The Initialization and Parameter Setting Problem in Tensor Decomposition-based Link Prediction

S. d. S. Fernandes, H. F. Tork and J. M. P. da Gama

Citation:
S. d. S. Fernandes, H. F. Tork and J. M. P. da Gama, "The Initialization and Parameter Setting Problem in Tensor Decomposition-Based Link Prediction," *2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, Tokyo, Japan, 2017, pp. 99-108.
DOI: 10.1109/DSAA.2017.83

URL:
http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8259768&isnumber=8259747

The Initialization and Parameter Setting Problem in Tensor Decomposition-based Link Prediction

Sofia Fernandes
LIAAD, INESC TEC
University of Porto
Porto, Portugal
sdsf@inesctec.pt

Hadi Fanaee-T
Department of Biostatistics
University of Oslo
Oslo, Norway
hadift@medisin.uio.no

João Gama
LIAAD, INESC TEC
University of Porto
Porto, Portugal
jgama@fep.up.pt

***Abstract*—Link prediction is the task of social network analysis whose goal is to predict the links that will appear in the network in future instants. Among the link predictors exploiting the time evolution of the networks, we can find the tensor decomposition-based methods. A major limitation of these methods is the lack of appropriate approaches for estimating their parameters and initialization. In this paper, we address this problem by proposing a parameter setting method. Our proposed approach resorts to optimization techniques to drive the search for an adequate parameter and initialization choice.**

I. Introduction

Social network analysis is the field of research which aims at understanding and unveiling the hidden patterns of interactions in the networks. This research area encompasses, among others, the task of link prediction. The link prediction problem in time-evolving networks may be described as follows: given the states of the network at the previous T time instants, how to predict future (new or re-occurring) links? Thus, given the sequence of states of the network from instants 1 to T, the goal is to predict which are the links which are more likely to occur at instant $T + 1$.

One of the directions followed to tackle this problem was to consider tensor decomposition-based methods [1], [2]. The idea behind these methods is to exploit the multi-way structure of time-evolving networks, which have a time dimension associated to the network topology. In particular, the authors combined PARAFAC tensor decomposition (CP) [3] with forecasting techniques to define the link predictor.

Since, this type of methods are not parameter free (both CP and forecasting methods have parameters); an attempt to use the CP-based link predictors will arise the question: how should the model parameters be set in order to obtain a good predictor? According to our preliminary experiments, the methods found in the literature, which only cover one parameter of the model, did not perform well when applied to this task.

This work is financed by the ERDF European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme within project POCI-01- 0145-FEDER-006961, and by National Funds through the FCT - Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) as part of project UID/EEA/50014/2013. S.F. also acknowledges the support of FCT via the PhD scholarship PD/BD/114189/2016. J.G. acknowledges a grant from Jean d'Alambert at Telecom Paris.

Another issue that has been neglected in the literature is the initialization of CP. Currently, it is not clear how the performance of these CP-based models is affected by the initialization: are the CP-based link predictors not sensitive to initialization? Or does the initialization have a strong impact on the performance of such models? It is important to understand the influence of this factor in order to guarantee the reproducibility of the results, a key issue in science.

In this work, we address both the initialization issue and the parameter setting problem in CP-based predictors. In particular:

- We provide empirical evidence that the initialization affects the performance of CP-based link predictors;
- We propose a method for estimating both the parameters and the initialization of a CP-based link predictor;
- We carried out a study on the performance of the models obtained by our procedure in real world data.

Based on this, we highlight that the problem being addressed in this work is not the link prediction problem itself, but the problem of parameter setting and initialization in tensor-based link predictors.

The rest of the paper is organized as follows. In Section II we describe the problem in more detail. In Section III, we cover the theoretical background associated with the work. The proposed solution is described in Section IV and the experiments results are exposed and analyzed in Section V. We present the future work and conclude the paper in Section VI.

II. Problem Description

After the introduction of CP-based link predictors in 2011 [1], [2], it would be expected that such models would raise the researchers' attention leading to the further emergence of new related works. However, few advances have been made in this context.

A limitation of this type of link predictors is the setting of their parameters: the parameters of the models are expected sources of variability and, consequently, an incorrect parameter setting may compromise the performance of the model.

Regarding CP parameters, the choice of the number of components is traditionally carried out using the core consistency diagnostic (CORCONDIA) [4], [5]; nevertheless, according

to our preliminary tests, the results of such tool lead to CP approximations with extremely low fitting, which were compromising the performance of the CP-models. On the other hand, the authors of one of the CP-based link predictors overcame the problem of choosing the number of components by using an ensemble of CP-based link predictors with varying number of components.

Besides these approaches (which cover only a parameter of the model), no appropriate methods for estimating the CP-based link predictors parameters were, to the best of our knowledge, developed.

We note that, despite of not being addressed in the literature, this problem is expected since parameter-dependent methods require tuning. However, this source of variability was not unique: during the application of these CP-based link predictors, we also observed that their performance was dependent on the initialization. To the best of our knowledge, this influence of the initialization on the performance of the models was not addressed nor reported in the literature so far.

Thus, when applying CP-based link predictors we identified two issues: the problem of initialization and the problem of parameter setting. Based on this, we drove our work in order to address such problems.

III. Background

Since the proposed method resorts to optimization techniques to tackle the problem of initialization and parameter setting in CP-based predictors, in this section we cover both the background associated to tensor theory and the optimization algorithm considered in the proposed approach.

A. Tensors Theory

1) Notation: The notation used in this work is summarized in table I.

TABLE I
NOTATION

Symbol	Description
$\circ$	vector outer product
$\mathbf{x}$	vector (bold lower case)
$\mathbf{x}(i)$	i^{th} entry of vector $\mathbf{x}$
$\mathbf{X}$	matrix (bold upper case)
$\mathbf{X}^T$	matrix transpose of $\mathbf{X}$
$\mathbf{X}(i,j)$	entry (i,j) of matrix $\mathbf{X}$
$\mathbf{X}(i,:)$	i^{th} row of matrix $\mathbf{X}$
$\mathbf{X}(:,j)$	j^{th} column of matrix $\mathbf{X}$
$\mathcal{X}$	tensor
$\mathcal{X}(i_1,\ldots,i_M)$	entry $(i_1,\ldots,i_M)$ of tensor $\mathcal{X}$
$\mathbf{X}_{(d)}$	mode-d matricization of tensor $\mathcal{X}$

2) Tensors: Informally, a tensor may be described as a high order generalization of a matrix. In mathematical terms, a M-order tensor is a M-dimensional array $\mathcal{X} \in \mathbb{R}^{N_1 \times N_2 \times \ldots \times N_M}$ where N_i is referred to as the dimensionality of mode i and $N_1 \times N_2 \times \ldots \times N_M$ is the size of the tensor. It should be noted that 1-order and 2-order tensors are, respectively, vectors and matrices.

The norm of such a M-order tensor is defined as

$$||\mathcal{X}|| = \sqrt{\sum_{i_1=1}^{N_1}\sum_{i_2=1}^{N_2}\ldots\sum_{i_M=1}^{N_M}[\mathcal{X}(i_1,i_2,\ldots,i_M)]^2} \ .$$

In some cases, it may be useful to rearrange the tensor as a matrix, such operation is known as unfolding or matricizing. Formally, given a tensor $\mathcal{X} \in \mathbb{R}^{N_1 \times N_2 \times \ldots \times N_M}$, the mode-$d$ matricization of $\mathcal{X}$ consists of reshaping the original tensor into a matrix in $\mathbb{R}^{N_d \times (\prod_{i \neq d} N_i)}$, obtained by fixing each mode-d index and varying the indexes the other modes. The resulting matrix is denoted by $\mathbf{X}_{(d)}$.

As illustrative example of the matricization operation, let us consider a generic 3-order tensor $\mathcal{X} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$, then entry (i,j,k) of the tensor will map to entry $(i, N_2(k-1)+j)$ of mode-1 matricization; to entry $(j, N_1(k-1)+i)$ of mode-2 matricization and to entry $(k, N_1(j-1)+i)$ of mode-3 matricization.

In order to simplify the notation, we restrict our theory exposition to 3-order tensors. However, we note that the methods exposed in this section can be generalized to higher orders.

3) PARAFAC (CP) Tensor Decomposition: The CP decomposition [3] of a 3-order tensor $\mathcal{X} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$ is given by

$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \tag{1}$$

where R is a positive integer, referred to as the number of components or factors, and $\mathbf{a}_r \in \mathbb{R}^{N_1}$, $\mathbf{b}_r \in \mathbb{R}^{N_2}$, $\mathbf{c}_r \in \mathbb{R}^{N_3}$.

Elementwise, expression (1) assumes the form:

$$\mathcal{X}(i_1,i_2,i_3) \approx \sum_{r=1}^{R} \mathbf{a}_r(i_1)\mathbf{b}_r(i_2)\mathbf{c}_r(i_3) \ .$$

The vectors associated to the same mode may be grouped in a matrix so that we obtain 3 matrices describing the decomposition result: $\mathbf{A} \in \mathbb{R}^{N_1 \times R}$, $\mathbf{B} \in \mathbb{R}^{N_2 \times R}$ and $\mathbf{C} \in \mathbb{R}^{N_3 \times R}$, each having the R corresponding vectors as columns, that is, $\mathbf{A}(:,r) = \mathbf{a}_r, \mathbf{B}(:,r) = \mathbf{b}_r$ and $\mathbf{C}(:,r) = \mathbf{c}_r$. These matrices are referred to as factor matrices.

The traditional algorithm for computing the CP decomposition is the alternating least squares CP (CP-ALS) [6]. Given a number of factors R, the goal of CP-ALS is to find factor matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ minimizing the approximation error, which is given by:

$$||\mathcal{X} - \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r|| \tag{2}$$

The idea of the algorithm is to iteratively update each of the factor matrices. The update of a given factor matrix is performed by solving the minimization problem considering the other factor matrices fixed. By considering the minimization problem with all but one matrix fixed, the authors obtain an explicit form of the solution.

Thus, besides the number of factors, the solution of CP-ALS is also influenced by: (i) the maximum number of

iterations allowed; (ii) the minimum level of change in the approximation error allowed between consecutive iterations and (iii) the order in which the modes are updated.

Moreover, the initial factor matrices must be provided to the algorithm. Two common approaches to generate the initial factor matrices are randomly or SVD-based. In the SVD-based approach, the factor matrices associated to each mode d are obtained by applying the singular value decomposition (SVD) to the matrix $\mathbf{X}_{(d)}\mathbf{X}_{(d)}^T$.

Given the CP decomposition results, the fitting rate of the approximation $\tilde{\mathcal{X}} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r$ is defined as:

$$\%_{\text{fitting}} = 100 \times \left(1 - \frac{||\mathcal{X} - \tilde{\mathcal{X}}||}{||\mathcal{X}||}\right) \ .$$

In case the fitting rate is low it means that the approximation is a poor representation of the original data. On the other hand, if the fitting rate is maximum (100%), it means that the equality $\mathcal{X} = \tilde{\mathcal{X}}$ holds and the approximation represents exactly the original data.

4) CP-based Link Predictors: Given a time-evolving network, we can construct a tensor by considering the sequence of adjacency matrices describing the state of the network at each instant. Therefore, the idea of link prediction CP-based methods is to exploit the $entities \times entities \times time$ structure of the tensor (and inherent interactions) to infer future links.

In this work we employ the term CP-based link predictors to refer to link prediction methods that combine CP-ALS decomposition with forecasting algorithms. In particular, we restrict this term to link prediction methods which encompass the following steps:

1) apply CP-ALS decomposition on the tensor formed by the t available network adjacency matrices to obtain the factor matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$, where $\mathbf{C}$ is the factor matrix associated to the time mode;
2) apply a forecasting method to the temporal factor matrix $\mathbf{C}$ in order to estimate the future temporal trend of the network, $\mathbf{C}(t+1,:)$, corresponding to the next row of the temporal matrix;
3) combine the estimated future temporal trend $\mathbf{C}(t+1,:)$ with the factor matrices associated to the entities modes, $\mathbf{A}$ and $\mathbf{B}$, to estimate the future state of the network, that is, to estimate the adjacency matrix of the network at instant $t+1$:

$$\mathbf{S} = \sum_{r=1}^{R} (\mathbf{a}_r \circ \mathbf{b}_r)\, \mathbf{C}(t+1, r)$$

We refer to the resulting estimation of the future network state, $\mathbf{S}$, as score matrix.

What differentiates two CP-based link predictors is the forecasting algorithm applied. In this context, Dunlavy *et al.* [1] considered the sum across the last (most recent) L available time instants, that is, the authors defined

$$\mathbf{C}(t+1, r) = \sum_{k=t-L+1}^{t} \mathbf{c}_r(k) \ .$$

Spiegel [2] employed exponential smoothing considering the same smoothing factor across the several components (columns of $\mathbf{C}$).

B. The Nelder-Mead method

The Nelder-Mead algorithm [7] is an optimization method which drives the search for the minimum based on the cost function values at a given simplex. The simplex vertexes are iteratively updated in order to sequentially discard the vertex associated with the largest cost function value.

Thus, assuming that the cost function, f, is defined in $\mathbb{R}^n$, the first step consists in finding a simplex of $n+1$ vertexes: $\{P_1, P_2, \ldots, P_{n+1}\}$. Then, the cost function values of the simplex vertexes are computed: $f_i = f(P_i)$.

Based on the result obtained, the indexes l and h are computed so that:

$$f_l = \min(\{f_i\}_1^{n+1}) \wedge P_l = \text{argmin}(\{f_i\}_1^{n+1})$$

and

$$f_h = \max(\{f_i\}_1^{n+1}) \wedge P_h = \text{argmax}(\{f_i\}_1^{n+1}) \ .$$

By ignoring P_h, a centroid is computed: $\bar{P} = \frac{1}{n}\sum_{i \neq h} P_i$.

The goal now is to find a new point which has a lower cost than P_h in order to replace P_h. The search for such point is carried out based on 3 operations: reflection, expansion and contraction. Each operation is controlled by a distinct parameter.

1) Reflection Step: First, the reflection point is computed as:

$$P_r = (1+\alpha)\bar{P} - \alpha P_h \ ,$$

with $\alpha > 0$. The corresponding cost function value is also computed: $f_r = f(P_r)$.

2) Expansion Step: Then, if $f_r < f_l$, which means that P_r is the best point found so far, the expansion point and corresponding cost function value, f_e, are computed:

$$P_e = (1-\beta)\bar{P} + \beta P_r \ ,$$

with $\beta > 1$.

If $f_e < f_l$, then P_h is replaced by P_e, otherwise, it is replaced by P_r.

3) Contraction Step: Alternatively, if $\exists i \neq h : f_r \leq f_i$, then P_h is replaced by P_r.

Otherwise, P_h is replaced by P_r (only in case $f_r \leq f_h$) and the contraction point and corresponding cost function value, f_c, are computed:

$$P_c = \gamma P_h + (1-\gamma)\bar{P}$$

with $0 < \gamma < 1$.

Finally, if $f_c \leq f_h$, P_h is replaced by P_c; otherwise, the simplex vertexes are updated according to the following expression:

$$P_i = (1-\delta)P_l + \delta P_i \ ,$$

$\forall i \neq l$, with $0 < \delta < 1$.

After these simplex updates, the stopping criteria is checked. In case convergence is not achieved, all this procedure (starting on the l and h indexes computation) is repeated.

IV. Proposed Method

We tackle the initialization and parameter setting problem in CP-based link predictors using a two stage procedure. Briefly, in the first stage we estimate the CP parameters and initialization using optimization, combined with validation techniques, while in the second stage, we use the CP decomposition result to estimate the forecasting parameters.

The core of the first stage of the method resides on the application of optimization methods to carry out a task-driven search: the CP parameters, such as the number of components, are computed in order to maximize the task performance evaluation metric.

The idea of the second phase is to consider the time-series, which are defined according to the decomposition result, to drive the search for the adequate forecasting parameter values.

For simplicity, we refer to the resulting model as tCPLP, meaning tuned CP-based link predictor.

A. Stage 1: CP Parameters Estimation

The first stage is defined upon the assumption that the networks change smoothly. Based on such assumption, it is expected that the CP parameters, namely, the number of factors, that best model the data in such instants, do not change dramatically when we add a new timestamp (temporal slice). Thus, in this stage, we start by splitting the available timestamps into training and validation sets so that the validation set is formed by the number of instants corresponding to the prediction period we are interested in.

Then, given the set of non-numeric parameter values we want to cover (including the initializations), we generate all the possible combinations of such values so that we obtain a set of parameter and initialization combinations.

For each of such combinations, we apply an optimization method known as Nelder-Meads [7] (see Section III-B, for details on the method) to find the numeric parameters that maximize the model performance on the validation set. The optimization process covers both CP and forecasting numeric parameters. We note that fixed numeric forecasting parameters could be used in this stage, however, such approach could lead to biased CP parameters, that is, the CP parameters obtained could be too adjusted to a model with those forecasting parameters.

Given the models and their performance on the validation set, we select the CP parameters as the parameters of the model which achieved the highest performance. The forecasting parameters of such a model are discarded.

It is important to note that, since the application of Nelder-Mead to each combination of non-numeric parameters and initialization is independent, it can be carried out in parallel, thus, allowing a speed up on the procedure run time.

B. Stage 2: Forecasting Parameters Estimation

After obtaining the CP parameters, we proceed to the second stage. We start by joining the validation set with the previous training set, to obtain a new larger training set. Then, we apply CP to the new training set using the estimated CP parameters and use the temporal factor matrix to estimate the forecasting parameters.

In this work we only consider the Spiegel *et al.* model, referred to as CPES. The only forecasting parameter of this model is the smoothing factor. Since each column of the temporal factor matrix is interpreted as a time-series, as estimation method we propose a generalization of the traditional method used to estimate the smoothing factor in univariate time-series [8]. Thus, the smoothing factor is computed as the value minimizing the mean squared error of the forecasts across all the time-series (temporal factor matrix columns).

V. Experiments

The experiments were carried out using MATLAB along with Tensor Toolbox [9], [10] in a machine with 2.7GHz processor and 12GB RAM.

A. Datasets

We considered 4 time-evolving (directed) social networks in our experiments: Friends&Family [11], Enron [12], Reality Mining [13], Social Evolution [14].

The Enron dataset consists of an email exchange network in which there is a link from person i to person j at time t if person i sent at least an email to person j during such instant.

The remaining datasets are phone calls networks in which there is a link from person i to person j at time t if person i called person j at least once during such instant. These phone call datasets were submitted to a pre-processing phase in which we discarded all the calls involving individuals not under study at the time the data was collected. Moreover, we also discarded missing calls.

Each dataset was processed using 3 different time granularity levels so that at the end we obtained a daily, weekly and monthly version of the same network. Regardless of the periodicity considered, we did not model the weight of the link, in our case, the number of emails or calls (all networks used are unweighted). The datasets were organized in $people \times people \times time$ tensors. A summary is presented in Table II. The number of links by month registered in each dataset is presented in figure 1.

TABLE II
Datasets summary.

Network	Content	Periodicity	Size
Friends&Family [11]	Phone calls	Daily	$129 \times 127 \times 505$
		Weekly	$129 \times 127 \times 73$
		Monthly	$129 \times 127 \times 18$
Enron [12]	Email exchange	Daily	$184 \times 184 \times 1317$
		Weekly	$184 \times 184 \times 189$
		Monthly	$184 \times 184 \times 44$
Reality Mining [13]	Phone calls	Daily	$67 \times 68 \times 318$
		Weekly	$67 \times 68 \times 46$
		Monthly	$67 \times 68 \times 11$
Social Evolution [14]	Phone calls	Daily	$80 \times 78 \times 297$
		Weekly	$80 \times 78 \times 44$
		Monthly	$80 \times 78 \times 10$

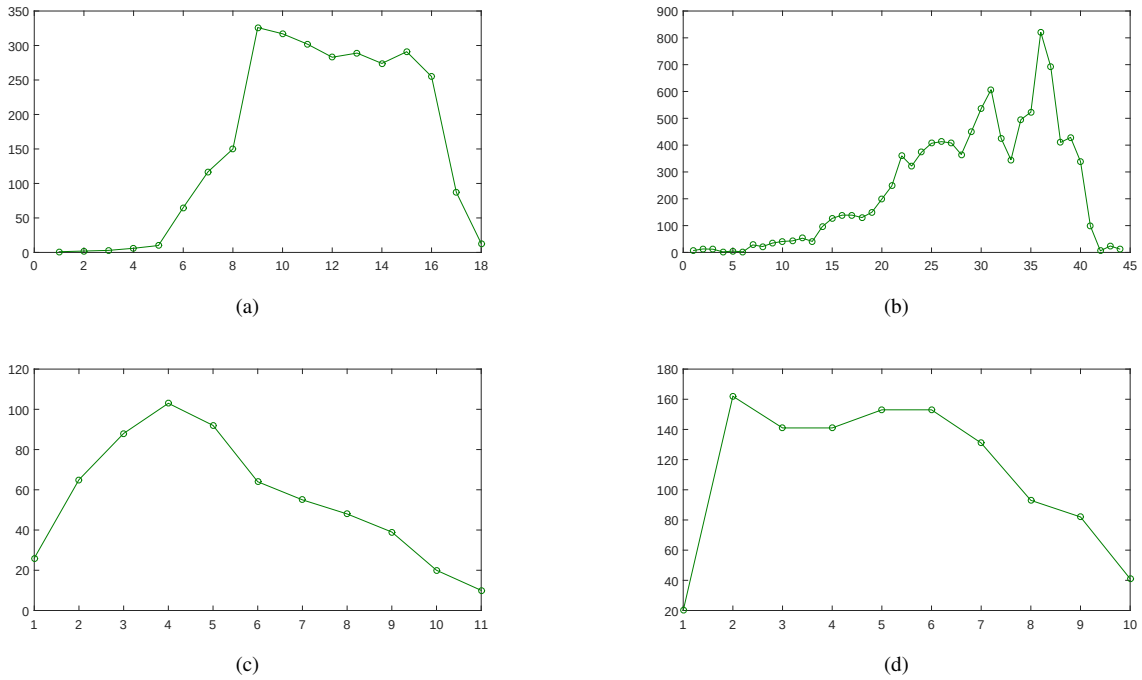

Fig. 1. Number of links by month on (a) Friends&Family dataset; (b) Enron dataset; (c) Reality Mining dataset; (d) Social Evolution dataset.

B. Baseline

As baseline, we considered Katz scores [15], [16], a method initially designed for static networks, which was later applied on time-evolving networks [1].

Katz score is a path-based node similarity measure. This type of measures takes into consideration paths with length greater than two. In particular, for a given static network, the Katz score between nodes v_1 and v_2 is given by:

$$s_{KS}(v_1, v_2) = \sum_{l=1}^{\infty} \beta^l |paths_{v_1,v_2}^{<l>}|$$

where $\beta \in (0, 1)$ and $|paths_{v_1,v_2}^{<l>}|$ is the number of length-l paths between v_1 and v_2. The idea of Katz score is to weight the paths according to their length so that paths with shorter length have a greater impact than the longer ones.

In the case of large networks, in order to reduce the computational complexity, a truncated version is recommended. In such variation, only the paths of length less (or equal) than an established value (L) are considered. In our work, we considered $\beta = 0.0005$ and $L = 4$ as suggested in [16], [17].

The extension of this method to a time-evolving network is carried out by first collapsing the network adjacency matrices into a single one and then computing the Katz scores based on the resulting matrix [1]. The collapsing technique employed consists of weightily summing the adjacency matrices so that the matrices associated with the most recent activity have more weight than the previous ones.

C. Evaluation Metric

The problem of link prediction may be interpreted as a classification task, where the classes are either "there is a link between the entities" or "there is no link between the entities" at a given time period.

Despite of being traditionally the most used evaluation metric, the area under receiver operator characteristic curve (AUCROC) may not be the most appropriate measure for assessing the link predictors performance in social networks, as pointed out by Yang *et al.* [18].

Social networks are usually very sparse and, in the context of link prediction, we are interested in predicting the presence of a link (not its absence). By considering AUCROC, we are considering predictions of type "there will be no link" as correct, which may bias the results given the amount of unlinked pairs in the network.

Thus, we considered in our experiments the area under precision-recall curve (AUCPR), as suggested in the work of Yang *et al.*.

D. Experimental Setting

In order to carry out our study, we needed to assess the quality of the CP-based link predictors in different scenarios. Thus, given the dataset and the time granularity, we split the timestamps into training and test sets so that if timestamp t was defined as test instant, then (i) the network state at time instants 1 up to $t-1$ were used for training, that is, for generating the models; (ii) the network state at instant t was used for assessing the models quality and (iii) the remaining timestamps, $t+1$ up to the last, were discarded.

TABLE III
AUCPR OF THE CPES MODELS OVER 10 DIFFERENT RANDOM CP INITIALIZATIONS.

Datasets	Periodicity	Mean	Min	Max	Max-Min
Friends&Family	Daily	0,100	0,083	0,124	0,040
	Weekly	0,647	0,636	0,663	0,027
	Monthly	0,653	0,636	0,662	0,027
Enron	Daily	0,188	0,134	0,233	0,099
	Weekly	0,398	0,379	0,430	0,051
	Monthly	0,378	0,361	0,393	0,032
Reality Mining	Daily	0,098	0,064	0,132	0,067
	Weekly	0,622	0,552	0,658	0,106
	Monthly	0,385	0,359	0,403	0,044
Social Evolution	Daily	0,662	0,632	0,683	0,051
	Weekly	0,439	0,399	0,469	0,069
	Monthly	0,360	0,314	0,398	0,085

TABLE IV
AUCPR OF THE CPES MODELS BEFORE (INITIAL MODEL) AND AFTER (FINAL MODEL - TCPLP) THE APPLICATION OF THE CPLP-TUNER.

Datasets	Periodicity	Initial Model	Final Model
Friends&Family	Daily	0,045	0,296
	Weekly	0,658	0,793
	Monthly	0,658	0,703
Enron	Daily	0,255	0,204
	Weekly	0,386	0,340
	Monthly	0,365	0,332
Reality Mining	Daily	0,110	0,087
	Weekly	0,511	0,604
	Monthly	0,378	0,480
Social Evolution	Daily	0,434	0,552
	Weekly	0,475	0,482
	Monthly	0,384	0,487

TABLE V
AUCPR OF THE TCPLP MODELS, THE BASELINE AND RANDOM PREDICTOR.

Datasets	Periodicity	tCPLP	Katz	Random
Friends&Family	Daily	0,296	**0,445**	0,002
	Weekly	**0,793**	0,734	0,010
	Monthly	0,703	**0,734**	0,018
Enron	Daily	**0,204**	0,101	0,001
	Weekly	0,340	**0,401**	0,005
	Monthly	**0,332**	0,328	0,013
Reality Mining	Daily	**0,087**	0,083	0,002
	Weekly	0,604	**0,660**	0,003
	Monthly	0,480	**0,602**	0,009
Social Evolution	Daily	**0,552**	0,439	0,002
	Weekly	0,482	0,482	0,006
	Monthly	0,487	**0,523**	0,015

E. Results

We organized our results in terms of research questions. The research questions we addressed in this work were:

1) RQ1: Does the initialization of CP influence the performance of the CP-based link predictors?
2) RQ2: Do the tCPLP models outperform the initial (untuned) models?
3) RQ3: Is the performance of the tCPLP models competitive when comparing to Katz method?

In order to address RQ1, we fixed all model parameters and varied the initialization. In this setting, for each dataset, we considered a parameter combination such that the CP decomposition had 30% of fitting and a smoothing factor of 0.5. Since few links were registered in the last months of the datasets (see figure 1), we discarded them. The daily and weekly versions of the datasets were also truncated in the same manner. In particular, we discarded the last 3 months of Friends& Family, the last 5 months of Enron and the last 2 of the remaining datasets. Results are exhibited in Table III.

In this set of experiments, we verified that the models performance depended on the initialization. However, different levels of variability in the models performance were observed. There were cases in which the performances were relatively stable, as it was the case of the models applied to the Friends&Family dataset. Nonetheless, we registered a performance oscillation of ≈ 0.1 between the best and the worst models in some scenarios, namely in the daily Enron and the weekly Reality Mining datasets. These results emphasize the need of taking into consideration the initialization as a possible source of variability in the models under study.

With the purpose of understanding if we could improve the models performance, we proceeded with the experiments by applying the proposed method to each of the datasets, thus, addressing RQ2. We opted to eliminate the randomness sources of our models by considering only the SVD initialization. Results are presented in Table IV.

We observed that in the majority of the scenarios, the tCPLP model outperformed the initial model. On average, the improvement of the tCPLP models performance over the initial models was ≈ 0.11. The smallest performance improvement was ≈ 0.01 in the weekly version of the Social Evolution dataset while the largest improvement was ≈ 0.25 registered on the daily version of Friends&Family.

Regarding the other scenarios, in which the parameter setting procedure failed, we verified that the maximum performance loss registered in such scenarios was ≈ 0.05 in the daily setting of Enron. We investigated these scenarios and verified that in the daily versions of Enron and Social Evolution, there were at most 6 links in the validation timestamp. Thus, since the number of links was small, a small change in the score matrix may have led to a great change in the performance of the model. Such issue may have compromised the parameter search. In the case of the monthly version of Enron, there was a great change in the network topology from the validation set to the test set: the number of links was similar in both instants but the majority of the links observed in the test set were not observed in the validation set.

When we compared the previous tCPLP models with the baseline (Katz) (see table V), we verified that the tCPES exhibited higher (or equal) performance than Katz in 50% of the scenarios. However, the tCPLP models performed always better than random.

Thus, with the goal of understanding how the models obtained by the CPLP-tuner perform in other scenarios, we addressed RQ3 by considering several test sets. Since the random predictor performance is given by the rate of links in the test set and, as we observed in figure 1, the networks are

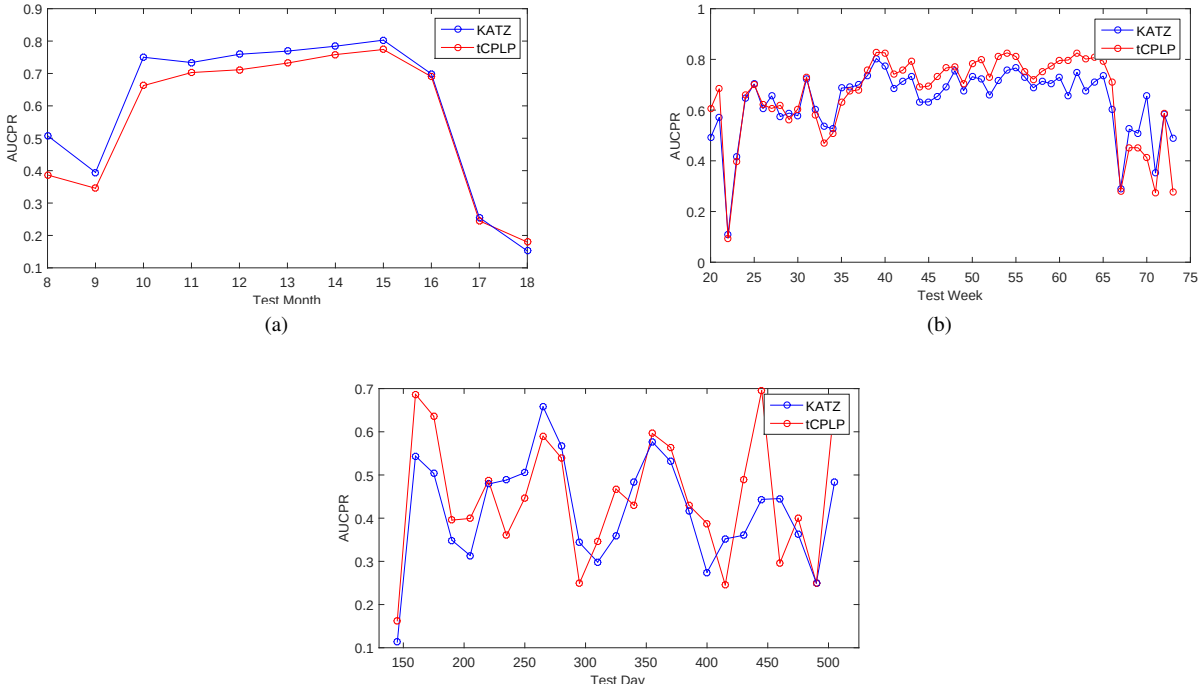

Fig. 2. Performance of the baseline (Katz) and the tCPLP models in each test instant on: (a) monthly; (b) weekly; and (c) daily versions of Friends&Family dataset.

very sparse, it assumes extremely low values whereby we did not consider the random predictor in this set of experiments.

Given a timestamp, the idea was to consider all the previous timestamps as training set; apply CPLP-tuner on the training set and evaluate the model on the given timestamp. The results are exposed on figures 2-5. Once more, for each dataset, we considered different degrees of time granularity. Due to the heterogeneity in the number of timestamps and in the distribution of the number of links by time slice, the number of test sets used depended on both the dataset and the periodicity. Moreover, in the case of the daily setting, we considered test days spaced by 15 days.

Regarding the Friends&Family dataset (figure 2), when we considered monthly data, we observed that both models, tCPLP and Katz, exhibited similar performance values, however, the tCPLPs were usually less accurate in their predictions. In particular, we observed a decrease in the second test set (month 9), which may be associated with the large amount of change. We further investigated and found that the number of new links appearing in such month was near 3 times larger than the following maximum registered. The decrease on the number of links to less than half observed in month 17 was also associated with a performance decrease which may be due such a change in the network. An analogous behavior was observed in the last test month.

With respect to the week setting of this dataset, we observed that the models generated by our method outperformed the baseline in almost all test sets considered. In the last weeks, we observed a performance decrease which, once again, corresponded to weeks with few links.

Finally, when we considered daily granularity, the CP-based models obtained by CPLP-tuner outperformed the baseline in $\approx 67\%$ of the test days. Katz outperformed the tCPLP models mainly on the test days belonging to months 7 to 10, which corresponded to a strong increase in the number of links (see figure 1a).

Similarly to what was observed in the Friends &Family monthly dataset, both methods exhibited identical performance in the monthly version of Enron dataset. When considering this setting of Enron, the tCPLP models outperformed the baseline in only $\approx 27\%$ of the test sets. Moreover, we observed that the low performance peak at month 33 was associated with a low peak in the number of links. This behavior was also observed on month 43, however, such peak was followed by a performance increase eventually due to the reduced change observed in the last 3 months (see figure 1b).

By considering week time granularity on the Enron dataset, we observed that tCPLP outperformed the baseline in $\approx 50\%$ of the test weeks. We also observed that the time periods in which tCPLP was outperformed by the baseline corresponded to low performance peaks in both methods, for example, weeks 137 to 140 and 161 to 167.

When we considered the daily setting of this dataset, a highly variant performance evolution was observed. In particular, we observed 7 zero performance peaks, which we further investigated and verified that they corresponded to time slices

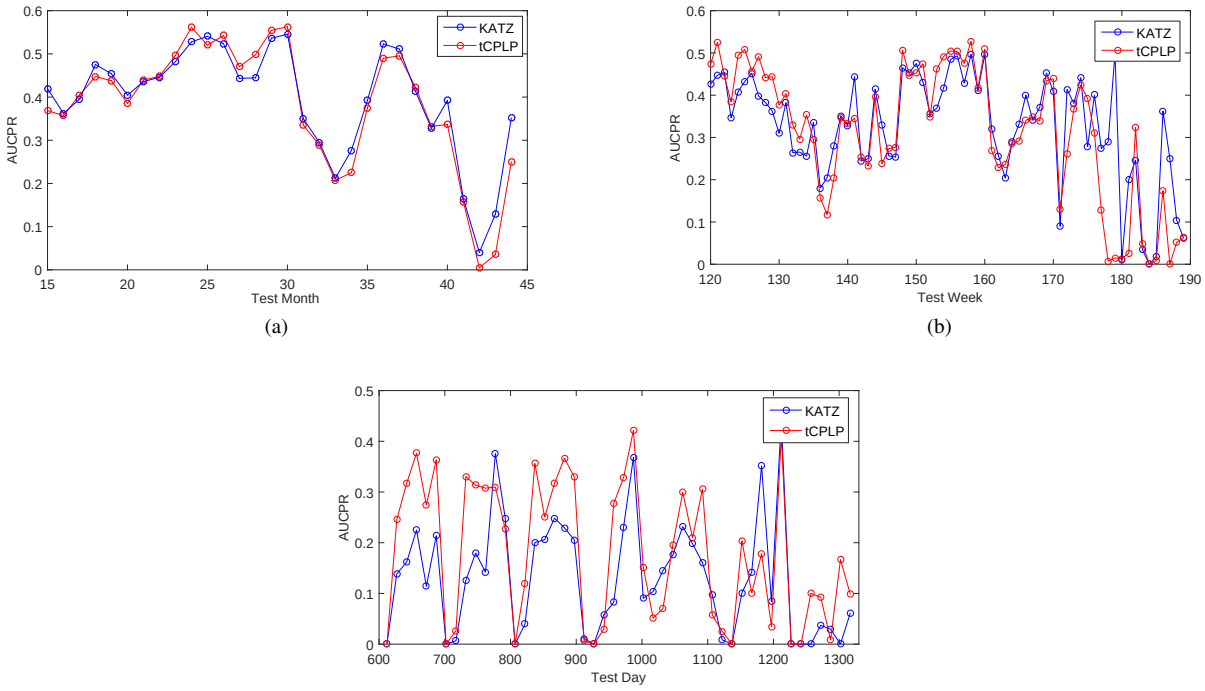

Fig. 3. Performance of the baseline (Katz) and the tCPLP models in each test instant on: (a) monthly; (b) weekly; and (c) daily versions of Enron dataset.

with a number of links between 0 and 2. Regarding the remaining day test sets, tCPLP outperformed the baseline in $\approx 70\%$ of the test sets. In most of which cases, tCPLP performance was considerably higher than the baseline. Another interesting observation was that the performance of the tCPLP models in this setting had a near periodical evolution.

With respect to the Reality Mining data, when we considered the month granularity, tCPLP outperformed the baseline in 50% of the test sets.

By considering weekly timestamps, such rate increased to $\approx 65\%$ of the test sets considered, while in the daily setting, if we restrict our analysis to the first 12 test days, such rate attained 75%. Regarding the last 3 test days, we verified that (i) the zero performance peaks (at days 288 and 303) were associated to days in which no links occurred and (ii) there were no links in the last validation set so that it was not possible to train the model; thus, the CP-model parameters were not tuned, which justifies the poor performance of the model in that test instant.

Concerning the last dataset, Social Evolution, we verified that the behavior of the models in the monthly setting was identical to the ones observed in the monthly settings of the first two datasets: the performance evolution of the 2 methods over time was similar, however, the tCPLP models exhibited always less (or equal) accuracy.

When we considered weekly periodicity, the tCPLP models outperformed the baseline in 40% of the test weeks. In the case of the daily setting, there were $\approx 85\%$ of test days in which the tCPLP models outperformed the baseline.

In general terms, based on the analysis of each setting, we observed that the tCPLP models achieved a better performance, when comparing to the baseline, as we refined the timestamps, from months to days.

Given a time-evolving network, the time refinement is usually associated with more time slices, more sparsity and more dynamics. Thus, further investigation, namely, a more extensive study, should be carried out in order to understand how such factors influence the performance of the CP-based link predictors.

VI. Conclusion

In this work we showed that both the parameters and the initialization have impact on the CP-based link predictors performance. Such impact is critical since it arises the problem of, on the one hand, choosing the appropriate parameters values and, on the other hand, guaranteeing the reproducibility of the results: different initialization may lead to different results and, consequently, initialization should be reported so that replicability can be ensured.

In order to tackle such problem, we proposed CPLP-tuner, a method for estimating the parameters and initialization of the CP-based link predictors. The proposed approach has the advantages that (i) it is easily generalizable for other CP and forecasting algorithms and (ii) it is parallelizable.

We applied the proposed method and studied the performance of the resulting models in different scenarios. We verified that the CP-based link predictors obtained using CPLP-tuner exhibited competitive performance, especially when considering more refined timestamps.

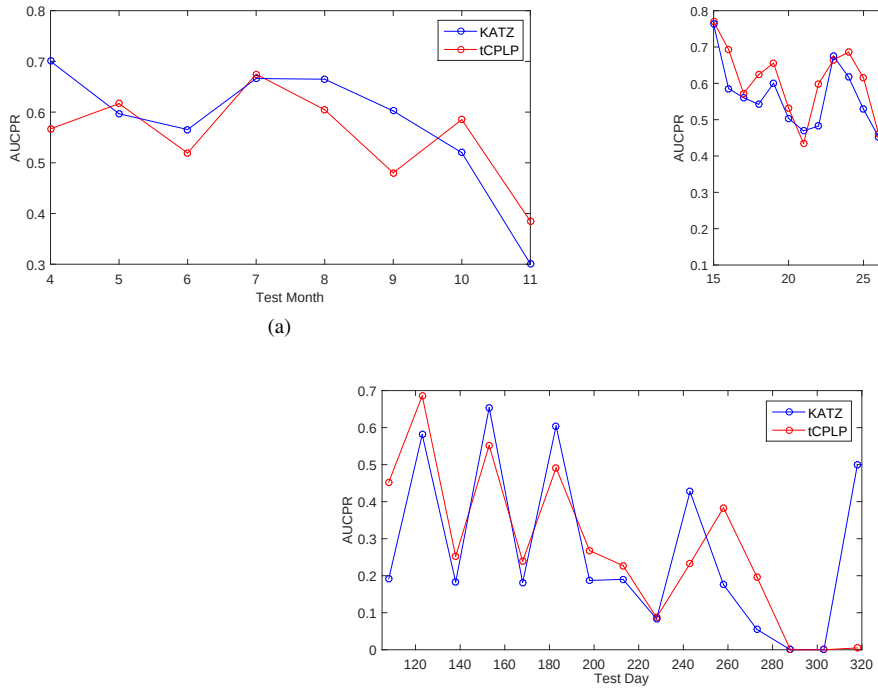

Fig. 4. Performance of the baseline (Katz) and the tCPLP models in each test instant on: (a) monthly; (b) weekly; and (c) daily versions of Reality Mining dataset.

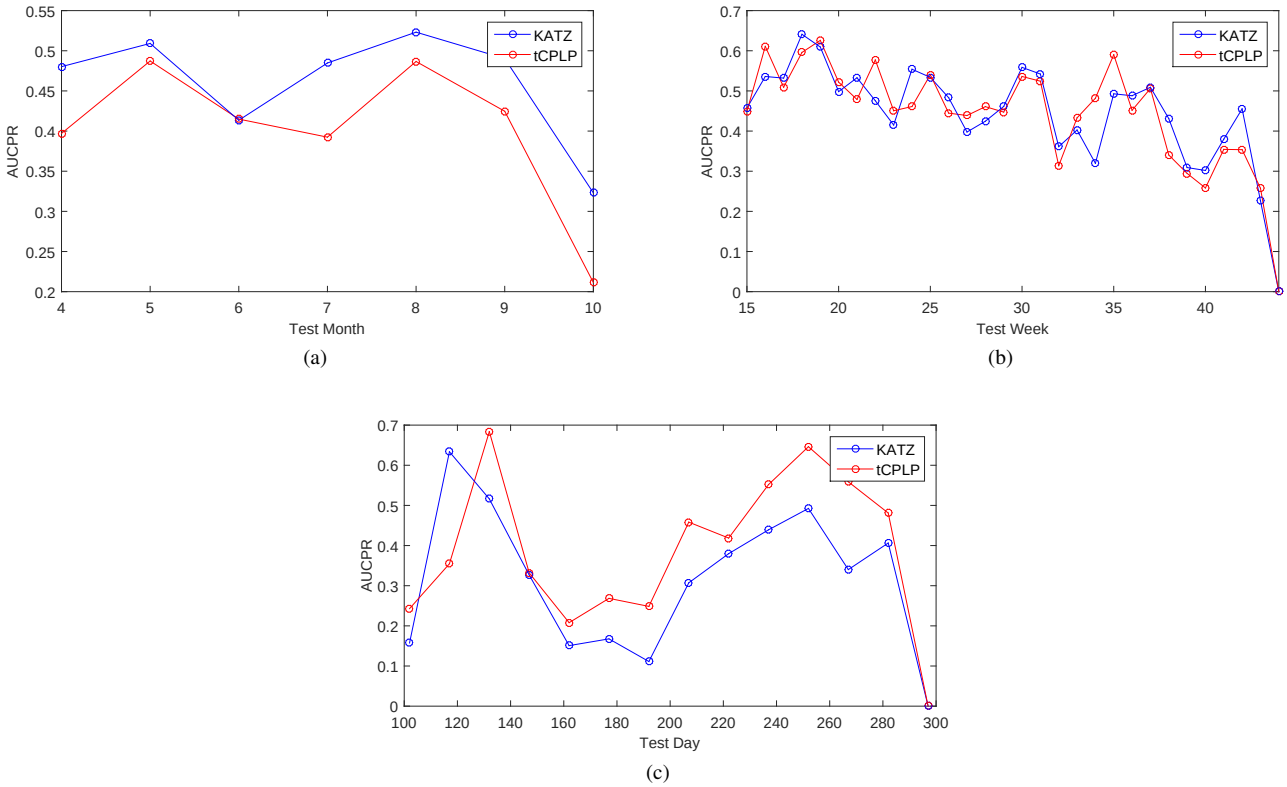

Fig. 5. Performance of the baseline (Katz) and the tCPLP models in each test instant on: (a) monthly; (b) weekly; and (c) daily versions of Social Evolution dataset.

Future work will include, at a first stage, the extension of the study to other larger networks. Another direction we are interested in is studying what are the most critical parameters of the models, that is, the ones that have more impact on the performance of the model so that we can improve the proposed method.

REFERENCES

[1] D. M. Dunlavy, T. G. Kolda, and E. Acar, "Temporal link prediction using matrix and tensor factorizations," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 5, no. 2, p. 10, 2011.

[2] S. Spiegel, J. Clausen, S. Albayrak, and J. Kunegis, "Link prediction on evolving data using tensor factorization," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2011, pp. 100–110.

[3] H. A. Kiers, "Towards a standardized notation and terminology in multiway analysis," *Journal of chemometrics*, vol. 14, no. 3, pp. 105–122, 2000.

[4] R. Bro and H. A. Kiers, "A new efficient method for determining the number of components in parafac models," *Journal of chemometrics*, vol. 17, no. 5, pp. 274–286, 2003.

[5] E. E. Papalexakis, "Automatic unsupervised tensor mining with quality assessment," in *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM, 2016, pp. 711–719.

[6] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM REVIEW*, vol. 51, no. 3, pp. 455–500, 2009.

[7] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.

[8] E. S. Gardner, "Exponential smoothing: The state of the artpart ii," *International journal of forecasting*, vol. 22, no. 4, pp. 637–666, 2006.

[9] B. W. Bader, T. G. Kolda *et al.*, "Matlab tensor toolbox version 2.6," Available online, February 2015. [Online]. Available: http://www.sandia.gov/~tgkolda/TensorToolbox/

[10] B. W. Bader and T. G. Kolda, "Efficient MATLAB computations with sparse and factored tensors," *SIAM Journal on Scientific Computing*, vol. 30, no. 1, pp. 205–231, December 2007.

[11] N. Aharony, W. Pan, C. Ip, I. Khayal, and A. Pentland, "Social fMRI: Investigating and shaping social mechanisms in the real world," *Pervasive and Mobile Computing*, vol. 7, no. 6, pp. 643–659, 2011. [Online]. Available: http://realitycommons.media.mit.edu/friendsdataset.html

[12] C. E. Priebe, J. M. Conroy, D. J. Marchette, and Y. Park, "Scan statistics on enron graphs," *Computational & Mathematical Organization Theory*, vol. 11, no. 3, pp. 229–247, 2005. [Online]. Available: http://cis.jhu.edu/~parky/Enron/

[13] N. Eagle and A. S. Pentland, "Reality mining: sensing complex social systems," *Personal and ubiquitous computing*, vol. 10, no. 4, pp. 255–268, 2006.

[14] A. Madan, M. Cebrian, S. Moturu, K. Farrahi *et al.*, "Sensing the" health state" of a community," *IEEE Pervasive Computing*, vol. 11, no. 4, pp. 36–45, 2012.

[15] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.

[16] D. Liben-Nowell and J. Kleinberg, "The Link Prediction Problem for Social Networks," *Proceedings of the Twelfth Annual ACM International Conference on Information and Knowledge Management (CIKM)*, pp. 556–559, 2003.

[17] C. Wang, V. Satuluri, and S. Parthasarathy, "Local probabilistic models for link prediction," in *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*. IEEE, 2007, pp. 322–331.

[18] Y. Yang, R. N. Lichtenwalter, and N. V. Chawla, "Evaluating link prediction methods," *Knowledge and Information Systems*, vol. 45, no. 3, pp. 751–782, 2015.