Stream Mining of Complex Event Tensor

Kota Nakamura
Superviesd by : Yasuko Matsubara, Koki Kawabata, Yasushi Sakurai
ISIR, Osaka University
kota88@sanken.osaka-u.ac.jp

ABSTRACT

Given that large tensor streams of time-evolving events (e.g., taxi rides), which contain multiple attributes, how do we obtain intuitive groups and temporal patterns? Also, how do we incrementally capture latent structure to achieve a meaningful summarization? In this paper, we propose a streaming algorithm, namely TriComp, which is designed to automatically find both latent groups and temporal patterns in such complex yet huge collections. Our method has the following advantages: (a) It is effective: it provides compact and interpretable representations that reveal similar features with respect to both attributes and time. (b) It is automatic: it automatically recognizes and summarizes temporal patterns without any parameter tuning. (c) It is scalable: it is incremental yet scalable, and thus requires computational time that is independent of data stream length. Extensive experiments on two real datasets demonstrate that TriComp provides a summarization that helps us understand the complicated data, and it consistently outperforms the state-of-the-art methods in terms of both execution speed and accuracy.

1 INTRODUCTION

Countless domains including location-based services [11], click logs on websites [1], e-commerce, medical records [16], incessantly observe time-stamped events along with the multiple categorical attributes associated with them. Moreover, thanks to the advent of the IoT, which enables us to access a massive volume and variety of time series. In this situation, data is being generated with no end in sight and not all of them can be stored. Thus, one of the most fundamental requirements, if we are to employ them for applications, is to obtain a good summary in a streaming fashion.

In practice, we need a summary that offers interpretability revealing underlying groups, adaptability so that it captures dynamics and switches according to the state of the world, and automaticity that requires no human intervention. For example, taxi rides are continuously recorded with attributes such as pick-up time, pick-up location, drop-off location, taxi type, and customer type. To utilize them to design precise marketing strategies and beneficial business planning while reflecting actual conditions, service providers want to know that there are groups of locations and customers, and the difference between periods, without any tuning that requires time and human resources.

To realize the above effective representation, it is necessary to achieve the following two challenging and important tasks: (1) Structure mining: the goal is to provide an interpretable description that includes groups and their participation degrees. Unlike continuous sequences represented by sensor data, event collections turn into a huge yet sparse tensor, since events have numerous categorical features. We refer to such data as "*complex events*". (2) Temporal pattern mining: the aim is to discover repeated time-evolving patterns. However, it is difficult because the number of patterns, their characteristic, and their duration are rarely known a priori. Thus it is necessary to learn them through data automatically. We also refer to such a temporal pattern as a "*regime*".

There are following related works in terms of the two challenges to be addressed:

Structure mining. Decomposition employing probability distributions helps us to discover underlying structures [15, 17]. TriMine [10] is a scalable method that extracts interpretable features using the concept of topic modeling for forecasting future events. CoSTCo [9] has been proposed as a convolutional neural network based model for sparse tensor completion. In addition, learning mixture distributions, such as Latent Dirichlet allocation (LDA) [3] and its variants [2, 7], enable us to analyze large sets of categorical data. However, these methods not intend to capture temporal patterns.

Temporal pattern mining. Classical approaches, such as hidden Markov models (HMM) are extended to model dynamics and discovery patterns in [12, 14]. TICC [5] reveals distinct patterns based on a Markov random field. StreamScope [8] discovers similar patterns in an online fashion, whereas CubeMarker [6] can handle a 3rd-order tensor. The previous methods are employed to capture temporal patterns in continuous time-series, they thus cannot address complex events with sparsity and categorical attributes.

In short, none of the previous studies specifically address both structure mining and temporal pattern mining in complex event streams. We thus aim to tackle an important yet difficult task that is to reveal similar features with respect to both attributes and time simultaneously, moreover in automatic and streaming settings. Informally, if each event has a timestamp and two attributes (i.e., entity1, entity2), the problem we want to solve is as follows.

InformalProblem 1. *Given triplet collections of streams (entity1, entity2, timestamp), (1) find the latent groups in complex event collections and (2) summarize all dynamical patterns into models automatically and incrementally, at any point in time.*

2 PROBLEM FORMULATION

In this section, we formally define the concepts behind our model and the problem we want to solve. We mainly focus on a 3rd-order tensor for simplicity throughout this paper. Our method however can be applied to higher-order tensors. Consider that we receive time-stamped event entries of the form *(entity1, entity2, timestamp)*. We then have a collection of entries with u unique entity1, v unique entity2 and n timestamps. We turn them into a 3rd-order tensor, i.e., $\mathcal{X} \in \mathbb{N}^{u \times v \times n}$.

Proceedings of the VLDB 2021 PhD Workshop, August 16th, 2021. Copenhagen, Denmark.

Definition 2.1 (Complex event tensor). Let $\mathcal{X}$ be a 3rd-order tensor of complex time-stamped events. The element $x_{i,j,t}$ of $\mathcal{X}$ shows the total number of event entries of the i-th entity1 and the j-th entity2 at time tick t.

We assume that an event entry has a common "*latent group*". In that case, the original tensor will be decomposed into three matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ to represent groups with respect to entity1, entity2, and time.

Definition 2.2 (Participation matrix $\mathbf{A}$ $(u \times k)$). Each entry $a_{i,j}$ shows the participation strength of entity1 i for group j to describe how strongly each entity1 participates in groups #1, #2, . . ., #k. We let any participation weight $a_{i,j}$ be non-negative, and their total become 1 among k groups, i.e., $\sum_{j=1}^{k} a_{i,j} = 1$.

The definitions of $\mathbf{B}$ and $\mathbf{C}$ are analogous, each of them corresponds to entity2 and time. Although the multi-way representation allows us to find latent groups, it is insufficient for mining tensor streams containing variable patterns; the latent groups and their participants should change following fluctuation of dynamics. Thus, we extend the representation by adding another higher-level architecture.

Definition 2.3 (Regime). Let θ be a regime consisting of the three matrices: $\theta = \{\mathbf{A}, \mathbf{B}, \mathbf{C}\}$ to represent a certain pattern, with which we can divide the entire tensor stream into segments. When there are m regimes, a full model set is defined as $\Theta = \{\theta_1 \dots \theta_m\}$.

We adopt all of the above components for dynamic modeling over $\mathcal{X}$. The problem that we want to solve is eventually as follows:

Problem 1 (Streaming summarization). **Given** *a tensor stream $\mathcal{X}$;* **Find** *full model set Θ that summarizes the whole input tensor $\mathcal{X}$, namely, $\Theta = \{\theta_1 \dots \theta_m\}$, and the number of regimes m,*

3 STREAMING ALGORITHMS

In this section, we proposed a streaming approach, TriComp, which solves Problem 1. Intuitively, the main idea behind our algorithm is to continuously generate a model parameter set (i.e., regime) from current tensor $\mathcal{X}^C$, and to try to update a full model set Θ using the generated regime. More specifically, the algorithm comprises the following two main procedures (Algorithm 1) :

(P1) TriComp-deComp: Estimate a candidate regime θ_c from a tensor $\mathcal{X}^C$. It is derived by online decomposition while considering past L temporal dependencies.

(P2) TriComp-Compress: Keep track of two regimes, namely the previous regime θ_p and the current candidate regime θ_c. In this step, the algorithm decides whether or not to employ θ_c and selects the optimal regime following our coding scheme. Also full model set Θ is suitably updated to settle streaming.

We roughly refer to the event streams of $\mathcal{X}^C$ as a partial tensor of $\mathcal{X}$, whose length is $\tau \ll n$, and whose elements are $x_{i,j,t-\tau+1}, \dots, x_{i,j,t}$. Our algorithm works when given a non-overlapping stream $\mathcal{X}^C$ at every time point t with interval τ.

3.1 TriComp-deComp

Here, our aim is to incrementally obtain matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ as a candidate regime θ_c, which summarizes the most recent stream $\mathcal{X}^C$. To discover several latent groups in $\mathcal{X}^C$ automatically, we propose using the concept of topic modeling, which enables us to probabilistically assign each event entry to a latent group based on Dirichlet priors.

For incremental inference, a straightforward approach repeats decomposition in each tensor $\mathcal{X}^C$; however, this approach discards previous distributions and group allocations lose temporal consistency at every inference. Therefore, TriComp-deComp tracks dynamics by conveying previous model parameters. We assume that participation strength fluctuate over time, and their intensity at the current time t is the same as that at those previous time $t-1$ unless otherwise confirmed by the newly observed data. Specifically, previous participation strength $\hat{\mathbf{A}}_{t-1,u}$,$\hat{\mathbf{B}}_{t-1,k}$ and $\hat{\mathbf{C}}_{t-1,k}$ are integrated in Dirichlet prior (e.g., Dirichlet($\alpha \hat{a}_{t-1,i}$)). Letting α, β, and γ be the parameters of the Dirichlet priors for $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$, respectively, the generative process of a matrices is:

$$\begin{aligned} Draw\, \mathbf{A}_i &\sim Dirichlet(\textstyle\sum_{l=1}^{L} \alpha \hat{a}_{t-l,i}), \\ Draw\, \mathbf{B}_r &\sim Dirichlet(\textstyle\sum_{l=1}^{L} \beta \hat{b}_{t-l,r}), \\ Draw\, \mathbf{C}_r &\sim Dirichlet(\textstyle\sum_{l=1}^{L} \gamma \hat{c}_{t-l,r}). \end{aligned} \tag{1}$$

By utilizing collapsed Gibbs sampling [13], we can efficiently draw latent variables $z_{i,j,t}$ for each non-zero element $x_{i,j,t}$ in $\mathcal{X}^C$ with the following probability:

$$\begin{aligned} &p(z_{i,j,t} = r | \mathcal{X}, \mathbf{A}', \mathbf{B}', \mathbf{C}', \alpha, \beta, \gamma, \hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}) \\ &\propto \frac{a'_{i,r} + \sum_{l=1}^{L} \alpha \hat{a}_{l,i,r}}{\sum_{r=1}^{k} a'_{i,r} + L\alpha} \cdot \frac{b'_{r,j} + \sum_{l=1}^{L} \beta \hat{b}_{l,r,j}}{\sum_{j=1}^{v} b'_{r,j} + L\beta} \cdot \frac{c'_{r,t} + \sum_{l=1}^{L} \gamma \hat{c}_{l,r,t}}{\sum_{t=1}^{n} c'_{r,t} + L\gamma}, \end{aligned} \tag{2}$$

where $a_{i,r}$, $b_{r,j}$, and $c_{r,t}$ are the total counts with which group r is assigned to the i-th entity1, the j-th entity2, and time t, respectively. The prime (e.g., $a'_{i,r}$) indicates that the summations which is excluded the count of the entry, $x_{i,j,t}$. After the sampler has burned-in, we produce estimated participation matrices. $\tilde{\mathbf{A}}$, $\tilde{\mathbf{B}}$, and $\tilde{\mathbf{C}}$ as follows:

$$\tilde{a}_{i,r} \propto \frac{a_{i,r} + \sum_l \alpha \hat{a}_{l,i,r}}{\sum_r a_{i,r} + L\alpha}, \tilde{b}_{r,j} \propto \frac{b_{r,j} + \sum_l \beta \hat{b}_{l,r,j}}{\sum_j b_{r,j} + L\beta}, \tilde{c}_{r,t} \propto \frac{c_{r,t} + \sum_l \gamma \hat{c}_{l,r,t}}{\sum_t c_{r,t} + L\gamma}. \tag{3}$$

Thanks to the introduction of previous participation strength for each group, we need not store previous tensors to represent dynamics. It thus omits memory space and time complexity.

3.2 TriComp-Compress

We next tackle the problem of how to optimize full model set Θ. We can solve this problem intuitively by monitoring the regime shift and continuously choosing an appropriate model in $\{\theta_p, \theta_c\}$ based on their data summarization qualities.

Here, we introduce novel coding scheme to define a good summary for a complex event tensor and determine a optimal compact description automatically. Our coding scheme composed of model description cost $< \Theta >$ and data coding cost $< \mathcal{X} | \Theta >$, which is based on the minimum description length (MDL) principle [4]. Since we cannot store and process all historical data in a streaming fashion, we only consider the increase in the total cost when a set of new event streams is added to $\mathcal{X}$. we define the additional description cost as follows:

$$\Delta < \mathcal{X}; \Theta > = < \theta_* > + < \mathcal{X}^C | \theta_* >, \tag{4}$$

where θ_* represents an employed regime for describing the arrival tensor $\mathcal{X}^C$. Since the model description cost is the number of bits

Algorithm 1 TriComp ($\mathcal{X}^C, \Theta$)

Input: Current tensor $\mathcal{X}^C \in \mathbb{N}^{u \times v \times \tau}$ and previous full model set Θ
Output: Updated full model set Θ'

```
/* (I) TriComp-deComp */
Compute A, B, C; // Equation (2) and (3)
θ_c ← {A, B, C};
/* (II) TriComp-Compress */
Compute < X^C; θ_p > and < X^C; θ_c >;  // Equation (4)
if < X^C; θ_p > is less than < X^C; θ_c > then
    /* Stay on the previous regime θ_p */
    θ'_p ← RegimeUpdate (θ_p, θ_c);
else
    θ_e = arg min_{θ∈Θ} < X^C; θ >;  // Equation (4)
    if < X^C; θ_c > is less than < X^C; θ_e > then
        /* Shift to the candidate regime θ_c */
        Θ' ← Θ ∪ θ_c;  m ← m + 1;
    else
        /* Shift to the existing regime θ_e */
        θ'_e ← RegimeUpdate (θ_e, θ_c);
    end if
end if
return Θ' = {θ_1, ..., θ_m};
```

needed to describe the model, $< \theta >$ is defined as $< \theta > = < \mathbf{A} > + < \mathbf{B} > + < \mathbf{C} >$, where[1],

$$< \mathbf{A} > = |\mathbf{A}| \cdot (\log((k-1) * u) + c^F) + \log^*(|\mathbf{A}|), \tag{5}$$

$$< \mathbf{B} > = |\mathbf{B}| \cdot (\log((v-1) * k)) + c^F) + \log^*(|\mathbf{B}|), \tag{6}$$

$$< \mathbf{C} > = |\mathbf{C}| \cdot (\log((n-1) * k) + c^F) + \log^*(|\mathbf{C}|), \tag{7}$$

where $|\cdot|$ describes non-zero elements when $1/k$, $1/v$, and, $1/n$ are subtracted and c^F is the floating point cost[2].

The data encoding cost of $\mathcal{X}$ given θ is the negative log-likelihood in Huffman coding, computed by: $< \mathcal{X}|\theta >= -\log P(\mathcal{X}|\mathbf{A}, \mathbf{B}, \mathbf{C})$. Thus the total encoding cost of $\mathcal{X}$ given Θ is:

$$< \mathcal{X}|\Theta > = \sum_{p=1}^{m} -\log P(\mathcal{X}[r_p]|\theta_p), \tag{8}$$

where, $\mathcal{X}[r_p]$ is a set of partial tensors assigned by the p-th regime.

4 EXPERIMENTS

We evaluated the performance of our proposed method on the two real data streams. (#1) NY-Taxi [3] dataset is the records of Yellow Taxi trips in New York City, which contains Pick up location ID ($u = 262$), Drop off location ID ($v = 263$), and the hourly timestamp of each ride ($n = 4368$). (#2) NY-Bike [4] dataset is the history of a bicycle ride-share service in New York City, whose attributes include the user-age groups separated by every five ages from 10 to 100 ($u = 19$), the start station ID ($v = 488$), and, the start time per hour ($n = 8760$).

Effectiveness. We will answer the question that *How well does our method provide meaningful summaries with latent dynamical patterns and groups?* Figure 1 (a) shows TriComp can incrementally and automatically find two regimes and these assignments. Concretely, our method first discovers "Regime 1" based on the three latent groups. After the dynamics behind the data have been changed at time tick 80, our method automatically generates "Regime 2" for the new pattern because the two regimes give us better summarization of the stream. These two representations coincide with realistic activities that correspond to weekdays and weekends. The "weekend" regime (Regime 2) successfully recognizes a public holiday at time tick 630 despite a non-periodic event.

Furthermore, TriComp provides participation matrices which show common groups and their attributes-wise participation weights. Figure 1 (b) and (c) show the estimated participation matrices, where there are three latent activity groups (shown with three colors) and the darker locations indicate stronger participation in the groups, whereas each sequence in Figure 1 (a) describes the time-wise intensities of each group.

As an example, we focus on Group 3 (green), which has a high intensity when assigned into the weekend regime; therefore, it is strongly related to weekend. Also, red circle areas in Figure 1 (c) shift from Group 2 (orange) to Group 3 on weekends, suggesting that these areas possess different features from the ones on weekdays. Considering these areas are located around vast parks and Lower Manhattan with a large number of bars and restaurants, it is thus likely to represent a group related to amusement. Activity groups are common among attributes, so we can know taxi rides which related to amusement are occurred from green area (Group 3) on the right in Figure1 (b). In short, our dynamic approach provides sense-making insights.

Accuracy and scalability. We will answer the following questions: *How accurately does our method summarize complex events? How does our method scale in terms of computational time?* To conduct quantitative evaluations in two real datasets, we evaluated how accurate the model represents original streams by comparing reconstruction accuracy in terms of 1) perplexity, and 2) RMSE. The competitors consist of TriMine [10] and CoSTCo [9] which are designed for sparse tensor decomposition.

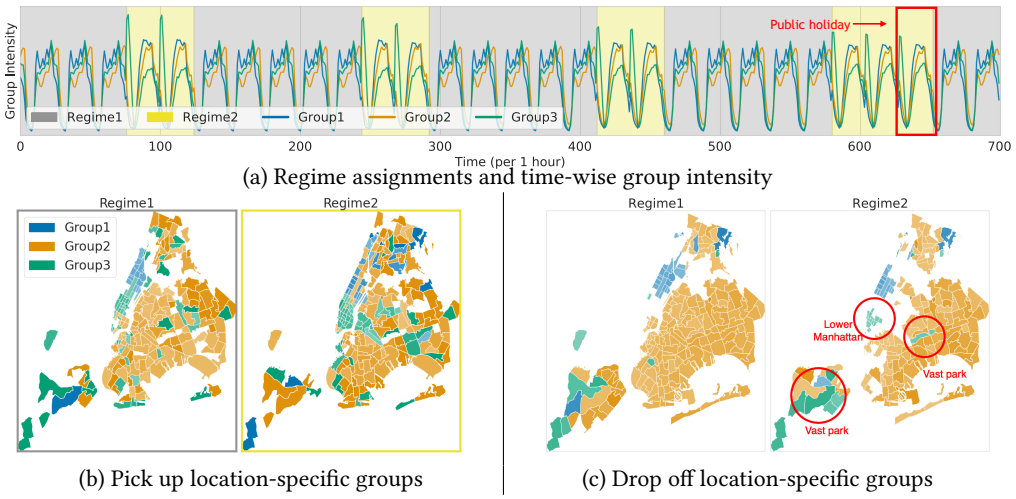
Figure 2 shows the average perplexity and the average root mean square error (RMSE) between the original tensor and estimated values using every current tensor $\mathcal{X}^C$ of length 24. With both of them, lower values indicate better model constructions. Overall, our proposed method outperforms the other offline methods because it can capture high-level patterns, i.e., regimes. CoSTCo is capable of handling sparse tensors but it is not optimized for complex event streams with latent dynamics.

Also, we evaluate the performance of TriComp in terms of computational time by comparison with its competitors. Figure 3 indicates the wall clock time of an experiment performed on each datasets. In each dataset, our method is faster than its competitors by up to four orders of magnitude thanks to our incremental update.

5 CONCLUSION AND FUTURE WORK

In this paper, we presented TriComp, an efficient streaming method for tensor streams of time-evolving events. Our proposed method has the following advantages: (a) It is *effective:* our experiments show that TriComp successfully discover both latent groups and typical patterns from huge yet complex events. (b) It is *automatic:* TriComp is carefully designed to summarize event tensor streams

[1] Here, log* is the universal code length for integers.
[2] We set 8 bits in our setting.
[3] https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
[4] https://www.citibikenyc.com/system-data

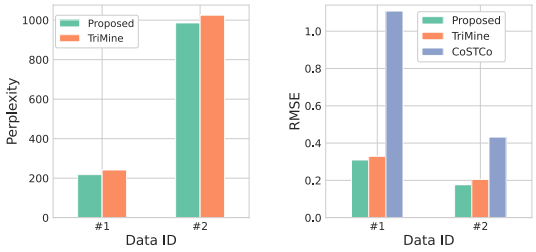


(a) Regime assignments and time-wise group intensity

(b) Pick up location-specific groups

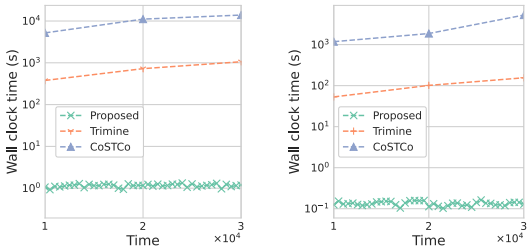
(c) Drop off location-specific groups

Figure 1: Modeling power of TriComp for taxi ride events: (a) Given the event tensor stream, it incrementally identifies regimes, i.e., underlying distinct dynamics (shaded rectangle with yellow/gray), and also captures time-wise intensity sequences of each group. In addition, it reveals the attribute-specific groups of (b) pick up and (c) drop off areas, in which the colors show their latent groups and depth of the colors show attribute-wise participation weights (too low degrees are not shown (< 0.4)).



(a) Average perplexity (b) Average RMSE

Figure 2: TriComp outperforms its baselines in accuracy (lower is better).



(#1) : NY-Taxi (#2) : NY-Bike

Figure 3: Wall clock time vs. stream length: TriComp summarizes data streams in constant time.

without prior tuning. (c) It is *scalable:* thanks to incremental model updating by TriComp, the computational time is constant with regard to the entire length of the input tensor.

In future work, we will make the proposed method more general by evaluating it on various datasets such as e-commerce, web access, and medical records. In addition, we will also conduct extensive experiments on more than three order tensors. Finally, we aim to design a fully automatic and fast algorithm. Specifically, we consider that it would automatically determine not only the number of regimes but also the number of groups.

REFERENCES

[1] Deepak Agarwal, Bee-Chung Chen, and Pradheep Elango. Spatio-temporal models for estimating click-through rate. In *WWW*, pages 21–30, 2009.

[2] David M Blei and John D Lafferty. Dynamic topic models. In *ICML*, pages 113–120, 2006.

[3] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[4] Peter D Grünwald, In Jae Myung, and Mark A Pitt. *Advances in minimum description length: Theory and applications.* MIT press, 2005.

[5] David Hallac, Sagar Vare, Stephen Boyd, and Jure Leskovec. Toeplitz inverse covariance-based clustering of multivariate time series data. In *KDD*, pages 215–223, 2017.

[6] T. Honda, Y. Matsubara, R. Neyama, M. Abe, and Y. Sakurai. Multi-aspect mining of complex sensor sequences. In *ICDM*, pages 299–308, 2019.

[7] Tomoharu Iwata, Shinji Watanabe, Takeshi Yamada, and Naonori Ueda. Topic tracking model for analyzing consumer purchase behavior. In *IJCAI*, 2009.

[8] Koki Kawabata, Yasuko Matsubara, and Yasushi Sakurai. Automatic sequential pattern mining in data streams. In *CIKM*, pages 1733–1742, 2019.

[9] Hanpeng Liu, Yaguang Li, Michael Tsang, and Yan Liu. Costco: A neural tensor completion model for sparse tensors. In *KDD*, pages 324–334, 2019.

[10] Yasuko Matsubara, Yasushi Sakurai, Christos Faloutsos, Tomoharu Iwata, and Masatoshi Yoshikawa. Fast mining and forecasting of complex time-stamped events. In *KDD*, pages 271–279, 2012.

[11] Rimma V Nehme, Elke A Rundensteiner, and Elisa Bertino. Tagging stream data for rich real-time services. *VLDB Endowment*, 2(1):73–84, 2009.

[12] Spiros Papadimitriou and Philip Yu. Optimal multi-scale patterns in time series streams. In *SIGMOD*, pages 647–658, 2006.

[13] Ian Porteous, David Newman, Alexander Ihler, Arthur Asuncion, Padhraic Smyth, and Max Welling. Fast collapsed gibbs sampling for latent dirichlet allocation. In *KDD*, pages 569–577, 2008.

[14] Yasushi Sakurai, Spiros Papadimitriou, and Christos Faloutsos. Braid: Stream mining through group lag correlations. In *SIGMOD*, pages 599–610, 2005.

[15] Aaron Schein, Mingyuan Zhou, David Blei, and Hanna Wallach. Bayesian poisson tucker decomposition for learning the structure of international relations. In *ICML*, pages 2810–2819. PMLR, 2016.

[16] Yichen Wang, Robert Chen, Joydeep Ghosh, Joshua C Denny, Abel Kho, You Chen, Bradley A Malin, and Jimeng Sun. Rubik: Knowledge guided tensor factorization and completion for health data analytics. In *KDD*, pages 1265–1274, 2015.

[17] Liang Xiong, Xi Chen, Tzu-Kuo Huang, Jeff Schneider, and Jaime G Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *SDM*, pages 211–222. SIAM, 2010.