Incremental Group-Level Popularity Prediction in Online Social Networks

JINGJING WANG, WENJUN JIANG, and KENLI LI, Hunan University
GUOJUN WANG, Guangzhou University
KEQIN LI, Hunan University and State University of New York

Predicting the popularity of web contents in online social networks is essential for many applications. However, existing works are usually under non-incremental settings. In other words, they have to rebuild models from scratch when new data occurs, which are inefficient in big data environments. It leads to an urgent need for incremental prediction, which can update previous results with new data and conduct prediction incrementally. Moreover, the promising direction of group-level popularity prediction has not been well treated, which explores fine-grained information while keeping a low cost. To this end, we identify the problem of incremental group-level popularity prediction, and propose a novel model *IGPP* to address it. We first predict the group-level popularity incrementally by exploiting the incremental CANDECOMP/PARAFCAC (CP) tensor decomposition algorithm. Then, to reduce the cumulative error by incremental prediction, we propose three strategies to restart the CP decomposition. To the best of our knowledge, this is the first work that identifies and solves the problem of incremental group-level popularity prediction. Extensive experimental results show significant improvements of the *IGPP* method over other works both in the prediction accuracy and the efficiency.

CCS Concepts: • **Information systems → Data analytics**; • **Networks → Social media networks**; **Online social networks;**

Additional Key Words and Phrases: Group level, incremental approach, information diffusion, online social networks, popularity prediction, tensor analysis

ACM Reference format:
Jingjing Wang, Wenjun Jiang, Kenli Li, Guojun Wang, and Keqin Li. 2021. Incremental Group-Level Popularity Prediction in Online Social Networks. *ACM Trans. Internet Technol.* 22, 1, Article 20 (September 2021), 26 pages.
https://doi.org/10.1145/3461839

This research was supported by NSFC grant 62172149 and 61632009, the National Outstanding Youth Science Program of NSFC grant 61625202, the science and technology program of Changsha city kq 2004017, and Open project of Zhejiang Lab 2019KE0AB02.

Authors' addresses: J. Wang, W. Jiang (corresponding author), and K. Li, College of Information Science and Engineering, Hunan University, Changsha, Hunan, 410082, China; emails: {wangjingjing2019, jiangwenjun, lkl}@hnu.edu.cn; G. Wang, School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou, Guangdong, China; email: csgjwang@gzhu.edu.cn; K. Li, College of Information Science and Engineering, Hunan University, Changsha, Hunan, 410082, China, and Department of Computer Science, State University of New York, New Paltz, New York, 12561; email: lik@newpaltz.edu.

1533-5399/2021/09-ART20 $15.00
https://doi.org/10.1145/3461839

1 INTRODUCTION

With the continuous emergence of a variety of new social platforms, various types of information, such as videos, pictures, and posts, are being posted and diffused to a wide range in **online social networks (OSNs)** every day. Popularity prediction aims to predict the future popularity of online content (i.e., how many users repost or view contents), and to predict how information is propagated. It is an essential issue for many real-world applications, including online advertising services [4], recommendation systems [11, 15, 18], and so on [17, 19, 42]. Due to the massive growth of online diffusion data, it is very challenging to predict popularity efficiently. In this article, we strive to explore an efficient incremental approach for popularity prediction. Our work can also be used to deal with more other applications in the big data environment.

As has been pointed out in [2], although there is a growing interest in web content popularity prediction, incremental prediction (or online learning) has not been addressed. In fact, the incremental method is essential and more suitable for real-world scenarios for the following two reasons. First, as an enormous amount of information diffusion data is continuously being generated, it is almost impossible to obtain the global data at once. Even if the global data is already available, the computation of non-incremental methods is extremely expensive due to high time and space complexity. Second, for content diffusing over time in real-world scenarios, when new diffusion data occurs, non-incremental methods have to rebuild the model from scratch and conduct popularity prediction, whereas the incremental method can reuse and update previous results to predict popularity efficiently based on the new data. Hence, there is an urgent need for incremental prediction.

Existing works on popularity prediction are usually from either the macro or the micro perspective. The macro approaches predict the *population-level popularity* [5, 6, 31, 34] (i.e., how many users in total will react to content). In this direction, most previous works [5, 31, 34] extract various type of features, including content features, temporal features, structural features, and user features [25, 40, 41], then predict the future popularity of information by training a regression or classification model based on the historical contents. These approaches are based on the intuition that contents from the same social media follow a similar diffusion pattern with respect to observed features [7]. There are also some generative approaches [6, 9, 33] that are devoted to characterizing and modeling the process that a content obtains attentions. These models generally assume that the diffusions of contents are independent of each other and learn content-specific parameters without the diffusion information of other contents [7]. The micro approaches predict the *user-level popularity* (i.e., predict which users will react to a content [24], or estimate the propagation probability that a content propagates from one individual to another [16]) by modeling behaviors of individual users [26].

Most recently, Hoang et al. [14] found that in many OSNs, users naturally form groups [12], reflecting their interests, communities, or locations; users in the same group are fairly consistent in reacting to a content. Motivated by the preceding observation, they proposed a novel framework for predicting the *group-level popularity* of contents based on the user network and historical contents. The group-level prediction is much less noisy and takes smaller computational cost than the user-level prediction, and it is more detailed and cohesive than the population-level prediction. To some degree, the population-level popularity and user-level popularity are two extreme cases of the group-level popularity (i.e., when all users are in the same group or each user is as a group).

However, the only method for the group-level popularity prediction, GPOP [14], cannot support the incremental prediction, because it exploits a global batch processing and a non-incremental framework. It conducts multiple static tensor decomposition with a hierarchical constraint on the global data to predict the group-level popularity. Therefore, it has to rebuild the model from scratch

every time for prediction the next time. In other words, it cannot reuse previous results and update incrementally with new data. In addition, it has no incremental version, because its incremental updates and restarting are the same in this setting (i.e., rebuilding the model from scratch). This violates the idea of incremental prediction—that is, reusing and updating the previous results with new data to predict incrementally.

To this end, we identify the problem of incremental group-level popularity prediction, which aims to take advantage of the evolving feature of information diffusion to predict group-level popularity incrementally. There are three main challenges to solve this problem:

(1) *Data organization*: For accurate popularity prediction, we need to integrate all the data of users' social network, content propagation, and temporal information. But how can organize these data cohesively and efficiently?

(2) *Incremental prediction*: As new data may occur now and then, it will be time consuming and space consuming to combine the new data in the current one and repeat all operations on the combined data to predict popularity. So, how can efficiently update the model and incrementally predict popularity with new data?

(3) *Error reduction*: In incremental prediction, we use the predictive value in the previous steps as the true value to predict the popularity the next time. Inevitably, there is cumulative error in the process of predicting popularity incrementally with new data. Then, how can we reduce the cumulative error caused by incremental prediction?

To address the preceding challenges, we propose *IGPP*, a novel incremental group-level popularity prediction method, which is based on two intuitions mentioned earlier (i.e., the similarity of users' behaviors and interests in the same group [14], and the strong correlation between the future and early popularity with respect to temporal features [34]). For data organization, *IGPP* uses a tensor [22, 30], which is a multidimensional or N-way array, to organize and represent all related data clearly and naturally. For incremental prediction, *IGPP* exploits incremental **CANDECOMP/PARAFAC (CP)** decomposition, which is an incremental low-rank tensor approximation technique for feature extraction, dimensionality reduction, and knowledge discovery on a tensor [52], for exploring the underlying patterns of newly added data and predicting group-level popularity incrementally. For the third challenge, we design three restarting strategies for cumulative error reducing. In summary, our contributions are as follows:

- We identify the problem of incremental group-level popularity prediction, which brings up new insights to track the evolving processes of contents over time incrementally, and predict the group-level popularity incrementally. To the best of our knowledge, this is the first work that identifies and addresses the problem of incremental group-level popularity prediction.
- We propose a novel *IGPP* model to address the problem. We first exploit the incremental CP decomposition to predict the group-level popularity incrementally. Then, we propose three restarting strategies to reduce the cumulative error caused by incremental updates and improve the prediction accuracy.
- We conduct extensive experiments to test the *IGPP* model variants on two real datasets of Behance and Twitter. The experimental results confirm that our methods achieve higher accuracy and take shorter running time than baselines. To be specific, *IGPP*-8h, *IGPP*-0.001, and *IGPP*-10% run up to 90.86×, 37.41×, and 45.43× faster than the state-of-the-art group-level popularity prediction method *GPOP* while having lower prediction errors.

It is worth noting that, popularity prediction is different from trending topic prediction [27, 28] and event prediction [29, 32, 48, 50], which are more complex issues. A topic is described as a coherent set of semantically related contents. Trending topic prediction [27, 28] needs to first retrieve related contents and detect trending topics, and then predict the future popularity of the

detected topics. An event refers to a real-world occurrence that happens at some specific time and location with a specific semantic topic [50]. Event prediction [29, 32, 50] focuses on anticipating events in the future, given the event data and historical event data. Its output are entities with rich information, such as time, location, and semantics. Real-time event prediction [32] requires continuous monitoring of the observed input data to trigger timely alerts of future potential events. To some degree, popularity prediction can be taken as an essential part of trending topic prediction. For instance, the owners of social platforms can use popularity prediction to trend topic tracking and to avoid serious information overload problems caused by super popular contents.

2 RELATED WORKS

Popularity prediction in recent years has become one of the most popular research contents in the field of social networking [2, 10, 37, 51]. Researchers make a lot of effort to study popularity prediction, and their work can be generally classified into three categories: the classification problem, the ranking problem, and precise popularity prediction. We will introduce them in detail.

2.1 Classification-Based Popularity Prediction

Some papers regard the popularity prediction as classification problems [43, 44]. Their popularity status space can be either a binary space {Popular, Unpopular} or {Low Popularity, Medium Popularity, High Popularity} or other more refined space containing multiple levels of popularity. Kim et al. [21] defined four different types of discrete temperature scale, such as explosive, hot, warm, and cold, and derived a sound regression model to predict the popularity temperature. Xu et al. [44] developed contextual bandits learning by incorporating the contextual information of the social network, and proposed Pop-Forecast, a systematic method for the popularity prediction of videos promoted through social networks. Xu et al. [43] proposed the Social-Forecast algorithm, which can choose to make a prediction classification using the currently observed context information or wait to make this prediction until the next period.

2.2 Ranking-Based Popularity Prediction

Some papers regard the popularity prediction as a ranking problem [35, 36, 38, 45], which aims to rank articles based on their predicted popularity. Yin et al. [45] assumed that each person has two personalities, Conformer and Maverick, which guide the voting behavior. Combining both of positive and negative votes, Yin et al. [45] proposed a Conformer-Maverick (CM) model to predict whether an item will be popular or not, and ranked top-k potentially popular items based on the early votes they received. Tatar et al. [35, 36, 38] considered two properties of news articles, the distribution of popularity and the lifetime of articles, and predicted the popularity of news articles based on user comments.

2.3 Precise Popularity Prediction

The preceding two types of popularity prediction cannot predict a numeric value, such as the exact amount of retweets or forwards that a content will receive, so they have limited application. More and more research focuses on the precise popularity prediction, which can be generally divided into three categories: popularity-level popularity prediction, user-level popularity prediction, and group-level popularity prediction.

2.3.1 Population-Level Popularity Prediction. Many papers predict the popularity from the population level that predicts how many users in total will react to a content [7, 31, 34]. Szabo and Huberman [34] found that a strong linear correlation exists between the logarithmically transformed popularity of content at early and later times, and presented a model (also the SH model) to pre-

dict future popularity. Pinto et al. [31] extended the SH model and proposed the multivariate linear model (also called the *Pinto model*) and the MRBF model. Cao et al. [7] improved the SH model [34] and the Pinto model [31] by dividing messages into groups and trained a group-specific model for the messages of each group. Bao et al. [5] focused their attention on the structural characteristics of the networks composed of early adopters (i.e., the link density and the diffusion depth). Then they improved the SH model [34] by using structural characteristics. Shen et al. [33] proposed a generative probabilistic framework using a reinforced Poisson process, *RPP*, to model and predict the popularity of individual items. Gao et al. [9] developed the *RPP* model that divides the whole process of retweeting dynamics into several subprocesses, and each subprocess is modeled by an *RPP* model. Bao et al. [6] proposed a probabilistic model using *SEHP* (Self-Excited Hawkes Process) to characterize and predict the popularity of individual microblogs.

2.3.2 User-Level Popularity Prediction. Some papers predict the popularity from the user level. In other words, they predict which users will react to a content [24] or estimate the propagation probability that a content propagates from one individual to another [16] by modeling behaviors of individual users. Huang et al. [16] found that the probability a message propagates between two individuals decays with the length of time latency since their latest interaction, obeying a power-law rule, and proposed a temporal model to estimate future propagation probabilities between individuals. Lerman and Hogg [24] proposed stochastic models of user behavior and predicted popularity based on early user reactions to new content. Matsubara et al. [26] introduced TriMine to find patterns and trends in large set of clicks, including predicting the clicks number from a specific user on the next day. Yu et al. [46] found two interesting phenomena, minor dominance and early stage dominance, and proposed a novel Networked Weibull Regression mode for behavioral dynamics modeling. Zaman et al. [47] developed a probabilistic model of the Bayesian approach to provide popularity predictions and posterior credible intervals for the predictions.

2.3.3 Group-Level Popularity Prediction. However, there are some shortcomings in the preceding works. The user-level popularity prediction is susceptible to noisy (i.e., missing) data and often is costly to learn [8]. The population-level popularity prediction is only able to provide a very coarse view. Recently, Hoang et al. [14] proposed a promising direction of group-level popularity, which is more fine grained than the aggregate network level while less noisy than the individual user level. They also address the problem by designing the *GPOP* model, which first groups user into cohesive clusters and then adopts CP decomposition to predict group-level popularity.

Although there are lots of works on web content popularity, there is a significant lack of solutions for incremental prediction (or online learning and data streams). This was previously noted in other works [2, 37] and has been unexplored so far. In this article, we identify the problem of incremental group-level popularity prediction, which focuses on the incremental prediction and combining the advantages of group-level popularity prediction. To address the problem, we propose the *IGPP* model based on the incremental CP decomposition method, which fully exploits the new data, updates the previous results incrementally, and predicts the group-level popularity of contents incrementally.

3 PROBLEM DEFINITION

In this section, we first identify the problem of incremental group-level popularity prediction. Then we introduce an overview of our solution. The notations used in this article are listed in Table 1.

3.1 The Problem of Incremental Group-Level Popularity Prediction

The incremental group-level popularity prediction problem. Given a network G, a set of historical contents $P = \{p_1, p_2, \ldots, p_m\}$ (each content is observed over a period of q timestamps), a set of

Table 1. Notations

Notation	Explanation
t_1	Initial size on temporal mode
k	Number of similar contents being selected
n	Number of nodes
m	Number of historical contents
q	Maximum timestamp
l	Number of groups
Δ	Fixed time interval for restarts
θ	Threshold for restarts
C	User groups
$x, \mathbf{x}, \mathbf{X}, \mathcal{X}$	Scalar, vector, matrix, and tensor
x_{ijt}	Element of $\mathcal{X}$, the accumulative popularity of content i over group j until timestamp t
$\mathbf{A}, \mathbf{B}, \mathbf{C}$	Loading matrices of $\mathcal{X}$ by CP decomposition
$\mathbf{A}^T, \mathbf{A}^{-1}$, $\mathbf{A}^\dagger, \|\mathbf{A}\|$	Transpose, inverse, Moore-Penrose pseudoinverse, and Frobenius norm of $\mathbf{A}$
$\mathbf{X}_{(i)}, \mathbf{X}_{old(i)}$	Unfoldings of $\mathcal{X}$, $\mathcal{X}_{old}$ along mode i
$\odot$, $\otimes$, $*$	Kronecker, Khatri-Rao, and Hadamard product

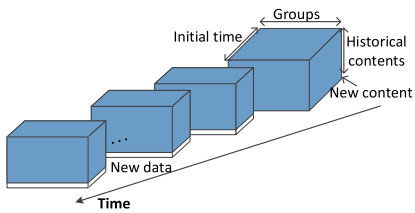


Fig. 1. Illustration of the incremental group-level popularity prediction problem.

user groups, $C = \{C_1, C_2, \ldots, C_l\}$, and the group-level popularity of a new content p_{m+1} during a observable period of t_1 timestamps ($t_1 < q$), the task is to *incrementally* predict the group-level popularity of p_{m+1} during the period $[t_1 + 1, q]$. In other words, suppose the popularity (i.e., the number of being retweeted or propagated) of p_{m+1} over all groups at t_1 is vector $\mathbf{x}_{m+1,t_1}$, given $\{\mathbf{x}_{m+1,1}, \mathbf{x}_{m+1,2}, \ldots, \mathbf{x}_{m+1,t_1}\}$, and as time goes on successively predicts its popularity in future time points $\{\mathbf{x}_{m+1,t_1+1}\}$, $\{\mathbf{x}_{m+1,t_1+2}\}, \ldots, \{\mathbf{x}_{m+1,q}\}$ (Figure 1).

3.2 Solution Overview

Taking the data organization, incrementally predicting, and error reducing into consideration, we propose an *IGPP* method by exploiting the incremental CP decomposition to solve the problem of incremental group-level popularity prediction. This is because the tensor can organize and represent all related data clearly and naturally, and incremental CP decomposition can track the CP decomposition incrementally for exploring and extracting the underlying patterns and the hidden information of new data. The *IGPP* has three steps: preprocessing, incrementally predicting, and restarting:

(1) *Preprocessing*: We preprocess information diffusion data to improve the data quality and build a group-level popularity tensor $\mathcal{X}_{init}$ for the target content p_{m+1} as the initial conditions (Section 4.1).

(2) *Incrementally predicting*: For each new chunk of diffusion data $\mathcal{X}_{new}$, where the group-level popularity of p_{m+1} is missing and needs to be predicted, $\mathcal{X}_c$ is expanded from $\mathcal{X}_{old}$ by appending $\mathcal{X}_{new}$ at its time mode. We exploit the incremental CP decomposition to predict the group-level popularity of p_{m+1} incrementally (Section 4.2).

(3) *Restarting*: To improve the prediction accuracy, we propose the CP decomposition restarting method with three strategies to reduce the cumulative error caused by incremental prediction (Section 4.3).

The differences in our work from the work of Hoang et al. [14]. We would like to highlight the differences between our work and the original group-level popularity prediction paper [14], including problems and methods. First, our problem is developed from the recent work in group-level popularity prediction [14]. However, our problem, the incremental group-level popularity prediction, focuses on the study of *incremental prediction* and has profound implications in real-world applications as mentioned in Section 1, which essentially motivates this article. Second, we also use the CP decomposition, but our method, *IGPP*, is very different from *GPOP* [14]. To be specific, the basic idea of *IGPP* is an incremental CP decomposition [52], which updates the previous results based on new data and tracks the CP decomposition of an online tensor incrementally. Based on this, we design different restarting CP decomposition strategies to reduce the cumulative error and improve the prediction accuracy. *GPOP* is a non-incremental method, and it is impossible to develop an incremental version. It first conducts the traditional CP decomposition on four tensors, which store the group-level popularity and population-level popularity of historical contents and the target content, respectively. Then, it predicts group-level popularity in a hierarchical constraint prediction framework.

The fundamental differences in the methods lead to different performances. We conduct extensive experiments to compare our method with *GPOP* in detail. Compared with *GPOP*, *IGPP* is more accurate and more efficient. What is more, we deeply study the effect of different restarting strategies on *IGPP*, and we recommend the appropriate restarting strategy to consumers with different requirements for accuracy and efficiency.

4 *IGPP*: THE MODEL DETAILS

In this section, we introduce the *IGPP* model in detail, including preprocessing, incrementally predicting, and restarting. We also analyze the time complexity.

4.1 *IGPP*: Preprocessing

We conduct some preprocessing to build the group-level popularity tensor, including grouping users and selecting top-k similar contents for p_{m+1}. We adopt the methods proposed by Hoang et al. [14] for preprocessing. We briefly introduce the basic idea as follows.

4.1.1 Grouping Users. Hoang et al. [14] define a network-constrained popularity graph G^*, which is obtained by weighting of the users' historical activities network and the user network. Then, they use the multilevel k-way partitioning algorithm [20] on G^* for graph clustering.

To study the effect of different groups on the *IGPP* method, we construct a user graph G, a popularity graph G^S, and a network-constrained popularity graph G^*, respectively (see the appendix for more details). Then, we obtain three different groups by using the multilevel k-way partitioning algorithm. We also group users randomly. We test the impacts of different groups via experiments.

After grouping users, we can build the group-level popularity tensor $\mathcal{X}$, whose element x_{ijt} refers to the accumulative popularity of content i over group j until timestamp t. It is worth noting that the life cycle of a content in OSNs is usually short, and the changes of the user network during this period are relatively insignificant. Therefore, without loss of generality, we assume that the user groups are unchanged. In other words, we use the user groups obtained at the maximum observed timestamp to predict the popularity in the future short period.

4.1.2 Selecting Top-k Similar Contents for the Target Content. To make $\mathcal{X}$ smaller and more relevant for low computational cost and high accurate prediction, we select k similar contents for

ALGORITHM 1: Preprocessing of *IGPP*

Input: $P = \{p_1, \cdot, \cdot, \cdot, p_m\}$: historical contents.
p_{m+1}: the target content being predicted.
t_1: the observable period of p_{m+1}.
Output: $\mathcal{X}_{init}$: initial group-level popularity tensor.
1 Set the number of groups as l;
2 Build a network-constrained popularity graph G^*;
3 Group users $C = \{C_1, C_2, \ldots, C_l\}$ on the graph using k-way partition algorithm;
4 Construct $\mathcal{X}$ for P and p_{m+1} based C;
5 Normalize $\mathcal{X}$ at timestamps t_1 with Equation (1);
6 Calculate $D^*_{t_1}(p_i, p_{m+1})$ $(i = 1, \ldots m)$ with Equation (3);
7 Select k contents with the smallest $D^*_{t_1}(p_i, p_{m+1})$ $(i = 1, \ldots m)$ as the top-k similar contents of p_{m+1};
8 Construct $\mathcal{X}_{init}$ based on C and top-k similar contents;
9 **return** $\mathcal{X}_{init}$;

p_{m+1} in a normalized space like that of Hoang et al. [14]. Given the observable time period $[1, t_1]$, we normalize every content in $\mathcal{X}$ at timestamp t_1 as follows:

$$\tilde{x}_{ijt} = x_{ijt} \Big/ \sum_j x_{ijt_1} \; \forall i, t, j, \tag{1}$$

where $\sum_j x_{ijt_1}$ is the accumulative popularity of content i over all groups until timestamp t and is called the *normalized factor of content i*. Then, the distance at timestamp t_1 between p_{m+1} and another content p_i is defined as the Euclidean distance as follows:

$$D_{t_1}(p_i, p_{m+1}) = \sqrt{\sum_{t=1,\ldots,t_1; j=1,\ldots,l} (\tilde{x}_{ijt} - \tilde{x}_{m+1,j,t})}. \tag{2}$$

To reduce the impacts of outliers (similar to p_{m+1} at timestamp t_1, but very different from p_{m+1} in the future), we also include an outlierness score, which is defined as the average distance at timestamp q between p_i and the rest of the historical contents. So, the distance is finally defined as

$$D^*_{t_1}(p_i, p_{m+1}) = D_{t_1}(p_i, p_{m+1}) \times \frac{\sum_{j=1,\ldots,m; j\neq i} D_q(p_i, p_j)}{m-1}. \tag{3}$$

Then, we select k contents with the smallest distance to p_{m+1} as the top-k similar contents of p_{m+1}. After grouping users and selecting top-k similar contents, we construct a group-level popularity tensor $\mathcal{X}_{init} \in \mathbb{R}^{(k+1)\times l\times t_1}$ for the k similar contents and p_{m+1} during the observable time period $[1, t_1]$ as the initial conditions.

4.1.3 Preprocessing Algorithm. The detailed preprocessing of *IGPP* is shown in Algorithm 1. *IGPP* first groups users on G^* (lines 1–3). Next, *IGPP* selects k similar contents for p_{m+1} (lines 4–7). Finally, *IGPP* constructs $\mathcal{X}_{init}$ based on the preceding groups and similar contents (lines 8 and 9).

Complexity analysis. In Algorithm 1, the time complexity of grouping users based on the multi-level k-way partitioning algorithm is $O(|E|)$. Selecting k similar contents needs to calculate the distance between each historical content and p_{m+1}, and chooses top-k contents with the smallest distance, whose time complexity is $O(m^2ql)$. Therefore, the total time complexity is $O(m^2ql + |E|)$.

4.2 *IGPP*: Incrementally Predicting

The incrementally predicting process is shown in Figure 2. It exploits the incremental CP decomposition [52] to track CP decomposition of the group-level popularity tensor and predict

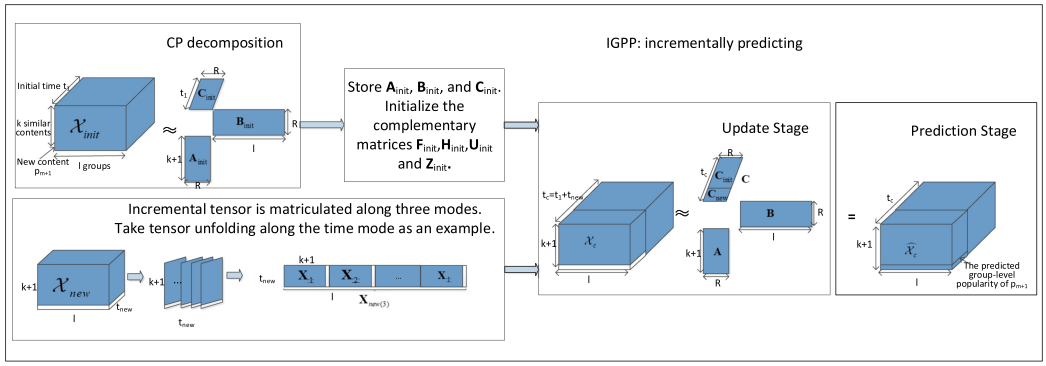


Fig. 2. *IGPP*: incrementally predicting.

group-level popularity incrementally. This process has three stages: the initialization stage, the update stage, and the prediction stage.

4.2.1 The Initialization Stage. As shown in the upper left part of Figure 2, for the initial group-level popularity tensor $\mathcal{X}_{init} \in \mathbb{R}^{(k+1)\times l\times t_1}$, we calculate its CP decomposition $\mathcal{X}_{init} \approx [\![\mathbf{A}_{init}, \mathbf{B}_{init}, \mathbf{C}_{init}]\!]$. $\mathbf{A}_{init} \in \mathbb{R}^{(k+1)\times R}$, $\mathbf{B}_{init} \in \mathbb{R}^{l\times R}$, $\mathbf{C}_{init} \in \mathbb{R}^{t_1\times R}$ are latent content, group, and time feature matrices, and their row vectors representing content-specific, group-specific, and time-specific latent feature vectors, respectively. R is the number of latent dimension. We also use complementary matrices to store the information of previous results in the process of incremental updates for $\mathbf{A}$ and $\mathbf{B}$. Here, we initialize complementary matrices $\mathbf{F}_{init}, \mathbf{H}_{init}, \mathbf{U}_{init}$, and $\mathbf{Z}_{init}$ as follows so that $\mathbf{A}_{init} = \mathbf{F}_{init}\mathbf{H}_{init}^{-1}$ and $\mathbf{B}_{init} = \mathbf{U}_{init}\mathbf{Z}_{init}^{-1}$. In other words, $\mathbf{F}_{init}, \mathbf{H}_{init}^{-1}$ are feature weight matrices for $\mathbf{A}_{init}$, and $\mathbf{U}_{init}, \mathbf{Z}_{init}^{-1}$ are feature weight matrices for $\mathbf{B}_{init}$:

$$\begin{aligned}
\mathbf{F}_{init} &= \mathbf{X}_{init(1)}(\mathbf{C}_{init} \odot \mathbf{B}_{init}),\\
\mathbf{H}_{init} &= (\mathbf{C}_{init}^T\mathbf{C}_{init}) * (\mathbf{B}_{init}^T\mathbf{B}_{init}),\\
\mathbf{U}_{init} &= \mathbf{X}_{init(2)}(\mathbf{C}_{init} \odot \mathbf{A}_{init}),\\
\mathbf{Z}_{init} &= (\mathbf{C}_{init}^T\mathbf{C}_{init}) * (\mathbf{A}_{init}^T\mathbf{A}_{init}),
\end{aligned} \tag{4}$$

where $\mathbf{X}_{init(1)}$ and $\mathbf{X}_{init(2)}$ are the unfoldings of $\mathcal{X}_{init}$ along mode 1 and mode 2, respectively. In addition, $\odot$ and $*$ are the Khatri-Rao product and Hadamard product [22], respectively.

4.2.2 The Update Stage. As time goes on, a piece of new data $\mathcal{X}_{new} \in \mathbb{R}^{(k+1)\times l\times t_{new}}$ comes, where the group-level popularity of p_{m+1} is missing (see the white part of $\mathcal{X}_{new}$ in Figure 2). The current group-level popularity tensor $\mathcal{X}_c \in \mathbb{R}^{(k+1)\times l\times t_c}$ $(t_c = t_{old} + t_{new})$ is expanded from $\mathcal{X}_{old} \in \mathbb{R}^{(k+1)\times l\times t_{old}}$ by appending the new data $\mathcal{X}_{new}$ at its time mode. We update the factor matrices based on the unfolding of $\mathcal{X}_{new}$, complementary matrices and the previous factor matrices of $\mathcal{X}_{old}$ (the lower left and middle parts of Figure 2).

The update process is similar to the classical alternating least squares algorithm [52]. In other words, we first fix the content mode $\mathbf{A}$ and the group mode $\mathbf{B}$ to update the time mode $\mathbf{C}$. We then update matrices $\mathbf{A}$ and $\mathbf{B}$ successively by fixing the other two matrices. It is worth noting that unlike $\mathbf{A}$ and $\mathbf{B}$, the size of factor matrix $\mathbf{C}$ on the time mode has changed (see Figure 2). So, the ways to update them are different.

Update the time mode $\mathbf{C}$. By fixing factor matrices of $\mathbf{A}$ and $\mathbf{B}$ as $\mathbf{A}_{old}$ and $\mathbf{B}_{old}$, the factor matrix of time mode $\mathbf{C}$ is updated by appending the projection $\mathbf{C}_{new}$ of $\mathbf{X}_{new(3)}$ via the factor matrices

$\mathbf{A}_{old}$ and $\mathbf{B}_{old}$, to $\mathbf{C}_{old}$—that is,

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_{old} \\ \mathbf{C}_{new} \end{bmatrix} = \begin{bmatrix} \mathbf{C}_{old} \\ \mathbf{X}_{new(3)}((\mathbf{B}_{old} \odot \mathbf{A}_{old})^T)^\dagger \end{bmatrix}. \tag{5}$$

Update content mode A and group mode B. First, we update $\mathbf{A}$. By fixing $\mathbf{B}$ and $\mathbf{C}$ as $\mathbf{B}_{old}$ and $\mathbf{C}$, the objective is to minimize the estimation error $\mathcal{L}$, which can be written as $\mathcal{L} = \frac{1}{2}\|\mathbf{X}_{(1)} - \mathbf{A}(\mathbf{C} \odot \mathbf{B}_{old})^T\|^2$. The derivative of $\mathcal{L}$ w.r.t. $\mathbf{A}$ is as follows:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{A}} = \mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B}_{old}) - \mathbf{A}(\mathbf{C} \odot \mathbf{B}_{old})^T(\mathbf{C} \odot \mathbf{B}_{old}). \tag{6}$$

By setting Equation (6) to zero and letting $\mathbf{F} = \mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B}_{old})$ and $\mathbf{H} = (\mathbf{C} \odot \mathbf{B}_{old})^T(\mathbf{C} \odot \mathbf{B}_{old})$, we have $\mathbf{A} = \mathbf{F}\mathbf{H}^{-1}$. By representing $\mathbf{X}_{(1)}$ and $\mathbf{C}$ with the old and new components, and $\mathbf{F}_{old} = \mathbf{X}_{old(1)}(\mathbf{C}_{old} \odot \mathbf{B}_{old})$, $\mathbf{F}$ can be represented as $\mathbf{F} = \mathbf{F}_{old} + \mathbf{X}_{new(1)}(\mathbf{C}_{new} \odot \mathbf{B}_{old})$. Similarly, $\mathbf{H}$ can be represented as $\mathbf{H} = \mathbf{H}_{old} + (\mathbf{C}_{new} \odot \mathbf{B})^T(\mathbf{C}_{new} \odot \mathbf{B})$. The Khatri-Rao product $(\mathbf{C}_{new} \odot \mathbf{B})^T(\mathbf{C}_{new} \odot \mathbf{B})$ can be obtained by calculating it as $(\mathbf{C}^T\mathbf{C} * \mathbf{B}_{old}^T\mathbf{B}_{old})$ [22].

Therefore, the factor matrix on content mode $\mathbf{A}$ can be updated incrementally as follows:

$$\begin{aligned} &\mathbf{F} \leftarrow \mathbf{F}_{old} + \mathbf{X}_{new(1)}(\mathbf{C}_{new} \odot \mathbf{B}_{old}) \\ &\mathbf{H} \leftarrow \mathbf{H}_{old} + \left(\mathbf{C}_{new}^T\mathbf{C}_{new}\right) * \left(\mathbf{B}_{old}^T\mathbf{B}_{old}\right) \\ &\mathbf{A} \leftarrow \mathbf{F}\mathbf{H}^{-1}. \end{aligned} \tag{7}$$

The update rule for the factor matrix on group mode $\mathbf{B}$ can be derived in a similar way, as follows:

$$\begin{aligned} &\mathbf{U} \leftarrow \mathbf{U}_{old} + \mathbf{X}_{new(2)}(\mathbf{C}_{new} \odot \mathbf{A}) \\ &\mathbf{Z} \leftarrow \mathbf{Z}_{old} + (\mathbf{C}_{new}^T\mathbf{C}_{new}) * (\mathbf{A}^T\mathbf{A}) \\ &\mathbf{B} \leftarrow \mathbf{U}\mathbf{Z}^{-1}. \end{aligned} \tag{8}$$

4.2.3 The Prediction Stage. Based on the incremental updates, we can obtain $\mathcal{X}_c \approx [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]_c = \widehat{\mathcal{X}_c}$, where $\mathbf{A}$ is the factor matrix for the k historical contents and p_{m+1}, $\mathbf{B}$ is the factor matrix for l groups, and $\mathbf{C}$ is the factor matrix for the time mode. These three factor matrices capture the latent representations at the group levels using the k historical contents, and also map p_{m+1} to the same latent space as that of the historical contents. We predict p_{m+1}'s group-level popularity at t_c timestamp as follows:

$$\mathbf{x}_{m+1,t_c} \leftarrow \{\widehat{x}_{m+1,j,t_c} | j = 1, \ldots, l\},\ t_c \in [t1+1, q]. \tag{9}$$

4.2.4 Case Study. Figure 3 is the flow chart of *IGPP* for a target content, whose observable period is three timestamps. In preprocess, we cluster users into $l = 5$ groups and select its $k = 5$ similar contents. Then, we build its group-level popularity tensor $\mathcal{X} \in \mathbb{R}^{6\times5\times3}$, where the sixth content is the target content (see Figure 4(a)). We normalize $\mathcal{X}$ with Equation (1) and obtain the normalized initial group-level popularity tensor $\mathcal{X}_{init}$ (the tensor in Figure 4(c)). Here, we set the rank $R = 1$ for simplicity.

In the initialization stage of incrementally predicting, we first calculate CP decomposition of $\mathcal{X}_{init}$ to get factor matrices $\mathbf{A}_{init}$, $\mathbf{B}_{init}$, and $\mathbf{C}_{init}$ (Figure 4(c)), and initialize all complementary matrices $\mathbf{F}$, $\mathbf{H}$, $\mathbf{U}$, and $\mathbf{Z}$ with Equation (4) (Figure 4(b)).

In the update stage, for each new data $\mathcal{X}_{new} \in \mathbb{R}^{6\times5\times1}$, where group-level popularity values of the sixth content are zeros and need to be predicted, we first update the factor matrix $\mathbf{C}$ by appending the projection of $\mathcal{X}_{new}$ to $\mathbf{C}_{init}$ with Equation (5). Next, we update the factor matrix $\mathbf{A}$ of content mode with Equation (7) and the factor matrix $\mathbf{B}$ of group mode with Equation (8)

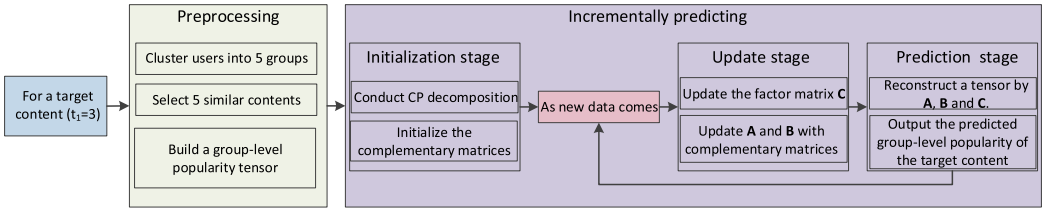
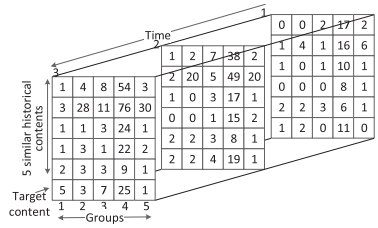
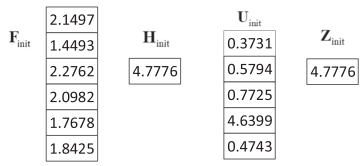


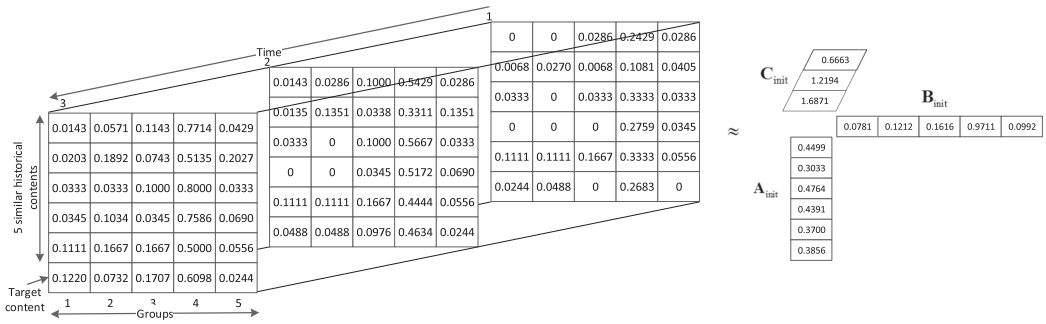
Fig. 3. A flow chart of the case study.



(a) initial group-level popularity tensor



(b) the initial complementary matrices



(c) initial group-level popularity tensor after normalization $\mathcal{X}_{init}$ and its CP decomposition

Fig. 4. An example.

(Figure 5(b) shows the updated complementary matrices). So, we get the updated factor matrices, **A**, **B**, and **C** (see Figure 5(c)).

In the prediction stage, the matrices **A**, **B**, and **C** are the factor matrices of the current tensor $\mathcal{X}_c \in \mathbb{R}^{6\times5\times4}$ by incremental updates. Then, the vector $\mathbf{x}_{6j4}$ $(j = 1, \ldots, 5)$ in the reconstructed tensor $\widehat{\mathcal{X}}_c$ by **A**, **B**, and **C** is the predicted group-level popularity of the sixth content at the fourth timestamp in the normalized space. Based on the normalized factor with Equation (1), we can get the predicted group-level popularity finally (see Figure 5(d)). Due to page limitations, we only show the update process once, and the predicted results of the sixth content are shown in Figure 5(d).

4.3 *IGPP*: Restarting

We test the prediction performance of *IGPP* and find that the **root mean square error (RMSE)** of *IGPP* increases greatly and monotonically with time (shown in Figure 6). There are two main reasons. First, as new data increases, *IGPP* has to continuously update incrementally. The reconstruction loss of the observable data increases due to the approximation on incremental updates of incrementally predicting. Thus, the prediction error of the missing data increases. Second, for each new data, *IGPP* updates the previous results to get new ones and predicts the missing data.

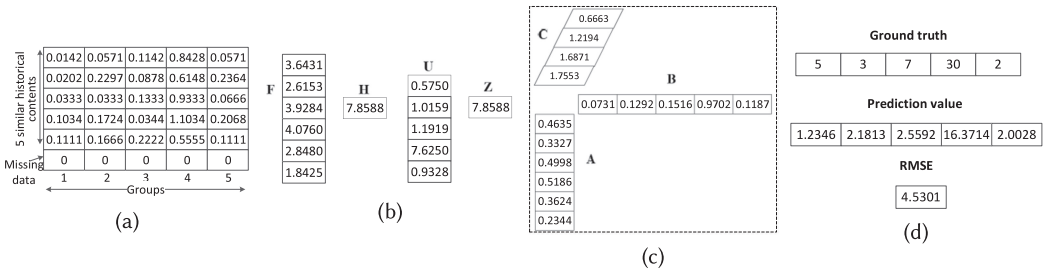


Fig. 5. Incrementally predicting for a new data. (a) New data 1. (b) Complementary matrices after update. (c) Factor matrices after update. (d) Ground truth, predicted group-level popularity and RMSE.

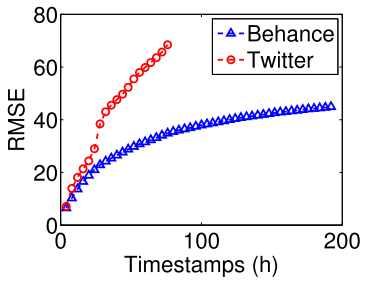


Fig. 6. Performance of *IGPP* without restarts (unit of timestamps: hours).

Then, it will reuse these new results to update incrementally for the next new data. In this process, the prediction results are taken as the true values for the next prediction, which further expand the prediction error.

To address the preceding issues, we propose an effective approach of restarting the CP decomposition at some timestamps, which can reset the cumulative error induced by incrementally predicting. Some work has been proposed to restart SVD in an incremental SVD method [49]. However, no work has been done to restart CP decomposition in incremental CP decomposition. We propose three heuristic strategies to determine the restarting time points in this article, as follows.

Fixing the time interval Δ (*IGPP-*Δ). We periodically restart the CP decomposition after a certain time interval Δ. It is simple and easy to apply.

Fixing the maximum relative cumulative error $\mathcal{I}$ *by a threshold* θ (*IGPP-*θ). In this strategy, we focus on the relative cumulative error $\mathcal{I}$ of the observable data. To be specific, we introduce the threshold θ and then restart the CP decomposition when the maximum relative cumulative error $\mathcal{I}$ is larger than θ. We first give two key concepts. Then, we present the detailed calculation process of $\mathcal{I}$.

Definition 4.1 (Reconstruction Error [39]). Given a 3-order tensor $\mathcal{X}$ and its factor matrices $\mathbf{A}, \mathbf{B}$, and $\mathbf{C}$ obtained by the CP decomposition or the incremental CP methods, the reconstruction error $\mathcal{J}$ refers to the loss between the reconstructed tensor by these factor matrices and the original tensor:

$$\mathcal{J} = \|\mathcal{X} - \mathbf{A} \odot \mathbf{B} \odot \mathbf{C}\|. \tag{10}$$

Definition 4.2 (Cumulative Error [39]). The cumulative error refers to the reconstruction error caused by incremental updates of the incremental CP decomposition, excluding the reconstruction loss by the optimal CP decomposition. For the same incremental tensor, assuming its reconstruction errors by the optimal CP decomposition and incremental CP decomposition are $\mathcal{J}^{oCP}$ and

$\mathcal{J}^{ICP}$, respectively, the cumulative reconstruction error $\Delta\mathcal{J}$ caused by the incremental CP decomposition can be calculated as follows:

$$\Delta\mathcal{J} = \mathcal{J}^{ICP} - \mathcal{J}^{oCP}. \tag{11}$$

Since $\mathcal{J}^{oCP}$ is intrinsic in the CP decomposition and cannot be reduced by restarting, it cannot guide the time point of restarting the CP decomposition. Instead, the cumulative error induced by incremental CP decomposition, $\Delta\mathcal{J}$, is a possible measure to guide CP decomposition restart. In addition, there are some missing data to be predicted, so we only calculate the cumulative error of observable data.

However, the determination of a CP rank is NP-hard [13], and there is no optimal (i.e., minimum) loss $\mathcal{J}^{oCP}$ proved in theory, so it is difficult to calculate the cumulative error $\Delta\mathcal{J}$ by $\mathcal{J}^{ICP} - \mathcal{J}^{oCP}$ directly. Thus, in this work, we perform the CP decomposition on $\mathcal{X}_c$ for each new data $\mathcal{X}_{new}$, which serves as the optimal decomposition with the minimum loss at timestamp t_c. The margin between the incremental CP decomposition and the optimal decomposition is taken as the actual cumulative error $\Delta\mathcal{J}$. We define the relative cumulative error $\mathcal{I}$ at timestamp t_c as follows:

$$\begin{aligned}\mathcal{I}(t_c) &= \frac{\Delta\mathcal{J}(t_c)}{\mathcal{J}^{oCP}(t_c)} = \frac{\mathcal{J}^{ICP}(t_c) - \mathcal{J}^{oCP}(t_c)}{\mathcal{J}^{oCP}(t_c)} \\ &= \frac{\|\Omega_c(\widehat{\mathcal{X}}_c - \mathcal{X}_c)\| - \|\Omega_c(\widehat{\mathcal{X}}'_c - \mathcal{X}_c)\|}{\|\Omega_c(\widehat{\mathcal{X}}'_c - \mathcal{X}_c)\|} \\ \Omega_c(i,j,t) &= \begin{cases} 0, \ if\ \mathcal{X}_c(i,j,t)\ is\ missing \\ 1, \ otherwise, \end{cases}\end{aligned} \tag{12}$$

where $\widehat{\mathcal{X}}_c$ and $\widehat{\mathcal{X}}'_c$ are the reconstruction tensor of factor matrices of $\mathcal{X}_c$ by the incremental CP decomposition and the optimal CP decomposition, respectively, and $\mathcal{J}^{ICP}(t_c)$ and $\mathcal{J}^{oCP}(t_c)$ are their reconstruction error at timestamp t_c, respectively. In addition, Ω_c is a mask tensor of $\mathcal{X}_c$, indicating the observed entries in $\mathcal{X}_c$. $\|\Omega_c(\widehat{\mathcal{X}}_c - \mathcal{X}_c)\|$ is the reconstruction error of observable data in $\mathcal{X}_c$ by incremental CP decomposition.

Fixing the maximum relative mean error of the previous prediction by a threshold σ (IGPP-σ). We compare the relative mean error of the previous prediction and the threshold σ to determine whether to restart CP decomposition. To be specific, we first calculate the relative mean error for the group REG_t at timestamps t (Section 4.2) as a feedback. Then, we restart the CP decomposition at timestamp $(t+1)$ if REG_t is larger than σ.

We have studied the problem of reducing the cumulative error of ICP in different OSN applications. The major differences between this work and our earlier work [39] are twofold. First, we have different focuses in the two works. In the earlier work [39], we focus on identifying the fundamental causes of cumulative errors by incremental CP decomposition in multiple OSN applications, and reducing errors. Second, we propose and address different problems in the earlier work [39]. We propose two optimization problems (i.e., minimizing the restarting times that keeping a small cumulative reconstruction error and prediction error, respectively), and proposed several restarting strategies, and applied them in three typical OSN applications.

4.4 IGPP: The Integrated Process

In this section, based on the preprocess with Algorithm 1, we combine the incremental prediction and restarting strategies to implement incremental group-level popularity prediction. Taking the

ALGORITHM 2: *IGPP*

Input: $\mathcal{X}_{init} \in \mathbb{R}^{(k+1)\times l\times t_1}$: the initial tensor obtained from Algorithm 1.
$\mathcal{X}_{new} \in \mathbb{R}^{(k+1)\times l\times t_{new}}$: newly added tensor.
Δ: the threshold for *IGPP* with Δ.
Output: $\{\widehat{\mathbf{x}}^{(t)}|t = 1, \ldots, T\}$: predicted group-level popularity of content p.

1 Calculate the CP decomposition on $\mathcal{X}_{init}$ to obtain its factor matrices $\mathbf{A}_{init}, \mathbf{B}_{init}, \mathbf{C}_{init}$ and initialize complementary matrices $\mathbf{F}_{init}, \mathbf{H}_{init}, \mathbf{V}_{init}$ and $\mathbf{Z}_{init}$ with Equation (4);
2 **Procedure of *IGPP*-Δ:**
3 Initialize the counter of time interval $C \leftarrow 0$;
4 **for** *newly added data* $\mathcal{X}_{new} \in \mathbb{R}^{(k+1)\times l\times t_{new}}$ **do**
5 $C \leftarrow C + t_{new}$;
6 **if** $C/\Delta = 0$ **then**
7 Calculate the current tensor $\mathcal{X}_c \in \mathbb{R}^{(k+1)\times l\times (t_1+C)}$ by appending $\mathcal{X}_{new}$;
8 Calculate the CP decomposition on $\mathcal{X}_c \approx [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$;
9 Calculate complementary matrices $\mathbf{F}, \mathbf{H}, \mathbf{U}$, and $\mathbf{Z}$ using Equation (4);
10 **else**
11 Update the temporal mode, $\mathbf{C}$ is updated with Equation (5);
12 Update the non-temporal mode, content mode, and group mode, $\mathbf{A}$ is updated with Equation (7) and $\mathbf{B}$ is updated using Equation (8);
13 **end**
14 $\widehat{\mathcal{X}} \leftarrow [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$;
15 $\hat{\mathbf{x}}_{m+1,t_1+j} \leftarrow \{\widehat{\mathcal{X}}_{m+1,j,t}|j = 1 \ldots l,\ t \in [t_1 + 1, q]\}$;
16 **end**
17 **return** $\{\hat{\mathbf{x}}_{m+1,t_1+1}, \ldots, \hat{\mathbf{x}}_{m+1,q}\}$.

first restart strategy (i.e., fixing the time interval Δ) as an example, the details of *IGPP* are shown in Algorithm 2. It is noticed that for different restarting strategies, IGPP has the same initialization: *IGPP* conducts the CP decomposition on $\mathcal{X}_{init}$ and initializes the factor matrices and the complementary matrices (line 1).

For *IGPP* with Δ, it first initiates a variable C for counting time intervals from the initial time t_1 (line 3). Then, for each newly added data $\mathcal{X}_{new}$, it adds t_{new} to the variable C, and determines restarting or not. If C is a multiple of Δ, it calculates the current tensor $\mathcal{X}_c$ by appending $\mathcal{X}_{new}$ to the previous tensor along the time mode, and restarts the CP decomposition on $\mathcal{X}_c$ to obtain factor matrices and complementary matrices (lines 6–9). Otherwise, it updates incrementally factor matrices $\mathbf{C}$, $\mathbf{A}$, and $\mathbf{B}$ successively, and complementary matrices (lines 10–13); then, *IGPP* predicts the group-level popularity of p_{m+1} in new timestamps based on $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ (lines 14–15). Finally, *IGPP* returns the predicted group-level popularity of p_{m+1} during the period $[t_1, +1, q]$ (line 17).

Complexity analysis. In the best case, for each new data $\mathcal{X}_{new} \in \mathbb{R}^{(k+1)\times l\times t_{new}}$, *IGPP* only updates factor matrices incrementally for prediction. Its time complexity is $O(Rl(k + 1)t_{new})$, where R is the number of latent dimension. There are a total of $(q - t_1)$ $\mathcal{X}_{new}$. So the total time complexity is $O(Rl(k + 1)t_{new}(q - t_1))$ in the best case. In the worst case, for every $\mathcal{X}_{new}$, *IGPP* needs to update factor matrices incrementally and restart the CP decomposition to predict the group-level popularity, whose time complexity is $O(Rl(k + 1)(2t_{new} + t_{old}))$. Then the total time complexity is $O(Rl(k + 1)(2t_{new} + t_{old})(q - t_1))$ in the worst case.

Table 2. Statistics in the Two Datasets and the Prediction Tasks

	Behance	Twitter
$\#Users$	85,092	22,255
$\#Edges$	13,428,364	575,819
$\#Contents$	1,326 projects	1,015 hashtags
$\#Timestamps$	60	24
Timestamp size	4 hours	4 hours
Prediction tasks		
History length $t1$	12 (2 days)	5 (20 hours)
Future length $q - t1$	48 (8 days)	19 (76 hours)

5 EXPERIMENTS

5.1 Experimental Setup

Datasets. We use two real-world datasets[1] for experiments: Behance [1] and Twitter [23] (Table 2). Both of them are typical social networks, and a content is a project in Behance or a hashtag in Twitter. The popularity of a content is the number of users who have appreciated it or the number of times it has been tweeted by users.

Baselines. We compare our method with variants of *IGPP*, the CP decomposition [22], and the most recent group-level prediction method, *GPOP* [14], which predicts group-level popularity using CP tensor decomposition with hierarchical constraints and cannot update the model incrementally over time.

Parameter setting. We set the number of groups as 12 for Behance and 11 for Twitter, consistent with *GPOP* in the work of Hoang et al. [14]. It is worth noting that the group-level popularity is a more general problem: when the number of groups is 1, it becomes the population-level popularity; when the number of groups is equal to the number of users, it becomes the user-level popularity. In addition, we set the size of new data at time mode $t_{new} = 4h$ for simplicity. Our method can be easily expanded to other cases with different t_{new} values. For each content, after preprocess (i.e., grouping users and selecting top-k similar content), its group-level popularity tensor possesses the feature of contents, groups, and temporal correlation. So, we set the number of latent dimension R as 2 [7]. Specifically, if *IGPP* needs to restart, we use the batch hot of CP decomposition, which is an implementation of the ALS algorithm in Tensor Toolbox [3], and use the decomposition results of the last timestep as the initialization for decomposing the current tensor. In each experiment, we run all algorithms 10 times on all contents and report the average results. All experiments are conducted on a server running CentOS Linux release 7.4.1708 with an Intel Core $i5\text{-}8600K$ $3.60GHz$ processor and 23.3 GB of RAM. All algorithms are implemented in Matlab, using Tensor Toolbox [3], Pablano Toolbox,[2] and the METIS library.[3]

5.2 Evaluation Metrics

In this work, we adopt three standard measurements, RMSE, REG, and average running time, as the evaluation metrics.

RMSE is a commonly used measurement of the differences between the predicted values and the values actually observed. In this work, we test the RMSE over m contents, l groups, and q timestamp, respectively:

[1] https://cs.ucsb.edu/~mhoang/gpop.tar.gz.
[2] https://github.com/sandialabs/poblano_toolbox.
[3] http://glaros.dtc.umn.edu/gkhome/metis/metis/download.

(1) RMSE over contents (*RMSE*)

$$RMSE = \frac{1}{m}\sum_{i=1}^{m}\sqrt{\frac{\sum_{j=1}^{l}\sum_{t=t_1}^{q}(\widehat{\mathcal{X}}_{ijt} - \mathcal{X}_{ijt})^2}{l \times (q - t_1)}}$$

(2) RMSE over groups ($RMSE_{groups}$)

$$RMSE_j = \frac{1}{m}\sum_{i=1}^{m}\sqrt{\frac{\sum_{t=t_1}^{q}(\widehat{\mathcal{X}}_{ijt} - \mathcal{X}_{ijt})^2}{(q - t_1)}}$$

(3) RMSE over time ($RMSE_{timestamps}$)

$$RMSE_t = \frac{1}{m}\sum_{i=1}^{m}\sqrt{\frac{\sum_{j=1}^{l}(\widehat{\mathcal{X}}_{ijt} - \mathcal{X}_{ijt})^2}{l}}$$

We introduce another metric, **relative mean error for the group (REG)** defined in the work of Hoang et al. [14] as follows:

$$REG = \frac{1}{m}\sum_{i}\frac{\sqrt{\sum_{j=1}^{l}\sum_{t=t_1}^{q}(\mathcal{X}_{ijt} - \widehat{\mathcal{X}}_{ijt})^2}}{\sqrt{\sum_{j=1}^{l}\sum_{t=t_1}^{q}\mathcal{X}_{ijt}^2}},$$

where $\widehat{\mathcal{X}}$ and $\mathcal{X}$ are all predicted group-level popularity and the ground truth of m contents, respectively.

In addition, we adopt the average running time for predicting popularity of one content during all future period $[t_1 + 1, q]$, measured in seconds, to validate the time efficiency of our model. Because *IGPP* predicts popularity incrementally, its running time is calculated as the sum of running time over all timestamps. For all m contents, the average running time of *IGPP* and other baselines is defined as follows:

$$Time_{IGPP} = \frac{1}{m}\sum_{i=1}^{m}\sum_{t=t_1}^{q} runningtime_{k,t},$$

$$Time_{Other} = \frac{1}{m}\sum_{i=1}^{m} runningtime_k.$$

To avoid any ambiguity, we use "time" to represent the average running time and "timestamps" to represents the future time period in experiments.

5.3 Experimental Results and Analysis

We conduct three groups of experiments: analyzing the impacts of several parameters, groups and restarting strategies, respectively, and comparing *IGPP* with different restarting strategies and baselines.

5.3.1 The Effects of Model Parameters. In this section, we test the effect of the number k of similar contents, the observable period t_1, and the thresholds on *IGPP* with different restarting strategies, respectively.

The effect of the number k of similar contents. To study the effect of the number k of similar contents, we conduct *IGPP*-Δ with $\Delta = 4h$ (i.e., *IGPP*-4h) on Behance and Twitter with varying k from 10 to 100. From Figure 7, we can observe that as k grows, both the RMSE and the REG of contents first decrease and then stabilize, because when k is too high, useless information is combined. At the same time, the curves of average running time are almost stable with some small

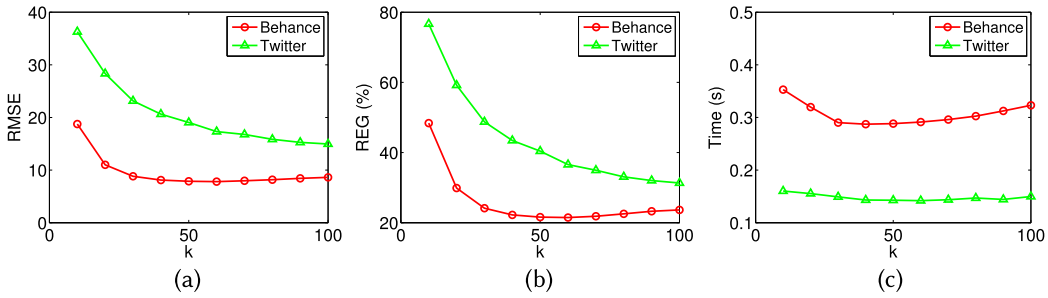


Fig. 7. RMSE, REG, and average running time (seconds) as the number k of similar contents varies.

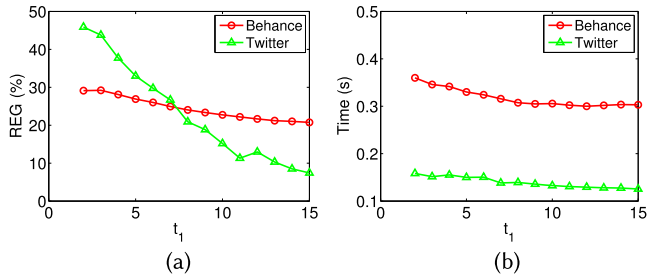


Fig. 8. REG and average running time (seconds) as the observable timestamp t_1 varies.

Table 3. Effects of Top-k Similar Contents on Performance

Dataset	Method	RMSE	REG (%)	Time (s)
Behance	*IGPP*-Top60	**7.80**	**21.43**	**0.30**
	IGPP-All	31.76	84.22	0.57
Twitter	*IGPP*-Top80	**15.82**	**32.98**	**0.15**
	IGPP-All	40.53	88.32	0.22

fluctuations. So, we choose the k with smaller errors—that is, $k = 60$ for Behance and $k = 80$ for Twitter.

In addition, we compare *IGPP* with the top-k similar contents and *IGPP* with all history contents. The comparison results in Table 3 show a strong indication to the effectiveness and efficiency of selecting top-k similar contents.

The effect of the observable period t_1. We evaluate the effect of the observable period t_1 on *IGPP*. To be specific, for the observable period t_1 of p_{m+1} from 2 to 15, we run *IGPP*-4h to predict group-level popularity in next $q - 15$ timestamps—that is, 45 timestamps on Behance and 9 timestamps on Twitter. The results are presented in Figure 8, and the timestamps size is 4 hours. There are two main findings. First, the REG almost decreases linearly with the increase of t_1 for the two datasets. It indicates that the longer the observation time of the target content, the higher the prediction accuracy. Second, the running time decreases slightly with the increase of t_1. In later experiments, we set $t_1 = 0.2 \times q$ (i.e., $t_1 = 12$ on Behance and $t_1 = 5$ on Twitter) and predict the group-level popularity in the remaining timestamps.

The effect of different groups. We conduct experiments of *IGPP*-4h using four different groups as mentioned in Section 4.1. Table 4 shows the results for the two datasets. We can see that *IGPP* based on the network-constrained popularity graph G^* (i.e., *IGPP* in G^*) has the smallest REG, followed

Table 4. Impact of Different Grouping Methods

Dataset	Group Methods	REG (%)	Average Running Time (s)
Behance	*IGPP* in G	23.99	0.30
	IGPP in G^s	21.96	0.31
	***IGPP* in G^***	**21.43**	0.30
	IGPP in G^r	26.53	**0.29**
Twitter	*IGPP* in G	39.32	0.18
	IGPP in G^s	37.78	0.16
	***IGPP* in G^***	**32.98**	**0.15**
	IGPP in G^r	42.24	0.17

Table 5. *IGPP* with Δ, *IGPP* with θ for Behance

	Methods	RMSE	REG (%)	Time (s)	Average Restarting Times
***IGPP*-Δ**	*IGPP*-4h	7.80	21.43	0.30	48
	IGPP-*8h*	9.02	24.47	0.16	24
	IGPP-12h	10.26	27.64	0.12	16
	IGPP-16h	11.48	30.72	0.09	12
	IGPP-20h	12.69	33.80	0.07	9
***IGPP*-θ**	*IGPP*-1‰	7.86	21.55	0.36	40.38
	IGPP-3‰	8.41	22.91	0.36	29.96
	IGPP-5‰	9.01	24.48	0.36	24.19
	IGPP-10‰	10.41	28.17	0.36	17.00
	IGPP-15‰	11.67	31.59	0.36	13.38
	IGPP-20‰	12.85	34.75	0.36	11.11
	IGPP-30‰	14.89	40.22	0.36	8.38
***IGPP*-σ**	*IGPP*-5%	7.77	21.35	0.30	46.96
	IGPP-8%	7.79	21.38	0.29	46.32
	IGPP-10%	7.82	21.44	0.29	45.12
	IGPP-12%	7.89	21.61	0.28	43.19
	IGPP-15%	8.08	22.04	0.25	39.18
	IGPP-18%	8.37	22.70	0.23	34.43
	IGPP-20%	8.61	23.29	0.21	30.97
	IGPP-25%	9.33	25.05	0.17	23.24
	IGPP-30%	10.19	27.22	0.14	16.99
	IGPP-35%	11.14	29.68	0.12	12.76
	IGPP-40%	12.15	32.34	0.10	10.18

by *IGPP* in G^S. Because G^* considers both the user networks G and users' historical activities G^S, user groups based on G^* are more homogeneous. *IGPP* with the random grouping method (*IGPP* in G^r) costs shorter time than others sometimes, but it has the largest REG. Therefore, we run experiments with *IGPP* based on a network-constrained popularity graph in later experiments.

The effect of the thresholds on restarting strategies. We evaluate the effect of the thresholds on *IGPP* with different restarting strategies with varying the thresholds (i.e., Δ from 4h to 20h, θ from 0.5% to 3%, σ from 5% to 50%). Experimental results are show in Tables 5 and 6, respectively.

IGPP with Δ (*IGPP*-Δ). As the time interval Δ increases, RMSE and REG of *IGPP*-Δ increase, but the running time and the number of restarts decrease. This is because when Δ increases, the number $(q - t_1)/\Delta$ of restarts decreases and the running time decreases greatly; at the same time, the cumulative error in the time interval Δ increases and the prediction errors increase.

IGPP with θ (*IGPP*-θ). With the increase of θ, its number of restarts decreases, whereas RMSE and REG increase. This is because as θ increases, the tolerance to the relative cumulative error $\mathcal{I}$ increases, which leads to the decrease of the number of restarts and the increase of errors. In

Table 6. *IGPP* with Δ, *IGPP* with θ for Twitter

	Methods	RMSE	REG (%)	Time (s)	Average Restarting Times
IGPP-Δ	*IGPP*-4h	15.82	32.98	0.15	19
	IGPP-8h	19.09	40.13	0.07	9
	IGPP-12h	22.28	47.06	0.05	6
	IGPP-16h	25.86	54.30	0.04	4
	IGPP-20h	28.25	59.69	0.03	3
IGPP-θ	*IGPP*-1‰	17.25	36.22	0.17	17.04
	IGPP-3‰	19.27	40.49	0.17	15.48
	IGPP-5‰	20.94	43.75	0.17	14.50
	IGPP-10‰	23.57	48.94	0.17	12.95
	IGPP-15‰	25.04	51.62	0.17	12.03
	IGPP-20‰	25.95	53.63	0.17	11.30
	IGPP-30‰	27.82	56.89	0.17	10.07
IGPP-σ	*IGPP*-5%	15.63	32.64	0.14	17.81
	IGPP-8%	15.68	32.76	0.14	17.56
	IGPP-10%	15.74	32.90	0.14	17.34
	IGPP-12%	15.82	33.05	0.14	17.05
	IGPP-15%	16.00	33.43	0.13	16.54
	IGPP-18%	16.21	33.88	0.13	15.88
	IGPP-20%	16.36	34.26	0.13	15.40
	IGPP-25%	17.04	35.64	0.12	14.01
	IGPP-30%	17.83	37.33	0.11	12.57
	IGPP-35%	18.77	39.36	0.10	11.26
	IGPP-40%	19.79	41.49	0.10	10.21

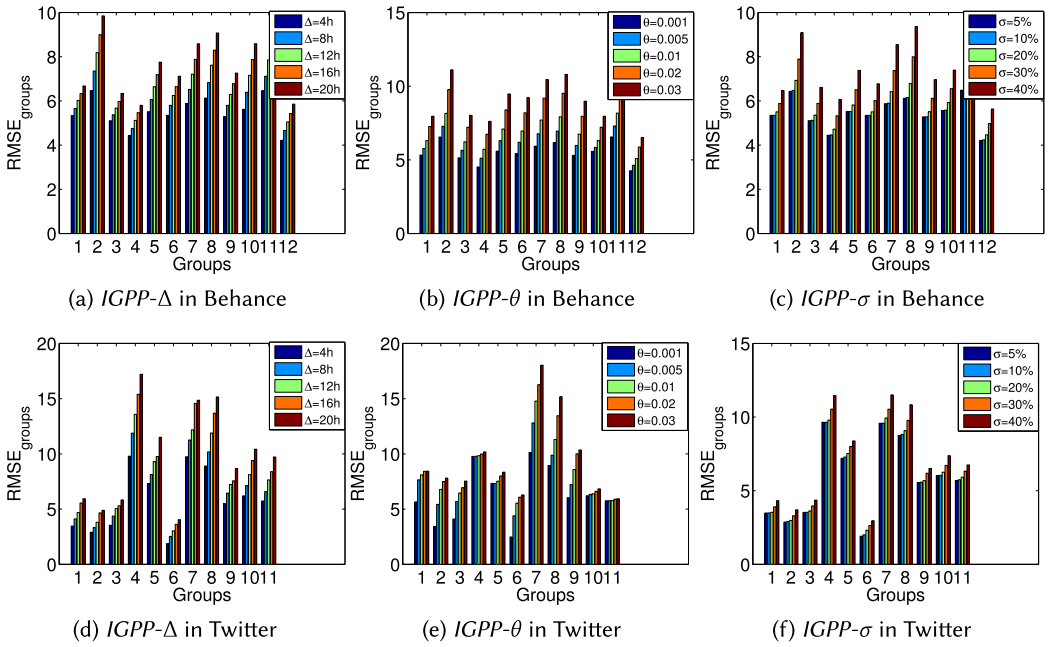
addition, the running time is basically unchanged with θ. This is because *IGPP*-θ has to compute CP decomposition, incremental CP decomposition, and $\mathcal{I}$ to decide whether to restart CP decomposition for each new data.

IGPP with σ (*IGPP*-σ). With the increase of σ, its number of restarts decreases, its average running time decreases, and RMSE and REG increase.

Fine-grained study on the effect of the thresholds on restarting strategies. We further conduct fine-grained study on the effect of the thresholds of our methods from two perspectives: the prediction performance over different groups and that over different timestamps.

We first use the metric of $RMSE_{groups}$ to study the error on each group of *IGPP* with different restarting methods with varying the thresholds. The results are shown in Figure 9. There are two main findings. First, from Figure 9, we observe that as thresholds (i.e., Δ, θ and σ) increase, $RMSE_{groups}$ of *IGPP* with three restarting strategies increase, respectively. Second, the $RMSE_{groups}$ in different groups is very different. For example, for Twitter, $RMSE_{groups}$ of the sixth group is far less than that of the seventh group for all variants of *IGPP*. We analyze the reason and find that this is because the magnitude of the $RMSE_{groups}$ is related to the true value, and true popularity in the sixth group is far less than that in the seventh group.

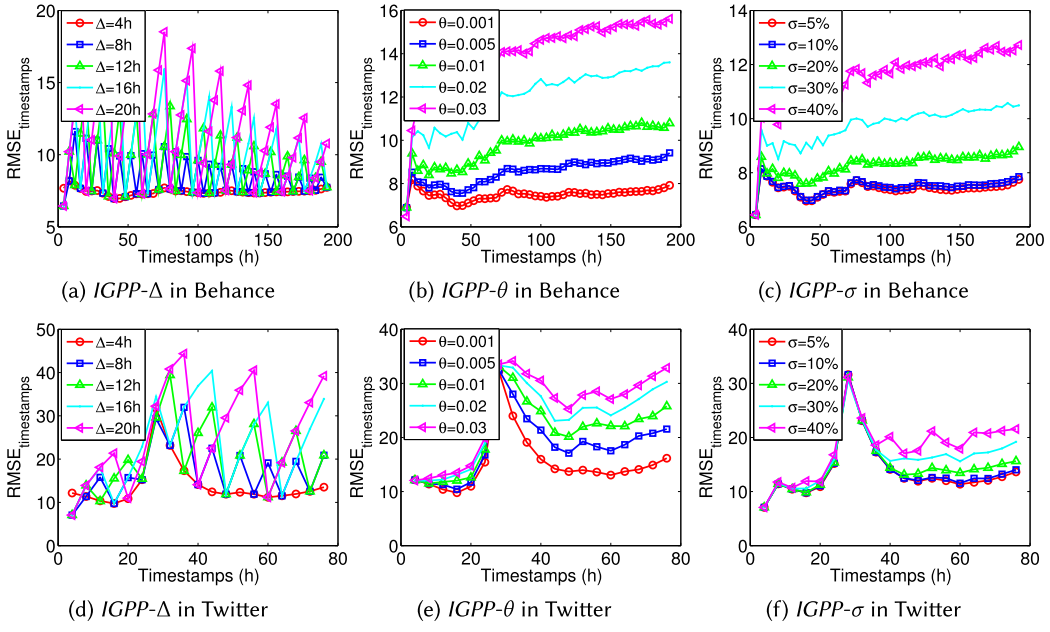
Next, we use the metric of $RMSE_{timestamps}$ to study the error at different timestamps of *IGPP* with three restarting methods with varying the thresholds. The results are shown in Figure 10. There are two main findings. First, from Figure 10(a) and (d), we find that as the prediction time increases, the $RMSE_{timestamps}$ curve of *IGPP*-4h remains stable, and the others have many peaks and valleys regularly, whose valleys are on the $RMSE_{timestamps}$ curve of *IGPP*-4h. Because *IGPP*-4h restarts the CP decomposition every 4 hours (i.e., every timestamps), it has the smallest $RMSE_{timestamps}$. Second, different from *IGPP* with Δ, *IGPP* with θ and *IGPP* with σ have the same rules. In other words, the $RMSE_{timestamps}$ curves of *IGPP* with θ and *IGPP* with σ increase approximately linearly with the prediction time on Behance (see Figure 10(b) and (c)), and there is



(a) *IGPP*-Δ in Behance (b) *IGPP*-θ in Behance (c) *IGPP*-σ in Behance

(d) *IGPP*-Δ in Twitter (e) *IGPP*-θ in Twitter (f) *IGPP*-σ in Twitter

Fig. 9. $RMSE_{groups}$ over groups of *IGPP* with three restart strategies for Behance and Twitter.



(a) *IGPP*-Δ in Behance (b) *IGPP*-θ in Behance (c) *IGPP*-σ in Behance

(d) *IGPP*-Δ in Twitter (e) *IGPP*-θ in Twitter (f) *IGPP*-σ in Twitter

Fig. 10. $RMSE_{timestamps}$ over timestamps (hours) of *IGPP* with three restart strategies for Behance and Twitter.

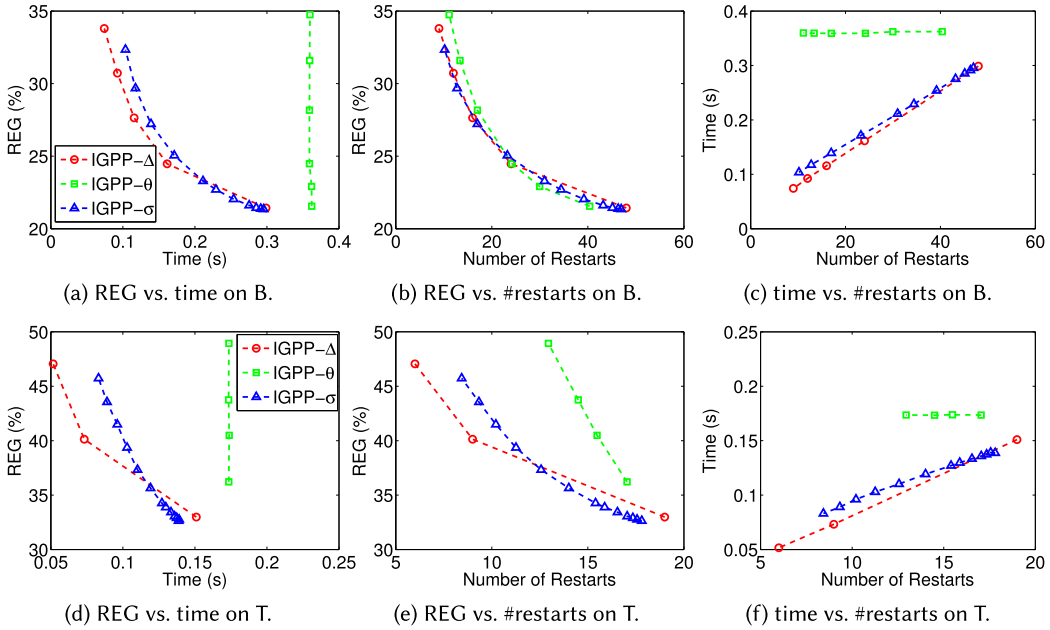


Fig. 11. The relations of REG (%), running time (seconds), and number of restarts with varying thresholds (i.e., Δ, θ, and σ) on Behance (B) and Twitter (T).

the same rule after a peak on Twitter (see Figure 10(e) and (f)). The peak on Twitter is because the magnitude of the $RMSE_{timestamps}$ is related to the true value, and true popularity at some timestamps (about 28 hours) is very high.

5.3.2 Comparison of Different Restarting Strategies. We compare different restarting strategies in detail, and we use the same parameter settings as used on the same dataset in Tables 5 and 6.

Accuracy vs. efficiency. We test the relations between the relative error (i.e., REG) and the running time, REG and the number of restarts, and the running time and the number of restarts, respectively. The results are shown in Figure 11. First of all, there are general patterns except for *IGPP-θ*: with the increase of running time, the relative error decreases; with the increase of the number of restarts, the relative error decreases and the running time increases. But the running time of *IGPP-θ* is unchanged throughout the experiments. This is because it needs to calculate the CP decomposition, ICP, and cumulative error for each new data, no matter if there are restarts or not.

For *IGPP-Δ* and *IGPP-σ*, they show different advantages with different running time. When the running time is short (i.e., less than 0.2 seconds on Behance, and less than 0.12 seconds on Twitter), or the number of restarts is small, *IGPP-Δ* has the smaller *REG* than *IGPP-σ*. That is because for determining whether to restart, *IGPP-Δ* only counts the time interval, whereas *IGPP-σ* needs to calculate the relative error of the previous results. Meanwhile, when the running time is long (i.e., larger than 0.2 seconds on Behance or 0.12 seconds on Twitter), or the number of restarts is large, *IGPP-σ* has higher accuracy than *IGPP-Δ*. We analyze the reason and find that: The time of restarting is much longer than that for determining whether to restart, therefore, the time to calculate the relative error of IGPP-σ can be ignored when the running time is long. What is more, according to the feedback of the previous results, *IGPP-σ* can restart the CP decomposition flexibly and in time.

Table 7. Prediction Error (RMSE and REG) and Average Running Time (s) of All Algorithms

Datasets	Methods	RMSE	REG (%)	Time (s)	Speedup Ratios over *IGPP*
Behance	*IGPP*-8h	9.02	24.47	**0.16**	**70.81**×
	IGPP-1‰	7.86	21.55	0.36	31.47×
	IGPP-10%	**7.82**	**21.44**	0.29	39.07×
	GPOP	11.96	36.60	11.33	1.00×
	CP	35.08	96.27	0.19	59.63×
Twitter	*IGPP*-8h	19.09	40.13	**0.07**	**90.86**×
	IGPP-1‰	17.25	36.22	0.17	37.41×
	IGPP-10%	**15.74**	**32.90**	0.14	45.43×
	GPOP	30.21	69.23	6.36	1.00×
	CP	47.41	99.69	0.13	48.92×

Comparing *IGPP*-Δ and *IGPP*-σ in detail, we find that when consumers are more concerned about efficiency, *IGPP*-Δ is better; when consumers are more concerned about accuracy, *IGPP*-σ is better. That is because *IGPP*-Δ has higher accuracy when the running time is short, whereas *IGPP*-σ has higher accuracy when the running time is long.

Discussion. By comparing *IGPP* with three restarting methods, we discuss their threshold settings and applicable scenarios.

For *IGPP*-Δ, its threshold Δ mainly depends on the speed of increased data. To reduce the number of restarts while keeping high accuracy, a large Δ is suitable when data increases slowly; a small Δ is suitable when data increases rapidly.

For *IGPP*-θ, with the increase of the number of restarts, its relative error decreases and its running time is constant. So, the smaller its threshold θ, the larger its number of restarts and the better its performance.

For *IGPP*-σ, it is a purpose-driven method, which exploits the previous feedback to determine flexibly whether to restart. Its threshold depends on the consumer's tolerance—that is, the consumer's requirements for prediction accuracy. The smaller its threshold, the smaller its relative prediction error.

In the remaining experiments, we set $\Delta = 8h$, $\theta = 0.001$, and $\sigma = 10\%$ as the default setting for *IGPP* with three restarting strategies, respectively.

5.3.3 Comparative Studies. We compare *IGPP* with different restart strategies in the default setting and baselines in terms of prediction accuracy and efficiency. We perform the same preprocess, and set the same parameters for our methods and the baselines. Especially, $k = 10$ for the baseline of *GPOP*, because when $k = 10$, *GPOP* has the highest accuracy [14]. The experimental results on the two datasets are displayed in Table 7, where we also show the speedup ratios of all algorithms over *GPOP* for comparison.

From Table 7, we can see that all variants of *IGPP* outperform the baselines of *GPOP* and CP. They achieve smaller RMSE and REG, and take shorter running time than the baselines. This is because our methods reuse the previous results and reduce computation by incremental updates, and the restarting strategies reduce the cumulative error caused by the approximation of incremental prediction. *GPOP* costs the longest running time, because it has to conduct CP decomposition multiple times on tensors for the hierarchical prediction. To be specific, *IGPP*-8h takes the shortest running time without affecting the accuracy seriously; *IGPP*-10% has the smallest prediction error without affecting the efficiency seriously. The running time of the CP decomposition is close to that of *IGPP*-8h, but the errors are the biggest among all methods. In general, *IGPP*-h8, *IGPP*-0.001, and

IGPP-10% run up to 90.86×, 37.41×, and 45.43× faster than *GPOP* while having lower prediction errors.

5.4 Summary of Experiments

The main findings are summarized as follows:

(1) *IGPP* with different restart strategies outperforms the baselines. To be specific, *IGPP*-8h, *IGPP*-0.001, and *IGPP*-10% run up to 90.86×, 37.41×, and 45.43× faster than the state-of-the-art group-level popularity prediction method *GPOP* while having lower prediction errors.

(2) By comparing *IGPP* with different restarting strategies, we find that *IGPP*-Δ is better for consumers who are more concerned about efficiency; *IGPP*-σ is better for consumers who are more concerned about accuracy. What is more, *IGPP*-σ can restart the CP decomposition flexibly and in time based on the feedback of previous results.

(3) We check the impacts of parameters on the performance of *IGPP*. As the number of similar contents k increases, the prediction errors of *IGPP* first decrease and then stabilize. As the observable period t_1 increases, the prediction errors of *IGPP* decrease. This means that the longer the observation time of the target content, the higher the prediction accuracy, which is consistent with the real-world scenarios.

(4) We test the impacts of groups on the performance of *IGPP*. *IGPP* with user groups by clustering on the network-constrained popularity graph G^* has the smallest error, because G^* combines both the user networks and user historical activities, and the user groups are more homogeneous.

6 CONCLUSION

In this article, we identify a novel problem of incremental group-level popularity prediction, and propose the *IGPP* model to solve it efficiently and effectively. It has two main steps: incrementally predicting by exploiting incremental CP decomposition and restarting CP decomposition to reduce cumulative error. Extensive empirical results demonstrate that *IGPP* outperforms other baselines in terms of prediction accuracy and running time. In current work, we mainly focus on exploring the dynamic diffusion over the time dimension. In future work, we would like to explore more general incremental approaches in OSNs, which can also model the changing groups over time. We are also interested in applying our incremental approach to more applications in big data environments.

APPENDIX

A GROUP USERS BASED ON DIFFERENT INFORMATION

The grouping goal is to divide that user into homogeneous groups with comparable size based on their interests and interactions. That is, the behaviors and interests of users should be similar within the same group and different across different groups; These groups have comparable sizes, in order to avoid the large computational cost when there are many tiny groups with only a few users. In [14], the authors define a network-constrained popularity graph G^* (see Figure 12 (c)), which is obtained by weighting of the users' historical activities network and the user network (see Figure 12 (a)). Then, they use the multilevel k-way partitioning algorithm [20] on G^* for graph clustering.

To be specific, given a user set $V = \{v_1, \ldots, v_n\}$ of user graph $G = (V, E)$, and the users' historical activities, we can first construct a popularity graph $G^S = (V^S, E^S, N^S, W^S)$, which is a weighted undirected bipartite graph (see Figure 12 (b)). In G^S, the popularity vertex set $S = \{s_{11}, \ldots, S_{mq}\}$ is the set of all combinations of m contents and q timestamps. So, vertex set $V^S = V \cup S$, vertex weights $N^S = N_V^* \cup N_S^*$, where user vertex v_j has weight as $N_{v_j}^S = 1$, while popularity vertex $s_{i,t}$ without weight $N_{s_{i,t}}^S = 0$. The edge set and edge weights are E^S and W^S, respectively. For

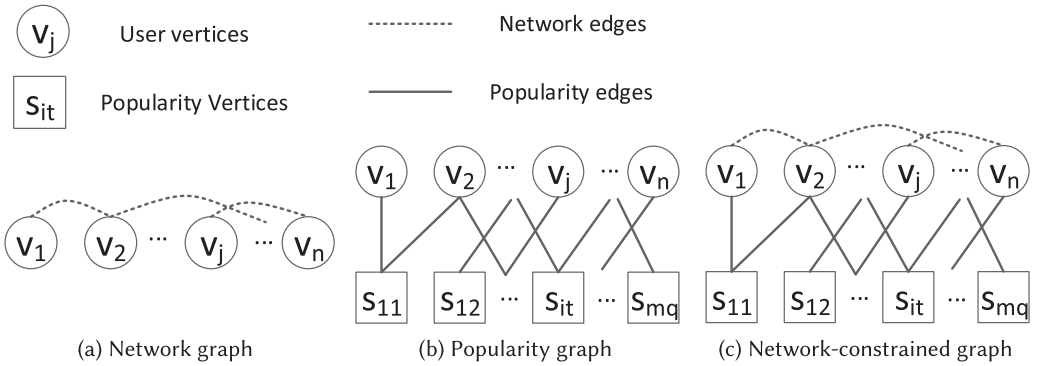


Fig. 12. Different graphs [14].

$\forall v_j \in V, \forall s_{it} \in S$, when user v_j has reacted to content p_i after the first t timestamps since p_i was created, there is an edge between v_j and s_{it}, with weight as the number of reacting to content p_i still timestamp t.

Combing G with G^S, we can construct a network-constrained popularity graph $G^* = (V^*, E^*, N^*, W^*)$ (Figure 12 (c)) with vertex set $V^* = V^S$, edge set $E^* = E^S \cup E$, vertex weights $N^* = N^S$, and edge weights $W^* = W^S \cup W$, where W is the edge weights of G, and $\forall (v_i, v_j) \in E$, $W^*_{v_i,v_j} = W_{v_i,v_j} = 1$. Interested readers may refer to [14] for grouping users in details.

In order to study the effect of different groups on the *IGPP* method, we construct an user graph G, a popularity graph G^S and a network-constrained popularity graph G^*, respectively. Then, we obtain three different groups by using the multilevel k-way partitioning algorithm, which is scalable and supports groups with comparable sizes. In addition, we group users randomly. We test the impacts of different groups via experiments.

REFERENCES

[1] Behance. net. 2018. Behance.net Social Network. Retrieved July 31, 2021 from http://www.behance.net/.
[2] Nuno Moniz and Luis Torgo. 2019. A review on web content popularity prediction: Issues and open challenges. *Online Social Networks and Media* 12 (2019), 1–20.
[3] Brett W. Bader and Tamara G. Kolda. 2015. MATLAB Tensor Toolbox Version 2.6. Retrieved July 31, 2021 from http://www.sandia.gov/~tgkolda/TensorToolbox/.
[4] Eytan Bakshy, Dean Eckles, Rong Yan, and Itamar Rosenn. 2012. Social influence in social advertising: Evidence from field experiments. In *Proc. ACM EC*. ACM, New York, NY, 146–161.
[5] Peng Bao, Hua-Wei Shen, Junming Huang, and Xue Qi Cheng. 2013. Popularity prediction in microblogging network: A case study on Sina Weibo. In *Proc. ACM WWW*. ACM, New York, NY, 177–178.
[6] Peng Bao, Hua Wei Shen, Xiaolong Jin, and Xue Qi Cheng. 2015. Modeling and predicting popularity dynamics of microblogs using self-excited Hawkes processes. In *Proc. ACM WWW*. 9–10.
[7] Qi Cao, Huawei Shen, Hao Gao, Jinhua Gao, and Xueqi Cheng. 2017. Predicting the popularity of online content with group-specific models. In *Proc. ACM WWW*. ACM, New York, NY, 765–766.
[8] Justin Cheng, Lada Adamic, P. Alex Dow, Jon Michael Kleinberg, and Jure Leskovec. 2014. Can cascades be predicted? In *Proc. ACM WWW*. ACM, New York, NY, 925–936.
[9] Jinhua Gao, Huawei Shen, Shenghua Liu, and Xueqi Cheng. 2016. Modeling and predicting retweeting dynamics via a mixture process. In *Proc. WWW*. 33–34.
[10] Xiaofeng Gao, Zhenhao Cao, Sha Li, Bin Yao, Guihai Chen, and Shaojie Tang. 2019. Taxonomy and evaluation for microblog popularity prediction. *ACM Transactions on Knowledge Discovery from Data* 13, 2 (2019), Article 15, 40 pages.
[11] Fei Hao, Shuai Li, Geyong Min, Hee-Cheol Kim, Stephen S. Yau, and Laurence T. Yang. 2015. An efficient approach to generating location-sensitive recommendations in ad-hoc social network environments. *IEEE Transactions on Services Computing* 8, 3 (2015), 520–533.

[12] Fei Hao, Doo-Soon Park, Geyong Min, Young-Sik Jeong, and Jong-Hyuk Park. 2016. k-Cliques mining in dynamic social networks based on triadic formal concept analysis. *Neurocomputing* 209 (2016), 57–66.
[13] Johan Håstad. 1990. Tensor rank is NP-complete. *Journal of Algorithms* 11, 4 (1990), 644–654.
[14] Minh X. Hoang, Xuan-Hong Dang, Xiang Wu, Zhenyu Yan, and Ambuj K. Singh. 2017. GPOP: Scalable group-level popularity prediction for online content in social networks. In *Proc. ACM WWW*. 725–733.
[15] Chunli Huang, Wenjun Jiang, Jie Wu, and Guojun Wang. 2020. Personalized review recommendation based on users' aspect sentiment. *ACM Transactions on Internet Technology* 20, 4 (2020), Article 42, 26 pages.
[16] Junming Huang, Chao Li, Wen-Qiang Wang, Hua-Wei Shen, Guojie Li, and Xue-Qi Cheng. 2014. Temporal scaling in information propagation. *Scientific Reports* 4 (2014), 5334.
[17] W. Jiang, J. Wu, F. Li, G. Wang, and H. Zheng. 2016. Trust evaluation in online social networks using generalized flow. *IEEE Transactions on Computers* 65, 3 (2016), 952–963.
[18] Wenjun Jiang, Jie Wu, Guojun Wang, and Huanyang Zheng. 2014. FluidRating: A time-evolving rating scheme in trust-based recommendation systems using fluid dynamics. In *Proc. IEEE INFOCOM*. 1707–1715.
[19] W. Jiang, J. Wu, G. Wang, and H. Zheng. 2016. Forming opinions via trusted friends: Time-evolving rating prediction using fluid dynamics. *IEEE Transactions on Computers* 65, 4 (2016), 1211–1224.
[20] George Karypis and Vipin Kumar. 1998. Multilevel k-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed Computing* 48, 1 (1998), 96–129.
[21] Su-Do Kim, Sung-Hwan Kim, and Hwan-Gue Cho. 2011. Predicting the virtual temperature of web-blog articles as a measurement tool for online popularity. In *Proc. IEEE CIT*. IEEE, Los Alamitos, CA, 449–454.
[22] Tamara G. Kolda and Brett W. Bader. 2009. Tensor decompositions and applications. *SIAM Review* 51, 3 (2009), 455–500.
[23] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. 2010. What is Twitter, a social network or a news media? In *Proc. ACM WWW*. 591–600.
[24] Kristina Lerman and Tad Hogg. 2010. Using a model of social dynamics to predict popularity of news. In *Proc. ACM WWW*. 621–630.
[25] Yuchen Li, Ju Fan, Yanhao Wang, and Kian-Lee Tan. 2018. Influence maximization on social graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering* 30, 10 (2018), 1852–1872.
[26] Yasuko Matsubara, Yasushi Sakurai, Christos Faloutsos, Tomoharu Iwata, and Masatoshi Yoshikawa. 2012. Fast mining and forecasting of complex time-stamped events. In *Proc. ACM SIGKDD*. ACM, New York, NY, 271–279.
[27] Zhongchen Miao, Kai Chen, Yi Fang, Jianhua He, Yi Zhou, Wenjun Zhang, and Hongyuan Zha. 2017. Cost-effective online trending topic detection and popularity prediction in microblogging. *ACM Transactions on Information Systems* 35, 3 (2017), Article 18, 36 pages.
[28] Zhongchen Miao, Kai Chen, Yi Zhou, Hongyuan Zha, and Wenjun Zhang. 2015. Online trendy topics detection in microblogs with selective user monitoring under cost constraints. In *Proc. IEEE ICC*.
[29] Lukás Neumann, Andrew Zisserman, and Andrea Vedaldi. 2019. Future event prediction: If and when. In *Proc. IEEE CVPR*. IEEE, Los Alamitos, CA, 2935–2943.
[30] Evangelos E. Papalexakis, Christos Faloutsos, and Nicholas D. Sidiropoulos. 2017. Tensors for data mining and data fusion: Models, applications, and scalable algorithms. *ACM Transactions on Intelligent Systems and Technology* 8, 2 (2017), 16.
[31] Henrique Pinto, Jussara M. Almeida, and Marcos A. Gonçalves. 2013. Using early view patterns to predict the popularity of YouTube videos. In *Proc. ACM WSDM*. ACM, New York, NY, 365–374.
[32] Felix Salfner, Maren Lenk, and Miroslaw Malek. 2010. A survey of online failure prediction methods. *ACM Computing Surveys* 42, 3 (2010), Article 10, 42 pages.
[33] Hua Wei Shen, Dashun Wang, Chaoming Song, and Albert-Laszlo Barabasi. 2014. Modeling and predicting popularity dynamics via reinforced poisson processes. In *Proc. AAAI*. 291–297.
[34] Gabor Szabo and Bernardo A. Huberman. 2010. Predicting the popularity of online content. *Communications of the ACM* 53, 8 (2010), 80–88.
[35] Alexandru Tatar, Panayotis Antoniadis, Marcelo Dias De Amorim, and Serge Fdida. 2012. Ranking news articles based on popularity prediction. In *Proc. IEEE/ACM ASONAM*. IEEE, Los Alamitos, CA, 106–110.
[36] Alexandru Tatar, Panayotis Antoniadis, Marcelo Dias De Amorim, and Serge Fdida. 2014. From popularity prediction to ranking online news. *Social Network Analysis and Mining* 4, 1 (2014), 174.
[37] Alexandru Tatar, Marcelo Dias De Amorim, Serge Fdida, and Panayotis Antoniadis. 2014. A survey on predicting the popularity of web content. *Journal of Internet Services and Applications* 5, 1 (2014), 8.
[38] Alexandru Tatar, Jérémie Leguay, Panayotis Antoniadis, Arnaud Limbourg, Marcelo Dias de Amorim, and Serge Fdida. 2011. Predicting the popularity of online articles based on user comments. In *Proc. ACM WIMS*. ACM, New York, NY, 67.
[39] Jingjing Wang, Wenjun Jiang, Kenli Li, and Keqin Li. 2021. Reducing cumulative errors of incremental CP decomposition in dynamic online social networks. *ACM Transactions on Knowledge Discovery from Data* 15, 3 (2021), Article 1, 33 pages.

[40] Yanhao Wang, Qi Fan, Yuchen Li, and Kian-Lee Tan. 2017. Real-time influence maximization on dynamic social streams. *Proc.eedings of the VLDB Endowment* 10, 7 (2017), 805–816.
[41] Yanhao Wang, Yuchen Li, Ju Fan, and Kian-Lee Tan. 2018. Location-aware influence maximization over dynamic social streams. *ACM Transactions on Information Systems* 36, 4 (2018), Article 43, 35 pages.
[42] Peike Xia, Wenjun Jiang, Jie Wu, Surong Xiao, and Guojun Wang. 2021. Exploiting temporal dynamics in product reviews for dynamic sentiment prediction at the aspect level. *ACM Transactions on Knowledge Discovery from Data* 15, 4 (2021), Article 68, 29 pages.
[43] Jie Xu, Mihaela Van Der Schaar, Jiangchuan Liu, and Haitao Li. 2015. Forecasting popularity of videos using social media. *IEEE Journal of Selected Topics in Signal Processing* 9, 2 (2015), 330–343.
[44] Jie Xu, Mihaela Van Der Schaar, Jiangchuan Liu, and Haitao Li. 2015. Timely video popularity forecasting based on social networks. In *Proc. IEEE INFOCOM*. IEEE, Los Alamitos, CA, 2308–2316.
[45] Peifeng Yin, Ping Luo, Min Wang, and Wang-Chien Lee. 2012. A straw shows which way the wind blows: Ranking potentially popular items from early votes. In *Proc. ACM WSDM*. ACM, New York, NY, 623–632.
[46] Linyun Yu, Peng Cui, Fei Wang, Chaoming Song, and Shiqiang Yang. 2015. From micro to macro: Uncovering and predicting information cascading process with behavioral dynamics. In *Proc. IEEE ICDM*. IEEE, Los Alamitos, CA, 559–568.
[47] Tauhid Zaman, Emily B. Fox, and Eric T. Bradlow. 2014. A Bayesian approach for predicting the popularity of tweets. *Annals of Applied Statistics* 8, 3 (2014), 1583–1611.
[48] Jifeng Zhang, Wenjun Jiang, Jinrui Zhang, Jie Wu, and Guojun Wang. 2021. Exploring weather data to predict activity attendance in event-based social network: From the organizer's view. *ACM Transactions on the Web* 15, 2 (2021), Article 10, 25 pages.
[49] Ziwei Zhang, Peng Cui, Jian Pei, Xiao Wang, and Wenwu Zhu. 2018. TIMERS: Error-bounded SVD restart on dynamic networks. In *Proc. AAAI*. 224–231.
[50] Liang Zhao. 2020. Event prediction in big data era: A systematic survey. arXiv:2007.09815.
[51] Fan Zhou, Xovee Xu, Goce Trajcevski, and Kunpeng Zhang. 2020. A survey of information cascade analysis: Models, predictions and recent advances. arXiv:2005.11041.
[52] Shuo Zhou, Nguyen Xuan Vinh, James Bailey, Yunzhe Jia, and Ian Davidson. 2016. Accelerating online CP decompositions for higher order tensors. In *Proc. ACM SIGKDD*. 1375–1384.

Received October 2020; revised March 2021; accepted April 2021