融合 SOM 功能聚类与 DeepFM 质量预测的 API 服务推荐方法

曹步清 肖巧翔 张祥平 刘建勋

(湖南科技大学计算机科学与工程学院 湖南 湘潭 411201)

摘 要 由于越来越多的企业和组织纷纷将自己的业务、数据或资源封装成服务,并通过 API 的形式发布到互联网上,API 服务的数量呈现倍增趋势.在此背景下,如何从这样一个大规模的 API 服务集合中,快速有效地找到满足开发者用户 Mashup 需求的 API 服务,已成为一个挑战性问题.为此,本文聚焦于"推荐合适的 API 服务以构建高质量 Mashup 应用"问题,以面向服务内容的功能聚类为基础,结合基于多维服务质量的评分预测,提出一种融合 SOM 功能聚类与 DeepFM 质量预测的 API 服务推荐方法,用于创建高质量的 Mashup 应用.该方法首先采用 Wikipedia 作为外部语料库扩充 API 服务文档的内容并利用 HDP 模型建模其主题分布.通过 WikiExtractor 抽取出 Wikipedia 中的语料数据,并利用 Word2vec 工具训练该语料数据获得其词向量模型.利用训练好的 Wikipedia 词向量模型对 API 服务描述文档进行扩充.针对扩充后的 API 服务文档,使用 HDP 主题建模技术,挖掘出其隐含的主题信息,自动确定最优主题个数,以准确地度量 API 服务文档之间的语义相似度.然后,采用 SOM 神经网络进行面向主题的 API 服务聚类.在 HDP 主题建模之后,对获得的"API 服务文档-主题"向量采用 SOM 神经网络聚类算法进行主题聚类,通过自组织过程,将众多的 API 服务划分到不同的功能类簇中,每一个功能类中包含多个具有相似功能的 API 服务.接下来,针对 API 服务类簇中所有具有相似功能的 API 服务,利用 DeepFM 模型建模和挖掘其多维 QoS 属性之间的复杂交互关系,预测并排序 API 服务的质量得分.DeepFM 模型自动地提取出 QoS 数据中(包括流行度、共现次数等)的有效的特征组合关系(包括高阶特征和低阶特征组合关系),预测并排序每一个 API 服务相对于目标 Mashup 应用的质量得分,推荐得分靠前的 N 个 API 服务给开发者用户.最后,在真实 Web 服务数据集上进行了实验比较与分析,实验结果表明:本文方法在准确率、召回率、纯度、熵、DCG、HMD 等性能方面都要整体优于其它六种方法.相比于 TF-IDF、LDA-K-CF、LDA-K-FM、HDP-K-CF、HDP-K-FM、HDP-S-FM,本文方法的准确率指标分别提升了 196.2%、49%、33.8%、31.2%、12.3%、10.3%,DCG 值分别提升了 161.8%、26.4%、18.6%、16.2%、6.73%、4.5%.

关键词 API 推荐;Mashup 应用;HDP 主题模型;SOM 神经网络;深度因子分解机

中图法分类号 TP301 **DOI 号** 10.11897/SP.J.1016.2019.01367

An API Service Recommendation Method via Combining Self-Organization Map-Based Functionality Clustering and Deep Factorization Machine-Based Quality Prediction

CAO Bu-Qing XIAO Qiao-Xiang ZHANG Xiang-Ping LIU Jian-Xun

(*School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan, Hunan* 411201)

Abstract More and more enterprises and organizations encapsulate their business, data or resources as API services and publish then on Internet, and the number of API services is growing fast. In this context, to find API services quickly and effectively that meets Mashup requirements

收稿日期:2018-05-22;在线出版日期:2019-03-08.本课题得到国家自然科学基金(61873316,61872139,61772193,61702181)、湖南省自然科学基金(2017JJ2098,2017JJ4036,2018JJ2139,2018JJ2136)资助.**曹步清**,博士,副教授,主要研究方向为面向服务的软件工程、服务计算与云计算.E-mail:buqingcao@gmail.com.**肖巧翔**,硕士研究生,主要研究方向为服务聚类.**张祥平**,硕士研究生,主要研究方向为服务推荐.**刘建勋**,博士,教授,博士生导师,主要研究领域为服务计算与云计算、工作流管理的理论与应用、面向服务的软件工程.

of developers from such a large collection of API services, has become a challenging problem. To address this problem, aiming to the issue of recommending appropriate API services to build high-quality Mashup applications, on top of service content-oriented functionality clustering and score prediction of quality of service with multi-dimension, an API service recommendation method via combining self-organization map-based functionality clustering and deep factorization machine-based quality prediction, is proposed in this paper to create novel Mashup applications with high-quality. This method, firstly uses Wikipedia as an external corpus to expand the contents of API service documents and models their topic distribution by adopting HDP model. WikiExtractor is used to extract corpus data from Wikipedia, and Word2vec tool is exploited to train the corpus data to obtain its word vector model. The trained Wikipedia word vector is regarded as the extension source of API service documents. As for the extended API service documents, hierarchical Dirichlet processes topic modeling technology is deployed to mine their implicit topic information, which automatically identifies the optimal number of topics to accurately measure semantic similarity between API service documents. Then, it exploits SOM neural network to cluster API services into various clusters with similar topic and functionality. After HDP topic modeling, the derived vector of API service document-topic is clustered with different topics by using the clustering algorithm of self-organization map-based neural network. That is to say, numerous API services are divided into different clusters through self-organizing process, each of which contains multiple API services with similar functionality. Next, as for all API services with similar functionality in the API service cluster, we use deep factorization machine to model and mine the complex interaction relationships between their multi-dimensional attributes of quality of service, and predict and rank the quality scores of API services. Deep factorization machine model automatically extracts effective feature composition relationships (including high-order feature and low-order feature composition relationships) between the data of quality of service (such as popularity, co-occurrences, and so on), predicts and ranks the quality score of each API service to the target Mashup application, and recommends the top-N API services with the high score to the developer. Finally, based on a real-world dataset of Web services, we conduct a comprehensive evaluation to measure the performance of the proposed method. The experimental results show that the performance of the proposed method in this paper is totally better than that of other six methods in terms of precision, recall, purity, entropy, DCG (Discounted Cumulative Gain) and HMD (Hamming Distance). Compared with the methods of TF-IDF, LDA-K-CF, LDA-K-FM, HDP-K-CF, HDP-K-FM, HDP-S-FM, the precision metric of the proposed method has increased by 196.2%, 49%, 33.8%, 31.2%, 12.3%, 10.3%, and DCG metric has increased by 161.8%, 26.4%, 18.6%, 16.2%, 6.73% and 4.5%, respectively.

Keywords API recommendation; Mashup application; Hierarchical Dirichlet Processes topic model; Self-Organizing Map-based neural network; deep factorization machine

1 引 言

以云计算、物联网、移动互联网、大数据为代表的新一代信息技术与服务业的深度融合[1],正在驱动软件产业变革,并加速新一代信息服务技术的跨界融合和创新发展[2].现代企业和组织需要快速地应对这种变革,动态、高效地构建、重组和优化其业务流程,随需应变,以提高互联网信息服务的利用率,缩短应用开发周期和成本,提升自身竞争力.面向服务的计算(Service-Oriented Computing,SOC)、面向服务的体系结构(Service-Oriented Architecture,SOA)倡导以服务及其组合为基础构造应用,为解决分布式异构环境中的企业应用集成问题,克服信

息孤岛带来了诸多便利[3]. 近年来，互联网上出现了一种轻量级的信息服务组合模式——Mashup 技术，它可以将两种或两种以上 API 服务混搭在一起，创建一个全新的 Web 应用[4]. 软件开发者用户可通过 Mashup 方式混搭互联网上已有的 API 服务资源，构建满足用户复杂需求的组合级应用. 例如，软件开发者通过调用并混搭 FedEx、Baidu Map 和 WeChat 这三个 API 服务，可快速构建一个新的 Mashup 应用，实现“在百度地图上跟踪联邦快递的包裹运送情况，并通过微信及时推送消息给用户”的功能.

在服务计算“服务化”思想驱动下，越来越多的企业和组织纷纷将自己的业务、数据或资源封装成服务，并通过 API 的形式发布到互联网上，API 服务的数量呈现倍增趋势[5-6]，各类服务注册库以及服务门户网站中所包含的服务多达 825 132 个. 其中，从 2011 年 6 月至 2018 年 3 月，ProgrammableWeb 网站中的服务数量从 3261 个增加到 19 000 多个，增幅高达 500%. 在此背景下，要从这样一个大规模的服务集合中，快速有效地找到满足开发者用户 Mashup 需求的 API 服务，犹如大海捞针. 例如，当一个开发者用户想要开发一个与移动相关的 Mashup 应用时，在 ProgrammableWeb 网站上搜索获得的 API 服务高达 1850 个，要从这些具有相同功能的海量服务集合中选择最合适的 API 服务来完成移动 Mashup 应用开发，非常困难. 此外，我们注意到：搜索出来的少数 API 服务并不能满足开发者用户的需求.

针对以上问题，一些研究者利用服务推荐技术来提升服务发现的精准度，主要包括基于内容的服务推荐[7-8]、基于服务质量的服务推荐技术[9-13]以及混合的服务推荐方法[14-15]. 最近，少数研究者引入服务聚类技术到服务推荐中[4,16-20]. 尤其是在大规模服务场景下，将功能相似的服务划分到类簇中，缩小了服务搜索的空间，提高了服务发现的效率. 无疑，这些方法大大提升了服务推荐的效果，然而，它们大都存在以下两个方面的问题：

首先，服务描述文档通常比较简短、特征稀疏、信息量少. 目前大部分主题模型很难对缺乏训练语料的短文本进行很好的建模. 虽然一些主题模型在训练的过程中引入了辅助信息[16-17]，如标签信息[18-19]和词聚类信息[20]等，在一定程度上有利于提取到更加准确的 Web 服务隐含语义信息. 但是，相比传统 LDA 主题模型而言，现有改进的主题模型对服务聚类的准确度提升并不明显.

其次，以传统 QoS 数据为基础并加入地理位置、历史协作记录等质量信息，使用矩阵分解、因子分解机等技术用于服务推荐[9-13]，在一定程度上缓解了数据稀疏、冷启动等问题，提高了预测与推荐的精度及效率. 然而，其它服务质量属性需要重点关注，如流行度、共现次数、上下文情境等，更重要的是，这些质量属性并不是独立的，它们相互之间存在一定的关联关系. 比如：地理位置临近的服务的响应时间不会相差太大. 进一步地，需要挖掘并利用它们之间的关联关系以改善服务推荐效果.

为此，本文聚焦于“推荐合适的 API 服务以构建高质量 Mashup 应用”问题，以面向服务内容的功能聚类为基础，结合基于多维服务质量的评分预测，提出一种融合 SOM（Self-Organizing Map Neural Network，自组织神经网络[21-23]）功能聚类与 DeepFM（Deep Factorization Machines，深度因子分解机[24-25]）质量预测的 API 推荐方法，用于高质量 Mashup 的创建. 该方法首先使用维基百科语料库作为外部语料库，对 Web 服务描述文档进行扩充，并利用 HDP（Hierarchical Dirichlet Processes）模型[26]挖掘 Web 服务描述文本的隐含主题. 接下来，针对“服务文档-主题”向量，采用 SOM 算法进行聚类，将 Web 服务划分到不同的功能类簇中. 最后，针对功能类簇中的 Web 服务，引入多维 QoS 属性信息（包括流行度、共现次数等），利用 DeepFM 模型挖掘和建模复杂的特征组合关系，预测并排序 Web 服务的得分，为开发者用户推荐 top-k 高质量 Web 服务. 概括地说，本文工作主要有以下贡献：

（1）结合 HDP 主题模型与 SOM 神经网络算法，挖掘出服务文档的最优主题数，实现面向主题分布的服务功能聚类. 采用维基百科作为外部语料库扩充 Web 服务文档的内容，利用 HDP 模型挖掘 Web 服务文档的主题信息，自动确定最优主题数. 使用 SOM 算法执行面向主题的服务聚类，更加准确地度量 Web 服务文档之间的相似度，提高了服务聚类的精度.

（2）以服务功能聚类为基础，采用深度因子分解机模型实现服务的评分预测及排序. 通过 SOM 服务功能聚类，将 API 服务划分到不同的功能类簇中，减小了服务搜索空间与范围；针对功能类簇中相似 API 服务，利用 DeepFM 建模和训练它们的多维 QoS 属性，发现并同时利用它们之间的低阶与高阶关联关系，提高了服务推荐的精准度.

（3）使用真实的 Web 服务 QoS 数据集对提出

的方法进行了实验评估.结果验证了本文所提方法的有效性,而且表明了该方法具有较好的聚类精度与推荐效果.

本文第2节介绍Web服务聚类与推荐的相关工作;第3节详细阐述基于SOM聚类与DeepFM预测的API推荐方法;第4节描述实验方法、过程及结果;最后一节对全文进行总结与展望.

2 相关工作

2.1 基于功能聚类的服务推荐

研究表明:引入服务聚类技术,预先对Web服务进行聚类,将大大提高Web服务搜索能力,为从海量服务中找到所需服务提供了便捷[27].通过将用户请求定位到特定服务类簇,并从中选择具有相似功能的服务,可以有效地减小服务搜索空间与范围,切实提高Web服务发现的效率与精度.

目前,面向功能的服务聚类已有大量研究.其中,一些研究人员利用向量空间模型,将服务内容文档建模成由一系列关键词组成的向量,并通过欧氏距离、余弦距离、JS距离等相似度度量方法来计算服务之间相似度,实现服务功能聚类[28-29].例如,Liu等人[28]提出从WSDL文本中提取出内容、上下文、主机名和服务名这四个特征实现Web服务聚类.Elgazzar等人[29]从WSDL文档中抽取5个关键特征,然后基于这些特征将服务聚类为功能相似的类簇.考虑到服务功能描述文档内容的有限、单一,一些研究工作引入了服务的Tag信息,以提高服务聚类精度[30-32].其中,Wu等人[30]提出一种混合的Web服务标签推荐策略WSTRec,首先将标签共现、标签挖掘和语义相关性度量技术用于标签推荐,然后综合考虑WSDL文本和Tag标签,提出一种新颖的Web服务聚类方法WTCluster[31],并开发了Titan系统[32].然而,向量空间模型依据的是词频信息,取决于共有词汇的数量,存在向量维度过高、语义稀疏等问题.

为了缓解上述问题,一些研究人员开始利用主题模型(如PLSA、LDA等),将服务内容文档从高维词向量空间映射到低维主题向量空间中,提取隐含主题及语义,实现基于主题概率的服务功能聚类[33].其中,李征等人[34]提出了一种领域服务聚类模型DSCM,并基于该模型对服务进行面向主题的聚类,把特定领域内具有相似功能的服务组织为主题类簇.田刚等人[19]使用本体辅助的支持向量机和面向领域的服务特征降维技术,构建服务特征向量,并利用LDA主题模型对服务进行离线的面向主题的服务聚类.在我们的前期工作中[35],融合Mashup描述文档、Web APIs服务描述文档以及它们的标签作为聚类的数据源,提出一种基于LDA主题模型的Mashup聚类方法.

以上这些工作大都以服务描述文档作为内容信息源或加入标签进行扩充,并采用LDA主题模型实现面向主题的服务聚类,在一定程度上提高了服务聚类的精度,但其效果有待进一步提升.主要的原因在于:服务描述文档简短导致训练语料不够充分,尽管融入其它辅助信息,但是传统的主题模型(如LDA等)在训练生成描述文档的隐含主题向量分布前,需要预先指定主题的个数,而主题数的选择会极大地影响到服务聚类效果.一般情况下,为取得较好的服务聚类效果,需要不断地调整主题数以寻找其最佳值,而这个过程需要反复地训练主题模型.为解决该问题,本文使用HDP主题模型来建模Web服务的主题分布以获得最佳的服务聚类效果.HDP模型[26],是由Teh等人于2004年提出的一种基于LDA的非参数主题模型,可实现多文档之间共享无限多个聚类的效果.与LDA等传统参数主题模型相比,HDP模型使用灵活,在应用于文档聚类任务时,可根据语料库自动确定聚类数目(即最优的潜在主题数目)并自动生成聚类中心的分布.因此,我们利用HDP主题模型挖掘出Web服务文档的最优主题数,实现较好的Web服务聚类效果.

此外,在进行面向主题的服务聚类过程中,通常是将每个服务聚类到其所包含主题概率最大的主题类簇[34],或者进一步地采用特定的聚类算法(如K-means聚类)度量Web服务文档之间的相似度[18-19,35],得到功能相似的服务主题类簇.这些方法存在一定的问题:(1)将服务"硬"聚类到其包含主题概率最大的类簇,不一定合适.因为服务可能同时属于多个主题,而在多个主题分布上进行服务相似性度量及聚类更为准确;(2)经典的聚类算法,如K-means,需要事先确定类的个数,其初始点的选取会影响到聚类效果.SOM神经网络聚类算法[21-23],是一种无监督的神经网络学习算法,能够自动寻找训练样本的基本属性和内在规则,并通过自适应和自组织特征改变神经元结构和网络变量.不同于K-means,SOM神经网络聚类算法无需事先确定类的个数,可获得较好的聚类效果;除了最佳匹配权重向量之外,SOM会更新网络图上的拓扑邻居,使得

最佳匹配向量周围的区域被拉伸到所代表的训练样本.我们利用 SOM 神经网络实现面向主题的服务聚类,通过自组织方法调整网络之间的权重,用大量训练数据调整网络的权值,使得网络的输出能够反映样本数据的分布情况,实现面向主题分布的 Web 服务功能聚类.

2.2　利用服务质量的服务推荐

服务质量是实现高质量服务推荐的关键因素之一.目前,国内外研究者在基于 QoS 的服务推荐方面开展了广泛的研究工作.其中,一些研究者利用协同过滤技术,根据共同的服务调用记录选取目标用户的相似邻居,预测 QoS 缺失值,以推荐具有最优 QoS 值的服务给用户.例如,Zheng 等人[9]提出了一种混合的协同过滤方法,以预测和评估 Web 服务的 QoS 值.Wang 等人[11]提出了一种 Web 服务推荐中的多维 QoS 预测方法.进一步地,一些工作[36-37]引入了用户和服务的地理位置信息并构建地域模型来预测 QoS 缺失值,以提高 Web 服务推荐的精度.

为缓解服务推荐中的数据稀疏和冷启动问题,一些研究者利用概率矩阵分解技术,将用户-服务矩阵分解为低维的用户潜在特征空间矩阵和服务潜在特征空间矩阵,预测目标用户对服务的未知 QoS 评分,排序并选取高 QoS 值的服务推荐给目标用户.其中,Zheng 等人[10]使用用户的直接历史经验与其他相似用户的间接经验,提出一种基于矩阵分解的协同过滤服务选择模型,预测与评估 Web 服务的 QoS 值.最近,一些研究工作[4,38-39]聚焦于利用 Web APIs 之间在共同的 Mashup 中的协作(组合)历史,推荐一系列相关的高质量 Web APIs 完成服务混搭.其中,Gao 等人[38]使用 Mashup 描述、API 描述以及 Mashup 与 API 之间的历史调用关系,提出了一种面向 Mashup 的基于流行排序的 APIs 推荐方法,使得所推荐的 APIs 不仅在 Mashup 功能上具有相似性,而且自身是流行的,相互之间也是相似的.Yao 等人[39]重点调研了 Mashup 和 APIs 的历史调用关系,以推导出 APIs 之间的隐含协作关系并集成这种关系到矩阵分解模型中,用于改进 APIs 推荐的精度及质量.

上述这些工作,使用协同过滤、矩阵分解等技术用于服务推荐,在一定程度上提高了预测与推荐的精度与效率.然而,不管是协同过滤还是矩阵分解,都存在矩阵稀疏性的问题,并且它们对输入数据要求比较严格,不具有普遍性.Rendle 等人提出了一种因子分解机模型[13],非常适用于高度稀疏情况下的一般性预测与推荐任务,可以很好地解决这一问题.该模型可将任意数量、任意长度的特征向量作为输入,即使矩阵高度稀疏也不会影响到推荐精度.我们曾利用因子分解机模型,提出了一种融合多维信息的主题自适应 Web API 推荐方法[13],使用 Mashup 之间的相似度、Web API 之间的相似度、Web API 的流行度以及共现性作为因子分解机模型的输入,为开发者用户推荐高质量 Web APIs 创建 Mashup 应用.尽管因子分解机的优势明显,但当其特征输入维度增加时(如服务质量属性个数等),多维特征之间的高阶交互关系(三阶及以上)变得纷繁复杂,而因子分解机却无法建模和利用这些高阶交互关系,导致其预测与推荐的精准度降低.深度因子分解机[24-25],是 2017 年提出来的一种基于深度神经网络的因子分解机模型,同时集成了因子分解机和多层感知器的优势.它不仅使用因子分解机建模低阶(一或二阶)的特征交互,而且利用深度神经网络建模高阶(三阶及以上)的特征交互,通过挖掘多个特征之间的相互关系,能够大大提高预测与推荐的精准度.为此,我们利用它来建模和训练 API 服务的多维 QoS 属性,发现并同时利用它们之间的低阶与高阶关联关系,以提高服务推荐的精准度.

3　方　法

假设开发者用户 Bob 需要开发一个与出行相关的 Mashup 应用,图 1 展示了采用现有搜索方法以及本文推荐方法下的 Mashup 创建流程.如图 1 上半部分所示,当 Bob 在 ProgrammableWeb 在线服务平台提交开发需求("I want to develop a Mashup related to travel")时,平台采用现有的基于关键词的搜索方法,针对用户需求关键词"travel",在服务注册库中进行简单匹配后,搜索获得 618 个 API.经过分析后发现:(1)一些与出行并不相关的 API 出现在搜索结果中,例如 MotleyBunch、TauP、MailHops 等;(2)一些具有出行功能的 API 却未找到,如 Cleartrip Hotel、Travel Booking Network 等.也就是说,搜索结果并不精准.在这种情况下,要找到满足开发者用户 Mashup 需求的高质量 API 服务,成为一个难题.在图 1 下半部分,Web 服务搜索引擎采用本文所提方法,通过 Mashup 的方式为 Bob 推荐一组可行的服务组合方案.一般来说,用户出行需求包括多方面互补的功能,需要到不同服务功能类别中检索满足需求的 API 服务.本文方法将

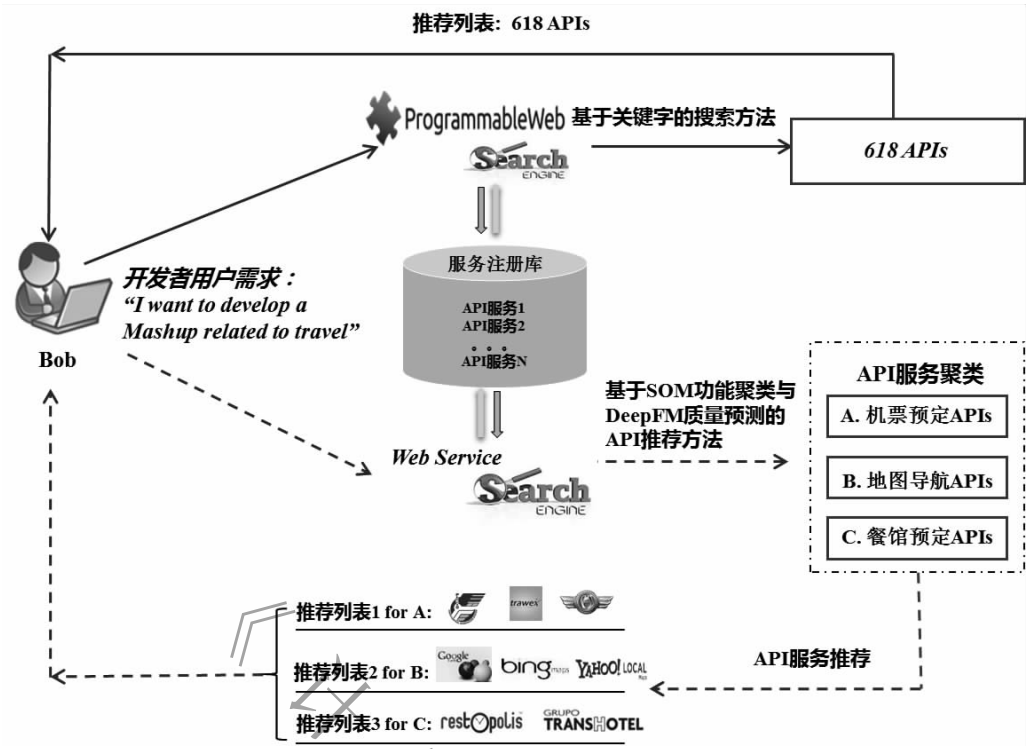

图 1 不同 Mashup 开发流程

服务组合过程分成多个步骤:(1)理解并分解用户的出行需求(如机票预定+地图导航+餐馆推荐);(2)利用 SOM 算法将 API 服务按照功能划分进行聚类(机票预定类、地图导航类、餐馆预定类);(3)针对具体的服务类别,基于 DeepFM 预测并排序 API 服务,产生服务推荐列表,供用户进行选择并完成 Mashup 的创建.

具体来说,本文所提方法的总体框架如图 2 所示,主要包括以下几个功能模块:

(1)服务文本扩充. 使用维基百科语料库作为外部语料库,对 Web 服务描述文档进行扩充. 通过 WikiExtractor 抽取出维基百科中的语料数据,并利用 Word2vec 工具训练该语料数据获得词向量模型. 使用已经训练好的词向量模型对 Web 服务的描述文档进行扩充.

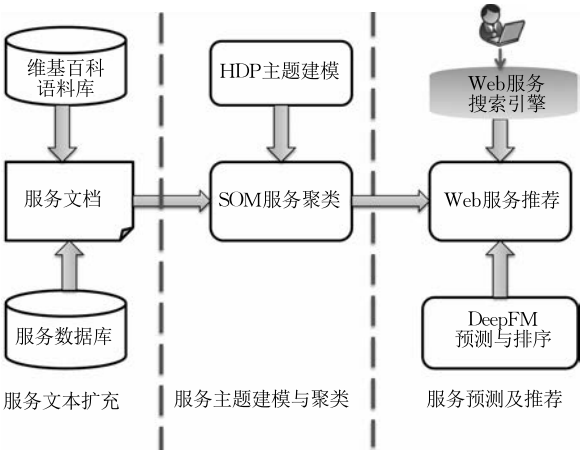

图 2 方法总体框架

(2)服务主题建模. 针对扩充后的服务文档,使用 HDP 主题建模技术,获取服务文本的隐含主题. HDP 主题模型能够自动确定最优的主题个数,也能够更加准确地度量 Web 服务描述文档之间的语义相似度.

(3)基于 SOM 的服务聚类. 在 HDP 主题建模之后,对获得的"服务文档-主题"向量采用 SOM 算法进行主题聚类,通过自组织方法,将不同的 Web APIs 划分在不同的功能类簇中.

(4)基于 DeepFM 的服务预测与排序. 针对服务类中所有具有相似功能的 Web APIs 服务,考虑服务之间的相似程度以及 QoS 属性(包括流行度、共现次数等)等多维特征信息,利用 DeepFM 建模和训练这些多维特征信息,挖掘复杂的特征组合关系,预测并排序 Web APIs 的得分.

(5)Web 服务推荐. 针对开发者用户的 Mashup 需求,服务推荐分成两个步骤来进行. 首先,计算 Mashup 需求描述与服务类簇描述(该类中所有服务描述文本的集合)之间的主题相似度,找到最匹配的服务类. 其次,针对该服务类中所有具有相似功能的 Web APIs 服务,根据它们的预测得分,对候选

服务进行排序，为开发者用户推荐 top-k 高质量的 Web APIs.

下面详细介绍方法中各个功能模块所包含的模型、算法以及实现过程.

3.1　服务文本扩充

Web 服务描述文本通常是短文本，词汇共现信息少，如果直接针对 Web 服务描述文本进行主题建模的话，难以充分挖掘出隐含的精确的主题信息. 为此，我们对 Web 服务描述文本进行扩充. 首先，使用 Word2vec 工具对 Wikipedia 语料库进行训练，获得其词向量模型. 然后，针对 Web 服务描述文本中的某个词，在训练好的词向量模型中找出其前 N 个相似词. 最后，将这些相似词合并到 Web 服务描述文本中予以扩充. 具体过程为：将 Web 服务描述文档 S_{text} 输入到 Wikipedia 词向量模型中，构建与 S_{text} 中每个词 w_i 最相似的词集合 $T_{w_i}=(t_1,t_2,\cdots,t_M)$，并选出相似度较高的前 N 个词对 S_{text} 进行扩充，$M\geq N$. 扩充后的 Web 服务描述文档记为 $S_{enrichedText-N}$，则 $S_{enrichedText-N}=\{(w_1,T_{w_1}),(w_2,T_{w_2}),\cdots,(w_n,T_{w_n})\}$. 其中 n 为单词总个数. 表 1 是 Earth 和 Google 两个词分别扩充 10 个词的实例，按照与原始单词之间的相似度大小从上到下进行排序.

表 1　服务文本扩充的例子

原始词	Earth	Google
扩充词	Planet	gmail
	martian	dropbox
	mars	evernote
	venusian	app
	planets	adsense
	spaceship	yahoo
	universe	microsoft
	planetary	flickr
	moon	hotmail
	deimos	mapquest

使用 Word2vec 工具对 Wikipedia 语料库进行训练，得到 Wikipedia 语料库的词向量模型. 由于 Word2vec 中使用了 Hierarchical Softmax 算法进行加速，所以在训练词向量模型阶段，其时间复杂度为 $O(\log N)$，实际中使用 Word2vec 工具对 Wikipedia 语料库进行训练，总计用时约为 12 h. Word2vec 工具会计算出语料库中所有词汇的相似度矩阵，计算相似度矩阵的时间复杂度为 $O(n^2)$，且空间复杂度也为 $O(n^2)$. 在对服务文本进行扩充的过程中，只需要在训练好的模型中通过查表的方式，即可获得某一个单词的最相似的前 n 个相似词. 因此，在服务文本扩充阶段的时间复杂度为 $O(1)$，但空间复杂度为 $O(n^2)$.

3.2　服务主题建模

主题建模是一种文本信息特征抽取技术，它能够将文本中潜在的隐含主题信息挖掘出来. 隐含主题分布模型(LDA)是一种典型的主题建模技术，可从大规模文档集或语料库中挖掘出隐含的主题信息. 然而，在应用 LDA 进行主题建模的过程中，需要人为设置主题个数，以找到最佳的主题数，时间开销很大. HDP 主题建模技术是一种多层蒂利克雷分布，可看成是 LDA 的一种无参估计特例[26]. 在应用 HDP 进行主题建模中，所有文档共享同一个主题集合[26]，可自动找到最佳的主题个数，不需要人为预先设定，可以获得更高质量的主题信息. 因此，我们选择 HDP 主题模型对上一步扩充好的 Web 服务文档进行主题建模. 在主题建模过程中，遵循以下两条重要的特性：

特性 1.　为 Web 服务描述文档中的某个词分配主题，该词被分配到某个主题的概率与该主题已分配到的词的词频总和成正比，同时该词也可以被分配到一个新的主题. 该特性用来获取 Web 服务描述文档的最优主题个数.

特性 2.　为 Web 服务描述文档分配主题，某个 Web 服务被分配到某个主题的概率与已分配到该主题的 Web 服务的个数成正比，允许一个 Web 服务被分配到多个主题. 该特性用来获取服务文档的主题向量.

针对 Web 服务文档的 HDP 主题建模过程如图 3 所示，相关符号及含义如表 2 所示.

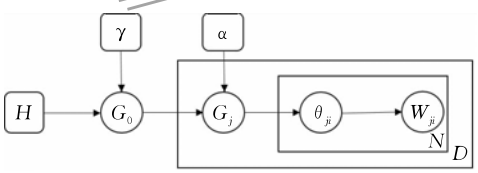

图 3　HDP 主题模型

表 2　HDP 主题模型符号及含义

符号	含义
H	基分布
G_0	文档集的主题分布，依赖参数 γ
G_j	第 j 篇文档的主题分布，依赖参数 α
γ	生成 G_0 的 Beta 分布参数
α	生成 G_j 的 Beta 分布参数
θ_{ji}	第 j 个服务文档中的第 i 个词的主题
W_{ji}	第 j 个服务文档中的第 i 个词

在图 3 所示 HDP 主题模型中，DP 表示蒂利克雷过程，全局变量 G_0 从服务语料库采样获得. 每一个 G_j 都从一个服务文档层获得，j 表示每个服务文

档的序号.θ_{ji}表示第 j 个服务文档中的第 i 个词的主题.D 表示 Web 服务文档个数,N 表示在每个 Web 服务文档中的词语个数,采用以下公式获得模型的变量及分布:

$$G_0 \sim DP(\gamma, H) \tag{1}$$

$$G_j \mid G_0 \sim DP(\alpha, G_0) \tag{2}$$

$$\theta_{ji} \sim G_j \tag{3}$$

$$W_{ji} \sim Multi(\theta_{ji}) \tag{4}$$

采用高斯采样中的马尔可夫链蒙特卡尔方法(Markov Chain Monte Carlo,MCMC),获得"词-主题"条件概率分布,并有以下关系存在:

$$p(T_{m,n}=k \mid \boldsymbol{W}, \alpha, \gamma, \eta, \theta, H) \propto$$

$$(n_{m,k}^{\neg m,n}+\alpha\theta_k) \cdot \frac{n_{k,t}^{\neg m,n}+\eta}{n_k^{\neg m,n}+V \cdot \eta} \tag{5}$$

其中,符号"$\propto$"表示成正比例,$T_{m,n}=k$ 表示为某一个词汇进行抽样后得到的主题,$\boldsymbol{W}$ 表示词向量,θ_k 表示先验概率值,$n_{m,k}$ 表示"词-主题"概率值,$n_{k,t}$ 表示"主题-服务文档"概率值,n_k 表示"主题-类别标志"的单词统计值.上述三个值的上标 $\neg m,n$ 表示统计过程中词 $w_{m,n}$ 是被排除在外的.V 表示词典大小.α,γ,η 三个参数共同决定什么样的主题会被提取出来,特别是参数 η,可以被用来控制主题的数量.式(5)的本质意义在于:一个新的词被分配到一个主题的概率取决于已经分配到该主题中的词的数量,这也是 HDP 模型的最重要的特性.表 3 为 HDP 主题模型生成的服务文档-主题矩阵,其中 d_i 表示服务文档集中的第 i 文档.T_i 表示服务文档集共享的第 i 个主题.图中颜色较深的为该行的最大值.之后,根据该"服务文档-主题"矩阵完成 Web 服务聚类过程.

表 3 HDP 主题模型生成的文档-主题矩阵

	T_1	T_2	T_3	…	T_M
d_1	0.12	**0.40**	0.18	…	0.04
d_2	0.11	0.07	**0.19**	…	0.01
d_3	**0.40**	0.36	0.03	…	0.03
…	…	…	…	…	…
$d_{\lvert D\rvert}$	**0.28**	0.11	0.15	…	0.02

综合上述,面向 Web 服务文档的 HDP 主题建模过程,如算法 1 所示.

算法 1. 面向 Web 服务文档的 HDP 主题建模.

输入:Web API 的描述文档的词向量 $\boldsymbol{W}$,参数 α,γ,η

输出:Web API 描述文档主题向量 $\boldsymbol{T}=(T_1,T_2,\cdots,T_M)$,最优主题数 K

//初始化

1. $K=K_0$(默认 $K=1$);

//所有的计数变量均初始化为 0;

2. $n_{m,k}=0$; $n_m=0$; $n_{k,t}=0$; $n_k=0$;

//U_1 表示使用的主题,U_0 表示未使用的主题列表索引文件

3. $U_1=[1,K]$, $U_0=[K+1,K_{\max}]$;
4. FOR 每一个 Web API 的描述文档 $m\in[1,M]$ DO
5. For Web APIs 描述文档 m 中的每一个词 $n\in[1,N_m]$ DO
6. 为该词汇抽样一个主题 $T_{m,n}=k\sim Multi(1/K)$;
7. 递增计数变量:$n_{m,k}+=1$, $n_{k,t}+=1$, $n_k+=1$;
8. 使用式(1)采样获取 G_0 和 G_m

//Gibbs 采样周期设定

9. WHILE 没有完成 Gibbs 采样 DO
10. FOR 每一个 Web APIs 的描述文档 $m\in[1,M]$ DO
11. FOR m 中每个词 $n\in[1,N_m]$ DO
12. 递减计数变量:$n_{m,k}-=1$, $n_{k,t}-=1$, $n_k-=1$;
13. 利用式(5)在 $[1,k+1]$ 的主题数范围内进行采样
14. 获取主题数 $\tilde{k}\sim p(T_{m,n}=k \mid \boldsymbol{W},\alpha,\gamma,\eta,\theta,H)$
15. IF $\tilde{k}\in[1,K]$ THEN
16. $k^*=U_1(\tilde{k})$;
17. 递增计数变量:$n_{m,k^*}+=1$, $n_{k^*,t}+=1$, $n_{k^*}+=1$;
18. ELSE

//由服务文档 m 中的词语 t 创建出新主题

19. $k^*=pop(U_0)$, push(U_1,k^*), $K+=1$;
20. $n_{m,k^*}=1$, $n_{k^*,t}=1$, $n_{k^*}=1$;
21. 使用式(1)采样获取 G_0 和 G_m
22. $T_{m,n}=k^*$;

//检查是否收敛,如收敛,则读出参数值(平均值)

23. IF 收敛 THEN 读取平均参数值
24. Else IF 没有收敛 THEN
25. FOR $k\in[1,K]$ DO
26. IF $n_k=0$ THEN

//移除空主题

27. $remove(U_1,k)$, push(U_0,k), $K-=1$
28. 计数变量归零:$n_{m,k}=0$, $n_{k,t}=0$, $n_k=0$;
29. 利用式(5)对主题数量进行反复抽样
30. 获取 α,γ 和 η 的参数值.

3.3 基于 SOM 的 API 服务聚类

SOM 是一种典型的无监督神经网络算法,可将高维度向量映射到低维度空间中[21],其网络结构由输入层和隐藏层组成.输入层神经元个数等于输入数据的维度大小,而隐藏层神经元个数则取决于输

入层数据大小. SOM 算法无需预先确定聚类个数，通过自组织方法调节网络节点之间的权重，取得良好的聚类效果. 在 HDP 主题建模之后，对获得的 Web API 的“服务文档-主题”向量，我们采用 SOM 算法进行面向服务主题的聚类，通过自组织方法，将不同的 Web APIs 划分到不同的功能类簇中，每一个类中包含多个具有相似功能的 Web APIs 服务. SOM 自组织神经网络的拓扑结构如图 4 所示.

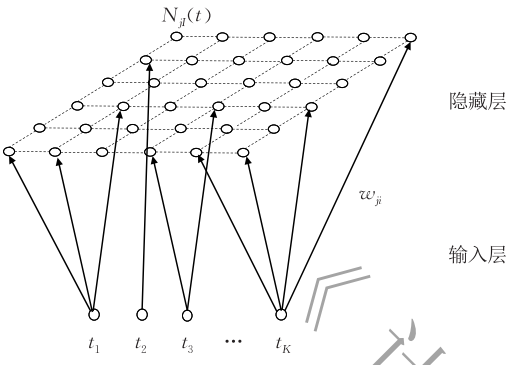

图 4　SOM 自组织神经网络拓扑结构

基于SOM的API服务聚类算法，具体过程如下：

(1) 初始化. 将上一步获得的 Web API 的“服务文档-主题”向量矩阵，作为 SOM 自组织神经网络输入层的输入. 输入层和隐藏层之间的权重 w_{ij} 随机赋值.

(2) 竞争. 与 Web API 服务最接近的神经单元被记为获胜单元(winner)，神经单元之间的相似度由以下式(6)与(7)计算获得：

$$f_j=\sum_{i=1}^{K}(t_i-w_{ji})^2,\ j\in\{1,2,\cdots,M\} \tag{6}$$

$$f_{I(t)}=\min_{j\in\{1,2,\cdots,M\}}\{f_j\} \tag{7}$$

其中，$I(t)$是获胜单元的编号.

(3) 相邻节点权重的计算. 在找到获胜单元后，其相邻节点 $N_{jI(t)}$ 的权重可通过式(8)计算：

$$N_{jI(t)}=\exp\left(-\frac{S_{j,I(t)}^2}{2\sigma^2}\right) \tag{8}$$

(4) 权重更新. 通过式(9)，对获胜单元及相邻节点的权重进行更新：

$$\Delta w_{ji}=\eta(t)N_{jI(t)}(t)(t_i-w_{ji}) \tag{9}$$

表 4 为基于 SOM 的 API 服务聚类算法中一些参数设置及含义. 图 5 为随机选择 10 个 Web API 服务的 SOM 聚类效果. SOM 算法根据相似度将每一个 Web API 服务映射到不同的节点/区域中实现聚类. 在图 5 中，行数和列数都为 2，在隐藏层中采用四个节点/区域用于聚类. 其中，区域(0,0)中包含两个同属于 Fun 分类的 Web APIs 服务，区域(0,1)为 Music 类，区域(1,0)为 deadpool 类，区域(1,1)为 Mapping 类.

表 4　SOM 算法中的一些参数及含义

符号	含义
t_i	输入层第 i 个节点的值
$s_{j,I(t)}$	隐藏层中，节点 j 与获胜节点 $I(t)$的距离
σ	获胜节点几何邻域大小随时间缩减的控制变量
$\eta(t)$	一种随时间而衰减的可变学习速度，表示在训练过程中，权值调整的幅度越来越小

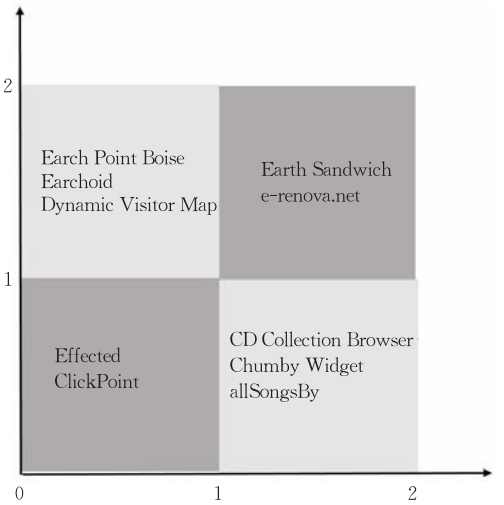

图 5　SOM 聚类实例及效果

3.4　基于 DeepFM 的 API 服务预测与排序

在推荐系统中，数据的不同特征组合方式将对推荐性能产生非常大的影响[24]. DeepFM 模型可同时提取出数据中的复杂的高阶特征和低阶特征组合关系[25]，将这些组合关系应用于服务推荐可大大提升推荐精准度. 如图 6 所示，DeepFM 模型由两部分组成：左边部分的因子分解模型，用于获得低阶特征组合关系；右边部分的深度神经网络模型，用于获取高阶特征组合关系. 因子分解模型[13]，能够对低阶(一阶和二阶)特征组合进行建模. 在数据特别稀疏

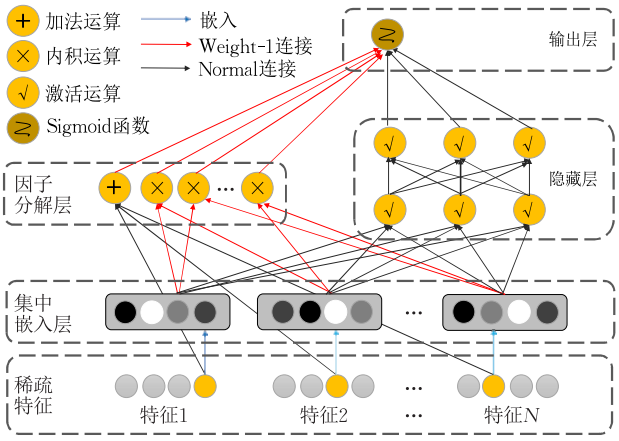

图 6　深度因子分解机模型

的情况下,运用二阶特征组合关系能够有效提升推荐性能.深度神经网络模型用于学习和挖掘数据中的高阶组合关系,可进一步提升推荐性能.

DeepFM 模型的预测公式如下:

$$\hat{y}=sigmoid(y_{\text{FM}}+y_{\text{DNN}}) \tag{10}$$

其中,y_{FM}为因子分解模型(FM)的输出结果,y_{DNN}为深度神经网络模型(DNN)的输出结果,$\hat{y}\in(0,1)$为预测结果.

基于 DeepFM 模型,每一个 API 相对于目标 Mashup 的评分值能够被预测出来,经过排序后获得最终的推荐列表.图 7 是一个典型的基于 DeepFM 模型的 Web API 预测评分实例,其中的数据由两部分组成:输入特征向量集(Feature Vector X)和输出目标集(Target Y).每一行由一个特征向量 $\boldsymbol{x}_i$ 及对应的目标评分值 y_i 构成.

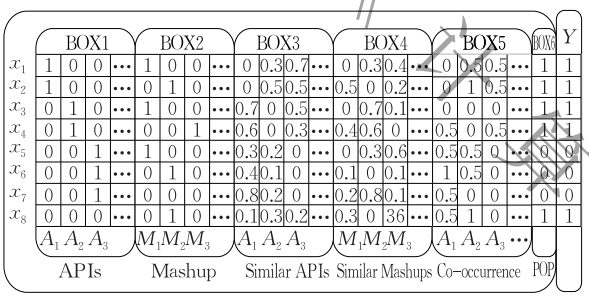

图 7 利用 DeepFM 模型的 Web API 评分实例

表 5 中给出了图 7 中内置的各个 Box 的含义,其具体说明如下:

(1) Box1. 假设 *Web API* 的 *one-hot* 编码的集合为:$O_A=\{o_{A1},o_{A2},\cdots,o_{AN}\}$,其中 N 为 Web API 服务的个数.同时,每个 o 的维度也为 N,那么第 1 个 Web API 的 one-hot 编码就是 $o_{A1}=[1,0,0,\cdots,0]_{1\times N}$.

(2) Box2. 假设 Mashup 的 one-hot 编码的集合为:$O_M=\{o_{M1},o_{M2},\cdots,o_{ML}\}$,其中 L 为 Mashup 的个数,同时,每个 o 的维度也为 L,那么第 1 个 Mashup 的 one-hot 编码就是 $o_{M1}=[1,0,0,\cdots,0]_{1\times L}$.

(3) Box3. 使用 Web API 的描述文本作为 HDP 主题模型的输入并进行主题建模,获得每一个 Web API 的服务文档-主题向量.采用余弦相似度计算公式计算出 Web API 之间的相似度.

(4) Box4. 使用 Mashup 的描述文本作为 HDP 主题模型的输入并进行主题建模,获得每一个 Mashup 的服务文档-主题向量.采用余弦相似度计算公式计算出 Mashup 之间的相似度.

(5) Box5. 从原始的数据集中提取出 Web API 的共现信息.共现信息表示的是 Web API 的历史组合关系,如果两个 Web API 同时被一个 Mashup 调用过,就说明它们是共现的. Web API 之间的共现系数计算公式如下:

$$Co(a_i,a_j)=\frac{|a_i\cap a_j|}{|a_i\cup a_j|} \tag{11}$$

其中,$|a_i\cup a_j|$表示 Web API a_i 与 Web API a_j 被 Mashup 调用的总次数,$|a_i\cap a_j|$表示 a_i 与 a_j 同时被同一个 Mashup 所调用的次数.

(6) Box6. Web API 的流行度反映 Web API 的流行程度.流行度越高,说明该 Web API 越容易被调用.流行度的计算公式如下:

$$pop(a_i)=\frac{Fre(a_i)-MinFre(Category(a_i))}{MaxFre(Category(a_i))-MinFre(Category(a_i))} \tag{12}$$

其中,$Fre(a_i)$表示 a_i 被 Mashup 调用的总次数,$Category(a_i)$表示与 a_i 具有相同分类的其它所有的 Web API 服务,$MinFre(*)$表示 Web API 被 Mashup 调用的最小次数,$MaxFre(*)$表示 Web API 被 Mashup 调用的最大次数.

表 5 多维信息矩阵

Box 名称	含义
Box1	对 Web API 标注序号,并且将每一个 Web API 映射成唯一的一个 one-hot 编码;
Box2	对 Mashup 标注序号,并且将每一个 Web API 映射成唯一的一个 one-hot 编码;
Box3	对 Web API 的描述文档进行主题建模,获得文档-主题向量,再计算每个 Web API 之间的余弦相似度;
Box4	对 Mashup 的描述文档进行主题建模,获得文档-主题向量,再计算每个 Mashup 之间的余弦相似度;
Box5	计算每个 API 之间的共现系数;
Box6	计算每个 API 的流行度;
Target Y	如果 API 被 Mashup 实际调用过,那么为 1,反之为 0.

3.5 Web 服务推荐

针对开发者用户的 Mashup 需求,服务推荐分成如下步骤来进行:

(1) 首先,当开发者用户提交 Mashup 需求时,通过使用 LDA 主题建模技术,其需求文本描述将被转换成一个 Mashup 需求主题向量.

(2) 集成每一个 Mashup 类簇中所有 Mashup 的描述文本,并使用 LDA 主题建模技术,获得 Mashup 类簇主题向量.

(3) 使用 JS 距离公式计算 Mashup 需求主题向量与 Mashup 类簇主题向量之间的相似度,找到匹配度最高的那个 Mashup 类簇.

(4) 使用 DeepFM 模型来预测并排序 API 质量

得分，推荐得分靠前的若干个 API 服务给开发者用户.

4　实验与分析

4.1　实验设置与数据集

实验中所有代码使用 Python 编程语言编写，电脑配置为 32GB RAM，Windows 10 OS，有一块能加速计算的图形显卡 NVIDIA Quadro K4000. 为了验证本文提出方法的有效性，我们从 ProgrammableWeb 平台上爬取了 12919 个 API 和 6791 个 Mashup，下载地址为：http://49.123.0.60:8080/MashupNetwork2.0/dataset.jsp，在该 URL 中分别提供数据集（Download Dateset）和数据预处理源代码（Download Code）的下载；实验中所采用的 SOM 算法和 DeepFM 算法的源代码下载地址分别为 https://github.com/JustGlowing/minisom 和 https://github.com/ChenglongChen/tensorflow-DeepFM.

我们选择包含最多 Mashup 的前 20 类作为实验数据集，其详细的信息如表 6 所示. 我们将该实验数据集分成 70%的训练集和 30%的测试集. 实验中使用的“Mashup 需求（文档）”即为测试集中真实存在的 Mashup 描述文档. 在实验设置中，Word2vec 算法的词向量维度为 200，窗口长度为 10，最小出现次数为 10；DeepFM 算法的学习率为 0.001，优化器为 adam，L2 正则系数为 0.01，向量 $\boldsymbol{V}$ 长度为 8.

表 6　包含最多 Mashup 的前 20 类的分布情况

类别	Mashup 数量	类别	Mashup 数量
Mapping	1038	Sports	112
Search	305	Telephone	99
Social	298	Blogging	98
Ecommerce	295	Reference	98
Photos	260	Electronic Signature	95
Music	251	Widgets	86
Travel	192	Visualizations	78
Video	174	Humor	67
Messaging	137	Government	66
Mobile	126	Games	54

4.2　Mashup 聚类实验及分析

Mashup 聚类是实现 Web API 推荐的基础，我们设置了一组实验来比较不同聚类方法的实验效果.

4.2.1　评估指标

采用准确率、召回率、纯度和熵这四种指标来衡量不同聚类方法的性能. 据调研，ProgrammableWeb 网站上的 Mashup 分类是人工设定的，不一定准确. 在此，我们针对 Mashup 分类进行优化：为 Mashup(MS) 设定一个标准分类结果 $SM=\{SM_1,SM_2,\cdots,SM_V\}$，该分类由多名领域专家和数十名用户共同确定，对应实验获得的分类结果为 $M=\{M_1,M_2,\cdots,M_K\}$，那么，准确率和召回率的计算方法如下：

$$Precision(M_i)=\frac{|SM_i\cap M_i|}{|M_i|} \tag{13}$$

$$Recall(M_i)=\frac{|SM_i\cap M_i|}{|SM_i|} \tag{14}$$

其中，$|SM_i|$ 表示类别 SM_i 中的 Mashup 的数量；$|M_i|$ 表示类别 M_i 中的 Mashup 的数量；$|SM_i\cap M_i|$ 表示类别 SM_i 和 M_i 中相同的 Mashup 的数量. 需要说明的是：实验获得的分类结果 M 需要与标准分类结果 SM 进行比对，以计算得到方法的性能指标. 例如：标准分类 Mapping 可能和实验分类 A、B、C 都有交集，但是 Mapping 和类簇 C 的 $|SM_i\cap M_i|/|M_i|$ 值最大，那么就认为类簇 C 就是 Mapping 分类. 本文采用文献[19]中的算法 1 实现实验分类与标准分类的匹配：首先获得 $|SM_i\cap M_i|/|M_i|$ 取值最大的 $(SM_{\max},M_{\max})$ 对并设置 $SM_{\max}$ 和 $M_{\max}$ 为一对匹配；然后将 $SM_{\max}$ 和 $M_{\max}$ 分别从标准分类和实验分类集合中删除，重复这个过程直至全部标准分类和实验分类得到匹配.

此外，我们采用纯度和熵来评估服务聚类的精确度. 其中，式(15)表示每一个 Mashup 类别 M_i 的纯度；式(16)表示每一个 Mashup 类别 M_i 的熵.

$$Purity(M_i)=\frac{1}{|M_i|}\max_j n_i^j,\ 1\le i\le K,\ 1\le j\le V \tag{15}$$

$$Entropy(M_i)=-\frac{1}{\log V}\sum_{j=1}^{V}\frac{n_i^j}{|M_i|}\log\left(\frac{n_i^j}{|M_i|}\right) \tag{16}$$

其中，$|M_i|$ 表示类别 M_i 中的 Mashup 的数量，n_i^j 表示属于类别 SM_j 且成功分类到 M_i 中的服务数量.

简言之，准确率、召回率、纯度的值越大，而熵值越少的话，表示服务聚类的效果越好.

4.2.2　基准方法

(1) TF-IDF. 由 TF-IDF 技术建模获得 Mashup 的词向量空间，并利用余弦相似度公式计算 Mashup 之间的相似度，实现 Mashup 的聚类[28-29].

(2) LDA. 使用 LDA 主题模型建模获得 Mashup 的“文档-主题”向量，并利用 KL 距离公式计算 Mashup 之间的主题相似度，实现 Mashup 的聚类[33].

(3) LDA-K. 该方法首先使用 LDA 主题模型建模获得 Mashup 的“文档-主题”向量；然后，利用

K-means 算法对 Mashup 进行聚类[34-35].

(4) HDP-K. 该方法首先使用 HDP 主题模型建模 Mashup 获得其“文档-主题”向量[26]；然后，利用 *K*-means 算法对 Mashup 进行聚类.

(5) HDP-SOM. 该方法首先使用 HDP 主题模型建模 Mashup 获得“文档-主题”向量[26]；然后，利用 SOM 神经网络算法对 Mashup 进行聚类[21-23].

4.2.3 实验结果及分析

图 8 给出了不同 Mashup 类别数(选择包 Mashup 类别数分别为 5、10、15、20 和 25，即 top-*N*=5/10/15/20/25)情况下的服务聚类性能. 实验结果表明 HDP-SOM 方法在准确率、召回率、纯度和熵值这四个指标上，明显地好于其它的服务聚类方法. 具体来说：

(1) TF-IDF 方法的性能是最差的. 因为它仅仅使用词语向量空间模型表征 Mashup 的内容特征，并未考虑其包含的丰富的语义信息. LDA、LDA-K、HDP-K、HDP-SOM 这四种方法都利用主题建模技术挖掘 Mashup 文档潜在的主题语义信息，改善了服务聚类效果. HDP-K 与 HDP-SOM 方法的效果要好于 LDA 与 LDA-K. 例如：如图 8(a)所示，在 Mashup 类别数为 10(top-*N*=10)的情况下，HDP-K 方法的准确率相比于 LDA、LDA-K 分别提升了 8.46%、4.73%；HDP-SOM 方法的准确率相比于 LDA、LDA-K 方法分别提升了 18.06%、14%. 理由在于：HDP 主题模型能够自动找到 Mashup 文档的最优主题数，提高了相似度计算的精度. HDP-SOM 方法的性能优于 HDP-K，原因在于：*K*-means 算法对初始点的选取有较高的依赖性，在算法运行过程中容易陷入局部极小值的问题；而 SOM 聚类算法能够利用输入向量的拓扑结构以及相似度进行聚类，从而获得较好的聚类效果.

(2) Mashup 类别数为 10 的情况下(top-*N*=10)的服务聚类效果是最好的. 如图 8(c)所示，HDP-SOM 方法的纯度相比于 TF-IDF、LDA、LDA-K、HDP-K 四种方法分别提升了155.1%、11.1%、7.39%、6.05%. 当 Mashup 类别数增大(top-*N*=10/15/20/25)或变小(top-*N*=5)时，其服务聚类性能随之降低. 一方面，当服务类别数太少时，其包含的个体服务数量偏少，导致用于聚类的内容信息过少. 另外，当服务类

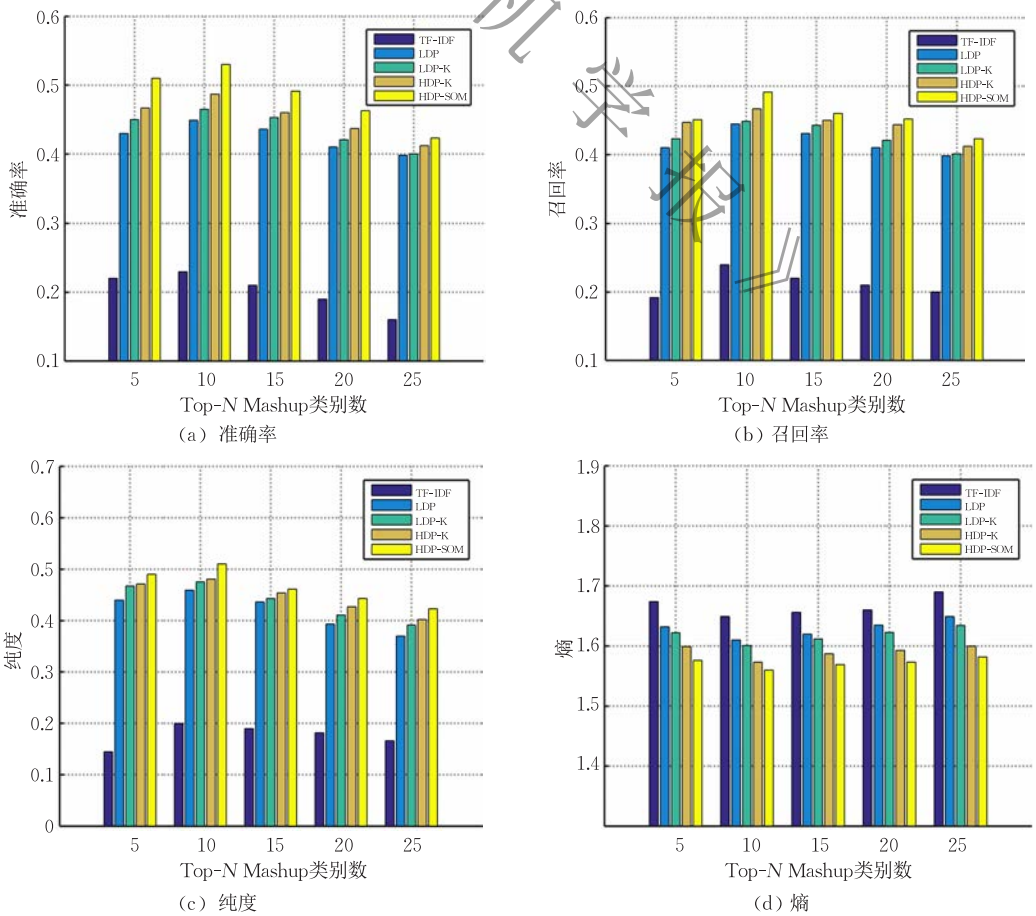

图 8 Mashup 聚类性能比较

别数量增大时，也将包含更多的服务数量少的那些服务类别，致使服务聚类性能降低.

此外，我们对不同聚类方法的时间复杂度进行分析比较. LDA 的时间复杂度为 $O(N_{\text{iter}}N_D K\bar{l})$，$N_{\text{iter}}$为迭代次数，$N_D$为服务文档个数，$K$ 为主题个数，$\bar{l}$为服务文档的平均长度；TF-IDF 的时间复杂度呈线性，低于 LDA；HDP 需要自动确定主题个数，时间复杂度高于 LDA；K-Means 的时间复杂度为 $O(T\times N_D\times k\times m)$，$N_D$为服务文档个数，$k$ 为聚类个数，m 为每个服务文档的特征项个数，T 为算法迭代次数，实际上也就是 $O(N_D)$；SOM 算法时间复杂度为 $O(N_D)$. 因此，不同聚类方法的时间复杂度关系为：HDP-SOM≈HDP-K＞LDA-K＞LDA＞TF-IDF. 也就是说，在花费一定时间开销的情况下，HDP-SOM 的服务聚类性能比其它方法要好.

4.3　API 推荐实验及分析

针对开发者用户的 Mashup 需求，以服务聚类的结果为基础，我们设置了一组实验来比较不同推荐方法的实验效果.

4.3.1　评估指标

除了准确率、召回率之外，我们使用 $DCG@R$ (Discounted Cumulative Gain@top R，折扣累计增益)指标来衡量 API 排序及推荐的精度，其中，R 表示推荐列表中 API 的个数，$r(i)$表示第 i 个推荐的 API 的相关性得分. 如果所推荐的第 i 个 API 被真实数据集中的 Mashup 所调用，那么 $r(i)=1$，否则 $r(i)=0$. $DCG@R$ 值越大，表明推荐精度越好. 其公式如下：

$$DCG@R=\sum_{i=1}^{R}\frac{2^{r(i)}-1}{\log_2(1+i)} \tag{17}$$

此外，使用汉明距离(Hamming Distance，HMD)来评估推荐的多样性. HMD 用来度量在不同 Mashup 需求下，推荐多样化 API 的能力及质量. 较大的 HMD 值，意味着更好的推荐多样性，其定义如下：

$$HMD(m_i,m_j)@R=1-\frac{Q(m_i,m_j)}{R} \tag{18}$$

其中，$Q(m_i,m_j)$表示在 Mashup 类中 m_i 和 m_j 所对应的 API 推荐列表中相同的 API 的数量，R 表示推荐列表的长度. 如果 m_i 和 m_j 的 API 推荐列表完全相同，则 $H(m_i,m_j)=0$；反之，$H(m_i,m_j)=1$.

4.3.2　基准方法

(1) TF-IDF. 通过使用词向量模型进行推荐，推荐与所给 Mashup 描述文档所包含的 API 相似的 API 服务[28-29]. 在该模型中，假设目标 Mashup(m)的词向量空间为 $V^{(m)}$. 那么，第 i 个 API 的词向量空间表示为 $V^{(a_i)}$，可以得到，第 i 个 API 与目标 Mashup 的文本相似度的计算公式为

$$Sim(a_i,m)=\frac{V^{(a_i)}V^{(m)}}{\|V^{(a_i)}\|\|V^{(m)}\|} \tag{19}$$

(2) LDA-K-CF. 该方法在 LDA-K 聚类方法的基础上[33-35]，使用协同过滤算法进行 API 推荐[9].

(3) HDP-K-CF. 该方法在 HDP-K 聚类方法的基础上，使用协同过滤算法进行 API 推荐[9].

(4) LDA-K-FM. 该方法在 LDA-K 聚类方法的基础上[33-35]，利用 FM 算法预测 API 得分[13]，将得分最高的前 N 个 Web API 作为推荐结果.

(5) HDP-K-FM. 该方法在 HDP-K 聚类方法的基础上[26]，利用 FM 算法预测 API 得分[13]，将得分最高的前 N 个 Web API 作为推荐结果.

(6) HDP-S-FM. 该方法在 HDP-SOM 聚类方法的基础上[21-23,26]，利用 FM 算法预测 API 得分[13]，将得分最高的前 N 个 Web API 作为推荐结果.

(7) HDP-S-DeepFM. 即本文提出的方法. 该方法在 HDP-SOM 聚类方法的基础上[21-23,26]，在与目标 Mashup 最相近的服务类簇中使用 DeepFM 算法计算 API 得分[25]，最后选择得分最高的前 N 个 Web API 作为推荐结果.

4.3.3　实验结果及分析

(1) 不同扩充情况对 API 服务推荐的影响

在 HDP-S-DeepFM 方法中，我们考察在不同 Mashup 类别数(即 top-N=5/10/15/20/25)，以及不同扩充词(分别为 3、5、10 以及无扩充)情况下的 API 服务推荐效果.

从图 9 中可以看出：当扩充词数为 3 时，其准确率、召回率、DCG 以及 HMD 值都要优于其它扩充情况. 扩充词数为 3 分别比无扩充情况、扩充词数为 5 以及扩充词数为 10 的情况下在准确率上要高出 12.4%、17.6%和 28.0%. 在无扩充以及随着扩充词数增加的情况下，聚类的效果反而降低. 一方面，不扩充时，原始的服务文本较短，包含的信息较少，导致服务推荐的效果一般；另外，过多的扩充词，在带来有用内容信息的同时也会带来噪音，会稀疏原始服务文本的语义信息，降低服务推荐的效果. 同样，Mashup 类别数为 10 的情况下(top-N=10)的服务推荐效果是最好的.

(2) API 服务推荐性能比较

在 HDP-S-DeepFM 方法中，我们选择扩充词

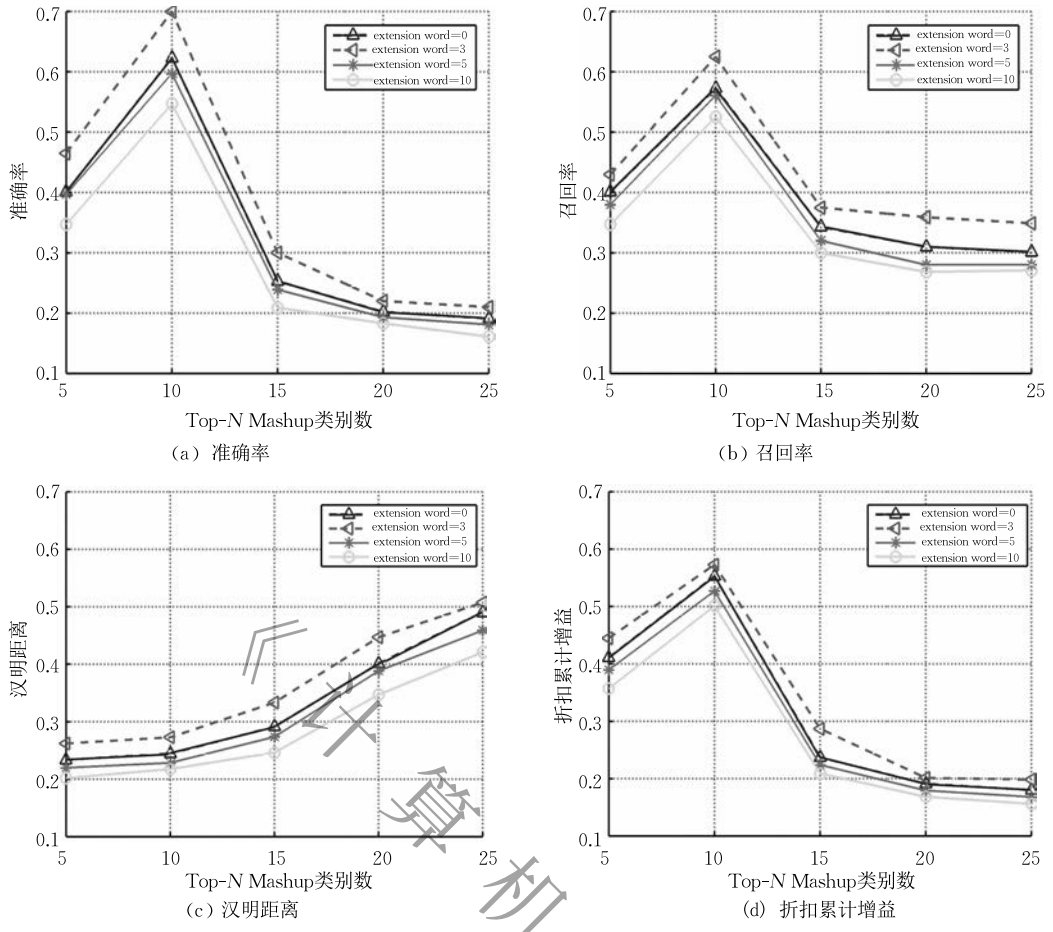

图 9 不同扩充情况下 HDP-S-DeepFM 方法推荐效果

数为 3、服务类别数为 10 的情况下，与其它方法进行对比实验. 表 7 给出了不同推荐方法的性能比较结果，包括准确率、召回率、DCG 和 HMD 值. 从实验结果中可知：HDP-S-DeepFM 方法的性能要明显地优于其它方法. 比如：HDP-S-DeepFM 方法的准确率相比于 TF-IDF、LDA-K-CF、LDA-K-FM、HDP-K-CF、HDP-K-FM、HDP-S-FM 六种方法分别提升了 196.2%、49%、33.8%、31.2%、12.3%、10.3%；而其 DCG 值相比于这六种方法分别提升了 161.8%、26.4%、18.6%、16.2%、6.73%、4.5%. 具体来说：

表 7 Web API 推荐性能比较

Methods	Precision	Recall	DCG	HMD
TF-IDF	0.2361	0.1537	0.2192	0.7336
LDA-K-CF	0.4693	0.4836	0.4538	0.5481
LDA-K-FM	0.5225	0.5019	0.4837	0.5376
HDP-K-CF	0.5331	0.5281	0.4936	0.5211
HDP-K-FM	0.6226	0.5419	0.5376	0.5135
HDP-S-FM	0.6341	0.5503	0.5491	0.5233
HDP-S-DeepFM	**0.6993**	**0.6151**	**0.5738**	**0.4975**

① TF-IDF 方法的推荐效果最差. 因为 Mashup 与 API 服务的描述文档通常都比较短小、包含的信息有限. TF-IDF 很难从这些短文本中提取出足够有用的信息用于推荐. 基于 LDA 主题模型的方法（LDA-K-CF、LDA-K-FM）的推荐效果要比基于 HDP 主题模型的方法（HDP-K-CF、HDP-K-FM、HDP-S-FM）的效果要差一些. 因为 LDA 主题模型所需要设置的参数较多，仅仅通过人工调整难以获得最佳的实验参数；HDP 主题模型可自动寻优到最佳的主题数.

② 包含因子分解机模块的方法（LDA-K-FM、HDP-K-FM）的实验效果也要优于普通的协同过滤方法（LDA-K-CF、HDP-K-CF）. 原因在于：协同过滤方法仅仅考虑了文档之间的相似关系；而因子分解机不但考虑了相似关系，还利用了 Mashup 与 API 之间的调用关系、API 之间的相似关系、Mashup 之间的相似关系以及 API 的流行度等多种因素.

③ HDP-S-DeepFM 的实验性能好于 HDP-S-FM. 深度因子分解机更擅长抽取输入数据中的高

阶组合关系. 例如，API 被 Mashup 调用的次数越多，那么与该 API 功能相似的 API 也有可能被调用，与该 Mashup 相似的 Mashup 也可能调用功能相似的 API. 将这些高阶信息考虑在内，能够获得更加准确的推荐结果. 此外，更多的（三阶及以上的）QoS 属性交互关系，也大大提升了推荐的性能.

同样，我们对不同推荐方法的时间复杂度进行分析比较. CF 和 FM 的时间复杂度都为 $O(N_D^2)$，N_D 为服务文档个数，DeepFM 的时间复杂度比 CF 算法和 FM 要高. 因此，以各种聚类方法的时间复杂度关系为基础，导出不同推荐方法的时间复杂度关系为：HDP-S-DeepFM＞HDP-S-FM≈HDP-K-FM≈HDP-K-CF＞LDA-K-FM＞TF-IDF. 也就是说，在花费一定时间开销的情况下，HDP-S-DeepFM 的服务推荐性能比其它方法要好.

5　总结与展望

本文针对“推荐合适的 API 服务以构建高质量 Mashup”问题，提出了一种融合 SOM 功能聚类与 DeepFM 质量预测的 API 服务推荐方法. 该方法首先采用维基百科语料库作为外部语料库，对 Web 服务描述文档进行扩充；然后，利用 HDP 模型建模 Web 服务的主题信息，并使用 SOM 神经网络进行 Mashup 聚类；最后，利用 DeepFM 模型挖掘预测并排序 Web 服务的得分，推荐高质量的 API 服务给开发者用户. 实验结果表明：相比于其它典型方法，本文所提方法取得了较好的聚类以及推荐效果. 在后续的工作中，我们将探索如何脱离现有的 API 服务集合，将提出的方法应用到更开放的服务环境中进行 API 的推荐.

参　考　文　献

[1] Wu Zhao-Hui. Modern services and services computing. Communications of the CCF, 2017, 13(2): 16-17(in Chinese)
(吴朝晖. 现代服务业与服务计算. 中国计算机学会通讯, 2017, 13(2): 16-17)

[2] Mei Hong. Understanding on some challenges to software technology for big data. Communications of the CCF, 2013, 9(12): 40-43(in Chinese)
(梅宏. 对大数据时代软件技术面临挑战的若干认识和思考. 中国计算机学会通讯, 2013, 9(12): 40-43)

[3] Deng S, Wu H, Hu D, Zhao J. Service selection for composition with QoS correlations. IEEE Transactions on Services Computing, 2016, 9(2): 291-303

[4] Xia B, Fan Y, Tan W, et al. Category-aware API clustering and distributed recommendation for automatic mashup creation. IEEE Transactions on Services Computing, 2015, 8(5): 674-687

[5] Xu Xiao-Fei, Wang Zhong-Jie, Feng Zhi-Yong. Big service: Applications and influence in the big data environment. Communications of the CCF, 2017, 13(2): 24-30(in Chinese)
(徐晓飞，王忠杰，冯志勇. 大数据环境下的大服务与应用及其影响. 中国计算机学会通讯, 2017, 13(2): 24-30)

[6] Deng S, Wu H, Taheri J, et al. Cost performance driven service Mashup: A developer perspective. IEEE Transactions on Parallel and Distributed System, 2016, 27(8): 2234-2247

[7] Liu X, Fulia I. Incorporating user, topic, and service related latent factors into Web service recommendation//Proceedings of the IEEE International Conference on Web Services (ICWS). New York, USA, 2015: 185-192

[8] Liu L, Lecue F, Mehandjiev N. Semantic content-based recommendation of software services using context. ACM Transactions on the Web, 2013, 7(3): 17-20

[9] Zheng Z, Ma H, Lyu M, King I. QoS-aware Web service recommendation by collaborative filtering. IEEE Transactions on Services Computing, 2011, 4(2): 140-152

[10] Zheng Z, Ma H, Lyu M, King I. Collaborative Web service QoS prediction via neighborhood integrated matrix factorization. IEEE Transactions Services Computing, 2013, 6(3): 289-299

[11] Wang S, Zheng Z, Wu Z, et al. Reputation measurement and malicious feedback rating prevention in Web service recommendation system. IEEE Transactions on Services Computing, 2015, 5(8): 755-767

[12] Tang M, Jiang Y, Liu J, Liu X. Location-aware collaborative filtering for QoS-based service recommendation//Proceedings of the IEEE International Conference on Web Services (ICWS). Honolulu, USA, 2012: 202-209

[13] Cao B, Liu J, Wen Y, et al. QoS-aware service recommendation based on relational topic model and factorization machines for IoT Mashup applications. Journal of Parallel and Distributed Computing, 2018, accepted

[14] Li C, Zhang R, Huai J, Sun H. A novel approach for API recommendation in Mashup development//Proceedings of the IEEE International Conference on Web Services (ICWS). Anchorage, USA, 2014: 289-296

[15] Yao L, Sheng Q, Ngu H, et al. Unified collaborative and content-based Web service recommendation. IEEE Transactions on Services Computing, 2015, 8(3): 453-466

[16] Tian G, Wang J, He K. Leveraging auxiliary knowledge for Web service clustering. Chinese Journal of Electronics, 2016, 25(5): 858-865

[17] Liu J, Liu F, Li X, et al. Web service clustering using relational database approach. International Journal of Software Engineering and Knowledge Engineering, 2015, 25(8): 1365-1393

[18] Chen L, Wang Y, Yu Q, et al. WT-LDA: User tagging

augmented LDA for Web service clustering//Proceedings of the 9th International Conference on Service Oriented Computing (ICSOC). Berlin, Germany, 2013: 162-176

[19] Tian Gang, He Ke-Qing, Wang Jian, et al. Domain-oriented and tag-aided Web service clustering method. Acta Electronica Sinica, 2015, 43(7): 1266-1274(in Chinese)
(田刚, 何克清, 王健等. 面向领域标签辅助的服务聚类方法. 电子学报, 2015, 43(7): 1266-1274)

[20] Shi M, Liu J, Zhou D, et al. WE-LDA: A word embeddings augmented LDA model for Web services clustering//Proceedings of the IEEE International Conference on Web Services(ICWS). Honolulu, USA, 2017: 9-16

[21] Kohonen T. Self-Organizing Maps. 3rd Edition. Berlin, Germany: Springer, 2001

[22] Vesanto J, Alhoniemi E. Clustering of the self-organizing map. IEEE Transactions on Neural Network, 2000, 11(3): 586-600

[23] Chang W, Pang L, Kai M. Application of self-organizing map to failure modes and effects analysis methodology. Neurocomputing, 2017, 249: 314-320

[24] Zhang S, Yao L, Sun A. Deep learning based recommender system: A survey and new perspectives. ACM Journal on Computing and Cultural Heritage, 2017, 1(1): 1-35

[25] Guo H, Tang R, Ye Y, et al. DeepFM: A factorization-machine based neural network for CTR prediction//Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI). Melbourne, Australia, 2017: 1725-1731

[26] Teh Y, Jordan M, Beal M. Hierarchical Dirichlet processes. Journal of the American Statistical Association, 2006, 101(476): 1566-1581

[27] Zhou Z, Sellami M, Gaaloul W, et al. Data providing services clustering and management for facilitating service discovery and replacement. IEEE Transactions on Automation Science and Engineering, 2013, 10(4): 1131-1146

[28] Liu W, Wong W. Web service clustering using text mining techniques. International Journal Agent-Oriented Software Engineering, 2009, 3(1): 6-26

[29] Elgazzar K, Hassan A, Martin P. Clustering WSDL documents to bootstrap the discovery of Web services//Proceedings of the IEEE International Conference on Web Services (ICWS). Miami, USA, 2010: 147-154

[30] Wu J, Chen L, Zheng Z, et al. Clustering Web services to facilitate service discovery. Knowledge and Information Systems, 2014, 38(1): 207-229

[31] Chen L, Hu L, Wu J, et al. WTCluster: Utilizing Tags for Web service clustering//Proceedings of the 9th International Conference on Service Oriented Computing (ICSOC). Paphos, Cyprus, 2011: 204-218

[32] Wu J, Chen L, Xie Y, Zheng Z. Titan: A system for effective Web service discovery//Proceedings of the 21st International World Wide Web Conference (WWW). Lyon, France, 2012: 441-444

[33] Cassar G, Barnaghi P, Moessner K. Probabilistic methods for service clustering//Proceedings of the International Workshop on Service Matchmaking & Resource Retrieval in the Semantic Web. Shanghai, China, 2011: 4-20

[34] Li Zheng, Wang Jian, Zhang Neng, et al. A topic-oriented clustering approach for domain services. Journal of Computer Research and Development, 2014, 51(2): 408-419 (in Chinese)
(李征, 王健, 张能等. 一种面向主题的领域服务聚类方法. 计算机研究与发展, 2014, 51(2): 408-419)

[35] Cao B, Liu X, Liu J, Tang M. Effective Mashup service clustering method by exploiting LDA topic model from multiple data sources//Proceedings of the Asia-Pacific Services Computing Conference (APSCC). Bangkok, Thailand, 2015: 1-16

[36] Chen X, Zheng Z, Yu Q, Lyu R. Web service recommendation via exploiting location and QoS information. IEEE Transactions on Parallel and Distributed System, 2014, 25(7): 1913-1924

[37] Lo W, Yin J, Deng S, et al. Collaborative Web service QoS prediction with location-based regularization//Proceedings of the IEEE International Conference on Web Services(ICWS). Honolulu, USA, 2012: 464-471

[38] Gao W, Chen L, Wu J, Gao H. Manifold-learning based API recommendation for mashup creation//Proceedings of the IEEE International Conference on Web Services(ICWS). New York, USA, 2015: 432-439

[39] Yao L, Wang X, Sheng Q, et al. Service recommendation for mashup composition with implicit correlation regularization // Proceedings of the IEEE International Conference on Web Services(ICWS). New York, USA, 2015: 217-224

CAO Bu-Qing, Ph. D., associate professor. His research interests include service-oriented software engineering, service computing and cloud computing.

XIAO Qiao-Xiang, M. S. candidate. Her research interests focus on service clustering.

ZHANG Xiang-Ping, M. S. candidate. His research interests focus on service recommendation.

LIU Jian-Xun, Ph. D., professor, Ph. D. supervisor. His research interests include the theory and application of workflow management, service computing and cloud computing, and service-oriented software engineering.

Background

Recently, Mashup technology, a lightweight service composition technology, has emerged on the Internet. Software developer can combine the existing API services through the way of Mashup to build novel Web applications for meeting users' complex requirements. Driven by the idea of service-oriented service computing, more and more enterprises encapsulate their data or resources as API services and publish them to the Internet, resulting in the number of API services doubled. It is becoming a challenge to quickly and effectively discover API services that meet the Mashup requirement of software developer from such a large-scale service set. To solve this problem, some researchers exploit service recommendation to improve the accuracy of service discovery, which includes content-based, quality of service based, clustering-based, and hybrid service recommendation technologies. As we investigated, although these service recommendation methods greatly improve the effect of service discovery, there are some problems for them. First of all, service documents are usually short, with sparse features and less information. It is difficult for traditional document modeling technique to fully model and mine the deep semantics from short service documents. Secondly, aiming to the data sparsity and cold start problems, besides the general QoS information (such as geographical location, historical collaboration records), the popularity and co-occurrences can be used as auxiliary information to facilitate service recommendation. Furthermore, both collaborative filtering and matrix factorization only are suitable for general recommendation tasks, which cannot model the high-order interaction relationship between QoS features.

To this end, this paper proposes an API service recommendation method via combining SOM-based functionality clustering and DeepFM-based quality prediction. In this method, Wikipedia is used as external corpus to extend the description of short service document, and HDP topic model and SOM neural network are integrated to perform the functionality clustering of oriented-content service. DeepFM is exploited to model the feature composition relationships (including high-order and low-order composition relationships) between QoS features, and recommend high-quality API services to software developers. Compared with other methods, the experimental results show that the proposed method achieves better accuracy of service clustering and recommendation.

The work is mainly supported by the National Natural Science Foundation of China under Grant No. 61873316, and the Hunan Provincial Natural Science Foundation of China under grant No. 2017JJ2098. The two projects mainly aim to study on service Mashup development method enhance by machine learning. This paper presents a SOM and DeepFM based API service recommendation to create Mashup application with high-quality, which is an effective study in service Mashup development.