# Collaborative Intent Prediction with Real-Time Contextual Data

YU SUN, University of Melbourne
NICHOLAS JING YUAN, Microsoft Corporation
XING XIE, Microsoft Research
KIERAN MCDONALD, Microsoft Corporation
RUI ZHANG, University of Melbourne

Intelligent personal assistants on mobile devices such as Apple's Siri and Microsoft Cortana are increasingly important. Instead of passively reacting to queries, they provide users with brand new proactive experiences, which aim to offer the right information at the right time. It is, therefore, crucial for personal assistants to understand users' intent, i.e., what information users need now. Intent is closely related to context. Various contextual signals, including spatio-temporal information and users' activities, can signify users' intent. It is, however, challenging to model the correlation between intent and context. Intent and context are highly dynamic and often sequentially correlated. Contextual signals are usually sparse, heterogeneous, and not simultaneously available. We propose an innovative *collaborative nowcasting* model to jointly address all these issues. The model effectively addresses the complex sequential and concurring correlation between context and intent, and recognizes users' real-time intent with continuously arrived contextual signals. We extensively evaluate the proposed model with real-world data sets from a commercial personal assistant. The results validate the effectiveness the proposed model, and demonstrate its capability of handling the real-time flow of contextual signals. The studied problem and model also provide inspiring implications for new paradigms of recommendation on mobile intelligent devices.

CCS Concepts: •**Information systems** →**Recommender systems;** *Collaborative filtering; Personalization;* •**Human-centered computing** →**Personal digital assistants;** •**Computing methodologies** →*Factor analysis;* Online learning settings;

Additional Key Words and Phrases: Proactive experiences, personal assistants, nowcasting, streaming context

**ACM Reference format:**
Yu Sun, Nicholas Jing Yuan, Xing Xie, Kieran McDonald, and Rui Zhang. 2017. Collaborative Intent Prediction with Real-Time Contextual Data. *ACM Transactions on Information Systems* 1, 1, Article 1 (January 2017), 33 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

A preliminary version (Sun et al. 2016b) of this article appeared in the Proceedings of the 25th International World Wide Web Conference (WWW 2016). Rui Zhang is supported by ARC Future Fellow project FT120100832 and Australian Research Council (ARC) Discovery Project DP130104587.
Authors' addresses: Y. Sun and R. Zhang, Department of Computing and Information Systems, University of Melbourne, Melbourne, Victoria, Australia; emails: {sun.y, rui.zhang}@unimelb.edu.au; N. J. Yuan, Microsoft Corporation, Suzhou, Jiangsu, China; email: nicholas.yuan@microsoft.com; X. Xie, Microsoft Research, Beijing, China; email: xing.xie@microsoft.com; Kieran McDonald, Microsoft Corporation, Redmond, Washington, USA; email: kieran.mcdonald@microsoft.com.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2017 Copyright held by the owner/author(s). Publication rights licensed to ACM. 1046-8188/2017/1-ART1 $15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION

Recently, mobile intelligent personal assistants offer a new paradigm of recommendation, in which personal assistants strive to proactively recommend "the right information at just the right time" [1] and help you "get things done" [2] even "before you ask" [3]. Some examples of such *proactive experiences* are shown in Fig. 1. From left to right, it shows the screenshot from Microsoft Cortana, Google Now, and Apple's Siri, respectively. We can see that the recommended content contains various types of information, including videos, news, traffic conditions, weather, apps, places, and many other types (e.g., calendars, stock prices, sports, events). Normally, as shown in this example, different types of information are presented by *cards*, which are the areas within the screen layout showing one type of information. Due to limited display sizes of mobile devices, usually only one or two cards can be effectively shown without users sliding up the screen canvas. This essentially requires personal assistants to determine precisely which type of information users intend to know now, i.e., what users' contemporary intent is.

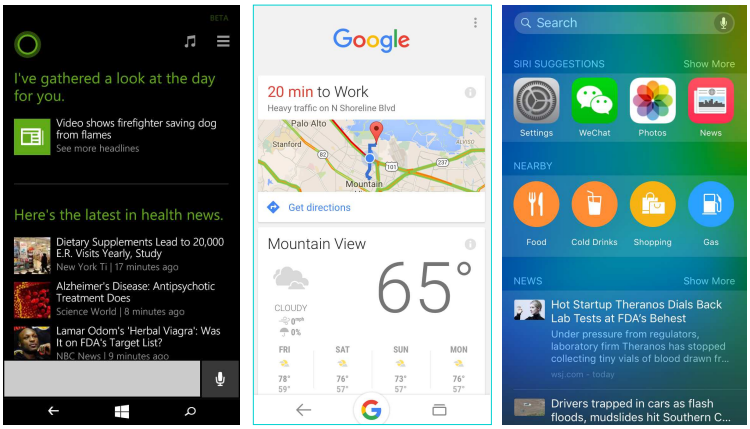



Fig. 1. Proactive experience on personal assistants

Intent is closely related to users' context, including both external context, e.g., location and time, and internal context indicated by user's current activities, e.g., usage of apps. For instance, i) When it is 6:00 p.m. in the evening and a user is in the office, then she may now intend to drive home; ii) When a user has just left a shopping center and is using Yelp, she may now intend to find a restaurant. Therefore, for personal assistants to proactively present the right information at the right time, we propose to continuously predict users' intent based on their real-time context, and we call this problem the *intent monitoring* problem.

Intent and context are dynamic and may swiftly change in a very short time, as users are usually moving around and doing different things (e.g., working and then taking a rest, driving and then dining). The correlation between intent and context exhibits complex sequential and concurring patterns. For example, when users are having breakfast, they may intend to check calendar or read news. Besides co-occurring with certain context, users' current intent may also be due to a previous context (e.g., have just left a shopping center), and conversely, the current context may result from the action triggered by a previous intent (e.g., intended to watch videos and now using

[1] http://www.google.com/landing/now/
[2] http://dev.windows.com/en-us/cortana
[3] http://www.apple.com/ios/whats-new/

Youtube). Fig. 2 illustrates such relationships among context, intent, and actions. Moreover, all contemporaneous information that is potentially correlated with intent can be included in context, which presents us highly heterogeneous contextual signals. In a real-time intent monitoring scenario, these various contextual signals are typically not available simultaneously, which in fact results in a real-time flow of contextual signals. As with traditional recommendation problems, contextual signals and intent are also very sparse. It is, therefore, a great challenge to jointly resolve all these characteristics of intent monitoring.

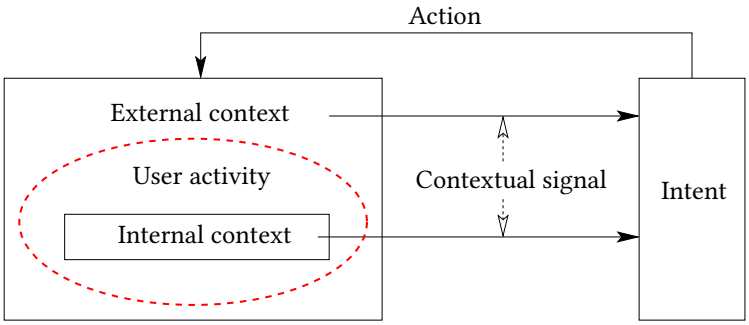



Fig. 2. Relationship between context, intent, and actions

Traditional recommendation algorithms often assume that the intent, e.g., to find interesting movies, music tracks, or books, is already or always there, and pay no attention to whether users have the intent and need such recommendations. Besides the ignorance of the existence of certain intent, existing recommendation approaches also cannot effectively tackle the characteristics of intent monitoring. For example, state-of-the-art recommendation models (Charlin et al. 2015; Koren 2009; Zhang et al. 2015) that capture the evolving of user preferences and item attributes cannot effectively solve the intent monitoring problem. This is because instead of evolving on a daily or monthly basis, the intent, together with context, may change swiftly within a very short time. Models (Jannach et al. 2015; Rendle et al. 2010) for short-term (e.g., next-basket) recommendation that depend on the similarity or co-occurring patterns between items cannot resolve the problem either because they overlook the context for intent monitoring. Although a few context-aware recommendation models (Adomavicius and Tuzhilin 2011; Liu et al. 2013) have looked at the context, the considered context usually contains only signals about physical environments, and their combinations are fixed and enumerable, e.g., 24 hours $\times$ 7 days $\times$ location types such as home and office (Karatzoglou et al. 2010; Wang et al. 2016; Zhu et al. 2015). While in intent monitoring there are numerous (internal and external) contextual signals, and the combinations of context cannot be enumerated (cf. Table 1).

Inspired by models explaining the chaotic weather and dynamic economy, we propose solving the intent monitoring problem with an innovative *collaborative nowcasting* model, which continuously predicts users' real-time intent with a streaming flow of contextual signals. Nowcasting is widely used in meteorology and macroeconomics. It is defined as the prediction of the present and very near future (cf. Section 7.1 for more details). A main difference between nowcast and forecast is the effective exploitation of *side data*, which are quantities contemporaneous with the variable of interest. Utilizing context as side data to intent, the proposed collaborative nowcasting model successfully resolves the challenge of continuously arrived contextual signals. It also addresses the sparsity and heterogeneity of contextual signals and effectively models the complex concurring and sequential correlation between context and intent. Specifically, utilizing collaborative capabilities

among users, we first summarize the shared temporal user behavior patterns among historical contextual signals with *collaborative latent factors*, and then with the continuously arrived real-time contextual signals, we generate serially correlated *personalized latent factors* to closely monitor users' real-time intent. The contribution of this paper is summarized as follows.

- We identify the intent monitoring problem, which is to closely track users' real-time intent. The problem has many real-world applications including emerging proactive experiences in mobile intelligent personal assistants.
- We propose an innovative collaborative nowcasting model, which successfully models the dynamic characteristics and complex sequential and concurring correlation between context and intent, and effectively solves the intent monitoring problem with real-time flow of contextual signals.
- We also investigate the feasibility of deploying the collaborative nowcasting model with a distributed infrastructure and discuss about the consistency of estimated latent factors.
- We extensively evaluate the proposed model in various aspects with real-world data sets from a commercial personal assistant. The results confirm the superiority of the collaborative nowcasting model over various baselines, demonstrate the model's capability of handling the real-time flow of contextual signals, and validate the feasibility of deploying the collaborative nowcasting model in parallel.

The rest of the paper is organized as follows. Section 2 formally defines the studied problem and introduces the nowcasting concept. Section 3 presents the collaborative nowcasting model. Sections 4 and 5 investigate the real-time data flow and parallel deployment, respectively. Section 6 reports the experiments. Section 7 summarizes related work and Section 8 concludes this paper.

## 2 PRELIMINARY

We first formally define the intent monitoring problem, and then introduce nowcasting concept and existing nowcasting models.

### 2.1 Problem Formulation

The intent we consider can be any potential information need of users', for example, the intent to read news, check weather or traffic conditions, find nearby restaurants, monitor stock prices, install new apps and so forth. Using the discrete time model, let $t \in \mathbb{Z}$ denote a time step. Within time step $t$, a user $u$ may have several types of intent. Let $\Gamma_t^u$ be the intent set. Given a type of intent $\gamma$, let $\mathcal{I}_{\Gamma_t^u}(\gamma)$ indicate whether user $u$ has the intent $\gamma$ within $t$, where

$$\mathcal{I}_{\Gamma_t^u}(\gamma) = \begin{cases} 1 & \text{if } \gamma \in \Gamma_t^u, \\ 0 & \text{if } \gamma \notin \Gamma_t^u. \end{cases}$$

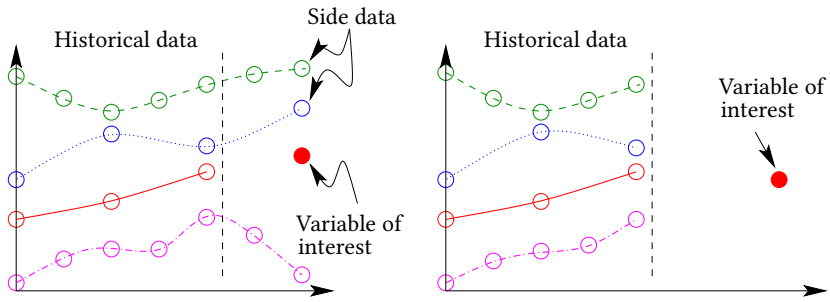
The context $\boldsymbol{x}_t^u$ of user $u$ contains any contemporaneous information potentially correlated with the user's intent, such as physical environments (e.g., spatial and temporal information) and activities users have recently performed (e.g., usage of apps, visit of venues). We then formally define the intent monitoring problem as follows.

*Definition 2.1 (Intent Monitoring).* Given a starting time $t_0$, a monitoring granularity $\Delta$, a type of intent $\gamma$ and context $\boldsymbol{x}_t^u$ of user $u$, the intent monitoring problem is to predict the value of $\mathcal{I}_{\Gamma_t^u}(\gamma)$ with context $\boldsymbol{x}_t^u$ for each time step $t$ of length $\Delta$ starting from $t_0$.

### 2.2 Nowcasting

In this section, we briefly introduce the nowcasting concept. To effectively utilize contemporaneous information relevant to an variable of interest, we need *nowcast* instead of forecast. Nowcast is

defined as the prediction of the current value of a variable of interest or its value in the very near future, e.g., two hours (hence nowcast is sometimes also referred to as short-term forecast).




(a) Nowcast (b) Forecast

Fig. 3. Difference between nowcast and forecast

The main difference between forecast and nowcast lies in the availability of *side data*. As illustrated in Fig. 3(a), side data, different from historical data, are quantities that are contemporaneous with, closely related to, and more frequently available than the variable of interest, e.g., the industrial output to the gross domestic product (GDP). For nowcasting, we may infer the value of variable of interest more accurately by utilizing both the historical and side data. While for forecasting, as shown in Fig. 3(b), all the information we can exploit is the historical data (relative to the variable of interest). To solve the intent monitoring problem, context is an important information source, which can be treated as side data to intent. Therefore, the intent monitoring problem fits into the nowcasting scenario very well. A widely used nowcasting model in macroeconomics (Giannone et al. 2005, 2008) first uses a few factors to describe the bulk movement of the time series of various macroeconomic variables, and then exploits the relationship between the factors and variable of interest for nowcasting. A direct application of this nowcasting model, however, is not sensible due to the following reasons. i) The nowcasting granularity of the above model is monthly or quarterly, which is quite different from the usually hourly granularity of the contextual recommendation scenario. ii) The macroeconomic variables in the above model are universal, while in the intent monitoring problem, context is personalized for each individual user. iii) The time series of macroeconomic variables are not sparse. Each series has a non-zero value at plenty of (usually all) time steps. However, in the intent monitoring problem, as we will see, contextual signals are often very sparse and contain many missing values in a real-time scenario. To the best of our knowledge, such a nowcasting model has never been applied to a recommendation scenario. Nevertheless, inspired by the nowcasting scenario and above model, we develop our collaborative nowcasting model.

## 3 COLLABORATIVE NOWCASTING MODEL

We first introduce the proposed model in Section 3.1 and then discuss the three steps for estimating the model parameters in Sections 3.2, 3.3, and 3.4, respectively. We also briefly discuss the model consistency in Section 3.5.

### 3.1 Model Formulation

Following existing work on nowcasting (Banbura et al. 2013, 2012; Giannone et al. 2008), we model the contextual information as stochastic processes and represent users' historical and side data as time series. Each type of contextual information is one stochastic process and produces one

Table 1. Example of a panel

| Time step | 10 a.m. | 11 a.m | 12 p.m. | 1 p.m. | Now |
|---|---|---|---|---|---|
| Facebook | 306 | 0 | 915 | 32 | 257 |
| Skype | 0 | 1853 | 0 | 0 | - |
| McDonald's | 0 | 1256 | 652 | 0 | 0 |
| IKEA | 0 | 0 | 0 | 532 | 1247 |
| Dist-to-Office | 10.4 | 8.3 | 9.1 | 21.3 | - |
| Day-of-Week | 6 | 6 | 6 | 6 | 6 |
| News Intent | 0 | 0 | 1 | 1 | ? |

series. All the available series for a user $u$ form a *panel* $\boldsymbol{X}^u$. Table 1 shows an example of a panel containing six series: two app series named Facebook and Skype, respectively; two venue series: McDonald's and IKEA; one spatial series: Dist-to-Office, and one temporal series: Day-of-Week. The monitoring intent is to read news. The monitoring granularity (i.e., time step length) is one hour and the panel shows the user's historical and side data from 10 o'clock in the morning to *now*. We denote by $x^u_{i,t}$ the $t$th random variable of the $i$th process in panel $\boldsymbol{X}^u$, which is also referred to as a *contextual signal*. The value of $x^u_{i,t}$ either indicates the length users use an app or visit a venue, or any other relevant quantities for the process such as the distance to users' office. In the sequel, we use the two words process and series interchangeably when the context is clear. Note that in the last time step (i.e., current/now), the side data may not be available in a synchronous manner, which means we may have *missing values* (denoted by the symbol "–" in the above example) for real-time nowcasting, and we will discuss in detail such real-time data flow in Section 4. In practice, there can be hundreds of series in a panel and the monitoring granularity can range from minutes to hours depending on the application at hand. Each user has contextual signals specific to herself and hence has a different number of series. We denote by $N^u$ the number of series in $\boldsymbol{X}^u$, and by $T$ the number of time steps. For expositional convenience, we will present the model using the panel of each individual user, and in the following part of this section, when the context is clear, we will drop the superscript $u$ for notational simplicity.

To obtain a parsimonious model and hence retain the model's prediction power, we assume that the dynamics of the panel are driven by a few latent factors. Let $R$ denote the number of factors for $\boldsymbol{X}$. We assume that the contextual signal $x_{i,t}$ in panel $\boldsymbol{X}$ has the following structure

$$x_{i,t} = \boldsymbol{\lambda}_i' \cdot \boldsymbol{f}_t + \xi_{i,t}, \quad 1 \le i \le N,\ 1 \le t \le T\,,$$

where $\boldsymbol{f}_t = (f_{1,t}, \ldots, f_{R,t})'$ contains the latent factors, $\boldsymbol{\lambda}_i = (\lambda_{i,1}, \ldots, \lambda_{i,R})'$ is called the factor *loading*, and $\xi_{i,t}$ is the random noise following a Gaussian distribution with zero mean and variance $\tilde{\psi}_{i,t}$. Note that the factor loading $\boldsymbol{\lambda}_i$ is only relevant to the $i$th series and the factor $\boldsymbol{f}_t$ is shared by all the series in the panel. Writing the above model in the matrix form, we have

$$\boldsymbol{x}_t = \boldsymbol{\Lambda}\boldsymbol{f}_t + \boldsymbol{\xi}_t\,, \tag{1}$$

where $\boldsymbol{\Lambda} = (\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_N)'$, $\boldsymbol{x}_t = (x_{1,t}, \ldots, x_{N,t})'$, and $\boldsymbol{\xi}_t = (\xi_{1,t}, \ldots, \xi_{N,t})'$ are the factor loading matrix, the panel column vector, and noise vector at time step $t$, respectively. We also collect the factors in a matrix $\boldsymbol{F} \in \mathbb{R}^{R\times T}$ and let $\boldsymbol{f}_t$ stand for the $t$th column of the factor matrix $\boldsymbol{F}$. To reflect the fact that users' behavior exhibits correlation, we assume that the factors across users are not independent, i.e.,

$$E(\boldsymbol{f}_t^{u_i} \boldsymbol{f}_t^{u_j\,\prime}) \neq \boldsymbol{0}, \quad 1 \le i, j \le M,\ 1 \le t \le T\,,$$

where $\boldsymbol{f}_t^{u_k}$ denotes the factors for user $u_k$ and $M$ is the number of users.

To handle the heterogeneity of contextual signals, we allow series to have different noise variances and, for model simplicity, we assume that the noise components are orthogonal across series and time steps, i.e.,

$$E(\boldsymbol{\xi}_t \boldsymbol{\xi}_t') = \boldsymbol{\Psi}_t = \mathrm{diag}(\tilde{\psi}_{1,t}, \ldots, \tilde{\psi}_{N,t}),$$
$$E(\boldsymbol{\xi}_t \boldsymbol{\xi}_{t-\delta}') = \mathbf{0}, \text{ for all } \delta > 0 .$$

To handle the missing value at the last time step and simplify the model, we set

$$\tilde{\psi}_{i,t} = \begin{cases} \psi_i , & \text{if } x_{i,t} \text{ is available} , \\ \infty , & \text{if } x_{i,t} \text{ is not available} , \end{cases}$$

which means one series has the same noise variance across different time steps and the missing value is treated as noise with a very large variance.

To fully exploit the sequential pattern and co-movement of the latent factors, we assume that the dynamics and autocorrelation of the latent factors have the following structure

$$\boldsymbol{f}_t = \boldsymbol{A}\boldsymbol{f}_{t-1} + \boldsymbol{B}\boldsymbol{\omega}_t , \tag{2}$$

where $\boldsymbol{A} \in \mathbb{R}^{R\times R}$ is the transition matrix, $\boldsymbol{B} \in \mathbb{R}^{R\times Q}$ is a matrix of full rank, and $\boldsymbol{\omega}_t$ is the white noise (i.e., $\boldsymbol{\omega}_t \sim \mathrm{WN}(0, \boldsymbol{I}_Q)$).

The given type of intent is also modeled as a stochastic process, where the value of the produced time series indicates the likelihood of a user having the intent. When the likelihood is above a chosen threshold, we say that the user has such intent. Let $\hat{y}_t$ be the value of the nowcasted likelihood at time step $t$. Assuming that the intent likelihood and contextual signals are jointly normal (which is common in our daily life), we obtain that the likelihood is a linear function of the estimated latent factors $\hat{\boldsymbol{f}}_t$ (Giannone et al. 2008), i.e.,

$$\hat{y}_t = \alpha + \boldsymbol{\beta}'\hat{\boldsymbol{f}}_t , \quad \text{for } 1 \le t \le T , \tag{3}$$

where $\alpha$ and $\boldsymbol{\beta}$ are coefficients. At this point, the model is fully established.

**Remark one**. The model described above can simultaneously address the characteristics of intent monitoring problem because of the following reasons. i) For each single user, it models the context and intent as time series in a panel and considers the within-series and across-series correlations in this panel. In this way, it fully takes into account the temporal dynamics and sequential patterns between context and intent. ii) Applying the law of parsimony (i.e., Occam's razor) (Seasholtz and Kowalski 1993), instead of estimating a full model which may introduce too much uncertainty due to a large number of parameters, the model restricts the estimation to only a few latent factors, which leads to a parsimonious model and retains the model's prediction power. iii) By considering the factor correlation across users, it is able to exploit the collaborative capabilities among users, and hence can effectively address the data sparsity problem, which is a big challenge for intent monitoring.

**Remark two**. Similar factor models have been under active research in the macroeconomic community. First, there are static factor models (Doz et al. 2011; Lawley and Maxwell 1962) which summarize the panel with a few latent factors, and then dynamic factor models (Stock and Watson 2002) which additionally consider the serial correlation (i.e., transition) between latent factors. There are also approximate dynamic factor models (Bai 2003) that considers the serial and cross-series correlation between the measurement noise. With the explosive growth of economic data, factor models dealing with panels of large sizes (Bai and Wang 2016) are investigated. When addressing real-time scenarios, models focusing on nowcasting (Banbura et al. 2012; Giannone et al. 2008) are also studied. We specific the proposed model (e.g., dynamics of factors, signal noises with different variances, etc.) with the characteristics of intent monitoring in mind (e.g., data

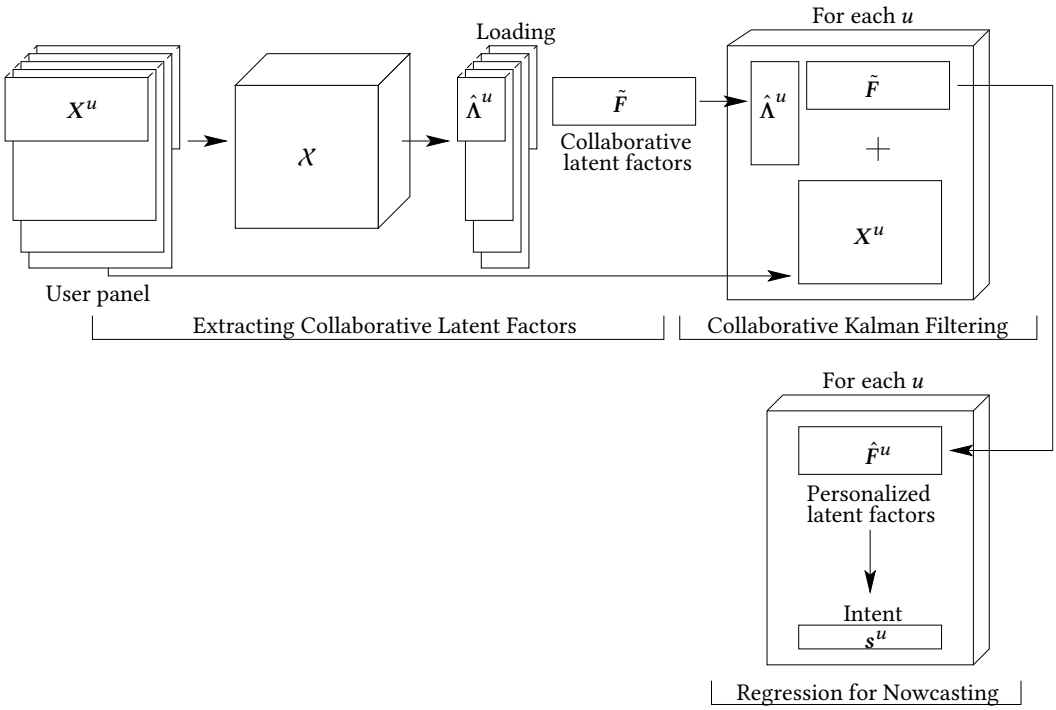



Fig. 4. Collaborative nowcasting model

sparsity, heterogeneous data sources, real-time monitoring, etc.). The proposed model is unique in that it incorporates a further dimension, the user dimension, and considers the correlation and collaboration across this dimension, which none of the existing factor models or nowcasting methods takes into account.

The remaining issue is estimating the parameters in the model. As mentioned above, one big challenge in the intent monitoring problem is that the panel (as illustrated in Table 1) is usually very sparse, and this will cause significant problems in estimating the model parameters. We propose solving this problem by exploiting the factor correlation across users, i.e., the collaborative capabilities among users. In particular, as illustrated in Fig. 4, we first i) collect the panels of all users and make these panels form a *tensor*, and then ii) use tensor decomposition techniques to extract *collaborative latent factors*, which are then iii) used in the collaborative Kalman Filtering step to obtain *personalized latent factors* and iv) finally we use the personalized factors in the nowcasting for each user.

### 3.2 Extracting Collaborative Latent Factors

To make use of the collaborative capabilities among users, we extract (i.e., estimate) latent factors by simultaneously utilizing the panels of all users via tensor decomposition. We call the obtained latent factors *collaborative latent factors*. Before discussing the methods of obtaining collaborative latent factors, we first introduce notation and some basics of tensor and tensor decomposition.

*3.2.1 Tensor and Tensor Decomposition Preliminary.* A tensor is a multi-way (i.e., multidimensional) array and the high-order generalization of vectors and matrices. As shown in Fig. 5, the

three-dimensional array, denoted by $\mathcal{X} \in \mathbb{R}^{N\times T\times M}$, is a three-way tensor. The way of a tensor is also known as modes or orders. In this paper, we will mainly focus on three-way, i.e., third-order, tensors. The general element of a three-way tensor $\mathcal{X}$ is denoted by $x_{ntm}$. Analogous to columns and rows in a matrix, the column, row and tube *fibers* of a tensor contain the elements of $x_{\cdot tm}$, $x_{n\cdot m}$, and $x_{nt\cdot}$, respectively, where the symbol "·" means all values for that subscript. Similarly, the horizontal, lateral, and frontal *slices* of a tensor consist of the elements of $x_{n\cdot\cdot}$, $x_{\cdot t\cdot}$, and $x_{\cdot\cdot m}$, respectively. For convenience, we also denote the $u$th frontal slice of $\mathcal{X}$ by $\boldsymbol{X}^u$.

Similar to matrix factorization, tensor decomposition decomposes a tensor into the sum of a few low-rank (in particular rank-one) tensors that best approximates the given tensor. Two common tensor decomposition techniques are the Tucker and CANDECOMP/PARAFAC (CP) decomposition. The CP decomposition can be treated as a special case of the Tucker decomposition. To avoid over parameterizing the model, we will mainly focus on the CP decomposition and its variants. For a given three-way tensor $\mathcal{X} \in \mathbb{R}^{N\times T\times M}$, the CP decomposition is expressed as

$$\mathcal{X} \approx \sum_{r=1}^{R} \boldsymbol{u}_r \circ \boldsymbol{v}_r \circ \boldsymbol{w}_r \,,$$

where $\boldsymbol{u}_r$, $\boldsymbol{v}_r$, $\boldsymbol{w}_r$ are vectors of size $N\times 1$, $T\times 1$, and $M\times 1$, respectively, and the symbol "$\circ$" stands for the outer product[4]. Fig. 5 illustrates the CP decomposition.

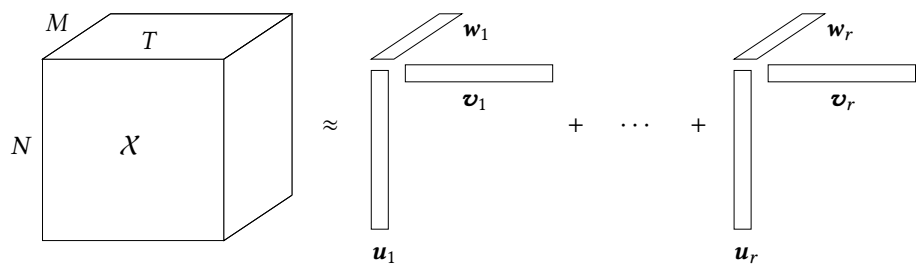



Fig. 5. CP decomposition

To obtain the CP decomposition, the following optimization problem is to be solved

$$\min \|\mathcal{X} - \hat{\mathcal{X}}\|, \text{ where } \hat{\mathcal{X}} = \sum_{r=1}^{R} \boldsymbol{u}_r \circ \boldsymbol{v}_r \circ \boldsymbol{w}_r \,,$$

where the symbol "−" denotes the element-wise subtraction (which produces a tensor $\mathcal{Z}$ with $z_{ntm} = x_{ntm} - \hat{x}_{ntm}$) and "$\|\cdot\|$" denotes the tensor norm which is (similar to the matrix Frobenius norm) defined as

$$\|\mathcal{X}\| = \sqrt{\sum_{n=1}^{N}\sum_{t=1}^{T}\sum_{m=1}^{M} x_{ntm}^2} \quad .$$

For convenience, we collect the vectors $\boldsymbol{u}_r$, $\boldsymbol{v}_r$, and $\boldsymbol{w}_r$ in matrices $\boldsymbol{U} \in \mathbb{R}^{N\times R}$, $\boldsymbol{V} \in \mathbb{R}^{T\times R}$, and $\boldsymbol{W} \in \mathbb{R}^{M\times R}$, respectively. A common method to solve the above optimization problem is the alternating least square (ALS) algorithm. ALS first initializes $\boldsymbol{U}$, $\boldsymbol{V}$, and $\boldsymbol{W}$ with singular value decomposition (SVD) method, and then fixes $\boldsymbol{U}$ and $\boldsymbol{V}$ and solves for $\boldsymbol{W}$ (which reduces the problem to an ordinary least square problem), and then fixes $\boldsymbol{U}$ and $\boldsymbol{W}$ and solves for $\boldsymbol{V}$, and so forth, until some convergence condition such as little or no change in $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{W}$ is met.

[4]The outer product of two vectors $\boldsymbol{a} = (a_1, \ldots, a_m)'$ and $\boldsymbol{b} = (b_1, \ldots, b_n)'$ is a matrix $\boldsymbol{M}$ of size $m\times n$ with the general entry $\boldsymbol{M}_{ij} = a_i b_j$, and similarly the outer product of a vector and a matrix is a three-way tensor.

*3.2.2 First Approach: CP Decomposition.* Next, we discuss the method of extracting collaborative latent factors from tensors. The simplest approach to forming a tensor from panels is to use each of the contextual information ($N$), time ($T$), and users ($M$) as one mode (i.e., dimension), as illustrated in Fig. 5. One difficulty, however, lies in forming the contextual information mode because each user $u$ has different types of contextual signals and hence a different panel size $N^u$.

It is not sensible to deploy a uniform contextual information mode (i.e., let each horizontal slice represent one type of contextual signal) by pooling together all types of contextual signals from each user. The reasons are: **i**) The types of contextual signals for all users are numerous since there are, for instance, tens of thousands of different apps and hundreds of thousands of venues from all users, which will result in an unnecessarily large tensor. **ii**) For each individual user, she may experience only a small portion of the various types of contextual signals in the pool, which means the frontal slice for this user will include a large amount of row fibers containing only zeros, and this contradicts our goal of reducing sparsity. **iii**) Unlike the user-item matrix widely used in traditional recommendations that contains target variables (e.g., ratings) to be predicted, the contextual signals are not to be completed like the user-item matrix, but to be exploited as historical/side data to extract latent factors that summarize the temporal dynamics and sequential patterns. It is thus meaningless to incorporate all types of contextual signals for a single user.

Therefore, as a first approach, we collect the individual panel of each user, assemble these panels together, and only append series containing zeros to small panels to make the contextual mode uniform in size. Let $M$ denote the number of users and

$$N = \max\{N^u | u = 1, \ldots, M\}$$

denote the number of series in the largest panel. As illustrated in Fig. 5, we obtain the tensor $\mathcal{X} \in \mathbb{R}^{N \times T \times M}$, where the first, second, and third modes, as discussed above, are the contextual information, time, and user dimensions, respectively.

After applying the CP decomposition to the obtained tensor $\mathcal{X}$, the panel of the $u$th user, i.e., the $u$th frontal slices of $\mathcal{X}$, is approximated by

$$\boldsymbol{X}^u \approx \boldsymbol{U}\boldsymbol{D}^u\boldsymbol{V}',$$

where $\boldsymbol{D}^u = \mathrm{diag}(\boldsymbol{W}_{u,1}, \ldots, \boldsymbol{W}_{u,r})$, and $\boldsymbol{U} \in \mathbb{R}^{N \times R}$, $\boldsymbol{V} \in \mathbb{R}^{T \times R}$, $\boldsymbol{W} \in \mathbb{R}^{M \times R}$ are the matrices obtained in the CP decomposition. The matrix $\boldsymbol{V}$ contains the collaborative latent factors, i.e.,

$$\tilde{\boldsymbol{F}} = \boldsymbol{V}'.$$

At this point, the latent factor matrix for user $u$ equals

$$\tilde{\boldsymbol{F}}^u = \tilde{\boldsymbol{F}},$$

and the factor loading matrix is computed by

$$\hat{\boldsymbol{\Lambda}}^u = \boldsymbol{U}^u\boldsymbol{D}^u,$$

where $\boldsymbol{U}^u$ contains the first $N^u$ rows of the matrix $\boldsymbol{U}$. The factor and loading matrices are then used in the following collaborative Kalman Filtering step.

The collaborative latent factors, different from those obtained from a single panel, contain prevalent behavioral features among a large number of users. They carry much more information on the common behavior pattern and shared temporal structure of the contextual data, which is not available from any single panel.

*3.2.3 Second Approach: PARAFAC2 Decomposition.* By making the contextual mode be of uniform size, the tensor $\mathcal{X}$ contains many manually-imposed zero elements, which bring noise into the parameter estimation procedure. To further reduce noise and data sparsity, in this method, we only assemble the panel of each user together, and make no modifications to any panel (i.e., equivalent to removing the appended zero-series from the tensor used in the CP decomposition). An example of the resulting tensor is shown in Fig. 6. In this setting, the tensor is a "jagged" tensor that contains slices of various sizes in the contextual mode.

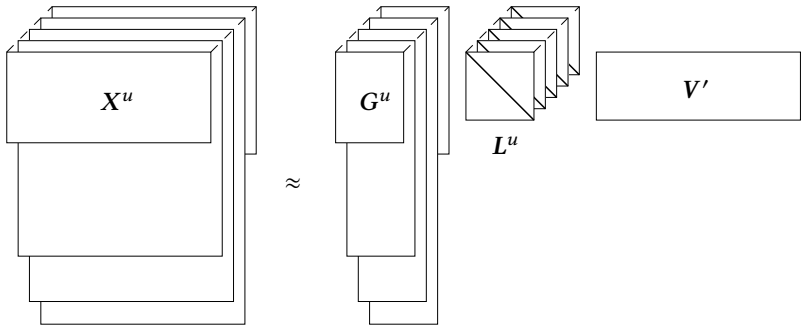



Fig. 6. PARAFAC2 decomposition

In order to obtain the collaborative latent factors, we use the PARAFAC2 (Harshman 1972) decomposition technique to perform tensor decomposition on the "jagged" tensor. PARAFAC2 is a variant of the CP decomposition that relaxes some constraints of the CP's. For a three-way tensor, the PARAFAC2 decomposition only requires two out of the three modes to have uniform sizes, which in our scenario are the time and user modes, while the third mode, i.e., the contextual mode, can be of various sizes. An illustration of the PARAFAC2 decomposition is also shown in Fig. 6. In our problem, the PARAFAC2 decomposition is equivalent to solving the optimization problem

$$\left(\tilde{\boldsymbol{F}}, \tilde{\boldsymbol{\Lambda}}^u\right)_{\{u=1,\ldots,M\}} = \min_{\boldsymbol{F},\boldsymbol{\Lambda}^u} \sum_{u=1}^{M} \|\boldsymbol{X}^u - \boldsymbol{\Lambda}^u \boldsymbol{F}\|_F^2$$

with certain uniqueness constraints (Kolda and Bader 2009), where $F$ stands for the Frobenius norm.

After decomposition, the panel for the $u$th user is approximated by

$$\boldsymbol{X}^u \approx \boldsymbol{G}^u \boldsymbol{H} \boldsymbol{L}^u \boldsymbol{V}' ,$$

where $\boldsymbol{G}^u \in \mathbb{R}^{N^u \times R}$ is an orthonormal matrix, $\boldsymbol{H} \in \mathbb{R}^{R\times R}$ is a matrix invariant to $u$, $\boldsymbol{L}^u \in \mathbb{R}^{R\times R}$ is a diagonal matrix, and $\boldsymbol{V} \in \mathbb{R}^{T\times R}$ is the matrix containing the collaborative latent factors. For the $u$th user, the initially estimated latent factors are

$$\tilde{\boldsymbol{F}}^u = \tilde{\boldsymbol{F}} = \boldsymbol{V}' ,$$

and the factor loading matrix is computed by

$$\hat{\boldsymbol{\Lambda}}^u = \boldsymbol{G}^u \boldsymbol{H} \boldsymbol{L}^u .$$

The PARAFAC2 decomposition is an effective approach because of the following reasons. i) The original structure of each panel is well approximated with no manually-imposed noise. ii) Since the temporal mode, i.e., time dimension, of the tensor is uniform across slices, PARAFAC2 is able to extract the shared temporal characteristics by utilizing such uniformity, which is vital in the intent monitoring problem. iii) The flexibility of PARAFAC2, i.e., allowing one mode to be of various

sizes, is particularly suitable for the non-uniform contextual mode, which introduces no extra constraints and hence retains more information than the CP decomposition. Extensive experiments (cf. Section 6.4) also validate the superiority of the PARAFAC2 decomposition. Therefore, we use this approach in the proposed model.

### 3.3 Collaborative Kalman Filtering

For the intent monitoring problem, it is not sufficient to utilize only the collaborative latent factors obtained from the tensor decomposition. The collaborative factors only reflect the static common structure of contextual signals. The dynamics of the factors and hence the correlation and co-movement of time series, however, are not fully taken into consideration. Moreover, the collaborative factors are extracted from the data of all users and hence are the same for all users, which is not suitable for personalized intent monitoring. Therefore, for each user $u$, we apply Kalman filter (Kalman 1960) to the collaborative factors $\tilde{\boldsymbol{F}}^u$ and the panel $\boldsymbol{X}^u$ to obtain the final estimation $\hat{\boldsymbol{F}}^u$ of the latent factors. The factors $\hat{\boldsymbol{F}}^u$ reflects both the collaborative and personalized patterns, and the static and dynamic structures of all the available data. For notational simplicity, in the sequel, we will drop the superscript $u$ as the following parameter estimation procedure is for each user.

*3.3.1 Estimating Required Parameters.* For the collaborative nowcasting model, we have obtained the estimations of the factors and loading matrix. To apply Kalman filer to each user, we first estimate the remaining parameters of Eq. 1 and 2. By applying vector autoregression (VAR) on the estimated collaborative factors, the estimations of matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are computed by

$$\hat{\boldsymbol{A}} = \sum_{t=2}^{T} \tilde{\boldsymbol{f}}_t \tilde{\boldsymbol{f}}_{t-1}' \left( \sum_{t=2}^{T} \tilde{\boldsymbol{f}}_{t-1} \tilde{\boldsymbol{f}}_{t-1}' \right) \text{ and } \hat{\boldsymbol{B}} = \boldsymbol{C}\boldsymbol{E}^{\frac{1}{2}} ,$$

respectively, where $\boldsymbol{E} \in \mathbb{R}^{Q\times Q}$ is a diagonal matrix containing the largest $Q$ eigenvalues of matrix $\boldsymbol{\Omega}$ (defined below), $\boldsymbol{C} \in \mathbb{R}^{R\times Q}$ is a matrix containing the corresponding eigenvectors, and

$$\boldsymbol{\Omega} = \frac{1}{T-1} \sum_{t=2}^{T} \tilde{\boldsymbol{f}}_t \tilde{\boldsymbol{f}}_t' - \hat{\boldsymbol{A}} \left( \frac{1}{T-1} \sum_{t=2}^{T} \tilde{\boldsymbol{f}}_{t-1} \tilde{\boldsymbol{f}}_{t-1}' \right) \hat{\boldsymbol{A}}'.$$

Let the sample covariance matrix $\boldsymbol{S}$ of the historical data (after standardized normalization) be

$$\boldsymbol{S} = \frac{1}{T} \sum_{t=1}^{T} \boldsymbol{x}_t \boldsymbol{x}_t' .$$

The covariance matrix $\boldsymbol{\Psi}$ in Eq. 1 is estimated by

$$\hat{\boldsymbol{\Psi}} = \text{diag}(\boldsymbol{S} - \boldsymbol{P}\boldsymbol{\Sigma}\boldsymbol{P}') ,$$

where $\boldsymbol{\Sigma} \in \mathbb{R}^{R\times R}$ is a diagonal matrix containing the largest $R$ eigenvalues of $\boldsymbol{S}$, $\boldsymbol{P} \in \mathbb{R}^{N\times R}$ is a matrix consisting of the corresponding eigenvectors with $\boldsymbol{P}'\boldsymbol{P} = \boldsymbol{I}$, and the "diag" means keeping only the elements at the main diagonal.

*3.3.2 Correcting the Factors with Kalman Filter.* With all required parameters at hand, we reestimate the factors by applying the Kalman filter. Let the a priori and a posteriori factors and the corresponding measurement error covariance matrices at each time step be $\tilde{\boldsymbol{f}}_t$, $\hat{\boldsymbol{f}}_t$, $\tilde{\boldsymbol{P}}_t$, and $\hat{\boldsymbol{P}}_t$, respectively. By Eq. 2, in the time update (prediction) step, the a priori factors for the next time step are computed by

$$\tilde{\boldsymbol{f}}_t = \hat{\boldsymbol{A}} \hat{\boldsymbol{f}}_{t-1} + \hat{\boldsymbol{B}} \boldsymbol{\omega}_t ,$$

and the a priori error covariance is computed by

$$\tilde{\boldsymbol{P}}_t = \hat{\boldsymbol{A}}\hat{\boldsymbol{P}}_{t-1}\hat{\boldsymbol{A}}' + \hat{\boldsymbol{B}}\hat{\boldsymbol{B}}' .$$

In the measurement update (correction) step, the Kalman gain $\boldsymbol{K}_t$ is obtained by considering the ratio of the measurement and transition error covariance and equals

$$\boldsymbol{K}_t = \tilde{\boldsymbol{P}}_t\hat{\boldsymbol{\Lambda}}'(\hat{\boldsymbol{\Lambda}}\tilde{\boldsymbol{P}}_t\hat{\boldsymbol{\Lambda}}' + \hat{\boldsymbol{\Psi}}_t)^{-1} .$$

With the Kalman gain, the a priori (collaborative) factors are corrected by utilizing the user's panel, and the corrected, i.e., personalized, factors are estimated by

$$\hat{\boldsymbol{f}}_t = \tilde{\boldsymbol{f}}_t + \boldsymbol{K}_t(\boldsymbol{x}_t - \hat{\boldsymbol{\Lambda}}\tilde{\boldsymbol{f}}_t) .$$

The a posteriori covariance used for next time step is then computed by

$$\hat{\boldsymbol{P}}_t = (\boldsymbol{I} - \boldsymbol{K}_t\hat{\boldsymbol{\Lambda}})\tilde{\boldsymbol{P}}_t .$$

The a posteriori factors $\hat{\boldsymbol{f}}_t$ are the estimated personalized latent factors we need for the next step. In practice, we can also apply the Kalman Smoother (RTS Smoother) to fully exploit all the available data. Following existing work (Sun et al. 2014), the above approach is referred to as the collaborative Kalman filtering because it uses the same latent factors extracted from the data of all users and the system parameters such as transition matrix $A$, covariance matrix $\boldsymbol{\Psi}$, etc. are estimated with such collaborative latent factors.

### 3.4 Regression for Nowcasting

The final step is to establish the relationship between the personalized latent factors and the intent, i.e., to estimate the coefficients in Eq. 3. We use the ordinary least square (OLS) regression to estimate the coefficients $\alpha$ and $\boldsymbol{\beta}$. In particular, let $\tau$ be the last time step where the intent is available (in the historical data). Let matrix $\bar{\boldsymbol{F}} = (\hat{\boldsymbol{f}}_1, \ldots, \hat{\boldsymbol{f}}_\tau)$ contain the personalized latent factors until time step $\tau$. Let the corresponding intent likelihood in the $\tau$ time steps be $\boldsymbol{y} = (y_1, \ldots, y_\tau)$. The coefficients $\alpha$ and $\boldsymbol{\beta}$ are then estimated by running OLS with $\bar{\boldsymbol{F}}$ and $\boldsymbol{y}$. The linear function of Eq. 3 is then used in the intent monitoring for following time steps. The threshold $\theta$ we use is the median of the fitted intent likelihood $\hat{y}_t$ for $1 \leq t \leq \tau$. If $\hat{y}_{\tau+\delta} > \theta$ for any $\delta > 0$, we say the user has the intent, i.e., $\mathcal{I}_{\Gamma_{\tau+\delta}}(\gamma) = 1$.

### 3.5 Discussion on the Consistency of Latent Factors

Next, we briefly discuss asymptotic properties, in particular the consistency, of the estimated latent factors. It has been proved (Bai 2003; Doz et al. 2011; Forni et al. 2009) that the latent factors, when estimated individually (i.e., for each user), are consistent estimates of the true factors when the number of time steps $T$ and size of panels $N$ approach infinity, provided that i) the initial factors and loadings are estimated by the principle component analysis (PCA), ii) the model parameters such as $\boldsymbol{A}$ and $\boldsymbol{B}$ are estimated by vector autoregression (VAR) on the initial factors, and iii) a few assumptions and conditions, which are standard in the literature (Doz et al. 2012; Stock and Watson 2002), hold such as contextual signals $x_{it}$ have uniformly bounded variance and all the eigenvalues of $\boldsymbol{\Lambda}'\boldsymbol{\Lambda}$ diverge at the same rate. For convenience, we denote this estimator by $\mathcal{M}_{indi}$.

Following the existing literature (i.e., making the same assumptions as (Doz et al. 2012) and (Stock and Watson 2002)), since we also use VAR to estimate the model parameters $\boldsymbol{A}$, $\boldsymbol{B}$, etc., the only difference between $\mathcal{M}_{indi}$ and the collaborative nowcasting model lies in the first step, i.e., estimating the initial factors. For the collaborative nowcasting model, we estimate the initial factors (i.e., collaborative latent factors) by jointly decomposing the panels of a large number of users with PARAFAC2 tensor decomposition. Such collaborative latent factors are not consistent with respect

to each user, because they represent a common structure and shared features of all users. If we can instead prove that the collaborative latent factors are consistent in terms of all users, then the factors estimated by the proposed model are overall consistent. Considering that the PCA of a panel $\boldsymbol{X}^u$ can be obtained by singular value decomposition (SVD) of $\boldsymbol{X}^u$ and that PARAFAC2 is a high-order generalization of SVD (Bro 1997; Chew et al. 2007; Kiers et al. 1999), it is viable to believe that the collaborative latent factors $\tilde{\boldsymbol{F}}$ will be consistent in terms of all users. In an empirical analysis using Monte Carlo simulation—a standard approach in the literature to empirically demonstrating consistency (Bai 2003; Doz et al. 2012; Stock and Watson 2002)—we do observe such consistency results (cf. Section 6.5.4).

In the sequel, we provide an initial comparison between PCA and PARAFAC2 and leave a rigorous and complete proof of consistency (which can be long) as future work. As mentioned above, the PCA of a panel $\boldsymbol{X}^u$ can be obtained by singular value decomposition of $\boldsymbol{X}^u$, i.e.,

$$\boldsymbol{X}^u \approx \boldsymbol{U}^{(u)}\Sigma^{(u)}\boldsymbol{V}^{(u)\prime},$$

where $\Sigma^{(u)} \in \mathbb{R}^{N\times T}$ is a rectangular diagonal matrix and contains the singular values of $\boldsymbol{X}^u$ in descending order, and $\boldsymbol{U}^{(u)} \in \mathbb{R}^{N^u\times N^u}$ and $\boldsymbol{V}^{(u)} \in \mathbb{R}^{T\times T}$ are orthonormal and contain the corresponding left and right singular vectors, respectively. Here, the superscript $(u)$ denotes that the matrices are obtained separately for each user. The initial factors, denoted by $\tilde{\boldsymbol{F}}_{pc}$, are estimated by the first $R$ rows of $\Sigma^{(u)}\boldsymbol{V}^{(u)\prime}$, i.e.,

$$\tilde{\boldsymbol{F}}_{pc}^{(u)} = \left(\Sigma^{(u)}\boldsymbol{V}^{(u)\prime}\right)_{[R]},$$

where the subscript $[R]$ denotes selecting the first $R$ rows. Comparing the SVD with PARAFAC2 (cf. Section 3.2.3), we can observe a similar decomposition of the panel in the sense that they both first obtain orthonormal and then diagonal matrices. The collaborative latent factors $\tilde{\boldsymbol{F}}$ are only different, up to a scale matrix, from $\tilde{\boldsymbol{F}}_{pc}^{(u)}$ in that $\tilde{\boldsymbol{F}}$ are shared by all users. Since $\tilde{\boldsymbol{F}}_{pc}$ are consistent with respect to each user, it is believed that $\tilde{\boldsymbol{F}}$ are consistent with respect to all users.

Another way to demonstrate the above point is to show that the personalized latent factors $\hat{\boldsymbol{F}}$ obtained by the collaborative nowcasting model converge to the final latent factors obtained by $\mathcal{M}_{indi}$. To achieve this, we can iteratively apply the second step of the collaborative nowcasting model (i.e., Kalman filtering) with its output as its input for the next iteration, i.e., to use the personalized latent factors $\hat{\boldsymbol{F}}$ as new 'initial' factors by setting $\tilde{\boldsymbol{F}} = \hat{\boldsymbol{F}}$. The loading matrix in each subsequent iteration is estimated by running VAR on $\boldsymbol{X}$ and $\tilde{\boldsymbol{F}}$, i.e.,

$$\Lambda = \boldsymbol{X}\tilde{\boldsymbol{F}}'(\tilde{\boldsymbol{F}}\tilde{\boldsymbol{F}}')^{-1},$$

and all other parameters are estimated as described in Section 3.3. We will use this method to establish another empirical evidence to support the consistency of the proposed model in the experiments.

## 4 COLLABORATIVE NOWCASTING MODEL FOR THE FLOW OF CONTEXTUAL SIGNAL

At this point, it remains unclear whether the collaborative nowcasting model can effectively address the flow of contextual signals which arrive at various speeds. It is also unclear whether the collaborative capabilities will retain and help address the missing signals/data. To the best of our knowledge, real-time data flow has never been formally specified in a recommendation scenario such as intent monitoring. Therefore, in this section, we first formally specify the data flow we focus on in Section 4.1, and then present how the collaborative nowcasting model addresses such data flow in Section 4.2.

### 4.1 Formal Specification of the Data Flow

Within each time step, various contextual signals (e.g, the focus time of certain apps and distance to home) may not be simultaneously available, as depicted by the missing values in the last column of Table 1. This phenomenon is caused by the frequency in the data collection process. The latency that a certain signal is retrieved and available in the intent monitoring/nowcasting component is determined by the frequency that the signal is sensed, processed, and sent to the monitoring component. Since we exploit a large amount of different signals and the ways of collecting these signals are very different in real-world applications (due to the various costs), the contextual signals will be available with different latency, and hence produce a real-time data flow.

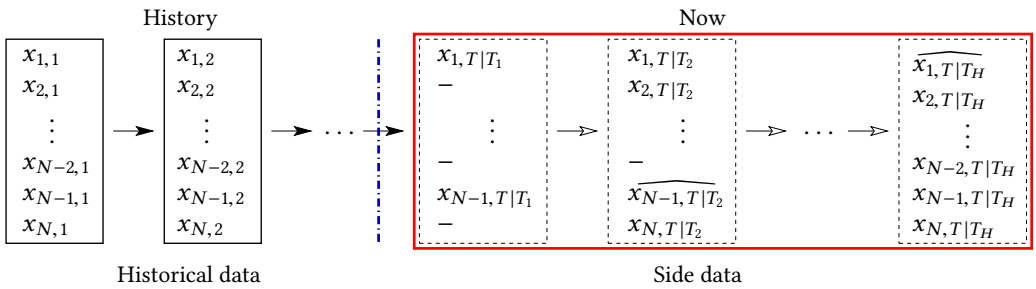



Fig. 7. Real-time data flow

To formally describe the flow of contextual signals, let $H$ be the times that the nowcasting component fetches contextual signals from various data sources within a time step. Let $T$ denote the current time step and $\mathcal{D}_k$ the set of signals that are newly released or updated up to the $k$th ($1 \leq k \leq H$) data fetch within $T$. The entire information set $\mathbb{D}_k$ at the $k$th data fetch consists of all the historical data (i.e., before $T$) and the side data $\mathcal{D}_k$ collected up to now (i.e., within $T$), i.e.,

$$\mathbb{D}_k = \{x_{it|k} \mid i = 1, \ldots, N,\ t = 1, \ldots, T_{i,k}\},$$

where $T_{i,k} = T$ for signal $x_i \in \mathbb{D}_k$ (i.e., available up to the $k$th data fetch), and $T_{i,k} = T - 1$ otherwise. Such real-time data flow is illustrated in Fig. 7. Within the last time step (i.e., current/now), as depicted by the large red rectangle on the right, at the first data fetch only signals $x_1$ and $x_{N-1}$ are available. Other signals are currently unavailable or missing, as depicted by the symbol "–". At the second data fetch, signals $x_2$ and $x_N$ are newly available, and the value of $x_{N-1}$ is updated, as depicted by $\widehat{x_{N-1}}$. At the last data fetch, with a well chosen monitoring granularity, all contextual signals will be available.

### 4.2 Collaborative Nowcasting Model for the Data Flow

*4.2.1 Collaborative Capability for Missing Data.* With the continuous arrival of contextual signals (i.e., data released and updated from various sources), users' contemporaneous intent will be increasingly perceptible. From the above model specification and factor estimation process, we can see that the collaborative nowcasting model captures the sequential and concurring patterns between context and intent as follows. First, from the last time step, it obtains 'initial' factors for the current time step by the transition of factors (cf. Eq. (2)). Then, within the current time step, it continuously corrects the initial factors to capture the effect of concurring context. The initial factors capture the sequential patterns of users' behavior because they are estimated based on their behavior at last time step. More importantly, since the whole factor transition process (e.g., transition matrix $\boldsymbol{A}$ and covariance matrix $\boldsymbol{\Psi}$ etc.) is estimated by leveraging the collaborative capabilities among users, such initial factors also capture the common behavior patterns among

users. In this way, the collaborative nowcasting model retains the collaborative capability among users and retains predictive power even when the signals are collected at a low speed which means a large amount of contextual signals for the current time step is missing. In our experiments, the collaborative nowcasting model is able to make reliable nowcast even if a major amount (e.g., 60%, cf. Section 6.5.2) of data is missing.

*4.2.2 Collaborative Kalman Filtering for the Data Flow.* Besides the initial factors estimated from the last time step, the collaborative Kalman filtering step also helps address the missing data in the data flow. Specifically, by assigning the missing data a very large measurement noise variance, the missing data almost have no effect on the obtained personalized latent factors. This is because the Kalman gain effectively balances the system noise and measurement noise, and will trust either the system transition (i.e., initial factors) or the real measurement (concurring context), depending on which is less noisy (Kalman 1960; Welch and Bishop 1995). Similarly, when there are various available contextual signals, which are subject to subsequent updates, the collaborative Kalman filtering will choose to trust more on the signals that have a reasonable measurement fluctuation at the current data fetch. Given that the measurement of mobile devices is often noisy and subject to subsequent amendments, the above property the collaborative Kalman filtering enables the proposed model to effectively handle the noisy data flow.

*4.2.3 Formal Description for Handling the Data Flow.* In summary, the collaborative nowcasting model has the ability to effectively capture the continuously increased perceptibility of users' intent by fully exploiting all the currently available information, including all historical and side data and the collaboration among users. Formally, the estimated intent likelihood $\hat{y}_{t|k}$ at the $k$th data fetch is the expected value of intent conditioned on the entire information set currently available, i.e.,

$$\hat{y}_{t|k} = \mathsf{E}[y_t|\mathbb{D}_k; \mathcal{M}],$$

where $\mathcal{M}$ denotes the proposed collaborative nowcasting model. Note that this equation abstracts the whole model including Eq. (1)–(3) and the factor estimation process. With the increasing amount of available contextual/side signals, the estimated latent factors will continuously approach to the true factors. So is the estimated intent likelihood to the true likelihood. Extensive experiments (cf. Section 6.5.2) also demonstrate such capabilities of the collaborative nowcasting model.

## 5 PARALLELIZATION OF COLLABORATIVE NOWCASTING MODEL

To enhance the nowcasting efficiency, in this section, we discuss a parallelized deployment of the collaborative nowcasting model. We first present the deployment in Section 5.1, and then analyze the communication cost of such a deployment in Section 5.2.

### 5.1 Parallel Deployment

For mobile personal assistants to provide real-time services, we propose to deploy the second and third (i.e., Kalman filtering and regression) steps of the collaborative nowcasting model (as shown in Fig. 4) onto each mobile device, as illustrated by Fig. 8. The advantage of such a deployment is more efficient nowcasting and less communication cost between the server and mobile devices. The improved efficiency is because we perform the computation of Kalman filtering and linear regression in parallel and effectively utilize the computing capabilities of mobile devices. The decreased communication cost is due to the reduced communication when we perform intent nowcasting with real-time data flow.

Such a deployment is a feasible choice because of the following reasons. i) Although mobile devices usually have limited capability of computation and need to save power, it is well known that the computation of Kalman filtering and linear regression is light-weighted, and thus such a

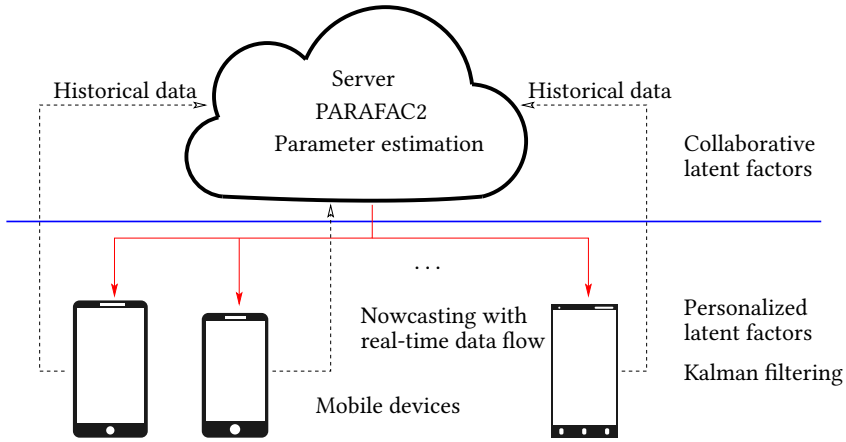



Fig. 8. System configuration of parallel collaborative intent nowcasting

deployment will not impose a heavy computation load to mobile devices. ii) The communication cost between the server and mobile devices will be lower, which is detailed in the following section.

### 5.2 Analysis of Communication Cost

Once the collaborative latent factors $\tilde{\boldsymbol{F}} \in \mathbb{R}^{R\times T'}$ (here $T'$ denotes the number of historical time steps used for parameter estimation), and other parameters $\hat{\boldsymbol{\Lambda}}^u \in \mathbb{R}^{N^u\times R}$, $\hat{\boldsymbol{A}} \in \mathbb{R}^{R\times R}$, $\hat{\boldsymbol{B}} \in \mathbb{R}^{R\times Q}$, and $\hat{\boldsymbol{\Psi}}^u \in \mathbb{R}^{N^u\times N^u}$ (which is a diagonal matrix) are obtained by the server utilizing the historical data, we only need to send these parameters to mobile devices once, and the mobile devices can provide effective nowcasting for quite a long time (e.g., approximately one week, as used in our experiments). Other communication cost is that mobile devices need to send back to the server their historical contextual signals, which can be done in a periodic and batch fashion. Therefore, the communication cost for $M$ users in such a deployment is

$$\begin{aligned} &O\left(\textstyle\sum_{u=1}^{M}(N^u\times T' + R\times T' + N^u\times R + R\times R + R\times Q + N^u)\right) \\ &= O\left(M(RT' + R^2 + RQ) + (T'+R+1)\textstyle\sum_{u=1}^{M} N^u\right) . \end{aligned}$$

Since we desire parsimonious models, $R$ and $Q$ are normally small constants (e.g., 4 and 2, respectively), therefore, the asymptotic communication cost is

$$O\left(MT' + T'\sum_{u=1}^{M} N^u\right) .$$

If instead, we deploy the centralized approach, i.e., let mobile devices send the flow of real-time data to the server and receive $\mathcal{I}_{\Gamma_t^u}(\gamma) \in \{0,1\}$ (i.e., whether a user has the intent) from the server, then to provide the same level of service for intent monitoring as the above infrastructure, the communication cost is

$$\begin{aligned} &O(\textstyle\sum_{u=1}^{M} N^u\times T' + N^u\times H\times\delta + H\times\delta) \\ &= O\left(\delta H\textstyle\sum_{u=1}^{M}(N^u+1) + T'\sum_{u=1}^{M} N^u\right) , \end{aligned}$$

where $H$ is the number of data fetches within one time step, and $\delta$ is the number of time steps we monitor the intent. Simplifying the above formula by dropping constant terms, we have the communication cost of

$$O\left(\delta H \sum_{u=1}^{M} N^u + T' \sum_{u=1}^{M} N^u\right).$$

Considering a normal scenario where on average $N^u \geq 30$ and $\delta = T'/3$, we have $\delta H \sum_{u=1}^{M} N^u \geq 10HMT'$. Since $H \geq 1$, comparing with these two communication costs, we can easily see that the former approach incurs at least one order of magnitude less communication cost than the later one.

We also empirically study the feasibility of the proposed parallel deployment, and measure the ratio of speed-up when we distribute the computation of Kalman filtering and linear regression to multiple computing units in Section 6.5.3.

## 6 EXPERIMENTS

We use the contextual recommendation task in personal assistants to empirically evaluate the collaborative nowcasting model. The experiments are conducted on a 64-bit Windows computer with a 2.8GHz Intel(R) CPU and 24GB main memory. The algorithms are implemented with Matlab.

### 6.1 Data Sets

The data sets we use are sampled from the recommendation log of a commercial personal assistant. When a user uses the personal assistant, various types of cards carrying different information such as news, weather, stock prices are recommended. If the user is interested in a card, she may click the card for more information or view the card for a while. We use such click and view (reading time per pixel is above a chosen threshold) as an indicator of the intent. Different types of cards indicate different types of intent. We pick out eight types of intent that are commonly monitored in most personal assistant applications. The eight types cover the aspects of News, Events, Weather, Places, Finance, Calendar, Traffic, and Sports, respectively. We sampled two data sets for these types of intent between 10 June and 9 July 2015 (the first), and 15 August and 10 September 2015 (the second), which in total contain 20, 807 and 16, 406 anonymous users, respectively. For each type of intent, we also collect the user's contemporaneous context, in particular, the apps used and the venues visited by the user. To protect users' privacy, we use an anonymous identifier for each app and venue, and remove the latitude and longitude of the venue.

### 6.2 Evaluation Criteria

We use the macro and micro F-measures on the predicted intent to evaluate the model performance. Let $\rho$ be the number of testing time steps. We denote by $\boldsymbol{s}^u = (s_1^u, \ldots, s_\rho^u)'$ the true intent of user $u$, where $s_t^u = 1$ means the user has the given intent (i.e., clicks or views the corresponding card) and $s_t^u = 0$ means no such intent at time step $t$. Let $\hat{\boldsymbol{s}}^u = (\hat{s}_1^u, \ldots \hat{s}_\rho^u)'$, $\hat{s}_t^u \in \{0, 1\}$ be the predicted intent. The precision and recall for user $u$ are computed by

$$\text{Prec}^u = \frac{\boldsymbol{s}^{u\prime}\hat{\boldsymbol{s}}^u}{\boldsymbol{1}'\hat{\boldsymbol{s}}^u} \quad \text{and} \quad \text{Rec}^u = \frac{\boldsymbol{s}^{u\prime}\hat{\boldsymbol{s}}^u}{\boldsymbol{1}'\boldsymbol{s}^u},$$

respectively. Let $\overline{\text{Prec}}$ and $\overline{\text{Rec}}$ be the average precision and recall among all users, respectively. The macro F-measure equals

$$\text{Macro F-measure} = 2 \times \frac{\overline{\text{Prec}} \times \overline{\text{Rec}}}{\overline{\text{Prec}} + \overline{\text{Rec}}}.$$

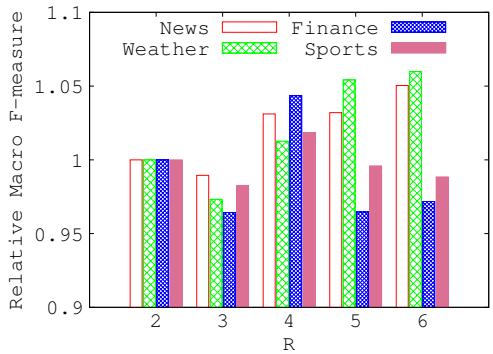



Fig. 9. Relative performance of the collaborative nowcasting model to $R = 2$ when $R$ is varied from 2 to 6 for four selected types of intent.

The precision and recall considering all testing instances are computed by

$$\text{Prec} = \frac{\sum_u \boldsymbol{s}^{u\prime}\hat{\boldsymbol{s}}^u}{\sum_u \boldsymbol{1}'\hat{\boldsymbol{s}}^u} \quad \text{and} \quad \text{Rec} = \frac{\sum_u \boldsymbol{s}^{u\prime}\hat{\boldsymbol{s}}^u}{\sum_u \boldsymbol{1}'\boldsymbol{s}^u},$$

respectively, and the micro F-measure equals

$$\text{Micro F-measure} = 2 \times \frac{\text{Prec} \times \text{Rec}}{\text{Prec} + \text{Rec}}.$$

The macro F-measure reflects the average performance among all users by weighting equally the precision and recall of each user. The micro F-measure evaluates the performance of the model per recommendation instance, which has a bias towards the users who have more intent records.

### 6.3 Methods to Compare

The methods we compare with the collaborative nowcasting model **CNowcast** include

- **BoostedTree**. BoostedTree (Wu et al. 2010) is an ensemble of regression trees (decision trees). It is used in existing contextual ranking models (Shokouhi and Guo 2015) and gives the best performance on the intent monitoring problem among several classic algorithms we have tried including linear regression, SVM, etc.
- **FM**. Factorization machine (FM) (Rendle 2012) is a state-of-the-art method for next-basket recommendations (Rendle et al. 2010), which recommend the items that will be in the user's shopping cart during the next time step. It also effectively performs many other recommendation tasks.
- **NowcastIndi**. This is the nowcasting model (Giannone et al. 2008) introduced in Section 2.2. In this method, the model is applied to the panel of each individual user.
- **CNowcastCP**. In this method, we use the CP tensor decomposition to obtain the collaborative latent factors, which is introduced in Section 3.2.2.

The temporal features are implicitly modeled by the nowcasting related methods. To help the BoostedTree and FM models utilize the temporal features, we also add the time of day and day of week as additional features. We use the first three quarters of the data sets to train the model and the remaining for testing. Unless otherwise specified, we parameterize the collaborative nowcasting model with four factors and two transition noise: $R = 4$, $Q = 2$, and use default parameter values for all other methods.

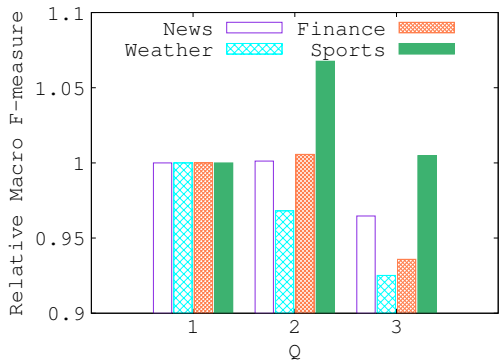



Fig. 10. Relative performance of the collaborative nowcasting model to $Q = 1$ when $Q$ is varied from 1 to 3 for four selected types of intent.

### 6.4 Results of Parameter Tuning and Comparison

*6.4.1 Effect of Parameters R and Q.* **Effect of** $R$. We first study the effect of the number of factors $R$ (i.e., dimension of $\boldsymbol{f}_t$) by varying $R$ from 2 to 6. Fig. 9 shows the relative performance of the collaborative nowcasting model on the first data set for four types of intent: News, Weather, Finance, and Sports. We can see that the performance, measured by the macro F-measure, of the model first decreases and then increases when $R$ varies from 2 to 4. This is because when $R = 2$, the fundamental structure and movement of the context can already be effectively captured (this is consistent with the findings in (Giannone et al. 2005) that many macroeconomic variables can be captured by two factors). When $R$ increases to 3, the increased uncertainty brought by estimating more parameters outruns the marginal benefits from capturing moderately more dynamics of the context. However, this situation is reversed when $R$ increases to 4. When we further increase $R$ to 5 and 6, the performance of the model keeps increasing moderately for News and Weather intent, but decreases for Finance and Sports intent. The reason for the increase is the same as before. The decrease is because 5 or 6 factors make the model overfit for these two types of intent. We will discuss in detail the difference between different types of intent in Section 6.4.3. From the figure we can also see that the performance variance of the proposed model is very small. In most cases, the variance is less than 5%. This indicates that the proposed model is robust to the choice of the number of factors. The relative performance measured by the micro F-measure is similar and hence omitted.

**Effect of** $Q$. Fig. 10 shows the relative macro F-measure of the model when $Q$ is changed from 1 to 3. We can see that when $Q = 2$, the performance of the model slightly increases (except for weather). When $Q$ increases to 3, the performance drops. This indicates that a two dimensional white noise can effectively model the other aspects in the dynamic transition between factors. The relative micro F-measure is similar and thus omitted.

*6.4.2 Comparison across Models.* Tables 2 and 3 respectively present the macro and micro F-measures of each method on the first data set for the eight types of intent when the monitoring granularity is one hour (i.e., $\Delta = 1$ hour). For expositional convenience, we report each method's relative F-measure to the BoostedTree method.

**Cnowcast vs. BoostedTree**. From the two tables we can see that the Cnowcast method consistently outperforms the BoostedTree method, and the performance advantage is up to 28 times. This demonstrates that the proposed model is able to effectively utilize the user's real-time

Table 2. The macro F-measure of each model relative to BoostedTree when $\Delta = 1$ hour

| Model | News | Events | Weather | Places | Finance | Calendar | Traffic | Sports |
|---|---|---|---|---|---|---|---|---|
| BoostedTree | 0.0380 | 0.0165 | 0.0039 | 0.0005 | 0.0014 | 0.0007 | 0.0059 | 0.0038 |
| FM | 0.738 | 0.747 | 0.922 | 1.791 | 2.770 | 5.788 | 0.192 | 0.699 |
| NowcastIndi | 2.586 | 3.720 | 5.806 | 24.26 | 14.28 | 14.61 | 2.387 | 5.181 |
| CNowcastCP | 2.625 | 3.845 | 5.796 | 25.54 | 13.66 | 17.27 | 2.940 | 5.533 |
| CNowcast | **3.024** | **4.410** | **6.479** | **28.23** | **16.50** | **18.13** | **3.068** | **6.426** |

Table 3. The micro F-measure of each model relative to BoostedTree when $\Delta = 1$ hour

| Model | News | Events | Weather | Places | Finance | Calendar | Traffic | Sports |
|---|---|---|---|---|---|---|---|---|
| BoostedTree | 0.0538 | 0.0271 | 0.0084 | 0.0008 | 0.0132 | 0.0018 | 0.0228 | 0.0166 |
| FM | 1.327 | 1.618 | 2.207 | 10.32 | 0.767 | 5.502 | 0.255 | 1.085 |
| NowcastIndi | 1.832 | 2.282 | 2.870 | 20.02 | 1.742 | 7.756 | 0.860 | 1.352 |
| CNowcastCP | 1.994 | 2.437 | 3.014 | 21.85 | 1.731 | 9.251 | 1.098 | 1.540 |
| CNowcast | **2.130** | **2.688** | **3.155** | **23.19** | **1.910** | **9.441** | **1.116** | **1.669** |

context, while the BoostedTree, although providing strong performance in many other problems, fails to capture the structure and dynamics of the context and intent. We can also see that the superiority of Cnowcast over BoostedTree is larger on the macro F-measure than micro F-measure. This indicates that the proposed model is able to monitor the intent of much more users effectively than the BoostedTree method. Therefore, the proposed model is more suitable for real-world applications where there are a large number of users and every user counts.

**Cnowcast vs. FM**. The Cnowcast method also consistently outperforms the FM method, with a performance advantage of up to 16 times (for places and traffic columns in Table 2). This shows that although the FM method provides state-of-the-art performance on the short-term next-basket recommendation problem, it is unable to make effective contemporaneous recommendations in a highly dynamic scenario like the intent monitoring problem. From Table 2, we can see that for many types of intent, FM also has a much lower macro F-measure than the Cnowcast method. This again supports that the proposed method is able to provide effective recommendations for more users and is more appropriate for real-world applications.

**Cnowcast vs. NowcastIndi**. From the two tables, we can see that the collaborative nowcasting model consistently and greatly outperforms the individual nowcasting model, in terms of both macro and micro F-measures. This confirms that by exploiting the panels of all users simultaneously, the proposed model is able to obtain the collaborative latent factors that capture the common characteristics for the intent-related context, and hence utilizes the collaborative capabilities of all users. This also validates that the proposed model can effectively address the data sparsity and personalized nowcast problem encountered by the nowcasting model when it is applied to the intent monitoring problem.

**Cnowcast vs. CnowcastCP**. We can see from Tables 2 and 3 that, across all types of intent, the proposed model significantly outperforms the CnowcastCP model (which appends zero-series to obtain the collaborative latent factors) in terms of both macro and micro F-measures. This validates that by keeping the panels in their original forms, the proposed model avoids the manually-imposing noise, which gives the model a significant advantage in effectively modeling the swiftly changing context and intent.

Table 4. The macro F-measure of each model relative to BoostedTree when $\Delta = 4$ hours

| Model | News | Events | Weather | Places | Finance | Calendar | Traffic | Sports |
|---|---|---|---|---|---|---|---|---|
| BoostedTree | 0.1670 | 0.0783 | 0.0191 | 0.0042 | 0.0187 | 0.0026 | 0.0093 | 0.0166 |
| FM | 0.877 | 1.102 | 1.459 | 3.465 | 1.263 | 9.179 | 1.332 | 1.395 |
| NowcastIndi | 1.746 | 2.643 | 4.403 | 12.70 | 3.788 | 14.92 | 5.800 | 4.221 |
| CNowcastCP | 1.766 | 2.513 | 4.329 | 12.16 | 3.412 | 15.33 | 5.483 | 4.195 |
| CNowcast | **1.963** | **2.950** | **4.904** | **14.13** | **4.680** | **16.95** | **7.377** | **5.264** |

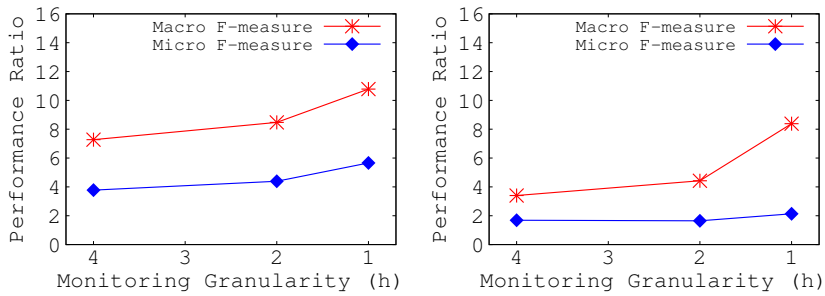
Table 5. The micro F-measure of each model relative to BoostedTree when $\Delta = 4$ hours

| Model | News | Events | Weather | Places | Finance | Calendar | Traffic | Sports |
|---|---|---|---|---|---|---|---|---|
| BoostedTree | 0.2154 | 0.1199 | 0.0405 | 0.0081 | 0.0559 | 0.0072 | 0.0379 | 0.0487 |
| FM | 1.040 | 1.280 | 1.497 | 4.951 | 0.932 | 7.231 | 1.114 | 1.276 |
| NowcastIndi | 1.365 | 1.733 | 2.223 | 8.073 | 1.526 | 8.019 | 1.997 | 1.625 |
| CNowcastCP | 1.422 | 1.686 | 2.301 | 7.893 | 1.427 | 8.447 | 2.048 | 1.636 |
| CNowcast | **1.513** | **1.927** | **2.432** | **9.026** | **1.822** | **8.888** | **2.572** | **2.037** |

*6.4.3 Comparison across Intent Types.* From Tables 2 and 3, we can observe that the performance of different models varies greatly across different types of intent. **i)** For the Places intent, the proposed model outperforms the BoostedTree and FM methods significantly more than other types, in terms of both macro and micro F-measures. This is because a place's type of intent depends on a more complex context than other types. The BoostedTree and FM methods are unable to effectively model the context and the extra complexity makes it more difficult for them to produce effective recommendations. **ii)** From Table 3, we can see that for Finance and Traffic intent, FM performs worse than the BoostedTree method, and for Sports, its performance is very close to that of the BoostedTree. In addition, for these three types of intent, the advantage of the proposed model over the BoostedTree method is also lower than other types (less than two times). These phenomenon are due to that the three types of intent are related to a relatively less complicated and less dynamic context. The modeling of such context can be to some extent narrowed down by the time of day and day of week features (e.g., users often check stock prices during the exchange time on weekdays). Nevertheless, for any type of intent, the related-context consists of much more information than only the time-related features. The best performance of the proposed method demonstrates that it can effectively model the structure of the context and the dynamic correlation between the context and intent, regardless of the intent type and complexity of the context.

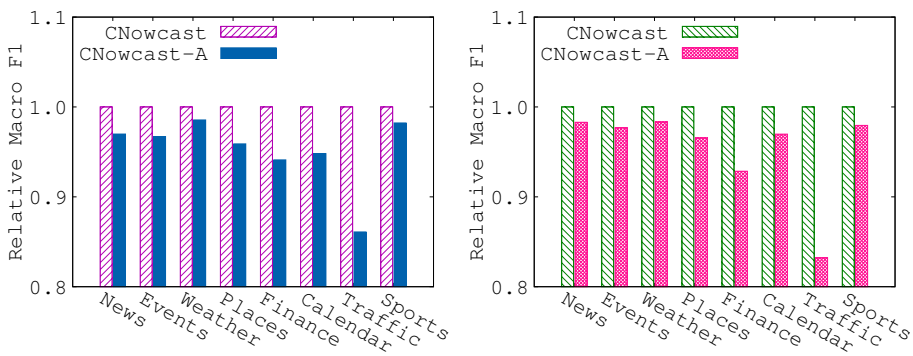
*6.4.4 Comparison across Monitoring Granularity.* Tables 4 and 5 present the macro and micro F-measures of each method on the first data set when the monitoring granularity is four hours, respectively. With the decrease of granularity (from 1 hour to 4 hours), the user's panel becomes less sparse, which gives the BoostedTree, FM and NowcastIndi methods an opportunity to outperform the proposed model if data sparsity is the main impediment. From these tables, we can see that the proposed model still consistently performs the best, and outperforms the other methods significantly. This indicates that the worse performance of the other methods is not mainly due to data sparsity, but because they fail to capture the structure and dynamics of the context and intent.

Fig. 11 presents the average performance ratio of the proposed model (across all types of intent) to the BoostedTree and FM methods on the first data set when the monitoring granularity $\Delta$ is




(a) Ratio to BoostedTree (b) Ratio to FM

Fig. 11. Average performance ratio of the collaborative nowcasting model to BoostedTree and FM across all types of intent when Δ varies from 4 to 1.




(a) Macro F-measure (b) Micro F-measure

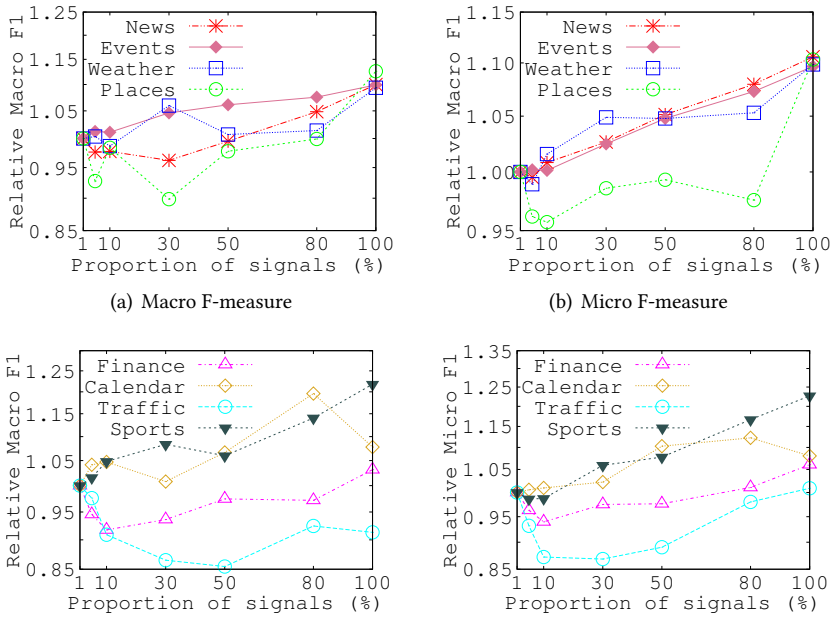
Fig. 12. Effect of correcting collaborative latent factors to personalized latent factors

varied from four hours to one hour, respectively. From the figures we can see that with the increase of the monitoring granularity (i.e., from 4 hours to 1 hour) the advantage of the proposed model over the BoostedTree and FM models also becomes increasingly larger. With the increase of granularity, the intent is closer to the present, i.e, "now". The increasing advantage indicates that the proposed model is particularly suitable for the nowcasting scenario where the user's real-time intent is closely tracked.

From these experiment results, we can see that, under various scenarios and in terms of both macro and micro F-measures, the proposed collaborative nowcasting model consistently performs best and outperforms state-of-the-art methods by a significant margin. The effectiveness and superiority of the proposed collaborative nowcasting model for the intent monitoring problem is thereby empirically confirmed.

### 6.5 Study of the Collaborative Nowcasting Model

*6.5.1 Effect of Personalized Latent Factors.* Next, we study the effect of correcting the collaborative latent factors to personalized latent factors, i.e., the second step of the proposed model. Essentially, without the second step, we cannot predict (and correct) the factors for the next time step. In order to study the effect, we let the PARAFAC2 tensor decomposition in the first step also contain the contextual signals in the testing part (which is in fact impractical in real-world




(a) Macro F-measure (b) Micro F-measure

(c) Macro F-measure (d) Micro F-measure

Fig. 13. Performance of collaborative nowcasting model with real-time data flow

applications due to efficiency issues), and denote this method by **CNowcast-A**. In this way, we can directly use the obtained collaborative latent factors for the regression (i.e., third) step.

Fig. 12(a) and Fig. 12(b) show the relative performance of the two methods CNowcast and CNowcast-A (with $\Delta = 4$) on the first data set measured by macro and micro F-measures, respectively. From the two figures, we can see that without the second step, the performance of CNowcast-A deteriorates significantly in terms of both macro and micro F-measures for all types of intent. The micro F-measure of CNowcast-A for the Traffic intent drops nearly 20% when compared with that of CNowcast. The worse performance of CNowcast-A indicates that it is crucial for the model to blend in sufficient personalization for each individual user. That CNowcast consistently outperforms CNowcast-A by a clear margin demonstrates that the second step of the collaborative nowcasting model, which corrects the collaborative latent factors to personalized latent factors, is an effective way to achieve such personalization.

**The second data set**. To further investigate the effectiveness of the collaborative nowcasting model, we conduct all the above experiments with the second data set. The results are similar and hence omitted. In the sequel, we proceed with the second data set to further conduct the following experiments (with the monitoring granularity being one hour, i.e., $\Delta = 1$).

*6.5.2 Handling the Real-Time Data Flow.* To study the collaborative nowcasting model's capability of processing real-time data flow, we test the model's performance by simulating the continuous arrival of contextual signals. We randomly permute the contextual signals for each user, and let them arrive at the nowcasting component in a streaming fashion. We assign a null value to all the currently unavailable signals.

Fig. 13(a) and Fig. 13(b) present the relative macro and micro F-measures of the collaborative nowcasting model, for four types of intent News, Events, Weather, and Places, when the proportion of available signals is increased from 1% to 100%, respectively. Fig. 13(c) and Fig. 13(d) plot the same

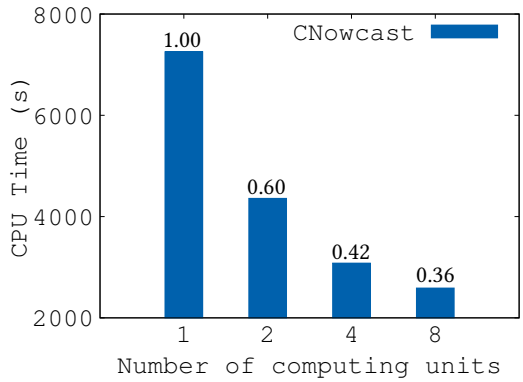



Fig. 14. Performance of collaborative nowcasting model against the number of computing units

performance measures for the other four types of intent Finance, Calendar, Traffic, and Sports. From these figures, we can see a clear increasing pattern when the proportion of available signals is more than 30% for all types of intent. The performance for the Sports intent is increased by more than 20%, in terms of both macro and micro F-measures (comparing when 100% signals are available with only 1% available). This clearly demonstrates that with the continuous arrival of contextual signals, the collaborative nowcasting model has increasing capability to capture users real-time intent.

We can also observe that when there are less than 30%, especially when only 5% or 10%, of contextual signals are available, the performance of the model becomes slightly worse for many types of intent such as News, Weather, Places, and Traffic. This is because the information for intent nowcasting comes from two sources: one is the transition/prediction from previous time steps (history), and the other is the contemporaneous contextual signals (current). When there are only a few signals, these signals moderately corrects the latent factors obtained from system transition, but are not sufficiently informative to accurately capture users' intent. This phenomenon indicates that these two components are equally important, relying on too much on either source will deteriorate the performance. From the subsequent steady increasing patterns, we can see that, for these types of intent, when there are sufficient contextual signals the collaborative nowcasting model can effectively balance these two information sources and provide effective intent monitoring.

Another observation is that with the arrival of only a small portion of signals, the model performance may be significantly improved, such as when the last 20% signals are available the increase of micro F-measure of the Places intent depiced in Fig. 13(b). The improvement is because such type of intent is more contextual, which means they need more contextual signals to be effectively monitored. For example, the Places intent depends on more complex context (and we have already observed this in Section 6.4.3), and hence it can be more effectively monitored when there are a plenty (e.g., more than 80%) of contextual signals.

In general, from the major increasing patterns, these experiments clearly demonstrates that the collaborative nowcasting model can effectively tackle the scenario of real-time data flow for intent monitoring.

*6.5.3 Effect of Intent Monitoring in Parallel.* Next, we empirically investigate the feasibility of deploying the intent nowcasting system in parallel, in particular, we test the ratio of computation speed-up when processing intent nowcasting in multiple computing units. To simulate the infrastructure described in Section 5, we use a main process to act as the server and multiple threads as mobile devices.

Fig. 14 plots the overall CPU time when the number of computing units (i.e., threads) is varied from 1 to 8. The number above each bin represents the relative CPU time when compared with the centralized configuration (i.e., deploying the whole system in one computing unit). From the figure, we can see that with the increase of computing units, the computation time drops rapidly, and the improved efficiency of the system is proportional to the number of computing units. The ratio of speed-up (e.g., 0.60) is larger than the reciprocal (e.g., 0.50) of the corresponding number of computing units is due to three reasons. i) The computation load on each computing unit is not perfectly balanced, which makes the one with the heaviest load halt last. ii) Some components of the computation such as the PARAFAC2 tensor decomposition are not parallelized. iii) There exist various overheads to maintain the distributed processing such as the communication cost (cf. Section 5). Since these are common issues in distributed systems, the significantly improved efficiency demonstrates the feasibility of deploying the intent nowcasting system in a parallel manner.

*6.5.4 On the Consistency of Latent Factors.* In the final set of experiments, we investigate the consistency of the estimated factors. Following existing studies on the consistency of estimated factors (Bai 2003; Doz et al. 2011, 2012; Stock and Watson 2002), we first run a Monte Carlo simulation with small synthetic data. The synthetic data are generated according to the factor model described in Section 3.1, where the measurement noise components are uncorrelated across series and time steps, and the factors follow a first-order linear transition correlation[5]. Specifically, entries of the loading matrix are independently drawn from a standard normal distribution. The factors are generated with the autoregression coefficient being 0.9 and transition noise being white noise. The measurement noise is drawn from a normal distribution with the noise-to-signal ratio uniformly drawn from $[0.1, 0.9]$ (cf. (Doz et al. 2012) and (Doz et al. 2011) for more details). We generate data of different sizes, where the maximum panel size $N$ varies from 10 to 100: $N = 10$, 25, 50, 100, and the number of time steps $T$ varies from 50 to 200: $T = 50$, 100, 200. To simulate different panel sizes for different users, and considering the skewed distribution of such sizes, we generate panel sizes with a power-law like distribution (Clauset et al. 2009) with 80% of panel sizes uniformly drawn from $[2R, 2R + 0.2(N - 2R)]$ and 20% from $[2R + 0.2(N - 2R) + 1, N]$ where $R = 4$.

We evaluate the precision of estimated factors by the following measure

$$\text{Precision of estimate} = \frac{\text{Tr}\left(\boldsymbol{F}'\hat{\boldsymbol{F}}(\hat{\boldsymbol{F}}'\hat{\boldsymbol{F}})^{-1}\hat{\boldsymbol{F}}'\boldsymbol{F}\right)}{\text{Tr}\left(\boldsymbol{F}'\boldsymbol{F}\right)},$$

where $\boldsymbol{F}$ and $\hat{\boldsymbol{F}}$ are the true and estimated factors, respectively. This measure is a trace of the multivariate regression of $\hat{\boldsymbol{F}}$ onto $\boldsymbol{F}$ and represents the correlation between the true and estimated factors (we use this measure is because the factors are identified up to a rotation). A value closer to one indicates a better estimation. To better observe the convergence of the estimated factors, we compare the estimates obtained from the collaborative nowcasting model with those from the individual nowcasting model (Doz et al. 2011) (NowcastIndi) and from principle components (i.e., PCA). We perform 1800 Monte Carlo repetitions and report the average results for $M = 20$ and $M = 50$ users in Table 6.

From the results, we can clearly see that as with the estimators of principle components and NowcastIndi, the precision of estimated factors by CNowcast increases with the panel size $N$ and number of time steps $T$. Since both principle components and NowcastIndi have been proved to be consistent estimators, this result serves as an empirical evidence for that the factors estimated by CNowcast is also asymptotically consistent. We can also see from the result that, the precision

[5]Equivalent to setting $\tau = 0$ and $d = 0$ in (Doz et al. 2012) and (Doz et al. 2011)

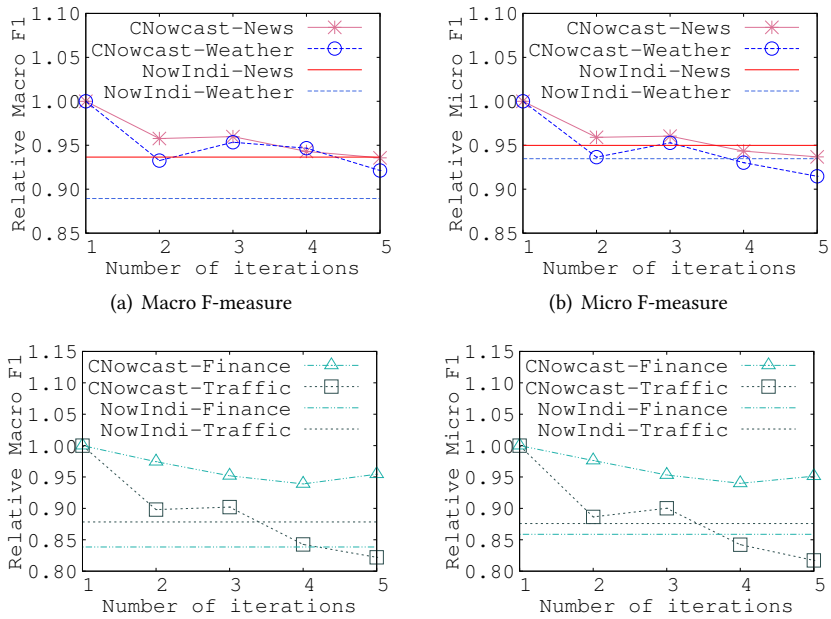
Table 6. Precision of factor estimation

| $T$ | $M = 20$ | | | | $M = 50$ | | | |
|---|---|---|---|---|---|---|---|---|
| | $N = 10$ | $N = 25$ | $N = 50$ | $N = 100$ | $N = 10$ | $N = 25$ | $N = 50$ | $N = 100$ |
| Principle components | | | | | | | | |
| 50 | 0.4203 | 0.4557 | 0.4926 | 0.5367 | 0.4178 | 0.4531 | 0.4908 | 0.5348 |
| 100 | 0.4785 | 0.5282 | 0.5787 | 0.6374 | 0.4764 | 0.5260 | 0.5773 | 0.6359 |
| 200 | 0.5102 | 0.5718 | 0.6329 | 0.7024 | 0.5088 | 0.5707 | 0.6323 | 0.7015 |
| NowcastIndi | | | | | | | | |
| 50 | 0.4488 | 0.4883 | 0.5261 | 0.5687 | 0.4460 | 0.4855 | 0.5242 | 0.5666 |
| 100 | 0.5182 | 0.5754 | 0.6278 | 0.6839 | 0.5159 | 0.5730 | 0.6263 | 0.6824 |
| 200 | 0.5580 | 0.6303 | 0.6938 | 0.7596 | 0.5561 | 0.6288 | 0.6932 | 0.7585 |
| CNowcast | | | | | | | | |
| 50 | 0.5486 | 0.5569 | 0.5641 | 0.5716 | 0.5452 | 0.5480 | 0.5518 | 0.5563 |
| 100 | 0.6062 | 0.6181 | 0.6282 | 0.6422 | 0.5710 | 0.5782 | 0.5865 | 0.5995 |
| 200 | 0.6408 | 0.6560 | 0.6689 | 0.6863 | 0.5881 | 0.5996 | 0.6117 | 0.6310 |

of estimated factors by CNowcast is higher than that of principle components and NowcastIndi in many cases especially when $N < 100$ and $T < 100$ (and that the precision of NowcastIndi is higher than that of principle components is consistent with the findings in (Doz et al. 2011) and (Doz et al. 2012)). This, again, confirms that the proposed model can exploit the collaborative capabilities among users when there are few data. Another interesting fact we can observe from the result is that CNowcast has a slower convergence rate compared with principle components and NowcastIndi especially when the number of users is larger.

Next, we use the real-world data set, iteratively apply the second step of the collaborative nowcasting model (cf. Section 3.5), and use the macro and micro F-measures to check the relationship between the iteratively corrected personalized latent factors (CNowcast-IntentType) and those estimated individually (NowIndi-IntentType).

Fig. 15(a) and Fig. 15(b) depict such relationship for the News and Weather intent, and Fig. 15(c) and Fig. 15(d) for Finance and Traffic. Due to the constraint of computation resources, we only present the results with the number of iterations up to five. Points at one iteration represent the collaborative nowcasting model, and horizontal lines represent the individual model. From Fig. 15(a) and Fig. 15(b), we can see that for the News intent, the performance of iteratively obtained personalized latent factors, in terms of both macro and micro F-measures, is very close to that of the factors estimated individually, especially when the number of iterations is greater than three. For the Weather intent, the micro F-measure of personalized latent factors is also very close to the line representing the performance of the individual model. From Fig. 15(c) and Fig. 15(d), we can see that for the Finance intent, however, the relation is not very clear as the F-measures of the two counterparts are not close to each other. This might be due to the limited number of time steps and panel sizes we use. Overall, from these figures, we can clearly observe a decreasing trend of the F-measures of the personalized latent factors with the increase of iterations. The collaborative latent factors estimated by PARAFAC2 set an initial value for the personalized factors. The decreasing trend indicates that iteratively applying the second step of the collaborative nowcasting model




(a) Macro F-measure (b) Micro F-measure

(c) Macro F-measure (d) Micro F-measure

Fig. 15. Performance relation between iteratively obtained personalized latent factors and individually obtained latent factors

enforces the factors to move toward those estimated individually whose initial values are set by individual principle components. Since the initial factors estimated by principle components are proved to be consistent, this is an empirical evidence that the initial factors estimated by PARAFAC2 are also consistent. By the analysis in Section 3.5, this provides another supporting evidence for that, overall, the factors estimated by the collaborative nowcasting model are consistent.

Finally, we can also observe from the above figures that using one iteration to correct the collaborative latent factors to personalized latent factors is able to blend in sufficient personalization in the collaborative nowcasting model. More iterations will make the model overemphasize on the personalized side and lead to deteriorated performance.

## 7 RELATED WORK

In this section, we review related work on nowcasting models and contextual recommendations in Sections 7.1 and 7.2, respectively.

### 7.1 Nowcasting Models

*7.1.1 Nowcasting in Meteorology.* The term nowcasting is first used in meteorology, which refers to: monitoring the current weather condition and forecasting the weather within the next three (or six) hours [6]. The current weather condition for a certain area can be highly dynamic and may not be directly observable by a limited number of observation stations. The side data that can be used in weather nowcast are radar reflectivity and satellite imagery (Dixon and Wiener 1993; Wilson et al. 1998). With the exponential increase of real-time surface observations, more and more side

[6]http://glossary.ametsoc.org/wiki/Nowcast

data are available for weather nowcast such as the vertical atmospheric conditions provided by commercial aircraft during ascent and descent (Moninger et al. 2010), water vapor distributions provided by ground-based GPS receivers (MacDonald et al. 2002), and large amounts of social media data from Facebook, Twitter etc. (Mass 2012; Mass and Mass 2011). The model used, for instance in thunderstorm nowcasting (Dixon and Wiener 1993), mainly uses a linear regression model with double exponential smoothing to effectively identify and track the storm and other physical atmospheric conditions. The model used in inclement weather nowcasting (Lin et al. 2015) with *tweets* (posts on Twitter) as side data uses the sum aggregate of weather related tweets within a certain spatio-temporal range followed by a linear regression to predict the impact of inclement weather. The variable of interest and side data that these models focus on are of quite different nature than the intent monitoring problem, and hence are inapplicable.

*7.1.2 Nowcasting in Macroeconomic.* Nowcasting is then used in macroeconomics (Giannone et al. 2008) to monitor the contemporaneous value of a variable of interest that is officially published with a significant lag such as the GDP. The side data used in such nowcast are macroeconomic figures that are released much more frequently than the variable of interest, which for instance in GDP nowcast includes: personal consumption, industrial production, surveys, financial variables (e.g., interest rates, stock prices, consumer price index (CPI)), Google Trend data (Scott and Varian 2014) etc. A widely used nowcasting model is proposed by Giannone etc. (Giannone et al. 2008), which is now applied in GDP nowcasting by many agencies (Banbura et al. 2012) including the Federal Reserve Board and European Central Bank.

*7.1.3 Nowcasting in Data Mining.* Recently, nowcasting is studied in data mining to obtain real-time information describing real-world phenomena such as the levels of rainfall, regional influenza-like illness rates (Lampos and Cristianini 2012), or the mood of the nation on some on-going events (Lansdall-Welfare et al. 2012). The side data currently exploited include search engine query log (e.g., Google Trend data) (Duncan and Elkan 2014), posts in social media (Lampos and Cristianini 2012) like Twitter, etc. The model in (Lampos and Cristianini 2012) uses tweets and the sparse learning method Bootstrapped Least Absolute Shrinkage and Selection Operator to select a consistent subset of textual features from the $n$-grams of web encyclopedias, and then regression is applied on the selected features and variable of interest. This model cannot apply to intent monitoring because it cannot address the personalized scenario. A non-trivial task is to first build from a high-quality textual corpus an initial set of good candidate textual features related to the personalized intent.

### 7.2 Contextual Recommendations

*7.2.1 Collaborative Filtering (CF).* CF is a technique widely used in traditional recommendation systems. The essential idea of CF is to make use of the data from other (in particular similar) users or items. Two common CF techniques are matrix factorization (MF) and neighborhood methods (Koren and Bell 2011). In the MF approach, the user-item matrix, containing the ratings of each user to each item, is factorized into the product of two low-rank matrices. In the neighborhood approach, recommendations are based on similar items or users. One problem in CF is that the user's context is not considered, which makes it inapplicable to intent monitoring.

*7.2.2 Time-Aware Recommendations.* By additionally considering the gradual evolving of user preferences and item attributes, there are several time-aware recommendation models. The timeSVD++ model (Koren 2009) augments the MF approach with gradually changing user preferences. The model includes in the MF a time-related preference bias, which is based on the mean date during the period a user rates the items. The dynamic Poisson factorization (Charlin et al.

2015) extends the timeSVD++ model by further allowing for progressively evolving item attributes. The auto-regressive moving average model in (Zhang et al. 2015) applies on the daily time series of token features extracted from product reviews and recommends the items expected to be popular in the future. These approaches cannot apply to intent monitoring because the context they consider is only time, and the gradually evolving preferences or attributes are quite different from the frequently varied intent.

*7.2.3 Context-Aware Recommendations.* Besides time, context-aware recommendation models (Adomavicius and Tuzhilin 2011; Liu et al. 2013) try to incorporate more evidence of a specific situation such as the location, device, purchasing purpose, etc. to model the user preferences on unseen items. Assuming that there are static latent contextual factors that influence the user preferences, these factors can be learned with the probabilistic latent semantic analysis (PLSA) (Hofmann 2003) or hierarchical linear models (HLMs) (Raudenbush and Bryk 2002). The PLSA and HLM models, however, cannot apply to the intent monitoring problem because the contextual factors are required to be static while in our problem the latent factors are highly dynamic and have strong serial and cross-sectional correlation. The model in (Mahmood et al. 2009) considers the dynamic contextual factors over the course of an interaction, e.g., conversation, with the user. However, in the proactive experiences where we monitor the intent, there is no interaction with the user. The multiverse recommendation (Karatzoglou et al. 2010) uses a multidimensional tensor: user-item-context, to model user preferences. This model cannot apply to intent monitoring either because the tensor in our problem is not to be completed, but to be utilized to continuously nowcast the intent at the last time step.

*7.2.4 Proactive Experiences.* The model in (Shokouhi and Guo 2015) addresses the proactive experiences in search engines and personal assistants. Unlike monitoring intent, it uses the reactive search history to re-rank a given list of cards. Therefore, the model cannot apply to intent monitoring. Deep learning and other learning-based methods (Song and Guo 2016; Sun et al. 2016a) are also used to predict users' repeated search patterns in search engines and improve proactive experiences on mobile devices. Although such methods may gain slightly better accuracy by introducing more degrees of freedom (e.g., more parameters), they require much more computation resources, and cannot effectively address the scenario of real-time data flow.

## 8 CONCLUSIONS

Proactively recognizing users' real-time intent has wide applications in proactive information triggering and recommendation tasks such as the newly emerged proactive experiences provided by intelligent personal assistants. Exploiting the real-time flow of contextual signals is an effective way to achieve such intent monitoring. The intent monitoring problem has many new characteristics that traditional recommendation tasks lack, and hence requires the development of new methods to jointly solve these characteristics all together.

We have proposed an innovative collaborative nowcasting model, which effectively resolves the intent monitoring problem by systematically utilizing both the collaborative capabilities among users and the power of nowcasting methods. By summarizing the shared co-movement and temporal structures of all panels with the parsimonious collaborative latent factors, it effectively solves the sparsity and heterogeneity of contextual signals. By obtaining the dynamic sequential correlation among collaborative latent factors and continuously correcting the collaborative latent factors to personalized latent factors with the real-time flow of contextual signals, the proposed model well balances the predictive power of both historical and real-time contextual data, and is able to perform effective intent nowcasting. The collaborative nowcasting model can also be easily

deployed in a distributed fashion, which incurs much less communication cost than the centralized infrastructure and also enables efficient intent nowcasting on mobile devices.

We have evaluated the collaborative nowcasting model in various aspects with real-world recommendation data sets from a commercial personal assistant. The results have demonstrated that the collaborative nowcasting model outperforms various baselines by a significant margin, that the model has effective capabilities of addressing the real-time data flow of contextual signals, and that the model can be easily and largely accelerated by using parallel computing units. We hope that the studied problem and model can draw more attention to new paradigms of recommendations on mobile intelligent devices.

**Future Work**. Since the consistency of estimated latent factors has not been formally and rigorously established, we plan to rigorously prove the consistency of latent factors in the future. Another future work is to develop mechanisms for the proposed model to automatically select more relevant and discard very noisy contextual signals.

## ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their detailed comments that have greatly improved this article.

## REFERENCES

Gediminas Adomavicius and Alexander Tuzhilin. 2011. Context-aware recommender systems. In *Recommender systems handbook*. Springer, 217–253.

Jushan Bai. 2003. Inferential theory for factor models of large dimensions. *Econometrica* 71, 1 (2003), 135–171.

Jushan Bai and Peng Wang. 2016. Econometric Analysis of Large Factor Models. *Annual Review of Economics* 8, 1 (2016), 53–80.

Marta Banbura, Domenico Giannone, Michele Modugno, and Lucrezia Reichlin. 2013. Now-casting and the real-time data flow. *Handbook of Economic Forecasting* (2013).

Marta Banbura, Domenico Giannone, and Lucrezia Reichlin. 2012. Nowcasting. *The Oxford Handbook of Economic Forecasting* (2012).

Rasmus Bro. 1997. PARAFAC. Tutorial and applications. *Chemometrics and intelligent laboratory systems* 38, 2 (1997), 149–171.

Laurent Charlin, Rajesh Ranganath, James McInerney, and David M. Blei. 2015. Dynamic Poisson Factorization. In *RecSys'15*. 155–162.

Peter A. Chew, Brett W. Bader, Tamara G. Kolda, and Ahmed Abdelali. 2007. Cross-language Information Retrieval Using PARAFAC2. In *KDD'07*. 143–152.

Aaron Clauset, Cosma Rohilla Shalizi, and M. E. J. Newman. 2009. Power-Law Distributions in Empirical Data. *SIAM Rev.* 51, 4 (2009), 661–703.

Michael Dixon and Gerry Wiener. 1993. TITAN: Thunderstorm identification, tracking, analysis, and nowcasting-A radar-based methodology. *Journal of Atmospheric and Oceanic Technology* 10, 6 (1993), 785–797.

Catherine Doz, Domenico Giannone, and Lucrezia Reichlin. 2011. A two-step estimator for large approximate dynamic factor models based on Kalman filtering. *Journal of Econometrics* 164, 1 (2011), 188–205.

Catherine Doz, Domenico Giannone, and Lucrezia Reichlin. 2012. A quasi–maximum likelihood approach for large, approximate dynamic factor models. *Review of economics and statistics* 94, 4 (2012), 1014–1024.

Brendan Duncan and Charles Elkan. 2014. Nowcasting with Numerous Candidate Predictors. In *ECML PKDD'14*. 370–385.

Mario Forni, Domenico Giannone, Marco Lippi, and Lucrezia Reichlin. 2009. Opening the black box: Structural factor models with large cross sections. *Econometric Theory* 25, 05 (2009), 1319–1347.

Domenico Giannone, Lucrezia Reichlin, and Luca Sala. 2005. Monetary policy in real time. In *NBER Macroeconomics Annual 2004, Volume 19*. 161–224.

Domenico Giannone, Lucrezia Reichlin, and David Small. 2008. Nowcasting: The real-time informational content of macroeconomic data. *Journal of Monetary Economics* 55, 4 (2008), 665–676.

Richard A Harshman. 1972. PARAFAC2: Mathematical and technical notes. *UCLA Working Papers in Phonetics* 22, 3044 (1972), 122215.

Thomas Hofmann. 2003. Collaborative filtering via gaussian probabilistic latent semantic analysis. In *SIGIR'03*. 259–266.

Dietmar Jannach, Lukas Lerche, and Michael Jugovac. 2015. Adaptation and Evaluation of Recommendations for Short-term Shopping Goals. In *RecSys'15*. 211–218.

Rudolph Emil Kalman. 1960. A new approach to linear filtering and prediction problems. *Journal of basic Engineering* 82, 1 (1960), 35–45.

Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. 2010. Multiverse Recommendation: N-dimensional Tensor Factorization for Context-aware Collaborative Filtering. In *RecSys'10*. 79–86.

Henk AL Kiers, Jos MF Ten Berge, and Rasmus Bro. 1999. PARAFAC2-Part I. A direct fitting algorithm for the PARAFAC2 model. *Journal of Chemometrics* 13, 3-4 (1999), 275–294.

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

Yehuda Koren. 2009. Collaborative Filtering with Temporal Dynamics. In *KDD'09*. 447–456.

Yehuda Koren and Robert Bell. 2011. Advances in collaborative filtering. In *Recommender systems handbook*. 145–186.

Vasileios Lampos and Nello Cristianini. 2012. Nowcasting events from the social web with statistical learning. *TIST* 3, 4 (2012), 72.

Thomas Lansdall-Welfare, Vasileios Lampos, and Nello Cristianini. 2012. Nowcasting the mood of the nation. *Significance* 9, 4 (2012), 26–28.

D.N. Lawley and A.E. Maxwell. 1962. Factor analysis as a statistical method. *Journal of the Royal Statistical Society. Series D (The Statistician)* 12, 3 (1962), 209–229.

Lei Lin, Ming Ni, Qing He, Jing Gao, Adel W Sadek, and Transportation Informatics Tier I Director. 2015. Modeling the Impacts of Inclement Weather on Freeway Traffic Speed: An Exploratory Study Utilizing Social Media Data. In *Transportation Research Board 94th Annual Meeting*.

Qi Liu, Haiping Ma, Enhong Chen, and Hui Xiong. 2013. A survey of context-aware mobile recommendations. *International Journal of Information Technology & Decision Making* 12, 01 (2013), 139–172.

Alexander E MacDonald, Yuanfu Xie, and Randolph H Ware. 2002. Diagnosis of three-dimensional water vapor using a GPS network. *Monthly Weather Review* 130, 2 (2002), 386–397.

Tariq Mahmood, Francesco Ricci, and Adriano Venturini. 2009. Improving recommendation effectiveness: Adapting a dialogue strategy in online travel planning. *Information Technology & Tourism* 11, 4 (2009), 285–302.

Clifford Mass. 2012. Nowcasting: The promise of new technologies of communication, modeling, and observation. *Bulletin of the American Meteorological Society* 93, 6 (2012), 797–809.

Clifford Mass and Clifford F Mass. 2011. Nowcasting: The Next Revolution in Weather Prediction. *Bulletin of the American Meteorological Society* (2011).

William R Moninger, Stanley G Benjamin, Brian D Jamison, Thomas W Schlatter, Tracy Lorraine Smith, and Edward J Szoke. 2010. Evaluation of regional aircraft observations using TAMDAR. *Weather and Forecasting* 25, 2 (2010), 627–645.

Stephen W Raudenbush and Anthony S Bryk. 2002. *Hierarchical linear models: Applications and data analysis methods*. Sage.

Steffen Rendle. 2012. Factorization Machines with libFM. *TIST* 3, 3 (2012), 1–22.

Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing Personalized Markov Chains for Next-basket Recommendation. In *WWW'10*. 811–820.

Steven L Scott and Hal R Varian. 2014. Predicting the present with bayesian structural time series. *International Journal of Mathematical Modelling and Numerical Optimisation* 5, 1-2 (2014), 4–23.

Mary Beth Seasholtz and Bruce Kowalski. 1993. The parsimony principle applied to multivariate calibration. *Analytica Chimica Acta* 277, 2 (1993), 165–177.

Milad Shokouhi and Qi Guo. 2015. From Queries to Cards: Re-ranking Proactive Card Recommendations Based on Reactive Search History. In *SIGIR'15*. 695–704.

Yang Song and Qi Guo. 2016. Query-Less: Predicting Task Repetition for NextGen Proactive Search and Recommendation Engines. In *WWW'16*. 543–553.

James H Stock and Mark W Watson. 2002. Forecasting using principal components from a large number of predictors. *Journal of the American statistical association* 97, 460 (2002), 1167–1179.

John Z Sun, Dhruv Parthasarathy, and Kush R Varshney. 2014. Collaborative kalman filtering for dynamic matrix factorization. *Transactions on Signal Processing* 62, 14 (2014), 3499–3509.

Yu Sun, Nicholas Jing Yuan, Yingzi Wang, Xing Xie, Kieran McDonald, and Rui Zhang. 2016a. Contextual Intent Tracking for Personal Assistants. In *KDD'16*. 273–282.

Yu Sun, Nicholas Jing Yuan, Xing Xie, Kieran McDonald, and Rui Zhang. 2016b. Collaborative Nowcasting for Contextual Recommendation. In *WWW'16*. 1407–1418.

Yingzi Wang, Nicholas Jing Yuan, Yu Sun, Fuzheng Zhang, Xing Xie, Qi Liu, and Enhong Chen. 2016. A Contextual Collaborative Approach for App Usage Forecasting. In *UbiComp'16*. 1247–1258.

Greg Welch and Gary Bishop. 1995. *An Introduction to the Kalman Filter*. Technical Report. Chapel Hill, NC, USA.

James W Wilson, N Andrew Crook, Cynthia K Mueller, Juanzhen Sun, and Michael Dixon. 1998. Nowcasting thunderstorms: A status report. *Bulletin of the American Meteorological Society* 79, 10 (1998), 2079–2099.

Qiang Wu, Christopher JC Burges, Krysta M Svore, and Jianfeng Gao. 2010. Adapting boosting for information retrieval measures. *Information Retrieval* 13, 3 (2010), 254–270.

Yongfeng Zhang, Min Zhang, Yi Zhang, Guokun Lai, Yiqun Liu, Honghui Zhang, and Shaoping Ma. 2015. Daily-Aware Personalized Recommendation Based on Feature-Level Time Series Analysis. In *WWW'15*. 1373–1383.

Hengshu Zhu, Enhong Chen, Hui Xiong, Kuifei Yu, Huanhuan Cao, and Jilei Tian. 2015. Mining mobile user preferences for personalized context-aware recommendation. *TIST* 5, 4 (2015), 58.