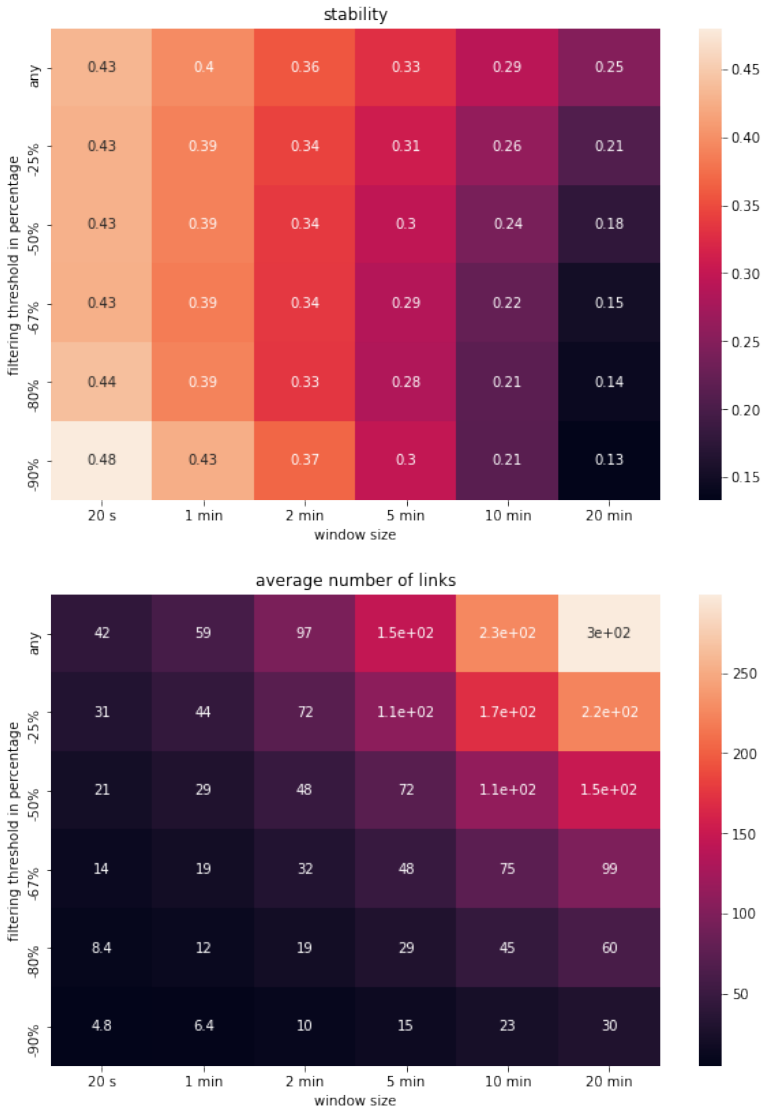

Figure 4.2: Stochastic filtering baseline for the case $w_f = 1\,\mathrm{h}$, $w_a < w_f$. Please refer to Figure 4.1 for details about the two heat maps.

Surprisingly, the qualitative behavior is rather different with respect to the previous case. The similarity score still decreases vertically for most of the window sizes, but far less. For the two smallest sizes and in the last row in particular, the score even increases slightly.

We believe that this difference between the two apparently not-so-different approaches lies in the particular distribution of contacts in face-to-face interaction networks (Section 6.1.1): very few links account for most of the contacts (the distribution of link weights is scale-free). Now, passing from w_a to the surrounding w_f, new links that are added are more likely to have a lower NOC, since they were not there before; thus, the number of links with higher NOC decreases, in percentage, and their

probability to not be drawn by chance increases. This hypothesis supports our idea that links making network structure are mainly those with higher link weight.

4.1.2 Filtering with number of contacts

As in the previous section, we fix a percentage of the links to be removed and let the absolute value of the threshold to vary accordingly, snapshot by snapshot. In this case, the choice is not stochastic, but according to the link weight: only the more frequent links within the time window are kept. The implementation is done following these steps:

1. Compute the weight of each link within the window
2. Sort the weights' list in ascending order
3. Save the threshold value θ corresponding to the position $p = \lfloor \mathrm{N}\% \cdot \ell \rfloor$, where N% is the filtering threshold and ℓ is the length of the weights' list
4. Remove all links for which the link weight is *strictly* less than θ

We detail the procedure to clarify the behavior of the average number of links within each snapshot, displayed by the second heat map in each figure. The key point is that having a strict "$<$" sign, there are actually more than (100−N)% of the values that pass the filter: all those with link weight greater *or* equal to the threshold value. A consequence of this choice is that, for instance, the stability is equal for all percentage thresholds in the range 0-25%, in the case $w_f = 1\,\mathrm{h}, w = 1\,\mathrm{min}$. This happens because the threshold value remains $\theta = 1$, which means no filtering at all (every link has at least a weight of 1 to be in the snapshot). Results for the case $w_f = w_a$ and $w_f = 1\,\mathrm{h}$ are in Figures 6.2 (in the Annex) and 4.3, respectively. We compare them to their respective baselines (Figures 4.1 and 4.2).

For the case $w_f = w_a$, the stability remains constant within a same column, instead of dropping from top to bottom (baseline heat map). This qualitative behavior is conserved at all window sizes. A rather different result is found in Figure 4.3, where the stability score increases monotonically from top to bottom. High absolute values are reached in the last row, especially for the columns at the middle, and comparing with the first row we notice that the score is more than doubled at $w \geq 5\,\mathrm{min}$. Values around 70% are reached when the first decile of the most frequent links is kept; a stability around 80% is reached at -95% (we recall that the number of links passing the filter is actually much bigger than 5%, around 80%). This is quite interesting for many reasons. First, even if values get doubled, stability can still be considered just fine: for a smooth transition between subsequent snapshots, we would prefer values around 80-90%. Second, a rather heavy filtering is necessary to reach high values: how much the obtained network remains faithful to the original where only 10% (or even less) of the links are kept? In the next section we show what fidelity score can tell us on this issue.

4.1.3 Fidelity: FPs vs FNs

We now display results for the second of the two evaluation properties defined in Section 3.1: fidelity to the original temporal network. The total distance between the filtered-aggregated network A and the original network O is computed through Equation 3.4; here, we want to distinguish between false positives (FPs, contacts in A but not in O) and false negatives (FNs, contacts in O but not in A).

Before showing our results, we discuss qualitatively what is expected. At fixed window size, the number of contacts that are cut out grows with the threshold value. So, FNs increase with the filtering threshold and FPs decrease. At fixed threshold, large windows have more space to be filled by FPs, so the number of FPs increases with w. Instead, the number of FNs is constant in the case with fixed $w_f = 1\,\mathrm{h}$, since the same links are removed at every window size (filtering window is in common among all the window sizes w_a); the behavior is not easily predictable in the case $w_f = w_a$. In both cases, we cannot know in advance what is the ratio between FPs and FNs, because this aspect depends on both the dataset and the filtering technique.

For the configuration $w_f = 1\,\mathrm{h},\ w_a < w_f$, we display some graphs at fixed threshold in Figure 4.4, some at fixed window size w_a in Figure 4.5. A complete 3D plot is available online. The analogues for the case $w_f = w_a$, with akin qualitative behavior, are available in the Annex (Figures 6.3 and 6.4). Firstly, we notice that the qualitative behavior is as expected, with a flat curve for FNs and a

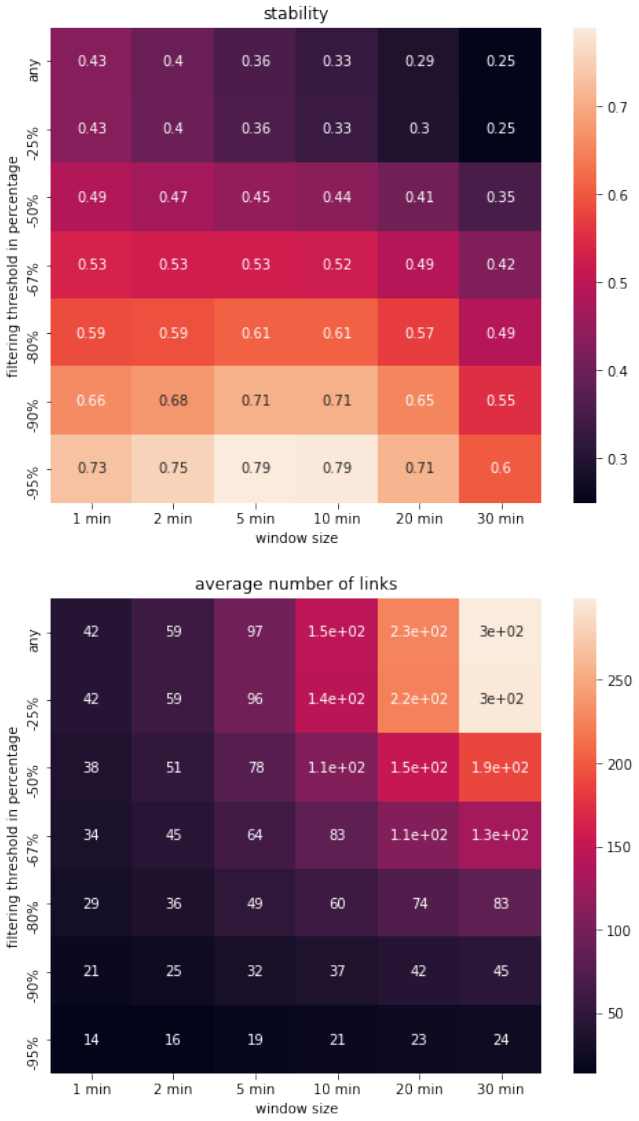

Figure 4.3: Filtering with the absolute number of contacts (NOC) for the case $w_f = 1\,\mathrm{h}$, $w_a < w_f$. *Above*: heat map with stability score (inside boxes) against window size (horizontal axis) and filtering threshold in percentage (vertical axis, −N% means that N% of the links are removed). *Below*: heat map with the average number of links (not the NOC, which is higher) within each snapshot. Values do *not* correspond exactly to what percentages on the left would give, as explained in the second paragraph of Section 4.1.2.

monotonic increase for FPs. Secondly, we can focus on the proportion between FPs and FNs. The first prevail until the threshold goes beyond 90%, then the number of FNs exceeds that of FPs, starting from the smallest values of w_a. This means that performing a time-window aggregation without filtering, as it is usually done in literature, plenty of FPs are introduced; but removing some of the links it is possible to reduce them to a quarter, at the cost of introducing a relatively small number of FNs. Comparing to distance curves for the stochastic baseline (Figure 6.5 in the Annex) we can see that the shape of the curves is almost conserved, but absolute values become a little smaller in the case of targeted filtering.

Moving to Figure 4.5, we observe that it also follows the expected qualitative behavior. The most interesting feature of the curves is perhaps that at high threshold the cumulative distance between original and aggregated network presents a minimum, or a plateau. This behavior comes from the opposite monotonicity of FPs and FNs. The qualitative behavior of the stochastic baseline (Figure 6.6 in the Annex and 3D plot online) is much different in this case. The most relevant difference is in the high filtering: there is no saturation (plateau or maximum) in the baseline, but a steep decrease towards zero.

We can interpret the region where distance is minimized — or it tends to a plateau — as suggesting a range of threshold values for which fidelity is optimized. The best aggregation would occur by filtering

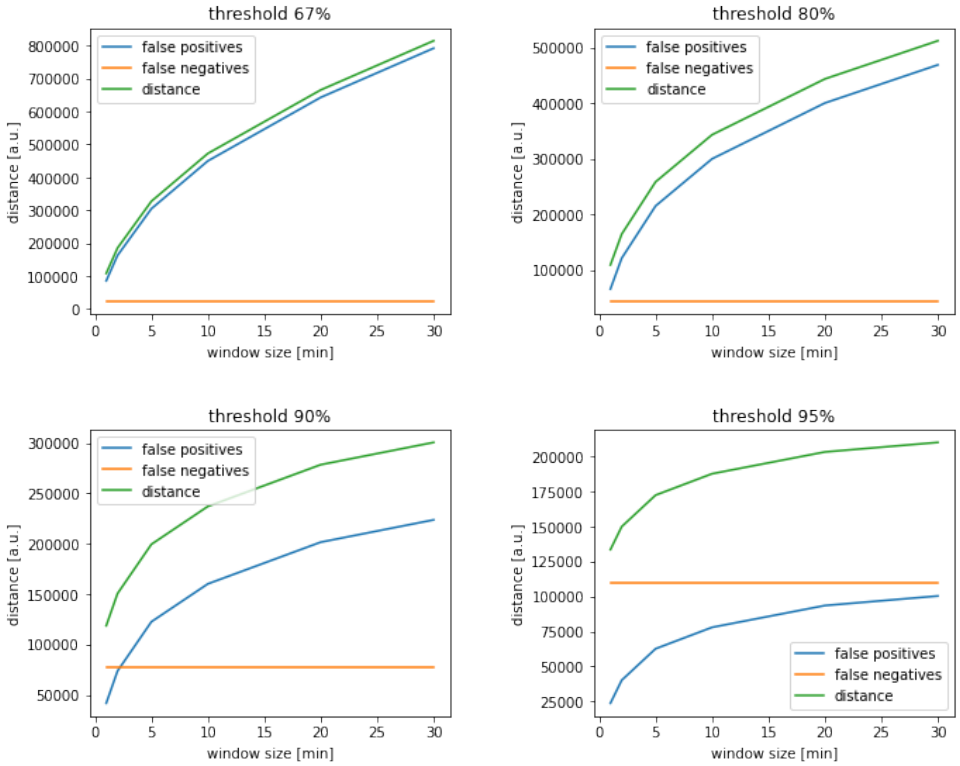

Figure 4.4: FPs, FNs and distance (Equation 3.4) between original network and filtered time-window aggregated network, in the configuration $w_f = 1\,\mathrm{h},\ w_a < w_f$. The fixed threshold value in percentage is indicated above each plot; note that percentage values do *not* correspond exactly to the actual number of links passing the filter, as explained in Section 4.1.2. The varying window size is on the horizontal axis.

out around 90% of the links in the network. It seems quite absurd to say that such a reduced subset is more representative of the original temporal network than by keeping all the links, because it would mean that most of the links do not really take part in making network structure (we remind that having a scale-free distribution for the NOC across the links population, all the links removed at -90% account for only 20% of the total number of contacts; that is why we consider that our filtering technique is appropriate).

There are mainly two possible interpretations: one is that for datasets of face-to-face interaction as the ones we worked with, the system behavior can truly be well characterized by only looking at the most active users, neglecting most of the others. In this case, temporal networks could be simplified in a rather straightforward manner. The second possibility is to think that rarer contacts are also crucial to characterize the network, therefore our fidelity score would not be a good measure of reliability in the time-window aggregated graph.

4.1.4 Filtering with other link weights

In this section we mention the results that we got using other choices for link weights, discussed in Section 3.2.1. We performed the same tests that are shown in the rest of this chapter, but replacing the simpler NOC with link weights defined by Equation 3.6 and Equation 3.7. Despite the two weights represent two different quantities, the resulting filtering is exactly the same, since moving from $f(x) = x$ (Equation 3.6) to $f(x) = 1 - \frac{1}{x}$ (Equation 3.7) does not change the ordering of link weight values. Therefore, we comment the obtained stability and fidelity for both at the same time.

Concerning stability (Figure 6.7 in the Annex), absolute values are quite similar with respect to the case with the NOC: they go from less than 50% for the unfiltered case to around 80% at high filtering (90-95%). But the position of the optimal region on the parameter grid is different: in the configuration $w_f = 1\,\mathrm{h},\ w_a < w_f$ stability peaks at the lowest scale of 1 min.

The interpretation is twofold: on the one side, it proves that different choices for the weight affect the stability score; on the other side, the dependence on the threshold value is conserved, qualitatively. Therefore, different weights end up discouraging the same region of the heatmap as less adapted for

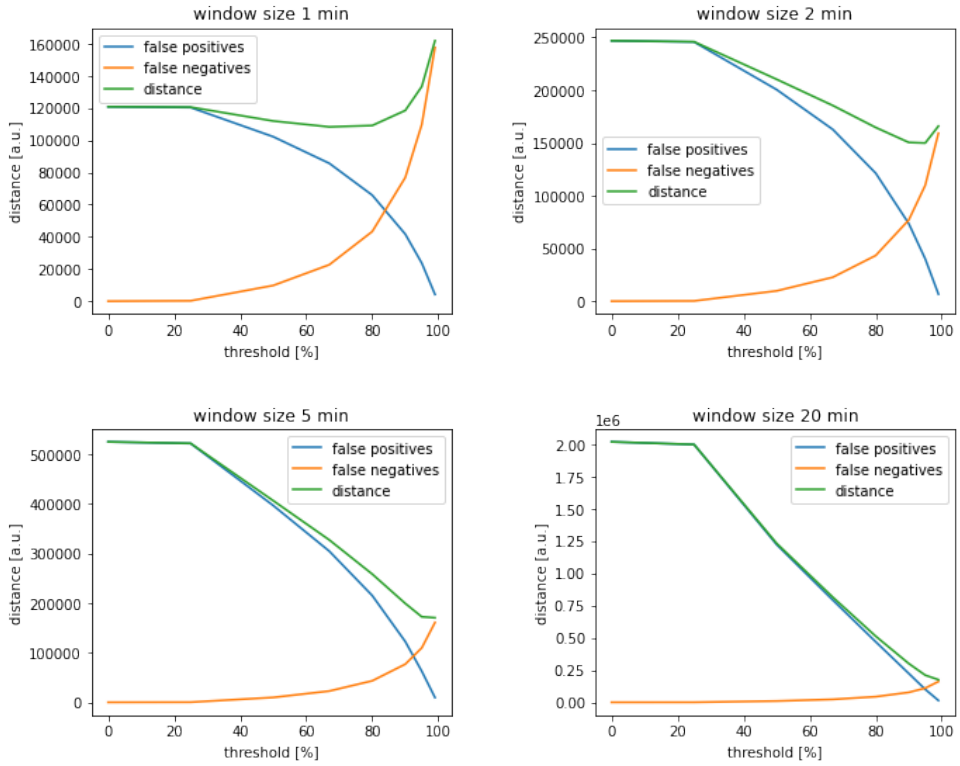

Figure 4.5: FPs, FNs and distance (Equation 3.4) between original network and filtered time-window aggregated network, in the configuration $w_f = 1\,\mathrm{h}$, $w_a < w_f$. The fixed window size is indicated above each plot; the varying threshold value in percentage is on the horizontal axis. Note that percentage values do not correspond exactly to the actual number of links passing the filter, as explained in Section 4.1.2.

time-window aggregation. This also means that, outcomes being akin, we can choose to use the weight that is more computationally efficient, namely the simple NOC.

The situation is rather similar for fidelity score. With weight as in Equations 3.6 and 3.7 the curves of FPs, FNs and distance are similar to the ones with the NOC (Figures 4.4 and 4.5).

4.2 Overlapping windows

In this section we display the results that we got with overlapping windows, shifted by a distance of a single time step $dt = 20\,\mathrm{s}$ from one another (Figure 2.3, b). We only show results for the case $w_f = 1\,\mathrm{h}$, $w_a < w_f$, to compare with the non-overlapped window aggregation. We remark that the situation is different however: here the aggregation window is always at the center of the filtering window, while in the non-overlapped case the smaller aggregation windows spanned all over the corresponding filtering window. As a consequence, the situation is actually in the middle between the configuration $w_f = w_a$ and $w_f > w_a$ of the non-overlapping window aggregation. Technical details on the computation of the fidelity score are available in the Annex (Section 6.1.2).

4.2.1 Stability and fidelity

Having windows that partially overlap, similarity between nearby snapshots becomes necessarily higher than with non-overlapping windows (Figure 6.8). With $w_f = 1\,\mathrm{h}$ and for $w_a \geq 5\,\mathrm{min}$, stability is higher than 90% but even at 1 min it is already at 75%. A filtering with the NOC increases stability even more, as expected, while the stochastic baseline (Figure 6.9) shows a drastic decrease with respect to the filtering percentage (rather close to the behavior of the case $w_f = w_a$ in non-overlapping windowing).

Curves of FPs, FNs (computed through Equations 6.1 and 6.2) and their sum (distance) are similar to the case of non-overlapping windows, both in shape and absolute values. Hence, on this side, there is no reason to prefer overlapping windows. We believe that this is due to the peculiar structure of our datasets, recording students in a school. The timetable is much regular during the week, with school days starting and ending always at the same hours. Hence, non-overlapping windows already work

quite well if they window size is a fraction of the school day *and* of the whole day of 24 h: it never happens that, for instance, a window of size 1 h is empty for all its period except the last 10 minutes — situation that would bring a lot of FPs —, as long as the beginning of the series of non-overlapping windows is in the right position.

We expect overlapping windows to allow a valuable improvement in fidelity for datasets where the periods of activity are distributed less regularly, so that non-overlapping windows become less effective.

4.2.2 Mask filters

As anticipated in Section 3.2.2, we also considered a convolution between the link weight and a time weight. In particular, we used a snapshot-dependent Gaussian mask $f(t) = e^{-\frac{(t-t_0)^2}{\sigma^2}}$ centered at the center t_0 of both the filtering and aggregation window: contacts toward the center have an higher time weight with respect to those at the limits of the window.

The idea is to take advantage of two time-scales at the same time: a larger $w_f = w_a$, which ensures to have a well-defined weights distribution and a smaller standard deviation σ of the Gaussian curve. Promoting links that are closer to the window center allows to "sample" at a higher resolution, possibly increasing fidelity and by tuning the value of σ we can sort of zoom in and out. To give an example, with the parameters $w_f = w_a = 1\,\text{h}$, $\sigma = 5\,\text{min}$ and $\theta = 3$, a link presenting a series of 5 — more than θ — contacts towards the beginning of the time window will be filtered out, considering that the link was already included in the filtered-aggregated network when the center of the window was closer to the series of contacts.

We were not able to prove a quantitative improvement with respect to weighting at $f(t) = 1$, on our datasets. This means that either we did not find the correct pair of w_f and σ, or the approach is not much valuable for the type of datasets that we considered. For instance, it could simply correspond to using a smaller window size. Nevertheless, we found interesting to explore this possibility of mixing weights and wanted to leave our idea as a suggestion for further developments.

Chapter 5

Conclusions

5.1 Future perspectives

To test how much the region of the parameter grid (of window size and threshold value) indicated by our scores is actually representative of the entire temporal network, it would be interesting to use independent evaluation frameworks. A starting point would be to visualize programmatically the filtered-aggregated networks and to look at time series of standard measures in network science, static or temporal [36].

Then, papers [28] and [29] (Section 2.4) propose interesting alternative approaches: predictions from these frameworks and from ours could be compared. As it is done in [28], we could choose some of the methods for automatic aggregation mentioned throughout Section 2.3 and test the aggregated networks that they produce, on the basis of our two scores.

This being said, we stress that our goal was more to suggest which window sizes could be less adapted for time-window aggregation, rather than to indicate the best situation overall. Different needs could indeed require different window sizes: our proposition is to always verify if the choice ensures not-too-bad similarity and fidelity, before performing an analysis on the aggregated graph. This point is crucial to understand the orientation of our study. We are aware that a blind maximization of stability and fidelity can bring to a too-heavy filtering, so that the snapshots end up having only very few links. This is a manifest limitation of our framework. A promising solution would be to modify the scores, or to add a third one, to control the size — or sparsity or other related properties — explicitly, to avoid an exaggerated "shrinking" of the network through filtering.

Another possible research direction is to define new quantities for temporal networks' comparison. Our scores or link weights could be modified to include time dependence of the single contacts more explicitly. For instance, link weights could decay in time, as the weights defined in [37]. This type of modifications can be done by tuning the mask $f(t)$ that we defined. In any case, it is important to control computational efficiency, so as not to fall in the problem of non-scalability that we wish to avoid.

For overlapping windows, we only considered the case of a shift $s = dt$ (2.3). It would be interesting to increase the value of s in order to have a situation in between overlapping at the resolution scale and non-overlapping windows. This would be interesting for two reasons. Firstly, from a computational point of view, since a reduction of the number of snapshots with respect to the case $s = dt$ would speed up the aggregation algorithm. Secondly, we can imagine to adjust the shift s with respect to the activity of the network, going faster (high value of s) when activity is low — there is not much going on in the network — and slower (small value of s) when the activity is high — so that it is worthwhile to augment time resolution of the snapshot. This recalls the objective of automatic windowing methods that we group in Section 2.3 under the name of dynamic windowing.

As a rather collateral result, we defined by Equation 3.7 a quantity that does not have a scale free distribution. It could be interesting to look at its distribution across different temporal networks, to see if it could be used as a meaningful analysis tool.

5.2 Discussion

The aim of our investigation was to contribute to the ongoing forge of temporal network's research field, that is, to the characterization of webs of interactions that evolve in time. In particular, we worked on one promising possibility to represent these networks, which is through the time-window aggregation. It consists in grouping data that is close in time under a same "snapshot", so that the temporal network becomes a series of frames made by static networks. The reason why we do not work directly on the temporal network as it is from the measurements — a table of instantaneous interactions — is that representations as the time-window aggregated graph can be handier for further analysis and visualization.

The aggregation process is non-trivial, as there are many ways to aggregate and different free parameters to set. It is thus important to establish quantitative frameworks to compare different time-window aggregated networks, both between them and with the original temporal network. With this in mind, we defined two different scores and tested them on datasets of face-to-face interactions between humans. These two scores are stability — which promotes smooth transition between the snapshots — and fidelity — which controls the difference between the time-window aggregated network and the original temporal network.

Moreover, we showed how to perform a targeted filtering of the links, leveraging a link weight that is computed while aggregating. This process allows to compress networks by removing, snapshot by snapshot, all the links with weight below — or above — a certain threshold. We compared two possibilities: (i) to filter on windows that correspond to the aggregating windows or (ii) to take larger windows for filtering than for aggregation. We found that in the second case the stability increases monotonically with respect to the filtering threshold value. These observations open up a lot of questions and also show the limits of our framework.

We found that simple (without filtering) aggregation with non-overlapping windows returns small values of stability, below 50%; even smaller if the aggregation window increases. But this is precisely the most common configuration for time-window aggregation that is used in the literature, where the resulting snapshot network is next used for the analysis of temporal network's evolution. We state that, proceeding in this way, authors can end up working with time series that are substantially unstable, with most of the links changing from one snapshot to the next one.

According to our evaluation framework, it is convenient to aggregate at small window sizes (around 1 min) and to filter out around 90% of the links in the network. We proved quantitatively this outcome through the computation of our measures of stability and distance. The latter is inversely correlated with fidelity and it represents rather intuitively the difference between the original and the aggregated temporal network. It can be striking to see that we suggest to filter out 90% of the links. But since the links that are maintained are the most important — the remaining 10% of them accounts for about 80% of the entire sequence of contacts — we believe that with datasets of face-to-face interactions as the ones we worked with, the resulting subset is truly representative of the original temporal network.

To conclude, we remark that it is crucial to test our evaluation framework on many other datasets too, in order to judge the specificity of our findings. It would be important to use datasets of both the same type and different type with respect to ours, to test the generality of our results and the potentiality of the framework we suggest.

Bibliography

[1]R. Molontay and M. Nagy, "Two decades of network science: as seen through the co-authorship network of network scientists", in Proceedings of the 2019 ieee/acm international conference on advances in social networks analysis and mining (2019), pp. 578–583.

[2]A.-L. Barabási, *Linked: the new science of networks*, 2003.

[3]K. Börner, S. Sanyal, and A. Vespignani, "Network science", Annual review of information science and technology **41**, 537–607 (2007).

[4]D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world'networks", nature **393**, 440–442 (1998).

[5]A.-L. Barabási and R. Albert, "Emergence of scaling in random networks", science **286**, 509–512 (1999).

[6]M. Girvan and M. E. Newman, "Community structure in social and biological networks", Proceedings of the national academy of sciences **99**, 7821–7826 (2002).

[7]L. A. Adamic and N. Glance, "The political blogosphere and the 2004 us election: divided they blog", in Proceedings of the 3rd international workshop on link discovery (2005), pp. 36–43.

[8]P. Holme, "Modern temporal network theory: a colloquium", The European Physical Journal B **88**, 1–30 (2015).

[9]P. Holme and J. Saramäki, "Temporal networks", Physics reports **519**, 97–125 (2012).

[10]M. Latapy, T. Viard, and C. Magnien, "Stream graphs and link streams for the modeling of interactions over time", Social Network Analysis and Mining **8**, 1–29 (2018).

[11]M. Jacomy, T. Venturini, S. Heymann, and M. Bastian, "Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software", PloS one **9**, e98679 (2014).

[12]M. Bastian, S. Heymann, and M. Jacomy, "Gephi: an open source software for exploring and manipulating networks", in Proceedings of the international aaai conference on web and social media, Vol. 3, 1 (2009).

[13]Y. Liu, T. Safavi, A. Dighe, and D. Koutra, "Graph summarization methods and applications: a survey", ACM Computing Surveys (CSUR) **51**, 1–34 (2018).

[14]B. Adhikari, Y. Zhang, A. Bharadwaj, and B. A. Prakash, "Condensing temporal networks using propagation", in Proceedings of the 2017 siam international conference on data mining (SIAM, 2017), pp. 417–425.

[15]N. Shah, D. Koutra, T. Zou, B. Gallagher, and C. Faloutsos, "Timecrunch: interpretable dynamic graph summarization", in Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining (2015), pp. 1055–1064.

[16]Q. Qu, S. Liu, C. S. Jensen, F. Zhu, and C. Faloutsos, "Interestingness-driven diffusion process summarization in dynamic networks", in Joint european conference on machine learning and knowledge discovery in databases (Springer, 2014), pp. 597–613.

[17]H. Wang, J. Wu, X. Zhu, Y. Chen, and C. Zhang, "Time-variant graph classification", IEEE Transactions on systems, man, and cybernetics: systems **50**, 2883–2896 (2018).

[18]M. Berlingerio, F. Bonchi, B. Bringmann, and A. Gionis, "Mining graph evolution rules", in Joint european conference on machine learning and knowledge discovery in databases (Springer, 2009), pp. 115–130.

[19]B. Wackersreuther, P. Wackersreuther, A. Oswald, C. Böhm, and K. M. Borgwardt, "Frequent subgraph discovery in dynamic networks", in Proceedings of the eighth workshop on mining and learning with graphs (2010), pp. 155–162.

[20]B. Klimt and Y. Yang, "The enron corpus: a new dataset for email classification research", in European conference on machine learning (Springer, 2004), pp. 217–226.

[21]S. Saganowski, P. Bródka, and P. Kazienko, "Influence of the dynamic social network timeframe type and size on the group evolution discovery", arXiv preprint arXiv:1210.5167 (2012).

[22]P. Sapiezynski, A. Stopczynski, D. D. Lassen, and S. Lehmann, "Interaction data from the copenhagen networks study", Scientific Data **6**, 1–10 (2019).

[23]G. Krings, M. Karsai, S. Bernhardsson, V. D. Blondel, and J. Saramäki, "Effects of time window size and placement on the structure of an aggregated communication network", EPJ Data Science **1**, 1–16 (2012).

[24]B. Ribeiro, N. Perra, and A. Baronchelli, "Quantifying the effect of temporal resolution on time-varying networks", Scientific reports **3**, 1–5 (2013).

[25]R. Sulo, T. Berger-Wolf, and R. Grossman, "Meaningful selection of temporal resolution for dynamic networks", in Proceedings of the eighth workshop on mining and learning with graphs (2010), pp. 127–136.

[26]S. Soundarajan, A. Tamersoy, E. B. Khalil, T. Eliassi-Rad, D. H. Chau, B. Gallagher, and K. Roundy, "Generating graph snapshots from streaming edge data", in Proceedings of the 25th international conference companion on world wide web (2016), pp. 109–110.

[27]S. Uddin, N. Choudhury, S. M. Farhad, and M. T. Rahman, "The optimal window size for analysing longitudinal networks", Scientific reports **7**, 1–15 (2017).

[28]B. Fish and R. S. Caceres, "A supervised approach to time scale detection in dynamic networks", arXiv preprint arXiv:1702.07752 (2017).

[29]Y. Léo, C. Crespelle, and E. Fleury, "Non-altering time scales for aggregation of dynamic networks into series of graphs", Computer Networks **148**, 108–119 (2019).

[30]R. K. Darst, C. Granell, A. Arenas, S. Gómez, J. Saramäki, and S. Fortunato, "Detection of timescales in evolving complex systems", Scientific reports **6**, 1–8 (2016).

[31]M. De Domenico, V. Nicosia, A. Arenas, and V. Latora, "Structural reducibility of multilayer networks", Nature communications **6**, 1–9 (2015).

[32]J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu, "Graphscope: parameter-free mining of large time-evolving graphs", in Proceedings of the 13th acm sigkdd international conference on knowledge discovery and data mining (2007), pp. 687–696.

[33]N. Masuda and P. Holme, "Detecting sequences of system states in temporal networks", Scientific reports **9**, 1–11 (2019).

[34]M. Coscia and F. M. Neffke, "Network backboning with noisy data", in 2017 ieee 33rd international conference on data engineering (icde) (IEEE, 2017), pp. 425–436.

[35]R. Mastrandrea, J. Fournet, and A. Barrat, "Contact patterns in a high school: a comparison between data collected using wearable sensors, contact diaries and friendship surveys", PloS one **10**, e0136497 (2015).

[36]V. Nicosia, J. Tang, C. Mascolo, M. Musolesi, G. Russo, and V. Latora, "Graph metrics for temporal networks", in *Temporal networks* (Springer, 2013), pp. 15–40.

[37]P. Holme, "Epidemiologically optimal static networks from temporal network data", PLoS computational biology **9**, e1003142 (2013).

[38]P. Vanhems, A. Barrat, C. Cattuto, J.-F. Pinton, N. Khanafer, C. Régis, B.-a. Kim, B. Comte, and N. Voirin, "Estimating potential infection transmission routes in hospital wards using wearable proximity sensors", PloS one **8**, e73970 (2013).

[39]J. Stehlé, N. Voirin, A. Barrat, C. Cattuto, L. Isella, J.-F. Pinton, M. Quaggiotto, W. Van den Broeck, C. Régis, B. Lina, et al., "High-resolution measurements of face-to-face contact patterns in a primary school", PloS one **6**, e23176 (2011).

[40]V. Gemmetto, A. Barrat, and C. Cattuto, "Mitigation of infectious disease at school: targeted class closure vs school closure", BMC infectious diseases **14**, 1–10 (2014).

[41]C. Cattuto, W. Van den Broeck, A. Barrat, V. Colizza, J.-F. Pinton, and A. Vespignani, "Dynamics of person-to-person interactions from distributed rfid sensor networks", PloS one **5**, e11596 (2010).

[42]A.-L. Barabasi, "The origin of bursts and heavy tails in human dynamics", Nature **435**, 207–211 (2005).

[43]M. Karsai, H.-H. Jo, K. Kaski, et al., *Bursty human dynamics* (Springer, 2018).

Chapter 6

Annex

6.1 Methods

We coded in Python 3 all the experiments that made us define the content of this report and that generated the results and pictures that we show. The main libraries that we used are pandas, numpy and tnetwork, for the computations and matplotlib, seaborn and plotly for the visualization. We found useful to work with interactive Jupyter notebooks on Google Colaboratory platform.

For networks visualization we used the Gephi software. Since the focus of our investigation was in temporal networks representation, not on their visualization, we only used this software to get an insight into the datasets we worked with, but we did not display images generated with Gephi.

This report was written in LaTeX on the Overleaf platform.

6.1.1 Datasets

All the datasets that we worked with are available at http://www.sociopatterns.org/datasets/. We downloaded and make use of the datasets named: "Hospital ward dynamic contact network" [38], "High school contact and friendship networks" [35] and "Primary school temporal network data" [39, 40]. They all record person-to-person interactions across closed communities (patients and doctors in the hospital, students and teachers in the school), for a duration on the order of the day or week. Each participant wears a detector (a Radio Frequency Identification Device, RFID) that is able to both receive and send signals. Since radio frequencies are absorbed by our body, transmission and reception can only happen from the front: this allows to naturally select proximity events that are more likely linked with having a conversation, which is the type of interactions in which the SocioPatterns collaboration is interested.

As explained on the SocioPatterns site and in [41], a non-trivial process is necessary to write down the contact sequence from the single events of detection. Very briefly, they end up using a sliding (overlapping) window of size 20 s, which allows to capture real face-to-face interactions with a probability of 99%. Hence, a contact with timestamp t stands for a conversation that has happened during the interval $[t - 20\,\mathrm{s}, t]$.

In the case of the dataset for which we displayed the results in this report, the basic units are students and teachers from 9 classes of a high-school in Marseilles (FR). The data gathering went on for 5 days in total, during the month of December 2013. Each row of the data table contains the following information: the ID labels of the two persons interacting; the timestamp; two additional labels for the category of the two nodes, as students' class. The total number of rows, i.e., the total number of contacts, is 188508. There are 327 different nodes and 5818 distinct links (out of the $327 \cdot 326/2 = 53301$ possible node pairs). The size of snapshot graphs at different window sizes can be read directly on the corresponding heatmaps.

SocioPatterns datasets fall into the field of human dynamics [42, 43] and social communication between humans or other living systems. Typical features of this type of temporal networks are: (i) burstiness in time of the contacts' structure, (ii) scale-free distributions for static centralities as the degree or temporal quantities as the waiting time between two consecutive contacts. (i) means that contacts are not distributed regularly in time, but short periods of high activity — with many contacts close in time — are spaced out by longer periods of low activity — with few contacts occurring now

and then. This happens at every scale: for single links, for the activity of a community as a whole or for the entire population. An important consequence of (ii) is that few nodes and links account for the majority of all the contacts of the temporal network. To give numbers that are meaningful for the results that we show, in our high-school dataset the 10% of the most frequent links make about 80% of the total number of contacts.

6.1.2 Computation of fidelity with overlapping windows

With non-overlapping windows we directly counted the number of false positives (FPs) and false negatives (FNs), snapshot-by-snapshot. But if the time windows do superpose, this simple procedure cannot be used. Hence, we proceeded in this way: first we build a filtered-aggregated interval graph, next we compute all FPs and FNs via a link-wise difference between the series of intervals in the filtered-aggregated graph and in the original interval graph.

The first part is done by considering a time window of size w_f, sliding by $s = dt$ at each iteration. At each step we update the link weights and keep track of which links are above the filtering threshold. When a new link joins this set of "accepted" links, we assign to it the timestamp that accounts for the beginning of the time interval during which it passes the filter: $t - w_a/2$, where t is the center of the filtering and aggregating windows and $-w_a/2$ is there because we consider convert links passing the filter into intervals that cover the entire time window. When a link goes beyond the threshold and leaves the set of "accepted" links, then we recover the timestamp of the beginning of its interval and complete it with the timestamp of the end of the interval: $t+w_a/2$, for symmetric reasons with respect to the one above.

For the second part, we first need to convert the original contact sequence into an interval graph. To do so, we run the same code that is used to build the filtered-aggregated interval graphs, but with the following parameters: $w_a = 20\,\mathrm{s}$ and $\theta = 0$. Then, for every link, we convert the differences between the sets of intervals in the aggregated-filtered interval graph A and in the original interval graph O into the number of FPs and FNs. Formally, naming E the set of all links, $I_{e,O}$ and $I_{e,A}$ the sets of intervals for link e in O and A, respectively, and using $|I_{e,*}|$ for the total duration of the set of intervals $I_{e,*}$, we have:

$$\mathrm{FPs} = \sum_{e \in E} \frac{|I_{e,A}| - |I_{e,A} \cap I_{e,O}|}{dt} \tag{6.1}$$

$$\mathrm{FNs} = \sum_{e \in E} \frac{|I_{e,O}| - |I_{e,A} \cap I_{e,O}|}{dt} \tag{6.2}$$

6.2 Supplementary figures

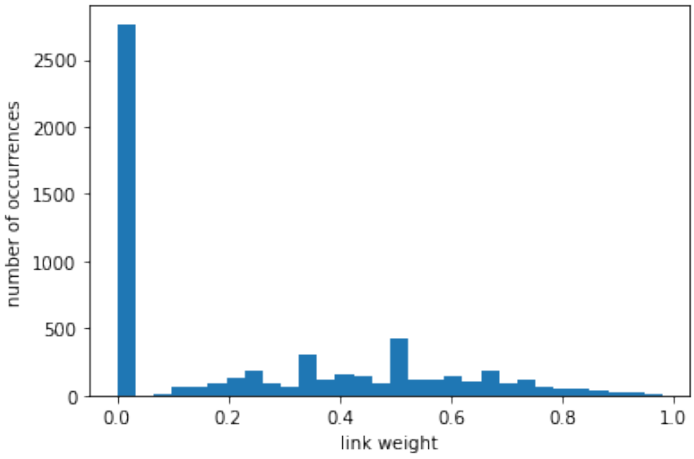

Figure 6.1: Distribution of the link weights defined by Equation 3.7 across the entire dataset. Horizontal axis is for the weight value, vertical axis reports the absolute number of links on the corresponding bin. Links with weight 0 are those with only one single contact, i.e., no intervals at all.

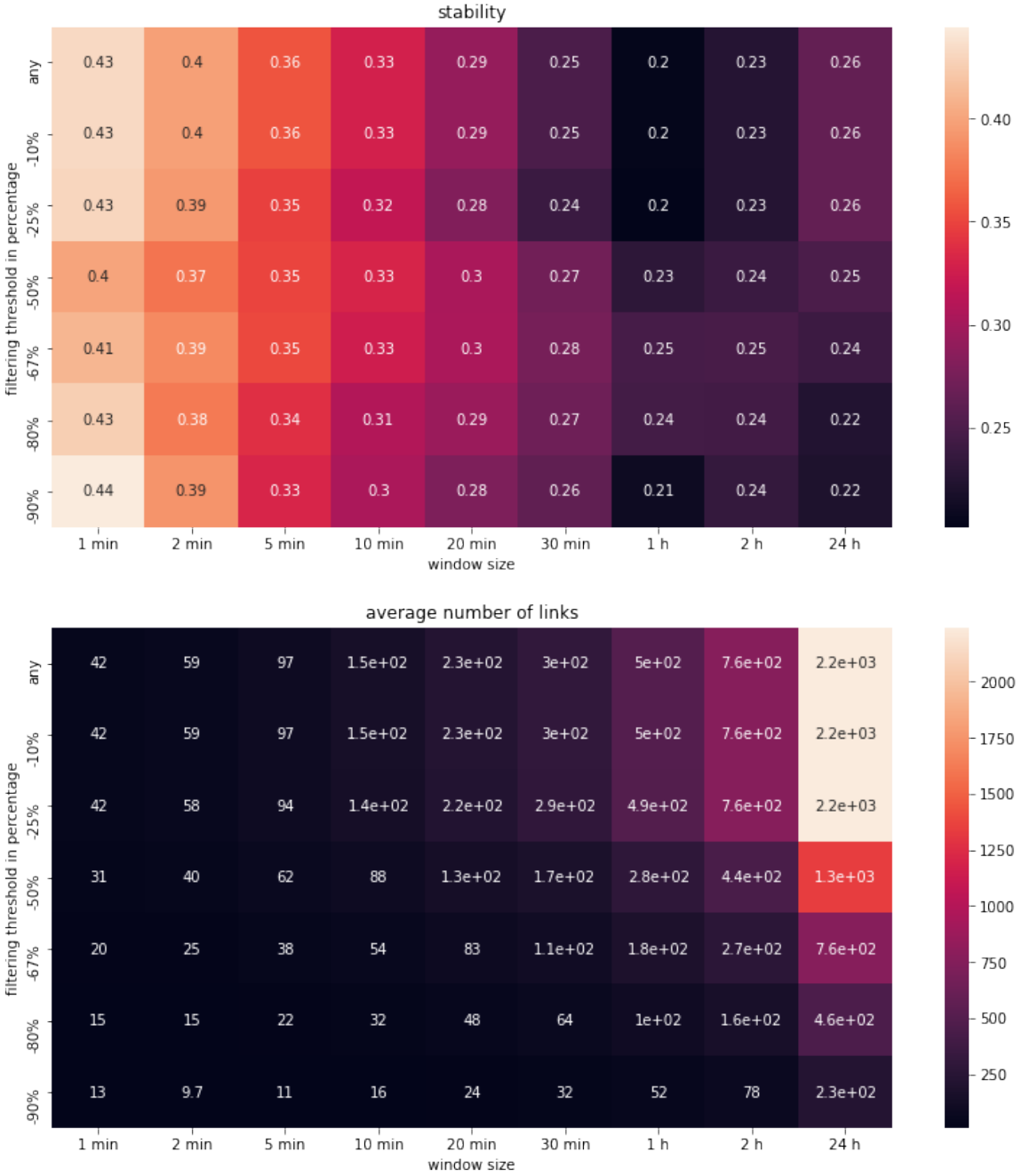

Figure 6.2: Filtering with the absolute number of contacts (NOC) for the case $w_f = w_a$. *Above*: heat map with stability score (inside boxes) against window size (horizontal axis) and filtering threshold in percentage (vertical axis, $-$N% means that N% of the links are removed). *Below*: heat map with the average number of links (not the NOC, which is higher) within each snapshot. Values do *not* correspond exactly to what percentages on the left would give, as explained in the second paragraph of Section 4.1.2.

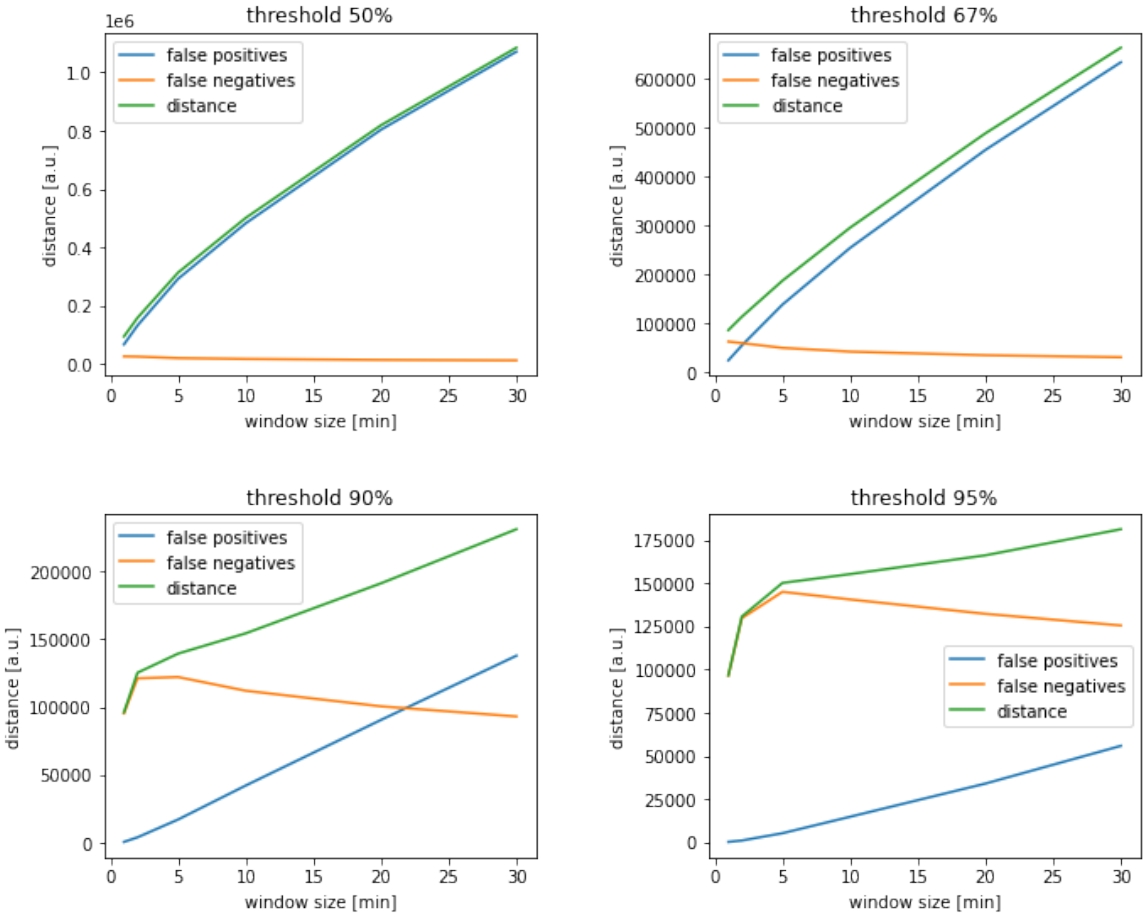

Figure 6.3: FPs, FNs and distance (Equation 3.4) between original network and filtered time-window aggregated network in the configuration $w_f = w_a$. The fixed threshold value in percentage is indicated above each plot; note that percentage values do not correspond exactly to the actual number of links passing the filter, as explained in Section 4.1.2. The varying window size is on the horizontal axis.

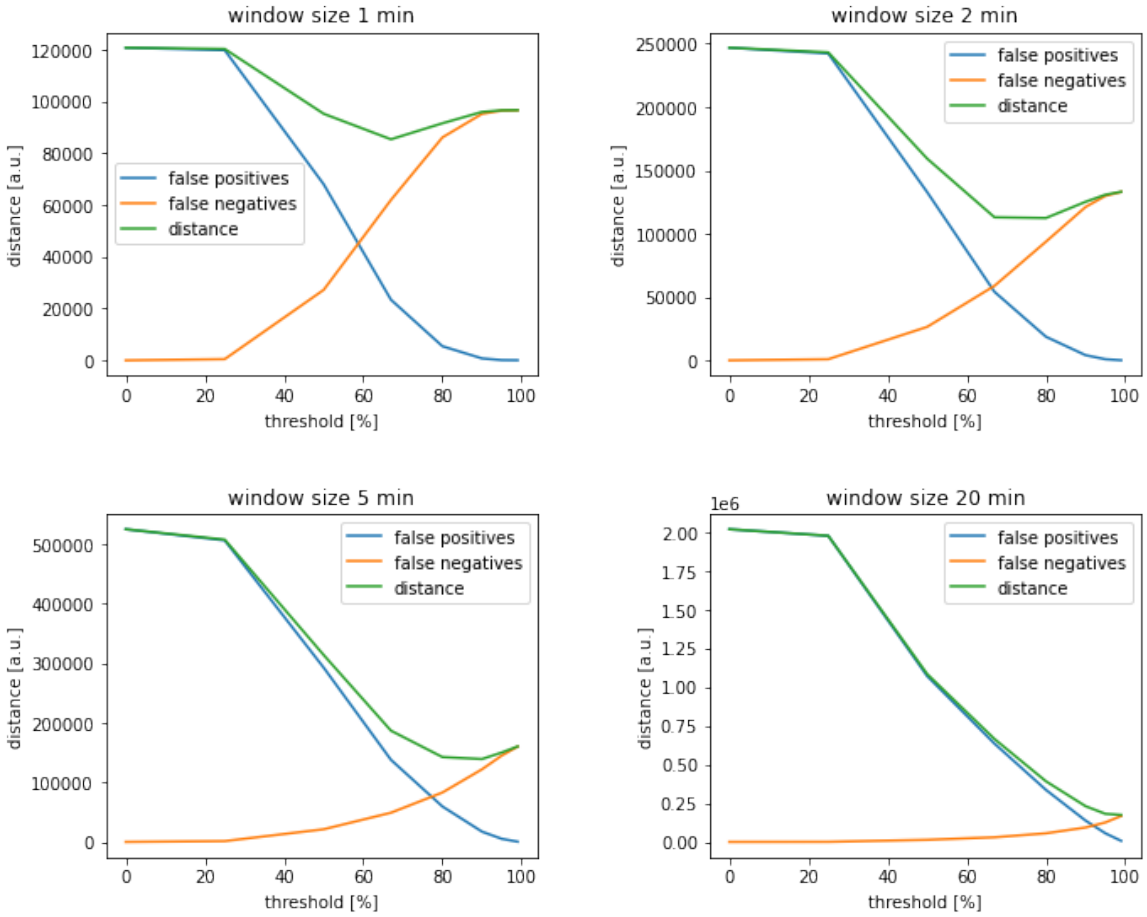

Figure 6.4: FPs, FNs and distance (Equation 3.4) between original network and filtered time-window aggregated network in the configuration $w_f = w_a$. The fixed window size is indicated above each plot; the varying threshold value in percentage is on the horizontal axis. Note that percentage values do not correspond exactly to the actual number of links passing the filter, as explained in Section 4.1.2.

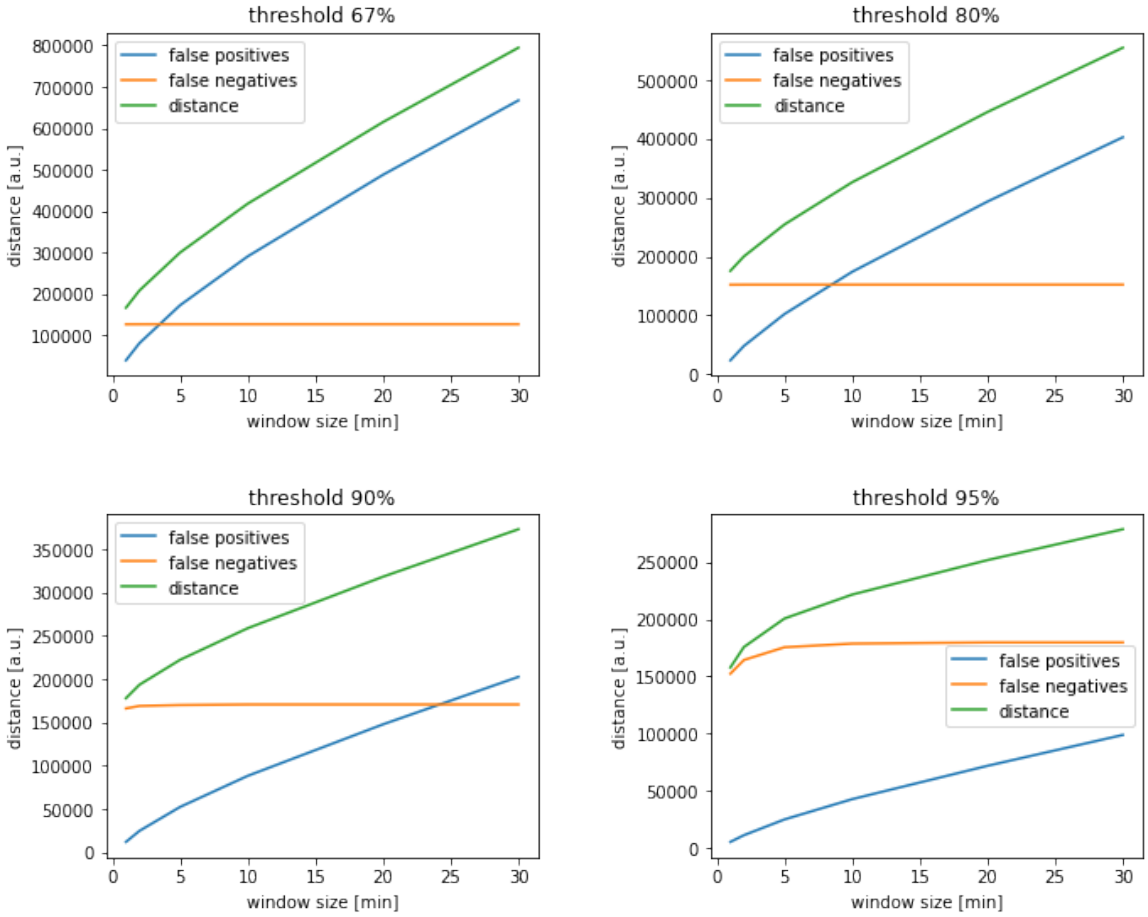

Figure 6.5: Stochastic filtering baseline for FPs, FNs and distance (Equation 3.4) between original network and filtered time-window aggregated network in the configuration $w_f = 1\,\mathrm{h}$, $w_a < w_f$. The fixed threshold value in percentage is indicated above each plot; note that percentage values do not correspond exactly to the actual number of links passing the filter, as explained in the caption of Figure 4.1. The varying window size is on the horizontal axis.

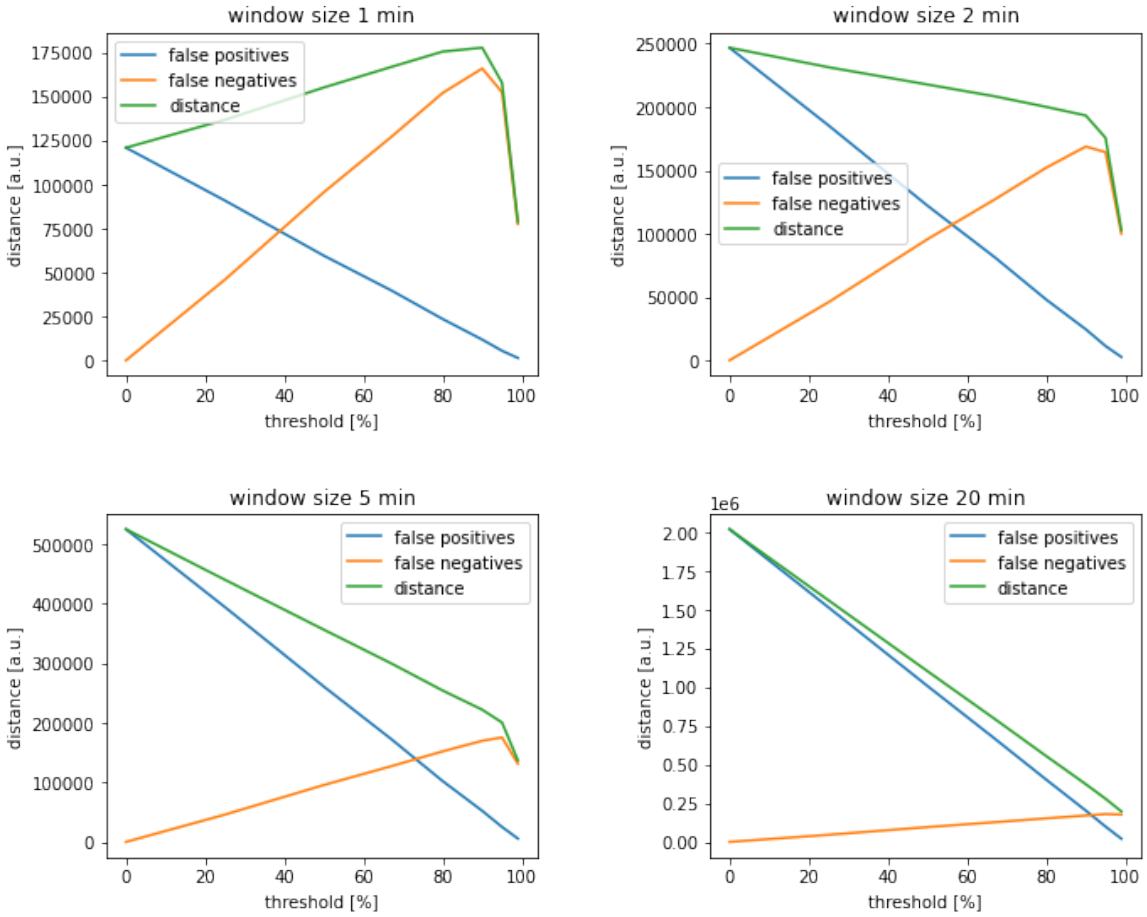

Figure 6.6: Stochastic filtering baseline for FPs, FNs and distance (Equation 3.4) between original network and filtered time-window aggregated network in the configuration $w_f = 1\,\mathrm{h}$, $w_a < w_f$. The fixed window size is indicated above each plot; the varying threshold value in percentage is on the horizontal axis. Note that percentage values do *not* correspond exactly to the actual number of links passing the filter, as explained in the caption of Figure 4.1

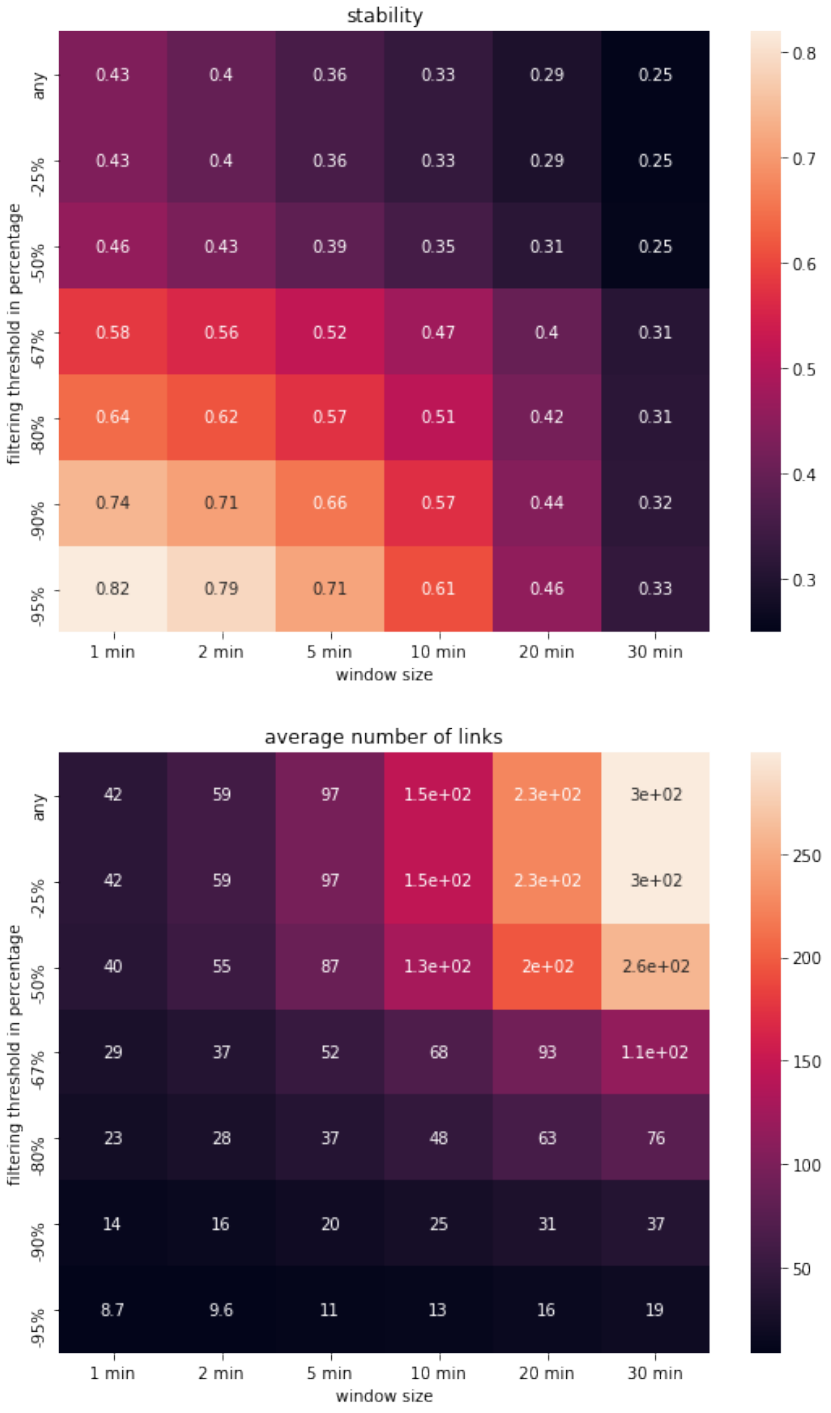

Figure 6.7: Filtering with the weight defined by Equation 3.6 or Equation 3.7 for the case $w_f = 1\,\mathrm{h}$, $w_a < w_f$. *Above*: heat map with stability score (inside boxes) against window size (horizontal axis) and filtering threshold in percentage (vertical axis, $-$N% means that N% of the links are removed). *Below*: heat map with the average number of links (not the NOC, which is higher) within each snapshot. Values do *not* correspond exactly to what percentages on the left would give, as explained in the second paragraph of Section 4.1.2

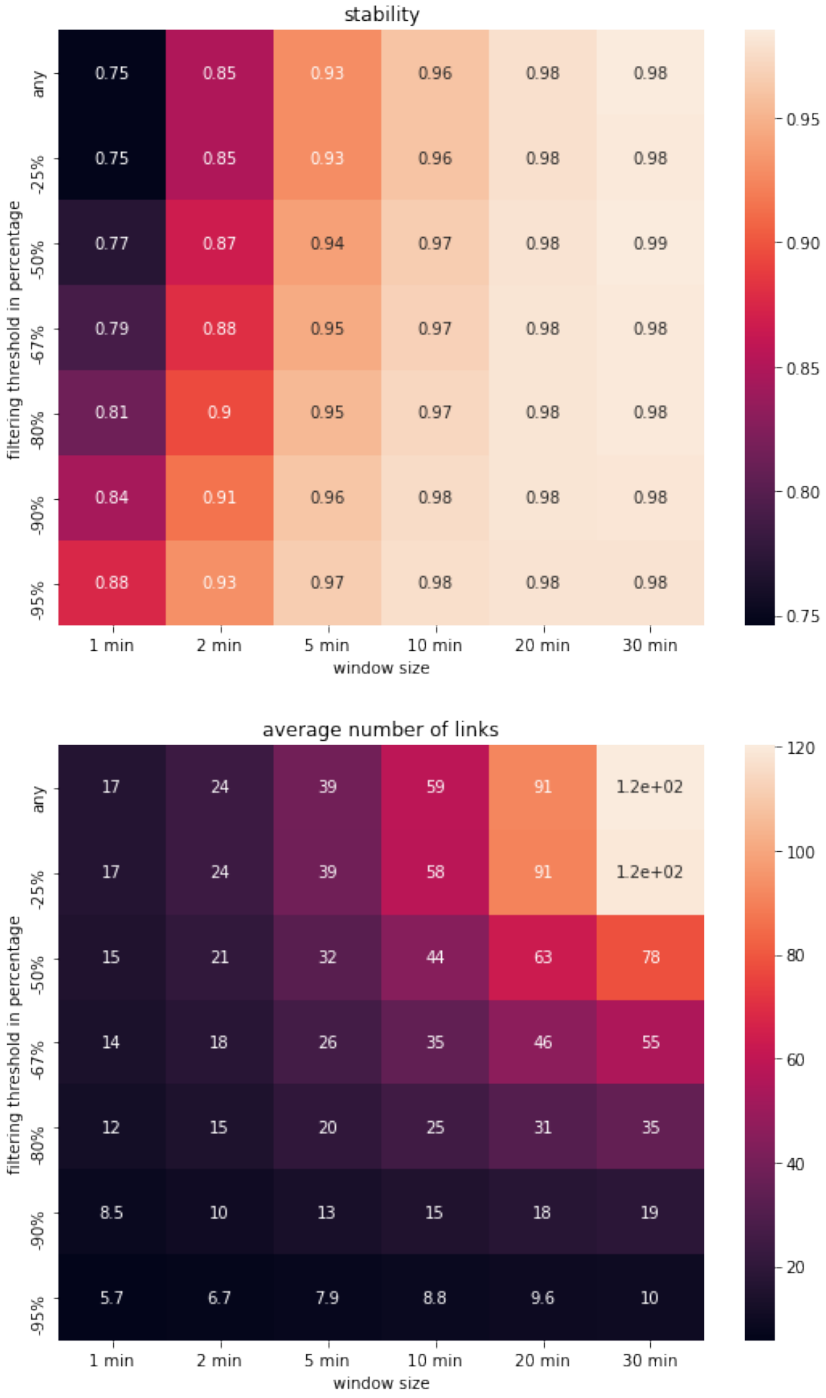

Figure 6.8: Filtering with the absolute number of contacts (NOC) for the case $w_f > w_a$ with $w_f = 1\,\text{h}$, with overlapping windows. *Above*: heat map with stability score (inside boxes) against window size (horizontal axis) and filtering threshold in percentage (vertical axis, $-$N% means that N% of the links are removed). *Below*: heat map with the average number of links (not the NOC, which is higher) within each snapshot. Values do *not* correspond exactly to what percentages on the left would give, as explained in the second paragraph of Section 4.1.2.

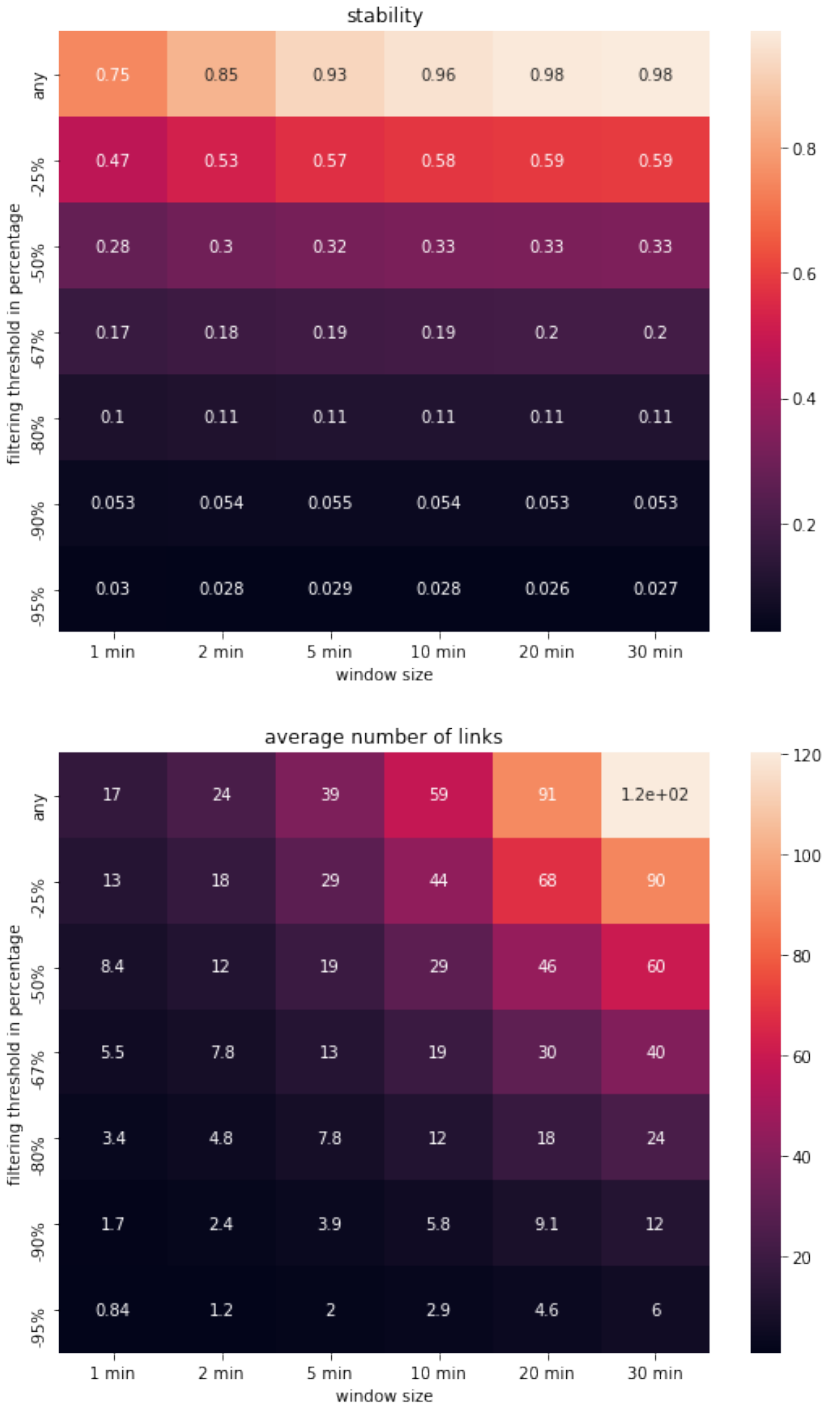

Figure 6.9: Stochastic filtering baseline for the case $w_f > w_a$ with $w_f = 1\,\text{h}$, with overlapping windows. *Above*: heat map with stability score (inside boxes) against window size (horizontal axis) and filtering threshold in percentage (vertical axis, $-$N% means that N% of the links are removed). *Below*: heat map with the average number of links (not the NOC, which is higher) within each snapshot.