Web API service recommendation for Mashup creation

Gejing Xu

School of Software,
Quanzhou University of Information Engineering,
Quanzhou 362000, China
Email: xugejing2021@163.com

Sixian Lian and Mingdong Tang*

School of Information Science and Technology,
Guangdong University of Foreign Studies,
Guangzhou, 510000, China
Email: 20191010003@gdufs.edu.cn
Email: mdtang@126.com
*Corresponding author

Abstract: Mashup refers to a sort of web application developed by reusing or combining web API services, which are very popular software components for building distributed applications. As the number of open web APIs increases, to find suitable web APIs for Mashup creation, however, becomes a challenging issue. To address this issue, a number of web API service recommendation methods have been proposed. Content-based methods rely on the description of the service candidates and the user's request to make recommendations. Collaborative filtering-based methods use the invocation records of services generated by a set of users to make recommendations. There are also some studies employing both the description and invocation records of services to make recommendations. In this paper, we survey the state-of-the-art web API service recommendation methods, and discuss their characteristics and differences. We also present some possible future research directions in this paper.

Keywords: web service; recommendation; collaborative filtering; Mashup creation.

Reference to this paper should be made as follows: Xu, G., Lian, S. and Tang, M. (2023) 'Web API service recommendation for Mashup creation', *Int. J. Computational Science and Engineering*, Vol. 26, No. 1, pp.45–53.

Biographical notes: Gejing Xu received her Bachelor degree in 2004 and Master degree in 2014 from the Huaqiao University. She is now working as a Lecturer at the Quanzhou University of Information Engineering. Her current research interests include software engineering and big data analytics.

Sixian Lian is currently pursuing his Masters degree with the School of Information Science and Technology, Guangdong University of Foreign Studies, Guangzhou, China. His research interests include services computing and data analytics.

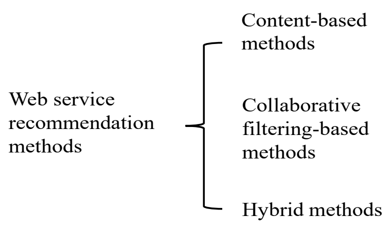
Mingdong Tang is a Professor in the School of Information Science and Technology at Guangdong University of Foreign Studies, Guangzhou, China. He received his PhD in Computer Science from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2010. He has published more than 100 peer-reviewed research papers in various academic journals and conferences. His current research interests include software engineering and data mining.

1 Introduction

Web application programming interface (API) services have become a popular technology and basic blocks for building distributed applications (Bouguettaya et al., 2014; Papazoglou et al., 2007; Yu et al., 2008). Traditionally, web API services (or web services for short) are mainly used for data sharing and business integration in an enterprise or a group of enterprises with cooperation relationships. In contrast, nowadays there are more and more enterprises engaging in opening their web services to the public and allowing customers or the third parties to access their data or functionalities in a programmable way. For instance, the IT giants like Google, Facebook, Amazon, etc. have already published dozens of web services for public access. By doing this, the enterprises can seize more economic benefits and eventually build business ecosystems surrounding their products. These open web services with various functionalities such as storage, messaging, computing, searching and mapping, have made the web become a huge programmable platform. That is, common developers can easily search for appropriate web services and employ them to create new applications or value-added services in a prompt way. With the prevalence of cloud, mobile and IoT applications, it can be expected that web services will become more and more popular in software development (Tan et al., 2016).

With the constant increase of the number of web services, to find suitable services for developers (i.e., service users) also becomes increasingly challenging (Cappiello et al., 2010; Yang et al., 2021). As can be observed from the most popular website of open web services, ProgramableWeb.com, most web services are described with plain text. Although keyword-based search can be used to relief the difficulty in finding web services, it is still a non-trivial job for users to efficiently and accurately identify the appropriate web APIs that meet their application development requirements. To address this issue, dozens of studies have been conducted on recommending web services to users based on their implicit or explicit requirements in the past decade. Techniques used for Web service recommendation can be roughly divided into content-based, collaborative filtering (CF)-based and hybrid. This paper surveys these works and compares their features. The paper also presents the challenges facing web service recommendation and suggests several future research directions.

Figure 1 Categorisation of web service recommendation methods



The rest of this paper is organised as follows. Section 2 introduces the background and problem of web service recommendation. Section 3 surveys the state-of-the-art web service recommendation methods and compares their features. Section 4 presents the challenges facing web API recommendation and possible research directions. Finally, Section 5 concludes this paper.

2 Background

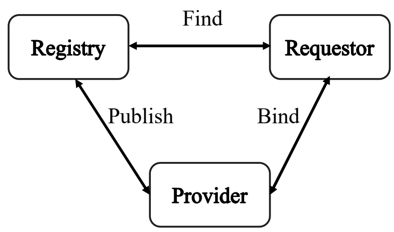
2.1 From SOAP-based services to REST-based services

Web services are software modules that can be remotely invoked using an internet-based protocol such as HTTP. A web service usually has a set of APIs, and each API represents a certain function.

With the evolution of web service technologies, there are different styles for the implementation of web services, and among which the major two implementation styles of web services are simple object access protocol (SOAP)-based and representational state transfer (REST)-based (Lanthaler and Gütl, 2010). SOAP-based web APIs, usually referred to as 'classical' web services, emerges before REST-based web APIs. The SOAP-based web service architecture complies with SOA, which comprises three entities (as shown in Figure 2):

- A service provider, which creates a service and publishes the service description in the service registry.
- A service registry, which enables online service discovery.
- A service requestor, which finds the service by querying the service registry. The requestor then retrieves the service description, uses it to bind to the service implementation, and begins interacting with it.

Figure 2 Model of service oriented architecture



SOAP-based web APIs (services), namely classical web services, are characterised by a set of standard protocols established by World Wide Web Consortium (W3C), such as SOAP, web service description language (WSDL) and universal description, discovery and integration (UDDI). SOAP is a protocol for exchanging XML-based messages over networks, normally based on HTTP. WSDL is an XML-based language for describing how to use a web service, including description of the service methods and binding information. UDDI is a protocol used for Web service publish and discovery.

REST-based services (or RESTful services) are nowadays much more prevalent than SOAP-based web services for its simplicity, scalability and easy to use (Hsieh et al., 2011). According to some statistics (Jiang et al., 2012), REST-based web services have taken more than 80% ratios in the open web service market. Different from SOAP-based web services, REST-based web services are often described solely through HTML or plain text that are thought for human understanding purpose. Although there exist proposals for improving the web services' understandability for machines, such as web application description language (WADL) and WSDL 2.0, they are rarely used in practice. Normally, REST-based services use HTTP to exchange messages in their invocations, and their input/output data are usually in JavaScript object notation (JSON) format. REST-based services also do not have any

UDDI-based registry supporting automatic publish/ discovery, and usually users have to find satisfactory web services manually.

2.2 Service-Mashup ecosystem

Mashup is a new application development pattern that combines different resources such as data and services distributed on the web to create innovative applications. The data or service providers usually open their data or services by offering an API. There have been a number of tools proposed for Mashup development, such as IBM Mashup Centre and Google Mashup Editor. Traditionally, the Mashup technique was mainly used for web application development. With the spread of mobile computing and internet of things (IoT), Mashup also attracts increasing attention in developing mobile and IoT applications (Im et al., 2013; Cao et al., 2019a).

Figure 3 Mashups and web services (see online version for colours)

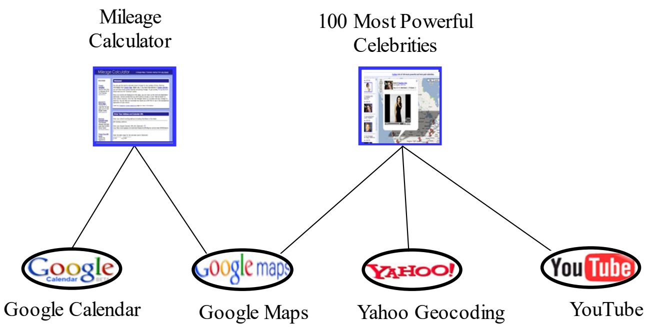


Figure 3 is an example illustrating the relationships between Mashups and web services. The Mashup 'mileage calculator', which can be used to calculate travel mileage for any number of trips, is created by combining the Google Calendar services and the Google Maps services. Actually, with the prevalence of web services and mashups, service-Mashup ecosystems have emerged (Salminen and Mikkonen, 2012). As a sort of software ecosystem (Bosch, 2009), a service-Mashup ecosystem cannot only improve the efficiency of application development, but also promote the reuse of software components around the world (Mikkonen and Taivalsaari, 2010; Taivalsaari, 2009).

ProgrammableWeb.com as the largest web service portal has contributed a lot to raise the service-Mashup ecosystem. It offers the functions that users can freely publish web services or Mashups, search web services or Mashups, or comment them. To character ProgrammableWeb, we crawled all web services and Mashup from it, and conduct a statistical analysis. Table 1 presents the statistics of the data of ProgrammableWeb.com. To the access date, the number of available web services is about 21,900 and the number of available Mashups is about 6,435. All web services fall into 425 primary categories like mapping, search, social, eCommerce, photos, etc. Most Mashups use only one or two APIs, though there exist a few Mashups that comprise dozens of web services. The average number of services used by a Mashup is about 2, as shown in the table. The web services and Mashups collected by ProgrammableWeb are likely to be only a small proportion of all web services and applications on the internet. There could be millions of applications that are developed by reusing or combining open web services. However, how to obtain those API usage data is a challenging problem.

Table 1 Statistics of the service-Mashup ecosystem of programmableweb

Property	*Value*
Number of web services	21,900
Number of Mashups	6,435
Number of service categories	425
Average number of services per Mashup	2.07
Length of service description (#words)	44
Length of Mashup description (#words)	27

2.3 The web service recommendation problem

This survey focuses on web service recommendation for Mashup creation. We focus on REST-based web services that are normally described in HTML or plain text. We do not assume that web services are described in structured languages like WSDL, for they are not the mainstream implementation of REST-based services. The service providers or developers may use a short text to describe their services or Mashups when publishing them in web service portals. Figure 4 is an example showing the description of web services (API) in ProgrammableWeb. The problem of web service recommendation is formally defined as follows.

Suppose that we have a set of web services, a set of Mashups and a set of historical invocation records occurred between them. If a Mashup has invoked a web service, there is an invocation record for them. Such relationships between services and Mashups can be represented by using a graph or matrix. Figure 5 shows the graph or matrix representations of the example from Figure 1. In the graph, each link is used to represent the invocation relationship between a Mashup and a service; while in the matrix, each item with value 1 represents an invocation relationship. Under this setting, the web service recommendation problem can be defined as: given a web service set, a Mashup set, and their invocation relationships, for a new Mashup with requirement description, to recommend a list of web services that best match the requirement of the new Mashup application.

Figure 4 Description of web services (APIs) in programmableweb (see online version for colours)

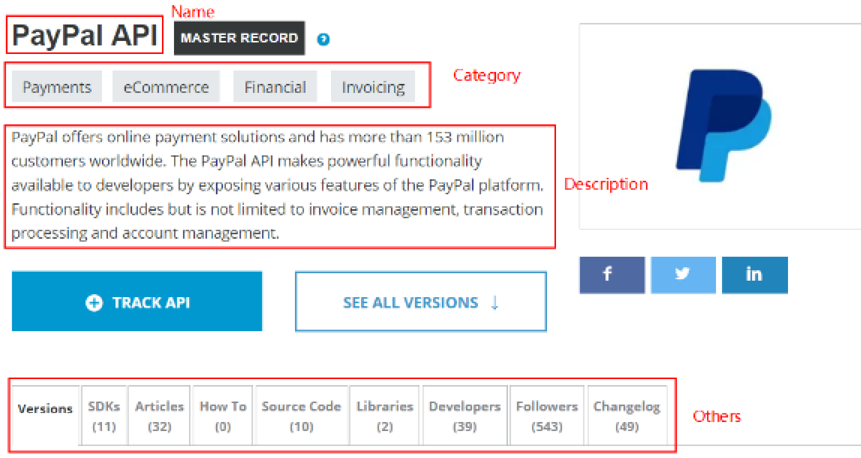
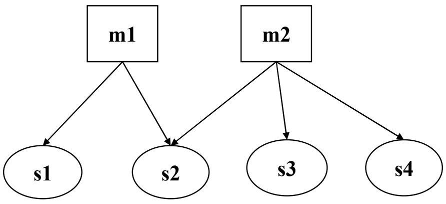


Figure 5 Representation of the relationships between web APIs and Mashups, (a) graph representation, (b) matrix representation



(a)

	s1	s2	s3	s4
m1	1	1	0	0
m2	0	1	1	1

(b)

3 The state-of-the-art methods

The state-of-the-art methods on web service recommendation can roughly be categorised into: content-based, CF-based and hybrid methods, as shown in Figure 1. In the following, we discuss and analyse the web service recommendation methods fallen in each category.

3.1 Content-based methods

Content-based service recommendation methods usually exploit only the description of users' requirements and web services' functionalities to measure their matching degrees, and recommend the most matched web services to the users. Content-based methods have been widely used for classical web service recommendation or discovery, which are typically described in WSDL. Some of them can also be applied to web services with unstructured description such as REST-based services. Generally, there are two categories of content-based web service recommendation methods: syntactic matching and semantic matching.

3.1.1 Syntactic matching

This kind of method usually employs classic text retrieval methods to find suitable web services. For example, Halevy et al. (2004) used TF/IDF to extract keywords from the description documents of web services. The extracted keywords as well as input/output parameters are both used to make matching's between web services to recommend similar web services. Tang et al. (2014) also employed the TF/IDF model to represent web services, and then measure its similarity with the user query. The most similar web services are finally recommended to the user. Wu (2012) introduced term tokenisation to relieve the syntactic inconsistency between the description of Web services and user queries, so as to improve the accuracy of syntactic matching. Liu et al. (2010) employed an external knowledge, e.g., lexical databases, to improve syntactic matching of web services.

3.1.2 Semantic matching

Semantic matching was introduced to overcome the inaccuracy of syntactic matching in web service recommendation or discovery. Semantic matching is usually based on the assumption that both web services and user requests are described in a machine understandable language. Several ontology languages have been proposed for web services, such as SAWSDL (Kopecký et al., 2007) and OWL-S (Martin et al., 2007). Most studies in this field assumed that web services and user requests are described using the same ontology language, which, however, is impractical. Junghans et al. (2010) focused on the semantic web service discovery scenario where web services and user requests are with different semantic descriptions. Sangers et al. (2013) proposed a method that enables users using natural language-based keywords to find semantic web services.

Although the above semantic matching methods normally perform better than syntactic ones in the accuracy of web service discovery or recommendation, their practicality is limited, for real web services are rarely described semantically. Lin et al. (2018) propose a neural-based method named NL2API, which also relies solely on service descriptions for recommending services. Different from the above methods, it uses a neural network model to generate embeddings for web services and Mashups based on their descriptions, and recommends services by computing their similarities based on the representations. Shi et al. (2019) also employ a neural work model to generate service and Mashup embeddings for service recommendation. To obtain richer semantics, they use the Attention mechanism and take into account both description and tags of services and Mashups.

3.2 CF-based methods

CF is one of the most popular recommendation techniques, and has also been widely applied to web service recommendation or service quality prediction (Zheng et al., 2020; Chen et al., 2019). Different from the content-based methods which focus on matching the contents of a user and a service, CF-based methods exploit in whole or in part the historical invocation records of other users and services for service recommendation. Among all CF-based web service recommendation methods, matrix factorisation (MF), factorisation machine (FM) and link prediction (LP) are the most widely used techniques. Therefore, this section focuses on the CF-based web service recommendation methods based on the three prevailing techniques, and discusses their features and differences.

3.2.1 Matrix factorisation

MF can be used to discover the latent factors from the service-Mashup matrix and to map all services and Mashups into the same space constructed by the latent factors. Consider a service-Mashup matrix R with invocation records by m Mashups for n services. The matrix R with m rows and n columns can be decomposed into two thin matrices M' and S'. M' will have $m \times 1$ dimensions and S' will have $n \times 1$ dimensions where 1 is the number of latent factors. The matrix R can be decomposed in such a way that the dot product of the matrix M' and transposed S' will yield a matrix with $m \times n$ dimensions that closely approximates the original ratings matrix, i.e., $R \approx M' \bullet S'$. Therefore, the key to the MF-based web service recommendation is finding the optimal M' and S', so that the results of their dot product can be used to accurately predict the missing values in R, which determines whether a service should be recommended to the new Mashup or not.

A number of MF-based web service recommendation methods have been proposed for Mashup creation. Xu et al. (2013) used a combination of MFs to predict recommendation scores of each service for a new Mashup, in which the Mashup-creator matrix, the Mashup-topic matric, the service-tag matrix and the service-topic matrix are incorporated. Yao et al. (2018) divided the correlation of services as an additional regularisation term into the probability matrix factorisation (PMF) to support more accurate service recommendation. Fletcher (2019a) introduced the user's implicit preferences into MF to enrich the web service invocation record so as to improve the accuracy and diversity of recommendations.

There are some studies that integrate other factors with MF to improve the prediction of service recommendation scores. In Liu and Fulia (2015), Jain et al. (2015) and Samanta and Liu (2017), the popularity of services was integrated with MF to rank the services in the recommendation list. In Tang et al. (2016), Liu et al. (2016) and Botangen et al. (2019, 2020), the geographic correlation and functional correlation were integrated with MF infer the priority of implicit Mashup-service invocations, so as to improve the accuracy of service recommendations. In Fletcher (2019b), the service quality indicators such as reliability and usability were integrated with MF to recommend top-k services.

3.2.2 Factorisation machine

FMs are generic supervised learning models that map arbitrary real-valued features into a low-dimensional latent factor space and can be applied naturally to a wide variety of prediction tasks including regression, classification, and ranking (Rendle, 2010). More specifically, the prediction task for a FM model is to estimate a function $\hat{y}$ from a feature set x to a target domain. The advantages of this model majorly lie in the way it uses a factorised parametrisation to capture the pairwise feature interactions. It can be represented mathematically as follows:

$$\hat{y} = w_0 + \sum_i w_i x_i + \sum_i \sum_{j>i} < v_i, v_j > x_i x_j$$

The three terms in this equation correspond respectively to the three components of the model:

- w_0 represents the global bias.
- w_i models the strength of the i^{th} variable.
- $<v_i, v_j>$ models the pairwise interaction between the i^{th} and j^{th} variable.

Several studies have employed FMs to recommend services for Mashup creation. Li et al. (2017) proposed a FM-based service recommendation method which exploits multi-dimensional data such as tags, topics, service co-occurrences and popularity factors. The tags and topics of Mashup and web services were employed to calculate the similarity between web services and the similarity between Mashups. Finally, similar Mashups, similar web services, co-occurrence and popular factors of web services are combined by FMs to predict the recommendation scores of web services. Cao et al. (2019a) improved (Cao et al., 2013) by using the relational topic model (RTM) to extract more accurate topic vectors of Mashups and services, and based on which to compute their similarities. To improve the representations of Mashups and services, Zhang et al. (2018) employed word2Vec to extract the functional information of Mashups and services, and use a FM-based neural network to integrate various features to compute the recommendation score services. The aforementioned work assumes that all features are with the same or equal importance. However, different features may have different levels of impacts. Some unnecessary features may even raise negative impacts in the recommendation performance for their data noise. Cao et al. (2019b) introduced an attention mechanism to distinguish the importance of each factor in the training data, so as to improve the recommendation performance. Tang et al. (2021) proposed a deep FM-based method and took into consideration the composition patterns of services to recommend suitable services for Mashup creation.

3.2.3 Link prediction

This class of service recommendation methods model the relationships between Mashups, web services and other related objects as a graph, and recommend services by predicting new links. Based on the graph structure, previous LP-based service recommendation methods can further be divided into homogeneous graph-based and heterogeneous graph-based. The former class of methods considers homogeneous service graphs consisting of only web services and their relationships. The latter class of methods considers heterogeneous service graphs which consist of different types of nodes such as web services, Mashups/apps, tags, and service providers.

Zhao et al. (2010) and Mokarizadeh et al. (2012) focused on mining the dependency relationship between the input/output data of two web services, and proposed service navigation and recommendation methods via the links generated. Tang et al. (2019) mined both the cooperation and similarity relationships between web services. The cooperation relationship between two web services was inferred by identifying whether they have been co-invoked by one or more Mashups. Then, according to a series of heuristic rules based on the cooperation and similarity relationships, potentially cooperative web services were recommended for a given service via LP. Huang et al. (2014) built a service-service homogeneous network based on the concurrence of web services in Mashups, and proposed a network model to mimic the evolution of the service network, through which potential service compositions are predicted.

The service-Mashup ecosystem (including services, Mashups, developers and providers) can be viewed as a heterogeneous information network (HIN) (Lian and Tang, 2021). A number of studies have explored the heterogeneous service networks to recommend services for Mashup creation. Liang et al. (2016) took into account the relationships between services, Mashups, tags, and service providers to build a heterogeneous service network, and calculated the similarities between different objects based on meta-paths in the network. Then, they predicted the possible links between a new Mashup and all services using a CF technique. Xie et al. (2019) proposed a Bayesian personalised ranking algorithm by sampling meta-paths in a heterogeneous service network and base on Mashup group preference to recommend services. Wang et al. (2019b) proposed a link-based service recommendation method based on the knowledge graph constructed by extracting the various relationships between services, Mashups, tags and service providers. They employed a random walk method with restart to estimate the relevance of API candidates for a Mashup.

3.3 Hybrid methods

Hybrid service recommendation methods combines both contents and relationships to recommend services according to the requirement description of the new Mashup development, which is typically expressed using keywords or natural language sentences.

Some studies use keywords to describe the users' functional requirements. He et al. (2017) proposed a service search/recommendation method, i.e., keyword search for service-based systems (KS3), which searches web API by entering keywords. They also assumed that the functionalities of services are also represented by keywords, and a service graph was built based on the invocation relationships between services. The goal is to find a subset of nodes (services) of the service graph, so that the keywords of the services will cover those keywords entered. In a similar manner, Qi et al. (2019) used a set of keywords to describe API functions and the user's request, and built a service graph based on the concurrence relationships between services. Then, they transformed the service search/recommendation problem into an optimal Steiner tree finding problem, and developed a dynamic programming (DP)-based method to solve the problem.

The keyword-based methods may have limitations such as poor recommendation performance and heavy dependence on correct keywords from users. In practice, the Mashup development requirements are likely to be specified by users with natural language-based sentences. It would be desirable for a system to recommend web services to users with natural language-based requests. Yao et al. (2014) proposed a service recommendation method that unifies CF and content-based recommendations. It considered simultaneously both rating data (e.g., QoS) and semantic content data (e.g., functionalities) of web services using a probabilistic generative model. Cao et al. (2016) proposed an integrated content and network-based service clustering and web service recommendation method for Mashup development. The method used a two-level topic model to mine the latent topics from the description of Mashups and the user's request, and mapped the user's request to similar Mashups. Then, it adopted a CF method which exploits the concurrence of web services in Mashups to recommend web APIs for the requirements of the new Mashup.

To further improve the effect of combining contents and relationships for service recommendation, some recent work proposed neural-based service recommendation methods by using the techniques such as convolutional neural networks (CNN), recurrent neural networks (RNN) and hierarchical attention networks (HAN). Xue et al. (2017) employed CNN and long short-term memory (LSTM) to integrate both contents and relationships of services to obtain better service classification effects for service discovery/ recommendation. Xiong et al. (2018) converted the natural language-based description of services and Mashup developments into semantic relationship triples (X, α, Y), and employ deep learning to learn semantic representation of relationships for service recommendation. Bai et al. (2020) used stacked denoising autoencoders (SDAE) to train services description text and exploit the service usage records as a regularisation of the encoding output of SDAE, to enhance the robustness of embeddings. Ma et al. (2020) proposed a multiplex interaction-based service recommendation approach named MISR, which incorporates three types of interactions between services and mashups into a deep neural network.

4 Challenges and future research possibilities

Although the above-mentioned works are valuable in addressing the service recommendation problem, they still have limitations in terms of recommendation performance. Besides, the service recommendation problem also faces some new challenges. Below, we present some challenges and future research possibilities.

Web service recommendation algorithms with higher accuracy. Although recent methods have improved the recommendation performance to some extent, there are still much room for further improvement. Emerging deep learning and neural model-based technologies are continuously introduced to various recommender systems, and have demonstrated their significant advantages over the other methods in many cases. For example, graph neural network (GNN), as a novel technology, has recently been applied to recommendation systems (Wang et al., 2019a) and achieved high performance. It can effectively solve the data sparseness and cold start problems, and is good at building a more robust and reliable recommendation system. In addition to the interactions between users and items, GNN can effectively use and integrate the relevant information from multiple sources that has explicit or implicit influence on the recommendation. How to design algorithms based on GNN or other neural models to improve web service recommendation for Mashup creation is a possible research direction.

Industrial applications of web service recommendation. web service recommendation is playing an increasingly important role in the development of service-based systems like Mashups. However, there is still a lack of research on the industrial implementation of web service recommendations. On the one hand, although there could be millions of services providing various functions on the web, most of them are not well maintained and the sustainability of those services cannot be guaranteed. Therefore, recommending with such a large-scale, highly unreliable services to meet the needs of developers is a difficult task. On the other hand, due to the privacy concerns, industrial companies (including both service providers and consumers) may be reluctant to disclose information about service usage. However, abundant service usage records are very important for service recommendations. Studies of industrial applications of web service recommendation may provide a promising direction which requires urgent attention.

Automated web service composition for Mashup creation. Web service composition is undoubtedly the most promising way to integrate business-to-business applications. However, the current solutions from industry and the academia often focus on either SOAP-based web services or semantic web services. As a lightweight and cost-effective alternative to SOAP-based services, RESTful services recently have shown their potential to compose reliable and visible web-scale applications based on the so-called Mashups. Previous studies on RESTful service composition are mostly manual or semi-automated. The manual composition requires the user manually complete the whole service composition process from service discovery to service binding, which is impractical when the service number is very large. As a contrast, the semi-automated composition techniques make suggestions to the user for service selection during the composition process. Nevertheless, the user still needs to select services and bind them in an appropriate order. At present, fully automated composition for RESTful services is still a challenging issue, for the RESTful services suffer from shortcomings on semantically describing, finding and composing services as well as the absence of a holistic framework covering the entire service lifecycle. Therefore, how to automate RESTful service composition is also a promising research direction.

5 Conclusions

Web service recommendation for Mashup creation has become a hot research topic in the service computing field. In this survey, we conducted a comprehensive review of the state-of-the-art methods in this topic, which are divided into three categories: content-based, CF-based and hybrid methods. The content-based methods primarily rely on the description text of services and user requirements to make service recommendations. The CF-based methods exploit the relationships between services and Mashups to make service recommendations. And the hybrid methods combine contents and relationships of services to make service recommendations. We also summarised the challenges faced by web service recommendation and suggested some future research directions.

References

Bai, B., Fan, Y., Tan, W. and Zhang, J. (2020) 'DLTSR: a deep learning framework for recommendations of long-tail web services', *IEEE Transactions on Services Computing*, Vol. 13, No. 1, pp.73–85.

Bosch, J. (2009) *From Software Product Lines to Software Ecosystems*, SPLC, pp.111–119.

Botangen, K.A., Yu, J., Sheng, Q.Z., Han, Y. and Yongchareon, S. (2020) 'Geographic-aware collaborative filtering for web service recommendation', *Expert Systems with Applications*, Vol. 151, No. 113347.

Botangen, K.A., Yu, J., Yongchareon, S., Yang, L. and Sheng, Q.Z. (2019) 'Integrating geographical and functional relevance to implicit data for web service recommendation', *International Conference on Service-Oriented Computing*, Springer, pp.53–57.

Bouguettaya, A., Sheng, Q.Z. and Daniel, F. (2014) *Web Services Foundations*, Springer.

Cao, B., Liu, J., Tang, M., Zheng, Z. and Wang, G. (2013) 'Mashup service recommendation based on user interest and social network', *2013 IEEE 20th International Conference on Web Services*, IEEE, pp.99–106.

Cao, B., Liu, X., Li, B., Liu, J., Tang, M., Zhang, T. and Shi, M. (2016) 'Mashup service clustering based on an integration of service content and network via exploiting a two-level topic model', *2016 IEEE International Conference on Web Services (ICWS)*.

Cao, B., Liu, J., Wen, Y., Li, H., Xiao, Q. and Chen, J. (2019a) 'QoS-aware service recommendation based on relational topic model and factorization machines for IoT Mashup applications', *Journal of Parallel and Distributed Computing*, Vol. 132, No. 2019, pp.177–189.

Cao, Y., Liu, J., Shi, M., Cao, B., Chen, T. and Wen, Y. (2019b) 'Service recommendation based on attentional factorization machine', *2019 IEEE International Conference on Services Computing (SCC)*.

Cappiello, C., Daniel, F., Matera, M. and Pautasso, C. (2010) 'Information quality in mashups', *IEEE Internet Computing*, Vol. 14, No. 4, pp.14–22.

Chen, Z., Shen, L., You, D., Ma, C. and Li, F. (2019) 'A location-aware matrix factorisation approach for collaborative web service QoS prediction', *International Journal of Computational Science and Engineering*, Vol. 19, No. 3, pp.354–367.

Fletcher, K. (2019a) 'Regularizing matrix factorization with implicit user preference embeddings for web API recommendation', *2019 IEEE International Conference on Services Computing (SCC)*.

Fletcher, K.K. (2019b) 'A quality-aware web API recommender system for mashup development', *Services Computing – SCC 2019*.

Halevy, A., Nemes, E., Dong, X., Madhavan, J. and Zhang, J. (2004) 'Similarity search for web services', *Proceedings of the 30th VLDB Conference*, pp.372–383.

He, Q., Zhou, R., Zhang, X., Wang, Y., Ye, D., Chen, F., Grundy, J.C. and Yang, Y. (2017) 'Keyword search for building service-based systems', *IEEE Transactions on Software Engineering*, Vol. 43, No. 7, pp.658–674.

Hsieh, M.Y., Lin, H.Y. and Li, K.C. (2011) 'A web-based travel system using mashup in the RESTful design', *International Journal of Computational Science & Engineering*, Vol. 6, No. 3, pp.185–191.

Huang, K., Fan, Y. and Tan, W. (2014) 'Recommendation in an evolving service ecosystem based on network prediction', *IEEE Transactions on Automation Science and Engineering*, Vol. 11, No. 3, pp.906–920.

Im, J., Kim, S. and Kim, D. (2013) 'IoT mashup as a service: cloud-based mashup service for the internet of things', *2013 IEEE International Conference on Services Computing*, IEEE, pp.462–469.

Jain, A., Liu, X. and Yu, Q. (2015) 'Aggregating functionality, use history, and popularity of APIs to recommend mashup creation', *International Conference on Service-Oriented Computing*, Springer, pp.188–202.

Jiang, W., Lee, D. and Hu, S. (2012) 'Large-scale longitudinal analysis of soap-based and restful web services', *2012 IEEE 19th International Conference on Web Services*, IEEE, pp.218–225.

Junghans, M., Agarwal, S. and Studer, R. (2010) 'Towards practical semantic web service discovery', *Extended Semantic Web Conference*, Springer, pp.15–29.

Kopecký, J., Vitvar, T., Bournez, C. and Farrell, J. (2007) 'Sawsdl: Semantic annotations for wsdl and xml schema', *IEEE Internet Computing*, Vol. 11, pp.60–67.

Lanthaler, M. and Gütl, C. (2010) 'Towards a RESTful service ecosystem', *4th IEEE International Conference on Digital Ecosystems and Technologies*, IEEE, pp.209–214.

Li, H., Liu, J., Cao, B., Tang, M., Liu, X. and Li, B. (2017) 'Integrating tag, topic, co-occurrence, and popularity to recommend web APIs for Mashup creation', *2017 IEEE International Conference on Services Computing (SCC)*.

Lian, S. and Tang, M. (2021) 'API recommendation for Mashup creation based on neural graph collaborative filtering', *Connection Science*, pp.1–15, DOI:10.1080/09540091.2021.1974819.

Liang, T., Chen, L., Wu, J., Dong, H. and Bouguettaya, A. (2016) 'Meta-path based service recommendation in heterogeneous information networks', *International Conference on Service-Oriented Computing*, Springer, pp.371–386.

Lin, C., Kalia, A., Xiao, J., Vukovic, M. and Anerousis, N. (2018) 'Nl2api: a framework for bootstrapping service recommendation using natural language queries', *2018 IEEE International Conference on Web Services (ICWS)*, IEEE, pp.235–242.

Liu, F., Shi, Y., Yu, J., Wang, T. and Wu, J. (2010) 'Measuring similarity of web services based on WSDL', *2010 IEEE International Conference on Web Services*.

Liu, J., Tang, M., Zheng, Z., Liu, X. and Lyu, S. (2016) 'Location-aware and personalized collaborative filtering for web service recommendation', *IEEE Transactions on Services Computing*, Vol. 9, No. 5, pp.686–699.

Liu, X. and Fulia, I. (2015) 'Incorporating user, topic, and service related latent factors into web service recommendation', *2015 IEEE International Conference on Web Services*.

Ma, Y., Geng, X. and Wang, J. (2020) 'A deep neural network with multiplex interactions for cold-start service recommendation', *IEEE Transactions on Engineering Management*, Vol. 68, No. 1, pp.1–15.

Martin, D., Burstein, M., Mcdermott, D., Mcilraith, S., Paolucci, M., Sycara, K., Mcguinness, D. L., Sirin, E. and Srinivasan, N. (2007) 'Bringing semantics to web services with OWL-S', *World Wide Web*, Vol. 10, pp.243–277.

Mikkonen, T. and Taivalsaari, A. (2010) 'The mashware challenge: bridging the gap between web development and software engineering', *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research*, pp.245–250.

Mokarizadeh, S., Küngas, P., Matskin, M., Crasso, M., Campo, M. and Zunino, A. (2012) 'Information diffusion in web services networks', *2012 IEEE 19th International Conference on Web Services*, IEEE, pp.488–495.

Papazoglou, M.P., Traverso, P., Dustdar, S. and Leymann, F. (2007) 'Service-oriented computing: state of the art and research challenges', *Computer*, Vol. 40, No. 11, pp.38–45.

Qi, L., He, Q., Chen, F., Dou, W., Wan, S., Zhang, X. and Xu, X. (2019) 'Finding all you need: web APIs recommendation in web of things through keywords search', *IEEE Transactions on Computational Social Systems*, Vol. 6, No. 5, pp.1063–1072.

Rendle, S. (2010) 'Factorization machines', *2010 IEEE International Conference on Data Mining*, IEEE, pp.995–1000.

Salminen, A. and Mikkonen, T. (2012) 'Mashups-software ecosystems for the web era', *IWSECO@ ICSOB*, pp.18–32.

Samanta, P. and Liu, X. (2017) 'Recommending services for new mashups through service factors and top-K neighbors', *2017 IEEE International Conference on Web Services (ICWS)*.

Sangers, J., Frasincar, F., Hogenboom, F. and Chepegin, V. (2013) 'Semantic web service discovery using natural language processing techniques', *Expert Systems with Applications*, Vol. 40, No. 11, pp.4660–4671.

Shi, M., Tang, Y. and Liu, J. (2019) 'Functional and contextual attention-based LSTM for service recommendation in Mashup creation', *IEEE Transactions on Parallel and Distributed Systems*, Vol. 30, No. 5, pp.1077–1090.

Taivalsaari, A. (2009) *Mashware: The Future of Web Applications*, Sun Microsystems, Inc.

Tan, W., Fan, Y., Ghoneim, A., Hossain, M.A. and Dustdar, S. (2016) 'From the service-oriented architecture to the web API economy', *IEEE Internet Computing*, Vol. 20, No. 4, pp.64–68.

Tang, B., Tang, M., Xia, Y. and Hsieh, M-Y. (2021) 'Composition pattern-aware web service recommendation based on depth factorisation machine', *Connection Science*, pp.1–21, DOI: 10.1109/TSC.2020.2995571.

Tang, M., Xie, F., Liang, W., Xia, Y. and Li, K-C. (2019) 'Predicting new composition relations between web services via link analysis', *International Journal of Computational Science and Engineering*, Vol. 20, No. 1, pp.88–101.

Tang, M., Zheng, Z., Chen, L., Liu, J., Cao, B. and You, Z. (2014) 'A trust-aware search engine for complex service computing', *International Journal of Web Services Research (IJWSR)*, Vol. 11, No. 1, pp.57–75.

Tang, M., Zheng, Z., Kang, G., Liu, J., Yang, Y. and Zhang, T. (2016) 'Collaborative web service quality prediction via exploiting matrix factorization and network map', *IEEE Transactions on Network and Service Management*, Vol. 13, No. 1, pp.126–137.

Wang, X., He, X., Wang, M., Feng, F. and Chua, T-S. (2019a) 'Neural graph collaborative filtering', *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Wang, X., Wu, H. and Hsu, C-H. (2019b) 'Mashup-oriented API recommendation via random walk on knowledge graph', *IEEE Access,* Vol. 7, No. 2018, pp.7651–7662.

Wu, C. (2012) 'WSDL term tokenization methods for IR-style web services discovery', *Science of Computer Programming*, Vol. 77, No. 3, pp.355–374.

Xie, F., Chen, L., Lin, D., Zheng, Z. and Lin, X. (2019) 'Personalized service recommendation with mashup group preference in heterogeneous information network', *IEEE Access*, Vol. 7, No. 2019, pp.16155–16167.

Xiong, W., Lu, Z., Li, B., Hang, B. and Wu, Z. (2018) 'Automating smart recommendation from natural language API descriptions via representation learning', *Future Generation Computer Systems*, Vol. 87, No. 2018, pp.382–391.

Xu, W., Cao, J., Hu, L., Wang, J. and Li, M. (2013) 'A social-aware service recommendation approach for mashup creation', *2013 IEEE 20th International Conference on Web Services*.

Xue, Q., Liu, L., Chen, W. and Chuah, M.C. (2017) 'Automatic Generation and Recommendation for API Mashups', *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*.

Yang, X., Xu, H., Shu, H., Wang, Y., Liu, K. and Ho, Y. (2021) 'Service component recommendation based on LSTM', *International Journal of Embedded Systems*, Vol. 14, No. 2, pp.201–209.

Yao, L., Sheng, Q.Z., Ngu, A.H., Yu, J. and Segev, A. (2014) 'Unified collaborative and content-based web service recommendation', *IEEE Transactions on Services Computing*, Vol. 8, No. 3, pp.453–466.

Yao, L., Wang, X., Sheng, Q.Z., Benatallah, B. and Huang, C. (2018) 'Mashup recommendation by regularizing matrix factorization with API co-invocations', *IEEE Transactions on Services Computing*, Vol. 14, No. 2, pp.502–515.

Yu, Q., Liu, X., Bouguettaya, A. and Medjahed, B. (2008) 'Deploying and managing web services: issues, solutions, and directions', *The VLDB Journal*, Vol. 17, No. 3, pp.537–572.

Zhang, X., Liu, J., Cao, B., Xiao, Q. and Wen, Y. (2018) 'Web service recommendation via combining Doc2Vec-based functionality clustering and deep FM-based score prediction', *2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom /SustainCom)*.

Zhao, C., Zhang, J., Zhang, J., Yi, L. and Mao, X. (2010) 'Hyperservice: linking and exploring services on the web', *2010 IEEE International Conference on Web Services*, IEEE, pp.17–24.

Zheng, Z., Xiaoli, L., Tang, M., Xie, F. and Lyu, M.R. (2020) 'Web service QoS prediction via collaborative filtering: a survey', *IEEE Transactions on Services Computing*, DOI is 10.1109/TSC.2020.2995571