A Social-Aware Service Recommendation Approach for Mashup Creation

Wenxing Xu[1], Jian Cao[1*], Liang Hu[1]
[1]Department of Computer Science and Engineering
[1]Shanghai Jiao Tong University
[1]Shanghai, China
[1]{wenxingxu, cao-jian, lianghu, li-ml}@sjtu.edu.cn

Jie Wang[2], Minglu Li[1]
[2]Department of Civil and Environmental Engineering
[2]Stanford University
[2] CA 94305, USA
[2]jiewang@stanford.edu

***Abstract*—Mashup is a user-centric approach to create value-added new services by utilizing and recombining existing service components. However, as services become increasingly more spontaneous and prevalent on the Internet, finding suitable services from which to develop a mashup based on users' explicit and implicit requirements remains a daunting task. Several approaches already exist for recommending specific services for users but they are limited to proposing only services with similar functionality. In order to recommend a set of suitable services for a general mashup based on users' functional specifications, a novel social-aware service recommendation approach, where multi-dimensional social relationships among potential users, topics, mashups, and services are described by a coupled matrix model, is proposed in this paper. Accordingly, a factorization algorithm is designed to predict unobserved relationships, and as a result, a comprehensive service recommendation model can be readily constructed. Experimental results for a realistic mashup data set indicate that the proposed approach outperforms other state-of-the-art methods.**

Keywords-service recommendation; social-aware; service social network; mashup creation

I. INTRODUCTION

Services-oriented computing has revealed a new paradigm and brought a technology revolution to traditional IT system development. As service technology has been developing rapidly, an increasing number of services are available on the Internet, and consequently terms such as Service Web and Internet of Services are proposed to describe this phenomenon.

Although a user can occasionally employ a single service to meet his needs directly, more often than not, the fulfillment of his needs relies on a set of composed services. Therefore, service composition has become a key technology of service-oriented computing and there is a tremendous amount of research on this particular topic. Recently, the mashup technique, which allows users to recompose existing services to create entirely new or additional value-added services, emerges as a promising service composition approach [1]. It essentially introduces a simple self-serve approach [2] in which every user is able to compose his own service applications by merely performing a "drag and drop" action within a web browser. Obviously from this perspective, a mashup is extremely consumer-centric and lightweight, and it becomes more attractive to most consumers using the Internet under the Web 2.0 paradigm. In recent years, a number of mashup platforms have been developed by various industry vendors.

When a user begins to develop a mashup, the first task is to discover and select suitable existing services. Since there is a huge number of services on the Internet, finding the appropriate ones can be really laborious even for an experienced user. Therefore, service recommendation is becoming increasingly critical as the number of services on the Internet continues to grow rapidly. Based on above observations, the problem we try to address in this paper is how to recommend, for a particular user's functional specification, a set of suitable services that the user can utilize to create a proper mashup. Although several service recommendation systems and approaches already exist, they are limited to proposing one or more services with similar functionality. However, in order to develop a mashup, a user often needs a set of related services rather than a single specific service. Furthermore, in order to more precisely recommend suitable services for a particular user, the user's implicit requirements, which are neglected by most current service recommendation systems and approaches, should be taken into consideration.

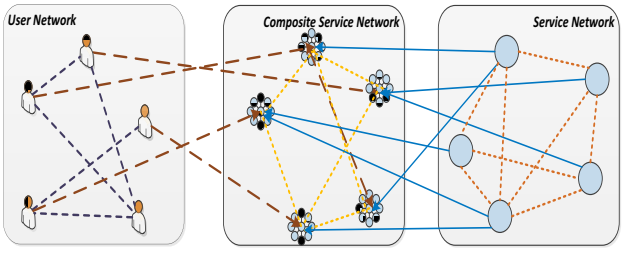

Figure 1. Services social network structure

Although users' implicit requirements and their relevant knowledge of certain tasks are difficult or even impossible to model, they can be partially inferred and then captured from some existing social networks formed over users, services, and previous mashups. Fig.1 depicts such a social network, where services, mashups, and users are mutually connected. Based on this network, the underlying relationships can be discovered. For example, two users with many common

*Jian Cao is the corresponding author

978-0-7695-5025-1/13 $26.00 © 2013 IEEE
DOI 10.1109/ICWS.2013.24

neighbors may likely share more common interests. For this reason, we propose a social-aware service recommendation approach for mashup creation. The contributions of this paper are three-fold: (1) we propose a coupled matrix model to describe the multi-dimensional relationships among users, mashups, and services, (2) we design a factorization algorithm to predict unobserved relationships in the model to support more accurate service recommendations, and (3) we present a set of experiments to evaluate our approach on a real data set. The experimental results indicate that our approach is promising and outperforms existing approaches.

II. Related Work

There are several systems and approaches previously developed to support service discovery and recommendation. Traditional information retrieval technologies such as vector space model and term frequency-inverse document frequency are used to build service search engines or service recommendation systems [3]. In these systems, candidate services are recommended according to the syntactic matches between the query string and service description documents. These approaches can be further improved by applying semantic technology [4]. Furthermore, in order to recommend services that are able to meet specific QoS requirements, a specific matrix can be defined to rank services [5]. However, all of these approaches will inevitably encounter the problem in which a user's complete requirements cannot be explicitly expressed when searching a service. Although some approaches have incorporated user preferences into the selection model, they can only be applied to rank services that have been selected based on explicit functional requirements [6].

Since services are chosen and invoked by multiple users, collaborative filtering (CF), which originates from the idea that people often get the best recommendations from others with similar previous experiences, has also been employed in service recommendation recently [7][8]. Although the CF model reflects part of users' implicit requirements, it ignores more extensive social relationships among users and services.

With services being widely deployed and utilized over the Internet, relationships within service social networks can be mined to help recommend and compose services [9][10]. However, these approaches only consider services relationships, not the interactions between users and services. Abderrahmane Maaradji *et al*. proposed a new approach for services recommendation to assist services composition using a mashup environment called SoCo [11]. In SoCo, proper services for a user are recommended based on an implicit social graph inferred from the common composition interests of users. In this paper, our approach also makes use of social information to support service recommendation, but the model of relationships is different since we address a different problem. For example, the services appearing in the same mashup have a co-appearance relationship in our model while their model only considers succeeding relationships. In addition, extra information, such as topic information, is also included in our model. These differences make our model more comprehensive when creating a mashup for recommending a set of services to satisfy users' functional specifications.

III. A Motivating Use Case

In this section, we will show a case study to illustrate how social information can enhance the process of service recommendation.

The user is planning to build a mashup application that emulate a social networking platform for Congress. First, the user describe the demand as "A mashup developed to emulate a social networking platform for Congress. It will use several APIs and links to: news, blogs, comments, bio information, voting records, campaign finance and more. ".

Then, recommendation engine starts up and incorporates the mashup's information, the services' information, the services provider's information, and also, the service consumer's information. Using the information of mashup and service, the engine will recommend the mashlets that have most common features with the demand description of mashup to user. On the other hand, recommendation engine will take the user's historical service invocation records and the mashlet provider's social features into consideration, give higher score to those mashlets whose provider have similar interest with the user, and those mashlets which the user have preference on them.

The recommendation-based method with focus on social relationship could enrich the composition scenario with additional services that were not initially taken into consideration during the specification of this scenario. In this senario, the recommendation engine will recommend YouTube, Flickr, Facebook and Eventful to the user, and the user finally select YouTube, Yahoo, Technorati, Open Secrets, Google Social Graph, Flickr, Eventful and Capitol Word to build the mashup application.

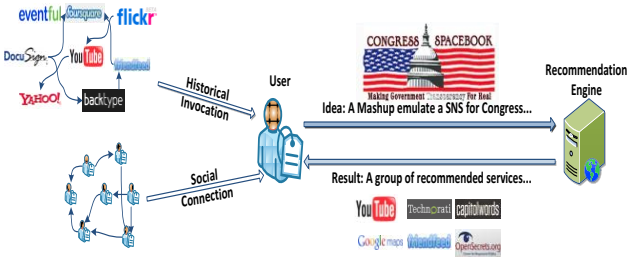

Figure 2. Service Recommendation Use Case.

IV. Model And Algorithms

A. Model Construction

The social-aware service recommendation model consists of a main matrix and several subsidiary matrices. The main matrix models relationships between mashups and services while subsidiary matrices help to incorporate additional relational information into the model.

1) Main Matrix

We denote the mashups as $M = \{m_1, m_2, \cdots, m_{|M|}\}$. Given a mashup m_i, we can retrieve the services that m_i includes, which are denoted as $S_{m_i} = \{s_{m_i,1}, s_{m_i,2}, \cdots, s_{m_i,|S_{m_i}|}\}$. A matrix X whose rows correspond to mashups and whose columns correspond to services can be constructed. Each entry of this matrix is assigned a binary value (1 or 0) to indicate whether a service is included in a mashup or not.

2) Subsidiary Matrices

Subsidiary matrices are applied to model other relationships. In our experiment, we construct matrices corresponding to *Mashups-Creators*, *Mashups-Topics*, *Services-Topics*, *Services-Providers*, and *Services-Tags* relationships.

For a user u_i, all mashups created by that user are extracted. Then the *Mashups-Creators* relationship can be represented by a binary matrix $\boldsymbol{MC}$ with rows corresponding to mashups and columns corresponding to users.

In order to model the relationships between a user's functional requirements and mashups, the semantic information of the specification must be obtained first. We use topic model to capture the semantic information. The topic model (TM) assumes that the distribution of words is different in different topics and each document can be treated as a mixture of k topics. Different topic models can be derived by different probabilistic generative processes of the documents. We use the LDA model [13] to capture the topic features. A LDA can discover topic distributions for each document, i.e., $P(w \mid d)$, with each topic described by words following a probability distribution, i.e., $P(w \mid z)$. This can be formalized as:

$$P(w_i \mid d) = \sum_{j=1}^{z} P(w_i \mid z_i = j) P(z_i = j \mid d) \quad (1)$$

where $P(w_i \mid d)$ is the probability of the i-th word for a given document d and z_i is a topic. $P(w_i \mid z_i = j)$ is the probability of w_i within topic j. $P(z_i = j \mid d)$ is the probability of generating a word from topic j in the document d. The number of topics Z is defined in advance and it can be used to control the differences among topics. LDA estimates the topic-word distribution $P(w \mid z)$ and the document-topic distribution $P(z \mid d)$ from an unlabeled corpus of documents using Dirichlet priors for the distributions and a fixed number of topics. We use the LDA algorithm to extract topic distributions from mashup descriptions so that a matrix $\boldsymbol{MT}$ for mashup-topic relationship can be constructed.

Similarly, we can define three additional matrices for all services. We use a binary matrix SP to denote the service-provider relationship. $\boldsymbol{ST}$ and $\boldsymbol{SG}$ are two other matrices that model the service-topic relationship and service-tag relationship respectively.

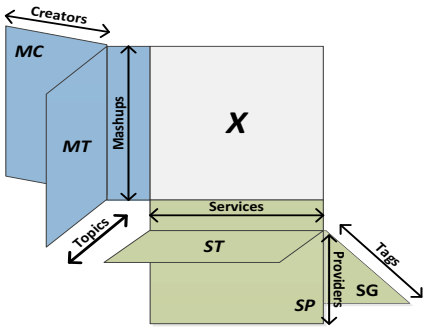

Figure 3. Coupled matrices model for service recommendation.

Therefore, we have a main matrix $\boldsymbol{X}$ and five subsidiary matrices. $\boldsymbol{MT}$ and $\boldsymbol{MC}$ couple with $\boldsymbol{X}$ on the "mashups" dimension. And $\boldsymbol{ST}$, $\boldsymbol{SP}$, and $\boldsymbol{SG}$ couple with $\boldsymbol{X}$ on the "services" dimension. (See Figure 2).

With this coupled matrix model in hand, the problem boils down to a blank value estimation problem. Inspired by the capability of matrix factorization techniques to factorize a matrix into two low-rank factor matrices, we propose a coupled matrix factorization approach to complete the value estimation task for our model.

B. Coupled Matrices Factorization Algorithm

So far we have built a coupled matrices model where all the data are arranged into a main matrix and subsidiary matrices. In this section, we discuss how to factorize these coupled matrices and predict desired services for mashup creation.

Typically, a relation modeled by a matrix $\boldsymbol{Y}$ can be factorized into two low-rank latent factor matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ [16], that is, $\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{V}^T$ where $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d$ (latent factor vectors) are the rows of $\boldsymbol{U}, \boldsymbol{V}$ and d is the dimensionality of the latent factor space. In this section, a graphical model based on the above constructed matrices is presented and a factorization method is also provided to find factor matrices so as to make a prediction.

1) Graphical Model for Coupled Matrices

With the main matrix $\boldsymbol{X}$ and five subsidiary matrices $\boldsymbol{MT}$, $\boldsymbol{MC}$, $\boldsymbol{ST}$, $\boldsymbol{SP}$, and $\boldsymbol{SG}$ in hand, the graphical model for these coupled matrices can be given by Figure 3. $\boldsymbol{X}$ can be factorized into F_M and F_S for dimensions *Mashup* and *Service*. Likewise, the factor matrices of $\boldsymbol{MT}$ for dimensions *Mashups* and *Topics* are represented by F_M and F_T; F_C and F_M are the factor matrices of $\boldsymbol{MC}$ for dimensions *Mashups* and *Creators*; F_S and F_T are the factor matrices of $\boldsymbol{ST}$ for dimensions *Services* and *Topics*; F_S and F_P are the factor matrices of $\boldsymbol{SP}$ for dimensions *Services* and *Providers*; F_S and F_G are the factor matrices of $\boldsymbol{SG}$ for dimensions *Services* and *Tags*. It is easy to see that $\boldsymbol{MT}$ and $\boldsymbol{MC}$ share the common factor matrix F_M with $\boldsymbol{X}$, i.e., F_M serves as the latent factor matrix for $\boldsymbol{X}$, $\boldsymbol{MT}$, and $\boldsymbol{MC}$, so that the mashup's creator information and mashup's topic distribution can influence the mashup preference for the services. Likewise, $\boldsymbol{ST}, \boldsymbol{SP}$, and $\boldsymbol{SG}$ share the common factor

matrix F_S with $\boldsymbol{X}$; $\boldsymbol{ST}$ shares the common factor matrix F_T with $\boldsymbol{MT}$. Since the shared latent factor matrices are learned through fitting multiple coupled matrices together, so we could set the weights (here are denoted as α in Figure 3) to scale the loss of fitting each matrix. The predicted main matrix $\widetilde{\boldsymbol{X}}$ can be approximated by $F_M {F_S}^T$, where each row of the mashup factor matrix F_M is the mashup's latent factor vector, which can be interpreted as the amount of mashup m_i's personalized preference over the latent factor of some services.

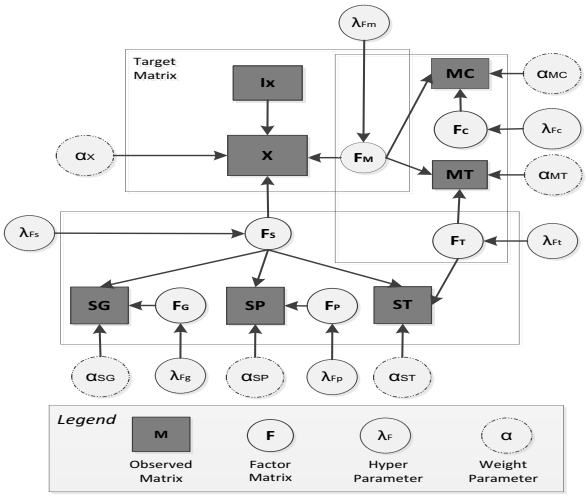

Figure 4. Graphical model for coupled matrices.

2) Coupled Matrices Factorization

According to the graphical model, it is easy for us to translate it into a loss function for optimization [14, 15] as follows:

$$f = L_X + L_M + L_S + \Omega \tag{2}$$

where f is composed of three parts of losses on all relations and regularization terms to prevent overfitting [17]. The objective function consists of the losses L_X, L_M, L_S and the regularization term Ω, where $\|\cdot\|$ denotes the Frobenius norms.

$$L_X = \frac{\alpha_X}{2}\left\|I_X * (X - F_M {F_S}^T)\right\|^2 \tag{3}$$

$$L_M = \frac{\alpha_{MT}}{2}\left\|MT - F_M {F_T}^T\right\|^2 + \frac{\alpha_{MC}}{2}\left\|MC - F_M {F_C}^T\right\|^2 \tag{4}$$

$$L_S = \frac{\alpha_{ST}}{2}\left\|\left(ST - F_T {F_S}^T\right)\right\|^2 + \frac{\alpha_{SP}}{2}\left\|SP - F_P {F_S}^T\right\|^2 + \frac{\alpha_{SG}}{2}\left\|SG - F_G {F_S}^T\right\|^2 \tag{5}$$

$$\Omega = \sum_{F\in(F_M,F_S,F_T,F_C,F_P,F_G)} \frac{\lambda_F}{2}\|F\|^2 \tag{6}$$

L_X models the loss on fitting the rating matrix, where I_X denotes the indicator matrix in which the entry **0** represents unused services and **1** stands for used ones. The operator $*$ denotes the element-wise product.

L_S and L_B (Eq. 4 & 5) are the loss functions for subsidiary matrices. F_i is the linear concatenation of auxiliary latent factor matrices and the weights of these matrices are determined by the parameter α_i.

Simply minimizing a loss may lead to overfitting, so the function Ω is augmented to normalize all factor matrices. λ_F in Eq. (6) is used to control the penalty of each latent factor matrix.

In order to minimize the object function, we need to compute the gradient with respect to each factor matrix and then we can use any first-order optimization algorithm. The partial derivatives of Ω are easy to derive with respect to each factor matrix as follows:

$$\frac{\partial \Omega}{\partial F} = \lambda_F F_{M,} F_S, F_T, F_C, F_P, F_G$$

The partial derivatives of L_X with respect to F_M and F_S are given as follows:

$$\frac{\partial L_X}{\partial F_M} = [I_X * (F_M {F_S}^T - X)]F_S$$

$$\frac{\partial L_X}{\partial F_S} = [I_X * (F_M {F_S}^T - X)]F_M$$

More details of the derivation can be found in [14, 15]. The partial derivatives with respect to other factor matrices F_T, F_C, F_P, F_G are zeros.

Similarly, the partial derivatives of L_M and L_S are given by:

$$\frac{\partial L_M}{\partial F_T} = \alpha_{MT}\left(F_M {F_T}^T - MT\right)F_M$$

$$\frac{\partial L_M}{\partial F_C} = \alpha_{MC}(F_M {F_C}^T - MC)F_M$$

$$\frac{\partial L_M}{\partial F_M} = \alpha_{MT}\left(F_M {F_T}^T - MT\right)F_T + \alpha_{MT}\left(F_M {F_C}^T - MC\right)C$$

$$\frac{\partial L_S}{\partial F_T} = \alpha_{ST}\left(F_T {F_S}^T - ST\right)F_S$$

$$\frac{\partial L_S}{\partial F_P} = \alpha_{SP}\left(F_P {F_S}^T - SP\right)F_S$$

$$\frac{\partial L_S}{\partial F_G} = \alpha_{SG}\left(F_G {F_S}^T - SG\right)F_S$$

$$\frac{\partial L_S}{\partial F_S} = \alpha_{ST}\left(F_T {F_S}^T - ST\right)F_T + \alpha_{SP}\left(F_P {F_S}^T - SP\right)F_P$$

$$+\alpha_{SG}\left(F_G {F_S}^T - SG\right)F_G$$

From Eq. (4), we can easily write the partial derivatives of f with respect to $F \in \{F_M, F_S, F_T, F_C, F_P, F_G\}$:

$$\frac{\partial f}{\partial F} = \frac{\partial L_X}{\partial F} + \frac{\partial L_M}{\partial F} + \frac{\partial L_S}{\partial F} + \frac{\partial \Omega}{\partial F}$$

The gradient of f can be written by vectorizing $\frac{\partial f}{\partial F}$ and then we can obtain Eq. (7):

$$\nabla f = \left[vec\left(\frac{\partial f}{\partial F}\right) \right]_{F \in \{F_M, F_S, F_T, F_C, F_P, F_G\}} \quad (7)$$

Now that we have built the objective function and derived its gradient, we can use some type of gradient-based optimization algorithm like the Nonlinear Conjugate Gradient (NCG) [17] or the Limited-Memory BFGS method (L-BFGS) to compute the factor matrices. As shown in Algorithm 1, we use an iterative algorithm to approximate the factor matrices:

Algorithm 1: Factorization by Iterative Gradient Descent

Initialize $\{F_M, F_S, F_T, F_C, F_P, F_G\}$ randomly;
$T \leftarrow$ *the maximal number of iterations;*
$t \leftarrow 1$;
while $t \leq T$
Compute the objective function f by Equation (2);
Compute the gradients ∇f by Equation (7);
Update $\{F_M, F_S, F_T, F_C, F_P, F_G\}$ by gradient descent algorithm;
if convergence then
break;
end if
$t \leftarrow t + 1$;
end while
Output $\{F_M, F_S, F_T, F_C, F_P, F_G\}$;

C. Service Recommendation

With this factorization model at hand, we can now focus on how to recommend a suitable set of services that a user is likely to employ to create the mashup based on that user's functional specification. During the service recommendation process, explicit profile knowledge about a user and a service is trivial to obtain; we run topic model to extract implicit information from the user's functional specification and incorporate this new vector into our model.

Through algorithm 1, we can obtain all the low rank latent factor matrices. To infer the potential services for mashup creators, we can reconstruct the missing data straightforwardly from the latent factor matrices F_M and F_S of main matrix $\boldsymbol{X}$:

$$\tilde{X} = F_M {F_S}^T \quad (10)$$

Given a mashup m_i, the recommendation ranks for the candidate services can be given by sorting the reconstructed values in a descending order:

$$rec(m_i) = \tilde{R}(m_i) \downarrow \quad (11)$$

We organize our service recommendation procedure as follows:

Procedure : Service Recommendation

Input user ID ***u*** and functional specification;
Create a new mashup ID ***m***, $|\mathrm{M}| \leftarrow |\mathrm{M}| + 1$;
Run topic model on the specification to get topic vector ***T***;
Add an entry to ***MT*** with vector ***T***;
Add an entry to ***MC*** with ***[u, m]***
Run the coupled factorization algorithm to obtain $\{F_M, F_S, F_T, F_C, F_P, F_G\}$
Construct the predict matrix $\widetilde{\boldsymbol{X}} = F_M {F_S}^T$
Sorting the reconstructed values $rec(m_i) = \widetilde{\boldsymbol{R}}(m_i) \downarrow$
Output the top *N* services;

V. EXPERIMENTS

A. DataSet

In order to construct a network view of the services ecosystem, we turned to the largest online repository of information about Web 2.0 mashups and services, namely, the ProgrammableWeb.com. The research of Shuli Yu and C. Jason Woodard [18] showed the services social network constructed on ProgrammableWeb.com dataset has the small-world network property, which means its characteristic path length is similar to that of a random network with the same density despite having a much larger clustering coefficient. This suggests that services with very different functionality are more likely to be connected through mashups than one might otherwise expect, and the dataset is realistic and suitable for our experiment.

This aggregator site provides the most comprehensive lists of services and mashups to date. Details about who creates a mashup and the services each mashup consists of are also available in the site. Our data set consists of 6691 mashups, 6111 services, and 2369 users. Table I lists detailed information about this data set.

TABLE I. STATISTICS OF PROGRAMMABLEWEB DATA FOR EVALUATION

#Services	# Mashups	#Users	Max. # Services for a User	Max. # Services of a Mashup
6111	6691	2369	81	54

We collected all profiles of the services, mashups, and users, which include the descriptions of services, types and tags of services, descriptions of mashups, and some attributes of users. We extracted topics of mashups and services from their descriptions using the LDA model. In the experiment, we set the number of topics to 100.

B. Comparative Methods

The following methods are evaluated for comparison, including our model and other state-of-the-art models.

MF: The MF model minimizes the regularized squared error by stochastic gradient descent.

User-Based: For a user-based algorithm, we first calculate the similarity, $w_{u,v}$, between the users u and v who have co-rated the same set of items. Typically, the similarity can be measured by computing the Pearson correlation:

$$w_{u,v} = \frac{\sum_{p \in p_{u,v}} (r_{u,p} - \bar{r}_u)(r_{v,p} - \bar{r}_v)}{\sqrt{\sum_{p \in p_{u,v}} (r_{u,p} - \bar{r}_u)^2 \sum_{p \in p_{u,v}} (r_{v,p} - \bar{r}_v)^2}} \quad (12)$$

where $\boldsymbol{p}_{u,v} = \boldsymbol{p}_u \cap \boldsymbol{p}_v$ ($\boldsymbol{p}_u = \bigcup_{d \in \boldsymbol{D}} \boldsymbol{p}_u^d$, $\boldsymbol{p}_v = \bigcup_{d \in \boldsymbol{D}} \boldsymbol{p}_v^d$) denotes the items over all domains $\boldsymbol{D}$ co-rated by u and v; $r_{u,p}$ and $r_{v,p}$ are the ratings on item p given by users u and v respectively; $\bar{r}_u$ is the average rating of user u for all the items rated. Then, the predicted rating of an item p for user u can be calculated by a weighted average strategy:

$$\hat{r}_{u,p} = \frac{\sum_{v \in U_{u,p}^k} w_{u,v} r_{v,p}}{\sum_{v \in U_{u,p}^k} |w_{u,v}|} \quad (13)$$

where $\boldsymbol{U}_{u,p}^{\boldsymbol{k}}$ denotes the set of top k users (k neighbors) that are most similar to user u who rated item p. To compensate for ratings scale variations, Eq. (14) is often used to adjust for users' mean ratings:

$$\hat{r}_{u,p} = \bar{r}_u + \frac{\sum_{v \in U_{u,p}^k} w_{u,v} (r_{v,p} - \bar{r}_v)}{\sum_{v \in U_{u,p}^k} |w_{u,v}|} \quad (14)$$

Item-Based: The item-based method also needs to compute the similarity, $w_{p,q}$, between a pair of items p and q. Given co-rated cases $U_{p,q}$ over items p and q, i.e., each case is that a user rated both p and q, the Pearson correlation is given as follows:

$$w_{p,q} = \frac{\sum_{u \in U_{p,q}} (r_{u,p} - \bar{r}_p)(r_{u,q} - \bar{r}_q)}{\sqrt{\sum_{u \in U_{p,q}} (r_{u,p} - \bar{r}_p)^2 \sum_{u \in U_{p,q}} (r_{u,q} - \bar{r}_q)^2}} \quad (15)$$

Then, the predicted value, $\hat{r}_{u,p}$, is taken as a weighted average of the ratings for neighboring k items rated by u, denoted $\boldsymbol{P}_{u,p}^{\boldsymbol{k}}$:

$$\hat{r}_{u,p} = \frac{\sum_{q \in P_{u,p:}^k} w_{p,q} r_{u,q}}{\sum_{v \in U_{u,p}^k} |w_{p,q}|} \quad (16)$$

Semantic-Based: The semantic-based method analyzes a set of descriptions of services previously used in a mashup, and builds a model of mashup interests based on the features of the services used by that mashup.

Hybrid: This method uses the historical composite services record to optimize the links in the semantic-based prediction result. The ranking score is the weighted average score of MF method and semantic-based method.

C. Evaluation Metrics

Recommending services to the user is equivalent to returning the top N ranked entries of a query. Hence it is natural to use ranking metrics to evaluate the performance. In the experiments, we used two commonly used metrics in information retrieval. They are defined as follows:

Precision@N: Precision of the top N ranked entries of each query:

$$Precision@N = \frac{|Result@N \cap Ob@N|}{N} \quad (17)$$

where Result@N is the predicted top N entries and Ob@N is the truly observed top N entries. We report the average precision over all test data.

MAE: Mean absolute error (MAE) that measures the rating prediction quality. MAE is defined as:

$$MAE = \sum_{r_{u,p} \in D_{test}} \frac{|r_{u,p} - \hat{r}_{u,p}|}{N} \quad (18)$$

where $r_{u,p}$ denotes the rating user u gave to item p, $\hat{r}_{u,p}$ is the predicted rating u gave to p, and N denotes the number of ratings for testing in $\boldsymbol{D}_{test}$.

D. Experiment Results

1) Selection of the result set size N

As for the recommendation system, the number of recommended entries significantly influences the recommendation performance and user experience. It is nearly impossible for a very small result set to achieve a satisfactory MAP, while a large result set would get a high MAP but it is not practical to push a lengthy candidate list for our mashup creator. The average number of services a mashup used in our dataset is 4.85. Figure 4 reports the MAP of different methods on varying sizes of the result set. Based on this, we chose N= 20 in our following experiments as it is a reasonable length for a candidate list and it generates a relatively high performance. Also, as we can see in the result, our social-aware method creates a more accurate ranking than other models:

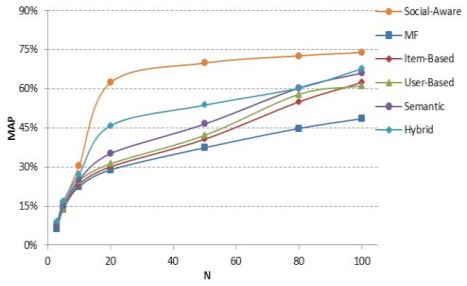

Figure 5. MAP of different methods on varying N

2) Impact of Data Density

Recommendation performance is also influenced by data density. Data density means how many records in the matrix

can be employed to predict the missing values. In order to study the impact of training data density, in this experiment, we vary density from 0.05 to 0.95. Figures 5 and 6 display the impact of data density on prediction accuracy for different approaches.

TABLE II. RESULTS OF MAP BY VARYING DATA DENSITY

Data Density(%)	5%	20%	80%	95%
MF	30.33%	33.89%	31.46%	32.59%
Item-Based	30.10%	30.06%	34.41%	40.28%
User-Based	30.82%	31.21%	34.70%	35.63%
Semantic	35.12%	35.12%	35.12%	35.12%
Hybrid	41.92%	45.84%	47.24%	50.71%
Social-Aware	**50.27%**	**62.34%**	**65.43%**	**69.88%**

TABLE III. RESULTS OF MAE BY VARYING DATA DENSITY

Data Density(%)	5%	20%	80%	95%
MF	0.9314	0.7228	0.5327	0.2278
Item-Based	0.9978	0.9323	0.7219	0.4654
User-Based	0.9817	0.8823	0.6547	03822
Semantic	0.9322	0.9322	0.9322	0.9322
Hybrid	0.9526	0.8728	0.6945	0.3889
Social-Aware	**0.7515**	**0.5298**	**0.4350**	**0.1577**

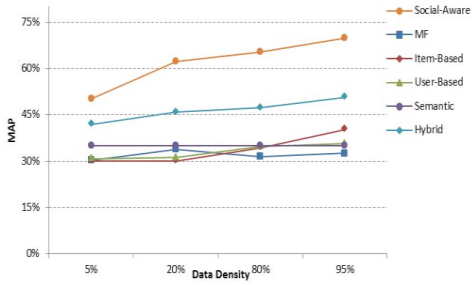

Figure 6. Results of MAP by varying data density

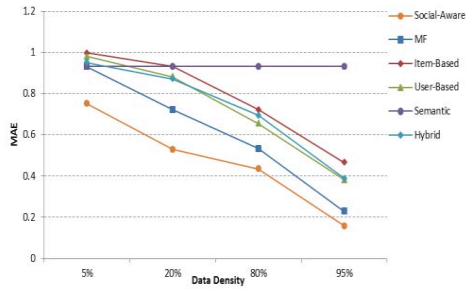

Figure 7. Results of MAE by varying data density

3) *Impact of Weights*

Our method offers a flexible mechanism to control the amount of impact from other sources on the target domain by tuning the weights of those sources. Too large a weight may impose too heavy an impact on the target domain, overwhelming the information learned from the target domain. On the other hand, too small a weight for auxiliary domains may cause the extra knowledge to be ignored in the target domain, resulting in not learning the global user preference. In experiments, we vary the weight on one source of the factors from 2^{-5} to 2^{5} step by 2, and fix the weight on other sources to be 1. Figures 7 and 8 depict the MAPs and MAEs with different weights on *mashup creator factor(MC)*, *mashup-topic factor(MT)*, *service provider factor(SP)*, *service-topic factor(ST), and service-tag factor(SL)*. As expected, we find that the best performance is achieved by choosing different weights for different sources. The optimal weight hence may depend on the distribution of the value of the factor and the correlation with the target domain.

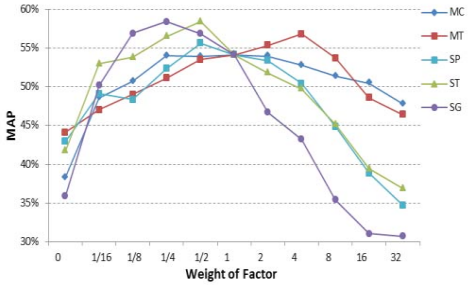

Figure 8. Results of MAP by varying weights

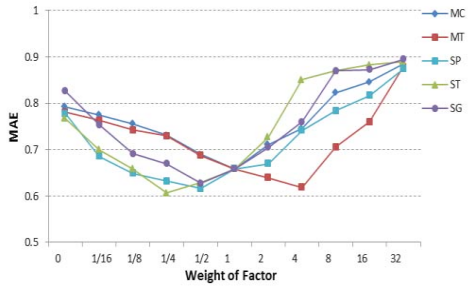

Figure 9. Results of MAE by varying weights

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we have discussed the necessity of considering social networks in the process of services recommendation and how this can help improve recommendations. We demonstrated that not only the social relationship between users but also the interactions between services are important for a more accurate recommendation, and it is more effective and feasible to view process information as a useful knowledge source for services recommendation.

To utilize the implicit knowledge both in text profiles and social networks, we used topic model to extract the implicit semantic information and proposed a new social-aware approach for services recommendation. Then we utilized the framework to design and implement an algorithm to rank services using a coupled matrices factorization method.

Finally, we performed a set of experiments to validate our approach on a realistic dataset. Experimental results

indicated that our approach outperformed other existing approaches for this particular dataset.

In the future, we plan to build an online collaboration platform based on this model. We believe that the more the model is used, and the more it collects information on how users build mashups for existing services, the better the resulting recommendations will be. In this case, we probably need to consider how to update the model to dynamically analyze information in real time, and to closely follow the latest trends in services selection and composition by users. And with the services social network become larger and larger in the future, we should verify whether the processing time scales with the number of services in the social network.

It should be mentioned that our coupled matrices model is also extendable and we can add more matrices to include more relationships if necessary. Of course, the weight value for each matrix should be carefully assigned due to changing contexts of specific experiments. We believe it is possible to design an approach to discover the best weighting strategy for a given model automatically. In this paper, we simply apply the topic model to capture the semantic information of the functional specification. Obviously, this is a simplification of the real semantics of the specification. If the order of the words and its implication in the specification can be considered, it's possible to suggest the order among services in the mashup to be created. This is also part of the future work we are going to explore.

VII. ACKNOWLEDGMENT

This work is partially supported by China National Science Foundation (Granted Number 61073021, 61272438), Research Funds of Science and Technology Commission of Shanghai Municipality (Granted Number 11511500102, 12511502704), Cross Research Fund of Biomedical Engineering of Shanghai Jiaotong University (YG2011MS38).

REFERENCES

[1] Liu, Xuanzhe, et al. "Towards service composition based on mashup." Services, 2007 IEEE Congress on. IEEE, 2007.

[2] Blake, M. Brian, and Michael F. Nowlan. "A web service recommender system using enhanced syntactical matching." Web Services, 2007. ICWS 2007. IEEE International Conference on. IEEE, 2007.

[3] Athman Bouguettaya, Surya Nepal, Wanita Sherchan, Xuan Zhou, Jemma Wu, Shiping Chen, Dongxi Liu, Lily Li, Hongbing Wang, Xumin Liu, "End-to-End Service Support for Mashups," IEEE Transactions on Services Computing, vol. 3, no. 3, pp. 250-263, July-Sept. 2010, doi:10.1109/TSC.2010.34.

[4] C. Platzer, S. Dustdar, A Vector Space Search Engine for Web Services, Web Services, 2005. ECOWS 2005. Third IEEE European Conference on In ECOWS '05: Proceedings of the Third European Conference on Web Services (2005), 62.

[5] Asma Adala, Nabil Tabbane, and Sami Tabbane, A Framework for Automatic Web Service Discovery Based on Semantics and NLP Techniques, Advances in Multimedia, Vol. 2011 (2011), doi:10.1155/2011/238683.

[6] Raj, R.J.R. , Sasipraba, T. , Web service recommendation framework using QoS based discovery and ranking process, 2011 Third International Conference on Advanced Computing (ICoAC), 14-16 Dec. 2011, 371 – 377.

[7] Liwei Liu, Nikolay Mehandjiev, Dong-Ling Xu: Multi-criteria service recommendation based on user criteria preferences. RecSys 2011: 77-84.

[8] Z. Zheng, H. Ma, M. R. Lyu and I. King. "QoS-Aware Web Service Recommendation by Collaborative Filtering", IEEE T. Services Computing, pp. 140 – 152, 2011.

[9] Jiang, Y., Liu, J. Tang, M., Liu, X. An Effective Web Service Recommendation Method Based on Personalized Collaborative Filtering. In ICWS (2011), 211 – 218.

[10] Yuanbin Han, Shizhan Chen, Zhiyong Feng, Optimizing Service Composition Network from Social Network Analysis and User Historical Composite Services, AAAI Technical Report SS-12-04 Intelligent Web Services Meet Social Computing, AAAI Technical Report SS-12-04 Intelligent Web Services Meet Social Computing .

[11] Maamar, Z. Santos, P. ; Wives, L. ; Badr, Y. ; Faci, N. ; de Oliveira, J.P.M. Using Social Networks for Web Services Discovery Internet Computing, IEEE , July-Aug. 2011 , Vol. 15(4): 48 – 54.

[12] Maaradji, Abderrahmane, et al. "Social Discovery and Composition of Web Services." EUD4Services Workshop-Empowering End-Users to Develop Service based Applications; Torre Canne, Italy, June 2011. 2011.

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

[14] Ma, H., Yang, H., Lyu, M. R. and King, I. SoRec: social recommendation using probabilistic matrix factorization. In Proceeding of the 17th ACM conference on Information and knowledge management (Napa Valley, California, USA, 2008).

[15] Salakhutdinov, R. and Mnih, A. Probabilistic matrix factorization. In Advances in neural information processing systems (2008).

[16] Koren, Y., Bell, R. and Volinsky, C. Matrix factorization techniques for recommender systems. Computer, 42, 8 2009), 30-37.

[17] Singh, A. P. and Gordon, G. J. Relational learning via collective matrix factorization. In Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (Las Vegas, Nevada, USA, 2008).

[18] Yu S, Woodard C J. Innovation in the programmable web: Characterizing the mashup ecosystem[C]//Service-Oriented Computing–ICSOC 2008 Workshops. Springer Berlin Heidelberg, 2009: 136-147.