Contextual Intent Tracking for Personal Assistants

Yu Sun [†1*], Nicholas Jing Yuan [§2], Yingzi Wang [⋆‡3], Xing Xie [‡4], Kieran McDonald [§5], Rui Zhang [†6]
[†] Department of Computing and Information Systems, University of Melbourne
{[1] sun.y, [6] rui.zhang}@unimelb.edu.au
[‡] Microsoft Research [§] Microsoft Corporation [⋆] University of Science and Technology of China
{[2] nicholas.yuan, [4] xing.xie, [5] kieran.mcdonald}@microsoft.com [3] yingzi@mail.ustc.edu.cn

ABSTRACT

A new paradigm of recommendation is emerging in intelligent personal assistants such as Apple's Siri, Google Now, and Microsoft Cortana, which recommends "the right information at the right time" and proactively helps you "get things done". This type of recommendation requires precisely tracking users' contemporaneous intent, i.e., what type of information (e.g., weather, stock prices) users currently intend to know, and what tasks (e.g., playing music, getting taxis) they intend to do. Users' intent is closely related to context, which includes both external environments such as time and location, and users' internal activities that can be sensed by personal assistants. The relationship between context and intent exhibits complicated co-occurring and sequential correlation, and contextual signals are also heterogeneous and sparse, which makes modeling the context-intent relationship a challenging task. To solve the intent tracking problem, we propose the *Kalman filter regularized PARAFAC2* (KP2) nowcasting model, which compactly represents the structure and co-movement of context and intent. The KP2 model utilizes collaborative capabilities among users, and learns for each user a personalized dynamic system that enables efficient nowcasting of users' intent. Extensive experiments using real-world data sets from a commercial personal assistant show that the KP2 model significantly outperforms various methods, and provides inspiring implications for deploying large-scale proactive recommendation systems in personal assistants.

Keywords

Recommendation; nowcasting; multi-task learning

1. INTRODUCTION

Intelligent personal assistants such as Apple's Siri, Google Now, and Microsoft Cortana are widely deployed on mobile devices (Cortana is also on Windows desktops). They offer *proactive experiences* that aim to recommend "the right information at just the right time" [1], and help you "get things done" [2] even "before you ask" [3]. Figure 1 shows examples of such proactive experiences (left to right) from Microsoft Cortana (on desktops), Apple's Siri, and Google Now, respectively. We can see that personal assistants proactively recommend various types of information such as weather, traffic conditions, stock prices, nearby places, and news. They also suggest apps that users probably need to complete certain tasks such as Uber for getting a taxi and Skype for sending a message. Due to limited display sizes (no matter whether on desktops or mobile devices), personal assistants can only effectively present one or two types of information. When they remind users (e.g., of contacting someone), or ask for confirmation to help users proactively complete certain tasks (e.g., playing music), they also only present one reminder or push one notification at the right time. Therefore, to effectively provide such proactive experiences, we need to closely track users' contemporaneous intent, i.e., what users currently *intend to know*, and what they *intend to do*.

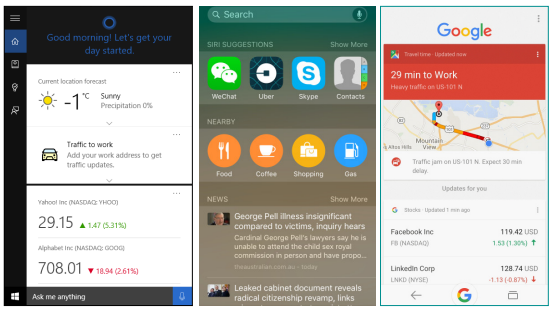


Figure 1: Proactive experiences

Users' contemporaneous intent is closely related to context, which includes not only external environments such as time and location, but users' internal activities that can be effectively sensed by mobile devices such as the apps users use and venues users visit. The relationship between context and intent is complicated, which exhibits complex co-occurring and sequential correlation. For example, users may intend to listen to music while driving or using browsers, or intend to take a taxi when going to or leaving a restaurant. Context itself (e.g., using browsers, driving, and leaving a restaurant) also consists of numerous heterogeneous signals collected from various sources (e.g., the duration of focusing on browsers, distance to home, and time spent at a restaurant), and these signals often contain very sparse records. Therefore, it is challenging to model the context-intent relationship. Moreover, to track users' intent in real-time, we need efficient and robust-to-noise tracking systems, as mobile devices have limited computation capacity and signals

*This work was done while the author was an intern at Microsoft Research.

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 ACM 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.

KDD'16, August 13-17, 2016, San Francisco, CA, USA
© 2016 ACM. ISBN 978-1-4503-4232-2/16/08…$15.00
DOI: http://dx.doi.org/10.1145/2939672.2939676

collected by mobile devices are prone to be noisy.

Traditional recommendation models that recommend music tracks [8], movies [6], etc., cannot apply to intent tracking. These models only address one specific type of intent (e.g., to find movies), and for the specific type of intent, they aim at recommending new items, and focus on the content of recommended items. While for intent tracking, the focus is the recurrence of various types of intent, i.e., type of information (e.g., weather, news) users intend to know and tasks (e.g., playing music) they intend to do. Following the line of traditional recommendation, time-aware recommendation models consider the evolving [15] or seasonal variance [24] of user preferences and item attributes. They cannot apply to intent tracking, either. Apart from the above differences, users' intent often changes swiftly in a very short time, rather than varying on a daily or monthly basis. Models [12, 20] that utilize item similarity or item co-occurring and sequential patterns (e.g., next-basket) also cannot apply because they overlook users' context.

We propose solving the intent tracking problem with a nowcasting approach. Nowcasting approaches [5] were originally developed in meteorology and successfully adopted in macroeconomics. Different from forecasting, nowcasting predicts the contemporaneous value of a variable of interest (intent) with more frequently available signals (context). In particular, we propose the KP2 (**K**alman filter regularized **P**ARAFAC**2**) nowcasting model to solve intent tracking. The KP2 model compactly represents the shared structure and co-movement of intent and contextual signals. It utilizes collaborative capabilities among users, and learns, for each user, a personalized dynamic system that effectively models the sequential correlation among contextual signals and intent. The dynamic system is also robust to noise and enables efficient nowcasting of users' real time intent on mobile devices. The contributions are summarized as follows.

- We identify the intent tracking problem for domains of both information intent (i.e., to know) and task-completion intent (i.e., to do), and model intent tracking as a collaborative nowcasting problem.
- We propose a novel KP2 nowcasting model to effectively solve the intent tracking problem. The model compactly represents the structure and sequential correlation of context and intent. Using collaborative capabilities among users, it innovatively learns for each user a personalized dynamic system that enables efficient prediction of users' contemporaneous intent.
- We conduct extensive experiments with real-world data sets from a commercial personal assistant. The results show that KP2 outperforms various methods by a significant margin, and is able to automatically generate intuitive if-do triggers that support large-scale deployment of effective proactive experiences.

The rest of the paper is organized as follows. Section 2 defines the problem. Section 3 analyzes the context and intent. Section 4 discusses the KP2 model. Section 5 presents experiments. Section 6 reviews related work and Section 7 concludes this paper.

2. PRELIMINARIES

2.1 Problem Definition

We study the problem of closely tracking users' contemporaneous intent [10, 21, 22]. The intent can be any tasks users intend to do or any information users intend to know, e.g., to order food, to book a meeting room, to know the headline news or weather conditions. Intent is closely related to context, which includes not only external physical environments, e.g., location and time, but internal states of users indicated by the activities users have recently engaged in, e.g., apps users recently used. Context is also personalized and frequently changed. Using the discrete time model, we denote by ζ a specific type of intent (e.g., listening to music), $\mathbf{x}_t^u$ the context of user u at time step t, and $F(\zeta, \mathbf{x}_t^u) \in \{0, 1\}$ whether user u has intent ζ at time t given context $\mathbf{x}_t^u$, where $F(\zeta, \mathbf{x}_t^u) = 1$ means positive and 0 means negative. We formally define the intent tracking problem as follows.

Table 1: Example of a panel

Time step	11 a.m.	12 p.m.	1 p.m.	2 p.m.	Now
Chrome	2345	784	0	435	23
Lync	0	1053	0	0	-
Starbucks	0	1251	766	0	0
Fitness First	0	0	0	143	1334
Dist-to-Home	3.45	5.34	10.3	15.7	-
Day-of-Week	5	5	5	5	5
Taxi intent	1	0	0	1	?

DEFINITION 1 (INTENT TRACKING). *Given a set of M users, a starting time t_0, a tracking granularity Δ, a type of intent ζ, and context $\mathbf{x}_t^u$ of user u, the intent tracking problem is to determine the value of $F(\zeta, \mathbf{x}_t^u)$ for every time step t of length Δ starting from t_0.*

2.2 Characteristics of Intent Tracking

The intent tracking problem has several challenging characteristics, which requires special attention and careful model design to address simultaneously.

i) **Real-time tracking**. In the current time step, not all contextual signals (e.g., focus time of various apps) are available simultaneously, which requires handling the continuous arrival of contextual signals.

ii) **Complex context-intent correlation**. The relationship between context and intent exhibits complex co-occurring and sequential correlation, e.g., users may intend to check calendar after arriving at offices.

iii) **Efficient and robust computation**. Personal assistants are widely deployed on mobile devices, which have limited computing capabilities, and hence require light-weight computation to response smoothly. Moreover, the data collected by mobile devices are also prone to be noisy.

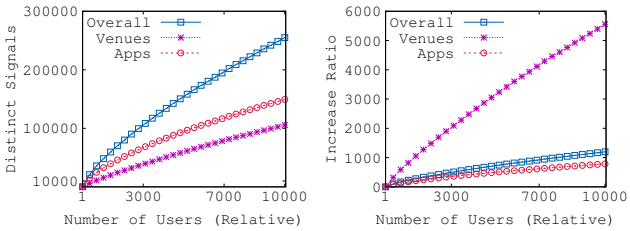
iv) **Personalized service**. Personal assistants are highly personalized and should be a private friend of users'. We need to pay attention to the contextual signals that are specific to each individual user, e.g., the venues only frequently visited by that user.

v) **Data sparsity**. As with traditional recommendation problems, the contextual data for a single user can be quite sparse and hence have limited predicting power.

3. DATA REPRESENTATION AND ANALYSIS

3.1 Context and Intent as Panel Data

Following existing nowcasting work [5, 9], we model each contextual signal as a random process which produces a time series. The contextual signals we consider include: activities that users engage in such as the apps users use



(a) Number of signals (b) Increase rate

Figure 2: Contextual signals v.s. number of users

and venues users visit, real-time spatial measurements w.r.t. users' home and work locations (e.g., distance to home), temporal dimensions w.r.t. calendars (e.g., day of week), etc. Values in the time series of these contextual signals represent quantities relevant to the intent, e.g., the duration that users focus on an app or visit a venue. We let all the contextual signals for a user u form a *panel*, which is denoted by $\mathbf{X}^u$. Table 1 shows an example of a panel, where there are two app signals Chrome and Lync, two venue signals Starbucks and Fitness First, one spatial signal Dist-to-Home, and one temporal signal Day-of-Week, and the tracking intent is the need for a taxi. Note that to effectively provide personalized services, we use for each user the contextual signals specific to herself, which results in panels of various sizes.

3.2 Context Explosion

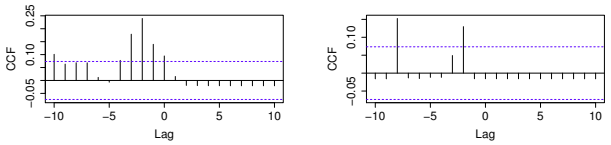
Next, to get insight into the data to assist model design, we analyze the size of contextual signals. In particular, we investigate the number of distinct apps used and venues visited by all users. Figures 2(a) and 2(b) plot the number and increase rate of distinct contextual signals against the number of users, respectively (here we use relative number of users to protect the usage statistics of the commercial personal assistant where we obtain the data). From Figure 2(a), we can see that the number of distinct contextual signals, including apps and venues, increases almost linearly with the increase of users. We can also see from Figure 2(b) that venue signals have the highest increase rate. This demonstrates that on average each user visits (resp. uses) a number of venues (resp. apps) different from others. With the fast growth of users, the size of contextual signals will also grow explosively. For example, in a sampled data set of a commercial personal assistant, if we put the contextual signals of all users together, there will be more than $6,000,000$ distinct venues and nearly $700,000$ distinct apps. Therefore, deploying a uniform contextual dimension is not feasible and will cause significant computation problems.

3.3 Sequential Correlation

Correlation between Context and Intent. Next, we investigate the sequential correlation between intent and contextual signals with the cross-correlation coefficient (CCF). CCF, which is also known as sliding inner-product, measures the similarity of two series at different lags, and is widely used to estimate the degree of sequential correlation. Specifically, the CCF between two series $\mathbf{x} = \mathbf{X}^u_{[i]}$ and $\mathbf{y} = \mathbf{X}^u_{[j]}$ from a panel $\mathbf{X}^u$ (here $\mathbf{X}^u_{[k]}$ denotes the kth row of $\mathbf{X}^u$) at lag τ is computed by

$$\mathcal{R}(\tau) = \frac{\sum_t [(\mathbf{x}(t) - \mu_\mathbf{x})(\mathbf{y}(t-\tau) - \mu_\mathbf{y})]}{\sqrt{\sum_t (\mathbf{x}(t) - \mu_\mathbf{x})^2}\sqrt{\sum_t (\mathbf{y}(t-\tau) - \mu_\mathbf{y})^2}},$$

where $\mathbf{z}(t)$ indicates the tth element in series $\mathbf{z}$ and $\mu_\mathbf{z}$ is the



(a) Visit an educational place v.s. Use document editor apps (b) Browse social media apps v.s. Visit a shopping mall

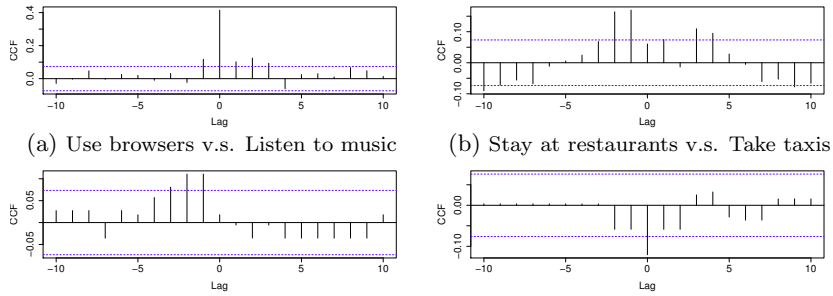
Figure 3: Correlation between contextual signals

mean of $\mathbf{z}$. Figure 4 shows the CCF at various lags between randomly selected contextual signals and intent. Here each lag is one hour. From Figure 4(a) we can see that while using browsers (context), some users often intend to listen to music (intent) because the two series have a significantly larger CCF at lag zero. We can also see from Figure 4(b) that users often intend to take taxis before and after dining at a restaurant (because at lags around zero the CCFs are significant). Similarly, from Figure 4(c) we know that users often intend to send messages after playing video games. We measure the context "arrive at office" by the distance to office, and we can see from Figure 4(d) that at lag zero the CCF between "arrive at office" and "check calendar" is significantly small. This means that users often intend to check their calendar when their distance to office is small, i.e., arriving at office. From these case studies, we can conclude that users' intent has strong sequential correlation with contextual signals. Note that this does not mean intent is *determined* by only a single signal. Intent in general has a complicated correlation with many signals. Different user groups also have very different patterns.

Correlation between Contextual Signals. We next investigate the correlation between contextual signals. Figure 5 shows the covariance (using heatmap) between contextual signals of a randomly selected panel. We can see that some signals exhibit strong covariance because there are many high heat (bright) points besides the main diagonal (which shows the covariance of a signal with itself, and has the highest heat). This indicates that there is redundancy in contextual signals, and hence we can compactly represent the structure and dynamics of a panel with low-dimensional factors. As with context and intent, we also investigate the sequential correlation between contextual signals. Figure 3 shows the CCF at various lags between randomly selected signals. We can see from Figure 3(a) that users often use document editor apps after they visit an educational place such as a university (as the CCFs from lags -4 to 0 are significant). We can also see from Figure 3(b) that users often visit a shopping mall several hours after using social media apps (e.g., Facebook). From these analyses, we can conclude that contextual signals are also closely correlated.

4. THE KP2 MODEL

The fundamental idea of the KP2 model is to represent the structure and dynamics of panels with low-dimensional latent factors, and utilize the evolution of these latent factors for intent tracking. The structure representation is achieved by techniques inspired by PARAFAC2 tensor decomposition, and the dynamics and sequential correlation in latent factors are enforced by the proposed Kalman filter regularization. To the best of our knowledge, this is the first time Kalman filter is used as a regularizer for tensor decomposition. It needs careful model design to seamlessly join the two components and make them collaborate optimally.



(a) Use browsers v.s. Listen to music (b) Stay at restaurants v.s. Take taxis

(c) Play video games v.s. Send messages (d) Distance to office v.s. Check calendar

Figure 4: Sequential correlation between contextual signals and intent

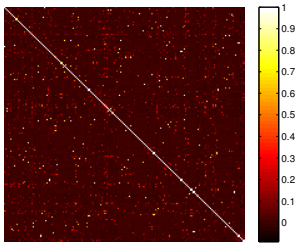


Figure 5: Sample covariance matrix of contextual signals in a randomly selected panel

4.1 Model Formulation

4.1.1 Obtaining Compact Representation

From the above analyses, we have seen that intent and contextual signals exhibit close co-occurring and sequential correlation. To obtain a compact model and hence retain effective prediction power, we assume that the bulk of dynamic interactions among contextual signals and intent can be represented by a few low-dimensional latent factors, i.e.,

$$\mathbf{X}^u \approx \mathbf{\Lambda}^u \mathbf{F}^u \,,$$

where $\mathbf{X}^u \in \mathbb{R}^{N^u \times T}$ is the panel of user u with N^u contextual signals, $\mathbf{F}^u \in \mathbb{R}^{R \times T}$ contains the latent factors with $R < N^u$, and $\mathbf{\Lambda}^u \in \mathbb{R}^{N^u \times R}$ is called the factor *loading* matrix. Since the series in a single panel often contain very sparse records, we propose to utilize the collaborative capabilities among users to address the data sparsity problem. Specifically, we propose to represent the common structure of all users' panels with the same set of latent factors, i.e. (note the removal of superscript u from $\mathbf{F}^u$),

$$\mathbf{X}^u \approx \mathbf{\Lambda}^u \mathbf{F} \quad \text{for all} \;\; u = 1, 2, \ldots, M \,.$$

Such latent factors $\mathbf{F}$ are shared by all users. They effectively summarize the dynamic interactions among all panels, and represent the universal co-movement of various series.

We can obtain such latent factors with the PARAFAC2 tensor decomposition [11, 14] by organizing all panels into a tensor and aligning them by the time dimension. Figure 6 illustrates the formed tensor and PARAFAC2 decomposition. In particular, PARAFAC2 decomposition can be formulated as the following optimization problem

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

where F stands for the Frobenius norm. An advantage of the above formulation is that it effectively handles the context explosion problem (cf. Section 3.2). The reason is that unlike Tucker or CANDECOMP/PARAFAC (CP) tensor decomposition (cf. [14]), which requires the tensor to be of uniform size in every dimension, this formulation allows the contextual dimension to be of various sizes, and hence avoids deploying a uniform contextual dimension for all users. Another advantage is that, without modifying (e.g., truncating) the panels to form a tensor, it well keeps all the personalized signals of each user, which enables us to provide high-quality personalized services.

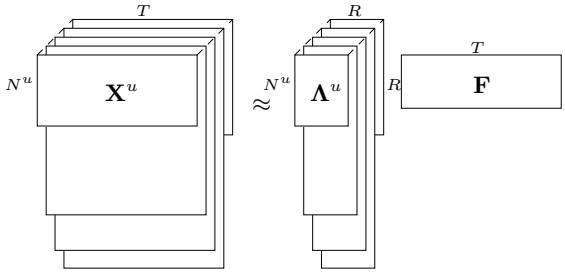


Figure 6: PARAFAC2 decomposition

4.1.2 Regularization with Kalman Filter

Utilizing only the PARAFAC2 decomposition cannot effectively model the sequential correlation within panels. It also cannot support real-time nowcasting because the decomposition is computationally expensive. To enforce sequential correlation, and enable efficient and robust-to-noise intent computation on mobile devices, we propose using Kalman filter to regularize the latent factors, and use the obtained dynamic system for real-time intent nowcasting. Specifically, we consider the linear dynamic system to which Kalman filter applies as follows:

$$\begin{cases} \mathbf{x}_t^u = \mathbf{\Lambda}^u \mathbf{f}_t + \boldsymbol{\xi}_t^u \,, & t = 1, \ldots, T \,, \\ \mathbf{f}_t = \mathbf{A}^u \mathbf{f}_{t-1} + \boldsymbol{\omega}_t^u \,, & t = 2, \ldots, T \,, \end{cases}$$

where $\mathbf{x}_t^u$ (resp. $\mathbf{f}_t$) is the tth column of matrix $\mathbf{X}^u$ (resp. $\mathbf{F}$), $\mathbf{\Lambda}^u \in \mathbb{R}^{N^u \times R}$ is the loading (observation) matrix, $\mathbf{A}^u \in \mathbb{R}^{R \times R}$ is the transition (system) matrix, and $\boldsymbol{\xi}_t^u$ and $\boldsymbol{\omega}_t^u$ are mutually independent Gaussian random variables with known positive definite covariance matrices $\mathbf{\Psi}^u$ and $\mathbf{Q}^u$, respectively. To further simplify the dynamic system, we assume that the covariance matrix $\mathbf{\Psi}^u$ is diagonal.

Let the vectorization of matrix $\mathbf{U} \in \mathbb{R}^{N \times T}$ be

$$\text{vec}(\mathbf{U}) = \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_T \end{bmatrix}, \text{ and let } \text{diag}(\mathbf{V}, T) = \left.\begin{bmatrix} \mathbf{V} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{V} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{V} \end{bmatrix}\right\} T$$

be the matrix with T input matrices V at the main diagonal. Using Kalman filter as a regularizer, we formulate the following optimization problem for intent tracking:

$$\min_{\mathbf{F}, \mathbf{\Lambda}^u, \mathbf{A}^u} \sum_{u=1}^{M} \|\mathbf{X}^u - \mathbf{\Lambda}^u \mathbf{F}\|_F^2 + \frac{\lambda}{2} \Big(\|\mathbf{H}^u \mathbf{f} - \mathbf{x}^u\|_{\mathbf{\Psi}_u^{-1}}^2 + \|\mathbf{G}^u \mathbf{f} - \mathbf{w}^u\|_{\mathbf{Q}_u^{-1}}^2 \Big) \,, \tag{1}$$

where $\mathbf{H}^u = \text{diag}(\mathbf{\Lambda}^u, T)$, $\mathbf{f} = \text{vec}(\mathbf{F})$, $\mathbf{x}^u = \text{vec}(\mathbf{X}^u)$,

$$\mathbf{G}^u = \begin{bmatrix} \mathbf{I} & \mathbf{0} & & \\ -\mathbf{A}^u & \mathbf{I} & \ddots & \\ & \ddots & \ddots & \mathbf{0} \\ & & -\mathbf{A}^u & \mathbf{I} \end{bmatrix}, \quad \mathbf{w}^u = \begin{bmatrix} \mathbf{A^u f}_0 \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix},$$

$\mathbf{\Psi}_u = \text{diag}(\mathbf{\Psi}^u, T)$, $\mathbf{Q}_u = \text{diag}(\mathbf{Q}^u, T)$, $\mathbf{f}_0$ are initial factors (usually $\mathbf{f}_0 = \mathbf{0}$), and the matrix norm of a vector $\|\mathbf{a}\|_{\mathbf{Y}}^2$ denotes $\mathbf{a}'\mathbf{Ya}$. Latent factors obtained from the above optimization problem not only compactly represent the panel structure shared by all users, but model the common temporal dynamics and sequential correlation within all panels.

After obtaining the latent factors $\mathbf{F}$ (cf. Section 4.2 for algorithms that solve the optimization problem), transition matrix $\mathbf{A}^u$, loading matrix $\mathbf{\Lambda}^u$, and noise covariance matrices $\mathbf{\Psi}^u$ and $\mathbf{Q}^u$, we can build up a personalized dynamic system for each user, and hence implement efficient real-time intent nowcasting. Specifically, we first estimate a priori latent factors $\tilde{\mathbf{f}}_t$ for the current time step t with system transition $\mathbf{A}^u\mathbf{f}_{t-1}$. When contextual signals $\mathbf{x}_t^u$ are continuously available within t, we compute a posteriori latent factors $\mathbf{f}_t$ with Kalman filter (details on computing a posteriori latent factors with Kalman filter are presented in Appendix A.1), and then use the a posteriori latent factors in the following regression for intent nowcasting.

4.1.3 Regression for Nowcasting

With the learned latent factors $\mathbf{F}$, we obtain the relationship between intent and latent factors with regression. In particular, assuming that intent and contextual signals are jointly normal (which is common in the real world), we obtain that the likelihood $\Gamma(\digamma(\zeta, \mathbf{x}_t^u) = 1)$ of user u has the intent ζ is a linear function of the latent factors $\mathbf{f}_t$ [5], i.e.,

$$\Gamma(\digamma(\zeta, \mathbf{x}_t^u) = 1) = \alpha^u + \boldsymbol{\beta}^{u\prime}\mathbf{f}_t\,,$$

where α^u and $\boldsymbol{\beta}^u$ are coefficients of the function which can be estimated by regression on historical intent and latent factors. We apply this function to the a posteriori latent factors and obtain the intent likelihood. When the likelihood is above a chosen threshold, we make the prediction that user u has the intent, i.e., $\digamma(\zeta, \mathbf{x}_t^u) = 1$. The threshold we use is the median of fitted likelihood.

4.2 Optimization Algorithm

We propose solving the optimization problem of Eq. 1 by first estimating noise covariance matrices $\mathbf{\Psi}^u$ and $\mathbf{Q}^u$ with principal component analysis, and then estimating parameters $\mathbf{F}$, $\mathbf{A}^u$, and $\mathbf{\Lambda}^u$ with stochastic gradient descent.

4.2.1 Estimating Noise Covariance Matrices

We estimate the observation noise covariance $\mathbf{\Psi}^u$ by residuals of approximating the sample covariance matrix of a panel with its largest R principal components. Specifically, we first standardize signals in a panel so that they have zero sample mean and unitary sample variance, i.e.,

$$x_{it}^u = \frac{1}{\hat{\sigma}_i^u}(\bar{x}_{it}^u - \hat{\mu}_i^u)\,,$$

where $\bar{x}_{it}^u$ means the original data of $\mathbf{X}^u$, $\hat{\mu}_i^u = \frac{1}{T}\sum_{t=1}^T \bar{x}_{it}^u$, and $\hat{\sigma}_i^u = \sqrt{\frac{1}{T}\sum_{t=1}^T (\bar{x}_{it}^u - \hat{\mu}_i^u)^2}$. Let the sample covariance matrix $\mathbf{S}^u$ of panel $\mathbf{X}^u$ be

$$\mathbf{S}^u = \frac{1}{T}\sum_{t=1}^T \mathbf{x}_t^u \mathbf{x}_t^{u\prime}\,.$$

Assuming that noises for different signals are independent, the covariance $\mathbf{\Psi}^u$ is estimated by

$$\mathbf{\Psi}^u \approx \text{Diag}(\mathbf{S}^u - \mathbf{W}^u\mathbf{\Sigma^u}\mathbf{W}^{u\prime})\,,$$

where $\mathbf{\Sigma}^u \in \mathbb{R}^{R\times R}$ is diagonal and contains the largest R eigenvalues of $\mathbf{S}^u$, $\mathbf{W}^u \in \mathbb{R}^{N^u\times R}$ consists of the corresponding eigenvectors subject to $\mathbf{W}^{u\prime}\mathbf{W}^u = \mathbf{I}$, and "Diag$(\cdot)$" means keeping only the elements on main diagonal.

The transition noise covariance $\mathbf{Q}^u$ is estimated in a similar way, except that we need to first obtain an initial estimate of latent factors $\tilde{\mathbf{F}}^u$ and transition matrix $\tilde{\mathbf{A}}^u$. In particular, we obtain approximated latent factors by projecting the signals to the largest R principle components of $\mathbf{S}^u$ (the space spanned by the largest R eigenvectors), i.e.,

$$\tilde{\mathbf{f}}_t^u = \mathbf{W}^{u\prime}\mathbf{x}_t^u\,,$$

and apply vector auto-regression (VAR) on these approximated latent factors to estimate the transition matrix, i.e.,

$$\tilde{\mathbf{A}}^u = \sum_{t=2}^T \tilde{\mathbf{f}}_t^u \tilde{\mathbf{f}}_{t-1}^{u\prime} \left(\sum_{t=2}^T \tilde{\mathbf{f}}_{t-1}^u \tilde{\mathbf{f}}_{t-1}^{u\prime}\right).$$

The covariance matrix $\mathbf{Q}^u$ is then estimated by the residuals of VAR

$$\mathbf{Q}^u = \frac{1}{T-1}\sum_{t=2}^T \tilde{\mathbf{f}}_t^u \tilde{\mathbf{f}}_t^{u\prime} - \tilde{\mathbf{A}}^u \left(\frac{1}{T-1}\sum_{t=2}^T \tilde{\mathbf{f}}_{t-1}^u \tilde{\mathbf{f}}_{t-1}^{u\prime}\right) \tilde{\mathbf{A}}^{u\prime}\,.$$

4.2.2 Gradient Computation

Next, we estimate parameters $\mathbf{F}$, $\mathbf{A}^u$, and $\mathbf{\Lambda}^u$ with stochastic gradient descent (SGD), where a key task is to obtain the gradient. We mainly use the following theorem [18] to obtain gradients w.r.t. $\mathbf{F}$, $\mathbf{\Lambda}^u$, and $\mathbf{A}^u$ in matrix forms.

THEOREM 1. *For a scalar function g of matrix $\mathbf{Y}$, the following identify holds*

$$\mathrm{d}g = \text{Tr}\left[(\frac{\partial g}{\partial \mathbf{Y}})'\mathrm{d}\mathbf{Y}\right].$$

PROOF. By the definition of scalar differential, the left hand side equals

$$\mathrm{d}g = \sum_{ij} \frac{\partial g}{\partial \mathbf{Y}_{ij}}\mathrm{d}\mathbf{Y}_{ij}\,.$$

Applying the trace operator, the right hand side equals

$$\text{Tr}\left[(\frac{\partial g}{\partial \mathbf{Y}})'\mathrm{d}\mathbf{Y}\right] = \sum_{ij}(\frac{\partial g}{\partial \mathbf{Y}})_{ij}(\mathrm{d}\mathbf{Y})_{ij}\,.$$

By the definitions of derivative of scalar functions of matrices and matrix differential, we have

$$(\frac{\partial g}{\partial \mathbf{Y}})_{ij} = \frac{\partial g}{\partial \mathbf{Y}_{ij}} \text{ and } (\mathrm{d}\mathbf{Y})_{ij} = \mathrm{d}\mathbf{Y}_{ij}\,,$$

respectively, which completes the proof. □

Applying the above theorem (cf. Appendix A.2 for more details), we obtain the gradients w.r.t. each parameter as

Table 2: Data sets

Name	Data	Time	Intent
Task-completion intent	App-launch log	11/02/2015-11/30/2015	send messages, listen to music, make reservations, get taxis
Information intent	Proactive-card log	08/15/2015-09/10/2015	news, weather, finance, calendar

follows, here J denotes the loss function of Eq. 1.

$$\frac{\partial J}{\partial \mathbf{A}^u} = \lambda \sum_{t=2}^{T} (\mathbf{Q}^u)^{-1}(\mathbf{A}^u \mathbf{f}_{t-1} - \mathbf{f}_t)\mathbf{f}'_{t-1}\,,$$

$$\frac{\partial J}{\partial \mathbf{\Lambda}^u} = 2\sum_{t=1}^{T} (\mathbf{\Lambda}^u \mathbf{f}_t - \mathbf{x}_t^u)\mathbf{f}'_t + \lambda \sum_{t=1}^{T} (\mathbf{\Psi}^u)^{-1}(\mathbf{\Lambda}^u \mathbf{f}_t - \mathbf{x}_t^u)\mathbf{f}'_t\,,$$

$$\frac{\partial J}{\partial \mathbf{f}_T} = 2\sum_{u=1}^{M} \mathbf{\Lambda}^{u\prime}(\mathbf{\Lambda}^u \mathbf{f}_T - \mathbf{x}_T^u) + \lambda \sum_{u=1}^{M} \mathbf{\Lambda}^{u\prime}(\mathbf{\Psi}^u)^{-1}(\mathbf{\Lambda}^u \mathbf{f}_T - \mathbf{x}_T^u) - \lambda \sum_{u=1}^{M} (\mathbf{Q}^u)^{-1}(\mathbf{A}^u \mathbf{f}_{T-1} - \mathbf{f}_T)\,,$$

$$\frac{\partial J}{\partial \mathbf{f}_t} = 2\sum_{u=1}^{M} \mathbf{\Lambda}^{u\prime}(\mathbf{\Lambda}^u \mathbf{f}_t - \mathbf{x}_t^u) + \lambda \sum_{u=1}^{M} \mathbf{\Lambda}^{u\prime}(\mathbf{\Psi}^u)^{-1}(\mathbf{\Lambda}^u \mathbf{f}_t - \mathbf{x}_t^u) + \lambda \sum_{u=1}^{M} \mathbf{A}^{u\prime}(\mathbf{Q}^u)^{-1}(\mathbf{A}^u \mathbf{f}_t - \mathbf{f}_{t+1}) - \lambda \sum_{u=1}^{M} (\mathbf{Q}^u)^{-1}(\mathbf{A}^u \mathbf{f}_{t-1} - \mathbf{f}_t) \;\text{ for }\; t = 1, \ldots, T-1\,.$$

We then apply SGD to estimate these parameters.

5. EXPERIMENTS

5.1 Set-Up

Data Sets. From a commercial personal assistant, we sample two real-world data sets for the two types of intent: task-completion intent and information intent, respectively. The sampled data sets consist of $8,857$ anonymous users in total. The data set for task-completion intent contains app-launch log recorded by the personal assistant from November 2nd to November 30th 2015. The app launch log effectively indicates users' task-completion intent because when a user asks the personal assistant to do some job, the personal assistant may open relevant apps to complete the task. In the experiments, we randomly select four types of task-completion intent: sending a message (**Message**), listening to music (**Music**), making a reservation (**Reservation**) and getting a taxi (**Taxi**). The data set for information intent contains proactive-card log of the personal assistant from August 15th to September 10th 2015. The log keeps records of the proactively triggered cards (which present various types of information such as events, sports, food and drinks, and stock prices) that are viewed by users for a certain amount of time or clicked for further information. In the experiments, we randomly pick out four types of cards: news (**News**), weather (**Weather**), stock prices (**Finance**), and calendar (**Calendar**). The details of these data sets are summarized in Table 2.

Contextual Signals. We use contextual signals that can be effectively sensed by mobile devices. In particular, we use the used apps, visited venues, distances to home and work places, time of day, and day of week (a panel example containing such signals is presented in Table 1).

Evaluation Metrics. The metrics we use to evaluate the model include the F-measure and Hit-ratio. The F-measure evaluates the precision and recall of model predicted intent, and hit-ratio measures the model's user coverage. In particular, for a given type of intent ζ, let ϕ be the number of time steps used for testing, $\mathbf{s}^u = (s_1^u, \ldots, s_\phi^u)$ with $s_t^u = \digamma(\zeta, \mathbf{x}_t^u)$ for $t = 1, \ldots, \phi$, i.e., $s_t^u = 1$ means user u has intent ζ at time step t and $s_t^u = 0$ means the opposite. Let $\mathbf{s}_p^u = (s_{p\,1}^u, \ldots, s_{p\,\phi}^u)$ be the model's prediction. The precision and recall are computed by

$$\text{Pre} = \frac{\sum_u \mathbf{s}^{u\prime}\mathbf{s}_p^u}{\sum_u \mathbf{1}'\mathbf{s}_p^u} \quad \text{and} \quad \text{Rec} = \frac{\sum_u \mathbf{s}^{u\prime}\mathbf{s}_p^u}{\sum_u \mathbf{1}'\mathbf{s}^u}\,.$$

respectively, and the F-measure equals

$$\text{F-measure} = 2 \times \frac{\text{Pre} \times \text{Rec}}{\text{Pre} + \text{Rec}}\,.$$

The Hit-ratio is computed by

$$\text{Hit-ratio} = \frac{|\{u | \mathbf{s}^{u\prime}\mathbf{s}_p^u > 0\}|}{M}\,,$$

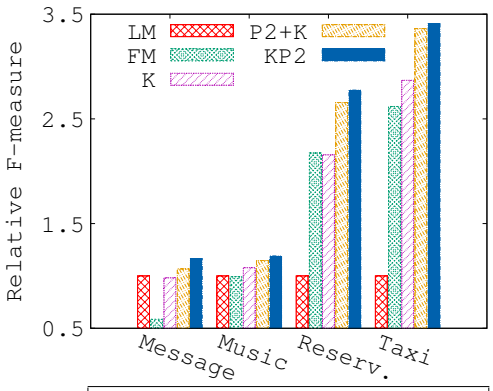
where the numerator is the cardinality of the set of users having at least one accurate intent prediction and the denominator is the number of all users. The hit-ratio measures the coverage of users who will have at least one pleasant experience with the personal assistant, which is an important factor for real-world product.

Compared Methods. The methods we compare with KP2 include the following

- **LambdaMART (LM)**. The LM method is essentially a boosted tree version of LambdaRank. It has strong performance in many real-world applications (e.g., search engines and contextual ranking [21]) and won the 2010 Yahoo! Learning to Rank Challenge.
- **Factorization machine (FM)**. FM is a state-of-the-art method for context-aware and sequential recommendations. It performs effectively in personalized item-tag recommendation and sequential next-basket recommendation [20].
- **Kalman filter (K)**. K is widely used in time series fore/now-casting. It effectively nowcasts the current gross domestic product (GDP) with the real-time data of various macroeconomic variables [5], and the method is applied in many agencies including Federal Reserve Board and European Central Bank.
- **PARAFAC2+Kalman filter (P2+K)**. The P2+K method uses the PARAFAC2 and Kalman filter in a sequential way[1] instead of utilizing the two components in a unified fashion like KP2. P2+K first obtains latent factors with PARAFAC2, and then smooths the transition of latent factors with Kalman filter.

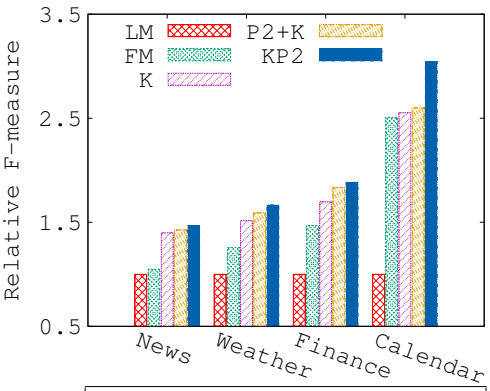
Experiment Setting. For each data set, we use the first three quarters (∼three weeks) to train the model, and

[1]We cannot compare with methods utilizing only P2 because the Kalman filter K is essential for prediction.



F-measure of LM	.2486	.1192	.0094	.0100

(a) Task-completion intent



F-measure of LM	.0912	.0214	.0130	.0070

(b) Information intent

Figure 7: Performance of each method evaluated by F-measure (relative to LM)

the last quarter (∼one week) for testing. We use one hour as the tracking granularity. We use mini-batch SGD with a batch size of 30 and an initial learning rate of 10^{-4} with bold driver adaption. We use default parameters for all compared methods, and set $\lambda = 0.5$ and $R = 2$ for KP2 by default. We will study the effect of parameters in Section 5.3.

5.2 Results

5.2.1 *Comparison across Models on F-measure*

In the first set of experiments, we evaluate the performance of KP2 by comparing it with various methods w.r.t. F-measure. Figures 7(a) and 7(b) show the performance of each method (relative to LM) on the four types of task-completion intent and four types of information intent, respectively (the absolute F-measure of LM is shown at the bottom of corresponding intent). From the two figures, we can see that KP2 significantly outperforms all other methods on all types of intent in terms of F-measure.

KP2 v.s. LM. We can see that KP2 largely outperforms LM on every type of intent. Its performance advantage to LM is up to 3.5 times (on Taxi intent) and is greater than 1.5 times on almost all types of information intent. LM is an assemble of decision trees. Although it has best performance in many ranking problems, the poor performance of LM indicates that it fails at building up the complex relationship between context and intent from raw contextual signals. The strong performance of KP2 indicates that it is an effective method to summarize the structure and dynamics of panels with a few latent factors, as it results in a compact model and retains stronger prediction power.

KP2 v.s. FM. We can also see that KP2 performs significantly better than FM on all types of intent, and its performance advantage to FM is up to two times (on Message intent). FM factorizes the panel, takes into account the sequential correlation, and learns an optimal pair-wise ranking model. We can see from Figures 7(a) and 7(b) that except for Message and Music intent, FM has better performance than LM on all other types of intent, which indicates the importance of modeling sequential correlation. The worse performance of FM than KP2, however, indicates that considering only pair-wise correlation is insufficient for intent tracking, which needs to consider longer sequential correlation in the temporal dimension.

KP2 v.s. K. On every type of intent, KP2 also performs considerably better than the K method. Like KP2, the K method also summarizes panels with a few latent factors and considers sequential correlation in the temporal dimension. The drawback of K is that it fails to leverage collaborative capabilities among users because it can only apply to panels of each individual user. By factorizing panels of all users simultaneously with PARAFAC2, KP2 effectively exploits collaborative power among users, captures common temporal recurring patterns, and proficiently handles the data sparsity problem.

KP2 v.s. P2+K. From Figures 7(a) and 7(b), we can also see that KP2 has much better performance than P2+K on all types of intent. Similar to KP2, P2+K is a compact model, considers the sequential correlation and utilizes collaborative capabilities among users. However, using PARAFAC2 and Kalman filter separately, P2+K is unable to control the relative importance of each component, and fails to make them collaborate optimally. On the contrary, KP2 seamlessly joins the two components and is able to fully control their cooperation.

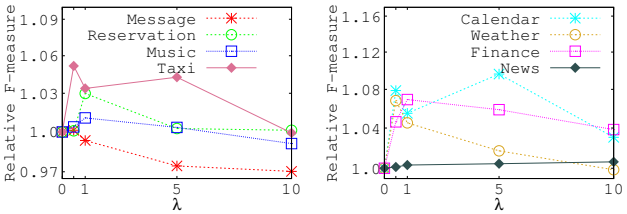
5.2.2 *Comparison across Models on Hit-ratio*

Next, we evaluate KP2 with hit-ratio. The hit-ratio reveals the percentage of users who have (at least one) satisfying proactive experiences with the personal assistant. Such experiences will motivate more interactions and produce a virtuous circle (as more user habits are learned), which is very important for real-world product. Tables 3 and 4 present the hit-ratio of each method on the four types of task-completion intent and four types of information intent, respectively. We can see that KP2 has the highest hit-ratio on all types of intent and significantly outperforms LM and FM. The hit-ratio of KP2 is up to 98.6% (on News intent) and is higher than 50% on almost all types of intent. This indicates that KP2 is able to provide satisfying proactive experiences for a large portion of users and is more suitable than other methods for real-world applications.

We can also see that although the hit-ratios of LM are the lowest among all methods, its hit-ratios on Message, Music, and News intent are higher than other types of intent. This is because these three types of intent correlate with a relatively simpler context, making it relatively less difficult to predict. For example, for the Message intent, some users tend to send messages in regular time windows (e.g., after 9 p.m. or Sunday), which makes the intent closely related to the time of day and day of week. For the Music intent, some users tend to listen to music while driving or using

Table 3: Hit-ratio for task-completion intent

Model	Message	Music	Reservation	Taxi
LM	.6633	.4685	.0104	.0186
FM	.7381	.5925	.1354	.1398
K	.9698	.9331	.4896	.4565
P2+K	.9741	.9547	.4583	.5013
KP2	**.9799**	**.9764**	**.5625**	**.5409**



(a) Task-completion intent (b) Information Intent

Figure 8: Relative performance against λ

browsers, which relates the intent to the spatial distances and a few widely used apps. Other types of intent such as Reservation and Taxi, however, occur in much more diverse context and are related to much more complex contextual signals. The two tables also show that nowcasting related methods including K, P2+K, and KP2 have a much higher hit-ratio than other methods. This demonstrates that nowcasting methods are more suitable for intent tracking.

From the above experiments, we can see that on all types of intent, KP2 has the strongest performance among all methods in terms of both F-measure and hit-ratio metrics. This demonstrates that KP2 is able to track users intent more accurately and has a very large user coverage for satisfying proactive experiences, which makes KP2 a more appropriate method for real-world applications.

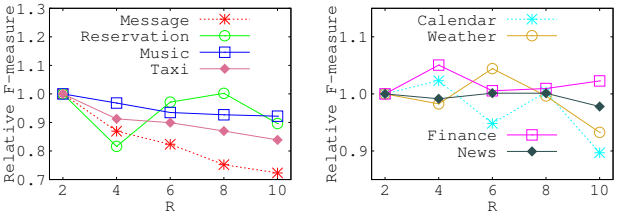
5.3 Effect of Parameters

5.3.1 *Effect of λ*

Next, we study the performance of KP2 under various settings. We first study the effect of λ by changing λ from 0 to 10. Figures 8(a) and 8(b) plot the F-measure of KP2 (relative to $\lambda = 0$) against λ on task-completion intent and information intent, respectively. The performance measured by hit-ratio is consistent with F-measure, and hence is omitted. Parameter λ controls the degree of Kalman filter regularization, which enforces the sequential correlation of latent factors. From the figures, we can see that with the increase of λ, the performance of KP2 first increases and then decreases on all types of intent (except News). When $\lambda = 0$, there is no enforced sequential correlation for the latent factors, which leads to decreased performance. When λ is greater than 5, the overemphasis on sequential correlation interrupts the utilization of collaborative capabilities and extraction of latent factors. This again demonstrates the significance of modeling sequential correlation and the importance of the ability to fully control the cooperation of PARAFAC2 and Kalman filtering. We can also see that for Taxi, Calendar and Finance intent, the effect of λ is more significant (up to 10%). This shows that these types of intent have a stronger sequential correlation with contextual signals (e.g., such intent often occur when users leave or arrive certain venues). The flexibility of KP2 on modeling such correlation enables us to properly and effectively handle the tracking of such intent.

Table 4: Hit-ratio for information intent

Model	Calendar	Weather	Finance	News
LM	.0056	.0457	.0193	.3748
FM	.0970	.1615	.1273	.6070
K	.4100	.6884	.4790	.9641
P2+K	.4127	.7210	.4874	.9853
KP2	**.4183**	**.7357**	**.5462**	**.9857**



(a) Task-completion intent (b) Information Intent

Figure 9: Relative performance against R

5.3.2 *Effect of R*

Next, we study the effect of R by varying R from 2 to 10. Figures 9(a) and 9(b) depict the performance of KP2 with the change of R on task-completion intent and information intent, respectively (relative to $R = 2$). From Figure 9(a), we can see that the performance of KP2 decreases with the increase of R on almost all types of task-completion intent (except Reservation). This is because two-dimensional latent factors are already able to capture the bulk of structure and co-movement of contextual signals related to task-completion intent (which is consistent with the findings in [9]). When R becomes larger, it needs more data to train an effective model because more parameters need to be estimated and the model also tends to overfit to the training data. From Figure 9(b), we can see that the information intent has more complex relationships with R. The performance of KP2 (on average) first increases and then decreases with the increase of R. This indicates that information intent is relevant to a higher dimensional latent space, which has a more complicated structure and the transition of latent factors is more flexible. We can also see that on information intent, the performance variance of KP2 is in most cases within five percent, which means KP2 has the ability to address the different requirements of various information intent.

5.4 If-Do Triggers Generation

Next, we experiment on generating if-do *triggers* from contextual signals. If-do triggers are conditions on which users want certain tasks (resp. information) to be proactively completed (resp. recommended). For instance, in the statement "show me today's headlines every 8:00 a.m. when I am at home", "today's headlines" are the recommended information and "every 8:00 a.m. when I am at home" is the trigger. Automatically generating such triggers (and let users decide which to deploy) will significantly enhance interactions between users and personal assistants. The triggers can also be easily deployed in large-scale applications.

The general methodology of generating triggers is to use decision trees to model the contextual signals in time steps where the intent probability predicted by KP2 is above a certain threshold. The reason we use the predicted probability is because: i) having no record of intent in the log does not imply users do not have such intent, which may

Table 5: If-do triggers for intent

Intent	Triggers
News	Between 6:00 a.m. and 10:00 a.m., Friday, or weekends, distance to office > 10km
Message	Between 5:30 p.m. and 7:30 p.m., weekday, arriving at a food and drink venue
Music	Later than 6:30 p.m., using browsers
Ride	Later than 8:30 p.m., weekday, distance to office > 8km, leaving a supermarket
Reservation	Earlier than 6:30 p.m., Sunday, playing computer games for a long time

be simply because users do not interact with the personal assistant; and ii) the KP2 model effectively generalizes the training instances with common recurring patters and latent characteristics. We follow the paths of decision trees to positive instances to generate the triggers. Table 5 shows one randomly selected trigger for each of the four types of task-completion intent and News intent. We omit certain contextual signals to protect the privacy of anonymous users. We can see that these triggers are quite intuitive for the corresponding intent and can apply to many users with similar habits. This experiment (to some extent) reveals the latent characteristics used by KP2 to predict users intent, and demonstrates the ability of KP2 to automatically generate effective if-do triggers.

6. RELATED WORK

6.1 Context-Aware Recommendation

Traditional recommendation models focus on a specific intent, e.g., to find movies [6], books [19] or music tracks [8], and aim at recommending new items for the intent, which are content-centered. The intent tracking problem, however, focuses on the recurrence of users' intent so as to perform tasks or present information closely related to the intent, which is user-centered. For instance, users may not be interested in watching repeated movies (where content matters), however, they may want to get a taxi every time after shopping (and which taxi does not matter). Following the line of content-centered recommendation, context-aware recommendation [4, 17] further considers users' context, such as time, locations, devices, etc., as users' preferences over new items may be different in different context. Besides the aforementioned difference, the context in context-ware recommendation usually contains only signals about physical environments, and their combinations can be enumerated, e.g., 24 hours $\times$ 7 days $\times$ location types such as home and office [13, 25]. However, in intent tracking, there are numerous contextual signals even for a single user, and the combinations of context cannot be easily enumerated (as illustrated in Table 5). There are also time-aware models that consider the evolving [15] or seasonal variance [24] of user preferences. These methods cannot apply to intent tracking, because instead of evolving on a weekly or a monthly basis, the intent changes swiftly in very short time.

6.2 Time Series Fore/Now-casting

Similar to KP2, there are methods that use correlated time series to predict the target series. The nowcasting model in [5, 9] utilizes monthly macroeconomic indicators such as industrial production and consumer prices to nowcast the contemporaneous GDP, which is only released quarterly. The model obtains a parsimonious approximation of the available information set by a few common factors and applies Kalman filter for prediction, which is the K method used in our experiments and we have seen that it is outperformed by KP2 because it does not consider collaborative capabilities among users. Similarly, the model in [16] uses the number of tweets containing certain automatically selected tokens to nowcast levels of rainfall; the model in [23] conducts forecast on case counts of influenza-like-illness with dynamic Poisson auto-regression using various indicator data sources such as national case counts and Google Flu Trends; and the model in [7] uses Markov process with additional constraints, e.g., network connections, to model co-evolving of multiple time series of signals from a sensor network. One common difference between these methods and KP2 is that they are well designed for one target variable, but not for use cases that provide personalized services for a large number of users, and hence these methods also do not utilize the collaborative capabilities among users.

6.3 Intent Mining and Ranking

Using search engine query log, the method in [10] identifies sequences of search queries with coherent intent, e.g., a sequence of queries on tasks of planning a wedding, interests of a sports team, or habits of regularly reading a news site. Different from tracking the recurrence of intent, this method is used to obtain coherent search queries, which can be further mined to discover new intent and obtain content relevant to users' long-term interests. Therefore, the method cannot apply to intent tracking. The model in [21] addresses proactive experiences in search engines and personal assistants. It uses users' cross platform behavior patterns to re-rank the proactively triggered information cards, and hence cannot apply to intent tracking, either.

7. CONCLUSIONS

It is important and beneficial for personal assistants to offer high-quality proactive experiences. Closely tracking users' intent helps personal assistants achieve this goal by presenting (resp. completing) the right information (resp. tasks) users intend to know (resp. do) at the right time. Users' intent has complicated correlation with various heterogeneous contextual signals. By modeling the contextual signals and intent as panel data, the proposed KP2 nowcasting model compactly represents the structure and comovement of context and intent by a few latent factors. By applying such representation to all users, the KP2 model effectively utilizes the collaborative capabilities among users. It avoids the context explosion problem with a tensor decomposition technique inspired by PARAFAC2. By regularizing the decomposition with Kalman filter, it enforces sequential correlation into the latent factors, and learns for each user a personalized dynamic system, which is robust to noise and can efficiently nowcast users' intent. Extensive experiments using real-world data sets from a commercial personal assistant have shown that the KP2 model outperforms various methods for both task-completion and information intent, and is able to generate if-do triggers that enable large-scale deployment of effective proactive experiences.

8. REFERENCES

[1] http://www.google.com/landing/now/.
[2] http://dev.windows.com/en-us/cortana.

[3] http://www.apple.com/ios/whats-new/.
[4] G. Adomavicius and A. Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, pages 217–253. Springer, 2011.
[5] M. M. Bańbura, D. Giannone, and L. Reichlin. Nowcasting. *The Oxford Handbook of Economic Forecasting*, 2012.
[6] R. M. Bell and Y. Koren. Lessons from the netflix prize challenge. *KDD Explorations*, 9(2):75–79, 2007.
[7] Y. Cai, H. Tong, W. Fan, P. Ji, and Q. He. Facets: Fast comprehensive mining of coevolving high-order time series. In *KDD*, pages 79–88, 2015.
[8] O. Celma. *Music recommendation*. Springer, 2010.
[9] D. Giannone, L. Reichlin, and D. Small. Nowcasting: The real-time informational content of macroeconomic data. *Journal of Monetary Economics*, 55(4):665–676, 2008.
[10] R. Guha, V. Gupta, V. Raghunathan, and R. Srikant. User modeling for a personal assistant. In *WSDM*, pages 275–284, 2015.
[11] R. A. Harshman. Parafac2: Mathematical and technical notes. *UCLA Working Papers in Phonetics*, 22(3044):122215, 1972.
[12] D. Jannach, L. Lerche, and M. Jugovac. Adaptation and evaluation of recommendations for short-term shopping goals. In *RecSys*, pages 211–218, 2015.
[13] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver. Multiverse recommendation: N-dimensional tensor factorization for context-aware collaborative filtering. In *RecSys*, pages 79–86, 2010.
[14] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.
[15] Y. Koren. Collaborative filtering with temporal dynamics. In *KDD*, pages 447–456, 2009.
[16] V. Lampos and N. Cristianini. Nowcasting events from the social web with statistical learning. *TIST*, 3(4):72, 2012.
[17] Q. Liu, H. Ma, E. Chen, and H. Xiong. A survey of context-aware mobile recommendations. *International Journal of Information Technology & Decision Making*, 12(01):139–172, 2013.
[18] J. R. Magnus, H. Neudecker, et al. Matrix differential calculus with applications in statistics and econometrics.
[19] R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 195–204, 2000.
[20] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *WWW*, pages 811–820, 2010.
[21] M. Shokouhi and Q. Guo. From queries to cards: Re-ranking proactive card recommendations based on reactive search history. In *SIGIR*, pages 695–704, 2015.
[22] Y. Sun, N. J. Yuan, X. Xie, K. McDonald, and R. Zhang. Collaborative nowcasting for contextual recommendation. In *WWW*, pages 1407–1418, 2016.
[23] Z. Wang, P. Chakraborty, S. R. Mekaru, J. S. Brownstein, J. Ye, and N. Ramakrishnan. Dynamic poisson autoregression for influenza-like-illness case count prediction. In *KDD*, pages 1285–1294, 2015.
[24] Y. Zhang, M. Zhang, Y. Zhang, G. Lai, Y. Liu, H. Zhang, and S. Ma. Daily-aware personalized recommendation based on feature-level time series analysis. In *WWW*, pages 1373–1383, 2015.
[25] H. Zhu, E. Chen, H. Xiong, K. Yu, H. Cao, and J. Tian. Mining mobile user preferences for personalized context-aware recommendation. *TIST*, 5(4):58, 2015.

APPENDIX

A. COMPUTATION DETAILS

A.1 Correcting Factors with Kalman Filter

Let the a priori and a posteriori latent factors and error covariance matrices at time step t be $\tilde{\mathbf{f}}_t$, $\hat{\mathbf{f}}_t$, $\tilde{\mathbf{P}}_t^u$ and $\hat{\mathbf{P}}_t^u$, respectively. In the time update (prediction) step, the a priori factors for the next time step are computed by

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

and the a priori error covariance by

$$\tilde{\mathbf{P}}_t^u = \mathbf{A}^u \hat{\mathbf{P}}_{t-1}^u \hat{\mathbf{A}}^{u\prime} + \mathbf{Q}^u \,.$$

In the measurement update (correction) step, the Kalman gain $\mathbf{K}_t^u$ equals

$$\mathbf{K}_t^u = \tilde{\mathbf{P}}_t^u \boldsymbol{\Lambda}^{u\prime} (\boldsymbol{\Lambda}^u \tilde{\mathbf{P}}_t^u \boldsymbol{\Lambda}^{u\prime} + \boldsymbol{\Psi}^u)^{-1} \,.$$

With the Kalman gain, the a priori factors are corrected by available contextual signals (missing signals are assigned a very large variance), and the a posteriori factors equals

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

The a posteriori error covariance for next time step equals

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

A.2 Details on Computing Gradients

We obtain the differential of the loss function J as follows

$$\begin{aligned}
\mathrm{d}J =& \mathrm{dTr}\left[\sum_{u=1}^{M}\sum_{t=1}^{T}(\boldsymbol{\Lambda}^u\mathbf{f}_t - \mathbf{x}_t^u)'(\boldsymbol{\Lambda}^u\mathbf{f}_t - \mathbf{x}_t^u)\right.\\
&+\frac{\lambda}{2}\sum_{u=1}^{M}\sum_{t=1}^{T}(\boldsymbol{\Lambda}^u\mathbf{f}_t - \mathbf{x}_t^u)'(\boldsymbol{\Psi}^u)^{-1}(\boldsymbol{\Lambda}^u\mathbf{f}_t - \mathbf{x}_t^u)\\
&\left.+\frac{\lambda}{2}\sum_{u=1}^{M}\sum_{t=1}^{T}(\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t)'(\mathbf{Q}^u)^{-1}(\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t)\right]\\
=&\sum_{u=1}^{M}\sum_{t=1}^{T}\mathrm{Tr}\left[\mathrm{d}((\boldsymbol{\Lambda}^u\mathbf{f}_t - \mathbf{x}_t^u)'(\boldsymbol{\Lambda}^u\mathbf{f}_t - \mathbf{x}_t^u))\right]\\
&+\frac{\lambda}{2}\sum_{u=1}^{M}\sum_{t=1}^{T}\mathrm{Tr}\left[\mathrm{d}((\boldsymbol{\Lambda}^u\mathbf{f}_t - \mathbf{x}_t^u)'(\boldsymbol{\Psi}^u)^{-1}(\boldsymbol{\Lambda}^u\mathbf{f}_t - \mathbf{x}_t^u))\right]\\
&+\frac{\lambda}{2}\sum_{u=1}^{M}\sum_{t=1}^{T}\mathrm{Tr}\left[\mathrm{d}((\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t)'(\mathbf{Q}^u)^{-1}(\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t))\right] .
\end{aligned}$$

To obtain the gradient w.r.t. $\mathbf{A}^u$, we have

$$\begin{aligned}
\mathrm{d}J =&\frac{\lambda}{2}\sum_{t=1}^{T}\mathrm{Tr}\left[\mathrm{d}((\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t)'(\mathbf{Q}^u)^{-1}(\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t))\right]\\
=&\frac{\lambda}{2}\sum_{t=1}^{T}\mathrm{Tr}\left[(\mathrm{d}(\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t))'(\mathbf{Q}^u)^{-1}(\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t)\right]\\
&+\frac{\lambda}{2}\sum_{t=1}^{T}\mathrm{Tr}\left[(\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t)'(\mathbf{Q}^u)^{-1}\mathrm{d}(\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t)\right]\\
=&\lambda\sum_{t=1}^{T}\mathrm{Tr}\left[(\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t)'(\mathbf{Q}^u)^{-1}\mathrm{d}(\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t)\right]\\
=&\lambda\sum_{t=1}^{T}\mathrm{Tr}\left[\mathbf{f}_{t-1}(\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t)'(\mathbf{Q}^u)^{-1}\mathrm{d}\mathbf{A}^u\right] .
\end{aligned}$$

Applying Theorem 1, we obtain

$$\frac{\partial J}{\partial \mathbf{A}^u} = \lambda\sum_{t=2}^{T}(\mathbf{Q}^u)^{-1}(\mathbf{A}^u\mathbf{f}_{t-1} - \mathbf{f}_t)\mathbf{f}_{t-1}' \,.$$

The gradient w.r.t. other variables is computed similarly.