南京大学

研 究 生 毕 业 论 文

（申 请 硕 士 学 位）

论 文 题 目 面向 **ProgrammableWeb** 网站的 **web** 服务推荐研究

作 者 姓 名 王勇

学科、专业方向 计算机技术

研 究 方 向 服务计算

指 导 教 师 胡昊 副教授

2018 年 **8** 月 **23** 日

学　　号：**MF1533054**

论文答辩日期：**2018** 年 **8** 月 **23** 日

指 导 教 师：　　（签字）

Research on Web Service Recommendation for ProgrammableWeb

by

Wang Yong

Supervised by

Associate Professor Hu Hao

A dissertation submitted to
the graduate school of Nanjing University
in partial fulfilment of the requirements for the degree of
MASTER
in
Computer Technology

Department of Computer Science and Technology
Nanjing University

August,23, 2018

南京大学研究生毕业论文中文摘要首页用纸

毕业论文题目：面向 ProgrammableWeb 网站的 web 服务推荐研究

计算机技术 专业 2015 级硕士生姓名：王勇

指导教师（姓名、职称）：胡昊 副教授

摘　要

随着面向服务架构（Service Oriented Architecture,SOA）的快速发展, 互联网上开始出现大量的 web 服务，面对 web 服务数量快速增长带来的信息爆炸的现状，服务推荐成为了服务计算领域一个热门的研究问题。伴随着 web 服务数量的飞速增长，以 ProgrammableWeb[①]网站为代表的服务平台逐渐成为了 web 服务发布和发现的主要中介，面对服务平台上大量的 web 服务，用户往往由于经验不足而难以选择，为用户推荐合适的 web 服务成为了当务之急。

当前 web 服务推荐大都关注服务的功能信息或者服务质量 (QoS)、时间等非功能信息，对于 web 服务自身丰富的边缘信息，例如功能主题信息、服务组合信息等没有给予足够的重视。同时，目前的服务推荐算法往往只致力于准确性的提高，对于推荐算法的多样性则很少关注，这将导致服务平台产生较多的长尾服务。针对目前 web 服务推荐方法存在的不足，本文基于服务平台 ProgrammableWeb, 在准确性和多样性方面作了如下工作：

1. 在准确性方面, 提出了一个基于功能主题信息和服务组合信息的 web 服务推荐算法，该算法基于 ProgrammableWeb 网站的真实数据，通过用户的隐式反馈来提取用户的兴趣偏好，通过将 web 服务的功能主题信息和服务组合信息融合进 SVD 矩阵分解模型，以预测用户兴趣评分。实验表明，与现有的协同过滤算法相比，我们的方法在准确性上取得了一定的提升。
2. 在多样性方面，针对目前服务推荐在多样性上的不足，提出了一种考虑用户偏差的改进重排名算法以提高推荐的多样性，该算法考虑了用户评分习惯差异性和用户兴趣分布差异性，并将这两种偏差结合进了阈值 T_R 的计算中，使得改进的重排序模型不再按照固定阈值 T_R, 而是通过考虑了用户差异性的个性化阈值来为用户提高多样性。实验表明，与经典的重排序算法相

[①]https://www.programmableweb.com/

比，该算法可以在尽量少的牺牲准确性的基础上，尽量多地提高 web 服务推荐的多样性。

关键词： 服务推荐；边缘信息；主题；服务组合信息；矩阵分解；多样性

南京大学研究生毕业论文英文摘要首页用纸

THESIS: Research on Web Service Recommendation for ProgrammableWeb

SPECIALIZATION: Computer Technology

POSTGRADUATE: Wang Yong

MENTOR: Associate Professor Hu Hao

Abstract

With the rapid development of SOA(Service Oriented Architecture), an increasing number of web services have been published on the Internet, which makes the information explosion. How to recommend suitable web services for users becomes a research hot spot in service computing. Currently the website named ProgrammableWeb is the main intermediary for service publishing and discovery. Thus in this paper we focus on how to recommend suitable web services for users from ProgrammableWeb.

The current web service recommendation methods mainly focus on QoS (Quality of Service) or the functional information of services, they pay little attention to rich side information of web services, such as topic information and compositional information. Furthermore, current service recommendation methods tend to focus only on the improvement of accuracy, while neglecting the recommendation diversity. This situation will lead to more long tail services generated on the service platform. To address the discussed issues, this paper has done the following work based on ProgrammableWeb:

1. To improve the accuracy of recommendation, we propose an algorithm based on topic information and compositional information of services. The algorithm is based on the real-world data from ProgrammableWeb. The implicit user feedback is used to calculate the ratings, and the topic information and compositional information of service is integrated to the SVD matrix factorization model to predict users' ratings and thus improve the accuracy of the recommendation. Experiment results show that the proposed algorithm outperforms the existing collaborative filtering algorithms on the recommendation accuracy.
2. To improve the recommendation diversity, we proposed a re-ranking algorithm that considers user bias . The algorithm considers the biases in user scoring habits and

user interest distribution, and combines these two biases into the calculation of the threshold T_R. Thus, the improved re-ranking model no longer increases the diversity according to the fixed threshold T_R. Experiments show that compared with the classical re-ranking algorithm, our methods can improve the diversity of web service recommendations while maintaining acceptable levels of recommendation accuracy.

keywords: Service Recommendation, Side Information, Topic, Compositional Information, Matrix Factorization, Diversity

目　　次

插图清单

附表清单

第一章　绪论

1.1　研究背景

随着面向服务的框架 (Servcie Oriented Architecture,SOA) 的快速发展，互联网开始涌现了大量的 web 服务。web 服务是基于 XML 和 HTTPS 的一种服务，其通信协议主要基于简单对象访问协议 (Simple Object Access Protocol,SOAP)，通过 web 服务描述语言（Web Service Descriptive Language,WSDL）进行功能描述，通过发现和集成协议 (Universal Description,Discovery and Integration,UDDI) 来发现和获得服务的元数据。web 服务就是一个应用程序，它向外界暴露出一个能够通过 web 进行调用的 API，能够用编程的方法通过 web 来调用这个应用程序。随着 web 2.0 技术的发展，一种新型的 web 服务 mashup 应用在因特网上逐渐流行，mashup 应用将多个外部的数据源和服务整合起来，利用现有的 web 服务或者数据源，组合这些资源建立一个新的 web 服务，并且新的服务的价值大于所使用服务组合的简单叠加。随着 web 服务的数量逐年上升 (如图 1-1[①])，以 ProgrammableWeb 为代表的服务平台逐渐成为 web 服务发布、查找和使用的主要中介。根据 web 服务网站 ProgrammableWeb 统计，截止到 2017 年 9 月，平台上发布 web 服务应用程序接口 (Application Programming Interface,API) 的数量已达到 14653，涵盖了社交、地图、搜索、天气等多个领域，而由这些 web 服务组合构建而成的 mashup 的数量达到了 6259。

面对大量的 web 服务，大部分用户由于缺乏足够的经验或者能力，很难快速找到自己想要的 web 服务，这就造成了信息过载[1] 问题。解决信息过载问题主要手段有两个，搜索引擎和推荐[2]。搜索引擎通过排序算法对网页进行排序，在用户进行查询时，根据用户查询的关键字，将和关键字最为相关的网页链接排序显示给用户。搜索引擎需要用户事先知道自己的需求并键入关键词，当用户不能准确描述自己需求的时候，搜索引擎便无能无力了。推荐系统的出现很好的弥补了搜索引擎这方面的不足，推荐系统不需要用户提供明确的需

[①] https://www.programmableweb.com/news/programmableweb-api-directory-eclipses-17000-api-economy-continues-surge/research/2017/03/13

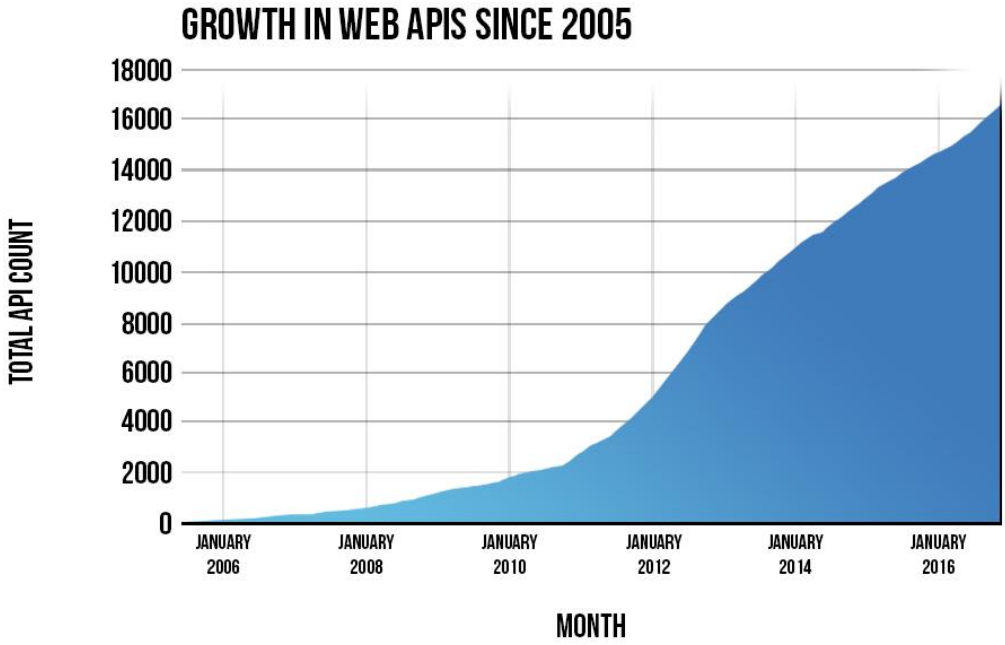

图 1-1: 2005-2016 年 web 服务数量

求，而是通过用户的历史行为给用户的兴趣建模，从而发掘出用户潜在的兴趣和需求，推荐出用户可能感兴趣的 web 服务。

1.2 研究现状

随着 web 服务数量的飞速增长，web 服务推荐问题吸引了越来越多的人研究，逐渐成为服务计算领域的一大热门。当前 web 服务推荐研究主要分为两类，为用户推荐可能感兴趣的 web 服务和为 mashup 应用推荐可以满足其功能需求的 web 服务 API，本文主要研究的问题是为用户推荐 web 服务，所以对为 mashup 推荐 web 服务 API 的研究不作阐述。目前，相关研究大多是依据用户对 web 服务的历史偏好信息或者是服务本身的 QoS 信息，通过“协同过滤”技术来进行推荐。下面将主要介绍基于用户历史信息的推荐以及基于服务质量 QoS 的推荐的相关研究现状。

基于用户的历史偏好信息的推荐，首先需要收集大量用户对 web 服务的历史评分信息，综合用户特征或者服务特征，找出相似的用户（user-based）或者相似的服务（item-based），并以此来预测用户可能感兴趣的服务，从而个性化地为用户进行推荐。文献[3] 综合了用户特征和用户评价的相似性，通过对用户情境聚类，对结果进行过滤，实现了满足用户个性化需求的推荐。文献[4] [5] 采

用了基于邻域 (neighbour-based) 的协同过滤方法，在计算用户或者服务相似度的时候，考虑了不同用户和服务的个性化的特征，最后通过最相近的 k 个用户或者服务的评分来预测评分。文献[6] 提出了一种时间感知的 web 服务推荐方法，通过对用户评分习惯随时间变化而变化，服务的热门程度随时间变化而变化，用户对服务的评分随时间变化而变化这三种场景进行建模，并结合矩阵分解方法进行评分预测。当前，这些方法主要的问题是，只考虑了用户的历史偏好信息，而忽视了 web 服务丰富的边缘信息[7]，对于 ProgrammableWeb 这样的服务平台，边缘信息往往很容易获取，通过引入边缘信息可以很好地弥补用户历史偏好的稀疏性。

基于 QoS 的服务推荐，通过收集大量的用户调用 web 服务时的 QoS 信息，例如响应时间、吞吐率、价格成本等等[8]，使用协同过滤技术预测 QoS 值，再根据预测的 QoS 值来进行推荐[9]。文献[10] 认为相似的用户倾向于接受拥有相似 QoS 值的 web 服务，并基于这一假设提出了一种基于用户的（user-based）的协同过滤方法来预测 web 服务的 QoS 值。文献[11] 使用矩阵分解技术来预测 web 服务的 QoS 值，取得了比基于邻域的协同过滤方法更好的准确度。为了提高 QoS 预测的准确性，有不少工作考虑在推荐模型中添加一些额外的信息，例如地点位置信息，用户的历史调用记录，以及时间信息等。文献[12] 使用用户的历史调用信息来推断用户的行为特征，通过历史调用信息来计算用户之间相似度，并将相似度与协同过滤方法相结合以计算 QoS 的值。文献[13] 将地理位置信息引入到了 web 服务推荐，将用户和服务按照地理位置进行划分，根据地理位置信息来寻找相似的用户或者相似的服务，从而通过协同过滤方法来预测缺失的 QoS 值。文献[14] 认为 QoS 的属性值随着时间的变化而具有波动性，文章提出了一种整合自回归积分滑动平均模型 (Autoregressive Integrated Moving Average Model,ARIMA) 和自回归条件异方差模型 (Autoregressive conditional heteroskedasticity model,GARCH) 模型的预测方法，以便能够捕捉 QoS 属性的波动性并提供准确的预测。基于 QoS 的 web 服务推荐最大的问题在于，服务的 QoS 的准确性往往和调用的地理位置有关，所以获取 QoS 信息是一件很困难的事。同时，通过 QoS 值来进行推荐，有时并不能有效地发掘用户的兴趣点，具有更高 QoS 数值的服务未必是用户感兴趣的。

1.3 本文工作

本文对当前的 web 服务推荐相关工作进行了深入调研，发现目前的相关研究主要存在三个问题，(1) 大量工作基于 QoS 进行推荐，而基于用户的历史偏好进行推荐的工作则比较少，而用户的历史偏好往往更能反映用户的兴趣，尤其是对于 ProgrammableWeb 网站的用户。（2）服务推荐中丰富的边缘信息没有能够很好的利用起来，例如功能主题信息和服务组合信息，这些边缘信息蕴含了 web 服务在功能和结构上的联系，可以弥补用户历史偏好信息的稀疏性，对推荐准确性的提升很有帮助。（3）当前的 web 服务推荐相关工作，在评估指标上往往只关注准确性，而忽视了推荐结果的多样性。针对当前工作中存在的以上问题，本文基于服务平台 ProgrammableWeb[①]网站的用户数据，在推荐的准确性和多样性方面作了如下工作：

1. 在准确性方面，提出了一个基于功能主题信息和服务组合信息的 web 服务推荐算法，该算法基于 ProgrammableWeb 网站的真实数据，通过用户的隐式反馈信息来提取用户的兴趣偏好，通过 ProgrammableWeb 网站的服务注册页面[②]来提取了服务的功能描述信息以及服务组合信息，并以此分别计算了 web 服务之间在功能主题以及组合关系上的相似度，并根据相似度获取了每个 web 服务在功能主题或者组合上的邻居 web 服务集合，将其与 SVD(Singular Value Decomposition) 矩阵分解模型相结合，以预测用户评分，提高推荐性能。
2. 在多样性方面，针对目前服务推荐在多样性上的不足，提出了一种考虑用户偏差的改进重排名算法以提高推荐的多样性，该算法考虑了用户评分习惯差异性和用户兴趣分布差异性，并将这两种偏差结合进了阈值 T_R 的计算中，使得改进的重排序模型不再按照固定阈值 T_R 而是通过加入了用户差异性的个性化阈值来为用户提高多样性。实验表明，与经典的重排序算法[15]相比，该算法可以在尽量少的牺牲准确性的基础上，尽量多地提高 web 服务推荐的多样性。
3. 使用 ProgrammableWeb 网站的真实数据作为数据集，其中包括超过 280,000 个用户服务交互记录，65,000 服务用户和 15,000 个 web 服务或 mashup, 基于该数据集进行了一系列的实验，以验证我们的方法的有效性。

[①] https://www.programmableweb.com/
[②] https://www.programmableweb.com/apis/directory

1.4 本文组织

本篇论文主要由五个章节组成，第一章作为全文的绪论，阐述了论文的研究背景，相关工作的研究现状，本文的主要工作和内容章节安排。论文的其他四章内容如下：

第二章：相关工作和技术

本章主要介绍与本文研究内容相关的工作、技术，主要包括相关推荐方法介绍、mashup 相关技术介绍、推荐的多样性提升方法介绍，为后续章节作好铺垫。

第三章：基于功能主题信息和服务组合信息的 web 服务推荐算法

本章主要介绍基于功能主题信息和服务组合信息的 web 服务推荐算法，首先给出了算法的提出动机，然后介绍了用户评分信息的计算，服务功能主题信息相似度和服务组合相似度的计算以及矩阵分解模型的融合，最后通过实验验证了我们方法的有效性。

第四章：基于重排序的服务推荐重排序算法研究

本章首先介绍了 web 服务多样性提出的动机，然后给出了用户偏差的定义和表示，并在考虑用户偏差的基础上提出了一种改进的重排序算法，最后通过实验进行验证。实验表明，本章提出的算法可以在尽量少牺牲准确度的基础上最大限度的提升服务推荐的多样性。

第五章：总结和展望

本章主要总结全文的工作，并对未来的研究工作进行展望。

第二章　相关工作和技术

2.1　推荐技术综述

随着互联网的发展，网上的信息量呈指数级增长，面对海量信息，普通用户往往会无所适从，推荐系统便是解决信息过载问题的有效手段之一，它根据用户的兴趣、行为、情境等信息，将用户可能感兴趣的内容推送给用户。自 20 世纪 90 年中期出现了第一批关于协同过滤的文章[16,17] 以来，推荐系统逐渐成为热门的研究领域。文献[18] 将推荐算法分为三类：基于协同过滤 (collaborative filtering) 的推荐、基于内容 (content-based) 的推荐以及混合了内容与协同过滤的推荐，而基于协同过滤的推荐算法又可以分为基于内存 (memory-based) 和基于模型 (model-based) 的方法。图 2-1展示了推荐算法的分类框架，下面将分别对这些方法进行介绍。

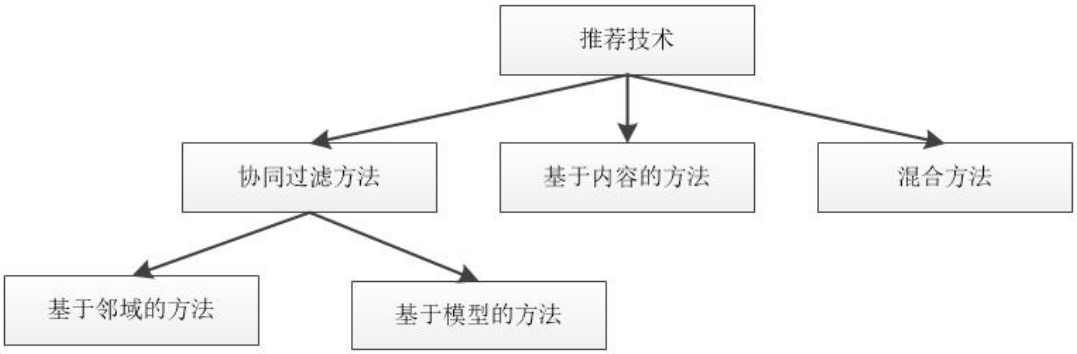

图 2-1: 推荐算法分类图

2.1.1　基于协同过滤的推荐

协同过滤（Collaborative Filtering, 简记为 CF）源自这样的想法，即志趣相投的人们通常对其他东西也有相似的兴趣，不同的用户对一些物品给予相似的评价，那么对另一些物品也会给出相似的评价。协同过滤技术通过对用户已经评价过的物品来发掘用户的兴趣偏好，再通过用户的兴趣偏好来寻找那

些偏好相似的用户，最后再把这些偏好相似用户所喜欢的东西推荐给当前用户。目前，基于协同过滤的推荐系统在工业界得到了广泛的应用，包括新闻推荐系统 GroupLens[17]、视频推荐系统 Video Recommender[16]、书本推荐系统 Amazon.com[19]。

协同过滤算法通常被分为两类，基于内存 (memory-based) 的协同过滤算法和基于模型 (model-based) 的协同过滤算法，下面对这两类算法进行介绍。

(1) 基于内存的协同过滤算法

基于内存的协同过滤算法又叫基于邻域 (neighbour-based) 的协同过滤算法, 它可以分为基于用户 (user-based) 的协同过滤和基于项目 (item-based) 的协同过滤。基于内存的协同过滤算法主要有以下三个过程：

(1.1) 建立用户-项目矩阵 (User-Item)

首先需要收集用户的各种历史偏好信息，例如用户对项目的评分信息，建立用户-项目矩阵。如图2–2所示，给定一个包含 M 个用户和 N 个项目的推荐系统，我们建立一个 $M \times N$ 的用户-项目矩阵，其中矩阵中的项 $r_{m,n}$ 表示用户 m 对项目 n 的评分值，如果 $r_{m,n}$ 值不存在，说明用户 m 从未对项目 n 评价过。

	$Item_1$	$Item_2$	…	$Item_n$
$User_1$	2	?	…	1
$User_2$	?	?	…	3
…	…	…	…	…
$User_m$	1	?	…	?

图 2–2: 用户-项目矩阵 (User-Item Matrix)

(1.2) 计算用户之间或者项目之间相似度

需要寻找那些相似的用户 (user-based) 或者相似的项目 (item-based), 这一步的关键是根据用户-项目矩阵，计算物品或项目之间的相似度。常用的相似度计算方法有余弦相似度[20] 和皮尔逊相关系数（Pearson Correlation Coefficient, 以下简称为 PCC）。

在基于用户 (user-based) 的协同过滤中，可以通过皮尔逊相关系数来定义用户 u 和用户 v 之间的相似度 $Sim(u,v)$ 如下：

$$Sim(u,v)=\frac{\sum_{i\in I_u\cap I_v}(r_{u,i}-\bar{r_u})(r_{v,i}-\bar{r_v})}{\sqrt{\sum_{i\in I_u\cap I_v}(r_{u,i}-\bar{r_u})^2}\sqrt{\sum_{i\in I_u\cap I_v}(r_{v,i}-\bar{r_v})^2}} \tag{2-1}$$

其中，I_u 和 I_v 分别表示用户 u、v 评价过的项目集合，$\bar{r_u}$、$\bar{r_v}$ 分别表示用户 u、v 对他们共同评价过的项目集合 $I_u\bigcap I_v$ 中的各项目评分的平均值。$Sim(u,v)$ 的值在-1 和 1 之间，值越大表明用户 u 和 v 之间越相似。

类似的，在基于项目的 (item-based) 的协同过滤中，需要衡量项目（item）之间的相似度，利用 PCC 来定义项目 i 和项目 j 之间的相似度 $Sim(i,j)$ 如下：

$$Sim(i,j)=\frac{\sum_{u\in U_i\cap U_j}(r_{u,i}-\bar{r_i})(r_{u,j}-\bar{r_j})}{\sqrt{\sum_{u\in U_i\cap U_j}(r_{u,i}-\bar{r_i})^2}\sqrt{\sum_{u\in U_i\cap U_j}(r_{u,j}-\bar{r_j})^2}} \tag{2-2}$$

其中，U_i 和 U_j 分别表示评论过项目 i 和项目 j 的用户集合，$\bar{r_i}$、$\bar{r_j}$ 分别表示用户集合 $U_i\bigcap U_j$ 中的各用户对项目 i 和项目 j 评分的平均值。$Sim(i,j)$ 的值在-1 和 1 之间，值越大表示项目 i 和项目 j 越相似。

在基于用户 (user-based) 的协同过滤中，也可以用余弦相似度来定义用户 u 和用户 v 之间的相似度 $Sim(u,v)$，定义如下：

$$Sim(u,v)=\cos(\vec{u},\vec{v})=\frac{\vec{u}\cdot\vec{v}}{\|\vec{u}\|\times\|\vec{v}\|}=\frac{\sum_{i\in I_u\cap I_v}r_{u,i}r_{v,i}}{\sqrt{\sum_{i\in I_u\cap I_v}r_{u,i}^2}\sqrt{\sum_{i\in I_u\cap I_v}r_{v,i}^2}} \tag{2-3}$$

其中，I_u 和 I_v 分别表示用户 u、v 评价过的项目集合，$Sim(u,v)$ 的值在 [0,1], 值越大表明用户 u 和 v 越相似。

类似的，在基于项目 (item-based) 使用余弦相似度计算项目 i 和项目 j 的相似度 $Sim(i,j)$ 如下：

$$Sim(i,j)=\cos(\vec{i},\vec{j})=\frac{\vec{i}\cdot\vec{j}}{\|\vec{i}\|\times\|\vec{j}\|}=\frac{\sum_{u\in U_i\cap U_j}r_{u,i}r_{u,j}}{\sqrt{\sum_{u\in U_i\cap U_j}r_{u,i}^2}\sqrt{\sum_{u\in U_i\cap U_j}r_{u,j}^2}} \tag{2-4}$$

其中，U_i 和 U_j 分别表示评论过项目 i 和项目 j 的用户集合，$Sim(i,j)$ 的值在 [0,1], 值越大表明项目 i 和项目 j 越相似。

(1.3) 通过最相似的若干邻居评分预测用户评分，产生推荐结果

在有了用户或者项目之间的相似度之后，需要通过若干邻居的聚合来预测评分，常用的预测如下：

$$\tilde{r}_{u,i} = \frac{\sum\limits_{u' \in U} Sim(u,u') r_{u',i}}{\sum\limits_{u' \in U} \left| Sim(u,u') \right|} \tag{2-5}$$

其中 $\tilde{r}_{u,i}$ 是用户 u 对项目 i 评分的预测值，U 表示与用户 u 最相近的若干个用户的集合。类似的，常用的基于项目的评分预测公式如下：

$$\tilde{r}_{u,i} = \frac{\sum\limits_{i' \in I} Sim(i,i') r_{u,i'}}{\sum\limits_{i' \in I} \left| Sim(i,i') \right|} \tag{2-6}$$

其中 $\tilde{r}_{u,i}$ 是用户 u 对项目 i 评分的预测值，I 表示与项目 i 最相近的若干个项目的集合。

基于内存 (memory-based) 的协同过滤方法的结果可解释很强，这在推荐系统中是一个很重要的方面，同时，算法简单易于实现，具有很强的扩展性。但是，当数据变得稀疏时，相似度的计算会不准确，算法的性能会极大地下降。

(2) 基于模型 (model-based) 的协同过滤算法

基于模型的协同过滤方法从用户的历史评分信息中学习出一个预测模型，然后通过这个模型进行评分预测。模型的学习通常需要借助机器学习的方法。常用的模型有贝叶斯网络、聚类模型、潜在语义模型、矩阵分解模型、马尔科夫模型等。下面重点介绍一下矩阵分解模型。

矩阵分解模型将用户的评分信息映射到一个低维的隐向量空间，用户对项目的评分可以表示成两个向量的内积。通常项目 i 表示为向量 $q_i \in \mathbb{R}^f$，向量 q_i 的元素表示了项目 i 具有这些隐因子的丰富程度，每个用户 u 表示为向量 $p_u \in \mathbb{R}^f$，向量 p_u 表示用户对各个隐因子的兴趣。用户的评分估计可以表示为这两个向量的乘积。

$$\hat{r}_{ui} = q_i^T p_u \tag{2-7}$$

矩阵分解所要做的，就是计算用户和项目到隐向量的映射，即找到两个矩

阵 P、Q，它们相乘之后得到的值，和待分解矩阵中有值的位置中的值尽可能地接近，这样一来分解出来的两个矩阵相乘就尽可能地还原了待分解矩阵，因为有值的地方，值都相差得尽可能地小，那么没有值的地方，通过这样的方式算出的值，也就比较符合趋势。对每个用户 u 和项目 i，找到这样的 p_u 和 q_i, 满足如下目标函数：

$$\min_{q,p} \sum_{(u,i)\in\mathcal{R}} (r_{ui} - q_i^T p_u)^2 \tag{2-8}$$

其中 $\mathcal{R}$ 是用户对项目的历史评分集合。

通常，为了防止过拟合，会在上式中加入正则项，加入正则项后的目标函数如下：

$$\min_{q,p} \sum_{(u,i)\in\mathcal{R}} (r_{ui} - q_i^T p_u)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2) \tag{2-9}$$

其中 λ 是正则项系数，用于控制正则化的程度。

求解上面的目标函数，通常采用随机梯度下降的方法。首先对于每个用户-项目评分，计算预测误差 $e_{u,i}$

$$e_{ui} = r_{ui} - q_i^T p_u \tag{2-10}$$

然后按照梯度下降的方向更新用户和项目的隐向量，直到算法收敛或者到达给定的迭代数：

$$q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i) \tag{2-11}$$

$$p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot_u) \tag{2-12}$$

其中 γ 是学习率，用于控制算法收敛的速度。

对于任意用户 u 以及任意用户 i，我们即可通过公式 (2-7) 计算得到的 p_u 和 q_i 来估算用户的评分 $\hat{r}_{ui}$。

基于模型的协同过滤算法相比基于内存的协同过滤算法，能够更好地解决原始矩阵的稀疏性问题，特别在处理大型稀疏数据集时。但是模型地训练通常需要大量时间，并且对于推荐项目的可解释性不如基于内存的协同过滤。

2.1.2 基于内容的推荐

基于内容的推荐算法的思想很简单，就是根据用户过去喜欢的项目 (item), 为用户推荐和他们过去喜欢的物品相似的物品，基于内容的推荐不依赖于用户对项目的评价，而是更多地从项目 (item) 的内容和用户的兴趣进行匹配，再推荐那些项目内容与用户兴趣相似度高的其他项目。

基于内容的推荐系统通常包含三个部分：用户特征提取、项目特征提取、生成推荐列表，如图2-3所示。用户特征提取，即利用一个用户过去喜欢（及不喜欢）的项目的特征数据，来学习出此用户的喜好特征（profile）。项目特征提取，即为每个项目抽取一些特征来表示这个项目。生成推荐列表，即通过将用户喜好特征 (profile) 与候选项目的特征进行比较，为此用户选择一组相关性最大的项目。

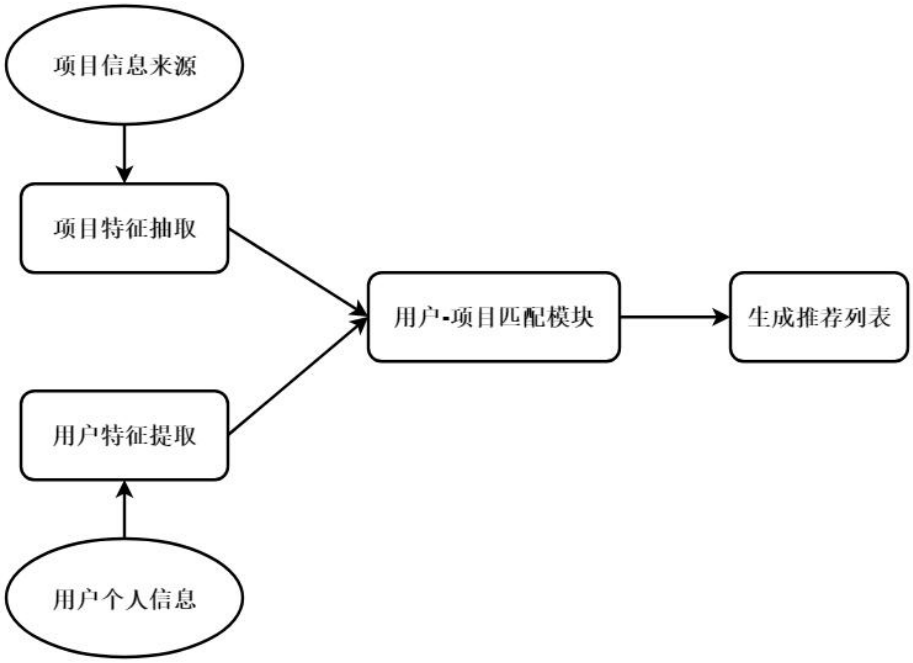

图 2-3: 基于内容的推荐系统流程

基于内容的推荐过程简单解释性强，推荐的结果容易被人接受，对于没有任何用户评分的新项目，也可以推荐给用户。但是基于内容的推荐也有一定的局限性：（1）可分析的内容有限，无法区分具有相同特征的不同项的质量。（2）推荐的项目没有很好的新颖性，不能为用户发现一些新的潜在的感兴趣的项目。

表 2-1: 符号说明表

符号	说明
U	用户集
I	项目集
M	用户的总数
N	项目的总数
T	测试集
r_{ui}	用户 u 对项目 i 的真实评分
$\hat{r}_{ui}$	用户 u 对项目 i 的预测评分
L_u	用户 u 的推荐列表
B_u	测试集中用户 u 给正反馈评分的物品
$N(u)$	训练集中用户 u 给正反馈评分的物品
$N(i)$	训练集中给物品 i 正反馈评分的用户
p_i	项目 i 的度，即评价过项目 i 的用户数量
$neighbor(u)$	用户 u 的邻居集合

2.1.3 推荐算法的评估指标

推荐算法的研究中，如何评价一个推荐算法的好坏是一个不可忽视的问题。当前很多工作对推荐系统的评估指标进行了研究，文献[21,22] 在对推荐系统的评估方法进行了总结，它将评估方法分为在线评估、用户调查和离线评估。由于用户调查和在线评估都需要大量用户的参与，评估代价较高，目前大多数研究主要采用的都是离线评估。文献[22] 对离线评价中常用的指标进行了总结，包括准确性指标和非准确性指标，准确性指标包括准确率、召回率，F-Measure、均方根误差 (RMSE) 等，非准确性指标包括多样性、新颖度、覆盖率等指标，下面将对重点介绍准确性指标和多样性指标。为了方便描述，表 2-1给出了后面可能用到的符号说明。

2.1.3.1 准确性指标

预测的准确性是推荐系统研究中讨论最多的属性，推荐系统有一个最基本的假设，即用户更喜欢推荐更精确的系统，因此大量国内外研究人员都致力于寻找更精确的预测算法。通常算法的准确性指标分为两大类，评分预测准确性

和使用预测准确性。

(1) 评分预测准确性

在一些推荐系统中，我们的核心任务就是预测用户对某个物品的评分 (如 1 到 5 分)。在这种情况下，我们可以通过评分预测准确性指标来评估推荐系统。在评分预测准确性指标中，均方根误差 (RMSE) 和平均绝对误差 (MAE) 是最常用的度量指标，它们体现了真实评分和预测评分之间的差异性大小，通常 RMSE 和 MAE 的值越小，预测的准确性越高。

均方根误差 (RMSE) 定义如下：

$$RMSE = \sqrt{\frac{1}{|T|}\sum_{(u,i)\in T}(r_{ui} - \hat{r}_{ui})^2} \tag{2-13}$$

平均绝对误差 (MAE) 定义如下：

$$MAE = \sqrt{\frac{1}{|T|}\sum_{(u,i)\in T}|r_{ui} - \hat{r}_{ui}|} \tag{2-14}$$

（2）使用预测准确性

在很多推荐系统中，推荐系统并不预测用户的偏好 (如电影评分)，而是预测用户可能会喜欢的项目，在这种情况下，我们并不关心推荐系统是否正确地预测了用户的评分，而更关心推荐的项目是否被添加到了用户喜欢的列表中。我们可以通过使用准确性指标来评估这类推荐系统。常用的使用预测准确性指标包括精度 (Precision)、召回率 (Recall)、F1-Measure 等。

精度 (Precision) 表示推荐给用户的项目中，用户喜欢的项目所占的比例，单个用户 u 推荐精度如下：

$$Precision(L_u) = \frac{|L_u \bigcap B_u|}{|L_u|} \tag{2-15}$$

召回率 (Recall) 表示用户喜欢的项目中，有多少比例是推荐系统推荐给用户的，召回率通常表示如下：

$$Precision(L_u) = \frac{|L_u \bigcap B_u|}{|B_u|} \tag{2-16}$$

F1-Measure 又称为 F1-score，是信息检索领域的一个常用标准。F1-Measure 是精度和召回率的调和平均，F1-Measure 较高表明算法越有效。F1-Measure 通常表示如下：

$$F1 = \frac{2PR}{P+R} \tag{2-17}$$

2.1.3.2 多样性指标

推荐系统的评估中，只考虑准确性指标有时是不够的，“准确”的推荐有时并不能让用户满意，有时候还必须让推荐的项目更加多种多样，因此在评估推荐系统时，讨论推荐的多样性也是有必要的。推荐的多样性指标主要分为两种，分别是个体多样性和总体多样性。个体多样性是指对单个用户而言推荐的项目是多种多样的，而总体多样性是指对于全体用户，系统推荐的项目是多样的。

个体多样性[2] 通常用推荐给用户的项目中项目的平均相似度来表示，平均相似度越高，多样性往往越低。通常推荐的多样性表示如下：

$$Diversity(L_u) = \frac{2\sum_{i,j\in L_u, i\neq j} similarity(i,j)}{|L_u|(|L_u|-1)} \tag{2-18}$$

其中 similarity(i,j) 表示项目 i 和项目 j 之间的相似度，相似度的计算可以参考上一节。

文献[23] 通过新颖度来衡量推荐算法的多样性，新颖度表示推荐给用户那些该用户从未听说过的项目的能力，在新颖度评估时通常基于这样的一个假设，即流行度越低的物品往往对用户来说是越新颖的，所以通常通过平均流行度来表示推荐的新颖度，表示如下：

$$Novelty(L_u) = \frac{2\sum_{i\in L_u} p(i)}{|L_u|} \tag{2-19}$$

整个系统的新颖度可以表示为：

$$Novelty = \frac{1}{M}\sum_{u\in U} Novelty(L_u) \tag{2-20}$$

总体多样性方面，通常使用推荐给所有用户的不同项目总数来衡量，称之为 diversity-in-top-N, 可以表示如下：

$$diversity - in - top - N = \left|\bigcup_{u\in U} L_u\right| \tag{2-21}$$

2.2 mashup 相关技术

本小节主要从两个方面介绍 mashup 相关技术，包括 mashup 的基本概念以及常见的 mashup 应用类型，为后续引入服务组合信息作铺垫。

2.2.1 mashup 基本概念

mashup 一词早期在英语中主要出现在音乐领域，指人们将一首歌曲的音频与另一首歌曲的音轨无缝地结合在一起，形成一首新的歌曲。随着 web 2.0 的发展，mashup 一词逐渐成为了 web 服务领域的一个术语。web 服务领域的 mashup 通常指一种 web 应用程序，将两种或者两种以上的数据源或者 web 应用集成到一起，并且形成的新的 web 应用的价值大于所使用的服务组合的简单叠加。mashup 应用的出现大大缩短了 web 服务的开发周期，并且提升了服务的可扩展性[24]，近年来逐渐成为了一种流行的应用形式，并得到了多个平台的支持，例如 Google Mashup Editor[①],IBM Mashup Center[②]以及 Yahoo Pipe[③]。根据 ProgrammableWeb 网站的统计，截止 2017 年 9 月，已经有超过 6379 个 mashup 应用，并且数量还在继续增长。

图 2-4是一个 mashup 的例子。这是一个叫做 WunderWalk[④]的 mashup 应用，它的功能是让用户能够搜索城市中的景点，为自己规划个性化地出游路线。WunderWalk 由两个 web 服务 API 组合而成，分别是 Google Maps API[⑤]和 Foursquare API[⑥]。Google Maps API 提供了基本的地图功能，例如位置搜索和标记特定位置。Foursquare API 提供了关于社交的功能，比如评论、与朋友分享图片。通过调用这两种类型的 API，WunderWalk 就能够为用户提供新的强大功能。例如查看和标记旅游景点的位置，记录或分享与位置相关的图片，分享出行的路线等。

mashup 应用的构建不需要用户有特别高的 web 应用开发能力，也不需要用户有非常强的架构能力。只需要用户熟悉所需要使用的 API 的使用方式以及 web 应用的工作方式，即可进行 mashup 应用的开发。目前几乎所有的服务提供商都会提供 web 服务 API 的说明文档，用户只需要阅读并了解相应的使用方式

[①] https://developers.google.com/mashup-editor/
[②] http://www-10.lotus.com/ldd/mashupswiki.nsf
[③] https://pipes.yahoo.com/pipes/
[④] http://www.wunderwalk.com/
[⑤] https://maps.googleapis.com/maps/api/js
[⑥] https://developer.foursquare.com/

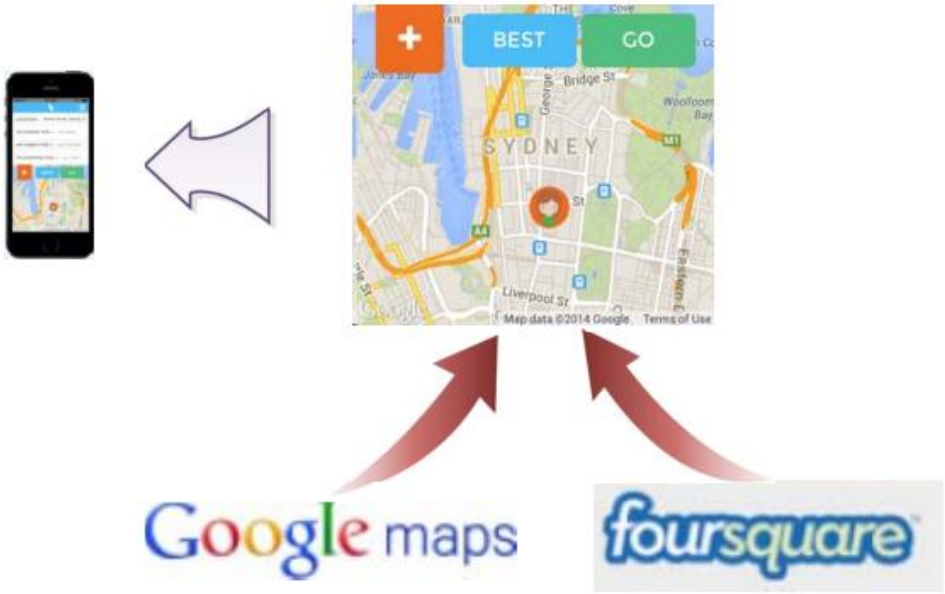

图 2-4: 一个 mashup 应用的例子[50]

后，就可以搭建开发自己的 mashup 应用，大大缩减了 web 应用的开发周期以及开发难度。

2.2.2 常见的 mashup 应用类型

ProgrammableWeb 网站上发布了大量的 mashup 应用，主要包括地图类、购物类、视频图片类、搜索类、旅游类以及新闻类。对于每种类型，我们通过下表 3-3给出 mashup 示例，并简要介绍其功能。

表 2-2: 典型 mashup 应用举例

类别	mashup 示例	功能
地图类	College Toolkit	将高校周边的设施服务显示在地图上
购物类	LivePlasma	调用 Amazon API 显示商品、品牌、店铺之间的关系
视频图片类	Actually	可以查看和分享来自 flickr 和 instagram 上的照片

旅游类	Vatsim Flight Tracker	用 Vatsim API 和 Google Maps 将用户过去一年的行程进行可视化
搜索类	Google vs Yahoo Visualized	将 Google 和 Yahoo 的搜索结果进行可视化对比
新闻类	BBC News Map	将各地的天气、交通、新闻与电子地图相合

2.3 基于重排名的多样性提高算法

推荐的多样性由 Smyth Barry 等人在 2001 年[25] 首次提出，多样性表明一个推荐系统应该包含多种多样的项目，因为用户的兴趣是广泛的，只推荐用户热门的项目或者单一种类的项目虽然有可能取得较高的推荐准确性，但是往往不能令用户满意[26]，因为有时盲目崇拜精确性指标可能会导致用户得到一些信息量为 0 的” 精准推荐”，使得用户视野变得越来越窄[27]。

推荐系统通常包含评分预测和推荐列表生成两大步骤，评分预测即根据用户的历史偏好信息预测用户对项目的评分，而列表生成则是根据预测评分，选择评分最高的 N 个项目 (top-N) 推荐给用户。因此，多样性提高算法也可以分为两类：第一类是在评分预测阶段进行[28–30]，通过考虑各种综合因素来产生多样化的预测评分。第二类是在列表生成阶段进行[31,32]，这种方法在已有的评分预测模型的基础上，通过对推荐列表按照一定的原则来重新排序，从而提高推荐列表的多样性。与第一类方法相比，第二类方法具有更好的扩展性，能够和任何具有评分预测功能的推荐算法相结合。下面我们将主要介绍重排名算法。

推荐系统通过某种推荐算法为每个用户预测出未知的评分后，会将预测评分按照从高到低排序，从而选出评分最高的的 N 个项目推荐给用户。这种将预测评分降序排列的排序方法被称为标准排名法[15]。标准排名法总是将具有最高预测评分的项目推荐给用户，虽然这样具有较高的推荐准确性，却不利于推荐多样性提高，因为标准排名法更容易将那些较为热门的项目推荐给用户。为了解决以上问题，文献[31] 提出了重排名方法 ((ReRanking Approach)，通过一个阈值 T_R 来平衡推荐结果的准确性和多样性。对于用户 i 和参数 T_R, 重排名函数

$rank_x(i, T_R)$ 如下式所示：

$$rank_x(i, T_R) = \begin{cases} rank_x(i), if\ R^*(u,i) \in [T_R, T_{max}] \\ \alpha_u + rank_{standard}(i), if\ R^*(u,i) \in [T_H, T_R] \end{cases} \tag{2-22}$$

其中，$I_u^*(T_R) = \{R^*(u,i) \geq T_R\}$，$\alpha_u = \max_{i \in I_u^*(T_R)} rank_x(i)$，$rank_{standard}(i)$ 按照标准排序法排序，$rank_x(i)$ 表示按照启发式的排名方法排序，T_R 为阈值，T_H 是判定项目是否与该用户有关的阈值，例如在 5 分制的评分中通常取经验值 3.5。

重排序法设置了一个阈值 T_R，当预测评分大于阈值 T_R 时，使用启发式的排名方法 $rank_x$ 进行排序；当预测评分小于阈值 T_R 时，使用标准排名法进行排序。α_u 的值表示的落在区间 $[T_R, T_{max}]$ 的项目的排名的最大值，其存在可以确保预测评分落在区间 $[T_R, T_{max}]$ 的项目排名比预测评分落在区间 $[T_H, T_R]$ 的项目更靠前 (拥有更小的名次，更高的推荐优先级)。当 T_R 越接近 T_{max}，预测评分高的项目越有可能被推荐，推荐的准确度也将越大；当 T_R 越接近 T_H, 重排名函数越接近启发式的排名方法，将会导致准确性下降，但是多样性提升。

文献[31] 给出了三种启发式的排名方法，分别是按照项目流行度排名，逆序排名以及随机排名。以项目流行度作为启发式的排名方法为例，重排名算法工作原理如图 2-5所示，图 2-5(a) 是 按照标准排序法，对每个项目按照从高到低

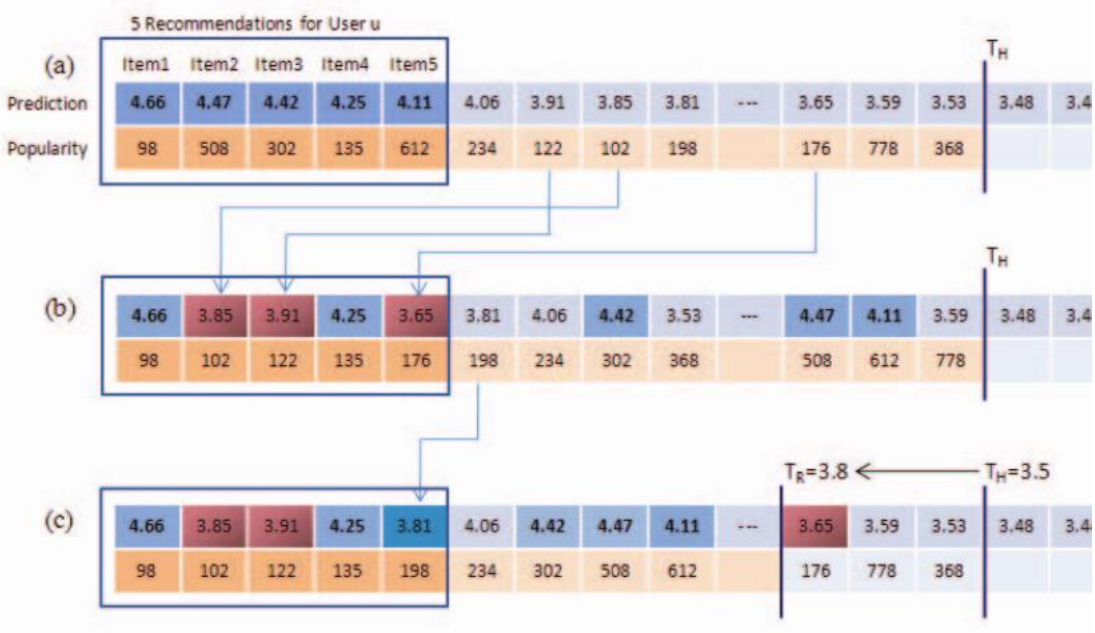

图 2-5: 重排序算法原理示例[31]

依次排序，图 2-5(b) 是基于项目流行度的排序方法，将与用户比较相关的项目

(预测评分大于 T_H) 按照流行度从低到高排序。图 2-5(c) 设置阀值 T_R 为 3.8, 此时原预测评分为 3.65 的项目因为小于阈值而排名下降，而原来预测评分为 3.81 的项目因为流行度较低而出现在了用户的候选推荐列表中，此时推荐的准确性有所下降，但是多样性有所提升。因此通过调整阈值 T_R，可以平衡推荐的准确性和多样性。

2.4 小结

本章首先介绍了推荐系统的相关算法，主要包括基于协同过滤的推荐、基于内容的推荐以及推荐系统的评估方式，然后介绍了 mashup 相关知识，最后对基于重排名的多样性提高算法进行了介绍，为后续章节的展开作铺垫。下面的章节将着重介绍我们的融合了主题信息和服务组合信息的 web 服务推荐算法。

第三章　融合主题信息和服务组合信息的 web 服务推荐

3.1　问题背景

随着 web 服务数量的增长，以 ProgrammableWeb 为代表的服务平台逐渐取代了传统的统一描述、发现和集成协议 (Universal Description,Discovery and Integration,UDDI) 服务注册中心，成为 web 服务发布、查找和使用的主要中介[33]。截止到 2017 年 9 月，平台上发布 web 服务应用程序接口 (Application Programming Interface,API) 的数量已达到 14653，涵盖了社交、地图、搜索、天气等多个领域，而由这些 web 服务组合构建而成的 mashup 的数量达到了 6259。经过 10 多年的发展，ProgrammableWeb 已经成为了互联网上最大的 web 服务公共平台之一。

面对如此多的 web 服务，网站的用户往往会由于缺乏足够的经验或者能力，无法找到自己可能喜欢的 web 服务。那么如何为网站的用户推荐他们可能喜欢的 web 服务呢？当前的许多 web 服务推荐方面的研究，大多数是基于服务质量（QoS）的，然而对于 ProgrammableWeb 网站的用户来说，他们通常不关心 web 服务的 QoS 值，他们更希望能够通过推荐发掘出他们的潜在兴趣，找到那些他们可能喜欢的 web 服务。如何发掘出用户的潜在兴趣，并结合 web 服务自身的一些边缘信息 (side information), 来为用户推荐他们可能会喜欢的 web 服务成为了一大问题。

为此，本章提出了一种融合了主题信息和服务组合信息的 web 服务推荐算法，以解决以上问题。我们将在后续的小节对算法的过程进行详细介绍。

3.2　用户评分信息获取

为了发掘用户的潜在兴趣，需要从用户的历史的显式或者隐式的历史记录中提取用户的偏好。然而在 ProgrammableWeb 中，用户的显式的评分信息很难

获取，并且很多用户往往并不会给自己喜欢的服务打一个显式的评分。为此，我们采用文献[6] 中的方法，通过用户的隐式反馈信息来提取用户的偏好[34]。对于 ProgrammableWeb 网站的用户来说，当他们对某一个 web 服务感兴趣的时候，他们可以选择“track this API”, 然后这个 web 服务就会被添加到该用户的观察列表 (watchlist①) 中, 图 3-1②是一个用户的观察列表的例子，通过每个用户的观察列表，我们就可以知道用户对什么样的 web 服务感兴趣，从而提取用户的兴趣偏好。

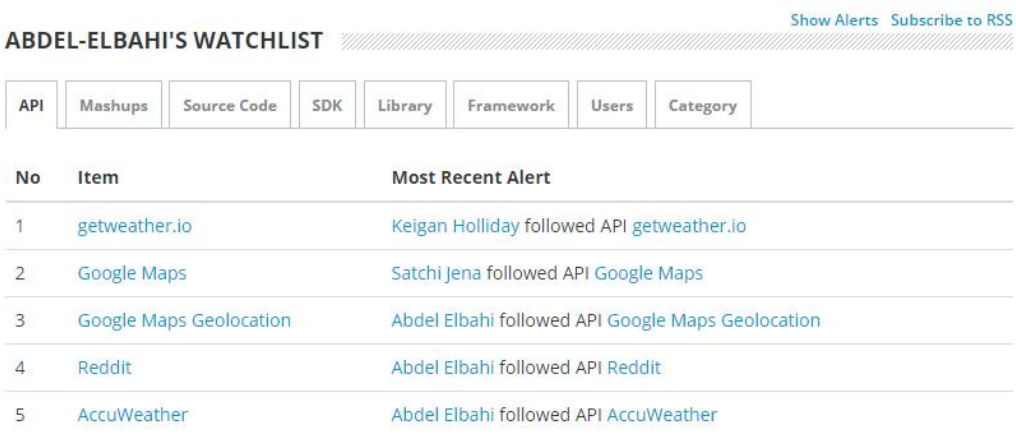

图 3-1: 一个用户的 watchlist 列表

近一步的，我们发现当用户的观察列表中的 web 服务每次更新或者被其他用户关注以后，系统都会在该用户的观察列表中自动产生一条记录，一个 web 服务被大量用户关注，说明该 web 服务很受欢迎，应该具有更高的推荐的优先级，同时，我们考察一个 web 服务的更新频率和该服务的受欢迎程度之间的关系，即是否经常更新的 web 服务更受欢迎。我们以 Google、Twitter、Yahoo 和 Amazon 这四个知名服务提供商为例，我们统计了数据集中这四个公司更新频繁的服务和更新不频繁的服务的平均用户人数，如表 3-1所示。以 Google 为例，Google 提供的 web 服务中，有 18 个更新较为频繁的服务 (更新超过 5 次)，有 103 个更新不太频繁的服务，对于每个频繁更新的服务，它拥有的平均用户数量是 72.8，即有 72.8 个用户将该服务放入推荐列表中，而对于更新不频繁的服务，拥有的平均用户数量是 22.75。对于 Amazon, 频繁更新的服务和不频繁更新的服务的平均用户数差距更大。这表明经常更新的服务更有可能受到用户的欢迎。 因此我们认为，如果用户的观察列表中某一个服务被关注或者更新的

①https://www.programmableweb.com/profile/userName/alerts#watchlist/
②https://www.programmableweb.com/

表 3-1: 服务更新频率与用户数量比较表

服务提供商	频繁更新的服务		不频繁更新的服务	
	服务数量	平均用户数	服务数量	平均用户数
Google	18	72.8	103	22.75
Twitter	2	34	5	29
Yahoo	5	48.4	50	38.14
Amazon	2	144.5	31	6.9

记录越多，这个服务就会更受到用户的欢迎，它应该有更高的优先级被推荐给用户，即应该具有更高的用户评分。

基于以上的阐述，我们结合用户的观察列表，给出用户的评分计算方式如下：

$$r_{us} = \frac{freq(s, Watchlist_u) + follow(s, Watchlist_u)}{Watchlist_u} \tag{3-1}$$

其中 r_{us} 是用户 u 对服务 s 的评分，$Watchlist_u$ 是用户 u 的观察列表中记录总条数，$freq(s, Watchlist_u)$ 是用户 u 观察列表中服务 s 的更新记录条数，$follow(s, Watchlist_u)$ 是用户 u 观察列表中服务 s 的被用户关注记录的条数。

公式 (3-1) 表明, 当一个用户的观察列表的某个服务，更新越频繁或者关注的用户越多，该服务获得的用户评分就越高，获得的推荐优先级也越高。

我们通过公式 (3-1) 量化了用户的兴趣偏好，为接下来的融合了主题信息和服务组合信息的 web 服务推荐算法的提出作铺垫。

3.3 主题信息提取

在有了用户的评分信息之后，我们便可以基于用户的历史评分信息，使用协同过滤方法进行推荐，然而，如图3-2所示，我们对数据集中的用户观察列表中的 web 服务数量进行了统计，发现大约有百分之六十的用户观察列表中的 web 服务数量不足 10 个，而观察列表中数量大于 10 个的用户总数加起来只有大约百分之四十，这表明在 web 服务推荐中，只根据用户的观察列表进行推荐会面临着数据稀疏的问题，用户评价过的 web 服务与 web 服务的总数相比太少。文献[35] 表明利用丰富的边界信息 (side information) 可以对用户-项目（User-Item）矩阵进行补充，对推荐的效果有很大的帮助。为此，我们考虑充分利各种 web 服务的边界信息来提高推荐的性能。 大量基于内容[36–38] 的推荐

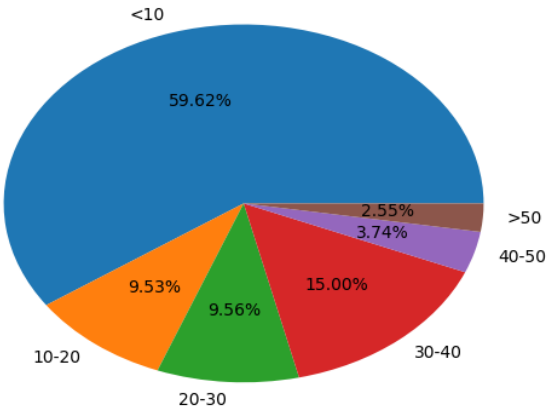

图 3-2: 用户观察列表中 web 服务数量统计

研究表明，用户喜欢某一个项目，那么该用户有很大可能会喜欢跟该项目内容上相似的项目。为此，我们考虑将 web 服务的功能信息融入到推荐中，我们认为，当一个用户喜欢某一个 web 服务，那么他很可能对和该 web 有类似功能 web 服务也感兴趣。在服务功能识别方面，最初是基于 web 服务描述语言（Web Service Descriptive Language,WSDL）进行，而随着基于 RESTful 的服务架构的兴起，第三方不再能获取丰富详细的 WSDL 文件，越来越多的研究工作考虑使用服务的描述文本来识别服务的功能[33]。

如图 3-3[①]所示是一个 web 服务在 ProgrammableWeb 网站中的界面，我们可以在该界面看到这个服务的名称（Name）、标签 (tags)、描述 (description) 等信息。web 服务的描述信息描述了这个 web 服务的主要功能，通过比较 web 服务描述信息之间的语义上的相关度，我们便可以衡量两个 web 服务的功能上是否相似。那么如何评估两个 web 服务在语义上的相关度呢，我们考虑通过 LDA 主题模型 (Latent Dirichlet Allocation) 来获取每个 web 服务的主题 (topic) 分布，从而根据主题来计算两个 web 服务在语义上的相似程度（本文称为主题相似度）。

[①]https://www.programmableweb.com/api/youtube

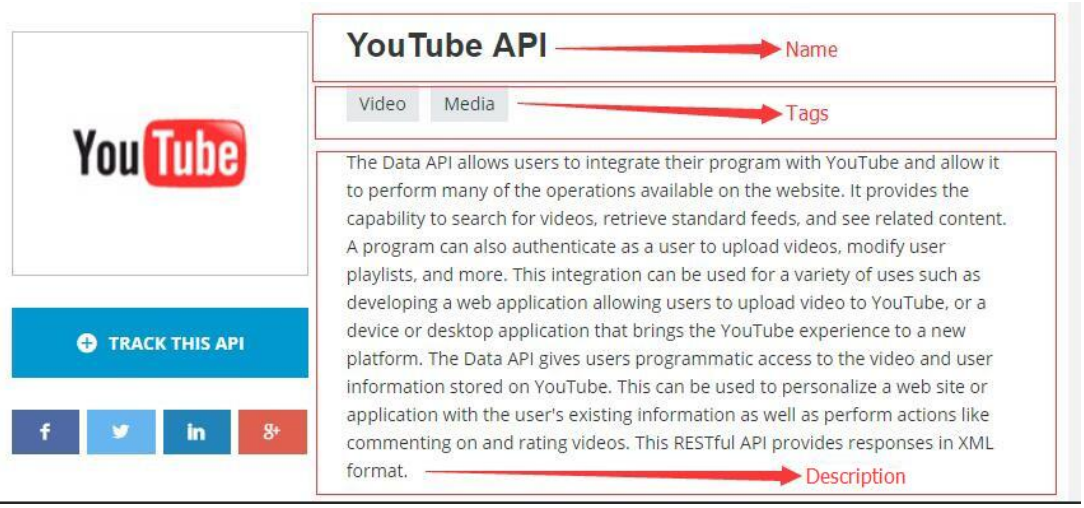

图 3-3: Youtube 在 ProgrammableWeb 中的界面

3.3.1 LDA 主题模型

在介绍 LDA 主题模型前，我们先介绍一下什么是主题。主题是对文本内容的一个提炼，当我们读一篇文章时，可能会想概括一下一下这篇文章主要讲什么，此时我们可能会回答，80% 在讲政治，15% 在讲娱乐，其余在讲废话，那么我们就可以从这篇文章提炼出三个主题：政治、娱乐、废话。那么我们又是怎么提炼出一篇文章的主题呢？我们通常是因为一篇文章出现了某些特定的词，而这些词往往和某个特定的主题想关联，所以通过这些词就可以确定某一篇文章的主题。以图3-4为例，我们在文本中出现了诸如”Film”,”Music”,”Show” 等关键词，所以我们通过这些词归纳出”Arts” 这一主题。同理，我们根据文本中的其他词，将文本归纳为了四个主题，分别”Arts”,”Budgets”,”Children”,”Education”，每个主题下会有能提现该主题特征的关键词。

LDA 主题模型由 David M.Blei 和 Michael I.Jordan 在 2003 年提出[39]，是一种的非监督的概率生成模型，可以用来发现文档的隐含主题。LDA 认为一篇文章的每个词都是由一个生成过程得到的，即文章以一定的概率选择了某一个主题，某个主题又以一定的概率选择了某一个词。对于语料库中的文档，LDA 模型生成过程如下：

(1) 从狄利克雷分布 α 中取样生成文档 i 的主题分布 θ_i

(2) 从主题的多项式分布 θ_i 中取样生成文档 i 第 j 个词的主题 $z_{i,j}$

(3) 从狄利克雷分布 β 中取样生成主题 $z_{i,j}$ 的词语分布 $\phi_{z_{i,j}}$

"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

图 3-4: 一个主题和关键词的例子[39]

(4) 从词语的多项式分布 $\phi_{z_{i,j}}$ 中采样最终生成词语 $w_{i,j}$

上述生成过程可以用图3-5表示：

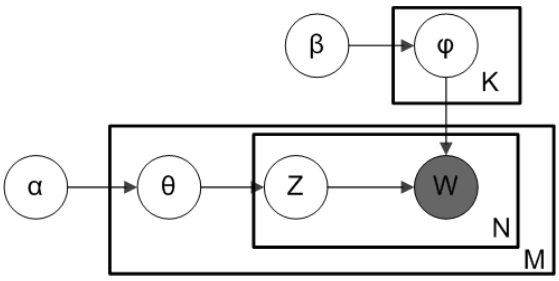

图 3-5: LDA 模型生成概率图[39]

该模型有两个分布参数需要学习：一个是文档-主题 (Document-Topic) 分布 θ, 一个是主题-词 (Topic-Word) 分布 ϕ。这两个参数可以通过 Gibbs 采样来估计得到，这里不再赘述，详细过程可参考文献[40], 最终，θ 和 ϕ 的估计公式如下：

$$\phi_{k,t} = (n_k^{(t)} + \beta_t)/(n_k + \beta_t) \tag{3-2}$$

$$\theta_{m,k} = (n_m^{(k)} + \alpha_k)/(n_m + \alpha_k) \tag{3-3}$$

其中，$n_k^{(t)}$ 表示词汇 t 属于主题 k 的次数，n_k 主题 k 中的词汇总频数，$n_m^{(k)}$ 表示第 m 篇文章中主题 k 出现的次数，n_m 表示第 m 篇文章的主题出现总频数。

3.3.2 文本预处理

在进行 LDA 模型训练之前，我们需要对每个 web 服务的描述文档进行文本预处理，预处理的过程如下：

(1) 内容清理

对文档中的句子进行清理，去掉数字、标点以及非字母符号, 并将字母统一为小写

(2) 拼写检查

对文档中的单词的拼写进行检查，将拼写错误的单词进行修正

(3) 去停用词 (Stop Words)

停用词是指一些出现次数很多，但是并没有什么实际含义的词，例如”the”、”is”、”and”、”what” 等，我们需要根据停用词表，将这些停用词进行删除。

(4) 词干化

对于英文单词通常需要词干化处理，主要包括将名词复数变为单数，将动词的过去时、进行时等时态变为最基本的形态，例如 needs->need,getting->get。

下面给出一个文本预处理的例子，web 服务 Google Buzz 的描述信息内容如下：

Google Buzz is a new way to share updates, photos, videos and more, and start conversations about the things you find interesting. The core platform extends from Google Gmail

预处理后，得到如下文本:

googl buzz share updat photo video start convers core platform extend googl gmail

3.3.3 计算主题相似度

在对 web 服务的描述文档进行文本预处理后，每个 web 服务的描述文档就可以看成一组词向量，即对 web 服务 s 来说，它的描述文档就是 s = <word1,word2,word3...>。

如图 3–6我们对所有 web 服务处理后的描述文档进行 LDA 主题模型训练，可以最终得到两个分布，即文档-主题分布 (Document-Topic) 和主题-词分布 (Topic-Word), 此处我们主要用到文档主题分布，由于一个描述文档对应于一个

web 服务，所以此处我们又称为服务-主题分布 (Service-Topic)。 根据服务-主题

图 3-6: 服务-主题分布生成示例图

分布，对于 web 服务 s_i 都可以用一个主题向量 $d_i =< w_{i1}, w_{i2}, w_{i3}...w_{in} >$ 表示。w_{ik} 表示服务 i 在主题 k 下的概率。本文采用余弦相似度来计算主题向量之间的相似性，主题相似度计算如下：

$$Sim_T(s_i, s_j) = \cos(\vec{d_i}, \vec{d_j}) = \frac{\vec{d_i} \cdot \vec{d_j}}{\left\|\vec{d_i}\right\| \times \left\|\vec{d_j}\right\|} = \frac{\sum_{k=1}^{N} w_{i,k} w_{j,k}}{\sqrt{\sum_{k=1}^{N} w_{i,k}^2} \sqrt{\sum_{k=1}^{N} w_{j,k}^2}} \tag{3-4}$$

其中 $Sim_T(s_i, s_j)$ 是服务 s_i 和服务 s_j 的主题相似度，N 为主题的个数。

3.4 组合信息提取

在 1.2 节中我们对 web 服务推荐的相关工作进行了介绍，我们发现，在这些工作中，mashup 应用所蕴含的服务组合信息没有引起足够的重视。什么是服务组合信息？我们在 2.3 节中给出了一个名为 WunderWalk 的 mashup 应用的例子，WunderWalk 由两个 web 服务 API 构成，分别是 Google Maps 和 fourSquare。WunderWalk 由 Google Maps 和 foursquare 组合而成，这就是这个 mashup 所蕴含的服务组合信息。ProgrammableWeb 网站的用户主要分为服务的开发者和服务的使用者，作为服务开发者，当我们对 WunderWalk 感兴趣的时候，我们是不是也有很大可能会对 Google Maps 以及 foursqure 保持一定的兴趣呢？同样的，当我们对 Google Maps 感兴趣的时候，我们是不是也会对那些使

用 Google Maps API 构成的 mashup 应用感兴趣呢？下一节，我们将从三个角度探讨引入组合信息的意义。

3.4.1 引入组合信息的意义

我们认为，引入服务组合信息有以下三个优势：

(1) mashup 中蕴含的服务组合关系，包含着服务之间的隐式联系

以图 3-7为例，m1,m2,m3 是三个 mashup，s1、s2、s3、s4 是四个原子服务，其中 m1 由 s1,s2,s4 组合而成，那么当一个用户关注了 s1 和 s2，那么将 s4 推荐给该用户将是一个明智的选择，同样的，m1 也可以推荐给该用户，因为该用户关注的 s1 和 s2 是和 m1 在组合关系上相关联的原子服务。当一个用户关注了 mashup 应用 m1, 那么该用户有很大可能也会对 m3 感兴趣，因为 m1 和 m3 都调用一些公共的原子服务 (s2 和 s4)。以 WunderWalk 为例，在关注 WunderWalk 的人当中，约有 25% 的人也关注了 foursquare, 有 12.5% 的人关注了 Google Maps, 同时，在关注 WunderWalk 的人当中，约有 20% 的人关注了一个叫 EVMapper 的 mashup 应用，该 mashup 由 Google Maps 和 Eventful 组合而成。

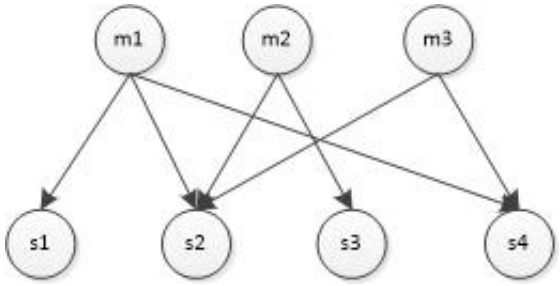

图 3-7: mashup-service 调用关系图

(2) 对主题信息的补充

主题信息的引入，旨在能够为用户推荐那些功能上相似的服务，而引入服务组合信息，可以为推荐增加新的维度，帮助用户发现那些可能在功能上不相似但是具有服务组合关系的 web 服务，比如，当一个用户关注了 foursquare，那么可以根据组合关系为用户推荐 Google Maps，尽管这两个 web 服务没有功能上的联系。同时，我们发现在 ProgrammableWeb 网站上，mashup 的功能描述信息通常较短，如图 3-8[①]所示，这么短的描述信息

[①]https://www.programmableweb.com/mashup/bilka-app

不能很好的刻画文档的主题分布，此时，服务组合信息的引入可以很好的弥补这个不足，让这些 mashup 可以因为在组合上的潜在关系而有机会被推荐给用户。

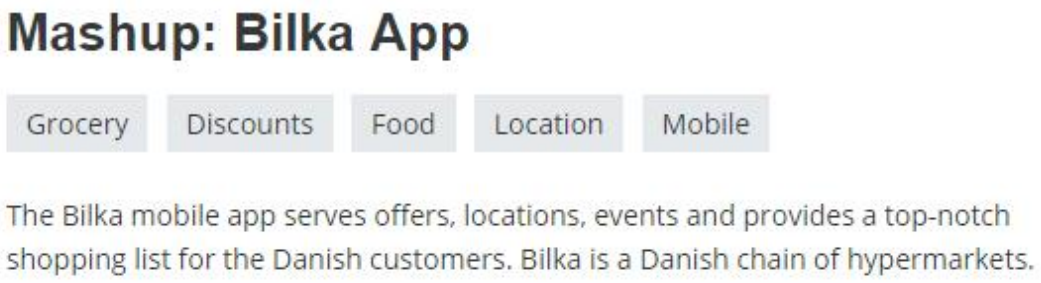

图 3-8: 一个 mashup 的描述信息

(3) 组合信息容易提取

组合信息作为 mashup 的重要信息之一，可以很容易从 ProgrammableWeb 网站获取，如图 3-9[①]所示，一个 mashup 应用在发布时，会以”Related APIs”的形式登记该 mashup 所依赖的原子服务 API。通过每个 mashup 页面的”Related APIs”，我们可以很容易得到每个 mashup 所蕴含的服务组合关系。

图 3-9: mashup 页面服务组合关系

3.4.2 组合相似度计算

为了能够利用服务组合关系，本文定义了组合相似度的，即两个 web 服务在组合关系上的相近程度。为了叙述上的不引起歧义，在本节我们将构成 mashup 的原子服务简称为 API。我们计算组合相似度主要分为两个部分：

[①]https://www.programmableweb.com/mashup/wunderwalk

(1) 根据 mashup 的服务组合信息建立 mashup-API 调用关系矩阵

对于 M 个 mashup 和 N 个 API，我们可以建立一个 $M \times N$ 的 mashup-API 调用关系矩阵，矩阵中的元素由 0 和 1 构成，对于矩阵中第 i 行第 j 列的值若为 1 则表示 mashup i 调用了 API j，若为 0 则表示 mashup i 没有调用 API j。以下表 3-2为例，有 3 个 mashup 和 4 个 API，它们的组合关系如表 我们可

表 3-2: mashup-API 调用关系表

mashup 名	所使用的 API
Anthems on Google Maps	Google Maps,Youtube
Audio Disco	MTV，YouTube
Favmvs	Google Search,MTV

以根据表 3-2建立如下 mashup-API 矩阵：

表 3-3: mashup-API 矩阵

	Google Maps	Youtube	MTV	Google Search
Anthems on Google Maps	1	1	0	0
Audio Disco	0	1	1	0
Favmvs	0	0	1	1

(2) 根据调用关系矩阵计算 web 服务之间的组合相似度

在生成调用关系矩阵后，我们可以根据矩阵来计算 web 服务之间的组合相似度。web 服务之间的组合相似度分为三种情况：

(2.1) 两个 web 服务都是 mashup 应用

我们假设这两个 mashup 应用为 mashup x 和 mashup y, 那么这两个 mashup 之间调用的公共 API 越多，则这两个 mashup 越相似，如果两个 mashup 之间没有公共调用的 API，那么这两个 mashup 在组合关系上应该是不相似的，即 mashup x 和 mashup y 的组合相似度为 0。如果两对 mashup 调用的公共 API 个数相同，那么包含更多 API 的 mashup 应该具有更低的相似度，为此我们定义两个 mashup 之间的组合相似度 $Sim_C(x,y)$ 如下：

$$Sim_C(x,y) = \frac{|A_1 \bigcap A_2|}{|A_1 \bigcup A_2|} \tag{3-5}$$

其中 A_1, A_2 为 mashup x 和 mashup y 所调用的 API 集合, 公式 (3-5) 可以

通过 mashup-API 调用矩阵 mashup x 和 y 的对应行求得。$Sim_C(x,y)$ 的值在 0 和 1 之间，值越大则表示越相似。

(2.2) 两个 web 服务一个是 mashup 一个是 API

这种情况可以将该 API 看成是一个特殊的 mashup，该 mashup 只调用一个 API, 那么此时这两个 web 服务之间的组合相似度可以看成是两个 mashup 之间的组合相似度，使用公式 (3-5) 来计算。

(2.3) 两个 web 服务都是 API

对于两个 API x 和 y，我们着重看这两个 API 被哪些 mashup 调用了，如果这两个 API 没有被任何 mashup 调用或者没有被某个 mashup 同时调用，那么这两个 API 之间在组合关系上应该是不相似的，即 API x 和 API y 的组合相似度为 0，如果这两个 API 被很多 mashup 同时调用，那么这两个 API 之间的组合联系就越强，即组合相似度就应该越大，同时，如果调用 API x 的 mashup 数越多或者调用 API y 的 mashup 数越多，则 API x 和 API y 之间的组合相关度应该被削弱，即组合相似度应该越低，因此，我们定义两个 API 之间的组合相似度 $Sim_C(x,y)$ 如下：

$$Sim_C(x,y) = \begin{cases} 0, & |M_1 \bigcup M_2| = 0 \\ \frac{|M_1 \bigcap M_2|}{|M_1 \bigcup M_2|}, & |M_1 \bigcup M_2| \neq 0 \end{cases} \tag{3-6}$$

其中，M_1 是调用 API x 的 mashup 集合，M_2 是调用 API y 的 mashup 集合，$|M_1 \bigcup M_2| = 0$ 表示 API x 和 API y 没有被任何 mashup 调用过。$Sim_C(x,y)$ 取值范围在 0 和 1 之间，值越大表示越相似，上式可以通过 mashup-API 调用矩阵的 API x 和 API y 的对应列求得。

3.5 模型融合

矩阵分解模型从用户历史偏好的数据集中训练出一个推荐模型，然后根据模型对用户的潜在偏好进行预测。矩阵分解模型不仅能带来很好的推荐结果，而且具有非常好的扩展性，能够融合各种特征，提升推荐的结果。

我们着重考虑的是带偏差的 SVD 分解模型 (bias-SVD，Singular Value Decomposition)。带偏差的 SVD 分解模型考虑用户和项目的偏差，即该模型认为，每个用户的评分会有差异性，以 5 分制的评分为例，有的用户整体打分偏低，在他的评分中 3 分就是高分了，而有的用户打分会整体偏高，可能会给很

多 4 分，5 分，而 3 分在该用户这边就是低分。同样，每个项目的评分也是有偏差的。因此，带偏差的 SVD 分解建立的预测模型如下：

$$\hat{r}_{ui} = \mu + b_u + b_i + \boldsymbol{p}_{\boldsymbol{u}}^T \boldsymbol{q}_{\boldsymbol{i}} \tag{3-7}$$

其中，$\hat{r}_{ui}$ 是用户评分的预测值,μ 表示训练集中所有用户的全局平均分，b_u 是用户偏置项，表示了用户评分习惯的差异性，b_i 是项目偏置项，表示了不同项目接受评分的差异性。$\boldsymbol{p}_{\boldsymbol{u}} \in R^f$ 是用户 u 的潜在因子向量,f 是潜在因子个数，$\boldsymbol{q}_{\boldsymbol{i}} \in R^f$ 是项目 i 的潜在因子向量。

对于这样的预测模型，我们可以通过求解以下目标函数来找到相应的参数的值：

$$\arg\min_{p_*,q_*,b_*} \sum_{(u,i)\in T} (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2 + b_u^2 + b_i^2) \tag{3-8}$$

其中，r_{ui} 是训练集中的真实用户评分，λ 是正则系数，用以防止过拟合，p_*, q_*, b_* 表示要求解的所有用户潜在因子向量，项目潜在因子向量以及偏差。

通常采用随机梯度下降的方法求解 (3-8)。根据随机梯度下降方法，该模型的参数更新公式如下：

$$\begin{aligned} b_u &\leftarrow b_u + \gamma \cdot (e_{ui} - \lambda b_u) \\ b_i &\leftarrow b_i + \gamma \cdot (e_{ui} - \lambda b_i) \\ p_u &\leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u) \\ q_i &\leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i) \end{aligned} \tag{3-9}$$

其中，$e_{ui} = r_{ui} - \hat{r}_{ui}$,$\gamma$ 是学习率，用以控制随机梯度下降的速度。

在前面的小节中，我们计算得到了 web 服务之间的主题相似度 Sim_T 和服务组合相似度 Sim_C，根据主题相似度，我们定义 web 服务 i 的在主题层面上的邻居集合 $N_t(i)$ 如下：

$$N_t^k(i) = \left\{ l | l \in Top^k(i), Sim_T(i,l) > 0, i \neq l \right\} \tag{3-10}$$

其中,$Top^k(i)$ 表示与 i 在主题相似度上最相近的 k 个 web 服务的集合，$Sim_T(i,l)$ 是服务 i 和服务 l 的主题相似度。

同样的，我们定义 web 服务 i 在服务组合层面上的邻居集合 $N_c(i)$ 如下：

$$N_c^k(i) = \left\{ l | l \in Top^k(i), Sim_C(i,l) > 0, i \neq l \right\} \tag{3-11}$$

其中，$Top^k(i)$ 表示与 i 在服务组合相似度上最相近的 k 个 web 服务的集合，$Sim_C(i,l)$ 是服务 i 和服务 l 的服务组合相似度。

我们将主题信息和服务组合信息融入到 SVD 分解模型中，通过两个权重参数来调整主题信息，服务组合信息所占的比重，因此构造如下评分预测：

$$\hat{r}_{us} = \mu + b_u + b_s + (1-\alpha-\beta)p_u^T \cdot q_s + \alpha \sum_{l \in N_t^{k1}(s)} \omega_{sl} p_u^T \cdot q_l + \beta \sum_{k \in N_c^{k2}(s)} \eta_{sk} p_u^T \cdot q_s \tag{3-12}$$

其中，$\hat{r}_{us}$ 是用户 u 对服务 s 的评分的估计值，α 是主题信息的平衡系数，β 是服务组合信息的平衡系数，$N_t^{k1}(s)$ 是与服务 s 在主题相似度上最相近的 k1 个服务的集合，$N_c^{k2}(s)$ 是与服务 s 在服务组合相似度上最相近的 k2 个服务的集合，ω_{sl} 是服务 s 和服务 l 的主题相似性权重，计算如下：

$$\omega_{sl} = \frac{Sim_T(s,l)}{\sum\limits_{i \in N_t^{k1}(s)} Sim_T(s,i)} \tag{3-13}$$

同样的，η_{sk} 是服务 s 和服务 k 的服务组合相似性权重，计算如下：

$$\eta_{sk} = \frac{Sim_C(s,k)}{\sum\limits_{i \in N_c^{k2}(s)} Sim_C(s,i)} \tag{3-14}$$

该预测模型可以通过如下的目标函数求解：

$$\arg\min_{p_*,q_*,b_*} \sum_{(u,s)\in T} (r_{us} - \hat{r}_{us})^2 + \lambda(\|p_u\|^2 + \|q_s\|^2 + \sum_{l \in N_t^{k1}(s)} \|q_l\|^2 + \sum_{k \in N_c^{k2}(s)} \|q_k\|^2 + b_u^2 + b_s^2) \tag{3-15}$$

我们可以通过随机梯度下降的方式求解该目标函数，随机梯度下降的更新

公式如下：

$$
\begin{aligned}
&b_u \leftarrow b_u + \gamma \cdot (e_{us} - \lambda b_u) \\
&b_i \leftarrow b_s + \gamma \cdot (e_{us} - \lambda b_s) \\
&p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot ((1-\alpha-\beta) q_s + \alpha \sum_{l \in N_t^{k1}(s)} \omega_{sl} q_l + \beta \sum_{k \in N_c^{k2}(s)} \eta_{sk} q_s) - \lambda \cdot p_u) \\
&q_i \leftarrow q_s + \gamma \cdot ((1-\alpha-\beta) e_{us} \cdot p_u - \lambda \cdot q_s) \\
&\forall l \in N_t^{k1}(s): \\
&q_l \leftarrow q_l + \gamma \cdot (e_{us} \cdot \alpha \cdot \omega_{sl} \cdot p_u - \lambda \cdot q_l) \\
&\forall k \in N_c^{k2}(s): \\
&q_k \leftarrow q_k + \gamma \cdot (e_{us} \cdot \beta \cdot \eta_{sk} \cdot p_u - \lambda \cdot q_k)
\end{aligned}
\tag{3-16}
$$

其中，$e_{us} = r_{us} - \hat{r}_{us}$,$\gamma$ 是学习率，用以控制随机梯度下降的速度。

我们的融合了主题信息和服务组合信息的 SVD 分解算法 (TC-SVD) 如下所示：

Algorithm 3.1 TC-SVD

Input:
Training Data set TD,latent factor number F
Topic Neighbour number K1,Composition Neighbour number K2
Topic similarity Matrix T,Composition similariy Matrix C
regularization value λ,learning rate γ
user numeber M,service number N

Output:
model parameters $B_U, B_S, P^{M\times F}, Q^{N\times F}$
/*===== Model learning =====*/

$B_U, B_S \leftarrow (0, 0, \cdots, 0)$
$P^{M\times F}, Q^{N\times F} \sim \mathcal{N}(0, 0.005)$
Compute the Topic Neighbour set N_t^{k1} of service
Compute the Compositional Neighbout set N_c^{k2} of service
repeat
  for $(u, s, r) \in TD$ **do**
    $e_{us} = r_{us} - \hat{r}_{us}$
    update $B_U(u), B_S(s), P(u), Q(s)$ by (3-16)
  end for
  $\gamma \leftarrow 0.99 * \gamma$
until the stop criterion is met
return model parameters

3.6 实验设计

3.6.1 实验数据及评估指标

为了评估主题信息和服务组合信息对服务推荐的影响，实验采用了 Gang Tian 等人[6] 提供的数据集 pweb[①], 该数据集爬取了 ProgrammableWeb 网站截止 2013 年 11 月 25 日的相关数据，包括 10000 多个 web 服务，7000 多个 mashup 以及 280000 多条用户 watchlist 记录，由于大多数用户的 wacthlist 中服务的数目非常有限，本文选择了 wacthlist 列表中至少包含 50 个服务或者 mashup 的用户，一共有 510 个用户和 4813 个服务被选中，其中 2713 个原子 web 服务和 2100 个 mashup。如表 3-4所示：

表 3-4: ProgrammableWeb 数据集数据统计

用户数	510
原子 web 服务数	2713
mashup 数	2100
平均 mashup 包含的服务数	2.2
数据稀疏率	99.13%

本实验的使用的是评分预测模型，所以采用的评估指标是均方根误差 (RMSE) 和平均绝对误差 (MAE)，均方根误差定义如下：

$$RMSE = \sqrt{\frac{1}{|T|}\sum_{(u,s)\in T}(r_{us} - \hat{r}_{us})^2} \tag{3-17}$$

平均绝对误差 (MAE) 定义如下：

$$MAE = \sqrt{\frac{1}{|T|}\sum_{(u,s)\in T}|r_{us} - \hat{r}_{us}|} \tag{3-18}$$

其中，r_{us} 是用户 u 对服务 s 的实际评分，$\hat{r}_{us}$ 是用户 u 对服务 s 的评分估计，T 为测试集。

[①] http://tiangang.theclever.me/supplements.html/

3.6.2 准确性验证

为了评测本章提出的服务推荐算法 TC-SVD 的性能，我们将它与 5 个典型的推荐方法进行了比较，这几个算法简介如下：

UPCC: 一种典型的基于用户的算法，使用皮尔逊相关系数作为相似度[20]。

IPCC: 一种基于项目的协同过滤算法，使用皮尔逊相关系数度量项目间的相似度[41]。

RSVD: 引入了正则项以防止过拟合的 SVD 分解方法。

RBSVD：引入了偏置项和正则项的 SVD 分解方法。[42]

TASR：引入了时间维度的矩阵分解方法。[6]

TC-SVD：我们的方法

我们将数据集划分为训练集和测试集，训练集和测试集的比例介于 0.5 到 0.9 之间，从而形成了不同稀疏程度 (data sparsity) 的训练矩阵, 表 3-5是不同稀疏率下各算法的性能。 由表 3-5可知，在不同稀疏率下，基于模型的方法普遍

表 3-5: 各算法 RMSE/MAE 比较

Algorithm	Training Set Ratio									
	0.9		0.8		0.7		0.6		0.5	
	Data Sparsity									
	0.9924		0.9933		0.9941		0.9950		0.9958	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
UPCC	0.0272	0.0196	0.0277	0.0198	0.0278	0.0202	0.0287	0.0210	0.0295	0.0215
IPCC	0.0302	0.0219	0.0301	0.022	0.0302	0.0221	0.0306	0.0225	0.0310	0.0228
RSVD	0.0155	0.00836	0.0171	0.00864	0.0174	0.00923	0.0184	0.0101	0.0203	0.0113
RBSVD	0.0128	0.00678	0.0129	0.00678	0.0132	0.00680	0.0145	0.00732	0.0145	0.00757
TASR	0.0125	0.00672	0.0128	0.00688	0.0129	0.0694	0.0143	0.00756	0.0141	0.0077
TC-SVD	**0.0124**	**0.00633**	**0.0127**	**0.00639**	**0.0130**	**0.0647**	**0.0142**	**0.00713**	**0.0140**	**0.00738**

比基于内存的方法（UPCC,IPCC）具有更好的准确度，和带偏差的 SVD 分解方法相比，我们的方法引入了主题信息和服务组合信息，在不同稀疏率下都具有更低的 RMSE 和 MAE, 和考虑时序的模型 TASR 相比，我们的方法在不同稀疏率下 MAE 的值都要更低，而 RMSE 的值在只稀疏率为 0.7 的时候表现不佳，在其他的稀疏率下都要略优于时序模型。总体而言，我们的方法在训练集比重占 0.9 时，相对不引入主题信息和服务组合信息的 RBSVD 方法，在 RMSE 上提升了 3.2%, 在 MAE 上提升了 7.1%。相对于引入了时序的 TASR 模型，RMSE 提升了 0.8%, 在 MAE 上提升了 5.8%。这说明，将主题信息和服务组合信息融

入到服务推荐的过程中，可以有效地提高推荐的预测性能。

3.6.3 各参数影响分析

我们的方法中的参数主要有平衡因子 α,β, 主题邻居数 K1 和组合邻居数 K2，潜在因子个数 F 以及 LDA 主题模型训练时所设置的主题个数 K, 实验中，我们为了简化调参的过程，设定主题邻居数 K1 和组合邻居数 K2 相同，即 K1=K2, 我们将正则项系数 λ 设为 0.001。下面将对各参数的影响作出分析。

(1) 平衡因子 α,β

实验中我们通过两个平衡因子 α 和 β 来控制主题信息和服务组合信息的权重。图3-10，3-11是 α 对 RMSE 和 MAE 的影响，α 控制着主题信息的权重，我们设置邻居个数为 15，主题个数为 50，潜在因子个数为 30，β 数值为 0.2, 取 α 的值在 [0,0.5] 之间。我们发现，随着 α 的增大，RMSE 和 MAE 的值在逐渐减小，准确性在上升，当 $\alpha = 0.1$ 时，RMSE 达到最小，当 $\alpha = 0.2$ 时，MAE 达到最小, 随后当 α 增大，RMSE 和 MAE 的值都在增大，这说明准确性在下降，这可能是由于主题信息的加入影响到了其他信息所占的比重。

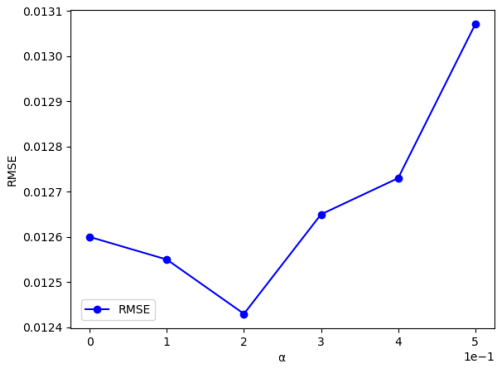

图 3-10: α 对 RMSE 的影响

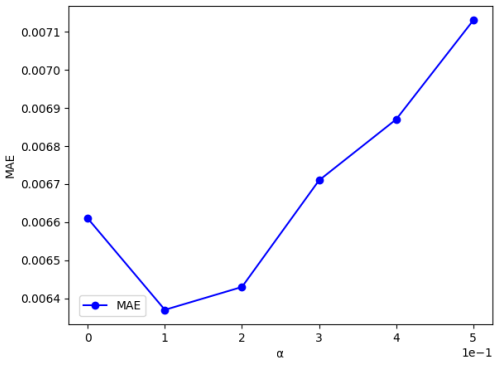

图 3-11: α 对 MAE 的影响

图3-12，3-13是 β 对 RMSE 和 MAE 的影响，从图中可以看出，随着 β 的增大，RMSE 和 MAE 都在下降，准确性在提高，当 $\beta = 0.2$ 时，RMSE 和 MAE 都取得最小值，后面随着 β 的值的不断上升，RMSE 和 MAE 的值也在上升，这说明推荐的准确性在下降，这是由于随着服务组合关系的影响权重在增大，影响了其他信息的所占的权重，从而带来了不好的效果。$\beta = 0.2$ 使得 RMSE 和 MAE 取得最小值，也说明了将服务组合信息融入推

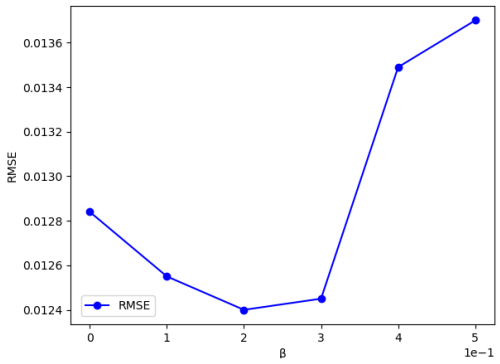

图 3-12: β 对 RMSE 的影响

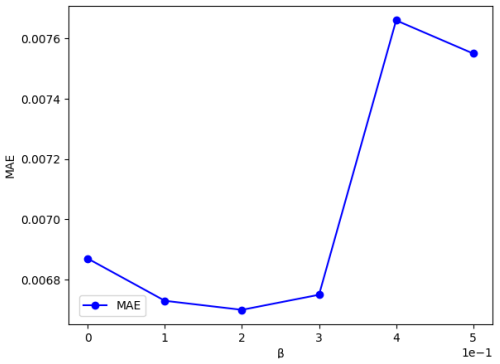

图 3-13: β 对 MAE 的影响

荐确实能够对提升推荐的准确性。

(2) 主题/服务组合邻居数 K1

实验中我们让主题邻居以及组合邻居的个数相同，我们将邻居个数 $K1$ 在区间 [5,50] 中调整，结果如图3-14，3-15所示，随着邻居数的增加，RMSE 和 MAE 的值都在降低，当 K1 在 15 到 50 之间增长时，RMSE 和 MAE 的值在上下波动，这可能是由于我们默认将主题信息邻居个数和服务组合邻居个数设置为相同，而邻居数在 15 到 50 之间时，这两种信息会对预测结果互相干扰。

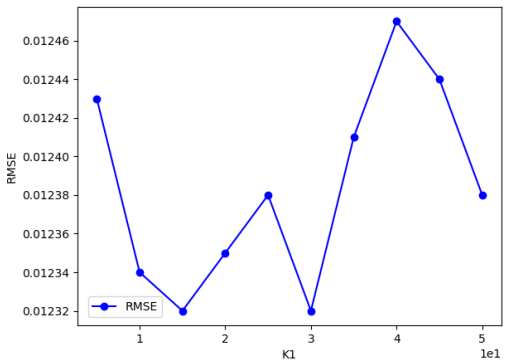

图 3-14: $K1$ 对 RMSE 的影响

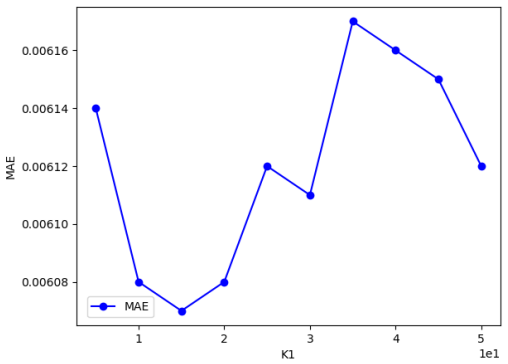

图 3-15: $K1$ 对 MAE 的影响

(3) 潜在因子个数 F

我们设置邻居数为 15，$\alpha = 0.2$,$\beta = 0.2$, 主题个数为 50，将潜在因子个数 F 在区间 [10,100] 中递增，结果如图3-16, 3-17所示，我们发现，随着潜在因子个数的增加，RMSE 和 MAE 都在减小，在 F=30 时，RMSE 达到最小值，随后 RMSE 在上下波动，并在 F=80 以后开始增大，可能是由于随着 F

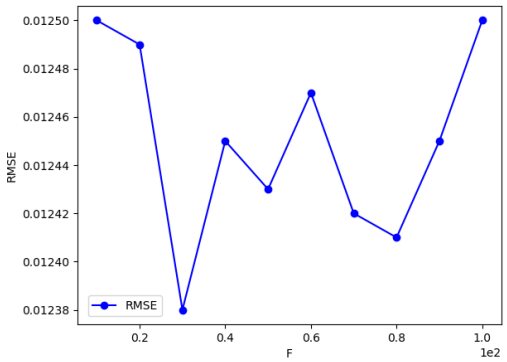

图 3-16: *F* 对 RMSE 的影响

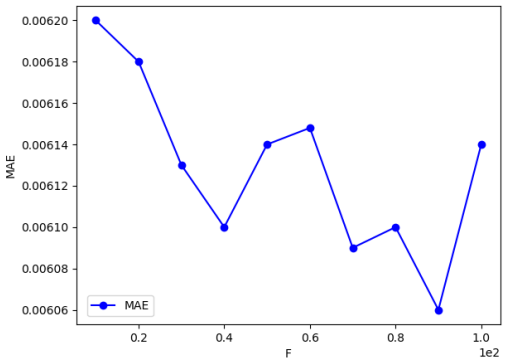

图 3-17: *F* 对 MAE 的影响

的增大，潜在因子维度不断增大，训练导致的数据过拟合现象开始越来越明显。同样的，当 F=90 时，MAE 取得最小值，随后也开始增大。我们为了减小计算开销，实验中最终设定的潜在因子个数为 30。

(4) 主题个数 K

我们使用了 LDA 主题模型来提取 web 服务的主题信息，LDA 主题模型有一个重要的参数，就是主题个数的设置，我们考察了主题个数的不同对 RMSE 和 MAE 的影响，我们设置主题个数在 [10,50] 之间，实验结果如图 3-18, 3-19所示，我们发现随着主题个数的增大，RMSE 和 MAE 总体呈下降趋势，这说明随着主题个数的增加，主题信息越来越准确，推荐的准确度也就越高。

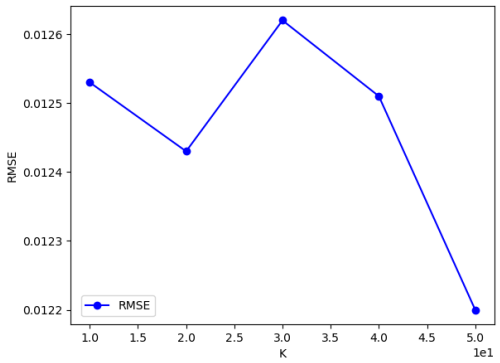

图 3-18: 主题数 *K* 对 RMSE 的影响

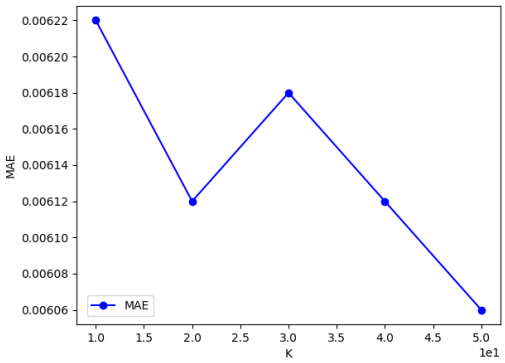

图 3-19: 主题数 *K* 对 MAE 的影响

3.7 本章小结

在这一章，我们提出了一种融合了主题信息和服务组合信息的 web 服务推荐算法，我们首先介绍了研究这一问题的背景，以及我们的主要研究对象 ProgrammableWeb 网站，然后介绍了引入主题信息和服务组合信息的动机和意义，并给出了主题相似度和服务组合相似度的定义，最后我们将主题信息和服务组合信息融合进 SVD 分解模型，并基于 ProgrammableWeb 网站的真实数据集来验证了我们的方法的有效性。

第四章　基于重排序的服务推荐多样性提高算法

4.1　问题背景

目前，大多数 web 服务推荐的研究都是基于精度、召回、均方根误差(RMSE) 等准确性指标，这些推荐算法力求通过在准确性指标上的提升来更好地预测用户的潜在兴趣，为用户推荐满意的服务。然而，文献[43] 表明，推荐的高精度不一定意味着较高的用户满意度。一方面，推荐列表中推荐的项目通常与用户的历史行为有着一定的关联性，而用户通常希望推荐列表中除了那些他们认为理所当然的推荐项目外，还需要有一点点惊喜。另一方面，通常具有高精度的推荐算法倾向于向用户推荐流行的项目，对于那些被少部分用户喜欢的，较为冷门的项目则难以被推荐。这会导致推荐系统中的项目，热门的项目更加热门，冷门的项目更加冷门。我们统计了 ProgrammableWeb 数据集中的每个服务的用户关注数，我们发现关注数最多的 20% 的服务，获得了大约 67.9% 的用户评分，而关注数最多的 100 个服务，获得了大约 32.5% 的用户评分。这表明，ProgrammableWeb 网站上已经出现了大量的长尾服务[44,45]，因此我们在基于 ProgrammableWeb 网站作服务推荐时，考虑推荐的多样性是很有必要的。

在服务推荐中考虑推荐的多样性，有以下好处：对 web 服务而言，冷门的服务有了更多被推荐给用户的机会，可以避免热门的服务越来越热门，冷门的服务无人问津的情况。对 ProgrammableWeb 网站的用户而言，可以帮助用户发现他们潜在喜欢的服务，也能发现一些用户意想不到的服务，给用户带来惊喜。同时，用户接触到更多样的服务，也能拓展用户的视野[46]。对 ProgrammableWeb 网站而言，可以让网站的大多数服务都有机会被用户发现，而不是只集中在少部分热门服务，更有利于网站的运营，增加网站的商业价值。

文献[26] 指出，提高推荐系统的多样性通常会降低系统的准确性，这也符合我们的直觉，即冷门的物品往往只被少数人所喜欢，如果把冷门的物品推荐给

不喜欢它的用户，显然会造成准确性的下降。推荐的多样性问题研究的难点，就是如何在准确性和多样性之间找到一个平衡，使得推荐结果可以在尽可能少的损失准确性的同时，尽可能多的提升推荐的多样性。

本章主要基于重排名模型，提出了一种考虑用户偏差的改进重排名算法，可以在尽可能少地牺牲准确性的情况下，尽可能多地提高推荐的多样性。

4.2 考虑用户偏差的重排名算法

经典的重排名算法使用固定阈值来调节推荐列表的准确性和多样性，但是用户是有差异的，使用统一的阈值来过滤用户评分，会影响模型的准确度，因此本章提出了考虑用户偏差的重排名算法。

4.2.1 用户偏差

在引入用户偏差之前，我们先给出服务流行度的定义，对于服务 i，流行度定义 $P(i)$ 如下：

$$P(i) = \left|\{u \in U | \exists R(u,i)\}\right| \tag{4-1}$$

其中 U 为用户集合，$R(u,i)$ 为用户 u 对服务 i 的评分。

服务流行度即表示喜欢该 web 服务的用户数量，对于一个 web 服务，如果喜欢该服务的用户越多，则该服务越热门，而喜欢该服务的用户越少，则该服务越冷门。一般而言，推荐冷门的物品会使系统的准确性降低，但是此时系统的多样性会增加，此时用户反而容易发现一些新颖的未接触过的物品[47]。

对于用户偏差，我们认为主要体现在以下两个方面：

1. 用户的评分的差异性。
   不同的用户评分习惯不同，有的用户评分比较严格，评分整体比较偏低，而有的用户则比较宽松，评分整体偏高。例如，在一个 5 分制的推荐系统中，用户 A 打分很严格，对喜欢的物品也只打 3 分，那么当评分阈值大于 3 分时，重排名将会对该用户失去作用，而用户 B 打分很宽松，对不喜欢的物品也打 3 分，那么当评分阈值小于 3 时，用户 B 将完全按照流行度进行排名。所以使用重排名算法时，考虑用户评分的差异性很有必要。
2. 用户兴趣分布的差异性。
   不同的用户兴趣分布不同，有的用户喜欢的都是偏热门的服务，那么给这

样的用户推荐很多冷门的服务就不是太合适，相反的，有的用户兴趣分布较广，喜欢的服务中既有热门的又有冷门的服务，那么为这样的用户适当冷门服务就理所应当。因此，考虑用户兴趣分布的差异性也是有必要的。

因此，在进行重排序的时候，相对于固定阈值的重排名模型，充分考虑用户的评分差异性和用户兴趣分布的差异性，可以更好地将模型多样性的提升作用于不同的用户。那么该如何表示用户的偏差呢？

对于用户评分差异性，可以通过用户评分偏置 (bias) 来表示。根据文献[48]用户偏置 b_u 可以用如下公式来估计:

$$b_u = \frac{\sum\limits_{(u,i)\in D} r_{ui}}{\lambda + \left|\{i|(u,i) \in D\}\right|} \tag{4-2}$$

其中 λ 是一个偏置调节参数，可以通过交叉验证得到，为了简化本文取 $\lambda = 0$。D 为用户评分的训练集，r_{ui} 是用户 u 对服务 i 的评分。

公式 (4-2) 实质上是通过计算用户以往评分记录的平均值来获取用户偏置 b_u 的值，如果用户评分比较严格，那么整体评分较低，用户偏置 b_u 的值也就相对较小，而用户评分较为宽松的话，整体评分较高，那么该用户的偏置也就相对较大，因此通过用户偏置 b_u 可以刻画用户评分的差异性。

对于用户兴趣分布的差异性，我们希望将兴趣分布较广的用户和兴趣较为单一的用户区分开来，以达到针对性地为用户提高多样性的目的，使得兴趣分布较广的用户能够获得更多样化的推荐，更有机会被推荐那些较为冷门的服务，而那些兴趣较为单一的用户，则需要减少冷门服务的推荐。为了能够表示用户兴趣分布的差异性，我们考虑使用用户偏好服务流行度的标准差来表示，即用户兴趣分布 d_u 表示如下：

$$d_u = \sqrt{\frac{\sum\limits_{i\in R(u)} (p_i - \bar{p}_u)^2}{\left|R(u)\right|}} \tag{4-3}$$

其中，p_i 是服务 i 的流行度，$\bar{p}_u$ 是用户 u 历史偏好服务集合中服务流行度的平均值，R_u 是用户历史偏好的服务集合，表示如下：

$$R(u) = \{i|(u,i) \in D\} \tag{4-4}$$

用户的兴趣分布 d_u 使用标准差来表示，当用户的 d_u 值较大时，表明该用

户喜欢的服务的流行度波动性较大，即该用户喜欢的服务较为多样，那么为该用户推荐一些冷门的服务是明智的。如果某个用户的 d_u 较小，则表明该用户喜欢的服务波动性较小，该用户品味较为“单一”，因此在推荐的时候应该降低推荐的多样性，减少冷门服务的推荐。

4.2.2 改进重排名模型

重排序模型通过引入排名阈值来对项目进行重新排名，从而使模型在准确性和多样性之间取得平衡。经典的重排名如下：

$$\begin{aligned} rank_x(i,T_R) &= \begin{cases} rank_x(i), if\ R^*(u,i) \in [T_R, T_{max}] \\ \alpha_u + rank_{standard}(i), if\ R^*(u,i) \in [T_H, T_R] \end{cases} \\ where \quad \alpha_u &= \max_{i \in I_u^*(T_R)} rank_x(i),\ I_u^*(T_R) = \{R^*(u,i) \geq T_R\} \end{aligned} \tag{4-5}$$

其中，$rank_x(i)$ 表示启发式的排名算法，$rank_{standard}(i)$ 是标准排名法，即根据预测评分的排序从高到低排名，$R^*(u,i)$ 是用户评分的预测值。

当评分大于阈值 T_R 时，会对项目按照 $rank_x(i)$ 来进行重新排名，而评分小于阈值 T_R 时，则使用标准排名法来进行排名。经典的重排名算法采用固定的阈值 T_R，为了进一步提高模型的多样性，我们考虑将用户偏差引入重排名模型中，即将用户评分偏差 b_u 和用户兴趣差异性 d_u 引入重排名模型，改进后的重排名模型定义如下：

$$\begin{aligned} rank_x(i,\theta) &= \begin{cases} rank_x(i), if\ R^*(u,i) \in [T_R(\theta), T_{max}] \\ \alpha_u + rank_{standard}(i), if\ R^*(u,i) \in [0, T_R(\theta)] \end{cases} \\ where\ T_R(\theta) &= \min(b_u + \frac{T_{max} - b_u}{d_u}\theta, T_{max}), \\ \alpha_u &= \min_{i \in I_u^*(T_R(\theta)} rank_x(i),\ I_u^*(T_R(\theta)) = \{R^*(u,i) \geq T_R(\theta)\} \end{aligned} \tag{4-6}$$

上式中阈值不是固定的，而是和用户评分偏差 b_u 以及用户兴趣差异 d_u 有关，$T_R(\theta) = \min(b_u + \frac{T_{max}-b_u}{d_u}\theta, T_{max})$。对于一个用户，我们通过 b_u 来控制用户的整体评分习惯的差异，根据文献[47,48], 一般认为认为阈值 T_R 必须大于 b_u, 同时我们通过 θ 来控制 $T_R(\theta)$ 的值，从而控制推荐结果的准确性和多样性。对于兴趣用户较广的用户，d_u 较大，则 $T_R(\theta)$ 较小，则对该用户更多地使用启发式排名算法，从而可以更多地提升多样性。而对于兴趣较为单一的用户，则 $T_R(\theta)$ 较

大，则该用户更多地使用标准排名算法，因此多样性提升较少。通过控制 θ 在 $[0, d_u]$ 之间，来控制 $T_R(\theta)$ 在 $[b_u, T_{max}]$ 之间变化，从而更有效地调整准确性和多样性。

我们的考虑用户偏差的改进重排名算法 (UBR,User Bias Reranking) 如下：

Algorithm 4.1 UBR

Input:
Predicted rating values of services,θ
Output:
Top-N candidate services for each user
/*===== General Steps For UBR=====*/
1: Compute web servises' popularity.
2: **for** each user u **do**
3: compute b_u and d_u
4: compute $T_R(\theta)$
5: rank all the predicted services according to $rank_{standard}(i)$
6: rerank the service whose predicted rating is above threshold $T_R(\theta)$ according to $rank_x(i)$
7: select top-N candidates services
8: **end for**

4.3 实验设计

4.3.1 数据集介绍及评估指标

为了验证我们提出的算法在提高 web 服务多样性方面的作用，实验依然采用第三章使用的 ProgrammableWeb 数据集 pweb, 并且对数据集进行了相同的预处理，即选择 watchlist 列表中至少包含 50 个服务或者 mashup 的用户，最终我们选择了 510 个用户和 4813 个 web 服务，用户对服务的历史评分在 $[0, 1]$ 之间。

本实验采用的精确度 (Presion) 作为准确性度量的方法，采用 Diversity-in-top-N[49] 作为多样性的评估指标，精确度定义如下：

$$Precision = \frac{|L_u \bigcap B_u|}{|L_u|} \tag{4-7}$$

其中，L_u 是用户的预测推荐列表,B_u 是用户 u 实际选择列表。

Diversity-in-top-N 定义如下：

$$Diversity - in - top - N = \left| \bigcup_{u \in U} L(u) \right| \tag{4-8}$$

4.3.2 不同启发式排名方法对比

重排序方法会对预测评分大于阈值 T_R 的服务使用启发式方法进行重新排序以提高多样性。通常启发式的重排序方法有以下三种：

逆序排名法 (Reverse Rating)：与标准排名 $Rank_{standard}$ 正好相反，将每个服务的评分从小到大排序，评分越低的服务排名越高

随机排名法 (Random Rating): 随机选择 N 个项目进行排名推荐。

项目流行度排名 (Item-Popularity Rating): 将候选项目按照流行度从低到高进行排序，即优先考虑候选服务集中不热门的服务进行推荐。

实验中将数据集的 80% 用作训练集，20% 用作测试，分别使用基于用户（user-based）和基于项目 (item-based) 的协同过滤来产生预测评分，邻居数设置为 50，推荐列表 N 长度设置 5。实验针对上述三种启发式的排名方法，测量了在不同 θ 取值下，准确性和多样性的变化曲线。

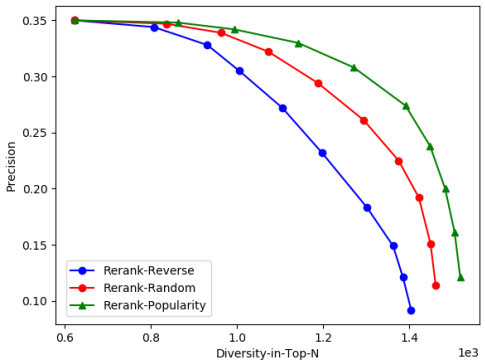

图 4-1: User-Based CF

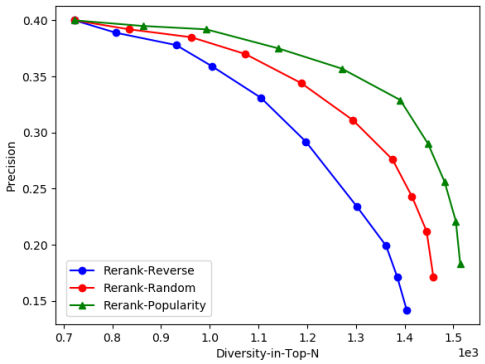

图 4-2: Item-Based CF

如图4-1,4-2所示，我们可以看到，对于 ProgrammableWeb 数据集，随着 θ 的降低，三种启发式排名的方法均可以通过损失一定的准确性来提高多样性，并且初始时损失很小的精度就可以获得较大的多样性性能的提升，随着 θ 的调节，精度损失越来越大，而多样性性能的提升越来越小。同时，我们发现，在 ProgrammableWeb 数据集中，以项目流行度作为启发式排名的模型，在损失相同精度的情况下，多样性性能提升最大, 这表明通过推荐冷门的服务给用户，能够很好地提升多样性。

4.3.3 与经典重排名算法的对比

我们将本文提出的考虑用户偏差的重排序算法与经典的重排序算法进行对比，分别使用基于用户（user-based）和基于项目 (item-based) 的协同过滤来产生预测评分, 使用服务流行度排名方法作为启发式排名的方法。实验结果如图 4-3, 4-4所示，我们可以看到，相对于经典的重排序算法，考虑了用户偏差的重排序算法可以在较少损失精度的情况下，较大的提升推荐的多样性。同时，考虑用户偏差的重排序算法最终推荐的服务种类也更加多样，这说明我们提出的算法是有效的。

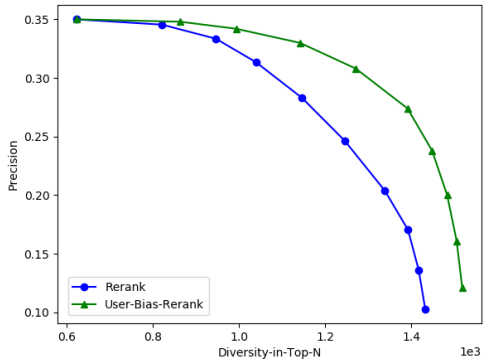

图 4-3: User-Based CF

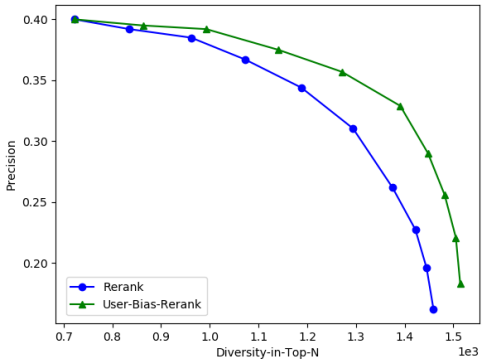

图 4-4: Item-Based CF

4.4 本章小结

本章我们阐述了服务推荐中引入推荐多样性的意义，介绍了常用的提升推荐多样性的策略，并提出了一种考虑用户偏差的改进重排名算法，该算法在重排名算法的基础上，考虑了不同用户在评分习惯和兴趣分布上的差异性, 从而更好地在准确性和多样性之间取得平衡。基于 ProgrammableWeb 网站的用户数据实验表明，本章的方法可以在尽量少的损失推荐准确性的同时，尽量多的提升推荐多样性，为用户推荐更加多样化的 web 服务。

第五章　总结与展望

5.1　工作总结

随着面向服务架构的发展，互联网上涌现了大量的 web 服务，如何为用户推荐合适的服务成为了当务之急。当前的服务推荐工作大都基于服务质量 QoS 或者用户的历史偏好信息，web 服务丰富的边缘信息没有得到很好地利用。同时，当前的 web 服务的相关研究工作，往往只关注了服务推荐的准确性，而忽略了推荐的多样性。本文鉴于当前研究的不足，基于服务平台 ProgrammaleWeb 的用户真实的历史偏好信息来发掘用户兴趣，在服务推荐的准确性和多样性方面作了如下工作：

1. 在服务推荐的准确性方面，提出了一个基于功能主题信息和服务组合信息的 web 服务推荐算法，该算法基于 ProgrammableWeb 网站的真实数据，通过用户的隐式反馈信息来提取用户的兴趣偏好，通过 ProgrammableWeb 网站的服务注册页面来提取了服务的功能描述信息以及服务组合信息，并以此分别计算了 web 服务之间在功能主题以及组合关系上的相似度，并根据相似度获取了每个 web 服务在功能主题或者组合上的邻居 web 服务集合，将其与 SVD(Singular Value Decomposition) 矩阵分解模型相结合，以预测用户评分的缺失值，实验表明，相对于 SVD 模型和引入时间信息的 SVD 模型，我们的算法在准确性上取得了一定的提升，这表明引入主题信息和服务组合信息的对提升推荐的准确性是有效的。
2. 在服务推荐的多样性方面，提出了一种考虑用用户偏差的重排名算法以提高推荐的多样性。该算法在经典的重排名算法的基础上，考虑了用户的评分习惯的差异性和兴趣分布差异性，并将这两种偏差结合进了阈值 T_R 的计算中，使得改进的重排序模型不再按照固定阈值 T_R，而是通过考虑了用户差异性的个性化阈值来为用户提高多样性。该算法不依赖于具体的推荐算法，可以方便地与其他推荐模型相结合。实验表明，考虑用户偏差的重排名算法，可以在尽量少降低准确性的情况下，大幅提升推荐的多样性。

5.2 研究展望

本文基于 ProgrammableWeb 网站的用户真实历史偏好信息，在服务推荐的准确性和多样性方面进行了研究，但是由于时间和精力有限，本文的工作还有很多可以提高的地方：

1. 在准确性方面，本文主要考虑了主题信息和服务组合信息，而对于用户间的关系，例如用户的社交关系没有进行考虑，同时，时间信息也是推荐中一个比较重要的因素。未来考虑在模型中综合考虑用户间的社交关系以及时间信息，使得推荐更加多元化。
2. 在多样性方面，本文在经典的重排名算法的基础上，考虑了用户的差异性，可以尽量少地降低精度的情况下，尽量多地提高推荐的多样性，但是这种方法不是一种最大化多样性的方法。因此未来考虑在允许一定精度损失的情况下，以最大化 web 服务推荐的多样性为优化目标进行研究。
3. 本文主要致力于服务推荐算法的研究，未来考虑将本文的算法集成到一个推荐系统中，增加系统的交互性，更好地进行推荐。

致　　谢

时间如白驹过隙，转眼间三年时间已经过去，研究生的学习生活也即将结束。在这三年的时间里，我的学习和生活都离不开老师和同学们的各种帮助，在此我要向他们致以最诚挚的谢意。

感谢我的导师胡昊副教授。无论是学习上还是生活上，胡昊老师都给我提供了非常多的帮助。本文从选题、实验到成文都离不开胡老师的悉心指导。胡老师一丝不苟的科研态度以及对学生无微不至的关怀，都给我留下了很深刻的印象，让我受益匪浅。

感谢南京大学软件研究所的各位老师。感谢吕建教授、马晓星教授、陶先平教授、徐峰教授、许畅教授、黄宇教授、余萍副教授、马骏老师、张建莹老师、徐经纬老师、汪亮老师、姚远老师、蒋炎岩老师、魏恒峰老师、匡宏宇老师等所有关心和帮助过我的老师。感谢你们在这研究生三年对我的指导和关怀，你们的言传身教将会是我一生最宝贵的财富。

感谢软件所所有的同学们，特别是张宗飞、孟占帅同学，非常荣幸能够和这些优秀的同学一起奋斗，一起度过人生最难忘的三年。感谢我的舍友吴少博同学在我的生活上的关心和帮助，让我即使远离家乡也能感受到家庭般的温馨。

最后我要感谢我的家人，你们一直是我最坚强的后盾，我的一路成长离不开你们的支持和鼓励。你们是我心灵的港湾，是我不断前进的动力。

参考文献

[1] BORCHERS A, HERLOCKER J, KONSTAN J, et al. Ganging up on Information Overload[J]. Computer, 1998, 31(4): 106–108.

[2] JANNACH D, ZANKER M, FELFERNIG A, et al. Recommender Systems: An Introduction[M]. [S.l.]: Cambridge University Press, 2010.

[3] LING-LAN, U. G. Web service recommendation method based on context[J]. Computer Engineering and Design, 2014.

[4] ZHENG Z, MA H, LYU M R, et al. WSRec: A Collaborative Filtering Based Web Service Recommender System[C] // IEEE International Conference on Web Services. 2009: 437–444.

[5] JIANG Y, LIU J, TANG M, et al. An Effective Web Service Recommendation Method Based on Personalized Collaborative Filtering[C] // IEEE International Conference on Web Services. 2011: 211–218.

[6] TIAN G, WANG J, HE K, et al. Time-Aware Web Service Recommendations Using Implicit Feedback[C] // IEEE International Conference on Web Services. 2014: 273–280.

[7] SHI Y, LARSON M, HANJALIC A. Collaborative Filtering beyond the User-Item Matrix: A Survey of the State of the Art and Future Challenges[M]. [S.l.]: ACM, 2014: 1–45.

[8] QI L, TANG Y, DOU W, et al. Combining Local Optimization and Enumeration for QoS-aware Web Service Composition[C] // IEEE International Conference on Web Services. 2010: 34–41.

[9] ZHENG Z, MA H, LYU M R, et al. QoS-Aware Web Service Recommendation by Collaborative Filtering[J]. IEEE Transactions on Services Computing, 2011, 4(2): 140–152.

[10] SHAO L, ZHANG J, WEI Y, et al. Personalized QoS Prediction forWeb Services via Collaborative Filtering[C] // IEEE International Conference on Web Services. 2007 : 439 – 446.

[11] ZHENG Z, MA H, LYU M R, et al. Collaborative Web Service QoS Prediction via Neighborhood Integrated Matrix Factorization[J]. IEEE Transactions on Services Computing, 2013, 6(3) : 289 – 299.

[12] ZHANG Q, DING C, CHI C H. Collaborative Filtering Based Service Ranking Using Invocation Histories[C] // IEEE International Conference on Web Services. 2011 : 195 – 202.

[13] TANG M, JIANG Y, LIU J, et al. Location-Aware Collaborative Filtering for QoS-Based Service Recommendation.[C/OL] // GOBLE C A, CHEN P P, ZHANG J. ICWS. [S.l.] : IEEE Computer Society, 2012 : 202 – 209. http://dblp.uni-trier.de/db/conf/icws/icws2012.html#TangJLL12.

[14] AMIN A, COLMAN A, GRUNSKE L. An Approach to Forecasting QoS Attributes of Web Services Based on ARIMA and GARCH Models[C] // IEEE International Conference on Web Services. 2012 : 74 – 81.

[15] ADOMAVICIUS G, KWON Y. TOWARD MORE DIVERSE RECOMMENDATIONS: ITEM RE-RANKING METHODS FOR RECOMMENDER SYSTEMS[J]. Workshop on Information Technologies & Systems, 2009.

[16] HILL W, STEAD L, ROSENSTEIN M, et al. Recommending and Evaluating Choices in a Virtual Community of Use[C/OL] // CHI '95 : Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. New York, NY, USA : ACM Press/Addison-Wesley Publishing Co., 1995 : 194 – 201. http://dx.doi.org/10.1145/223904.223929.

[17] RESNICK P, IACOVOU N, SUCHAK M, et al. GroupLens: An Open Architecture for Collaborative Filtering of Netnews[C/OL] // Proc. of ACM Conference on Computer Supported Cooperative Work (CSCW). Chapel Hill, USA : ACM, 1994 : 175 – 186. http://dx.doi.org/http://doi.acm.org/10.1145/223904.223929.

[18] BALABANOVIć M, SHOHAM Y. Fab: Content-based, Collaborative Recommendation[J/OL]. Commun. ACM, 1997, 40(3): 66–72. http://doi.acm.org/10.1145/245108.245124.

[19] LINDEN G, SMITH B, YORK J. Amazon.com recommendations. Item-to-item collaborative filtering[J]. IEEE Internet Computing, 2003, 7(1): 76–80.

[20] BREESE J S, HECKERMAN D, KADIE C. Empirical analysis of predictive algorithms for collaborative filtering[C] // Fourteenth Conference on Uncertainty in Artificial Intelligence. 1998: 43–52.

[21] SARWAR B, KARYPIS G, KONSTAN J, et al. Item-based collaborative filtering recommendation algorithms[C] // International Conference on World Wide Web. 2001: 285–295.

[22] 朱郁筱, 吕琳媛. 推荐系统评价指标综述 [J]. 电子科技大学学报, 2012, 41(2): 163–175.

[23] WENG L T, XU Y, LI Y, et al. Improving Recommendation Novelty Based on Topic Taxonomy[C] // Ieee/wic/acm International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops. 2007: 115–118.

[24] CAPPIELLO C, DANIEL F, MATERA M, et al. Information Quality in Mashups[J]. IEEE Internet Computing, 2010, 14(4): 14–22.

[25] SMYTH B, MCCLAVE P. Similarity vs. Diversity[C] // International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development. 2001: 347–361.

[26] ZHOU T, KUSCSIK Z, LIU J G, et al. Solving the apparent diversity-accuracy dilemma of recommender systems[J]. Proceedings of the National Academy of Sciences of the United States of America, 2010, 107(10): 4511.

[27] MCNEE S M. Being accurateis not always good: How accuracy metrics have hurt recommender systems[J]. Acm Special Interest Group on Computer Human Interaction, 2006: 1097–1101.

[28] PARK Y J, TUZHILIN A. The long tail of recommender systems and how to leverage it[C] // ACM Conference on Recommend System. 2008 : 11 – 18.

[29] SINGH S, BAG S, JENAMANI M. Relative similarity based approach for improving aggregate recommendation diversity[C] // India Conference. 2016 : 1 – 6.

[30] YIN H, CUI B, LI J, et al. Challenging the long tail recommendation[J]. Proceedings of the Vldb Endowment, 2012, 5(9) : 896 – 907.

[31] ADOMAVICIUS G, KWON Y O. Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques[J]. IEEE Transactions on Knowledge & Data Engineering, 2012, 24(5) : 896 – 911.

[32] ADOMAVICIUS G, KWON Y. Maximizing aggregate recommendation diversity: A graph-theoretic approach[J], 2011, 816 : 3 – 10.

[33] 刘轶, 范玉顺, 黄科满. 全局视角下的 Web 服务系统模型及推荐策略 [J]. 计算机集成制造系统, 2016, 22(1) : 133 – 143.

[34] DEHKORDI Y H, THOMO A, GANTI S. Incorporating User Reviews as Implicit Feedback for Improving Recommender Systems[C] // IEEE Fourth International Conference on Big Data and Cloud Computing. 2015 : 455 – 462.

[35] SHI Y, LARSON M, HANJALIC A. Collaborative Filtering beyond the User-Item Matrix: A Survey of the State of the Art and Future Challenges[J/OL]. ACM Comput. Surv., 2014, 47(1) : 3:1 – 3:45.
http://doi.acm.org/10.1145/2556270.

[36] BASU C, HIRSH H, COHEN W. Recommendation as classification: using social and content-based information in recommendation[C] // Fifteenth National/tenth Conference on Artificial Intelligence/innovative Applications of Artificial Intelligence. 1998 : 714 – 720.

[37] JIANG S, XUE F. An Improved Content-based Recommendation Method Through Collaborative Predictions and Fuzzy Similarity Measures[J]. New Technology of Library & Information Service, 2014.

[38] WANG F H, JIAN S Y. An Effective Content-based Recommendation Method for Web Browsing Based on Keyword Context Matching[J], 2006, 1 : 49 – 59.

[39] BLEI D M, NG A Y, JORDAN M I. Latent dirichlet allocation[J]. J Machine Learning Research Archive, 2003, 3 : 993 – 1022.

[40] HEINRICH G. Parameter Estimation for Text Analysis[J]. Technical Report, 2008.

[41] KANG G, LIU J, TANG M, et al. AWSR: Active Web Service Recommendation Based on Usage History[J]. Applied Mathematics & Information Sciences, 2012, 10(3) : 903 – 913.

[42] KOREN Y. Collaborative filtering with temporal dynamics[M]. [S.l.] : ACM, 2010 : 447 – 456.

[43] CREMONESI P, GARZOTTO F, NEGRO S, et al. Looking for “Good” Recommendations: A Comparative Evaluation of Recommender Systems[M]. [S.l.] : Springer Berlin Heidelberg, 2011 : 152 – 168.

[44] ANDERSON C. The Long Tail: Why the Future of Business Is Selling Less of More[M]. [S.l.] : Hyperion, 2006 : 274 – 276.

[45] LEVY M, BOSTEELS K. Music Recommendation and the Long Tail[J]. Womrad Workshop on Music Recommendation & Discovery Acm Recsys, 2010, 33(3) : 1 – 20.

[46] CHENG P, WANG S, MA J, et al. Learning to Recommend Accurate and Diverse Items[C] // International Conference on World Wide Web. 2017 : 183 – 192.

[47] 钟足峰, 段尧清, 杨曼. 可提高多样性的基于重排序图书推荐算法研究 [J]. 现代情报, 2017, 37(12) : 59 – 63.

[48] 彭飞, 邓浩江, 刘磊. 加入用户评分偏置的推荐系统排名模型 [J]. 西安交通大学学报, 2012, 46(6) : 74 – 78.

[49] GE M, DELGADO-BATTENFELD C, JANNACH D. Beyond accuracy:evaluating recommender systems by coverage and serendipity[C] // ACM Conference on Recommender Systems. 2010 : 257 – 260.

[50] YAO L, WANG X, SHENG Q Z, et al. Service Recommendation for Mashup Composition with Implicit Correlation Regularization[C] // IEEE International Conference on Web Services. 2015 : 217 – 224.

[51] LIU X, AGGARWAL C C, LI Y-F, et al. Kernelized Matrix Factorization for Collaborative Filtering[C] // SDM. 2016.

[52] ANON. GSVD++: Supporting implicit feedback on recommender systems with metadata awareness[C]. 2013 : 908 – 913.

简历与科研成果

基本信息

王勇，男，汉族，1993 年 4 月出生，江苏省泰州人。

教育背景

2015 年 9 月 — 2018 年 9 月 南京大学计算机科学与技术系 硕士
2011 年 9 月 — 2015 年 6 月 南京大学计算机科学与技术系 本科

攻读硕士学位期间完成的学术成果

1. 胡昊，王勇。专利“一种基于主题和服务组合信息的 web 服务推荐方法”. 申请号:201810424947.3

学位论文出版授权书

本人完全同意《中国优秀博硕士学位论文全文数据库出版章程》（以下简称“章程”），愿意将本人的学位论文提交“中国学术期刊（光盘版）电子杂志社”在《中国博士学位论文全文数据库》、《中国优秀硕士学位论文全文数据库》中全文发表。《中国博士学位论文全文数据库》、《中国优秀硕士学位论文全文数据库》可以以电子、网络及其他数字媒体形式公开出版，并同意编入《中国知识资源总库》，在《中国博硕士学位论文评价数据库》中使用和在互联网上传播，同意按“章程”规定享受相关权益。

作者签名：_____

_____年_____月_____日

论文题名	面向 ProgrammableWeb 网站的 web 服务推荐研究				
研究生学号	MF1533054	所在院系	计算机科学与技术系	学位年度	2018
论文级别	☐ 硕士　☑ 硕士专业学位 ☐ 博士　☐ 博士专业学位　（请在方框内画勾）				
作者电话	15651826383		作者 Email	1558448539@qq.com	
第一导师姓名	胡昊 副教授		导师电话	18913961638	

论文涉密情况：

☑ 不保密

☐ 保密，保密期：_____年_____月_____日 至 _____年_____月_____日

注：请将该授权书填写后装订在学位论文最后一页（南大封面）。