TensorCast: Forecasting Time-Evolving Networks with Contextual Information

Miguel Araújo[1,2], **Pedro Ribeiro**[1] and **Christos Faloutsos**[2]
[1] University of Porto and CRACS/INESC-TEC
[2] School of Computer Science, Carnegie Mellon University
miguelaraujo.cs@gmail.com, pribeiro@dcc.fc.up.pt, christos@cs.cmu.edu

Abstract

Can we forecast future connections in a social network? Can we predict who will start using a given hashtag in Twitter, leveraging contextual information such as who follows or retweets whom to improve our predictions? In this paper we present an abridged report of TENSORCAST, a method for forecasting time-evolving networks, that uses coupled tensors to incorporate multiple information sources. TENSORCAST is *scalable* (linearithmic on the number of connections), *effective* (more precise than competing methods) and *general* (applicable to any data source representable by a tensor). We also showcase our method when applied to forecast two large scale heterogeneous real world temporal networks, namely Twitter and DBLP.

1 Introduction

If a group has been discussing the #elections on Twitter, can we predict who is going to join the discussion next week? Intuitively, we should take into account not only other used hashtags (#), but also user to user interactions such as followers and retweets. Similarly, can we forecast who is going to publish on a given conference next year? We should consider not only previous conferences where the author published, but also user to user information such as co-authorships.

Today's data sources are often heterogeneous, characterized by different types of entities and relations that we should leverage in order to enrich our datasets. We propose to model these heterogeneous evolving graphs as Coupled Tensors that, jointly, generate better predictions than when considered independently. In particular, we will show how the evolution of user to user connections can be leveraged to forecast user to entity relations, e.g. information about who retweets whom improves the prediction of who is going to use a given hashtag, and co-authorship information improves the prediction of who is going to publish at a given venue.

Using a *naive* approach, one would have to individually forecast every pair of users and entities - a prohibitively big number that quadratically explodes. How can we avoid this? How can we obtain the top most likely interactions without iterating trough them all?

We propose TENSORCAST, a tensor forecasting method that is able to merge multiple data sources in order to predict future membership relations. Our method combines a non-negative coupled factorization of generalized tensors with an efficient algorithm that is able to obtain the top-K records from an existing factorization, without reconstructing the full tensor. We achieve over 20% higher precision in top-1000 queries and double the precision when finding new relations than comparable alternatives and our method scales well (linearithmically) with the input size (we show experimental results in graphs with over *300M* connections).

TENSORCAST is open-source and can be obtained at `http://www.dcc.fc.up.pt/~pribeiro/tensorcast/`
This is abridged version of a longer paper which got the ICDM'2017 best paper award [Araujo *et al.*, 2017].

2 Background and Related Work

Notation. As commonly used, we denote vectors by boldface lowercase letters (e.g., $\boldsymbol{a}$), matrices by boldface uppercase letters (e.g., $\boldsymbol{A}$) and tensors by boldface caligraphic letters (e.g., $\boldsymbol{\mathcal{X}}$). We refer to the f-th column of $\boldsymbol{A}$ as $\boldsymbol{a_f}$ and to the (i, j, k) entry of 3-mode tensor $\boldsymbol{\mathcal{X}}$ as $\boldsymbol{\mathcal{X}}_{ijk}$. $\boldsymbol{M^t}$ denotes a mtriz transpoe, $\|\boldsymbol{\mathcal{X}}\|_F$ denotes the Frobenius norm of tensor $\boldsymbol{\mathcal{X}}$, and $\boldsymbol{\mathcal{X}}$ its mode-k matricization. The vector outer product is denoted by $\circ$ and the Khatri-rao product by $\odot$.

2.1 Tensor Factorizations

Tensors are multidimensional arrays that generalize the concept of matrices, and are a popular choice when representing time-evolving relations, such as Facebook interactions [Papalexakis *et al.*, 2012] or sensor networks [Sun *et al.*, 2006]. When properly applied, tensor factorizations identify the underlying low-dimensional latent structure of the data, which can be used for tasks such as identifying anomalies or estimate missing values. The PARAFAC [Harshman, 1970] (also called CP) decomposition is one of the most popular among the many tensor factorizations flavors [Kolda and Bader, 2009], as it factorizes a tensor into a sum of rank-1 tensors. In three modes, the problem is usually framed as finding factor matrices $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ that minimize the squared error between $\boldsymbol{\mathcal{X}}$ and the reconstructed tensor: $\min_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}} \left\| \boldsymbol{\mathcal{X}} - \sum_f \boldsymbol{a_f} \circ \boldsymbol{b_f} \circ \boldsymbol{c_f} \right\|_F^2$.

2.2 Coupled Factorizations

We are often interested in analyzing real-world tensors when additional information is available from distinct sources. For example, in a simple recommendation task with user×movie ratings, we might have user demographics data available which we wish to incorporate when predicting future ratings.

Coupled Matrix-Tensor Factorizations and Coupled Tensor-Tensor Factorizations are a natural extension to the standard tensor factorization formulation. For instance, the factorization of a third-order tensor $\boldsymbol{\mathcal{X}}$ coupled with a matrix $\boldsymbol{M}$ on its first mode can be obtained by minimizing

$$\min_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D}} \left\|\boldsymbol{\mathcal{X}} - \hat{\boldsymbol{\mathcal{X}}}\right\|_F^2 + \alpha \left\|\boldsymbol{M} - \hat{\boldsymbol{M}}\right\|_F^2 \qquad (1)$$

where α is a parameter representing the strength of the coupling, i.e., how important $\boldsymbol{M}$ is to improve the prediction.

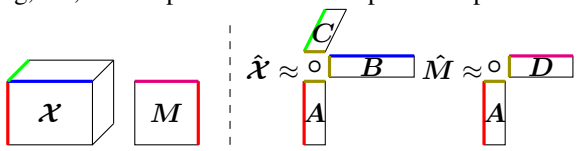

Figure 1: A simple Coupled Matrix-Tensor Factorization.

The matrix part of the Coupled Matrix-Tensor Factorization depicted in Fig. 1 is useful to model additional static information about one of the modes of the tensor of interest. Whenever the side information available is time-evolving, a model where two tensors are coupled along one of the dimensions is more appropriate, preserving the time component:

$$\min_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{T}} \left\|\boldsymbol{\mathcal{X}} - \hat{\boldsymbol{\mathcal{X}}}\right\|_F^2 + \alpha \left\|\boldsymbol{\mathcal{Y}} - \hat{\boldsymbol{\mathcal{Y}}}\right\|_F^2 \qquad (2)$$

where $\hat{\boldsymbol{\mathcal{X}}} = \sum_f \boldsymbol{a_f} \circ \boldsymbol{b_f} \circ \boldsymbol{t_f}$ and $\hat{\boldsymbol{\mathcal{Y}}} = \sum_f \boldsymbol{a_f} \circ \boldsymbol{c_f} \circ \boldsymbol{t_f}$

Many techniques have been proposed to solve this non-negative optimization problem, such as projected Stochastic Gradient Descent (SGD) [Beutel *et al.*, 2014] (i.e., additive update rules) and multiplicative update rules. Most of this work extends Lee and Seung's multiplicative matrix updates formulae [Lee and Seung, 2001] for matrices, notably the simple extension for tensors [Welling and Weber, 2001] and the many coupled extensions, e.g. Generalized Tensor Factorization [Yılmaz, 2012; Şimşekli *et al.*, 2013].

2.3 Skewed Building Blocks

When factorizing real-life graphs, non-negative factors scores are not uniformly distributed but decrease sharply. For instance, it has been shown that the internal degree distribution of big communities can be well approximated by a power-law [Araujo *et al.*, 2014] and that eigenvectors of Kronecker graphs exhibit a multinomial distribution [Leskovec *et al.*, 2005]. TENSORCAST leverages this property in order to speed-up its computation of Top-K elements without reconstructing the forecasted *tensor*.

2.4 Top-K Elements in Matrix Products

Given the widespread applications of matrix factorizations, finding the top-K elements of a matrix product is an important problem which can be stated as: given matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ of sizes $N \times F$ and $M \times F$, find the top-K (i, j) pairs of the $\boldsymbol{AB^t}$ matrix product. A *naive* solution requires $O(NMF)$ operations, iterating over the (originally) implicitly defined reconstruction matrix. Some attention has been given to this problem, since Ram and Gray [Ram and Gray, 2012] proposed the use of Cone Trees to speed-up this search,but this is a non-convex optimization problem in general.

2.5 Link Prediction

In the classical *structural* link prediction problem [Liben-Nowell and Kleinberg, 2007] the goal is to predict which links are more likely to appear, assuming that links are never or seldom removed. With time-evolving graphs, *temporal* link prediction methods have been developed to predict future snapshots, where links are not guaranteed to persist over time. In this setting we distinguish methods that rely on collapsing (matricizing) the input data (e.g., exponential decay of edge weights [Sharan and Neville, 2008]) from methods that deal directly with the increased dimensionality, such as tensor-based methods. CP Forecasting [Dunlavy *et al.*, 2011] finds a low-rank PARAFAC factorization and forecasts the time-component. TriMine [Matsubara *et al.*, 2012] similarly factorizes the input tensor, but then applies probabilistic inference in order to identify hidden topics. These methods are not able to integrate contextual information. Other approaches integrate structure and content in the same prediction task, e.g. Gao et al [Gao *et al.*, 2011] suggest a coupled matrix factorization and graph regularization technique to obtain the latent factors after an exponential decay of the temporal network. Another approach is to use time series analysis and regression based methods such as VAR [Zellner, 1962] or ARIMA [Box and Pierce, 1970].

However, none of these methods fulfills all the requirements for forecasting when contextual information is considered. Table 1 contrasts TENSORCAST against the state of the art competitors on key specs: (a) linear **scalability** with sparse data; (b) **interpretability** of the underlying model; (c) **time-awareness** for forecasting periodic, growing and/or decaying relations; (d) ability to deal with additional **contextual information**; (e) the ability to **forecast** the disappearance of existing relations; and (f) the ability of providing an **ordered** ranking of future events by likelihood of occurrence.

Property	Truncated SVD	Truncated Katz	Coupled Matrices	VAR, ARIMA	CP Forecasting	TriMine	TENSORCAST
Scalability	✓	✓	✓		✓	✓	✓
Interpretability	✓	✓	✓		✓	✓	✓
Coupled setting			✓				✓
Time-awareness				✓	✓	✓	✓
Context-awareness			✓				✓
Forecasting				✓	✓	✓	✓
Ordered Forecasting				✓	✓		✓

Table 1: TENSORCAST integrates context and time-awareness.

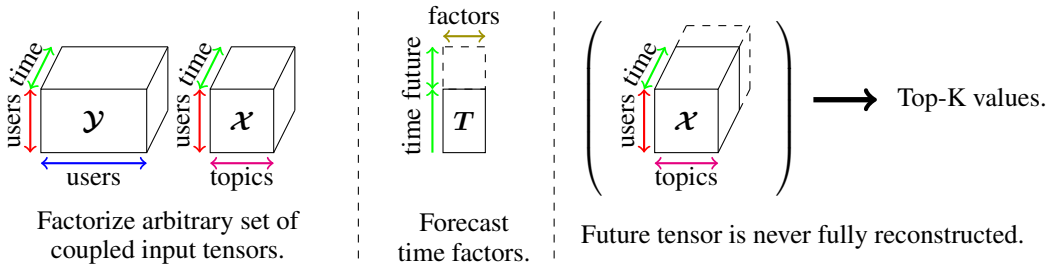

Figure 2: Overview of TENSORCAST.

3 Proposed Approach: TENSORCAST

We assume a coupled-tensors setting where multiple tensors are related by a common mode. One of these tensors should be a 3-dimensional binary tensor, with one mode corresponding to a time component which we would like to forecast. There are many scenarios that fall under this setting. For example, we could have a tensor of membership records *(user, topic, time)* and an additional collection of user interaction records *(user, user, time)*, with a possible forecasting problem of predicting which users will interact with which topics, taking advantage of the both sources of information.

We tackle the following general tensor forecasting problem: given coupled tensors ($\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$), a number of K relations and S time-steps, **forecast** the ranked list of K most likely non-zero elements of $\boldsymbol{\mathcal{X}}$ in the next S time-steps .

Overview. TENSORCAST is comprised of three successive steps, described in more detail in the following subsections: (1) **Non-negative Coupled Factorization:** the factorization will tie together the various input tensors and identify their rank-1 components.; (2) **Forecasting:** given the low dimensional space identified, we use standard techniques to forecast the time component; (3) **Top-K elements:** we exploit the factorization structure and identify the top elements without having to reconstruct the prohibitively big future tensor. Figure 2 illustrates the intuition of our method.

3.1 Non-negative Coupled Factorization

Consider that the tensor of interest, $\boldsymbol{\mathcal{X}}$, is a 3-dimensional $N\times M\times T$ dataset and that the time component corresponds to the last index of the tensor. Then, naively, the number of elements to be forecasted ($S\times N\times M$) is a prohibitive number when we consider $\boldsymbol{\mathcal{X}}$ to be big and sparse. Factorizing the input data reduces the number of elements to be forecasted, but also, and more importantly, it co-clusters similar elements together enabling generalization. A careful factorization will allow the forecast of previously unseen relations. We opted for a non-negative coupled factorization in order to improve the interpretability of the model.

We explore how user interactions can be leveraged to improve forecasting of user-entity relations. We model the problem as two coupled tensors where tensor $\boldsymbol{\mathcal{Y}}$ is a $N\times N\times T$ symmetric tensor. In order to guarantee convergence, we modify the update of the symmetric factor matrix to:

$$\boldsymbol{A} \leftarrow \boldsymbol{A} \otimes \sqrt[3]{\frac{\boldsymbol{\mathcal{X}}_{(1)}(\boldsymbol{B}\odot\boldsymbol{T}) + \alpha\boldsymbol{\mathcal{Y}}_{(1)}(\boldsymbol{A}\odot\boldsymbol{T})}{\boldsymbol{A}(\boldsymbol{B}\odot\boldsymbol{T})^t(\boldsymbol{B}\odot\boldsymbol{T}) + \alpha\boldsymbol{A}(\boldsymbol{A}\odot\boldsymbol{T})^t(\boldsymbol{A}\odot\boldsymbol{T})}}$$

3.2 Forecasting

Let $\boldsymbol{T}$ be the $T\times F$ factor matrix (time component) obtained from the previous step. It consists of a small set of F dense factor vectors, hence easy to forecast, that will provide an approximation $\hat{\boldsymbol{\mathcal{X}}}$ of the next time-step. The most appropriate forecasting mechanism is data-dependent. We forecast using basic exponential smoothing (Holt's method), but other methods can be applied, e.g. Holt-Winters double exponential smoothing when seasonality is present.

3.3 Tensor Top-K elements

The forecast of the next time-step is a $N\times M\times S$ tensor represented as $\sum \boldsymbol{a_f}\circ\boldsymbol{b_f}\circ\boldsymbol{s_f}$ where $\boldsymbol{A}$ is $N\times F$, $\boldsymbol{B}$ is $M\times F$ and $\boldsymbol{S}$ is $S\times F$. We extend the literature on the retrieval of maximum entries in a matrix product to the tensor case, leveraging the fact that the factorization was performed on non-random graph that follows typical properties. The goal is to identify the K (i,j,k) positions with highest value $\sum \boldsymbol{A}_{if}\boldsymbol{B}_{jf}\boldsymbol{S}_{kf}$.

Top-K of a single factor. We start by creating a data structure to get the next biggest element in $O(\log(SM))$ time, with $O(S\log S + M\log M + N\log N + SM)$ preprocessing. Firstly, we sort the vectors ($\boldsymbol{s}$, $\boldsymbol{a}$ and $\boldsymbol{b}$) in decreasing order. We now know that the biggest element is given by $\boldsymbol{a}_1\boldsymbol{b}_1\boldsymbol{s}_1$, and that an element $\boldsymbol{a}_i\boldsymbol{b}_j\boldsymbol{s}_k$ only needs to be considered after $\boldsymbol{a}_{i-1}\boldsymbol{b}_j\boldsymbol{s}_k$, $\boldsymbol{a}_i\boldsymbol{b}_{j-1}\boldsymbol{s}_k$ and $\boldsymbol{a}_i\boldsymbol{b}_j\boldsymbol{s}_{k-1}$ have all been identified as one of the biggest K. Hence, we can create a priority queue which holds, at most, $O(SM)$ elements at a time.

Combining multiple factors. The major hurdle is handling the interaction between multiple factors. We propose a greedy Top-K selection algorithm that is efficient under realistic scenarios. We keep a list ($\boldsymbol{R}$) of the K biggest positions evaluated so far. In each iteration, we consider the element with the highest score in one of the factors and add it to the list after evaluating it across all the factors. We terminate when the sum of the next best scores on each factor becomes smaller than the K^{th} biggest element in $\boldsymbol{R}$.

With this, we check at most $KSF^{1+\frac{1}{\alpha}}$ elements if every frontal slice $\mathbf{a_f}\circ\mathbf{b_f}$ follows a power-law (see [Araujo *et al.*, 2017] for a detailed description of the algorithm and respective proofs). This means TENSORCAST is linear on the number of elements we want to obtain times the number of time-steps forecasted and agrees with intuition: sharper (i.e., quickly decreasing, higher exponent) power-laws require less elements to be checked, while near-clique factors imply lower exponents and more elements to be analyzed. Empirical evaluation on real datasets confirmed our bounds.

Users	Groups	Timesteps	Memberships	Interactions	Description
1 734 902	5 476	79	8 049 559	21 423 244	DBLP - venues published and co-authorships.
12 426 133	2 326 843	31	30 281 817	282 280 158	TWITTER - hashtags used and retweets.

Table 2: Summary of real-world networks used.

4 Experiments

TENSORCAST is tested on two big datasets detailed in Table 2 In DBLP we have a tensor of authors and venues in which they published from 1970 to 2014, while the co-authorship tensor is used as contextual information. Evaluation is performed on the 2015 data. In TWITTER, the main tensor relates users and hashtags they used from June to December 2009, while retweets are used as the auxiliary tensor. Tweets are grouped weekly and evaluation is performed on week 51.

Every factorization uses 10 factors (both ours and competing methods). In TWITTER, we weighted the reconstruction of the main tensor 20x more relevant than the context tensor, and in DBLP 2.66x higher (so that both tensors have the same reconstruction error when considering empty factors).

How fast is TENSORCAST? We start by evaluating our method's scalability in TWITTER when changing the number of non-zeros in both tensors. By changing the number of weeks under consideration, we create a sequence of pairs of tensors that increase in size. For each pair, we measure wall-clock time when performing a rank-4 coupled tensor factorization, forecasting and identification of the top-1000 forecasted non-zeros. Figure 3 shows TENSORCAST's linear scalability. Similar results were obtained for DBLP.

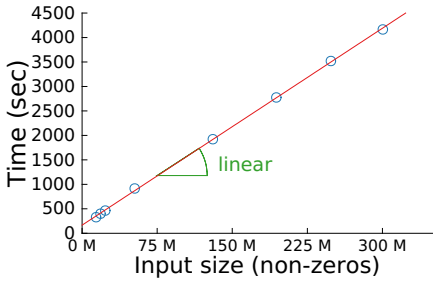

Figure 3: TENSORCAST scales linearly with the number of non-zeros in the TWITTER dataset.

How effective is TENSORCAST? Figure 4 showcases TENSORCAST's accuracy on the task of predicting top-K relations on future time steps, as we increase K on the DBLP dataset (similar results were obtained for TWITTER).

Note the importance of TENSORCAST's ability of being simultaneously contextual and time-aware (competing state-of-the-art is limited due to ignoring either one of these aspects). The competing *CP Forecasting* was run using Holt forecasting, given the lack of data seasonality. The results of *Coupled Matrices* were obtained by finding non-negative factors that minimize the reconstruction error of the collapsed tensors, weighted for the same importance.

Considering only the harder problem of predicting new relationships (that didn't exist on any previous time step), the overall precision decreases, but the advantage over the competing methods is even higher (we double their precision).

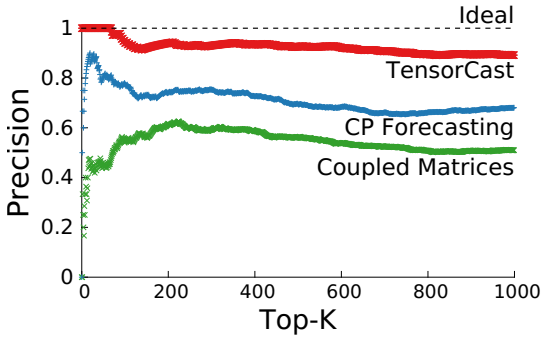

Figure 4: Higher precision vs competing methods when forecasting $(author, venue)$ relations in the DBLP dataset.

Precision over Time. We evaluate TENSORCAST's precision as the forecasting horizon is increased. We use the DBLP dataset, doing five runs with each method when considering different "training" periods (i.e., the first run considered every publication before 2010, while the last run considered every publication before 2015). For each run, we obtained the 1000 most likely non-zeros for each of the next 5 years and calculated each method's precision. Figure 5 shows the average precision for each forecasting horizon.

In addition to forecasting, the groups found by TENSORCAST are interpretable due to the non-negativeness of the factors (ex: in TWITTER one of the groups we found was characterized by conservative political hashtags, and another one was related to human rights and the iranian elections).

5 Conclusions

We presented TENSORCAST, a method which addresses the forecasting problem on big time-evolving datasets when contextual information is available. We leverage typical graph properties in order to create a precise algorithm that can find novel relations in very big datasets efficiently.

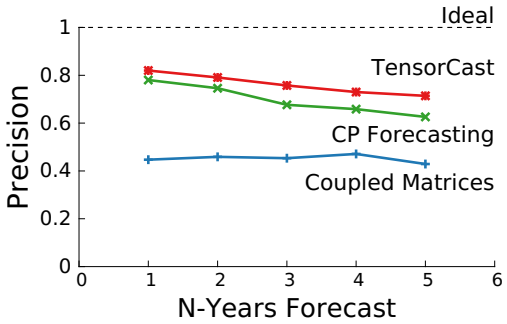

Figure 5: TENSORCAST achieves higher precision at every forecasting horizon in the DBLP dataset.

Acknowledgements

This work was partly financed by the ERDF through the 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.

This material is based upon work supported by by the Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Army Research Laboratory, the U.S. Government, or other funding parties. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[Araujo *et al.*, 2014] Miguel Araujo, Stephan Günnemann, Gonzalo Mateos, and Christos Faloutsos. Beyond blocks: Hyperbolic community detection. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 50–65. Springer, 2014.

[Araujo *et al.*, 2017] Miguel Ramos Araujo, Pedro Manuel Pinto Ribeiro, and Christos Faloutsos. Tensorcast: Forecasting with context using coupled tensors (best paper award). In *Data Mining (ICDM), 2017 IEEE International Conference on*, pages 71–80. IEEE, 2017.

[Beutel *et al.*, 2014] Alex Beutel, Partha Pratim Talukdar, Abhimanu Kumar, Christos Faloutsos, Evangelos E Papalexakis, and Eric P Xing. Flexifact: Scalable flexible factorization of coupled tensors on hadoop. In *SDM*, pages 109–117. SIAM, 2014.

[Box and Pierce, 1970] George EP Box and David A Pierce. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American Statistical Association*, 65(332):1509–1526, 1970.

[Dunlavy *et al.*, 2011] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(2):10, 2011.

[Gao *et al.*, 2011] Sheng Gao, Ludovic Denoyer, and Patrick Gallinari. Temporal link prediction by integrating content and structure information. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1169–1174. ACM, 2011.

[Harshman, 1970] Richard A Harshman. Foundations of the parafac procedure: Models and conditions for an" explanatory" multi-modal factor analysis. 1970.

[Kolda and Bader, 2009] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[Lee and Seung, 2001] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[Leskovec *et al.*, 2005] Jurij Leskovec, Deepayan Chakrabarti, Jon Kleinberg, and Christos Faloutsos. Realistic, mathematically tractable graph generation and evolution, using kronecker multiplication. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 133–145. Springer, 2005.

[Liben-Nowell and Kleinberg, 2007] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.

[Matsubara *et al.*, 2012] Yasuko Matsubara, Yasushi Sakurai, Christos Faloutsos, Tomoharu Iwata, and Masatoshi Yoshikawa. Fast mining and forecasting of complex time-stamped events. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 271–279. ACM, 2012.

[Papalexakis *et al.*, 2012] Evangelos E Papalexakis, Christos Faloutsos, and Nicholas D Sidiropoulos. Parcube: Sparse parallelizable tensor decompositions. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 521–536. Springer, 2012.

[Ram and Gray, 2012] Parikshit Ram and Alexander G Gray. Maximum inner-product search using cone trees. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 931–939. ACM, 2012.

[Sharan and Neville, 2008] Umang Sharan and Jennifer Neville. Temporal-relational classifiers for prediction in evolving domains. In *2008 Eighth IEEE International Conference on Data Mining*, pages 540–549. IEEE, 2008.

[Şimşekli *et al.*, 2013] Umut Şimşekli, Beyza Ermiş, A Taylan Cemgil, and Evrim Acar. Optimal weight learning for coupled tensor factorization with mixed divergences. In *21st European Signal Processing Conference (EUSIPCO 2013)*, pages 1–5. IEEE, 2013.

[Sun *et al.*, 2006] Jimeng Sun, Dacheng Tao, and Christos Faloutsos. Beyond streams and graphs: dynamic tensor analysis. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 374–383. ACM, 2006.

[Welling and Weber, 2001] Max Welling and Markus Weber. Positive tensor factorization. *Pattern Recognition Letters*, 22(12):1255–1261, 2001.

[Yılmaz, 2012] Yusuf Kenan Yılmaz. *Generalized tensor factorization*. PhD thesis, Citeseer, 2012.

[Zellner, 1962] Arnold Zellner. An efficient method of estimating seemingly unrelated regressions and tests for aggregation bias. *Journal of the American statistical Association*, 57(298):348–368, 1962.