HAL open science

Predicting interactions between individuals with structural and dynamical information

Thibaud Arnoux, Lionel Tabourier, Matthieu Latapy

To cite this version:

Thibaud Arnoux, Lionel Tabourier, Matthieu Latapy. Predicting interactions between individuals with structural and dynamical information. Journal of Interdisciplinary Methodologies and Issues in Science, 2019, Graph and network analysis, Analysis of networks and graphs, pp.3. 10.18713/JIMIS-150719-5-3 . hal-02191126

HAL Id: hal-02191126

https://hal.science/hal-02191126v1

Submitted on 23 Jul 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

JOURNAL OF INTERDISCIPLINARY METHODOLOGIES AND ISSUES IN SCIENCE

Predicting interactions between individuals with structural and dynamical information

Thibaud ARNOUX[1], **Lionel TABOURIER**[*1], **Matthieu LATAPY**[1]

[1]Sorbonne Université, CNRS, Laboratoire d'Informatique de Paris 6, LIP6, F-75005, Paris, France

*Corresponding author: lionel.tabourier@lip6.fr

DOI: 10.18713/JIMIS-150719-5-3
Submitted: Mars 19, 2019 - Published: July 5, 2019
Volume: **5** - Year: **2019**
Issue: **Graph and network analysis**
Editor: *Vincent Labatut*

Abstract

Capturing both structural and temporal features of interactions is crucial in many real-world situations like studies of contact between individuals. Using the link stream formalism to model data, we address here the activity prediction problem: we predict the number of links that will occur during a given time period between each pair of nodes. To do this, we take benefit from the temporal and structural information captured by link streams. We design and implement a modular supervised learning method to make prediction, and we study the key elements influencing its performances. We then introduce classes of node pairs, which improves prediction quality and increases diversity.

Keywords

Link stream; Activity prediction; Link prediction; Real-world networks

I INTRODUCTION

The study of sequences of interactions over time is an important research field with numerous application areas, such as security, mobility or recommendation systems. They have been studied as graphs for a long time, but the importance of dynamics has become more and more obvious (Holme and Saramäki, 2012). Therefore, other representations have been developed to better model the temporal nature of these systems. We use here the link stream representation because it captures both the structure and the dynamics of interactions (Latapy *et al.*, 2018).

A link stream (see Figure 1) represents a set E of triplets (t, uv) indicating that an interaction occurred between u and v at time t. Many real-world datasets may be studied using link streams, such as e-mail exchanges, contacts between individuals, phone calls or IP traffic (Latapy *et al.*, 2018; Viard and Latapy, 2014; Viard *et al.*, 2018). They are similar to temporal networks

©JIMIS, Creative Commons

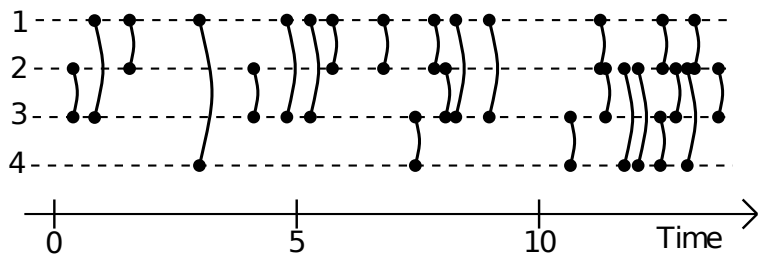

Figure 1: An example of link stream, where nodes 2 and 3 interact at time 0.5, nodes 1 and 3 at time 1, and so on.

(Holme and Saramäki, 2012) or time varying graphs (Casteigts *et al.*, 2012), which encode the same information.

We study here the activity prediction problem: we predict the number of interactions that will occur between each pair of nodes during a given period of time. This problem is related to the classical problem of link prediction in graphs, but it aims at predicting not only who will interact with whom, but also when.

Our goal here is not to obtain the best prediction, with advanced methods and on specific datasets. Instead, we propose a first framework addressing the activity prediction problem and opening the way to further investigation. We design and implement our framework in a modular way: each step of the prediction (considered features, their combination method, the optimization scheme, etc) may easily be changed. We illustrate this by making several simple choices for each building block, by applying our framework to several real-world datasets, and by studying obtained performances, thus gaining insight on important features for activity prediction. With this framework, it is easy to study other datasets and other building block choices.

More precisely, we first capture independently some structural and dynamical features of link streams. We then use a basic supervised learning algorithm to combine these metrics in order to estimate future activity. We finally compare the obtained prediction to ground truth in order to assess the relevance of our approach. This shows that combining different types of metrics, structural and temporal, indeed improves prediction. Going futher, we observe that prediction is biased towards a specific kind of links. We therefore introduce classes of node pairs with different behaviors, which improves prediction further and, more interestingly, increases its diversity. We illustrate the use and performances of our framework on four real-world traces of interactions between individuals (Mastrandrea *et al.*, 2015; Scott *et al.*, 2009; Eagle and Pentland, 2005; Bracciale *et al.*, 2014). We also investigate how different metrics can be used to selectively predict different classes of links and how the introduction of classes allows to preserve diversity while still improving prediction.

II RELATED WORK

Activity prediction is at the crossroad of two classical problems, namely link prediction in graphs and time-series prediction, illustrated in Figure 2. We briefly present both topics and give more details about works related to our approach. Then, we review works using categories in order to improve prediction, since we use this technique in our work.

2.1 Link prediction in graphs

In the classical link prediction problem, data is represented as a graph (Liben-Nowell and Kleinberg, 2007; Huang *et al.*, 2005; Wang *et al.*, 2015), and the task consists in predicting links to

©JIMIS, Creative Commons
Volume: 5 - Year: 2019, DOI: 10.18713/JIMIS-150719-5-3

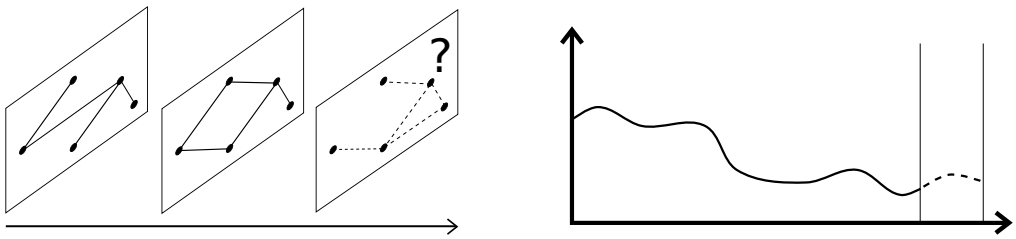

Figure 2: Basic principle of prediction in dynamic graphs (left) and in time series (right)

appear in this graph. One uses the graph structure for prediction; for instance, if two nodes have many neighbors in common then a link between them is likely to appear (Kossinets, 2006).

When temporal evolution is a key element of data, a usual approach is to slice time into several windows T_i, and consider a graph sequence $G_i = (V, E_i)$ with $E_i = \{uv : \exists(t, uv) \in E,\ t \in T_i\}$. It allows to use traditional link prediction methods on these graphs. The information contained in data is then extracted using graph-based measurements. In this field, many metrics have been developed to obtain the most relevant information (Al Hasan *et al.*, 2006; Lü and Zhou, 2011). They often consist in evaluating the similarity between two nodes according to various criteria, which produces a score or a ranking correlated to the apparition probability of a link between these nodes. Similarity measures based on the temporal patterns of activities of nodes and links have also been proposed, *e.g.* (Tabourier *et al.*, 2016).

Several methods exist to combine the metrics computed for improving prediction. It is possible to use classification algorithms to determine the predicted links (Al Hasan *et al.*, 2006; Davis *et al.*, 2013). Another approach is to rank node pairs using the values of different metrics. The predicted links connect the n first node pairs, with n fixed as a parameter and determined using the system behavior (Pujari and Kanawati, 2012; Lichtenwalter *et al.*, 2010). However, the use of time windows commands a time scale and leads to the loss of some temporal information. For example, the information associated to a link repetition between two nodes within a time window disappears. One of the stakes of our work is to conserve this information by using the link stream formalism, better suited to the data.

Another approach to link prediction using time varying graphs is to aggregate the temporal information in the system by attributing weights to links, based on previous interactions (Murata and Moriyasu, 2007; Tabourier *et al.*, 2019; Dunlavy *et al.*, 2011). This process allows to gather structural information as well as indirectly using temporal information in the data.

2.2 Time series to predict repeated interactions

It is also possible to approach link prediction by focusing on the dynamics of interactions between two nodes rather than on the structural properties (da Silva Soares and Prudêncio, 2012). The sequence of links between each pair of nodes is then considered as a time series and numerous tools have been developed to predict the future behavior of such series. For example it is possible to focus on the link apparition frequency in the past to predict future interactions (Tylenda *et al.*, 2009). This approach focuses on predicting future occurrences of links that have appeared in the past. It is also its main limitation in regards to activity prediction, as it is not suited to predict new links appearing in the system. As such, it is complementary to link prediction in graphs.

©JIMIS, Creative Commons
Volume: **5** - Year: **2019**, *DOI: 10.18713/JIMIS-150719-5-3*

2.3 Mixed approaches

Some studies use both link prediction in graphs and a time series approaches (Huang and Lin, 2009). However they mostly rely on time windows, which leads to important information losses and issues regarding time window choices, as already explained.

2.4 Link categories

To improve prediction quality in a supervised learning problem, it is often efficient to divide items into categories (Lichtenwalter *et al.*, 2010; Brockwell and Davis, 2013). In the case of link prediction, node pairs with similar properties can be gathered in order to model their behavior more accurately.

For example it is possible to group node pairs by communities: pairs inside a community are more likely to have similar behaviors than pairs between communities (Clauset *et al.*, 2008). Other works use node or link attributes to build categories of pairs, with the assumption that items with similar attributes have similar behaviors (Scholz *et al.*, 2012).

III PROBLEM DEFINITION AND PREDICTION FRAMEWORK

In this section, we first present the prediction problem that we address in this paper. We then propose a modular framework for solving it. We describe each of its building blocks as well as the assumptions on which they rely. Figure 5 gives a global view of this framework.

3.1 The activity prediction problem

We consider a set of nodes V representing entities in the system. We observe interactions between these entities for a period of time $T = [A, \Omega]$, that we model as a link stream (Latapy *et al.*, 2018) $L = (T, V, E)$, where $E \subseteq T \times V \otimes V$, and $(t, uv) \in E$ means that an interaction occurred between u and v at time t. In this work, $uv \in V \otimes V$ denotes the unordered pair of nodes $u \in V$ and $v \in V$: $uv = vu$ and $u \neq v$. We call L and T the *input stream* and *input period*, respectively. We represent future interactions as a link stream $L' = (T', V, E')$ with $E' \subseteq T' \times V \otimes V$ and $T' = [A', \Omega']$ with $\Omega \leq A' < \Omega'$. We call L' and T' the *prediction stream* and *prediction period*.

Our goal is to predict the number of interactions between each node pair uv in $V \otimes V$, which we call the *activity* of uv. We denote by $\mathcal{A}(uv) = |\{(t, uv) \in E\}|$ the activity of uv during the input period, by $\mathcal{A}^p(uv)$ the predicted activity of uv during the prediction period (computed by our algorithm below), and by $\mathcal{A}^a(uv) = |\{(t, uv) \in E'\}|$ the actual activity during the prediction period. Prediction quality is measured by comparing $\mathcal{A}^p(uv)$ to $\mathcal{A}^a(uv)$, as detailed in Section 3.5.

3.2 Prediction metrics

Our approach relies on metrics that capture various features, which may *a priori* be important for activity prediction and that we describe here. Indeed, information contained in a link stream is of different kinds, like for instance, the number of past interactions between two nodes or the density of a node neighborhood. In general, existing methods focus either on structural features (in the case of link prediction in graphs) or on temporal features (for time-series prediction), while we intend to use metrics adapted to link streams, which combine temporal and structural information. The metrics presented in this section and their associated acronyms are summarized in Table 2.

3.2.1 *Structural metrics*

As a first step, we adapt structural metrics from link prediction in graphs to the context of activity prediction in link streams. Considering that the neighborhood of a node u is defined as $\mathcal{N}(u) = \{v : \exists(t, uv) \in E\}$, we can define traditional metrics widely employed in link prediction: the *Common Neighbors index (CN)* (Zhou *et al.*, 2009), the *Jaccard index (JI)* (Jaccard, 1901), the *Sørensen index (SI)* (Sorensen, 1948), the *Adamic-Adar index (AA)* (Adamic and Adar, 2003) and the *Resource Allocation index (RA)* (Zhou *et al.*, 2009). We do not recall the definitions of these classical metrics here. Notice that all these metrics are independent of time.

3.2.2 *Temporal metrics*

To contrast with graphs, the link stream formalism also captures temporal information. As it is usually done in the field of time-series prediction, we first use as a benchmark the extrapolation of past activity, which captures the frequency of an interaction in the input stream. To keep it as simple as possible, we use the number of interactions between u and v occurring during T. In the following we refer to this metric as the Pair Activity Extrapolation defined as $PAE(uv) = |\{(t, uv) \in E\}|$ (which is actually identical to $\mathcal{A}(uv)$).

Then, we define two other metrics that describe more precisely the temporal behavior of the system. They are adapted versions of the pair activity extrapolation that focus on the most recent activity during the input period. This choice is made on the ground that recent interactions affect more the dynamics than the older do (Dunlavy *et al.*, 2011).

First, we only take into account the activity during a recent period of time of duration δ: for each pair of nodes, we compute the $PAE\delta S(uv) = |\{(t, uv) \in E \ : \ t \in [\Omega - \delta, \Omega]\}|$. Second, we take into account the activity of each pair of nodes between Ω and the occurrence time of the kth link between them before Ω. The corresponding index is $PAEkL(uv) = k/(\Omega - t_k)$ with t_k such that $|\{(t, uv) \in L, \Omega \geq t \geq t_k\}| = k$.

3.2.3 *Hybrid metrics*

We also use hybrid metrics, which capture a mixture of structural and aggregated temporal information. We use weighted variation of link prediction metrics as proposed by Tabourier *et al.* (2019).

- The *Weighted Common Neighbors (WCN)* metric emphasizes the common neighbors which have often interacted with both nodes:

$$WCN(uv) = \sum_{w \in \mathcal{N}(u) \cap \mathcal{N}(v)} \mathcal{A}(uw) \cdot \mathcal{A}(vw). \tag{1}$$

- The *Weighted Sørensen index (WSI)* is similar to the Sørensen index but takes into account the activity related to each node:

$$WSI(uv) = \frac{\sum_{w \in \mathcal{N}(u) \cap \mathcal{N}(v)} \mathcal{A}(uw) + \mathcal{A}(vw)}{\sum_{k \in \mathcal{N}(u)} \mathcal{A}(uk) + \sum_{k \in \mathcal{N}(v)} \mathcal{A}(vk)}. \tag{2}$$

©JIMIS, Creative Commons
Volume: **5** - Year: **2019**, *DOI: 10.18713/JIMIS-150719-5-3*

- The *Weighted Adamic-Adar (WAA)* decreases the weight of shared neighbors with high degree and high level of activity:

$$WAA(uv) = \sum_{w \in \mathcal{N}(u) \cap \mathcal{N}(v)} \frac{1}{\sum_{k \in \mathcal{N}(w)} \log(\mathcal{A}(wk))}. \tag{3}$$

- The *Weighted Resource Allocation (WRA)* is similar to the Weighted Adamic-Adar but gives weights differently:

$$WRA(uv) = \sum_{w \in \mathcal{N}(u) \cap \mathcal{N}(v)} \frac{1}{\sum_{k \in \mathcal{N}(w)} \mathcal{A}(wk)}. \tag{4}$$

3.2.4 Normalization

Each metric is then normalized with the maximum value of the metric index. This allows for comparable metric index values and better understanding of each metric weight in the prediction. This normalization process is done for each pair of nodes and each metric in order to keep the absolute values comparable between different metrics. We obtain the normalized index $\hat{m}(uv)$:

$$\hat{m}(uv) = \frac{m(uv)}{max_{uv \in V \otimes V}\{m(uv)\}}, \tag{5}$$

where $m(uv)$ is the index associated to metric m over the pair of nodes uv.

3.3 Prediction index

In order to use the features captured by metrics presented above, we combine the information contained in the normalized metric indexes into a prediction index. Precisely, we build a prediction index $\mathcal{F}$, such that for all $uv \in V \otimes V$, $\mathcal{F}(uv)$ represents how likely a link between u and v during $[A', \Omega']$ is. Here, we build it using a linear combination for simplicity reasons, but a user could define another combination method, based on his or her choice of metrics. Formally,

$$\mathcal{F}(uv) = \sum_{m \in \mathcal{M}} \alpha_m \cdot \hat{m}(uv), \tag{6}$$

where $\mathcal{M}$ is the set of metrics used and $\hat{m}$ is the normalized metric index associated to the metric m. The parameters α_m control each metric weight in the prediction index. We make the strong assumption that the metrics $\hat{m}$ are chosen in such a way that $\mathcal{F}(uv)$ is proportional to the probability for an interaction between u and v to happen. Note that the important information does not lie in the absolute value of $\mathcal{F}(uv)$, but rather in the relative values of $\mathcal{F}(uv)$ in comparison to other $\mathcal{F}(u'v')$ as we will see in Section 3.4.

Given such a prediction index, a standard method consists in learning on the training period values of α_m which optimize a given evaluation criterion. Then, these weights are used for actual prediction on the prediction period. Our method to optimize weights is presented in Section 3.6.

3.4 Global prediction and link allocation

Now, we describe the core of our approach to activity prediction in link streams. It relies on the following design choice: first, we predict the overall activity on the stream, then we allocate a fraction of all predicted links to each pair of nodes depending on its prediction index. The overall activity can be predicted using methods usually available to predict time series. How to allocate links to node pairs is a broad question in itself, which we cannot address in details here, therefore we will make a basic choice to give the reader an idea of how to deal with this issue in practice.

Precisely, to predict the number of interactions between each pair of nodes during $T' = [A', \Omega']$, we first estimate the global number of links N between all node pairs during this period. We make the simplistic assumption that the global activity in L' is the same as in L, and therefore, extrapolate linearly the stream activity to determine the global number N of links to predict:

$$N = |E| \cdot \frac{\Omega' - A'}{\Omega - A}. \tag{7}$$

Then, for each pair of nodes, the metric indexes m and subsequent normalized metric indexes $\hat{m}$ are computed on the link stream L. As the prediction index reflects how likely the occurrence of a link between two nodes is, we use it to distribute the N links between all pairs in $V \otimes V$. Precisely, supposing that the prediction index $\mathcal{F}$ is build in such a way that it is proportional to the number of links appearing for any pair uv, we allocate the N links estimated previously according to the following allocation rule:

$$\mathcal{A}^p(uv) = N \cdot \frac{\mathcal{F}(uv)}{\sum_{xy \in V \otimes V} \mathcal{F}(xy)}. \tag{8}$$

Of course, we have $\sum_{xy \in V \otimes V} \mathcal{A}^p(xy) = N$. As mentioned previously, it is thus the relative values of $\mathcal{F}(uv)$ which are important in the computation of $\mathcal{A}^p(uv)$.

This framework allows to predict the future activity of node pairs, that is the number of links appearing between each pair of nodes during T'. It is important to note that due to the specificities of our prediction task and in contrast to what is usually done for link prediction in graphs, this number is not necessarily an integer.

3.5 Evaluation score and protocol

To evaluate the efficiency of our protocol, we need to define a quality estimator specific to the activity prediction problem. Practically, we have to compare for all $uv \in V \otimes V$ the number of predicted links $\mathcal{A}^p(uv)$ to the number of links that have actually occurred $\mathcal{A}^a(uv)$. Note in particular that our method of computing $\mathcal{A}^p(uv)$ does not guarantee that it is an integer, so that the estimators should be suited to this. Nevertheless, we would like to make the evaluation method as close as possible to the tools used in classification tasks, as it would make the framework closer to the familiar vocabulary of link prediction.

Thus, we adjust the usual definition of true positives, false positives and false negatives to the context of activity prediction in link streams, as these measurements underly many evaluation metrics in prediction tasks. Precisely, for each pair uv, we compare $\mathcal{A}^p(uv)$ to $\mathcal{A}^a(uv)$, see Figure 3. We then define the number of *TP*, *FP* and *FN* as follows:

$$\begin{cases} TP(uv) = \min(\mathcal{A}^p(uv), \mathcal{A}^a(uv)) \\ FP(uv) = \max(\mathcal{A}^p(uv) - \mathcal{A}^a(uv), 0) \\ FN(uv) = \max(\mathcal{A}^a(uv) - \mathcal{A}^p(uv), 0) \end{cases}. \tag{9}$$

©JIMIS, Creative Commons
Volume: 5 - Year: 2019, DOI: 10.18713/JIMIS-150719-5-3

The sum of each of these indicators over all node pairs yields the number of TP, FP and FN for all predictions. Note that these definitions allow to get the usual relationships between indicators. In particular, $TP + FP$ is the number of predictions and $TP + FN$ is the total number of interactions actually occurring during T'. Moreover, they convey the same idea as the usual *TP*, *FP* and *FN* do, as *TP* and *FP* reflect respectively the number of good and false predictions among all, while *FN* corresponds to actual occurrences of interactions which have not been predicted.

Consequently, we can compute more sophisticated performance indicators, like the Precision $\frac{TP}{TP+FP}$ and the Recall $\frac{TP}{TP+FN}$. We also use the F-score to quantify the quality of prediction, which is the harmonic mean of these two indicators: $2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$. Other indicators could be defined in this context, like the ROC curve, but we do not use them in this study.

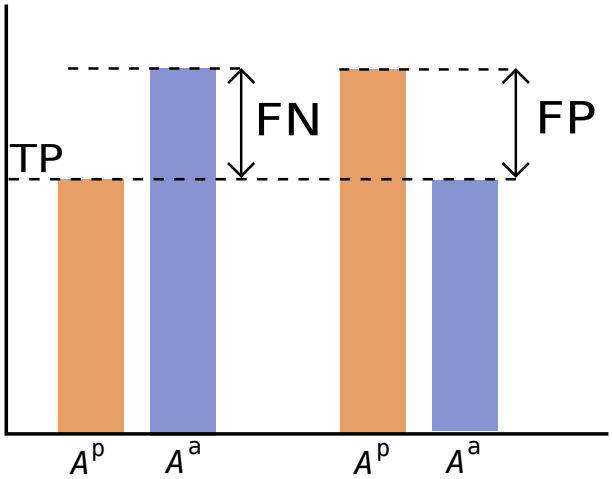

Figure 3: Examples of evaluation scores for activity prediction in link streams. Left: prediction underestimates the actual number of occurring links. Right: prediction overestimates it.

3.6 Learning of parameters

The choice of values for parameters α_m is a crucial step to achieve accurate prediction, as they determine the balance between combined metrics. An automated way to choose these values is also needed when using multiple metrics simultaneously, as systematically explore the parameter space would be too costly. For this purpose we use a machine learning algorithm and define training and testing phases.

We use a hold-out technique, which means that we divide the data into sub-streams in order to define a training phase during which we optimize parameters and then extrapolate these parameters for the purpose of the prediction itself. Note that k-fold validation is much used in prediction tasks, however it is difficult to implement in a context where the time-ordering plays an essential role, as using randomly chosen periods of time would lead to an important loss of temporal structure.

Practically, we proceed in the following way, see Figure 4: we divide the input stream L into two sub-streams: a *training sub-stream* $L_1 = (T_1, V, E_1)$ with $T_1 = [A_1, \Omega_1]$, used to compute the metrics during the training phase, and a *validation sub-stream* $L_2 = (T_2, V, E_2)$ with $T_2 = [A_2, \Omega_2]$. The values of parameters α_m are then computed using a learning algorithm to optimize activity prediction on T_2 using the information contained in T_1. Prediction metrics are then

©JIMIS, Creative Commons
Volume: 5 - Year: 2019, DOI: 10.18713/JIMIS-150719-5-3

computed on the actual observation sub-stream $L_2 = (T_2, V, E_2)$ with $T_2 = [A_2, \Omega_2]$, to predict the activity during the prediction stream $L' = (T', V, E')$ with $T' = [A', \Omega']$.

In our implementation, the validation and the effective observation sub-streams are identical. This choice is not mandatory, but we make it because it would probably be the operator choice in many real-time applications, as it means that the latest observations are used in order to predict coming events.

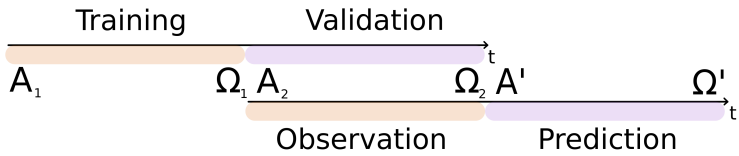

Figure 4: Illustration of the different periods used in our protocol: the training and validation periods (top) used to train the algorithm and the observation and prediction periods (bottom) used to make the actual prediction.

The initial set of parameters given as input to the learning algorithm is drawn randomly in the parameter space for each prediction. Then, we use a gradient descent algorithm to explore the parameter space and find good values for each parameter α_m. Note that using a linear combination of metric indexes allows to avoid repeating a number of heavy computational steps. This reduces greatly the computational cost of the method.

3.7 Method summary

We represent the different steps of our approach in Figure 5 and we summarize notations in Table 1. This framework is able to combine different kinds of information to optimize prediction. We did our best to make it very modular, in order to make it easy to adapt it to specificities of a given activity prediction problem, or to specific dataset features. For example one may adapt the metrics chosen, the combination method, the optimization technique, or the quality estimator used during the evaluation phase. Our implementation is available online[1].

Notation	Meaning
A_i	Start time of link stream L_i
Ω_i	End time of link stream L_i
N	Overall number of predicted links
N'	Overall number of occurring links
$\mathcal{A}^p(uv)$	Activity predicted for pair uv
$\mathcal{A}^a(uv)$	Actual activity for pair uv
$\mathcal{F}$	Prediction index $\mathcal{F}(uv) = \sum_{m \in \mathcal{M}} \alpha_m \cdot \hat{m}(uv)$
	$\hat{m}$ is a normalized metric and α_m the weight of metric $\hat{m}$ in $\mathcal{F}$

Table 1: Summary of notations, where $L_i = (T_i, V_i, E_i)$ is a link stream with $E_i \subseteq T_i \times V_i \otimes V_i$, $T_i = [A_i, \Omega_i]$ is a time interval, and $(t, uv) \in E_i$ means that an interaction occurred between u and v at time t.

[1] https://github.com/ThibaudA/linkstreamprediction

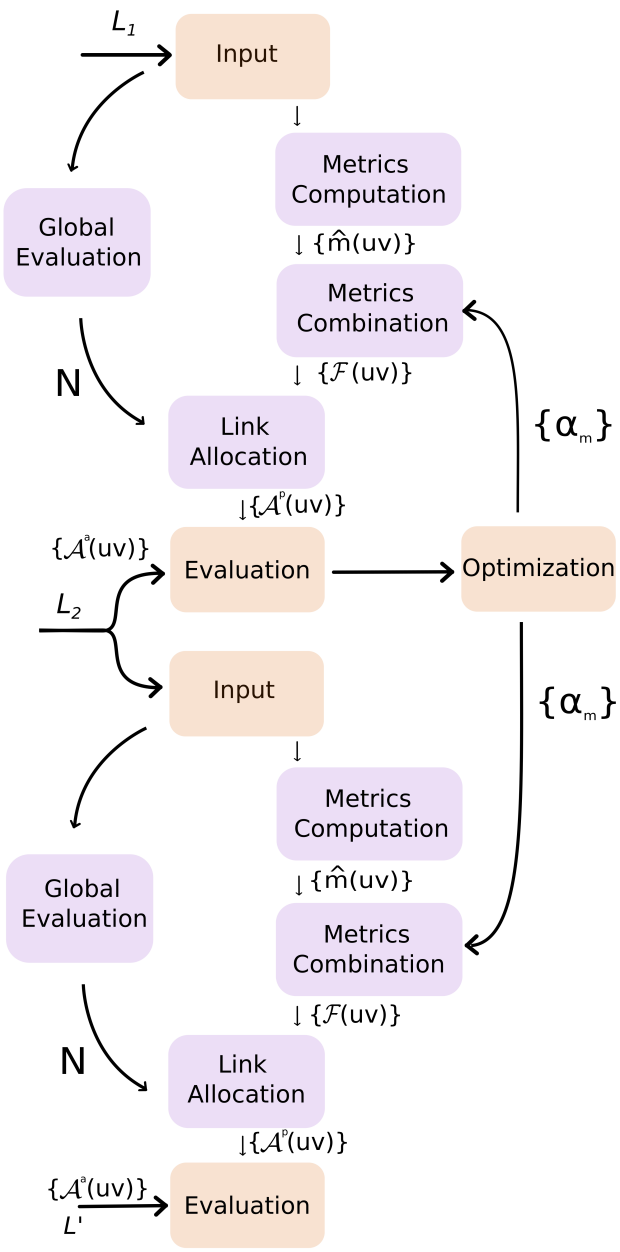

Figure 5: Summary of the prediction protocol. The learning and evaluation phases are described in agreement with the implementation depicted in Figure 4

.

IV DATASETS AND EXPERIMENTS

In this section, we study the results of our activity prediction method on four different datasets. First, we briefly describe these datasets. Then, we implement the standard prediction protocol aforementioned and investigate more closely how the learning method distributes weights among combined metrics. Finally, we identify the strengths and weaknesses of the procedure and important challenges to improve the activity prediction.

4.1 Data description

In order to assess the performances of our framework, we conducted experiments on four datasets which gather contacts detected with sensors. Each undirected link (t, uv) means that

©JIMIS, Creative Commons
Volume: 5 - Year: 2019, DOI: 10.18713/JIMIS-150719-5-3

the sensor carried by nodes u or v detected the sensor carried by the other nodes at time t, which means in turn that these two nodes were close enough at time t for the detection to happen. We call this a contact between nodes u and v. We present the datasets briefly in this section.

The first trace was collected in a French high school in 2012 (*Highschool* dataset), see (Mastrandrea *et al.*, 2015) for full details. It is a link stream of 181 nodes and 45047 links, connecting 2220 distinct pairs of nodes over a period of approximately 8 days.

The second dataset has been collected during IEEE INFOCOM 2006 in Barcelona (*Infocom* dataset) – see (Scott *et al.*, 2009). The Bluetooth devices used in this experiment recorded connections with one another. This dataset contains 98 nodes and 283,100 links. During this 3 days long experiment, 4,338 pairs of nodes have interacted. Note that the *Infocom* dataset, which is also a contact sensor trace, involves less nodes but contains more links and more active node pairs than the *Highschool* dataset does.

Our third dataset is the *Reality Mining* dataset, which gather contacts among students of the Massachusetts Institute of Technology as recorded by their mobile phones – see (Eagle and Pentland, 2005). The trace contains 1,063,063 links between 96 nodes during 9 months with 2,539 distinct pairs of nodes involved in interactions.

Finally, we study the *Taxi* dataset. It is based on the GPS locations of 305 taxis in Rome, recorded during the month of February 2014 – see (Bracciale *et al.*, 2014). We consider that two taxis have interacted if they are within a 30 meters range from each other. This yields a link stream of 22,364,061 links. During the experiment 16,799 pairs of nodes have interacted, which makes this dataset significantly larger than the other datasets under study.

4.2 Experimental implementation

As a reminder of the protocol described in Section 3.6, we divide our input period L in two substreams, the training stream $L_1 = (T_1, V, E_1)$ with $T_1 = [A_1, \Omega_1]$, and the validation/observation stream $L_2 = (T_2, V, E_2)$ with $T_2 = [A_2, \Omega_2]$. Values of parameters α_m are then computed using a gradient descent algorithm that optimizes activity prediction of the activity on L_2 using the information contained in L_1. The prediction metrics are then computed on the observation sub-stream $L_2 = (T_2, V, E_2)$ with $T_2 = [A_2, \Omega_2]$, to predict the activity in prediction stream $L' = (T', V, E')$ with $T' = [A', \Omega']$. The acronyms corresponding to these metrics are indicated in Table 2.

In our experimental set-up the durations of training, validation, observation and prediction periods are chosen equal. In the cases of *Highschool* and *Infocom* datasets, we use 1, 2 and 3 hours long periods starting on a Monday at 8:30 and 9:00 respectively. In the cases of *Reality Mining* and *Taxi* datasets, the timescales are typically longer, so we use 1, 2 and 3 days long periods, starting on a Tuesday at 1:30 am for the *Reality Mining* dataset and a Wednesday at 8:00 am for the *Taxi* dataset. We report in Table 3 the exact starting and ending times for *Highschool* and *Infocom* experiments. The choice of these periods have important consequences on the prediction quality, as we shall see in the next section.

Acronym	Metric
CN	Number of Common Neighbors
AA	Adamic-Adar Index
RA	Resource Allocation Index
SI	Sørensen Index
JI	Jaccard Index
WCN	Weighted Number of Common Neighbors
WAA	Weighted Adamic-Adar
WRA	Weighted Resource Allocation Index
WSI	Weighted Sørensen Index
PAE	Pair Activity Extrapolation
PAE10L	Activity by unit of time during the last 10 links
PAE100S	Activity during the last 100 seconds
PAE1000S	Activity during the last 1,000 seconds
PAE10000S	Activity during the last 10,000 seconds

Table 2: Metric Acronyms

Dataset	Duration	A_1	$\Omega_1 = A_2$	$\Omega_2 = A'$	Ω'
Highschool	1h	8:30 am	9:30 am	10:30 am	11:30 am
	2h	8:30 am	10:30 am	12:30 am	2:30 pm
	3h	8:30 am	11:30 am	2:30 pm	5:30 pm
Infocom	1h	9:00 am	10:00 am	11:00 am	12:00 am
	2h	9:00 am	11:00 am	1:00 pm	3:00 pm
	3h	9:00 am	12:00 am	3:00 pm	6:00 pm
Reality Mining	1d	Tuesday	Wednesday	Thursday	Friday
	2d	Tuesday	Thursday	Saturday	Monday
	3d	Tuesday	Friday	Monday	Thursday
Taxi	1d	Wednesday	Thursday	Friday	Saturday
	2d	Wednesday	Friday	Sunday	Tuesday
	3d	Wednesday	Saturday	Tuesday	Friday

Table 3: Starting and ending time of each periods for each dataset

4.3 Standard prediction

Results are summarized in Table 4. We report the F-score, the Recall and the Precision for each experiment, as well as the number of predicted links (N) and the number of links which actually occurred during the prediction period (N'). The values presented are the averages obtained on 10 realizations for each set of parameters. The standard deviation for F-score, Precision and Recall is in all cases lower than 0.01.

4.3.1 Highschool

We first focus on the *Highschool* dataset experiments. We can see that during the 1h experiment, the algorithm predicts slightly more links than what actually appear. Still, Recall is 0.69, meaning that 69% of interactions observed have been correctly predicted, therefore the method seems to quite precisely account for a large part of the activity in the dataset during this period.

However, a significant drop in the prediction quality can be observed during the 2h experiment, which is visible on both Precision and Recall (and consequently on the F-score). This is prob-

©JIMIS, Creative Commons
Volume: 5 - Year: 2019, DOI: 10.18713/JIMIS-150719-5-3

Dataset	Duration	F-score	Precision	Recall	N predicted	N' occurring
Highschool	1h	0.44	0.33	0.69	1,123	857
	2h	0.17	0.18	0.15	1,751	2,178
	3h	0.29	0.22	0.41	3,072	1,900
InfoCom	1h	0.59	0.58	0.59	8,167	8,220
	2h	0.55	0.50	0.60	16,737	14,051
	3h	0.67	0.63	0.70	22,568	20,850
R.Mining	1d	0.47	0.41	0.56	6,105	8,733
	2d	0.09	0.05	0.65	18,537	1,591
	3d	0.41	0.73	0.29	11,395	29,078
Taxi	1d	0.18	0.19	0.17	872,173	971,204
	2d	0.10	0.07	0.16	1,904,076	910,457
	3d	0.16	0.20	0.14	1,843,329	2,865,449

Table 4: F-score, Precision, Recall and number of links predicted and occurring.

ably due to the fact that the observation period spreads over lunch break. The overall activity predicted over the prediction link stream underestimates the number of links that actually appear. It reflects that the activity from the observation stream is not as high as the actual activity from the prediction stream. But the main reason for this drop in the prediction quality certainly stems from the fact that lunch break is an opportunity for interactions which are different from the ones occurring during class hours, and this seems to have a significant impact. We explore this question in more details in Section 4.5.

Considering the 3h experiment, we observe a slight increase in Precision and Recall compared to the 2h experiment. We think that longer observation periods mitigate the effect of behavior changes, like the one that happens at lunch break. The overall activity predicted is greater that the activity that actually occurs. While this effect impacts the prediction quality, we see that it is not as significant as with the 2h experiment.

These experiments show that predictions on longer periods are negatively affected by the strong variation in the behavior of the system over time, mostly due to the high-school schedule, which alternates breaks and class hours.

4.3.2 Infocom

For the *Infocom* dataset, performances are more stable compared to the *Highschool* case, in particular when comparing the number of predicted links to the number of occurring links. Indeed, the divergence between these values remains consistently lower than 20%. This leads to better performances on each experimentation. We suggest that, as this dataset reports contacts at a conference, it may be explained by the fact that differences in behaviors are less marked between talks and breaks. At a much smaller scale than in the *Highschool* case, we also observe a loss in prediction quality when using 2h periods compared to 1h periods, but the effect disappears on longer periods: 3h training and prediction periods yield even better results than the 1h case.

©JIMIS, Creative Commons
Volume: **5** - Year: **2019**, *DOI: 10.18713/JIMIS-150719-5-3*

4.3.3 *Reality mining*

The *Reality Mining* dataset performs relatively well on 1 day and 3 days periods, however we can see a dramatic loss in prediction quality using 2 days periods. In this case, the loss seems to come from the vast overestimate of predicted links compared to the links which appear. This poor performance prediction is investigated more thoroughly in Section 4.5.

In the case of the 3 days training and prediction periods experiment, the overall predicted activity is now lower than the occurring activity, which leads to a low Recall value (0.29) relatively to the 1 day experiment (0.56), where the underestimation of activity was not as significant. However, predicted interactions are quite accurate, as the Precision is 0.73. This tends to shows that in certain instances our protocol is able to achieve suitable prediction despite an underestimated activity during prediction period.

4.3.4 *Taxi*

Considering the *Taxi* dataset, we can see that prediction quality in terms of F-score is lower than for other datasets. This is due to a greater number of nodes and interactions in the dataset, leading to a more difficult prediction task, as there are more candidate node pairs which can be predicted. Notice also that the nature of the dataset is different and geographical proximity between two taxis may be less meaningful than proximity of students or conference attendees.

The 1 day experiment predicts an overall activity closer to the actual activity, leading to the best prediction. We can see that for 2 days and 3 days predictions, the extrapolation yields less accurate results regarding the F-score, and in each case the overall activity prediction has been largely ill-estimated (respectively over and underestimated).

These experiments show that the choice of the prediction period plays a key role for prediction quality. It is closely related to the overall activity extrapolation, leading to important variations in the prediction quality.

4.4 Metric combinations

In this section we observe how the algorithm mixes different metrics in order to optimize prediction. The combination realized for each dataset is shown in Figure 6. The height of each bar corresponds to the mean of coefficients obtained for the experiments presented above (average on 10 realizations for each set of parameters). We have represented metric combinations for the following training/prediction periods and datasets: 1 hour for *Highschool*; 2 hours for *Infocom* and 1 day for *Reality Mining* and *Taxi*. The meaning of metric acronyms used can be found in Table 2.

Considering the *Highschool* dataset, algorithm mainly uses the *Pair Activity Extrapolation* (PAE) metric with a small influence from the *activity during the last* 100 *seconds* (PAE100S). This tells us that the algorithm basically extrapolates the dynamics of each pair of nodes during the end of the observation period to predict future interactions.

Considering the *Infocom* dataset, the algorithm also focuses on temporal metrics. It allows the algorithm to gather information about dynamics related at different time scales. The algorithm also makes use of one of the weighted metric, namely the *Weighted number of Common Neighbors* (WCN).

The patterns of used metrics in the case of the *Reality Mining* is quite similar to the *Infocom* case. Indeed, the algorithm gives most weight to temporal metrics, with a marginal use of

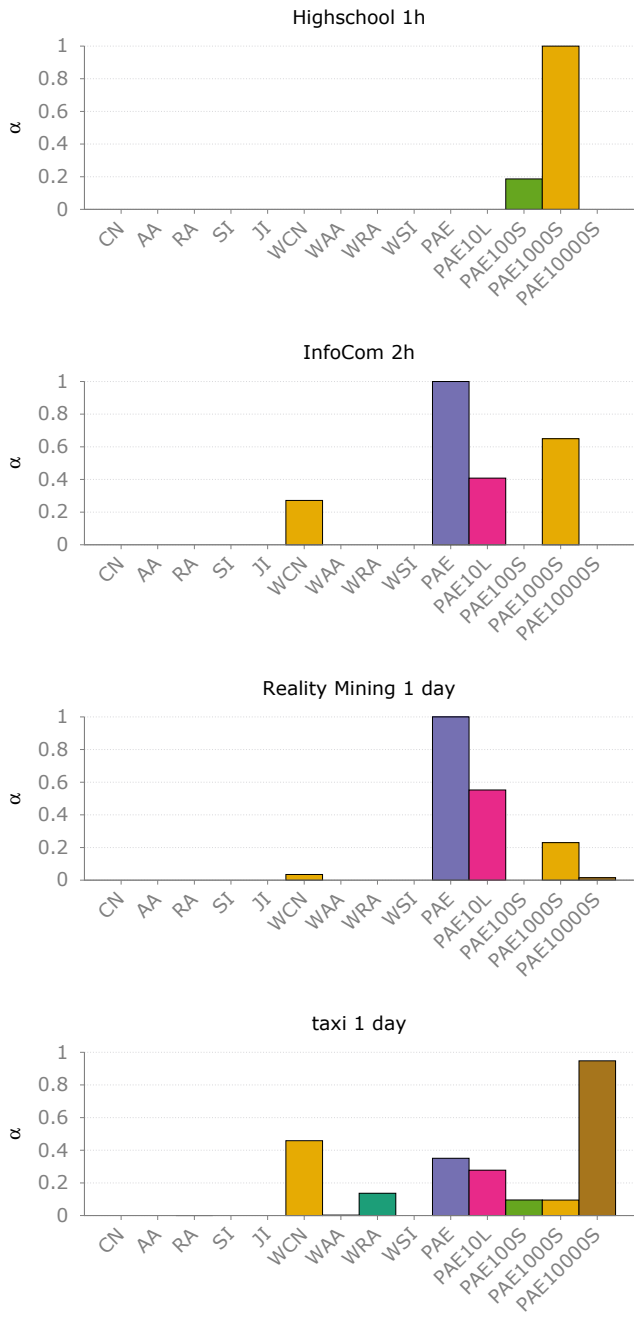

Figure 6: Metric coefficients repartition computed by the learning algorithm for the following datasets and training periods (from top to bottom): *Highschool* (1h), *InfoCom* (2h), *Reality Mining* (1d) and *Taxi* (1d).

Weighted number of Common Neighbors, more precisely the *activity during the last* 1,000 *seconds* (PAE1000S) and the *activity by unit of time during the* 10 *last links* (PAE10L). The time granularity being 120 seconds, *activity during the last* 100 *seconds* does not appear.

Finally, concerning the *Taxi* dataset, we can see that our algorithm mainly uses *activity during the last* 10,000 *seconds* (PAE10000S). However, all the other temporal metrics are also used with relatively lower weights, indicating that all timescales carry a certain amount of information in regards to this prediction. Besides that, we also see a notable contribution of structure-oriented index such as the *Weighted number of Common Neighbors* as well as a slight use of the

Weighted Resource Allocation index.

For each dataset the algorithm uses slightly different metric combinations. However, we can note that temporal metrics are often favored. Purely structural metrics are mostly ignored by our algorithm, yet hybrid metrics collect structural information, which is used by the algorithm to improve the prediction in most experimental settings. It is also interesting to note that temporal metrics corresponding to different time scales are represented, meaning that combining these metrics allows the algorithm to capture different information.

An important point is that by extrapolating the past activity of a pair of nodes, temporal metrics are simply not able to predict new interactions, while structural and hybrid ones may. This indicates that it is much harder to predict interactions that appear for the first time than previously observed interactions. Indeed, our framework combines metrics in order to focus on repeated links rather than on new links. We explore this intuition thereafter.

4.5 The case of poor predictions

In this section, we investigate in more details two cases seen in Section 4.3 which exhibit particularly low prediction quality in terms of F-score, compared to other experiments on the same datasets.

Considering the 2 days experiment on *Reality Mining*, we have observed an important drop of the overall activity during the prediction period. We show in Figure 7 the number of links through time during this experiment with a granularity of $1{,}000$ seconds. The system displays a large decrease on the number of links appearing during the last two days. The experiment starting on a Tuesday, this mean that the training period is from Tuesday to Wednesday, the validation and observation from Thursday to Friday, while the prediction period is Saturday and Sunday, which leads to a huge decrease of activity.

Concerning the *Highschool* experiment with 2 hours periods, we have previously noted an underestimation of the activity during the prediction period. Again, we show in Figure 7 the activity during this experiment with a granularity of 40 seconds. We have also seen in Table 3 that the prediction period $[A', \Omega']$ spans from 12:30 pm to 2:30 pm. This corresponds to lunch break, when students leave their classes and have the opportunity to mix with other students. It is hard to draw any conclusion from Figure 7, as the activity decreases from the training period (8:30 am to 10:30 am) to the validation period (10:30 am to 12:30 pm) and again from the validation period to the prediction period. But the predicted overall activity reported in Table 4 seems not far enough from the actual activity to explain alone the drop in prediction quality. To investigate further this problem, we achieve a prediction task, which is not realistic but allows us to have a better grasp of the reason for the former observation. We repeat the previous experiment, but setting predicted activity to actual activity measured in the prediction stream. This experiment yields a F-score of 0.17, Precision is 0.16 and Recall is 0.17. These values are close to the quality obtained during the original experiment. This shows that predicted overall activity alone does not explain the drop in quality. Thus, the prediction is mostly affected by the change of students contact behavior from classes to lunch time.

Notice that these results enlighten the importance of the choice of the training, validation and prediction periods in the protocol.

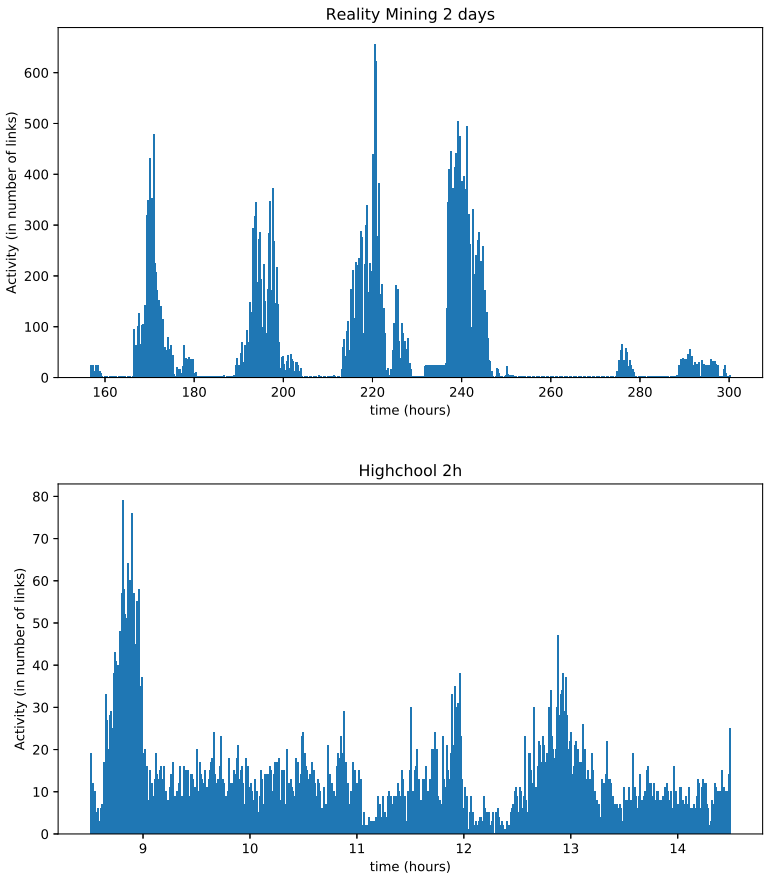

Figure 7: Number of links appearing over time during *Reality Mining* 2 days experiment (Granularity: 1000 seconds) and *Highschool* 2 hours experiment (Granularity: 40 seconds).

V INVESTIGATING COMBINED METRICS

In this section, we use a simplified learning model, identical to the one presented above except for the fact that it mixes only two different metrics. This simplification allows to clearly represent the mixing achieved by the method, depending on the values of learning parameters. Our purpose is to measure what kind of links are predicted depending on the mixing. To do so, we focus on the *Highschool* and *Infocom* datasets. As discussed later, this will give us additional hints about the relatively low contribution of structural metrics in previous series of experiments.

5.1 Simplified experimental setting

The experimental protocol focuses on the performance gain that can be achieved by combining two link stream metrics. Here, we focus on the training and validation periods only. Then, to understand the information brought by each metric, we combine two metrics at a time, the weight of each metric being related to the parameter α, according to the following equation:

$$\mathcal{F}(uv) = \alpha \cdot \widehat{PAE1000S}(uv) + (1-\alpha) \cdot \hat{m}(uv), \tag{10}$$

where $PAE1000S$ is the index associated to the *activity during the last* 1,000 *seconds* (see Section 3.2.2). We use it as a reference because we have seen that it is the metric which gets most weight in the case of the *Highschool* dataset experiment and the second most weighted metric in the *Infocom* dataset experiment. For each experiment we combine the *activity during*

the last 1,000 *seconds* with $\hat{m}(uv)$ the index corresponding to another metric, as indicated in the legend of each figure.

5.2 Structural and temporal metrics combinations

We study how the use of different metrics with different weights affects predictions on *Highschool* and *Infocom* datasets. For this purpose, we combine three of the metrics presented in Section 3.2.1 with the *activity during the last* 1,000 *seconds*. For each dataset we combine it to the most important metric index used by our learning algorithm, that is to say the *activity during the last* 100 *seconds* for the *Highschool* dataset and the *Pair Activity Extrapolation* for the *Infocom* dataset. We also combine it with the *number of Common Neighbors* and the *Sørensen Index* to study how the algorithm mixes *activity during the last* 1,000 *seconds* to structural metrics. Plots in Figure 8 represent the F-score for different pairs of metric combinations, as a function of α for each of the two datasets. Higher values of α represent a greater weight of the *activity during the last* 1,000 *seconds* in the prediction, while lower α indicate a greater weight of the other metric involved.

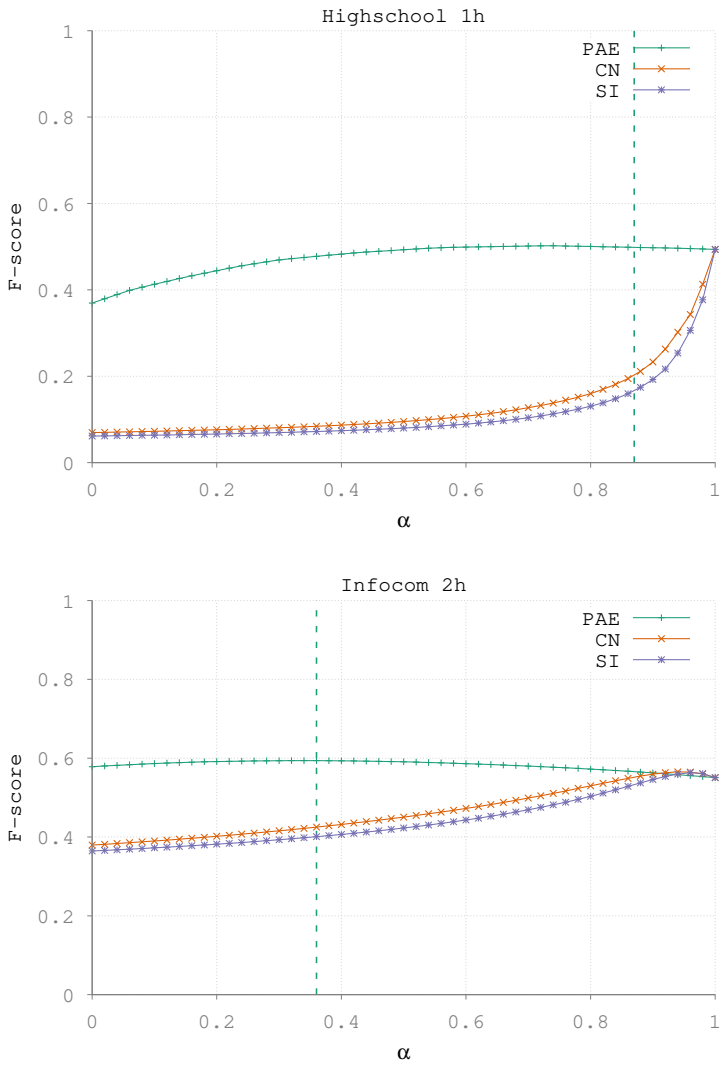

Figure 8: F-score of the predictions with different weight ratios between the *activity during the last* 1000 *seconds* and three other metrics for *Highschool* (1h) (top) and between the *Pair activity extrapolation* and three other metrics for *Infocom* (2h) (bottom). Dashed lines indicate the ratio between the two main metrics during the experiment in Section 4.4 (in green in both figures).

In the case of *Highschool*, we use a 1 hour training period. We combine successively *PAE1000S* with *PAE100S*, the *number of Common Neighbors* (CN) and the *Sørensen Index* (SI). The plots for *PAE100S* show that F-score increases with α until reaching a maximum for $\alpha \simeq 0.74$ and then decreases until $\alpha = 1$. With both structural metrics we observe roughly similar behaviors: a slow increase until $\alpha \simeq 0.8$ then a rapid increase until a maximum at $\alpha = 1$. This shows that the algorithm simply does not benefit at all from the structural metrics in this experiment. Indeed, when $\alpha = 1$, the prediction index comes down exactly to the *activity during the last* 1000 *seconds*.

We then apply the same protocol to *Infocom*, with a 1 hour training period. This time we combine *PAE1000S* with the *Pair Activity Extrapolation*, *CN* and *SI*, successively. The mixing between temporal metrics (*PAE1000S* and *PAE*) behaves qualitatively in the same way as the mixing of temporal metrics in the *Highschool* case, that is a convex curve with a maximum for $\alpha \simeq 0.35$. The combination with structural metrics behave qualitatively in a different way than in the *Highschool* case. Both plots have similar shapes, showing an increase until a maximum is reached at $\alpha \simeq 0.94$ then decreasing until $\alpha = 1$. These observations indicate that in this experiment, combining a temporal metric with a structural metric may lead to an improvement of the F-score.

Observations of this section confirm that the algorithm is able to find a trade-off between metrics that improve the performance of the prediction. Moreover, we see more clearly that the mixing tends to heavily favor temporal metrics. We will now turn our attention to the kind of links which are consequently predicted.

5.3 Nature of predicted links

Using the same prediction protocol, we divide the set of node pairs into two categories, to see what kind of links are predicted in each case. On the one hand, some pairs have not interacted during T, so that predicting the occurrence of a link between nodes of this kind is predicting a new link in the stream. We call *new link* any (t, uv) in the prediction stream L' such that $\nexists\, x \in T, (x, uv) \in E$. On the other hand, other pairs have interacted during T and predicting such an interaction is predicting the repetition of a link. We call *recurrent link* any (t, uv) in the prediction stream L' such that there exists a link $(x, uv) \in E$. We apply the evaluation method on the complete set of pairs and on each of these two subsets. We display in Figure 9 the obtained F-score as a function of α for the two categories of pairs aforementioned on both datasets as well as for the complete set of pairs.

We can see on the *Highschool* results that the F-score corresponding to the recurrent link category increases to a maximum for $\alpha \simeq 0.98$, while for the new link category it remains almost constant until $\alpha \simeq 0.98$, at which point it decreases to zero. The F-score for the complete set of links is the same that previously, with a maximum at $\alpha = 1$. *Activity during the last* $1{,}000$ *seconds* alone is not able to predict new links by construction and thus yields a null F-score. The performance of the prediction of recurrent links improves as more weight is given to the *Pair Activity Extrapolation*. However, we do not see the same effect of stagnation for a wide range of α.

The plot corresponding to the *Infocom* dataset shows a quite different behavior. We can see that the F-score accounting for new link predictions starts from 0.22 for $\alpha = 0$ and slowly decreases until $\alpha = 0.8$ at which point it sharply decreases to 0. Regarding the recurrent links prediction the F-score starts from 0.41 and reaches a maximum of 0.62 for $\alpha \simeq 0.92$ and then decreases to 0.56. We observe that new link prediction quality is noticeably lower than the recurrent link

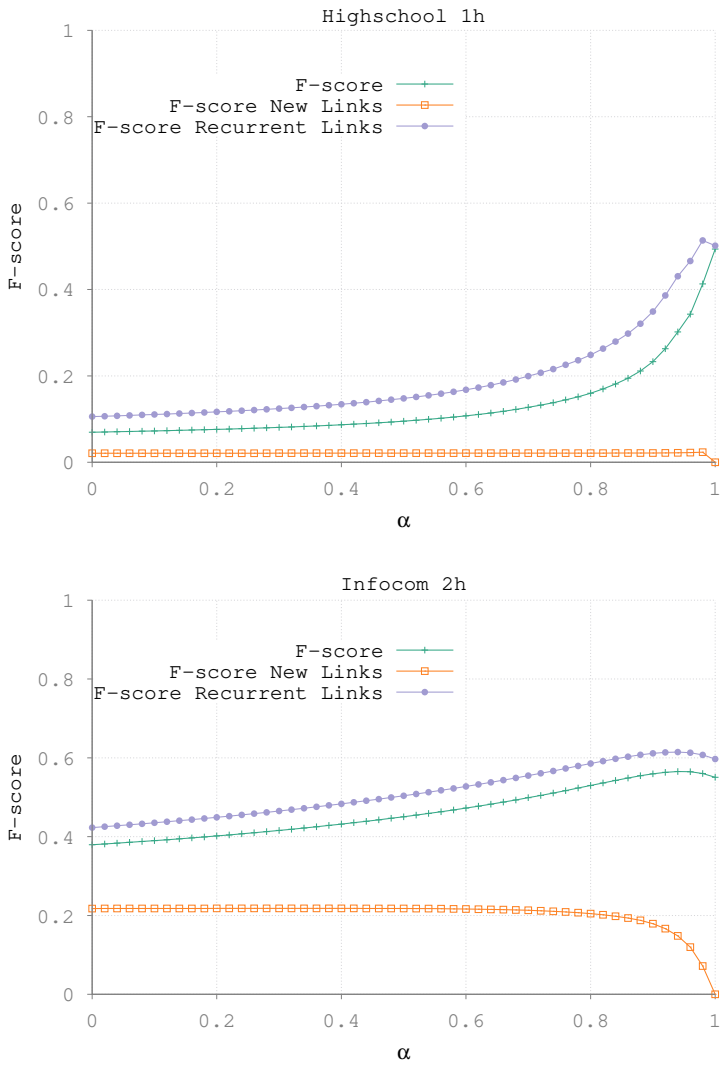

Figure 9: F-score of the predictions for the mixing between *PAE1000S* and *CN* as a function of α for different categories of node pairs for the *Highschool* (top) and *Infocom* (bottom) datasets using respectively 1 hour and 2 hours long observation and predictions periods. Green crosses: All links, Orange squares: New links, Purple circles: Recurrent links.

prediction quality in both experiments. The difficulty of this task is mainly due to the class imbalance problem, which is a well-known issue in the field of link prediction Lichtenwalter *et al.* (2010). It is related to the fact that the number of links actually occurring is small in regards to the potential number of links, which is simply the number of node pairs.

We observe that the number of links predicted in each category plays an important role in the prediction quality. Obviously, different metrics tend to predict preferentially different types of activity. These experiments show that, by choosing specific metrics combination, the prediction focuses on different kinds of activity, involving different kinds of links. With that in mind, we can put forward an explanation for the lower weights of structural metrics in the combinations computed by the learning algorithm in Section 4.4. Indeed, by only selecting temporal metrics the algorithm avoids the difficult task of predicting new links to favor the easier task of predicting recurrent ones. While it improves the overall F-score, it is also detrimental to the variety of links predicted. To counterbalance this unwanted behavior, we explore in the next section how to integrate the notion of class of node pairs to the protocol.

VI CLASSES OF PAIRS

As seen in Section 5.3, specific metrics combinations favor different kinds of node pairs. However, our protocol does not make this distinction. In this section, we propose a way to improve activity prediction in link streams by differentiating node pairs. For this purpose, we introduce classes of node pairs based on previous activity. Other choices of class definitions are possible, but our previous observations indicate that new links and recurrent links behave very differently. Moreover, previous studies of related prediction problems also agree with this – see for example (Scholz *et al.*, 2012) on link prediction. We compute a specific set of values for parameters α_m for each of these classes, allowing to adapt metric weights to each type of node pairs.

6.1 Class definition and metric to optimize

We choose to separate classes by their level of activity. The underlying idea is to create classes reflecting on the one hand link prediction in graphs and on the other hand time series prediction. Indeed, predicting node pairs with no or low past activity is a task related to the prediction of new links, while predicting the future activity of recurrent links resembles a time series prediction task.

We define three classes of node pairs. Class C1 is populated with node pairs which have not interacted during the training period. Class C2 gathers the pairs with less than a given number of links k during the training period. Lastly, the remaining high activity pairs are assigned to the class C3. Formally:

$$C1 = \{uv \in V \otimes V,\ |(t, uv) \in E_1| = 0\}, \tag{11}$$

$$C2 = \{uv \in V \otimes V,\ 0 < |(t, uv) \in E_1| \leq k\}, \tag{12}$$

$$C3 = \{uv \in V \otimes V,\ |(t, uv) \in E_1| = 0 > k\}. \tag{13}$$

In the following experiments, in a *proof of concept* spirit we set the threshold between the class C2 and C3 to $k = 5$ links. However determining an adequate value of the threshold in general is a difficult task, which depends on the dataset under consideration.

We follow the same experimental protocol for each class, using different values of parameters, which are computed during the learning phase. However, if we aim at optimizing the overall F-score, this protocol tends to favor high activity classes. Therefore, we have to update adequately the metrics to optimize with the gradient descent algorithm, in order to ensure we predict a wide variety of links. We then define the prediction score $\overline{F}$ as the harmonic mean of the F-scores for each class, that is

$$\frac{1}{\overline{F}} = \frac{1}{3}\left(\frac{1}{F(C_1)} + \frac{1}{F(C_2)} + \frac{1}{F(C_3)}\right), \tag{14}$$

where $F(C_1)$, $F(C_2)$ and $F(C_3)$ and the F-scores computed over each subset of node pairs defined by our classes.

Once the parameters are optimized, we reassign each node pairs in our different classes based on their activity during the observation stream L_2. We then follow our usual protocol, combining the metrics for each classes using its specific set of parameters.

©JIMIS, Creative Commons
Volume: 5 - Year: 2019, DOI: 10.18713/JIMIS-150719-5-3

6.2 Reduction of the parameter space

In order to achieve computations in a reasonable amount of time, we reduce the set of metrics compared to the experiments in Section 4.4. In these series of experiments, we use the following metrics: *number of Common Neighbors*, the *Sørensen Index*, the *Weighted number of Common Neighbors* and our temporal metrics, *Pair Activity Extrapolation* (*PAE*) , *PAE10L*, the activity during the last 10 links, *PAE1000S* and *PAE10000S*, the activity during the last 1,000 and 10,000 seconds respectively, drawing benefit from the fact that several metrics are highly correlated.

Indeed, we reported in Figure 10 the correlation matrix of the score given by each metric over every nodes for the training set of the experiment on *Infocom* for periods of 2h. We can see that most structural metrics are highly correlated. Particularly, *number of Common Neighbors*, *Adamic-Adar index* and *Resource Allocation index* seem to bring similar information. We also see that *Sørensen Index* and *Jaccard Index* are very correlated to each other. Therefore, we choose one metric for each group, the *number of Common Neighbors* and the *Sørensen Index*. This will allow some variety in the structural metrics available. Similarly, except for the *Weighed number of Common Neighbors*, the weighed variation of structural metrics are close to the other structural metric. Thus, in the following experiments, we keep the *Weighted number of Common Neighbors*. Temporal metrics tend to focus on different time scales of the dataset and, while correlated, bring different information about the dynamic of the system. Therefore we keep them all for our following experiment.

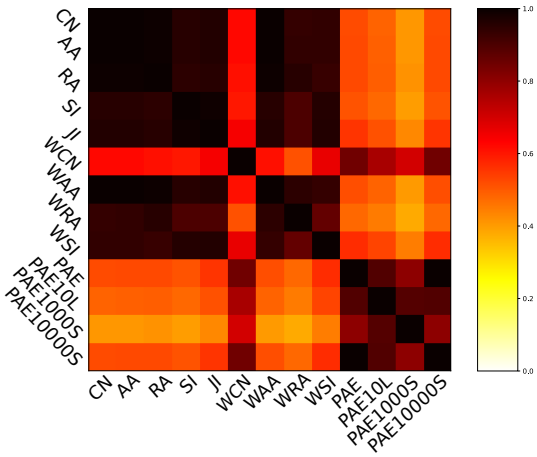

Figure 10: Correlation matrix of the values of each metric over all pair of nodes during the 2 hours training phase on *Infocom*.

6.3 Experimental results

We apply the protocol on *Infocom* using the same training, validation, observation and prediction periods as in the 2h experiments detailed in Section 4.3. Results are summarized in Table 5. We present the results for each class C1, C2 and C3, as well as the overall results obtained when combining the prediction for each class, denoted *All Class*. For comparison purposes, we also present the results obtained in the same experiments but without using classes of node pairs. The corresponding results are prefixed C0. Thus, the F-scores associated with C0-1 C0-2 and C0-3 are the scores obtained when applying the evaluation protocol on the subset of node pairs relative to each pair classes. Precision, Recall and F-score are reported, as well as overall activity predicted and occurring. These results correspond to the average over 10 runs of the experiments, with standard deviation below 0.01 for the F-score, below 0.05 for the Precision and Recall, and below (up to) 22% for the number of predicted links.

©JIMIS, Creative Commons
Volume: 5 - Year: 2019, DOI: 10.18713/JIMIS-150719-5-3

Class	F-score	Precision	Recall	N predicted	N' occurring
C0	0.55	0.50	0.60	16,737	14,051
C0-1	0.05	0.31	0.02	209	2,668
C0-2	0.37	0.46	0.30	1,400	2,134
C0-3	0.63	0.51	0.84	15,126	9,137
AllClass	0.53	0.49	0.59	16,737	14,051
C1	0.25	0.23	0.29	3,300	2,668
C2	0.41	0.37	0.48	2,800	2,134
C3	0.65	0.60	0.70	10,500	9,137

Table 5: F-score and number of link predicted and appeared by class on *Infocom* using 2h periods.

First, we observe that the number of predicted links in class C1 is more than 15 times higher than in class C0-1. The Precision slightly decreases, but the Recall is so much higher that the new protocol achieves a 0.25 F-score, to be compared to the 0.05 F-score obtained on this class in the standard series of experiments. We also note that the activity predicted over the whole class C2 nearly doubles, which also leads to a performance improvement, from 0.37 to 0.41 F-score. It is very interesting to note that the introduction of classes also improves the prediction in the classes C3, from a 0.63 to a 0.65 F-score. The number of links predicted in this class largely decreases to balance the fact that the activity predicted in the two other classes increases, making it closer to the actual activity observed. We investigate this effect in more details later.

Quite counter-intuitively, while the prediction is improved in each class, the F-score predicted overall decreases from 0.55 to 0.53. This phenomenon is a sort of amalgamation paradox. Precisely, the F-score is the harmonic mean of the Precision and the Recall, hence its computation is related to the proportion of links that should be predicted and that is actually predicted in each class. This may lead to the fact that the F-score over the union of subsets and the F-scores of each subset considered separately evolve according to different trends.

We display in Figure 11, the values of parameters α_m computed by the learning algorithm to achieve a prediction over each class (averaged over 10 experiments). As expected, the method tends to use different combinations to make a prediction over each class. Considering class C1, the algorithm favors structural and hybrid metrics. These metrics are commonly used for link prediction in graphs, which makes sense given that predicting new link activity is closely related to this problem.

Class C2 tends to mix structural metrics to temporal ones. Pairs in this class have intermediary behavior and the algorithm combines information of different nature to predict these behaviors. Interestingly, prediction on class C3 also mixes temporal metrics to structural ones, even if temporal metrics remain predominant. It contrasts with prediction in the experiments without class, where no or little weight was distributed to purely structural metrics, and little weight to hybrid ones. Our interpretation is that in the former protocol, the optimization favored temporal metrics in order to exclude node pairs which had never interacted and then are much more risky to predict, while in the later protocol, the introduction of classes allows to focus on recurrent links only, so that giving weight to structural metrics becomes an interesting option again.

Experiments on *Highschool* and *Reality Mining* present similar results and are not reported here. Concerning the *Taxi* dataset, we think that the different nature of interactions between nodes necessitate a specific analysis which is left for future works.

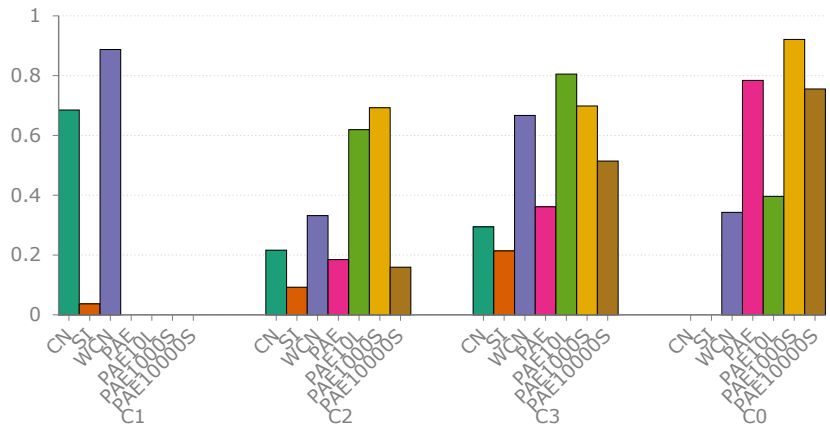

Figure 11: Metric coefficients repartition computed by the learning algorithm during prediction by classes on *Infocom* using 2 hours periods.

In the light of these experiments, we can see that introducing classes of pairs in the problem of activity prediction may be a good option in order to improve the diversity of links to be predicted, which opens interesting leads for future works on the topic. In particular, finding a way to compute automatically relevant activity thresholds would be a significant improvement to the current version of the classes definition.

VII CONCLUSION

In this work, we proposed an activity prediction protocol adapted to the link stream formalism, making possible to advantageously use the rich information contained in this modeling. It relies on a flexible way to combine the information from metrics which capture characteristics of the stream. We also proposed evaluation metrics adapted to our problem. Our experiments show that our protocol is able to find efficient combinations of structural and temporal metrics that lead to performance improvements, compared to benchmarks such as the past activity extrapolation. We also investigated how this algorithm tends to exclude specific types of node pairs, leading to less variety in the predicted links. However, we showed that it is possible to mitigate this issue by introducing classes of node pairs in activity prediction, so that a better balance of activity can be obtained in each class. Most importantly, our protocol is designed in a modular way, such that each part is independent from the others and can be replaced or improved, depending on the application a user is interested in.

Different improvements are considered for future works. The metrics presented in this work are classical metrics used for link prediction in graphs or basic ways to capture the temporal information of the stream. As our protocol is ready to combine new metrics, we intend to design refined ones that are able to detect more subtle dynamical metrics of the stream, for example implementing pattern mining techniques to identify typical motifs of the short term dynamics. We also made the assumption that the activity remains constant from the observation period to the prediction period. However, this hypothesis is not always satisfied and greatly depends on the data under concern. Models developed in the context of time series prediction, like the ARIMA model which extrapolates precisely past activity (Huang and Lin, 2009), would certainly allow to better evaluate the number of predicted links. Finally, we also want to investigate further how our algorithm behaves with different classes of pairs. Other definitions are possible, for example structural classes, where node pairs of a same community would belong to a same class.

References

Adamic L. A., Adar E. (2003). Friends and neighbors on the web. *Social networks 25*(3), 211–230. doi:10.1016/S0378-8733(03)00009-1.

Al Hasan M., Chaoji V., Salem S., Zaki M. (2006). Link prediction using supervised learning. In *SIAM Conference on Data Mining / Workshop on link analysis, counter-terrorism and security*, pp. 12. URL: https://archive.siam.org/meetings/sdm06/workproceed/Link%20Analysis/12.pdf.

Bracciale L., Bonola M., Loreti P., Bianchi G., Amici R., Rabuffi A. (2014, July). *CRAWDAD dataset roma/taxi (v. 2014-07-17)*. Downloaded from https://crawdad.org/roma/taxi/20140717/taxicabs. traceset: taxicabs. doi:10.15783/C7QC7M.

Brockwell P. J., Davis R. A. (2013). *Time series: theory and methods*. Springer. doi:10.1007/978-1-4419-0320-4.

Casteigts A., Flocchini P., Quattrociocchi W., Santoro N. (2012). Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems 27*(5), 387–408. doi:10.1080/17445760.2012.668546.

Clauset A., Moore C., Newman M. E. J. (2008). Hierarchical structure and the prediction of missing links in networks. *Nature 453*(7191), 98–101. doi:10.1038/nature06830.

da Silva Soares P. R., Prudêncio R. B. C. (2012). Time series based link prediction. In *International Joint Conference on Neural Networks*, pp. 1–7. IEEE. doi:10.1109/IJCNN.2012.6252471.

Davis D., Lichtenwalter R., Chawla N. V. (2013). Supervised methods for multi-relational link prediction. *Social network analysis and mining 3*(2), 127–141. doi:10.1007/s13278-012-0068-6.

Dunlavy D. M., Kolda T. G., Acar E. (2011). Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data 5*(2), 10. doi:10.1145/1921632.1921636.

Eagle N., Pentland A. S. (2005, July). *CRAWDAD dataset mit/reality (v. 2005-07-01)*. Downloaded from https://crawdad.org/mit/reality/20050701. doi:10.15783/C71S31.

Holme P., Saramäki J. (2012). Temporal networks. *Physics Reports 519*(3), 97–125. doi:10.1016/j.physrep.2012.03.001.

Huang Z., Li X., Chen H. (2005). Link prediction approach to collaborative filtering. In *5th ACM/IEEE-CS Joint Conference on Digital libraries*, pp. 141–142. ACM. doi:10.1145/1065385.1065415.

Huang Z., Lin D. K. J. (2009). The time-series link prediction problem with applications in communication surveillance. *INFORMS Journal on Computing 21*(2), 286–303. doi:10.1287/ijoc.1080.0292.

Jaccard P. (1901). Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin de la Société vaudoise des sciences naturelles 37*, 547–579. doi:10.5169/seals-266450.

Kossinets G. (2006). Effects of missing data in social networks. *Social networks 28*(3), 247–268. doi:10.1016/j.socnet.2005.07.002.

Latapy M., Viard T., Magnien C. (2018). Stream graphs and link streams for the modeling of interactions over time. *Social Network Analysis and Mining 8*(1), 61. doi:10.1007/s13278-018-0537-7.

Liben-Nowell D., Kleinberg J. (2007). The link-prediction problem for social networks. *Journal of the American society for information science and technology 58*(7), 1019–1031. doi:10.1002/asi.20591.

Lichtenwalter R. N., Lussier J. T., Chawla N. V. (2010). New perspectives and methods in link prediction. In *16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 243–252. doi:10.1145/1835804.1835837.

Lü L., Zhou T. (2011). Link prediction in complex networks: A survey. *Physica A 390*(6), 1150–1170. doi:10.1016/j.physa.2010.11.027.

Mastrandrea R., Fournet J., Barrat A. (2015). Contact patterns in a high school: a comparison between data collected using wearable sensors, contact diaries and friendship surveys. *PLoS ONE 10*(9), e0136497. doi:10.1371/journal.pone.0136497.

Murata T., Moriyasu S. (2007). Link prediction of social networks based on weighted proximity measures. In *Proceedings of the IEEE/WIC/ACM international conference on web intelligence*, pp. 85–88. IEEE Computer Society. doi:10.1109/WI.2007.52.

Pujari M., Kanawati R. (2012). Supervised rank aggregation approach for link prediction in complex networks. In

21st International Conference on World Wide Web, pp. 1189–1196. doi:10.1145/2187980.2188260.

Scholz C., Atzmueller M., Stumme G. (2012). On the predictability of human contacts: Influence factors and the strength of stronger ties. In *International Conference on Privacy, Security, Risk and Trust / International Conference on Social Computing*, pp. 312–321. doi:10.1109/SocialCom-PASSAT.2012.49.

Scott J., Gass R., Crowcroft J., Hui P., Diot C., Chaintreau A. (2009, May). *CRAWDAD dataset cambridge/haggle (v. 2009-05-29)*. Downloaded from http://crawdad.org/cambridge/haggle/20090529. doi:10.15783/C70011.

Sorensen T. (1948). A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on danish commons. *Kongelige Danske Videnskabernes Selskab 5*, 1–34. URL: http://www.royalacademy.dk/Publications/High/295_S%C3%B8rensen,%20Thorvald.pdf.

Tabourier L., Bernardes D. F., Libert A.-S., Lambiotte R. (2019). Rankmerging: a supervised learning-to-rank framework to predict links in large social networks. *Machine Learning in press*, 1–28. doi:10.1007/s10994-019-05792-4.

Tabourier L., Libert A.-S., Lambiotte R. (2016). Predicting links in ego-networks using temporal information. *EPJ Data Science 5*(1), 1. doi:10.1140/epjds/s13688-015-0062-0.

Tylenda T., Angelova R., Bedathur S. (2009). Towards time-aware link prediction in evolving social networks. In *3rd workshop on social network mining and analysis*, pp. 9. ACM. doi:10.1145/1731011.1731020.

Viard T., Fournier-S'niehotta R., Magnien C., Latapy M. (2018). Discovering patterns of interest in ip traffic using cliques in bipartite link streams. In *International Workshop on Complex Networks*, pp. 233–241. Springer. doi:10.1007/978-3-319-73198-8_20.

Viard T., Latapy M. (2014). Identifying roles in an ip network with temporal and structural density. In *Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 801–806. IEEE. doi:10.1109/INFCOMW.2014.6849333.

Wang P., Xu B., Wu Y., Zhou X. (2015). Link prediction in social networks: the state-of-the-art. *Science China Information Sciences 58*(1), 1–38. doi:10.1007/s11432-014-5237-y.

Zhou T., Lü L., Zhang Y.-C. (2009). Predicting missing links via local information. *The European Physical Journal B 71*(4), 623–630. doi:10.1140/epjb/e2009-00335-8.

A ACKNOWLEDGEMENTS

This work is supported in part by the ANR (French National Agency of Research) under grant ANR-15-CE38-0001 (AlgoDiv) and by the Paris Ile-de-France Region and program FUI21 under grant 16010629 (iTRAC).