Time-aware Collaborative Poisson Factorization for Service Recommendation

Shuhui Chen, Yushun Fan
Tsinghua National Laboratory for Information Science and Technology
Department of Automation
Tsinghua University
Beijing 100084, China
chensh13@mails.tsinghua.edu.cn
Corresponding: fanyus@tsinghua.edu.cn

Wei Tan
IBM Thomas J. Watson Research Center
Yorktown Heights, NY 10598, USA
wtan@us.ibm.com

Jia Zhang
Carnegie Mellon University
Silicon Valley
jia.zhang@sv.cmu.edu

Bing Bai, Zhenfeng Gao
Tsinghua National Laboratory for Information Science and Technology
Department of Automation
Tsinghua University
Beijing 100084, China
bb13@mails.tsinghua.edu.cn
gzf13@ mails.tsinghua.edu.cn

***Abstract*—With the booming number of web services, it is a challenge for inexperienced developers to select suitable services and make service compositions. Therefore, recommending services based on user queries becomes a necessity. For modeling the queries and services' descriptions, many recent studies are based on LDA (Latent Dirichlet Allocation). However, some previous empirical works indicate that LDA model doesn't gain high accuracy in generating latent presentation which is subject to the restrictive assumption of the Dirichlet-Multinomial distribution. In this paper, we propose a Time-aware Collaborative Poisson Factorization (TCPF) to tackle the problem. TCPF takes Poisson Factorization as the foundation to model mashup queries and service descriptions separately, and incorporate them with the historical usage data together using collective matrix factorization. Experiments on the real-world ProgrammableWeb dataset show that our model outperforms the state-of-the-art methods (e.g., Time-aware collaborative domain regression) by 7.7% in terms of mean average precision, and costs much less time on the sparse, massive and long-tailed data set.**

Keywords- service recommendation; mashup creation; Time-aware; Poisson Factorization

I. INTRODUCTION

With the wide adoption of Service-Oriented Architecture (SOA), reusing web services has become a popular way to develop new software services [1]. Composing services to form mashups helps creating value-added software efficiently and effectively [2]. However, as the number of services published on the web gets larger, it becomes a challenge for inexperienced developers to choose suitable web services to create mashups. As a result, service recommendation becomes a desired method when creating a mashup.

Existing approaches for service recommendation can be divided into two types: (1) Recommending services based on the non-functional features, i.e., QoS [3, 4]. (2) Recommending services based on the functionalities [5, 6, 7, 8]. For many inexperienced developers, a natural way to recommend services is taking a textual query for the desired mashup. For example, a mashup developer's query is described as "*Search local auctions and classified listings and show them on a map view*". Since this mashup inherently requires services with different functions, the developer finally chose services from two functions domains: *Google maps* from mapping domain and *ebay* services from e-commerce domains. It will be of great help to recommend relevant services according to developer query.

Many existing works use LDA (Latent Dirichlet Allocation, [9]) to model both user queries and service descriptions, which are in the form of nature language. However, LDA not good enough to generate accurate latent presentation because of the restrictive assumption of the Dirichlet-Multinomial distribution [10] and put a limitation on the quality of the extracted features from description content and queries. Thus the accuracy of recommendation is restricted to a relative low level. Our previous work TCDR (Time-aware Collaborative Domain Regression) [12] extended the idea of CTR (Collaborative Topic Regression) and used historical usage data to help the feature extraction of content, and gained a performance promotion. In this paper, we go a step further by adopting Poisson Factorization (PF) to model queries and service descriptions.

Poisson Factorization, which gives Gamma-Poisson distribution assumptions to the components of the date set, has been proved to be extremely effective and efficient in modeling document latent vectors in text analysis [13, 14, 15, 16]. We exploit the idea and propose a Time-aware Collaborative Poisson Factorization (TCPF) to make a better service recommendation to queries. Thanks to the properties of Poisson Factorization, the TCPF model can skip the zero observations, so it can achieve a great computation efficiency while promoting the accuracy of recommendation significantly when dealing with the sparse, massive and long-tailed data of the service system.

In addition, similar to [12], information evaporation is also taken into consideration in TCPF. As service system is constantly evolving over time, a usage record of several years ago is of little help in the service selection right now. We take the release time of mashups into the model, and give a larger trust to newer usage records.

The contributions of this paper are as follows:

- By using Poisson Factorization as basic units, we propose the Time-aware Collaborative Poisson Factorization for service recommendation.
- The TCPF model uses multi-factor including text descriptions of both services and mashups and historical usage data. By incorporating them together TCPF can gets more accurate and eventually better recommendation results.

978-1-5090-2675-3/16 $31.00 © 2016 IEEE
DOI 10.1109/ICWS.2016.33

- Comprehensive experiments on the real-world dataset from ProgrammableWeb.com show that our approach gains a 7.7% improvement in terms of recommendation accuracy, compared with the state-of-the-art method, i.e. Time-aware collaborative domain regression [12].

The rest of this paper is organized as follows. Section II formulates the problem we studied and then give a brief introduction to related methods. Section III introduces the proposed model including the generative process and set-method of Auxiliary variables. In section IV, the training and recommendation algorithms are described in detail. Section V reports the experimental results. Section VI discuss the related work and section VII draw a conclusion for the whole paper.

II. BACKGROUND

In this section, we first provide some necessary definitions and formulate the service recommendation for mashup creation problem. Then a brief introduction of two related methods are presented.

A. Problem Formulation

Definition I: Problem Topology. The set $G = (M, \mathbf{MD}, MT, S, \mathbf{SD}, \mathbf{R})$ is used to describe the topology of the problem. Each symbol of the set is defined as follows: $M = \{m_1, m_2, \cdots, m_i\}$ represents the set of mashups. $S = \{s_1, s_2, \cdots, s_j\}$ denotes the set of services. $\mathbf{MD}_m = \{w_{m1}, w_{m2}, \cdots, w_{mn_m}\}$ represents the mashup $m \in M$ described by the set of n_m words. Similarly, $\mathbf{SD}_s = \{w_{s1}, w_{s2}, \cdots, w_{sn_s}\}$ represents the service $s \in S$ described by the set of n_s words. $MT = \{t_1, t_2, \cdots, t_M\}$ records the timestamp of mashup release time and in each time interval has the same amount of mashups. $\mathbf{R} = (r_{ms})_{m=1,s=1}^{M \times S}$ records the historical usage between mashups and services. If mashup m invokes service s then $r_{ms} = 1$, else $r_{ms} = 0$.

Definition II: Service Recommendation for Mashup Creation. For Mashup Creation, a new mashup requirement $l \in \mathbf{QD}$ described as a collection of words $\mathbf{QD}_l = \{w_{l1}, w_{l2}, \cdots w_{ln_l}\}$. The goal is to return a ranked list of services based on G and services with higher probability to be involved will be recommended to the developer.

B. Poisson Factorization

The graphical model of Poisson Factorization (PF) is shown in figure 1. Matrix $\mathbf{Y}$ can be decomposed into the form of $\mathbf{Y} = \Lambda\mathbf{X}$ with Matrix Factorization (MF). The key difference between PF and MF is that PF assumes that $\Lambda, \mathbf{X}$ follow the Gamma distribution and $\mathbf{Y}$ follows Poisson distribution. For example, given the observed rating y_{ui} of user u to item i, the generative process of Poisson matrix factorization unit is formulated as follows:

$$\begin{aligned} \theta_i &\sim Gamma(\lambda_{ia}, \lambda_{ib}) \\ \eta_u &\sim Gamma(\lambda_{ua}, \lambda_{ub}) \\ y_{ui} &\sim Poisson(\eta_u^T \theta_i) \end{aligned} \tag{1}$$

Where $\lambda_{\cdot a}$ is the shape parameter of the Gamma distribution, $\lambda_{\cdot b}$ is the rate parameter of the Gamma distribution and $\eta_u^T \theta_i$ is the shape parameter of Poisson factorization. The goal of Poisson Matrix factorization is to get optimal θ_i and η_u to reconstruct original data [16].

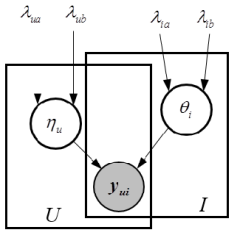

Figure 1. Graphical model of the PF model. PF is a variant of probabilistic matrix factorization. The observed y_{ui} are modeled with a Poisson, parameterized by the inner product of θ_i and η_u

To this point, PF is a modified version of matrix factorization [17]. For the user-item recommendation problem, matrix factorization characterizes both items and users by vectors of factors inferred from item rating patterns. It has been wildly used in many recommender systems, but the biggest shortcoming is that there are unexplained negative values in the results.

On the other hand, all parameters of PF are Non-negative. This feature is similar to the NMF (Non-negative Matrix Factorization) proposed in [18]. NMF addresses the shortcoming of MF and has extremely good performance in clustering. However NMF needs significantly more time to convergence when dealing with large dataset.

From a perspective of generating process, PF is a little bit similar to Latent Dirichlet Allocation (LDA). The core difference between them is the choice of prior distribution assumptions of parameters. And from the point of computation amount, PF has more advantages than LDA. The assumption of gamma distribution has many great properties. If θ_{i_1} and θ_{i_2} are gamma variables with parameters $(\lambda_{i_1 a}, \lambda_b)$ and $(\lambda_{i_2 a}, \lambda_b)$ respectively, then their sum $\theta_{i_1} + \theta_{i_2}$ is a gamma variable with parameters $(\lambda_{i_1 a} + \lambda_{i_2 a}, \lambda_b)$, this will lead to great computation efficiency.

In this paper, we use collective matrix factorization to combine several basic PF units together and solve the mashup-side cold-start service recommendation problem as in [12].

C. Collective Matrix Factorization

Collective matrix factorization (CMF) is proposed for solve the optimal problems about several relational matrices with some shared elements [2]. For example, suppose there are two rational matrices $M1 \sim f(a,b)$ and $M2 \sim g(b,c)$ with the shared element b, collective matrix factorization combines the lose function together to measure error. The modified hybrid objective function is defined as

$$\mathcal{L} = \alpha_1 \mathcal{L}_1(M1; a, b) + \alpha_2 \mathcal{L}_1(M2; b, c) \quad (2)$$

where α_1 and α_2 are relative weights depends on the validation dataset.

There are several basic Poisson matrix factorization units in our model, and we use collective matrix factorization to synthesize them together.

III. Overview of Methodology

The overview of our proposed methodology is illustrated in Fig 2. Aiming to provide the service ranking list for developers, our methodology contains two main components: the generative process and the recommendation process. The generative process takes service textual description as input and output the service comprehensive feature information which consists of service topic offsets, service latent factor, and the time factor. Then the recommendation process integrate the service comprehensive feature information from the generative process and use a Poisson matrix factorization like method to provide service ranking list according to the new mashup requirement of the developer.

The generative process is a complex process and the detail will be explained in below.

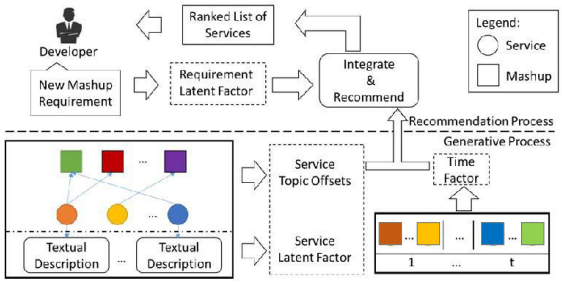

Figure 2. Methodology of TCPF model. TCPF model utilizes service descriptions and the mashup historical usage data separately, then combined with time factor for the more accurate recommendation

A. Generative Process

In the basic PF unit we assume there is a K-dimensional latent factor. Through decomposing observing matrix we get the joint probability distributions between variables belong to row or column of matrix and the latent factor. But in our problem the only information about the new mashup we got is the developer's queries. So we cannot get the mashup-service invoking historical usage because the new mashup does not exist yet, and also cannot directly associate the latent factor with each of them. That is why we called the problem with mashup-side cold-start recommendation [12]. Similar to [12], we use the content of queries to help overcome the mashup-side cold-start problem.

The generative process of our proposed model is illustrated in Fig 3.

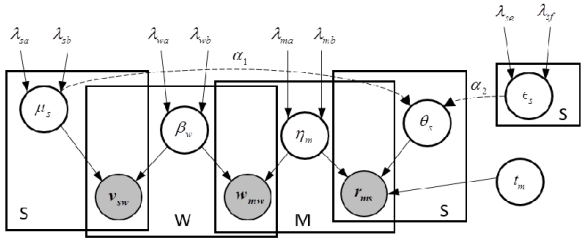

Figure 3. Graphical model of the generative process. This process takes PF as the foundation, and incorporate them with collective matrix factorization.

As the graphical model shown above, the TCPF model first utilizes PF as its basic unit and links them together through the idea from CMF. We collect mashup description, service description and mashup-service usage from the dataset. With the hypothesis of all variables following Gamma distribution, we get the mashup latent factor η and word latent factor β directly. Service latent factor is set with the summation of two parts μ and ϵ, where μ is gained from service description and ϵ is gained from mashup-service usage. θ is the linear combination of μ and ϵ.

The information evaporation theory is also used in recommendation. The core idea is that the recent records worth more than the past ones. Time factor t_m is adopted to scale as in [12].

In general, the generative process of TCPF model is described as follows,

1) For each word w, Draw latent factor $\beta_w \sim Gamma(\lambda_{wa}, \lambda_{wb})$.

2) For each mashup m,

a) Draw latent factor $\eta_m \sim Gamma(\lambda_{ma}, \lambda_{mb})$.

b) For each word w in the introduction content, draw word occurrence count, $v_{mw} \sim Poisson(\eta_m^T \beta_w)$.

3) For each service s,

a) Draw latent factor $\mu_s \sim Gamma(\lambda_{sa}, \lambda_{sb})$.

b) For each word w in the introduction content, draw word occurrence count, $v_{sw} \sim Poisson(\mu_s^T \beta_w)$.

4) For mashup-service pair (m,s),

a) For historical usage information, draw service topic offsets $\epsilon_s \sim Gamma(\lambda_{se}, \lambda_{sf})$.

b) Draw the preference response, $r_{ms} \sim Poisson(\eta_m^T \theta_s)$, where $\theta_s = (\alpha_1 \mu_s + \alpha_2 \epsilon_s)$.

c) The adjusted rating $\tilde{r}_{ms} \sim t_m r_{ms}$ according to information evaporation theory, where the time factor t_m is defined as

$$t_m = \lambda_\eta \frac{e^{-\lambda_t (t_{current} - t_m)}}{\frac{1}{M}\sum_m e^{-\lambda_t (t_{current} - t_m)}} \tag{3}$$

B. Auxiliary variables

To reduce the complexity and apply Variational Bayesian inference, we augment TCPF with auxiliary variables [14]. For each mashup-word pair (m, w), we add K latent variables $w_{mw,k} \sim Poisson\left(\eta_{m,k}^T \beta_{w,k}\right)$, which are integers and satisfy $w_{mw} = \sum_k w_{mw,k}$. This formula-transform process utilizes the probability addible attribute of Poisson distribution. Each variable $w_{mw,k}\ (\forall k \in K)$ can be regarded as the contribution from component k to the observed data w_{mw}. Similar operation is applied for $v_{sw,k}$. A little difference appearing while the formula-transform process to mashup-service pair (m, s) for $\theta_{s,k}$. The parameter $r_{ms,k}$ can break into two parts as

$$r_{ms,k}^a \sim Poisson\left(\alpha_1 \eta_{m,k}^T \mu_{s,k}\right)$$
$$r_{ms,k}^b \sim Poisson\left(\alpha_2 \eta_{m,k}^T \epsilon_{s,k}\right)$$

The summation of them should satisfy the equation $r_{ms} = \sum_k \left(r_{ms,k}^a + r_{ms,k}^b\right)$. Auxiliary variables are necessary for the Variational Bayesian inference. Table I summarize the latent variables, complete conditionals and variational parameters of our model.

TABLE I. TCPF: LATENT VARIABLES, COMPLETE CONDITIONALS AND variational parameters

Latent Variable	Complete Conditional	Variational Parameters
$\beta_{w,k}$	$\lambda_{wa} + \sum_m w_{mw,k} + \sum_s v_{sw,k}$, $\lambda_{wb} + \sum_m \eta_{m,k} + \sum_s \mu_{s,k}$	$\tilde{\beta}^{shp}, \tilde{\beta}^{rte}$
$\eta_{m,k}$	$\lambda_{ma} + \sum_w w_{mw,k} + \sum_s \left(r_{ms,k}^a + r_{ms,k}^b\right)$, $\lambda_{mb} + \sum_w \beta_{w,k} + \sum_s \left(\alpha_1 \mu_{s,k} + \alpha_2 \epsilon_{s,k}\right)$	$\tilde{\eta}^{shp}, \tilde{\eta}^{rte}$
$\mu_{s,k}$	$\lambda_{sa} + \sum_m r_{ms,k}^a + \sum_w v_{sw,k}$, $\lambda_{sb} + \sum_m \eta_{m,k} + \sum_w \beta_{w,k}$	$\tilde{\mu}^{shp}, \tilde{\mu}^{rte}$
$\epsilon_{s,k}$	$\lambda_{se} + \sum_m r_{ms,k}^b$, $\lambda_{sf} + \sum_m \eta_{m,k}$	$\tilde{\epsilon}^{shp}, \tilde{\epsilon}^{rte}$
$w_{mw,k}$	$\log\left(\eta_{m,k}\right) + \log\left(\beta_{w,k}\right)$	ϕ_{mw}
$v_{sw,k}$	$\log\left(\mu_{s,k}\right) + \log\left(\beta_{w,k}\right)$	ψ_{sw}
$r_{ms,k}$	$\begin{cases} \log(\alpha_1) + \log\left(\eta_{m,k}\right) + \log\left(\mu_{s,k}\right), & if \quad k \le K \\ \log(\alpha_2) + \log\left(\eta_{m,k}\right) + \log\left(\epsilon_{s,k}\right), & if \quad K < k \le 2K \end{cases}$	ξ_{ms}

IV. PARAMETER LEARNING AND RECOMMENDING

In this section, we illuminate how to use the methodology described above to carry out the factual model for solving the real-world problem. First how the optimal parameters of the model are obtained is illustrated, and then we explain how to use this well-trained model for a real-world recommendation in details.

A. Parameter optimization approach

The joint probability of generating all visible data is defined to be

$$\begin{aligned} &q\left(\beta, \eta, \nu, \delta, \epsilon, w, v, c, r\right) = \\ &\prod_{w,k} q\left(\beta_{w,k}\right) \prod_{m,k} q\left(\eta_{m,k}\right) \prod_{s,k} q\left(\mu_{s,k}\right) q\left(\epsilon_{s,k}\right) \\ &\prod_{mw,k} q\left(w_{mw,k}\right) \prod_{sw,k} q\left(v_{sw,k}\right) \prod_{ms,k} q\left(r_{ms,k}\right) \end{aligned} \tag{4}$$

As shown in table Ⅰ, the latent variables $\epsilon_{s,k}$, $\mu_{s,k}$, $\eta_{m,k}$, $\beta_{w,k}$ follow Gamma distribution. We denote the shape parameter with the superscript "*shp*" and the rate parameter with the superscript "*rte*". The Variational factors w_{mw}, v_{sw} are multinomials. r_{ms} is also a multinomial but its variational parameter ξ_{ms} is a point in $2K$-simplex.

Our goal is to obtain the posterior distribution of these latent variables. As for many Bayesian models, these exact posterior are computationally intractable. Then the Variational Bayesian inference is used to efficiently approximate these posteriors.

Variational Bayesian inference is an optimization-based strategy for approximating posterior distributions in complex probabilistic models [14, 19]. This algorithm assumes a family of distributions over the target latent variable. The supposed family of distributions indexed by free "Variational" parameters and by adjusting the parameters we find a member of this family closet to the true posterior in Kullback-Liebler (KL) divergence. With this algorithm the inference problem turns to an optimization problem which is easier to accomplish.

And in coordinate ascent we can optimize one variational parameter with the others holding constant. And we iterate this operation until convergence. So the problem remained is to find the complete conditional distribution [14] of each latent variable.

We choose some latent variables to illuminate the iterative algorithm. For Gamma distribution, we take ϵ_s as an example. For each $\epsilon_{s,k}\ (\forall k \in K)$, we extract the relevant terms from (4). Then we get the equation as follow,

$$\begin{aligned} &q\left(\epsilon_{s,k} \mid r, \eta, \lambda_{se}, \lambda_{sf}\right) \\ &\propto \epsilon_{s,k}^{\lambda_{se}-1} e^{-\lambda_{sf}\epsilon_{s,k}} \prod_m \left(\eta_{m,k}\epsilon_{s,k}\right)^{r_{ms,k}} e^{-\eta_{m,k}\epsilon_{s,k}} \end{aligned}$$

$$\begin{aligned} &\propto \epsilon_{s,k}^{\lambda_{se}+\sum_m r_{ms,k}-1} e^{-\left(\lambda_{sf}+\sum_m \eta_{m,k}\right)\epsilon_{s,k}} \\ &= Gamma\left(\lambda_{se}+\sum_m r_{ms,k}, \lambda_{sf}+\sum_m \eta_{m,k}\right) \end{aligned} \tag{5}$$

This gamma distribution can be considered as the distribution with "Variational" parameters. The update for the Variational parameter $\epsilon_{s,k}$ is

$$\left(\tilde{\epsilon}_{s,k}^{shp}, \tilde{\epsilon}_{s,k}^{rte}\right) = \left(\lambda_{se}+\sum_m r_{ms,k}^b, \lambda_{sf}+\sum_m \frac{\tilde{\eta}_{m,k}^{shp}}{\tilde{\eta}_{m,k}^{rte}}\right) \tag{6}$$

where $\tilde{\epsilon}^{shp}$ means the shape of gamma distribution and $\tilde{\epsilon}^{rte}$ means the rate of gamma distribution. $\frac{\tilde{\eta}_{m,k}^{shp}}{\tilde{\eta}_{m,k}^{rte}}$ in the formula is the mean of the gamma variable $\eta_{m,k}$ and can be labeled as $E\left[\eta_{m,k}\right] = \frac{\tilde{\eta}_{m,k}^{shp}}{\tilde{\eta}_{m,k}^{rte}}$.

The gamma distribution has two parameters $\left(\lambda_{se}, \lambda_{sf}\right)$. We can initialize this two parameters separately for higher accuracy. And we also can initialize the parameter

$$\lambda_{sf} = c \cdot \lambda_{se} \tag{7}$$

for convenience as in [15]. The scale c is update as $c^{-1} = \frac{1}{SK}\sum_{s,k} E\left[\epsilon_{s,k}\right]$.

For Poisson distribution, we choose w_{mw} as an example. As the section III -B described, each $w_{mw,k}\left(\forall k \in K\right)$ is a Poisson distribution and $w_{mw} = \sum_k w_{mw,k}$. If we choose the appropriate parameters $\phi_{mw,k}$ and hold the following formula $w_{mw,k} = w_{mw}\phi_{mw,k}$, then w_{mw} also can be considered as a multinomial and $\sum_k \phi_{mw,k} = 1$. We also assume that there is a Poisson distribution p and approach to true distribution with Variational Bayesian inference method. And we get the complete conditional distribution $\phi_{mw,k}$ as below,

$$\begin{aligned} \phi_{mw,k} &= \frac{\eta_{m,k}\beta_{w,k}}{w_{mw}} \\ &\propto \exp\left(\log\left(E_p\left[\eta_{m,k}\beta_{w,k}\right]\right)\right) \\ &= \exp\left(E_p\left[\log\left(\eta_{m,k}\beta_{w,k}\right)\right]\right) \\ &\propto \log\left(\eta_{m,k}\right) + \log\left(\beta_{w,k}\right) \end{aligned} \tag{8}$$

Then the update for the multinomial ϕ_{mw} is

$$\phi_{mw} \propto \exp\left\{\Psi\left(\tilde{\eta}_{m,k}^{shp}\right) - \log\left(\tilde{\eta}_{m,k}^{rte}\right) + \Psi\left(\tilde{\beta}_{w,k}^{shp}\right) - \log\left(\tilde{\beta}_{w,k}^{rte}\right)\right\} \tag{9}$$

where $\Psi(\bullet)$ is the digamma function and $E_p\left(\log\eta_{m,k}\right) = \Psi\left(\tilde{\eta}_{m,k}^{shp}\right) - \log\left(\tilde{\eta}_{m,k}^{rte}\right)$.

Similar to (6) and (8), we integrated list of conditional distributions for all latent variables as Table I. We iterate those process repeatedly until we get the satisfactory results closing enough to the optimal values.

Note that Poisson matrix factorization is non-negative, so we can easy deduce that all $\phi_{mw,k}\left(\forall k \in K\right)$ can set equal to zero if w_{mw} is zero. Similar process is applied to initialize ψ_{sw} and ζ_{ms}. It is very useful for decreasing the complexity of variational parameter space and the computation of parameter learning.

In summary, the parameter learning stage is described as follows.

Algorithm: Parameter learning for TCPF

01. Initialize the shape and rate parameter of each gamma distribution with a small random offset
02. With table I , **repeat**
03. For each service description $v_{sw} > 0$, update ψ_{sw}
04. For each mashup description $w_{mw} > 0$, update ϕ_{mw}
05. For each mashup-service pair $r_{ms} > 0$, update ζ_{ms}
06. Update each gamma distribution parameter
07. **Until** convergence

When adopting in practice, the resulting topics and proportions of LDA can be used to initialize the means of gamma distribution. This method can help to reduce the convergence time of the algorithm.

B. Recommending

After the parameter learning stage we get β_w and $w_{mw} \sim Poisson\left(\eta_m^T\beta_w\right)$ as the basic units of TCPF. When a mashup query is submitted by the developer, the recommendation stage of TCPF is applied. First, we hold the other parameters fixed and update $\eta_{m,k}$ and $\phi_{mw,k}$ with the query document. The iterative formula of $\phi_{mw,k}$ is similar to that in the generative process and the iterative formula of $\eta_{m,k}$ changes as follow because there have no service usage information.

$$\left(\tilde{\eta}_{m,k}^{shp}, \tilde{\eta}_{m,k}^{rte}\right) = \left(\lambda_{ma}+\sum_m w_{mw,k}, \lambda_{mb}+\sum_m \frac{\beta_{w,k}^{shp}}{\beta_{w,k}^{rte}}\right) \tag{10}$$

Then we get the new η_m from the query and compute the expectation of the dot product with η_m and θ_s (learned from TCPF) under the Variational distribution.

$$\hat{r}_{ms} = E\left[\eta_m^T\theta_s\right] \approx E_p\left[\eta_m^T\theta_s\right] \tag{11}$$

The computing processes is similar to the update formular (9). Finally, for the specified mashup requirement m , a ranked list $\hat{r}_s$ which represents the scores of services will be provided for the developer.

V. EXPERIMENTS

In this section, first we introduce the real-world dataset used in the experiments. Next we illuminate the evaluation metrics, adoption baseline methods and the parameter

setting of the proposed model. Then, the recommendation result of the methods is given for comparisons. Last, we summarize some interesting features of TCPF.

A. Dataset Preparation

We crawled the data of services and mashups from September 2005 to February 2015 from PrgrammableWeb.com. The basic properties of the data set summarized as table II.

TABLE II. DATE SET ON PROGRAMMABLEWEB.COM

Number of services	12711
Number of services used in at least one mashup	1120
Number of mashups	6120
Size of vocabulary	21328

B. Evaluation metric

Mean Average Precision at top N (MAP@*N*) is a wildly accepted measure in information retrieval and recommendation system. It is defined as follows,

$$MAP@N = \frac{1}{\left|M_q\right|}\sum_{m\in M_q}\frac{1}{N_q}\sum_{s\in S_m}\left[\frac{n(m,s)}{r(m,s)}\bullet I(m,s)\right] \quad (12)$$

Where $\left|M_q\right|$ denotes the number of mashups queries. $N_q = \min\left(N,\left|s_m\right|\right)$ and s_m represents the set of actually used services in queries $m \in M_q$. For each service $s \in \mathbf{s}$, $n(m,s)$ means the ranking position of s for s both in the S_m and recommendation list while $r(m,s)$ refers to the ranking position of s which only in the recommendation list.

$I(m,s)=1$ when $r(m,s) \le N$ and $I(m,s)=0$ when not.

To avoid the randomness of the test sample, we test several groups of testing data and get a mean value for the last MAP@N results. A higher number of MAP@N means a better accuracy of the recommendation method.

C. Baseline methods

To evaluate the performance of TCPF, we compare it with some state-of-the-art models listed as follows.

1) ***Baseline Method 1:*** *LDA + Matrix Factorization (MF):*

In this method we first use LDA to extract the latent factors of mashup description and the developer's query. Then use MF to factorize the matrix R_{ms} for the optimized latent factors of services. The recommendation rating defined as follows,

$$MF(q,s) = \zeta_q v_s \quad (13)$$

Where ζ_q denotes the topic proportions of query and v_s denotes the topic proportions of service $s \in S$.

2) ***Baseline Method 2:*** *Collaborative Poisson Factorizaiton (CPF)*

CPF is based on our proposed model and sets time factor T_m to the constant 1. It means that CPF ignores the information evaporation.

3) ***Baseline Method 3:*** *Service-description-based Matching (SDM)*

The core idea of SDM is that the query of a new mashup is semantically similar to the description of services. The steps of this algorithm described as below. First, we apply the LDA model for extracting the latent factors of services' description and developer's query respectively. Then for $s \in S$ we calculate the cosine similarities $sim(q,s)$ between them as the result for recommendation rating.

$$SDM(q,s) = sim(q,s) \quad (14)$$

4) ***Baseline Method 4:*** *Mashup-description-based Collaborative Filtering (MDCF)*

MDCF is based on the idea of traditional neighborhood-based collaborative filtering, with assuming that similar mashups have a higher possibility to use similar service. MDCF uses LDA model to calculate the topic proportions of each mashup description $m \in M$ and the developer's query q respectively. Then calculates the cosine similarity $sim(q,m)$ with time factor t_m applied. The equation defined as bellows,

$$MDCF(q,s) = \frac{\sum_{m\in U(N,q)}\left[sim(q,m)t_m\right]}{\sum_{m\in U(N,q)} sim(q,m)} \quad (15)$$

where $U(N,q)$ denotes the Top N similar mashups with the developer's query.

5) ***Baseline Method 5:*** *Time-aware Collaborative Domain Regression (TCDR)*

TCDR is similar to our proposed model except that the PF model is replaced by LDA model. The detailed process is described in [12]. By applying LDA model TCDR learns the topic proportions of services $v_s\left(s \in S\right)$ and the topic proportions of developer's query ζ_q. Then the recommendation rating defined as follows,

$$TCDR(q,s) = \zeta_q v_s \quad (16)$$

D. Experimental Results

1) *Experiment settings*

We segment the data with thirty mashups as a group in chronological order instead of shredding the dataset monthly to avoid the tremendous differences of the amount within different months. And when testing the model, services which have never been used are removed. By this way we get 204 groups. We select the last 30 groups and treat them as the test data while its previous data serves as the training data. And then we run the algorithms for the value of MAP@*N*. We repeat this process for all of the last 30 groups and get a mean value as the last result. Actually these 30 groups contain the mashups from May 2012 to February 2015. This means TCDF has been tested adequately for the test date has covered a wide range of time span.

The parameters are set as follows. For all method we set topic number $K = 40$. For TCPF and CPF, we set

$\alpha = [1 \quad 1]$. The shape parameter λ_{wa} is set to 0.07 and other shape parameters are set to 0.03. The rate parameters are set following the (7) for convenience. For LDA model we set the hyper-parameters $\alpha = 1.25$ and $\beta = 0.01$. For MDCF, N is set to 250. t_m in TCPF and TCDR, λ_η is set to 1 and λ_t is 0.08.

2) *Performance comparison*

The comparison results are shown in figure 4. From the result of SDM, we can get a conclusion that it is unreliable to judge whether a service should be invoked by a mashup only based on semantic similarity. MDCF is a neighborhood-based method essentially. For taking the historical information into consideration, MDCF get a small performance promotion. MF is a better method than MDCF, for this model captures the signals encompassed in the mashup historical usage. TCDR continue improves the performance by taking information evaporation into consideration. CPF adopts the prior hypothesis of the data and get a higher performance. At last TCPF synthesizes the merits of TCDR and CPF, promotes the accuracy of recommendation significantly, and outperforms all the baseline methods.

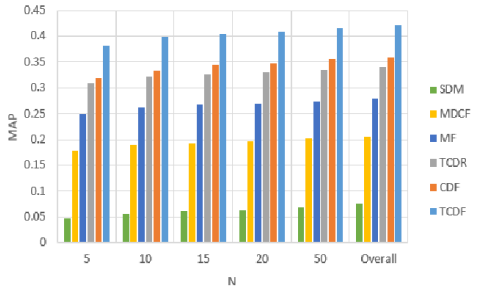

Figure 4. MAP@N of the different methods. This figure shows the TCPF model outperforms all the baseline methods in accuracy.

The detailed performance is summarized in Table III. For our proposed model TCPF, while comparing with the CPF, it can be inferred that taking information evaporation into consideration brings a promotion of about 6%. And compare with TCDR, which has similar process but based on LDA, TCPF promotes the performance of about 7.7%.

TABLE III. PERFORMANCE COMPARISON OF DIFFERENT ALGORITHMS ON MAP@20 AND OVERALL MAP

Recommendation Algorithm	MAP@20	Overall MAP
SDM	6.25%	7.53%
MDCF	19.34%	20.58%
MF	26.9%	27.85%
TCDR	32.71%	34.06%
CDF	34.37%	35.96%
TCDF	**40.48%**	**42.05%**

3) *Time Complexity Analysis of TCPF*

TCPF has some wonderful features inherited from PF model. The most outstanding feature is this method has a very low computational cost. As described in the Algorithm, the computational cost of the model mainly depends on the number of non-zero elements in matrices such as mashup descriptions matrix, service descriptions matrix and mashup-service historical usage matrix. TCPF makes calculation only when the elements of these matrices are not zero during iteration. Because of this character, TCPF is specifically more suitable for the massive, sparse and long-tail data comparing with other model. Coincidentally, the data we crawled from ProgammableWeb just conforms to this type. And, this character is conformed to the vast majority of the situation when users call for services. We analysis all observed matrices after data cleaning and get the results as table IV.

TABLE IV. EXPERIMENT DATA ANALYSIS

Statistic Info.	Value	Sparsity
Size of vocabulary	21328	
Average # words in one service description	38.59	99.82%
Average # words in one mashup description	14.75	99.93%
Number of services for computation	1120	
Average # services in one mashup description	1.95	99.83%

We choose r_{ms} to illustrate the advantage of this character. The dimension of $[r_{ms}]$ is 5709×1120. Comparing with other methods which need to store and calculate all the elements (zero ones and non-zero ones), TCPF only consider the non-negative elements. So TCPF only needs to store and calculate the sparse matrix which the number of elements is about 5709×1.95. Same conclusion can be drawn for w_{mw}, v_{sw} and c_{sw}. To quantifiably verify the efficiency of TCPF, we compare the learning time cost of TCPF and TCDR. Both of these two algorithm sets 50 iterations to get the training model. The results is provided in Table V. As we can see, TCPF costs much less time than TCDR, which confirms to the analysis above.

TABLE V. THE COMPARISON OF TRAINING TIME

Model	Time cost of one iteration	Total time of model training
TCPF	1.4s	70s
TCDR	27.3s	1365s

VI. RELATED WORK

A. . Service Recommendation

As described in [1], existing service recommendation approaches mainly based on their functionality [5-8], non-functional features [3 13], and related with social networks [20, 21]. The functional approaches emphasize the services functionalities during the recommendation, while the non-functional approaches pay more attention on the Quality of the Service (QoS) and the social networks based approaches consider the recommendation from the relationships of the users.

With respect to the functional approaches, the top priority is recommending the service with most suitable functions to users rather than thinking about index like QoS or Ease of Use. Under this condition some functional approaches use content matching like key-words search [6, 7]. Other approaches use semantic-based search [5, 8] to

increase the accuracy of the recommendation. With the development of machine learning, researcher find more approaches for Web recommendation based on probabilistic models (such as Latent Dirichlet Allocation (LDA) [11, 12] or others [22]).

In this paper, we import TCPF model based on PF for recommendation. To the best of our knowledge, we are the first to introduce PF model for service recommendation.

B. Poisson Factorization

Poisson matrix Factorization (PF) was proposed in [10] as a GAP model. [14] changed the computational method and use it for documents recommendation. Because this model combine good scalability with predictive accuracy, PF became very popular in recent years and widespread in multiple areas. [15] use PF for automatic music tagging. A promising tendency in the development of this model is using it as a basic unit to solve more difficult problems. The most recent paper is [16], which developed CBPF method based on PF for cold-start local event recommendation. All the results of these papers indicate that it is a superior model especially for the content-based data sets.

In this paper we also treat PF as foundation and solve the mashup-side cold-start recommendation problem.

VII. CONCLUSION

As the service system is constantly developing with the rapid increasing number of published services on the Internet, service recommendation becomes desired when developers are developing mashups.

May recent studies are based on LDA, however, the restrictive distribution assumptions of LDA put a strict limitation to the performance of modeling mashup queries and service descriptions. In this paper, we propose a Time-aware Collaborative Poisson Factorization (TCPF) model to overcome the problem. Poisson Factorization is chosen as the foundation to model mashup queries and service descriptions separately, incorporate them with the historical usage data together using collective matrix factorization. Experiments on the real-world ProgrammableWeb dataset show that our model outperforms the state-of-the-art methods by 7.7% in terms of MAP.

In the future, we plan to take the comments information into consideration for a better performance of recommendation. For we believe the comments of mashups and services will play an increasingly important role in recommendation.

ACKNOWLEDGMENT

This research has been partially supported by the National Natural Science Foundation of China (No.61174169), and the Specialized Research Fund for the Doctoral Program of Higher Education (No. 20120002110034).

REFERENCES

[1] Liu, Xumin, and I. Fulia. "Incorporating User, Topic, and Service Related Latent Factors into Web Service Recommendation." In ICWS, 2015, pp.185-192.

[2] Zhang, J., Tan, W., Alexander, J., Foster, I., & Madduri, R. "Recommend-As-You-Go: A Novel Approach Supporting Services-Oriented Scientific Workflow Reuse." In SCC, 2011,pp.48-55

[3] Yu, Qi. "Decision Tree Learning from Incomplete QoS to Bootstrap Service Recommendation." In ICWS, 2012,pp.194-201

[4] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Qos-aware web service recommendation by collaborative filtering," IEEE Transactions on Services Computing, vol. 4, no. 2, pp. 140–152, 2011.

[5] Chukmol, Uddam, A. N. Benharkat, and Y. Amghar. "Bringing Socialized Semantics into Web Services Based on User-centric Collaborative Tagging and Usage Experience.." In APSCC, 2011.

[6] Liu, Fangfang, L. Wang, and J. Yu. "Context-aware similarity search of web service." In ICIST,pp.596-602, 2012

[7] Platzer, Christian, and S. Dustdar. "A vector space search engine for Web services." In ECOWS'05, pp.9, 2005.

[8] Tian, Q., L. Lei and L. Pin, Web Service Discovery with UDDI Based on Semantic Similarity of Service Properties. 2007. p. 454-457.

[9] Blei, David M., A. Y. Ng, and M. I. Jordan. "Latent dirichlet allocation." Journal of Machine Learning Research 3(2003):993-1022.

[10] Canny, John. "GaP: a factor model for discrete data." International Acm Sigir Conference on Research & Development in Information Retrieval 2004:p.122-129, 2004

[11] Zhong, Y., Fan, Y., Huang, K., Tan, W. "Time-Aware Service Recommendation for Mashup Creation." IEEE Transactions on Services Computing 8.3. pp.356-368, 2014

[12] Bai, B., Fan, Y., Huang, K., & Tan, W. "Service Recommendation for Mashup Creation Based on Time-Aware Collaborative Domain Regression." In ICWS, pp.209-216, 2015.

[13] Jiang, Y., Liu, J., Tang, M., & Liu, X. "An Effective Web Service Recommendation Method Based on Personalized Collaborative Filtering." In ICWS, pp.211-218, 2011.

[14] Gopalan, P., L. Charlin and D.M. Blei, "Content-based recommendations with Poisson factorization. " 2014: Montreal, QC, Canada. p. 3176 - 3184.

[15] D. Liang, J. Paisley, and D. Ellis. "Codebook-based scalable music tagging with Poisson matrix factorization." In ISMIR, 2014.

[16] Wei Zhang, J.W., "A Collective Bayesian Poisson Factorization Model for Cold-start Local Event Recommendation", in KDD'15. 2015.

[17] Koren, Y., R. Bell, and C. Volinsky. "Matrix factorization techniques for recommender systems." IEEE Computer 42.8(2009):30-37.

[18] Xu, Wei, X. Liu, and Y. Gong. "Document clustering based on non-negative matrix factorization." In ACM SIGIR, pp.267-273, 2003.

[19] Hoffman, M., Blei, D., Wang, C., & Paisley, J. "Stochastic variational inference. arXiv:1206.7051." Journal of Machine Learning Research 14.1(2012):1303-1347.

[20] McDowall, J. and L. Kerschberg, "Leveraging Social Networks to Improve Service Selection in Workflow Composition." 2012. p. 1278-1283.

[21] Xu, W., Hu, L., Wang, J., Li, M., & Cao, J. "A Social-Aware Service Recommendation Approach for Mashup Creation." International Journal of Web Services Research 10.1(2013):107-114.

[22] Jiang, B., Zhang, X., Pan, W., & Hu, B. "BIGSIR: A Bipartite Graph Based Service Recommendation Method." In IEEE SERVICES, pp.363-369, 2013.