

# 武汉理工大学

(申请工学硕士学位论文)

## 基于“用户-标签”网络的问 答社区专家发现方法研究

培 养 单 位：计算机科学与技术学院

学 科 专 业：计算机科学与技术

研 究 生：黄辉

指 导 教 师：刘永坚 教授

2019 年 5 月

分类号\_\_\_\_\_

密 级\_\_\_\_\_

UDC \_\_\_\_\_

学校代码 10497

# 武汉理工大学

## 学 位 论 文

题 目 基于“用户-标签”网络的问答社区专家发现方法研究

英 文 Research on Expert Finding Method for CQA Service

题 目 Based on User-Tag Network

研究生姓名 黄 辉

指导教师 姓名 刘永坚 职称 教授 学位 学士

单位名称 计算机科学与技术学院 邮编 430070

申请学位级别 工学硕士 学科专业名称 计算机科学与技术

论文提交日期 2019 年 3 月 论文答辩日期 2019 年 5 月

学位授予单位 武汉理工大学 学位授予日期 2019 年 6 月

答辩委员会主席 邹承明 评阅人 教育部学位中心盲审

教育部学位中心盲审

2019 年 5 月

## 独创性声明

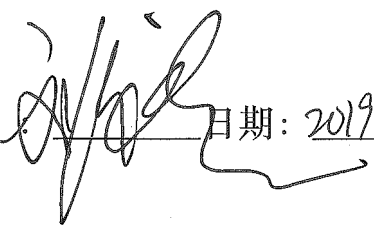
本人声明,所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知,除了文中特别加以标注和致谢的地方外,论文中不包含其他人已经发表或撰写过的研究成果,也不包含为获得武汉理工大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

签名: 黄辉 日期: 2019.5.20

## 学位论文使用授权书

本人完全了解武汉理工大学有关保留、使用学位论文的规定,即学校有权保留并向国家有关部门或机构送交论文的复印件和电子版,允许论文被查阅和借阅。本人授权武汉理工大学可以将本学位论文的全部内容编入有关数据库进行检索,可以采用影印、缩印或其他复制手段保存或汇编本学位论文。同时授权经武汉理工大学认可的国家有关机构或论文数据库使用或收录本学位论文,并向社会公众提供信息服务。

(保密的论文在解密后应遵守此规定)

研究生(签名): 黄辉 导师(签名):  日期: 2019.5.20

## 摘要

问答社区（Community Question Answering）为互联网用户提供了提问和回答的知识共享平台，如 Quora、Yahoo! Answers 和 Stack Exchange 等。这些问答社区满足了用户获取和发布知识的需求，因此吸引了各行各业的大量用户，并得到了快速的发展。在问答社区中，用户提出自己的问题并等待他人回答，如果问题不能及时的被解答，提问者很可能对问答社区失去信任感，从而造成用户的流失，影响社区的进一步发展。同时问答社区中用户的擅长领域各有不同，专业知识水平也参差不齐。因此，问答社区需要一种方法为提出的问题寻找到能提供高质量回答的专家。本文针对问答社区中的专家发现方法进行了研究，主要工作如下：

1) 针对因用户在创建标签过程中难以获得完美的统一而导致的标签过于细化问题和多命名问题，提出一种标签相似度衡量的方法，并结合马尔可夫聚类算法融合相似的标签，该标签聚类方法的有效性在 Stack Exchange 数据集上进行了实验验证。然后进一步探讨用户与标签之间的关系，构建出一种以用户和标签作为节点的网络，并应用网络嵌入方法生成用户向量，使得用户向量能同时含有标签信息和网络结构信息。

2) 针对问题设有最佳回答的问答场景，提出一种基于深度学习的专家发现方法。首先将问题的标题、正文和标签组合起来建立问题文本，并对问题文本进行若干数据清洗步骤得到了仅由重要词干组成的词序列，利用 word2vec 词向量模型训练生成问题向量。然后构造两个结构相同、参数不同的 DNN 分别从用户向量和问题向量提取特征，通过这二者的余弦相似度大小预测专家列表。最后在 Stack Exchange 数据集上进行实验，证明了提出的方法的有效性。

3) 针对在开放问答模式下问题有多个高质量答案的场景，本文将专家发现任务视为专家排序任务，并提出一种基于强化排序学习的专家发现方法。首先将专家发现任务形式化为马尔科夫决策过程，然后利用策略梯度方法学习模型的参数。最后在 Stack Exchange 问答数据集上进行了全方面的实验对比，证明了提出的方法的有效性。

**关键词：**问答社区，专家发现，深度学习，强化学习

## Abstract

Community Question Answering (CQA) such as Quora, Yahoo! Answers and Stack Exchange provides users with a knowledge sharing platform. CQA meets the needs of users learn and share knowledge. Thus, CQA has attracted a large number of users from various industries and develops rapidly. In CQA, askers post their questions and wait for others to answer. If the questions cannot be answered in time, the askers may not to trust CQA. This will result in the loss of users and affect CQA's further development. At the same time, the user's areas of expertise and levels of professional knowledge in CQA are different. Therefore, CQA needs a way to find experts who can provide high-quality answers for the posted questions. This thesis studies the expert finding methods in CQA. The main contents include:

1) In CQA, it is difficult for askers to create the most related tags for questions, resulting in that tags are too detailed or have multiple names. This thesis proposes a method for measuring the similarity of tags, and combines similar tags with the Markov Clustering Algorithm. The experiments on the CQA site Stack Exchange show that proposed method performs well. Further, this thesis discusses the relationship between users and tags, and constructs a network with user and tag as nodes. The network embedding method is applied to generate user representations. Therefore, this user representations contain tag information and structure information at the same time.

2) Facing that each question has one best answer, this thesis proposes an expert finding method based on deep learning (EF-DL). First, EF-DL combines the title, body and tags of the question to construct the question text. After several data cleaning steps for the question text, EF-DL can get a sequence of words consisting of important words, and the question representations is generated by word2vec. Then EF-DL constructs two DNNs with the same structure but different parameters to extract feature from the user representations and the question representations, and compares the cosine similarity of user feature and question feature to predict expert list. Finally, experiments on the Stack Exchange dataset show that EF-DL has better performance than other methods.

3) Facing that each question has several high-quality answers provided by experts, this thesis regards the expert finding task in CQA as the expert ranking task and

proposes an expert finding method based on reinforcement learn to rank (EF-RLTR). First, EF-RLTR formalizes the expert finding task in CQA as the Markov decision process, and then uses policy gradient algorithm to learn the parameters of the model. Finally, the thesis conducts extensive experiments on the well-known CQA site Stack Exchange. And experiments show that EF-RLTR can achieve better performance on expert finding tasks.

**Keywords:** Community Question Answering, Expert Finding, Deep Learning, Reinforcement Learning

# 目录

|                             |           |
|-----------------------------|-----------|
| 摘要 .....                    | I         |
| <b>Abstract .....</b>       | <b>II</b> |
| 第 1 章 绪论 .....              | 1         |
| 1.1 研究背景和意义 .....           | 1         |
| 1.2 国内外研究现状 .....           | 3         |
| 1.2.1 问答社区介绍 .....          | 3         |
| 1.2.2 基于主题生成模型的专家发现方法 ..... | 6         |
| 1.2.3 基于机器学习的专家发现方法 .....   | 7         |
| 1.2.4 基于问答关系网络的专家发现方法 ..... | 8         |
| 1.3 本文的研究内容 .....           | 9         |
| 1.4 本文组织结构 .....            | 10        |
| 第 2 章 “用户-标签”网络 .....       | 12        |
| 2.1 问题描述与分析 .....           | 12        |
| 2.2 融合相似标签 .....            | 13        |
| 2.2.1 标签相似度衡量 .....         | 13        |
| 2.2.2 针对相似标签的融合 .....       | 14        |
| 2.3 “用户-标签”网络 .....         | 17        |
| 2.3.1 用户与标签的联系 .....        | 17        |
| 2.3.2 构建“用户-标签”网络 .....     | 17        |
| 2.3.3 网络表示 .....            | 19        |
| 2.4 实验设计及结果分析 .....         | 21        |
| 2.4.1 数据集 .....             | 21        |
| 2.4.2 评价标准 .....            | 21        |
| 2.4.3 实验结果分析 .....          | 22        |
| 2.5 本章小结 .....              | 23        |
| 第 3 章 基于深度学习的专家发现方法 .....   | 24        |
| 3.1 问题描述与分析 .....           | 24        |

|                             |    |
|-----------------------------|----|
| 3.2 基于深度学习的专家发现方法 .....     | 25 |
| 3.2.1 算法思想 .....            | 25 |
| 3.2.2 问题文本向量 .....          | 26 |
| 3.2.3 深度学习模型 .....          | 28 |
| 3.3 实验设计及结果分析 .....         | 29 |
| 3.3.1 数据集和数据预处理 .....       | 29 |
| 3.3.2 结果评价 .....            | 31 |
| 3.3.2.1 评价标准 .....          | 31 |
| 3.3.2.2 对比方法 .....          | 31 |
| 3.3.3 实验结果与分析 .....         | 32 |
| 3.4 本章小结 .....              | 34 |
| 第 4 章 基于强化排序学习的专家发现方法 ..... | 35 |
| 4.1 问题描述与分析 .....           | 35 |
| 4.2 基于强化排序学习的专家发现方法 .....   | 37 |
| 4.2.1 算法思想 .....            | 37 |
| 4.2.2 马尔可夫决策过程 .....        | 38 |
| 4.2.3 策略梯度学习 .....          | 40 |
| 4.3 实验设计及结果分析 .....         | 42 |
| 4.3.1 数据集和数据预处理 .....       | 42 |
| 4.3.2 结果评价 .....            | 43 |
| 4.3.2.1 评价标准 .....          | 43 |
| 4.3.2.2 对比方法 .....          | 43 |
| 4.3.3 实验结果与分析 .....         | 44 |
| 4.4 本章小结 .....              | 47 |
| 第 5 章 总结与展望 .....           | 48 |
| 5.1 总结 .....                | 48 |
| 5.2 展望 .....                | 49 |
| 致谢 .....                    | 50 |
| 参考文献 .....                  | 51 |
| 研究生期间发表论文 .....             | 55 |



# 第 1 章 绪论

## 1.1 研究背景和意义

互联网的蓬勃发展给用户带来很大的便利，人们在互联网中的各种行为促进了信息时代的来临。在信息时代，信息的爆炸增长使得知识随时随地获取成为了现实，同时也给精准地获取知识带来了不便。对于一个用户来说，如果他遇到了疑问，他首先会通过搜索引擎获取想要的答案。而现阶段的搜索引擎主要是通过关键词对文档进行检索，几乎不考虑查询语句的语义信息，经常搜索出大量相关度较高、但是不准确的答案。因此促进了一些问答社区（Community Question Answering, CQA）的产生。

2001 年，谷歌公司建立了 Google Questions and Answers 以提供问答服务。但这种服务很快就因为用户的需求远远超过预期而停止，于是一年后谷歌重启了此服务，并改名为 Google Answers。与之前不同，Google Answers 采取了新的问答模式。在这种交互式问答模式下，用户付费提出自己的问题，经过严格认证和选拔的专家负责解答问题并给出优质的答案。此后出现了大量提供免费问答服务的网站，问答社区也因此得到更进一步的发展。

简而言之，问答社区是一个提供问答服务的知识共享平台，能满足用户提出问题和回答问题的需求。在问答社区中，用户提出自己的问题并等待其他用户的回答，其他用户能对该问题下的回答评价为“赞同”或“反对”，提问者也可以采纳其中一个答案，将之设置为“最佳答案”以结束该问题的讨论。这种互动方式使得提问者可以获得更具有针对性的答案，能有效减少用户在互联网中获取知识的成本。由于其具有开放性、交互性等特点，这种问答模式获得了广大用户的喜爱。问答社区经过长足的发展，在日益增多的用户的作用下，积累了海量数据，逐渐成为了一个庞大的知识库。

近年来，问答网站已经建立了非常大的问题和答案库<sup>[1]</sup>，比如百度知道<sup>[2]</sup>、知乎<sup>[2]</sup>、Stack Exchange<sup>[3]</sup>、Quora<sup>[4]</sup>以及 Yahoo! Answers<sup>[5]</sup>等，这些问答网站积累了数量庞大的问答数据，并且随着时间的推移，问答数据仍在急剧增加中。百度知道作为国内影响力最大的问答平台，截止到 2012 年 9 月 15 日，已累计为用

户解决了超过 2 亿个问题。知乎于 2013 年 3 月向公众开放注册，在短短一年的时间内就拥有了超过 400 万的用户。Stack Exchange 是一系列具有相同问答模式的问答网站集合，其含有 133 个网站，Stack Overflow 作为它的一员，是计算机编程领域最大的问答社区，以高质量的回答质量获得了广大用户的高度评价。而 Stack Overflow 发展至今，已经拥有超过 900 万注册用户。

在问答社区中，用户的行为主要有提出问题、解答问题以及对回答给出评价。问答社区的用户来自各个行业，专业知识领域各不相同，水平也有所差别，既有大量的“小白”用户，也有能提供高质量回答的专家用户<sup>[6]</sup>。专家用户主要为专业水平较高并且能提供较高质量答案的回答者，具体表现为给许多问题提供了让提问者满意的回答。

随着问答社区的日益发展，用户量的逐渐增多，问答社区中积累了大量的问答数据。如图 1-1，本文以 Stack Overflow 为例分析 2009 年到 2015 年之间的问题的增长情况和问题被解答百分比的趋势。从 2009 年开始，问题的数量不断快速增长，到了 2013 年累积的问题数量已经超过 200 万，但是至此之后，问题的增长速度明显减慢。另外一方面，问题被解答的百分比由 2009 年的 99.5% 下降到 2013 年的 90.9%，然后在两年内进一步下降至 79.5%。可以看出 2013 年是一个重要转折点，2013 年之前该社区发展迅速，但之后就停滞下来。综上所述，未被解答的问题的增多会影响新问题的提出，侧面说明社区的活跃度处于不断的降低中。

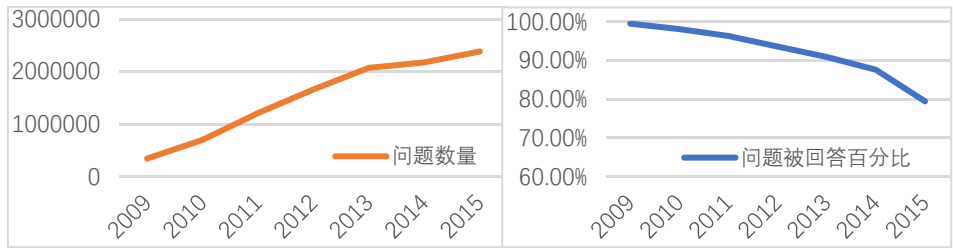


图 1-1 Stack Overflow 问答数据

如果大量新问题被提出，但是在一定时间内都得不到任何正确的回答，这无疑会影响提问者对社区的信任感，也会降低提问者再次访问问答社区网站的概率，可以认为未被解答的问题增多会阻碍社区的向前发展。因此问答社区需要一种方法将问题及时推送给具有丰富的专业知识并且能提供高质量的专家，此类方法被称为专家发现方法<sup>[7,8]</sup>。针对为问题寻找回答者的专家发现方法为问答社区的发展做出了卓越的贡献，同时专家给出的回答也能极大的提高社区中问答

数据的质量。

## 1.2 国内外研究现状

### 1.2.1 问答社区介绍

目前，绝大多数问答社区中都包含三个基本要素：用户、问题和回答。

（1）用户：是问答社区的参与者，也是问答社区的主要服务对象。在以用户为中心提供问答服务的问答社区中，用户可以提出自己的问题，也可以对问题的给出回答，还可以对社区中的问题或者回答给出自己的评价。通过这种模式，问答社区让用户体验到了交互式问答的乐趣，在不断的问答中，用户获得了前所未有的满足感。细分之下，用户可分为提问者、回答者和浏览者。一个人能同时扮演这三种角色：如果他感到困惑而提出问题，此时他的身份为提问者；当遇到其他人提出的问题而自己又恰好有相关的专业知识能提供回答时，他的身份此时为回答者；在问答社区中浏览感兴趣的问题或者评价他人给出的回答时，此时他是浏览者。

（2）问题：提问者在问答社区中发布的要求解答自身疑问的相关信息。如图 1-2 所示，问题由问题的标题、问题的正文以及若干个标签这三个元素所构成。其中标题能简要概括问题的内容，正文能详细描述提问者的信息需求，标签由提问者创建，是能概括问题内容和所属领域的关键字。提问者通过给问题创建标签，让回答者或浏览者更方便地、更准确地找到自己感兴趣的相关问题。

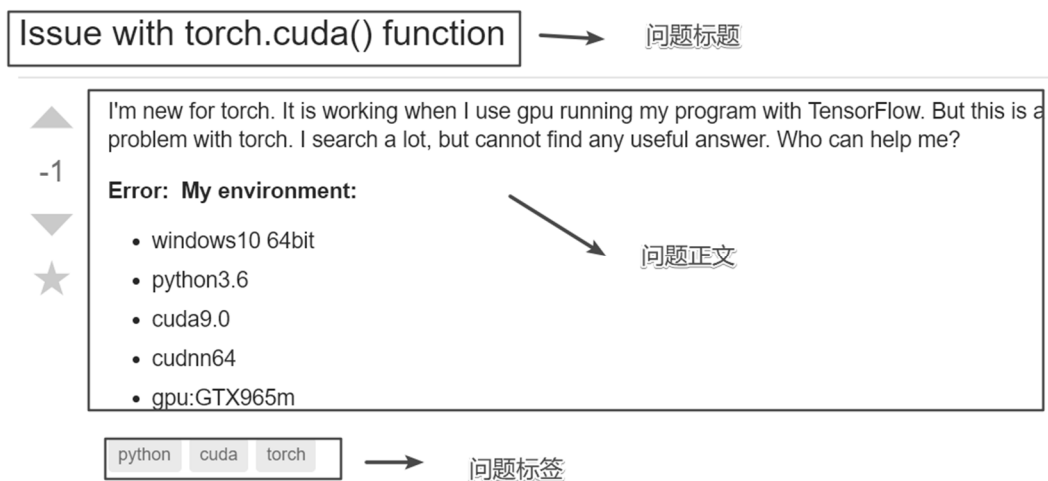


图 1-2 问题的三个元素

(3) 回答：问答社区中回答者对问题的回答。浏览者能评价回答，通过“赞同”或“反对”表达自己对该回答的满意度。如果提问者对某个回答满意，将之设置成“最佳回答”，那么该问题将会被标记为已解决。近期一些网站采用开放的问答模式<sup>[9]</sup>，如知乎，不再设置“最佳回答”以结束讨论，从而鼓励更多用户参与讨论，使得问题下的回答更加全面，回答质量更高。在这种开放的问答模式下，浏览者的评价就显得非常重要，问答社区一般根据回答获得的“赞同”数评估回答的质量。图 1-3 展示了回答内容。

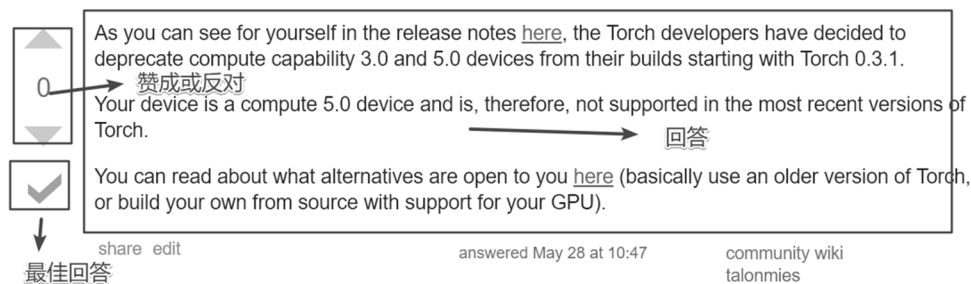


图 1-3 问题下的回答内容

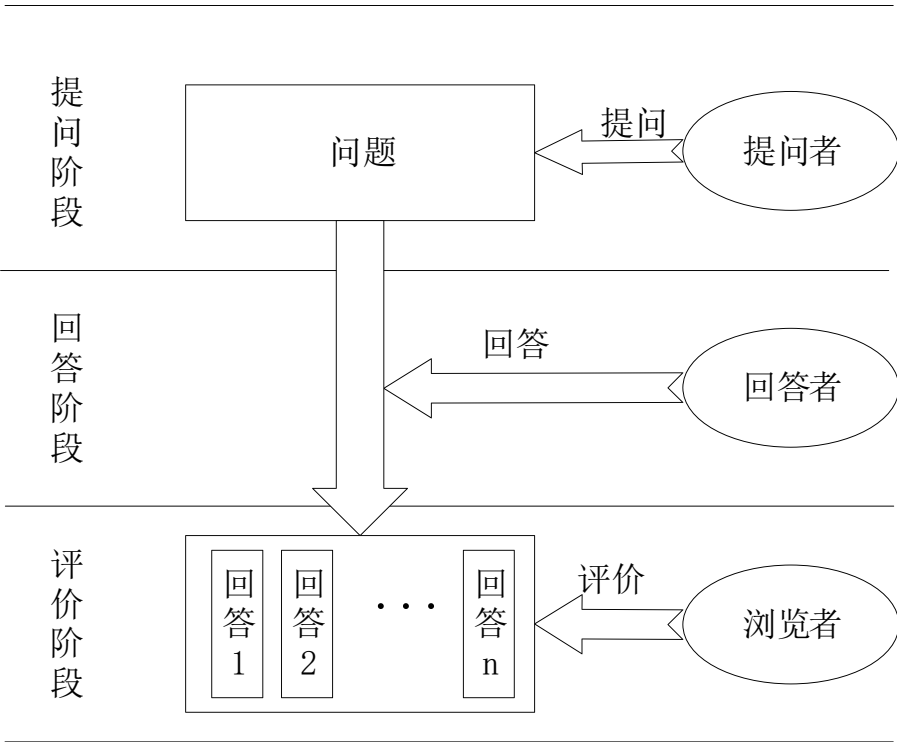


图 1-4 问答流程

问答社区围绕问题的提出、问题的解决以及对问答进行评价提供了各种服务，因此如图 1-4 所示，问答流程共分为提问阶段、回答阶段和评价阶段。

在提问阶段时，提问者在问答网站上输入标题、正文以及若干标签等问题的相关信息。如图 1-5 所示，问题提交后，会被展示在问答社区网站的首页以供其他用户浏览，同时提问者进入等待状态，等待其他用户的解答。

回答阶段时，问题等待回答者的解答。在信息时代的影响下，问答社区每时每刻都可能产生大量的问题，一个问题很可能未得到任何解答就被大量的新问题挤出了首页，从而被回答者发现并解答的几率变得更低。而提问者的耐心是有限的，如果在一定时间内提出的问题得不到解答，会严重影响提问者对问答社区的信任程度。专家发现方法在这一阶段则发挥出了作用，此类方法能为提出的问题寻找到能提供高质量答案的专家，以提高问题被解答的概率，同时专家提供的高质量答案也能丰富问答社区的知识库。因此，针对问题的专家发现方法成为了热门研究课题。

评价阶段时，问题下已有了若干个回答。浏览者可以是其他用户，也可以是回答者，还可以是提问者。浏览者对回答表示赞同或反对，提问者如果对其中的一个回答感到满意，可以将之设置成“最佳回答”以结束该问题的讨论，然后问答社区将该问题的状态转变成已解决。这个阶段伴随着大量交互数据的生成，一些研究者利用这些数据完成专家发现任务。

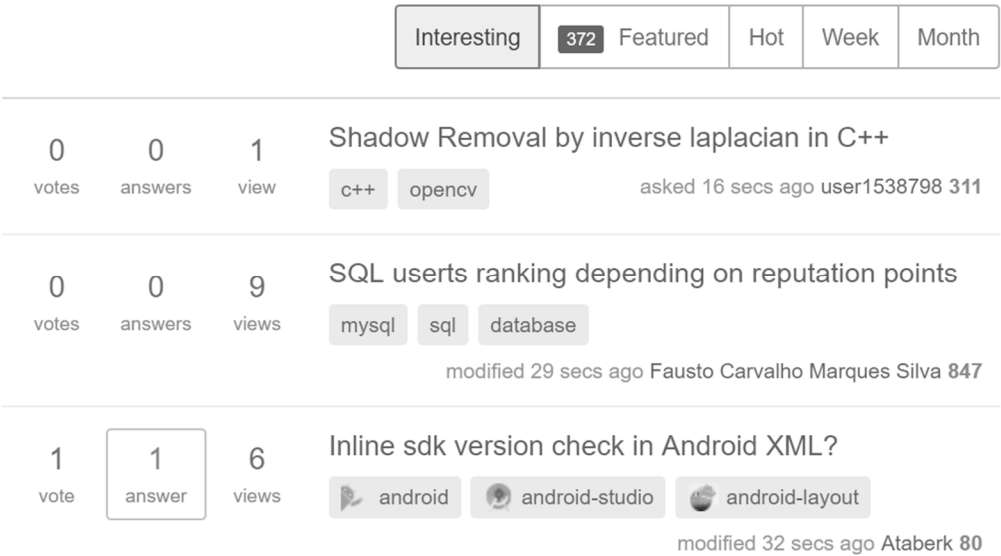


图 1-5 新提出的问题被展示在首页

随着问答社区的不断发展,问答社区吸引了大量学者的关注,其中针对专家发现问题,产生了大量的研究,研究者们提出了许多技术和模型。早期的研究主要采用基于统计分析的方法,通常使用一些统计特性来进行专家排序,如回答的相关性、回答者擅长的领域、活跃度等。一些特性通常由人工手动生成,随着社区中问答数据量的增多,在效率上已不能满足问答社区的需求。

通过对近期相关文献的阅读发现,社区问答中的专家发现方法主要分为三大类:基于主题生成模型的专家发现方法、基于机器学习的专家发现方法以及基于问答关系网络的专家发现方法。

### 1.2.2 基于主题生成模型的专家发现方法

主题模型是一种概率图模型,在文本挖掘研究中发挥着重要作用。其中最为经典的则是 LDA 模型<sup>[10]</sup>。LDA 模型由 Blei 等人于 2003 年发现并提出的一种无监督统计模型,是一个由词、主题和文档三层贝叶斯网络构成的文档主题生成模型,擅于挖掘文本中的潜在主题信息。

问答社区中的问答数据包含着大量的文本,因此一部分工作从挖掘文本中主题信息的角度出发,寻找问题和用户之间的潜在主题信息的关联<sup>[11,12]</sup>。大多数工作通常使用 LDA 主题模型寻找用户擅长的领域,计算用户在各个类别内的专业程度从而进行专家排序<sup>[13]</sup>。李科霖依据问答社区中的问答流程,提出“问题-回答者-话题”模型<sup>[14]</sup>,并结合社区中的对回答的评价行为,计算用户在各个主题下的专业知识水平,最后利用改进的 PageRank 算法,得到用户在主题下的专家得分。Riahi 等人<sup>[15]</sup>应用了分段主题模型(STM)来解决专家发现问题,并且对比了 TF-IDF 模型,语言模型以及 LDA 模型,实验结果显示 STM 表现更好。Rosen-Zvi 等人<sup>[16]</sup>提出一种“作者-主题”模型(ATM),以寻找作者、文档、主题和词之间的关系。

在问答社区中,问题或者回答的内容通常为较短的文本,而传统的主题模型方法在短文本上提取重要主题信息方面取得的效果不能让人满意<sup>[17,18]</sup>。问答社区中的问题通常附带有若干个标签,这几个标签能大致反映出问题的所属领域。Yang 等人<sup>[19]</sup>发现问题附带的若干个标签比文本的潜在主题具有更高的价值,因此基于问题标签研究用户的专业度。另外一部分工作认为标签能扩充短文本的语义信息,并引入标签完善主题模型以解决 LDA 模型在短文本上的缺陷。Cheng 等人<sup>[20,21]</sup>提出一种“标签词-主题”模型(Tagword Topic Model, TTM),利用问

题文本和问题附带的标签,使得每个问题中的文本集和标签集组成笛卡尔积,建立“标签-词”集,大大丰富了短文本中的文本信息,解决了短文本上提取主题信息的困难,为主题模型在短文本上的应用提供了新的思路。

### 1.2.3 基于机器学习的专家发现方法

问答社区中的问答数据的快速增长,导致了一些传统的算法不再适合,而机器学习(Machine Learning)在大数据下发挥了其训练数据量越大性能越好的优势<sup>[22]</sup>。Ji 等人<sup>[23]</sup>在传统的支持向量机的基础上,提出了 RankingSVM 模型,通过训练一个二分类器对样本进行分类,就能将排序转化为一个分类问题,从而用机器学习的方法进行排序。Yan 等人<sup>[24]</sup>则将专家发现任务视为排序任务,利用 ROC 曲线对候选回答者排序。

深度学习<sup>[25]</sup>是机器学习和人工智能研究的最热门的课题之一,也被称为分层学习(Hierarchical Learning)。为了学习复杂的功能,深度架构被用于多个抽象层次,即深度学习模型使用了多层次的非线性信息处理和抽象。深度学习通常根据网络的特征被分为三类:深度神经网络(DNN)、卷积神经网络(CNN)和循环神经网络(RNN)。这三种深度学习模型被广泛应用于图像识别、文本分类、语音识别等任务<sup>[26]</sup>。

深度学习使用神经网络对数据进行表征学习,在专家发现任务上表现出了巨大的潜力。一些工作将深度学习模型用于专家发现,这些方法通常根据用户的历史回答记录建立用户文档,然后从中利用深度学习模型提取用户特征。Zhou 等人<sup>[27]</sup>用自编码神经网络将问答社区中的问题和回答表示成含有语义信息的向量,根据问题和回答的向量计算相关度,提高了专家发现的精度。Zhang 等人<sup>[28]</sup>提出了一种将问答数据表示成向量形式的高效方法,并结合传统的语言模型,在 Yahoo! Answers 和百度知道的数据集上进行实验,验证了所提出方法的效果。Hsu 等人<sup>[29]</sup>利用 RNN 学习训练问题的文本,并使用迁移学习和多任务学习等技术解决了数据稀疏和不平衡标签的问题。Azzam 等人<sup>[30]</sup>使用深度语义匹配模型<sup>[31]</sup>提取问题文本特征和用户文档特征,然后根据这二者的余弦相似度从大到小排序获得专家列表。Wang 等人<sup>[32]</sup>使用卷积神经网络提取问题文本的特征,也取得了不错的效果。

但是深度学习模型需要大量的标注过的数据以支撑模型的训练,同时模型中含有大量的参数,并且需要较长时间的训练才能得到预期的参数,训练过程中

参数的优化也会受到各种因素的影响。

#### 1.2.4 基于问答关系网络的专家发现方法

在问答社区中，传统的方法通常根据问答关系构建网络，然后采用链接分析法计算用户的权威值，这类方法使用 PageRank 算法<sup>[33]</sup>或 HITS 算法<sup>[34]</sup>为社区寻找合适的专家。PageRank 算法基于“从许多优质的网页链接过来的网页，必定还是优质网页”的回归关系判断网页的重要程度，在专家发现任务中，网页节点则被替换成了用户和问题节点，一个用户如果被许多问题所链接，那么该用户的重要性较高。基于 HITS 的专家发现方法将使用用户分为两种类型，其中 Hub 组是提问者集合，Authority 组为回答者集合，然后根据网络的迭代求解 Hub 值和 Authority 值。Yang 等人<sup>[35]</sup>首先利用 LDA 模型学习用户兴趣的主题分布，然后结合链接分析法提出以用户为中心的专家发现方法。这些方法无法发现那些使用作弊手段（例如使用脚本自动回复）回答的用户，并且错误地认为他们是“专家”用户，同时也无法判断专家擅长的领域，简单地认为专家能回答所有领域的问题，在传统的小而专业度高的问答社区中能发挥作用，但随着问答社区不断扩大、综合性不断提高，发挥的作用也越来越小。

近期的一些方法结合网络嵌入的方法从问答关系网络中学习出用户或问题的特征，这种特征含有重要信息。Zhao 等人<sup>[36]</sup>构建用户与用户、用户和问题之间的异构网络，再通过 Random-Walk 算法<sup>[37]</sup>得到用户和问题的向量表示，最后使用深度神经网络学习提取特征并比较这二者之间的余弦相关度。Liu 等人<sup>[38]</sup>在文献<sup>[36]</sup>的基础上提出能联合网络结构信息和文本信息的动态门装置，然后使用 Neural Tensor Network<sup>[39]</sup>得到问题特征和用户特征的匹配分数。

以上的方法围绕问答关系，抽象出用户和问题作为节点构建网络，往往忽视了问题附带的标签的重要性。标签集中虽然包含着大量的无用标签，也不能完全正确的反映出问题的领域，但作为问题文本的补充信息，在问答社区中发挥了重要作用。

问答社区中存在的主要问题是大量问题得不到令人满意的回答，原因主要有两点：一是大多数的用户对回答其他人提出的问题不感兴趣；二是由于问题数量的增多，问题能被相应领域的专家所发现具有极大的随机性。因此针对问题的专家发现方法的研究与不断改进成为了问答社区良性发展的最大因素。



### 1.3 本文的研究内容

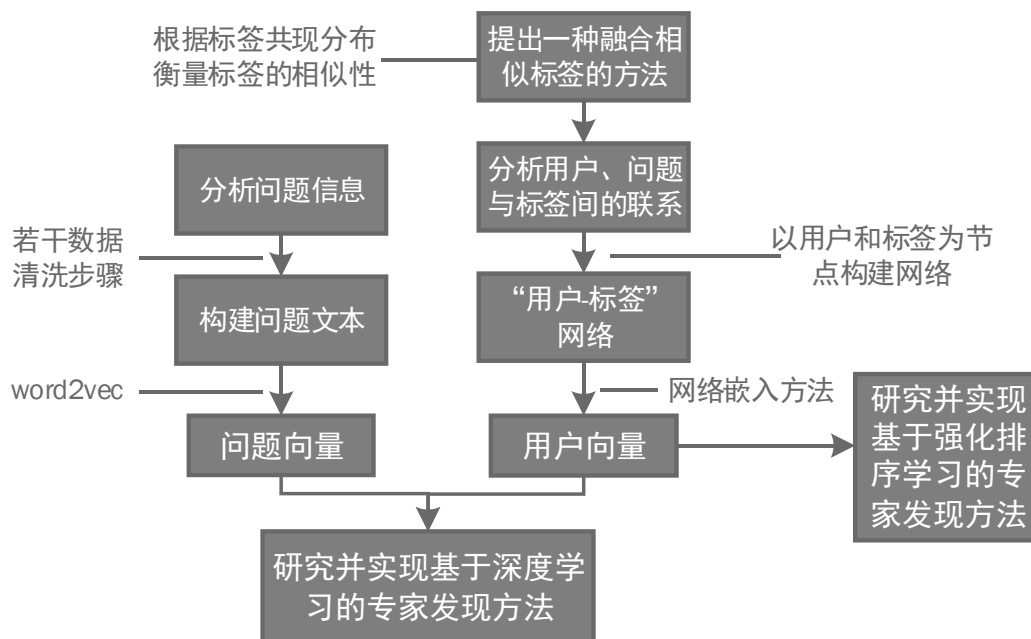


图 1-6 本文研究内容框架

如图 1-6 所示，本文对问答社区中的专家发现方法进行了研究。首先针对标签过于细化问题和多命名问题，提出了一种融合相似标签的方法，以此为基础构建“用户-标签”网络，并应用网络嵌入方法生成用户向量。然后，根据问题有“最佳回答”的特点提出一种基于深度学习的专家发现方法，为问题寻找能提供高质量答案的专家。最后，针对在开放的问答模式下问题有多个高质量答案的场景，提出一种基于强化排序学习的专家发现方法。具体工作如下：

#### （1）“用户-标签”网络。

问答社区中用户在创建标签时难以获得完美的统一，由此导致了标签过于细化问题和多命名问题。本文提出一种标签相似度衡量的方法，并结合聚类算法融合这些相似标签。然后进一步探讨用户与标签之间的关系，构建出一种以用户和标签作为节点的网络，并应用网络嵌入方法迭代网络生成用户向量。这种用户向量能同时含有结构信息和标签信息，能被后面提出的两种专家发现方法所使用。

#### （2）基于深度学习的专家发现方法。

针对问题设有最佳回答的问答场景，本文结合近期热门的深度学习提出一种基于深度学习的专家发现方法。首先将问题的标题、正文和标签组合起来建立

问题文本，经过对问题文本的若干数据清洗步骤过滤掉了文本中的无用词汇和符号，得到了仅由重要词干组成的词序列，利用 word2vec 词向量模型得到问题向量。对于用户向量，现有的方法根据用户的历史回答记录建立用户文档。此类用户文档由问题文本拼凑而成，导致了用户文档和问题文本过于相似，使用深度学习模型难以得出最好的效果。因此本文使用从“用户-标签”网络学习出的用户向量，这种用户向量与 word2vec 学习出的文本向量有一定差异，并且包含了更多的信息。最后构造两个结构相同、参数不同的 DNN 分别从用户向量和问题向量提取特征，并比较余弦相似度，根据大小预测专家列表。

### (3) 基于强化排序学习的专家发现方法。

近期越来越多的问答社区采用开放的问答模式，不再设置“最佳回答”以结束讨论，而是通过浏览者对回答进行评价的形式鼓励更多用户尽可能地深入探讨问题。在这种问答模式下，问题拥有若干个专家提供的高质量答案，基于深度学习的专家发现方法在准确寻找这若干个专家上表现出了不足。因此本文从专家排序的角度提出一种基于强化排序学习的专家发现方法，首先将专家发现任务形式化为马尔科夫决策过程，然后利用策略梯度方法学习模型的参数。智能体利用训练好的参数依次为问题从用户集中挑选出专家，最终得到专家列表。

## 1.4 本文组织结构

本文内容共分为五章，详细内容安排如下：

第一章为绪论部分。首先介绍了问答社区的背景，并阐述了专家发现方法的重要性。其次，介绍了问答社区的基本要素、问答流程以及三类专家发现方法。然后，介绍了论文主要的研究内容。最后，介绍了论文的组织结构。

第二章为“用户-标签”网络部分。首先阐述标签在定位问题上的重要性，以及标签系统存在的一些问题。其次，详细介绍了提出的融合相似标签的方法。然后，探讨用户和标签之间的联系，并利用这二者之间的关系构建“用户-标签”网络。最后在来自 Stack Exchange 的两个站点的标签集上进行实验，验证提出的融合相似标签方法的效果。

第三章为基于深度学习的专家发现方法部分。首先，阐述深度学习在信息时代下的优势。然后，详细介绍了提出的基于深度学习的专家发现方法，包括问题向量的生成方式以及神经网络的结构。最后在来自 Stack Exchange 的两个数据集上验证提出的方法的效果。

第四章为基于强化排序学习的专家发现方法部分。首先，介绍了专家发现任务和排序任务的联系，并探讨了强化学习和强化排序学习的关系。然后，提出了一种基于强化排序学习的专家发现方法，并从马尔科夫决策过程和策略学习这两个方面详细介绍提出的方法。最后，在 Stack Exchange 的子数据集“Super User”上进行全方面的实验对比，验证所提出的方法的效果。

第五章为总结与展望部分，对全文的工作进行了总结并且探讨下一步可进行的工作。

## 第2章 “用户-标签”网络

在问答社区中，提问者提出一个问题时需要输入与该问题相关的标签信息，这些标签能反映出问题的领域。由于社区将标签的创建权限开放给庞大的用户群体，随着时间的流逝，产生了庞大的标签库。另一方面，用户在创建标签时难以获得完美的统一，导致了标签过于细化和多命名问题。本文提出一种标签相似度衡量的方法，并结合马尔可夫聚类算法融合这些相似标签，在来自 Stack Exchange 的两个站点的数据集上验证提出的相似标签融合方法的有效性。然后更进一步地探讨用户和标签之间的联系，提出一种以用户和标签作为节点的网络构建方式，并应用网络嵌入方法从网络中学习出用户向量，使得用户向量同时含有标签信息以及网络的结构信息。

### 2.1 问题描述与分析

问答社区中的提问者、回答者和浏览者构成了用户群体，其中提问者是问题的生产者，提问者在提出一个新问题时需要输入问题的相关信息，包括标题、正文以及若干个标签。标签是能概括问题内容和所属领域的关键字，问答社区通过让问题附带标签信息，让用户方便地、准确地找到自己感兴趣的问题来回答或浏览。问答社区也能根据标签更好地管理庞大的问题库，并提供基于标签的搜索业务，让用户能搜索出某个标签下的一类问题。

标签系统在问答社区发展的推动上起了巨大作用，但随着问答社区的不断向前发展、用户的急剧增长，标签的数量逐渐变得非常庞大，呈现出多层次、多样化等特点，进而造成了传统主题模型的过拟合或过泛化的现象<sup>[40]</sup>，影响了传统方法的准确度，同时也由于其稀疏性而难以利用。

另一方面，由于问答社区将标签的创建权限开放给用户群体，而提问者在为问题创建标签时难以获得完美的统一，导致了标签过于细化问题或者对同一个标签会出现多个命名问题。例如在关于“ubuntu-14”这一类的问题上，一些用户创建标签“ubuntu-14.04”指定问题，另一些创建标签“ubuntu-14.03”，这属于标签层次过于细化问题。又例如标签“google-chrome”和“chrome”，这两个标签表意相同却命名不同，属于标签的多命名问题。标签过于细化问题和多命名问题

给标签集带来了噪声，因此减少标签集的噪声并利用该标签集对专家发现方法来说具有重大意义。

针对以上的问题，本文提出一种标签相似度衡量的方法，并结合马尔可夫聚类算法融合相似的标签以减少标签集中的噪声，然后通过构建“用户-标签”网络以利用该标签集。下面将从融合相似标签和“用户-标签”网络这两个方面详细阐述。

## 2.2 融合相似标签

### 2.2.1 标签相似度衡量

标签的过于细化问题和多命名问题一般表现为两个标签比较类似，因此需要一种方式度量它们的相似度，本文利用标签的共现次数分布来度量两个标签间的相似度。主要思想为：如果两个标签与其它标签的共现次数分布越接近，则这两个标签间的相似度也越高。

设标签集为  $T = \{t_1, t_2, \dots, t_n\}$ ，标签数量为  $n$ 。首先根据标签与标签在问题中的共现次数建立标签共现矩阵  $A_{n \times n}$ ，如式（2-1）。其中  $a_{ij}$  是矩阵内的第  $i$  行、第  $j$  列的元素，计算方式如式（2-2）所示：

$$A_{n \times n} = \begin{pmatrix} 0 & a_{1,2} & \cdots & a_{n,1} \\ a_{2,1} & 0 & \cdots & a_{n,2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & 0 \end{pmatrix} \quad \text{式（2-1）}$$

$$a_{ij} = \begin{cases} |S_i \cap S_j|, & i \neq j \\ 0, & i = j \end{cases} \quad \text{式（2-2）}$$

$S_i$  为标签  $i$  标注过的问题集， $S_j$  为标签  $j$  标注过的问题集， $a_{ij}$  表示为两个问题集的交集中的元素数量，即标签  $i$  与标签  $j$  的共现次数，当  $i = j$  时， $a_{ij} = 0$ 。因此可以令  $A_i = (a_{i1}, a_{i2}, \dots, a_{in})$  为标签共现矩阵的行向量，对应着标签  $i$  的特征向量，通过标签特征向量能计算相似度。

在使用余弦相似度函数计算标签  $i$  和标签  $j$  的相似度之前，先令  $a_{ij}$  和  $a_{ji}$  交换各自的值，以消除这两个标签间的自身联系，等到计算结束后再还原。其中标签之间相似度的计算方式如式（2-3）所示：

$$s_{ij} = \frac{\sum_{k=1}^n a_{ik} \times \sum_{k=1}^n a_{jk}}{\sqrt{\sum_{k=1}^n a_{ik}^2} \times \sqrt{\sum_{k=1}^n a_{jk}^2}} \quad \text{式 (2-3)}$$

根据各标签之间的相似度，建立标签相似度矩阵 $S_{n \times n}$ ，如式 (2-4)。其中当 $i = j$ 时，令 $s_{ij} = 0$ ，以此忽略标签与自身的相似度。

$$S_{n \times n} = \begin{pmatrix} 0 & s_{1,2} & \cdots & s_{n,1} \\ s_{2,1} & 0 & \cdots & s_{n,2} \\ \vdots & \vdots & \ddots & \vdots \\ s_{n,1} & s_{n,2} & \cdots & 0 \end{pmatrix} \quad \text{式 (2-4)}$$

本文定义标签的相似等级如表 2-1 所示，当 $s_{ij} > 0.9$ 时，定义标签 $i$ 与标签 $j$ 相似，当 $s_{ij} \leq 0.9$ 时，标签 $i$ 与标签 $j$ 不相似。本文的目标是找出标签集中的所有相似标签，并对它们进行聚类，聚类之后的标签集能解决标签过于细化问题 and 多命名问题，同时也减少了标签集的噪声。下面将介绍基于马尔可夫聚类的相似标签融合。

表 2-1 标签相似等级定义

| $s_{ij}$ | 相似等级定义 |
|----------|--------|
| (0.9, 1] | 相似     |
| [0, 0.9] | 不相似    |

### 2.2.2 针对相似标签的融合

本文利用马尔可夫聚类算法（The Markov Cluster Algorithm, MCL）<sup>[41]</sup>的思想通过标签之间的相似度对标签进行聚类，以解决标签的过于细化问题 and 多命名问题。

马尔可夫聚类算法是一种基于随机游走的图聚类算法，其基本思想为：在一个稀疏图中，如果某个区域是稠密的，则从此区域中任意一个顶点开始随机游走 $k$ 步，仍在此区域的概率很大。所以如果在图中随机游走 $k$ 步，出现在某一个区域内的概率很大，则该区域内的所有顶点构成一个聚类簇。如图 2-1，马尔可夫聚类算法首先用邻接矩阵表示图，再标准化成概率矩阵，接着就是对概率矩阵不断的重复展开（Expansion）和膨胀（Inflation）操作，直至矩阵稳定，最后得到{1,2,4}

和{3}这两个聚类簇。

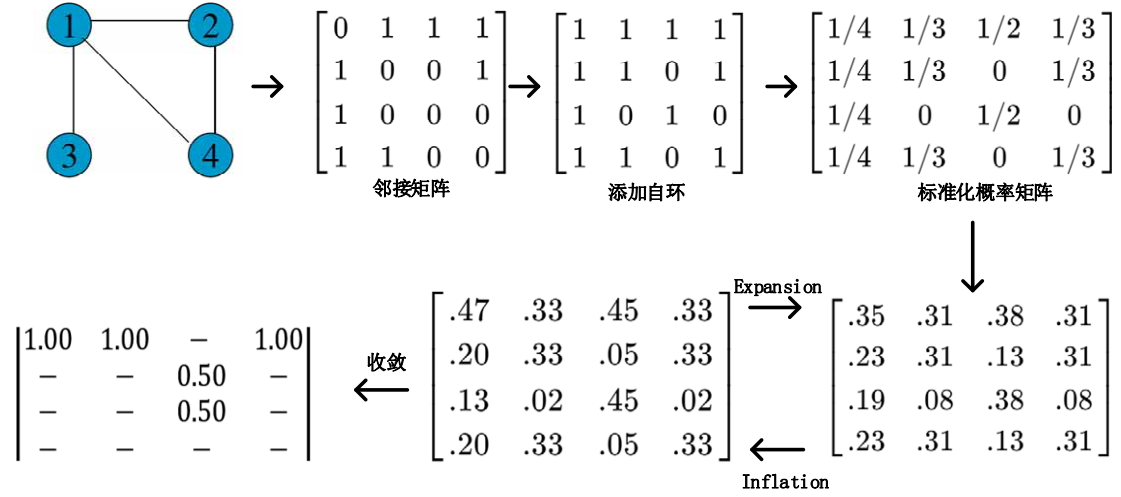


图 2-1 马尔可夫聚类过程

由于本文要做的是仅仅将相似标签融合，而不是对完整的标签集进行聚类。所以首先将相似( $s_{ij} > 0.9$ )的标签对抽取出来建立新的标签集 $T_s = \{t_1, t_2, \dots, t_m\}$ ，设这些标签的数量为 $m$ ，并建立标签相似度扩大矩阵 $D_{m \times m}$ ，定义矩阵内元素 $d_{ij}$ 如式 (2-5)：

$$d_{ij} = f(x) = \begin{cases} (s_{ij} - 0.9) \times 100, & \text{if } s_{ij} > 0.9 \\ 0, & \text{otherwise} \end{cases} \quad \text{式 (2-5)}$$

根据马尔可夫聚类算法的思想，需要对该矩阵标准化为概率矩阵 $C_{m \times m}$ ，定义矩阵内元素 $c_{ij}$ 如式 (2-6)：

$$c_{ij} = \frac{d_{ij}}{\sum_{k=1}^m d_{kj}} \quad \text{式 (2-6)}$$

以上过程如图 2-2 所示，由 6 个标签构成的标签相似度矩阵中，相似度大于 0.9 的标签被抽取出来，包括{1,2,3,4,6}，并建立相似标签集。然后建立标签相似度扩大矩阵，最后标准化生成了马尔可夫聚类算法需要的概率矩阵。

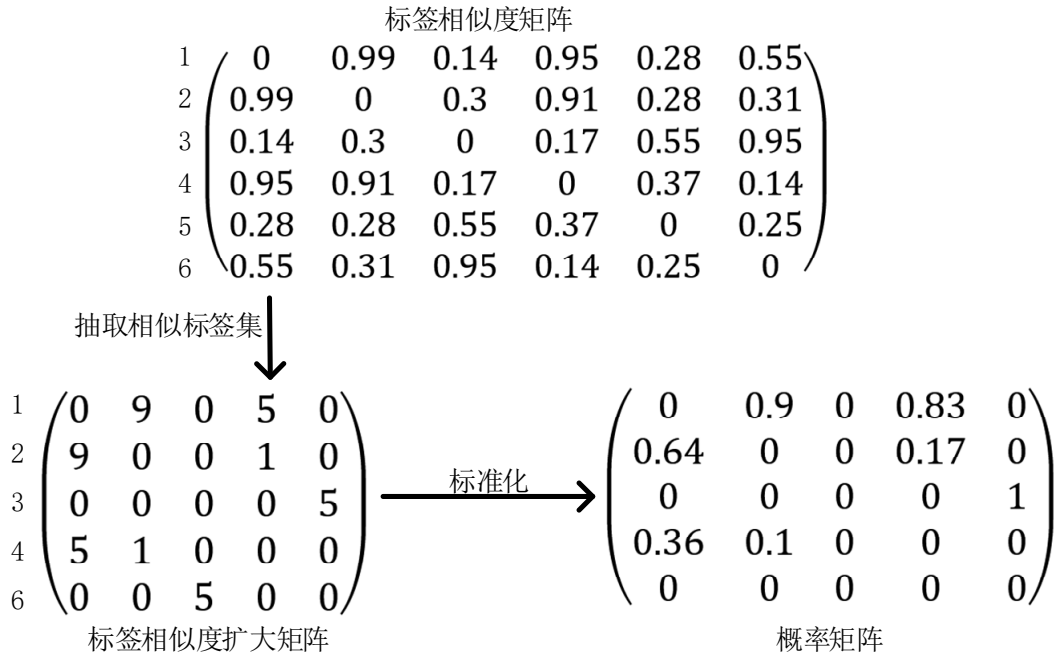


图 2-2 抽取相似标签集过程以及标准化概率矩阵

对于该概率矩阵，不断地重复展开和膨胀操作，直至概率矩阵收敛，其中 MCL 中的展开和膨胀操作的定义如下。

**展开 (Expansion):** 不断对矩阵进行幂次运算，相当于模拟顶点在图中随机游走。矩阵幂运算如式 (2-7) 所示，这里的  $k$  是随机游走的步数。

$$Expand(C) = C^k \quad \text{式 (2-7)}$$

**膨胀 (Inflation):** 这个过程是为了解决展开过程所导致的概率趋同问题，简单来说是将概率矩阵中的每一个值进行了一次幂次扩大，这样能强化紧密的点，弱化松散的点。它的扩大公式如下所示，其中  $r$  表示膨胀系数

$$c_{ij}^{(inf)} = \frac{c_{ij}^r}{\sum_{k=1}^m c_{kj}^r} \quad \text{式 (2-8)}$$

当概率矩阵收敛后，如果  $C$  中第  $j$  列中如果有非零值  $c_{ij}$ ，则表示标签  $i$  和标签  $j$  处于同一个聚类簇内。利用这种方式实现了相似标签的融合，有效的减少了标签集中的噪声。下节将介绍如何利用该标签集。



## 2.3 “用户-标签”网络

### 2.3.1 用户与标签的联系

用户作为问答社区的最重要的元素，一般不会事先为自己设置几个关键字表明自己的属性，而是通过问答的过程持续地更新自身的信息。在问答社区中，提问者输入问题的相关信息，并为这个问题创建若干个标签。回答者和浏览者能通过这些标签判断问题的大致内容以及所属的领域，标签起到了“关键字”的作用。回答者如果回答了这个问题，能说明这些标签标注的问题是回答者所感兴趣的问题，这些标签也能反映出回答者的领域偏好和用户属性。因此本文通过这种方式挖掘出用户与标签之间的联系。对于某个标签，如果回答者回答此标签下的问题越多，说明回答者与此标签越“亲密”。对于问答系统来说，如果新提出的问题标注了这个标签，则应该优先推荐给该回答者。

如图 2-3 所示，用户回答了某个问题，该问题附有三个标签，因此认为这三个标签与该用户有一定的联系，这三个标签能反映出该用户的兴趣偏好和属性。

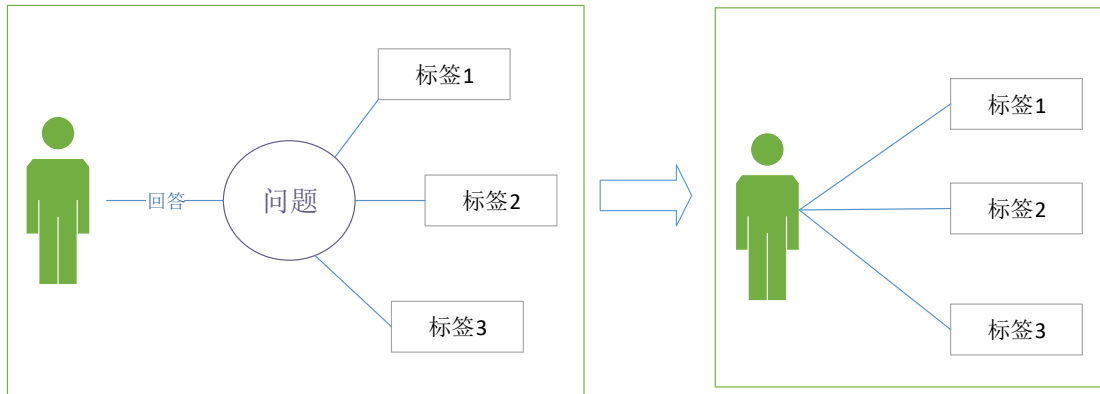


图 2-3 用户与标签之间的联系

### 2.3.2 构建“用户-标签”网络

设问题集为 $Q = \{q_1, q_2, \dots, q_l\}$ ，用户集为 $U = \{u_1, u_2, \dots, u_m\}$ ，标签集为 $T = \{t_1, t_2, \dots, t_n\}$ 。基于用户集和标签集构建“用户-标签”网络，该网络能反映出用户与标签在网络层面中的关系。设该“用户-标签”网络为 $G = (V, E)$ ，其中节点集 $V$ 包含用户集 $U$ 和标签集 $T$ 这两种类型的节点，边集合 $E$ 由用户-标签关系和标

签-标签关系组成，两种关系的详细定义如下所述：

- a) 标签-标签关系。定义标签与标签在问题集 $Q$ 中每一个问题中的共现次数累积和为边的权值，因此设标签与标签之间的边的权值为 $E^{(a)} \in R^{n \times n}$ 。如 $e_{ij}^{(a)} = k$ 表示：第 $i$ 个标签和第 $j$ 个标签在问题集 $Q$ 中每一个问题中的共现次数和为 $k$ 。
- b) 用户-标签关系。如上一节所述，用户一旦回答了某个问题，那么该问题附带的若干个标签能反应出此用户的标签偏好。设用户与标签之间的边的权值为 $E^{(b)} \in R^{m \times n}$ 。如 $e_{ij}^{(b)} = k$ 表示：第 $i$ 个用户在取得最佳回答的全部问题附带的标签中，第 $j$ 个标签的出现次数为 $k$ 。

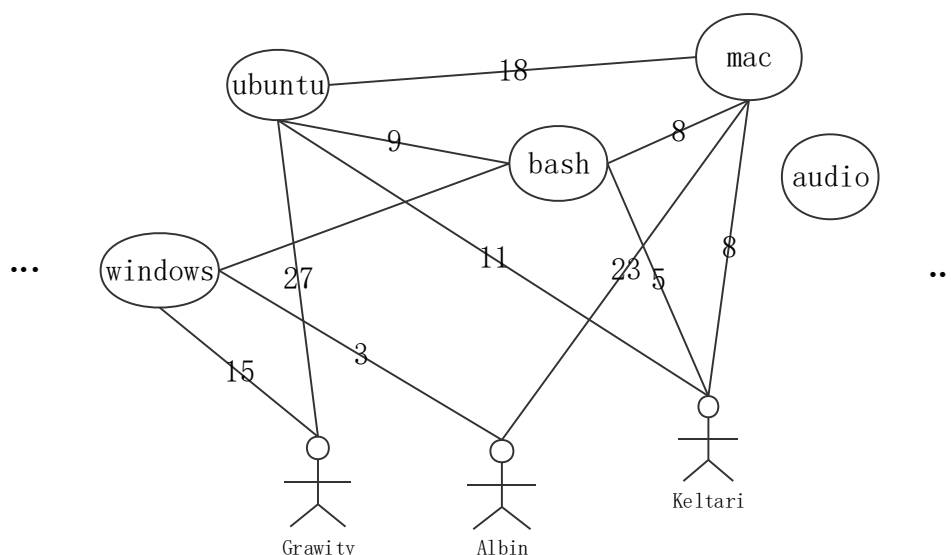


图 2-4 “用户-标签”网络的一个例子

图 2-4 展示了“用户-标签”网络的一个例子，其包含有两种节点：用户节点和标签节点，以及代表这些节点之间亲密程度的边。可以看中，图中有 3 个用户以及 5 个标签，他们之间被边所连接着。其中用户“Keltari”与标签“ubuntu”、“bash”、“mac”连接，边的权值分别是 11、5 和 8，说明在用户“Albin”回答过的问题中，标签“ubuntu”出现了 11 次，标签“bash”出现了 8 次。同时，标签与标签之间也相互连接着，标签“ubuntu”与“bash”的边权值为 9，说明他们共同出现在一个问题中的总次数为 9，这个数字越高则代表它们所描述的领域越接近。此外，标签“audio”没有和任何其他节点相连接，说明这是一个孤立标签。

### 2.3.3 网络表示

信息社会的数据随着时间呈现指数增长，而很多数据都相互关联，具体表现为网络形式，如社交网络、论文引用网络、互联网、生物网络等。这些信息网络的规模从几百万个节点数十亿个节点不等。信息网络非常庞大，因此在整个网络上执行复杂的推理过程可能会非常困难。传统的网络表示一般使用高维的稀疏向量，但是面对这种庞大的网络形式已无能为力。近年来，随着表示学习技术在自然语言处理等领域的发展和广泛应用，学者们受此启发，提出了用于解决该问题的一种方法，网络嵌入（Network Embedding）<sup>[42,43]</sup>。网络嵌入方法旨在学习网络中节点的低维度向量，所学习到的向量可以用作基于网络的各项任务，例如分类、聚类、链路预测和可视化等。因此网络嵌入的中心思想就是找到一种映射函数，将网络中的每一个节点转换成低维度的向量。

总的来说，网络嵌入方法应具有如下特征：

（1）适应性（Adaptability）：现实中的各种网络处于不断发展中，网络的规模随着时间会逐渐增长，因此对于不断变化的网络，网络嵌入算法能在不重复学习过程的情况适应网络中新增加的节点。

（2）可扩展性（Scalability）：真实世界下的各种网络通常都比较庞大，因此网络嵌入算法应该具有在短时间内处理大规模网络的能力。

（3）群体感知（Community aware）：节点之间的距离能用于评估节点与节点的某种关系。

（4）低维（Low dimensional）：生成的节点向量维度较低。

（5）连续（Continuous）：学习出的节点向量在空间内应具有连续性。

Tang 等人<sup>[44]</sup>提出了一种名为 LINE 的网络嵌入方法，该模型将网络中的节点根据其关系的疏密程度映射到向量空间中去，使联系紧密的节点被投射到相似的位置中去。而在衡量网络中两个节点联系紧密程度的一个重要指标就是这两个节点之间边的权值。LINE 不仅仅只考虑了一阶关系，即两个点之间如果有较大权值的边相连就认为它们比较相似；同时也考虑了二阶关系，即两个点或许不直接相连，但是如果它们的公共邻居节点比较多那么它们也会被认为是比较相似的。图 2-5 是一个信息网络，以它为例说明两种相似关系。该网络含有 10 个节点和若干条边，边的粗细代表了两个节点之间的权重，线越粗则节点间的权重越大。可以看出节点 6 和节点 7 之间的连线最粗，说明这二者之间的边权值较高，因此这两个节点具有高一阶相似性。另一方面，节点 5 和节点 6 之间虽然没

有边，但它们的邻居节点集高度重合，因此也可以认为节点 5 和节点 6 具有高二阶相似性。也就是说，通过 LINE 模型为“用户-标签”网络生成的向量中，具有同样标签偏好的用户向量应具有很高的相似度。

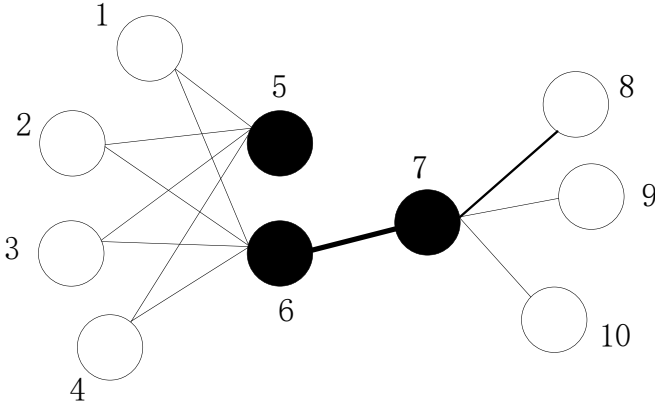


图 2-5 一阶关系和二阶关系

Node2vec<sup>[45]</sup>通过改变随机游走来采样的方式进一步扩展了 DeepWalk<sup>[37]</sup>算法。DeepWalk 是第一个基于 skip-gram 模型的网络嵌入方法。该方法模仿词嵌入模型中文本生成的过程，从网络上的任意一个节点出发并随机游走，通过这种方式生成节点的序列，然后利用 skip-gram 模型和 softmax 学习节点的向量表示。DeepWalk 从当前节点游走到下一个节点的概率是均匀随机的，而 Node2vec 引入了宽度优先搜索（BFS）和深度优先搜索（DFS）来对网络进行采样。基于 BFS 和 DFS 的方式提供了两种获取网络结构信息的视角。如图 2-6，使用宽度优先搜索会经过节点  $u$  周围的节点  $\{a, b, c\}$ ，这种方式能获取局部微观结构。使用深度优先搜索会经过  $\{d, e, g\}$ ，能获取全局宏观结构。

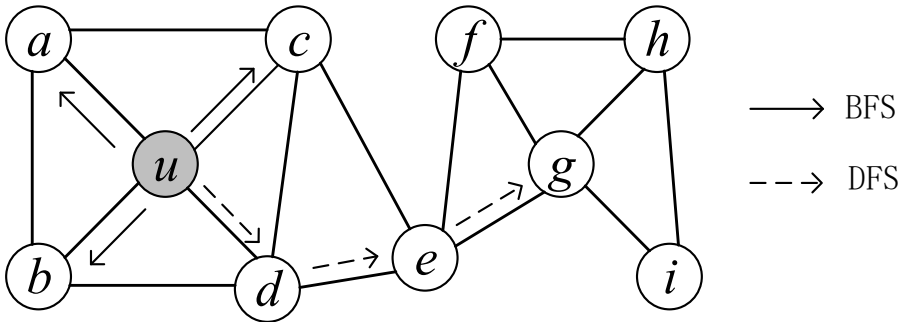


图 2-6 基于 BFS 和 DFS 的方式进行随机游走

本文利用网络嵌入方法从“用户-标签”网络学习出用户的向量，该向量具有低维、连续的特点，能同时包含标签信息和网络结构信息。在第3章和第4章中，本文使用该用户向量完成专家发现任务，并在第3章中通过实验比较 LINE 和 node2vec 这两种网络嵌入方法学习出的用户向量在专家发现任务中起到的效果。

## 2.4 实验设计及结果分析

### 2.4.1 数据集

Stack Exchange 是一系列具有相同问答模式的问答网站集合，其含有 133 个网站，含有计算机编程、数学、服务器、数据科学等不同领域的问题。经过了长足的发展，Stack Exchange 已经成为一个巨大的知识图书馆。Stack Overflow 是其中的第一个成员，其他的问答网站都根据 Stack Overflow 的模式而建立。在这种问答模式下，用户可以提出新问题并将回答设置成“最佳回答”，可以浏览并回答其他人提出的问题，也可以对其他问题的答案表示赞成或者反对。由于提问者在编写问题时需要输入该问题的标签，因此时至今日网站中包含大量的标签，用以反映问题的领域并且对问题分类。

本文的数据集来自 Stack Exchange 上的“Super User”和“Server Fault”这两个站点，并取 2010 年 01 月至 2018 年 07 月之间的数据进行实验。其中“Super User”数据集包含了 362386 个问题，这些问题共包含了 5259 个不同的标签；“Server Fault”数据集包含了 240752 个问题，共涉及到 3600 个不同标签。

根据标签在同一个问题中的共同出现的次数，建立标签共现矩阵，然后计算标签与标签之间的相似度矩阵，发现在“Super User”数据集上的 5259 个标签中有 956 个标签与其它标签存在相似关系，“Server Fault”数据集的标签集上有 626 个标签与其它标签相似。然后将这些相似标签抽取出来建立新的标签集，应用马尔可夫聚类算法进行相似标签的聚类。

### 2.4.2 评价标准

#### (1) Jaccard 系数

Jaccard 系数经常用于评价聚类结果中聚类簇之间的相似度，其定义为两个聚类簇的交集顶点与并集顶点的商。具体计算方式如式(2-9)所示， $c_i$ 表示顶点

$i$ 所在的聚类簇。可以看出，Jaccard 系数越小，聚类效果越好：

$$Jaccard = \sum_i \sum_j \left| \frac{c_i \cap c_j}{c_i \cup c_j} \right| \quad \text{式 (2-9)}$$

(2) 模块度

模块度经常用于评价聚类效果，模块度越大则表明聚类效果越好，各模块内顶点之间的联系越强，如式 (2-10) 所示。

$$Q = \sum_{ij} \frac{1}{2m} (A_{ij} - \sqrt{k_i \cdot k_j}) \delta(c_i, c_j) \quad \text{式 (2-10)}$$

其中 $m$ 为顶点的总数， $A_{ij}$ 为顶点 $i$ 和顶点 $j$ 之间的权重， $k_i$ 表示与顶点 $i$ 相连接的所有边的权值之和， $c_i$ 表示顶点 $i$ 所在的聚类簇，当两个聚类簇不同时， $\delta(c_i, c_j)$ 的值为 1，否则为 0。

2.4.3 实验结果分析

本文对相似标签融合效果进行评价，首先根据相似标签集构建概率矩阵，并使用马尔可夫聚类算法对标签进行聚类。在实验过程中，令随机游走步数 $k = 2$ 或 $k = 3$ 为参数，同时调整膨胀系数 $r$ 的值，最后得到以下实验结果。

表 2-2 不同膨胀系数下聚类效果对比

| 数据集             | $r$ | $k = 2$   |               |       | $k = 3$   |               |       |
|-----------------|-----|-----------|---------------|-------|-----------|---------------|-------|
|                 |     | 聚类簇<br>数目 | Jaccard<br>系数 | 模块度   | 聚类簇<br>数目 | Jaccard<br>系数 | 模块度   |
| Super<br>User   | 1.5 | 135       | 0.089         | 0.215 | 154       | 0.098         | 0.201 |
|                 | 2.0 | 202       | 0.067         | 0.421 | 226       | 0.075         | 0.36  |
|                 | 2.5 | 273       | 0.059         | 0.389 | 259       | 0.063         | 0.347 |
| Server<br>Fault | 1.5 | 89        | 0.103         | 0.128 | 91        | 0.145         | 0.108 |
|                 | 2.0 | 103       | 0.087         | 0.34  | 134       | 0.096         | 0.287 |
|                 | 2.5 | 159       | 0.066         | 0.305 | 148       | 0.099         | 0.261 |

当 $k = 2$ 时，平均 Jaccard 系数为 0.0785，平均模块度为 0.2997；当 $k = 3$ 时，平均 Jaccard 系数为 0.096，平均模块度为 0.2607。在随机游走步数 $k = 2$ 的状态

下, Jaccard 系数更小, 模块度更大, 因此聚类效果更好。

对于“Super User”上的相似标签集的聚类过程, 随着膨胀系数 $r$ 的增大, 聚类簇的数目也逐渐变多。当膨胀系数为 1.5 时, 产生 135 个聚类簇, Jaccard 系数为 0.089, 模块度为 0.215, 其中模块度值较小, 说明聚类簇内的各个顶点比较离散。当膨胀系数为 2.0 时, Jaccard 系数降低了 24.72%, 模块度提高了 1.95 倍。当膨胀系数增大为 2.5 时, Jaccard 系数相比于膨胀系数为 2.0 时的结果降低了 11.94%, 而模块度不增反降。综合比较不同膨胀系数下的 Jaccard 系数和模块度的差异, 当膨胀系数为 2.0 时, Jaccard 系数的下降速度降低, 模块度最大, 因此聚类效果最好。同样对于“Server Fault”上的相似标签集, 也是 $r$ 为 2.0 时聚类效果最好。

经过针对相似标签的融合后, 标签集中相似标签聚集在一起成为了新的标签, 标签集中的数量也得到了减少, 因此“用户-标签”网络中的因相似标签导致的噪声得到了减少。

## 2.5 本章小结

本章针对庞大的标签集难以利用的问题进行研究, 提出了一种标签相似度衡量标准, 并对相似标签集进行了聚类, 解决了标签中的过于细化和多命名问题。在“Super User”和“Server Fault”数据集上证明了该种聚类方法的有效性。然后通过探讨用户与标签之间的联系, 构建了以用户和标签为节点的异构网络, 应用网络嵌入模型学习出“用户-标签”网络中节点的向量表示, 这种向量同时包含网络结构信息以及标签信息, 能被后文提出的两种专家发现方法所利用。

## 第3章 基于深度学习的专家发现方法

快速增长的问答数据给传统方法带来了巨大挑战，而目前热门的深度学习方法对于数据的增长具有很好的适应性。基于以上的考虑，本章提出一种基于深度学习的专家发现方法（An Expert Finding Method Based on Deep Learning, EF-DL）。本章将从两个方面介绍提出的方法，首先介绍问题文本的数据清洗流程并使用 word2vec 生成问题向量，其次介绍神经网络的具体构成。最后通过在来自 Stack Exchange 的两个站点的数据集上进行实验，证明 EF-DL 的有效性。

### 3.1 问题描述与分析

互联网和信息技术的不断迭代，见证了如 Stack Exchange、知乎等问答社区的快速发展。在问答社区中，提问者通过将令自己最满意的回答设置为“最佳答案”以结束该问题的讨论，因此这类已解决的问题都有一个“最佳答案”与之对应，提供“最佳答案”的用户则为最佳答案者。

问答社区积累了体量庞大的问答数据，例如知乎，作为最受欢迎的中文问答社区之一，千万级别的用户活跃在该社区中，这些用户每时每刻都在产生关于各种话题的大量问答数据。然而，提出的新问题容易被淹没在众多问题中，被用户所忽略而得不到任何回答，同时回答者的专业知识水平良莠不齐，所擅长的领域也各有不同。因此，针对问题的专家发现方法被问答社区采用，准确的为新提出的问题找到能给出高质量回答的专家。另一方面，快速增长的问答数据给传统的方法带来了巨大挑战，一些研究者利用深度学习（Deep Learning）模型处理庞大的问答数据。

深度学习是近年来最热门的研究课题之一，被广泛应用于各种领域。深度学习通过建立多层神经网络模仿人脑内的神经元机制进行分析学习，在图像识别、机器翻译等领域表现突出。深度学习极度依赖数据，数据量越大则预测效果越好。如图 3-1，与传统的方法相比，可以看出深度学习随着数据量的增加优势变得明显。

基于以上的考虑，本章将结合深度学习模型提出基于深度学习的专家发现方法，下面将详细介绍所提出的方法。



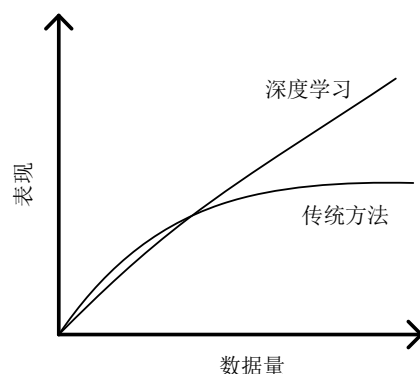


图 3-1 数据量对不同方法表现的影响

## 3.2 基于深度学习的专家发现方法

### 3.2.1 算法思想

图 3-2 展示了基于深度学习的专家发现方法。首先以用户和标签为节点，根据标签与标签之间的关系、用户与标签的关系构建“用户-标签”网络，利用上一章介绍的网络嵌入方法得到用户节点的向量表示。

其次，对于问题文本，本文将问题的标题、正文和附带的标签组合起来构成问题文本，文本进行若干数据清洗步骤得到包含问题重要信息的词序列，利用 word2vec 模型生成问题的向量表示。

然后应用深度学习模型从问答数据中训练出能使回答者的用户向量和他曾给出“最佳回答”的问题向量保持最大余弦相似度的参数。最后利用训练好的模型，预测专家列表。

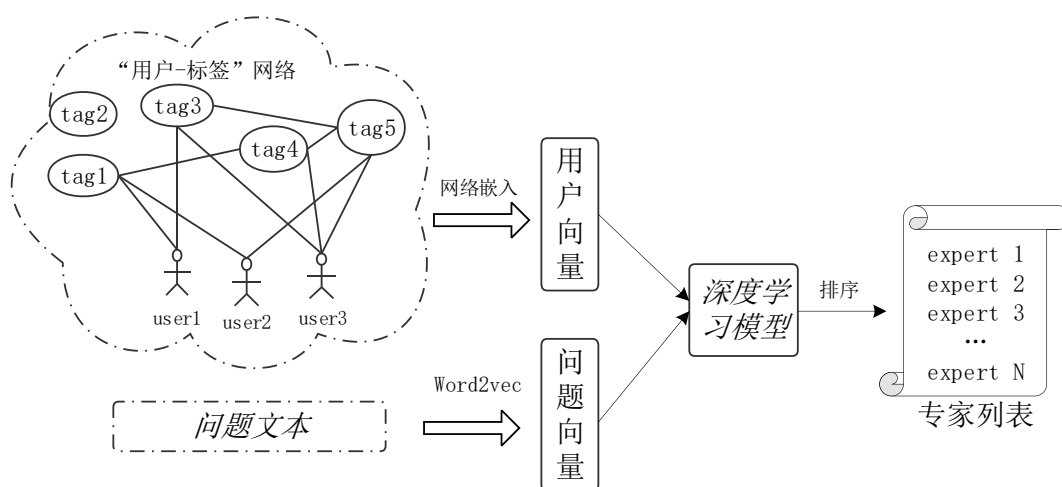


图 3-2 基于深度学习的专家发现方法

### 3.2.2 问题文本向量

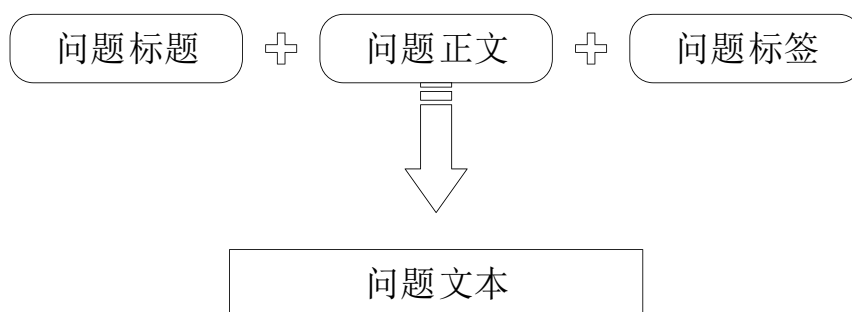


图 3-3 问题文本的组成

如图 3-3，本文将问题的标题、标签和正文组成起来构成问题的文本。由于文本中包含了大量与问题无关的信息，这些信息给问题文本带来了极大的噪声，因此需要对问题文本进行数据清洗。数据清洗包含以下步骤：

（1）移除 HTML 标签：问题正文包含了大量 HTML 标签以便于网页上的排版显示，但这些 HTML 标签和问题中的信息无关，因此需去除无用的标签，只保留含有重要信息的句子文本。

（2）停止词过滤：英文词汇中含有大量的冠词、介词、副词和连词，如“a”、“the”等。这些词出现频率很高，但重要性远远不如文本中的名词、动词，因此需要过滤掉文本中所有的停止词。

（3）去除代码段：移除被“<code>”和“</code>”包围的代码段。据文献<sup>[30]</sup>，移除代码段产生的噪声，能取得更好的结果。

（4）词干提取（stemming）：词干提取是去除词缀得到词根的过程，比起词根提取，词干提取的效率更高。例如，单词“fished”提取词干后为“fish”。

最后得到的问题文本仅由单词词干组成，这些词都包含着与问题相关的重要信息。

在自然语言处理任务中使用最广泛的词向量表示法是独热编码（one-hot）。One-hot 表示法将每一个词表示成长度固定的向量，每个词只有一个维度上的值为 1，其它维度上的值设置为 0。使用独热编码表示词向量的方式很简单，每个词都有自己的向量。然而在这种词向量表示法下，词与词之间是孤立的，它们没有任何相关度。此外，向量的维度取决于词库中词的总数，在庞大的词库下使用独热编码会导致维度灾难问题。因此在一些任务中，这种词向量表示法生成的高维向量会导致算法的复杂度难以接受。

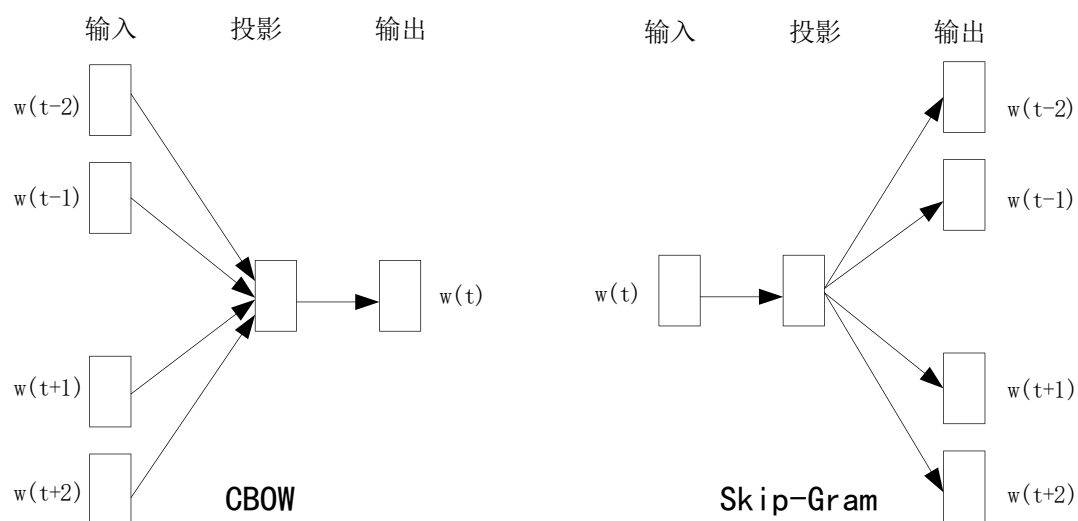


图 3-4 CBOW 模型和 Skip-Gram 模型

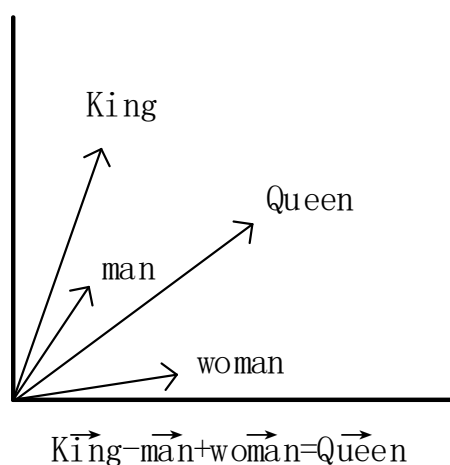


图 3-5 词向量的加法组合运算

Word2vec 是谷歌在 2013 年开源的词向量工具，它能在百万数量级的词库和上亿的文本集上进行高效地训练，并训练出低维度的词向量。它利用浅层神经网络训练词向量进而处理词与词之间的关系，得到的词向量能很好地度量词与词之间的相关性，例如“爸爸”和“父亲”这两个词意思相近，它们的词向量也是相关的，余弦相似度接近 1。Word2vec 具体的实现方式有两种，一种是连续词袋模型（Continuous Bag-of-Words, CBOW），另一种是 Skip-Gram 模型。如图 3-4

所示，CBOW 利用上下文预测目标词，而 Skip-Gram 用一个词来预测上下文。问答社区中的问题文本和回答文本大多是短文本，短文本的上下文信息的缺失不适合 CBOW 模型的训练模式，而 Skip-Gram 模型采用预测上下文信息的方式能适应短文本词向量的训练。Word2vec 的另一个特点为词向量能自由地进行加法组合运算，图 3-5 显示了几个词之间加法运算的一个例子。

利用 word2vec 的这些特点，问题文本的每一个词被转化成低维的词向量，最后将同一个问题中的所有词向量相加起来就得到了该问题的向量。

### 3.2.3 深度学习模型

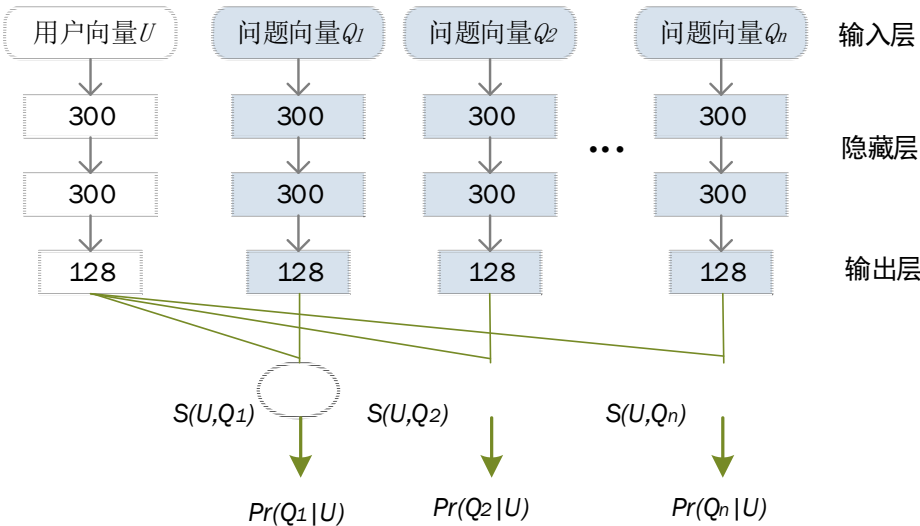


图 3-6 深度学习模型结构

如图 3-6 所示，本文应用深度学习模型来预测专家列表，该模型包含两个共享结构、但参数不同的全连接神经网络（DNN），第一个 DNN 输入为用户向量  $U$ ，第二个 DNN 输入为问题文本向量  $Q$ 。DNN 的隐藏层有两层，每层含有 300 个神经元，输出层含有 128 个神经元。最后是一个余弦相似度层，计算用户特征和每一个问题特征的相似度。

用户向量  $U$  和问题向量  $Q$  输入后，经过两个共享网络结构的 DNN，但是这两个 DNN 的权值矩阵  $W$  和偏置向量  $b$  不相互共享。其中隐藏层的定义如下：

$$h_1(k) = W_1(k) \cdot x \quad \text{式 (2-1)}$$

$$h_i(k) = F(W_i(k) \cdot h_{i-1}(k) + b_i) \quad \text{式 (2-2)}$$

$$F(x) = ReLU(x) = \max(0, x) \quad \text{式 (2-3)}$$

如式 (2-1) 和式 (2-2) 所示, 首先使用  $W_1$  乘以输入向量  $x$  得到能被后续隐藏层接受的值  $h_1$ , 然后下一层接受上一层的输出,  $W_i(k)$  为第  $i$  层的权值,  $b_i$  是第  $i$  层的偏置向量,  $k$  代表时间, 式 (2-3) 定义了激活函数  $F$ 。

输出层含有 128 个节点, 则输出的特征维度为 128。使用余弦函数来计算两个 DNN 分别输出的用户特征  $y_U$  和问题特征  $y_Q$  的相似度, 如式 (2-4) 所示:

$$S(y_U, y_Q) = \cosine(y_U, y_Q) = \frac{y_U^T \cdot y_Q}{\|y_U\| \cdot \|y_Q\|} \quad \text{式 (2-4)}$$

如式 (2-5) 所示, 一个用户能在多个问题下取得“最佳回答”, 设  $K$  为用户取得“最佳回答”的问题总数,  $r$  为从问题集中随机抽取的非该用户回答的问题数量。因此每一组数据包含一个用户、该用户取得最佳回答的问题集以及  $r$  个非该用户回答的问题。基于随机抽取从问题集中得到  $r$  个非该用户回答的问题, 本文中  $r$  设为 3。然后应用 *Softmax* 函数处理  $y_U$  和每一个  $y_Q$  的余弦相似度, 确保他们的概率总和为 1。

$$Pr(Q_k|U) = \frac{e^{S(y_U, y_{Q_k})}}{\sum_{k=1}^{K+r} e^{S(y_U, y_{Q_k})}} \quad \text{式 (2-5)}$$

式 (2-6) 定义了损失函数, 在训练过程中需要使 *Loss* 最小。当用户取得最佳回答的问题特征  $y_Q$  和用户特征  $y_U$  余弦相似度最大且非最佳回答问题特征  $y'_Q$  与用户特征  $y_U$  余弦相似度最小时, *Loss* 到达最小值。如果一个用户回答了许多的问题, 可以令  $K = 10$ , 将一组数据拆分为多组数据以便于神经网络的训练。

$$Loss = -\log(\sum_{k=1}^K Pr(Q_k|U)) \quad \text{式 (2-6)}$$

### 3.3 实验设计及结果分析

#### 3.3.1 数据集和数据预处理

与 2.5.1 节一样, 本文使用的是“Super User”和“Server Fault”的这两个站点的数据集, 其中取 2010 年 01 月到 2016 年 12 月之间的数据作为训练集, 取 2017 年 01 月到 2018 年 07 月之间的数据作为测试集。

首先根据时间段以及类型 (在数据源文件 *Post.xml* 中, 问题和回答的类型序

号分别是 1 和 2)，从数据源文件中得到问题集。然后剔除没有设置“最佳回答”的问题，最后根据时间节点将数据集切分成训练集和测试集，保持训练集与测试集的数据比例为十比一左右。问答社区中问题的回答质量良莠不齐，甚至存在恶意回答问题的情况。因此本文认为“最佳回答”得到了提问者的认可，是正确答案。表 3-1 显示了这两个数据集中的训练集问题数量和测试集问题数量。

表 3-1 训练集和测试集问题数量

| 数据集          | 训练集问题总数 | 测试集问题总数 |
|--------------|---------|---------|
| Super User   | 85675   | 9029    |
| Server Fault | 67325   | 5922    |

表 3-2 不同情况下的用户数、训练集或测试集问题数

| 数据集          | 用户集      | 用户数  | 训练集问题数 | 测试集问题数 |
|--------------|----------|------|--------|--------|
| Super User   | $U_5$    | 2424 | 55142  | 4092   |
|              | $U_{10}$ | 1141 | 46978  | 3675   |
|              | $U_{15}$ | 751  | 42375  | 3351   |
|              | $U_{20}$ | 548  | 38983  | 3216   |
| Server Fault | $U_5$    | 1854 | 46822  | 2560   |
|              | $U_{10}$ | 938  | 40950  | 2225   |
|              | $U_{15}$ | 635  | 37407  | 2051   |
|              | $U_{20}$ | 470  | 34655  | 1875   |

如表 3-2 所示，设 $N$ 为用户取得最佳回答的问题数，本文通过 $N \geq 5, 10, 15, 20$ 作为依据构造 4 个用户集 $U_N$ 。例如在表 3-2 所示的数据集“Server Fault”中，取得“最佳回答”数量至少为 20 的用户有 470 位，他们回答过的问题在训练集中有 34655 个，在测试集中有 1875 个，每一个问题的最佳回答者都属于这 470 位用户。随着参数 $N$ 的增大，可以看出用户数显著减少，但是他们回答的问题数却没有显著减少。据研究表明，数量仅为 0.5%的用户回答了超过 35%的问题（有答案的问题中）<sup>[15]</sup>，这些用户为问答社区做出了极大的贡献。因此可以通过参数 $N$ ，区分出专业度不同的用户，在不同的用户集上进行实验能全面的验证提出的专家发现方法的效果。

### 3.3.2 结果评价

#### 3.3.2.1 评价标准

为了比较公正的评价提出的方法,本文使用平均排序倒数(Mean Reciprocal Rank, MRR)指标。MRR 指标作为国际上通用的评价标准,经常被用于对搜索算法进行评价:对于查出来的结果列表,如果第一个结果匹配,那么分数为 1,第二个匹配则分数为 0.5,第 $n$ 个匹配则分数为 $1/n$ ,如果没有任何匹配,则 MRR 分数为 0。在本文中,提出的方法为测试集中的每一个问题预测相应的专家列表,如果该问题的最佳回答者出现在预测的专家列表的第 $n$ 个位置,那么此次 MRR 分数为 $1/n$ 。计算公式如下所示:

$$MRR = \frac{\sum_t^T rank_t}{T} \quad \text{式 (2-7)}$$

其中 $T$ 为问题数, $rank_t$ 表示第 $t$ 个问题下预测出的第一个正确专家的排序位置倒数。

#### 3.3.2.2 对比方法

提出的方法与四种不同类型的方法在两个数据集“Super User”和“Server Fault”上进行了对比实验。其中 STM 从用户文档以及问题文本中挖掘潜在主题信息,根据用户的潜在主题信息估算其专业知识,因此能计算得出新提出的问题能否被该用户回答。RankingSVM 将排序问题转化成分类问题,为新提出的问题生成候选专家列表。QR-DSSM 首先建立用户档案,从用户档案和问题文本中使用 DSSM 模型学习用户特征和问题特征。与该三类方法进行对比,能全方面地、客观地评价本文提出的专家发现方法的效果。此外将 2.3.3 介绍的两种网络嵌入方法 LINE<sup>[43]</sup>和 node2vec<sup>[44]</sup>用于“用户-标签”网络并学习出用户向量,EF-DL 分别使用这两种用户向量,然后比较这两种方法的优劣性。

(1) STM<sup>[15]</sup>: 一种基于分段主题模型的专家发现方法。通过分段主题模型提取文本的潜在主题,并比较问题文本和用户文档的相似性,按照相似度大小生成专家列表。

(2) RankingSVM<sup>[23]</sup>: 一种基于 RankingSVM 的专家发现方法。根据用户文档和问题的相关度分数从大到小对候选专家列表进行排序。

(3) QR-DSSM<sup>[30]</sup>: 通过 Trigram 表示词向量,使用 DSSM 模型提取语义特

征，即从用户文档、问题文本中分别提取用户特征、问题特征，并根据这两者的余弦相似度从高到低得到候选专家列表。

(4) EF-DL with node2vec: 使用 Nodec2vec 迭代“用户-标签”网络并生成用户向量。

(5) EF-DL with LINE: 使用 LINE 迭代“用户-标签”网络并生成用户向量。

### 3.3.3 实验结果与分析

本次实验的结果均在内存为 16.00GB、处理器为 *Inter(R) Core(TM) i7-6700HQ CPU@2.60GHz* 的计算机得出。在本次实验中，提出的方法与四种不同的模型进行了对比实验。对比实验结果如表 3-3 所示：

表 3-3 实验数据

| 数据集          | 用户集      | STM    | RankingSVM | QR-DSSM | EF-DL with node2vec | EF-DL with LINE |
|--------------|----------|--------|------------|---------|---------------------|-----------------|
| Super User   | $U_5$    | 0.0924 | 0.1098     | 0.1275  | 0.1123              | <b>0.1308</b>   |
|              | $U_{10}$ | 0.0991 | 0.1121     | 0.1331  | 0.132               | <b>0.1383</b>   |
|              | $U_{15}$ | 0.1095 | 0.1387     | 0.1565  | 0.1609              | <b>0.1614</b>   |
|              | $U_{20}$ | 0.1288 | 0.1625     | 0.189   | 0.1855              | <b>0.1959</b>   |
| Server Fault | $U_5$    | 0.0891 | 0.1084     | 0.123   | 0.1091              | <b>0.1269</b>   |
|              | $U_{10}$ | 0.0949 | 0.1155     | 0.1282  | 0.1132              | <b>0.1335</b>   |
|              | $U_{15}$ | 0.1054 | 0.1285     | 0.1508  | 0.1394              | <b>0.1563</b>   |
|              | $U_{20}$ | 0.1235 | 0.157      | 0.1831  | 0.1723              | <b>0.1923</b>   |

从表 3-3，图 3-7 和图 3-8 中可以看出，在数据集“Super User”和“Server Fault”上的四个用户集 ( $N \geq 5, 10, 15, 20$ ) 中，提出的方法均取得了更好的效果。

数据集“Super User”的用户量比数据集“Server Fault”要多出大约 18%到 30%，而各方法在“Super User”上的绝大多数结果要优于在“Server Fault”上得出的结果，说明数据量的提升对于各模型来说具有一定的提升。其中，EF-DL with LINE 在“Super User”上取得了最高的 MRR 分数，0.1959。

首先比较两种不同的网络嵌入方法的差异，可以看出 EF-DL with LINE 明显要优于 EF-DL with word2vec，说明了 LINE 能在“用户-标签”网络中学习出更



适合本文提出的方法的用户向量。

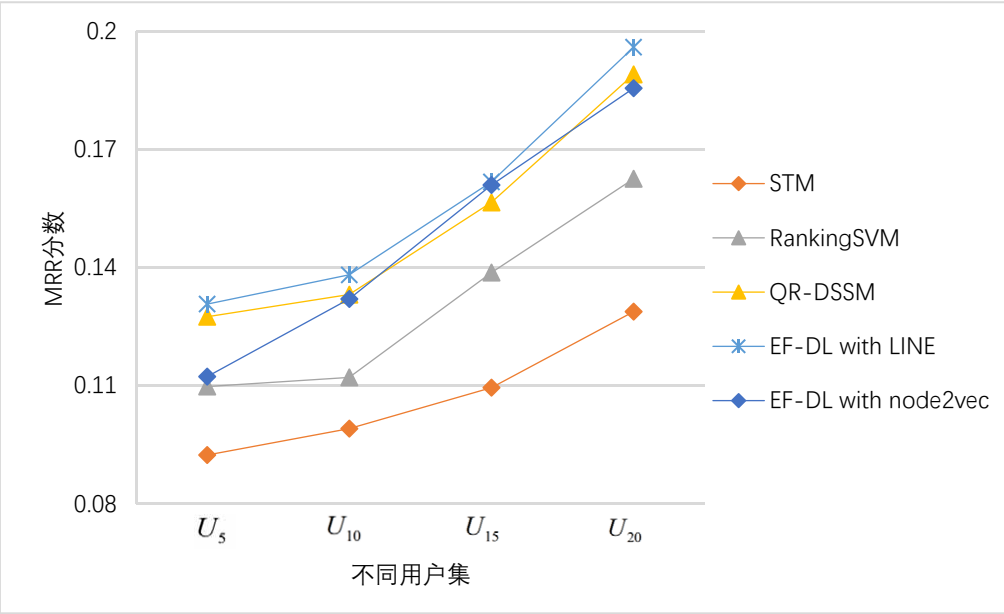


图 3-7 “Super User”数据集下实验结果

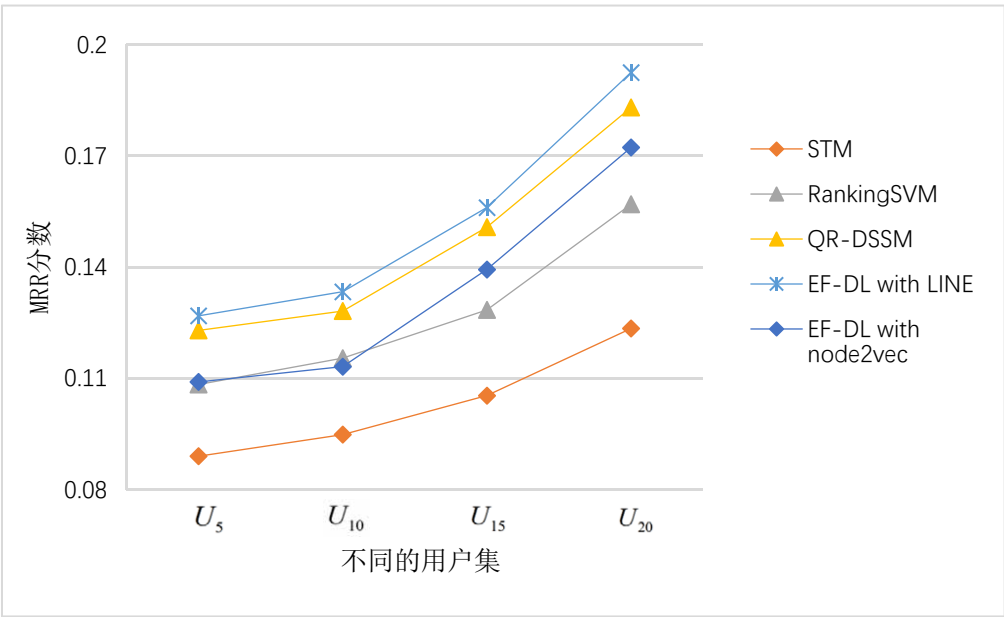


图 3-8 “Server Fault”数据集下实验结果

EF-DL with LINE 与基于主题模型的方法 STM 相比较，在两个数据上的平均 MRR 指标分别提高了 45%和 46%。这表明，传统的主题模型在短文本上难以

挖掘潜在主题信息，而本文提出的方法从网络中学习出的用户向量能包含更多的信息。与 **RankingSVM** 的专家发现方法相比，平均 **MRR** 指标分别提高了 19.8% 和 19.1%，与 **QR-DSSM** 相比，提出的方法在也提高了 2% 和 5% 之间。**QR-DSSM** 根据用户历史回答记录建立用户文档，问题附带的标签被当成普通的文本处理，重要的标签信息经过神经网络的学习逐渐被忽略。而本文提出的方法则从“用户-标签”网络中学习用户向量，从而使得用户向量同时包含网络结构信息以及标签信息，使用深度学习模型能更好地寻找用户与问题的映射关系。

综上所述，本文提出的方法能更准确地寻找到能提供正确答案的专家，使用 **LINE** 模型生成的用户向量能取得更好的效果。

### 3.4 本章小结

以上一章构建的“用户-标签”网络为基础，本文提出一种基于深度学习的专家发现方法。首先，使用 **Word2vec** 工具训练得到问题文本的低维度向量表示，然后应用深度学习模型分别提取用户特征和问题特征，最后根据这二者的余弦相似度生成专家列表。本文在来自 **Stack Exchange** 的两个站点的真实世界数据集上进行了充分的实验，实验结果表明，本文提出的专家发现方法优于其他方法，能为问题寻找到更合适的专家。

## 第 4 章 基于强化排序学习的专家发现方法

排序任务和专家发现任务有着千丝万缕的联系，从一定角度上，专家发现可被看成排序任务。本文将从排序的角度提出一种基于强化排序学习的专家发现方法(An Expert Finding Method Based on Reinforcement Learn to Rank, EF-RLTR)，针对近期越来越多的问答社区不再设置“最佳回答”以结束讨论的问答场景，通过一个问题有多个答案的问答数据进行强化排序学习模型的训练。本文将从两个方面详细介绍 EF-RLTR：首先介绍如何将专家发现任务形式化成马尔科夫决策过程，然后介绍如何使用策略梯度的方法学习模型的参数。最后在 Stack Exchange 问答数据集上对 EF-RLTR 进行了全方面的实验对比。

### 4.1 问题描述与分析

排序是推荐中经常能遇到的场景：针对不同的用户，如何进行个性化的候选产品的推荐排序，从而优化业务指标则成为了一个研究课题，为此产生了许多方法与模型。在大量数据的支撑下，可以通过一些流行的机器学习算法来实现排序任务的学习，比如利用传统模型计算出分数进行排序，又或者利用排序学习等更有针对性的排序算法。

近期越来越多的问答社区采用开放的问答模式，不再设置“最佳回答”以结束讨论，而是通过浏览者对回答进行评价的形式鼓励更多用户尽可能地深入探讨问题。在这种问答模式下，问题拥有若干个专家提供的高质量答案，根据获得的“赞同”数量区别答案质量的高低。也就是说，一个问题与若干个专家对应着。如果将问题视为“用户”，将专家视为“候选产品”，那么问答社区中的专家发现任务可以被视为排序任务，研究目标则成为了对问题的候选专家列表进行排序。

强化学习(Reinforcement Learning)算法<sup>[46]</sup>又称再励学习、评价学习，是机器学习领域中的一个重要分支。强化学习模型是一个从交互中不断学习，以达到预期目标的框架。如图 4-1，在强化学习模型中学习和决策的模块被称为智能体(agent)，与智能体相交互的模块被称为环境(environment)。在交互过程中，智能体选择执行某个动作，环境对智能体执行的动作做出反应，从而改变状态，使得智能体进入新的环境中。与此同时，环境给予智能体一定的回报奖励，而智能

体的目标就是在一段时间内最大化这些回报。强化学习模型善于控制一个能够在某个环境下自主行动的个体，通过和环境之间的互动，不断改进它的行为。强化学习问题包括学习如何做、如何将环境映射为行动，从而获得最大的奖励。在强化学习中，智能体不会被告知该执行什么动作，而是经过反复尝试，来学习能获得最大奖励的行为。一般情况下，行动不仅会影响当前的奖励，而且会影响下一个时间点的环境，因此也会影响后续所有的奖励。行动会影响到环境，环境又会影响到后续的行动，所以从本质上讲，强化学习是一个闭环控制问题。

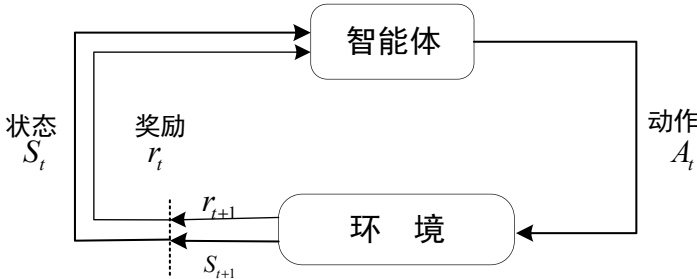


图 4-1 智能体与环境的交互

强化排序学习（Reinforcement Learning To Rank）<sup>[47,48]</sup>与强化学习算法提供了不一样的解决思路：它利用强化学习在环境和智能体之间进行交互式学习的特性，同时进行自身的学习和排序操作。强化排序学习最大的特点是实现了从“scoring and sorting”到“sequential decision making”的改变。如图 4-2 和图 4-3，以传统排序算法最常见的思路为例，模型输入为单个用户和单个候选产品的特征，输出为该候选产品对该用户的分值。在排序过程中，模型生成每个候选产品对该用户的分值，然后根据分值从大到小对候选者进行排序。总结整个过程则是先“scoring”然后再“sorting”。强化排序学习则考虑到两点：一是候选产品之间的相互关联；二是当前排序位置对分值的影响。从而，智能体每次在决定某个排序位置的候选产品时考虑当前所有已经排序好的候选产品对后续步骤的影响，因此能实现“sequential decision making”。

综上所述，在开放的问答模式下，一个问题与若干个专家对应着，因此专家发现任务可以被视为专家排序任务。本文利用这些特点提出一种基于强化排序学习的专家发现方法，下面将详细介绍所提出的方法。

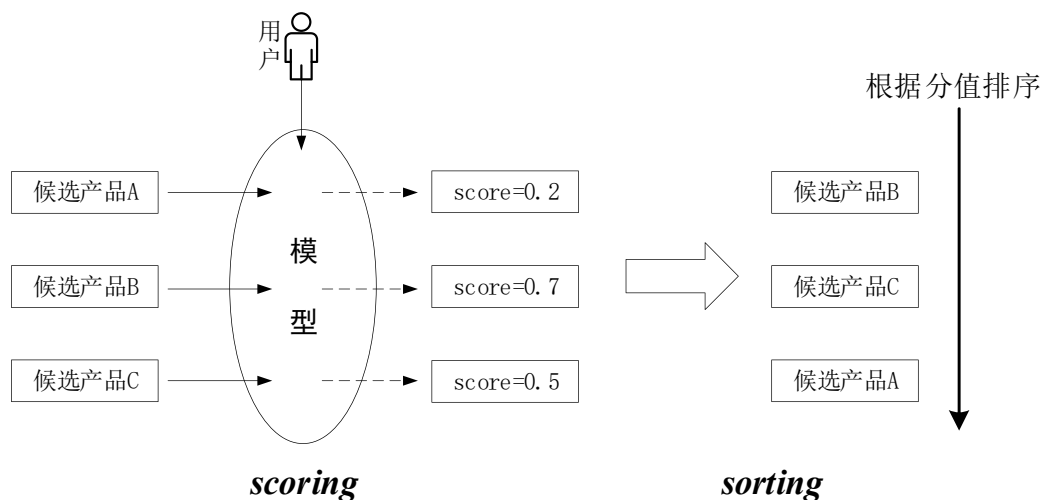


图 4-2 scoring and sorting 过程

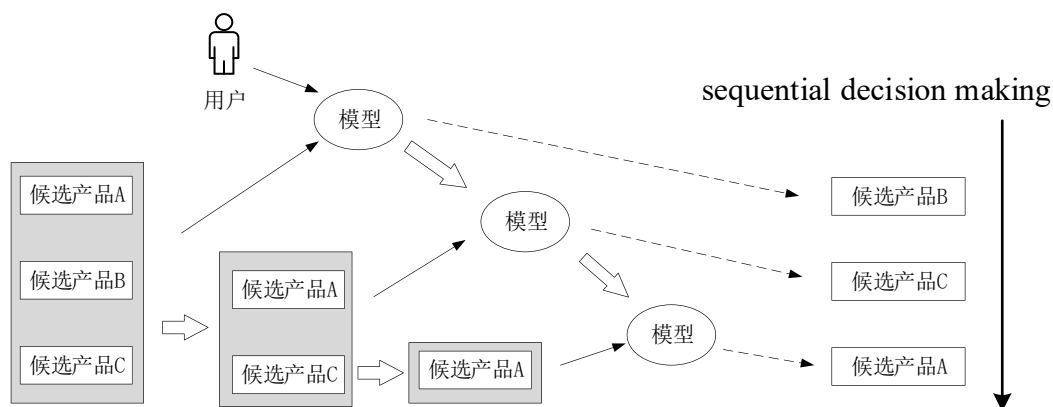


图 4-3 sequential decision making 过程

## 4.2 基于强化排序学习的专家发现方法

### 4.2.1 算法思想

给定问题集合  $Q = \{q_1, q_2, \dots, q_n\}$ , 用户集合  $U = \{u_1, u_2, \dots, u_m\}$ 。对于每一个问题, 系统依次从用户集中选择一位用户, 并将该用户置于系统预测的专家列表的末尾, 不断重复下, 最终得到专家列表。近年来, 越来越多的问答社区采用开放的问答模式, 不再设置“最佳回答”以结束讨论, 而是通过设置“赞同”与“反对”的形式鼓励更多用户尽可能地深入探讨问题。因此对于每一个问题, 根据专家在该问题下的回答获得的“赞同”数量从大到小排序, 为问题生成有序的回答者列表表示为  $\{q^{(n)}, C^{(n)}\}$ 。

基于以上的思想，本文提出一种基于强化排序学习的专家发现方法。如图 4-4，问答社区（Community Question Answering, CQA）中的问答数据用于强化排序学习模型的训练，每当有新的问题被提出时，CQA 系统利用当前的强化排序学习模型预测该问题的专家列表。这些专家在解决问题时产生的新一批的问答数据会被加入到模型参数的更新中。但模型的参数不是实时更新的，也就是说当有新的问答数据产生时，不会立即加入到模型的训练中。只有当问答社区积累了足够的新数据，才会重新训练模型以更新 CQA 中的参数。

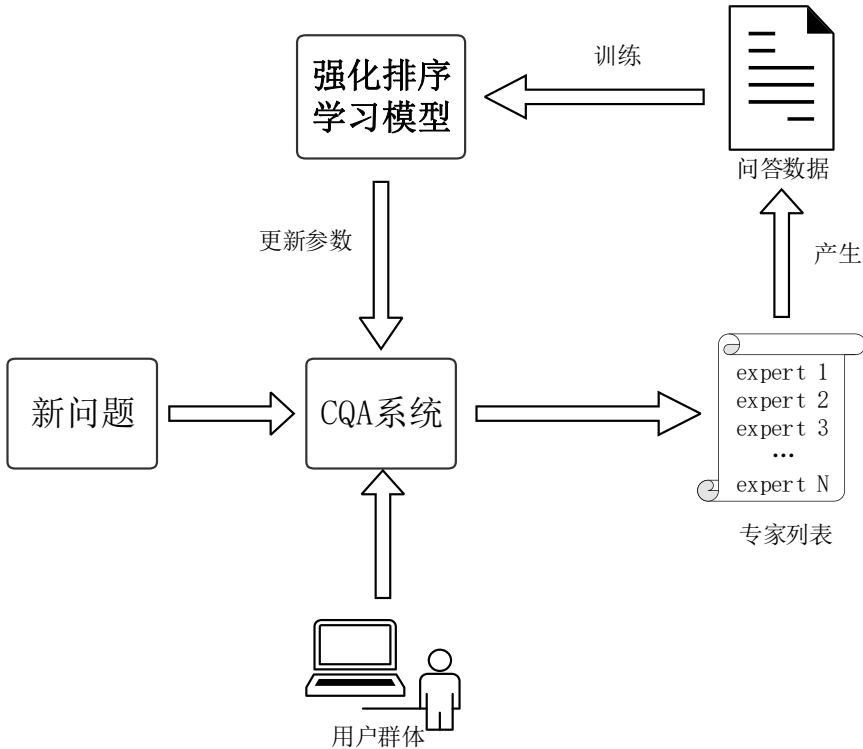


图 4-4 基于强化排序学习的专家发现方法框架

下面将从马尔科夫决策过程和策略梯度学习两个方面详细介绍该强化排序学习模型。

#### 4.2.2 马尔可夫决策过程

假设使用五元组  $M = (S, A, T, R, \pi)$  表示马尔科夫决策过程，分别为环境状态集  $S$ 、动作集  $A$ 、转移函数  $T$ 、回报函数  $R$  和策略  $\pi$ 。

(1) 环境状态  $S$ ：用于描述环境状态集。在为问题预测专家的过程中，智能

体每次从用户集中选择一位用户加入到预测的专家列表中，随着智能体的逐步操作，环境状态都在不断变化中。在预测过程中，环境状态包括预测专家列表 $Z_t$ ，剩余用户集 $U_t$ 和当前步骤 $t$ ，因此环境状态 $S_t$ 表示为 $[t, Z_t, U_t]$ 。

(2) 动作集 $A$ ：用于描述智能体可以执行的动作集。智能体可以执行的动作取决于当前环境状态 $S_t$ ，定义为 $A(S_t)$ 。在步骤 $t$ 时，智能体通过动作 $a_t \in A(S_t)$ 从剩余用户集 $U_t$ 中选择一位用户 $u_{m(a_t)}$ 排到预测的专家列表的第 $t+1$ 个位置，其中 $m(a_t)$ 表示动作 $a_t$ 选择的用户在剩余用户集 $U_t$ 中的位置， $u$ 为第2章“用户-标签”网络中学习出的用户向量。

(3) 转移函数： $T(S, A)$ 为转移函数 $S \times A \rightarrow S$ ，表示在当前环境状态 $S_t \in S$ 下，智能体选择一个动作 $a_t \in A(S_t)$ 后，转移到新的环境状态 $s_{t+1}$ 的过程。如式(4-1)所示，步骤 $t$ 时，智能体挑选了一个用户并将之加入到预测的专家列表 $Z_t$ 的第 $t+1$ 个位置上，状态从 $s_t$ 转变成 $s_{t+1}$ 。

$$s_{t+1} = T([t, Z_t, U_t], a_t) = [t+1, Z_t + u_{m(a_t)}, U_t - u_{m(a_t)}] \quad \text{式 (4-1)}$$

(4) 回报函数 $R(S, A)$ ：智能体的目的是通过选择获得最多的回报，根据回报的定义可知，在强化学习模型中定义合理的即时回报对于智能体来说是非常重要的。回报函数的定义如式(4-2)，根据回报函数计算出剩余用户集中每一位用户的即时回报，智能体以此选择出最优用户。其中 $C$ 为当前问题下的真实回答者列表， $Rank(C, u_{m(a_t)})$ 为该步骤下选择出的用户在真实回答者列表中的位置，如果该位置等于此时的步骤 $t$ ，那么其回报为1，否则为0。由于智能体总是倾向于执行能使回报奖励最大的动作，因此总能选择出正确的回答者放在正确位置上。

$$R(s_t, a_t) = f(x) = \begin{cases} 1, & \text{if } Rank(C, u_{m(a_t)}) = t \\ 0, & \text{otherwise} \end{cases} \quad \text{式 (4-2)}$$

(5) 策略 $\pi(a|s)$ ： $A \times S \rightarrow [0,1]$ 表示智能体在环境状态 $S_t \in S$ 下，选择某个动作 $a_t \in A(S_t)$ 执行的概率。智能体选择从剩余用户集中选择候选回答者的概率如式(4-3)所示，其中 $w \in R^K$ 为模型参数，维度与待排序用户特征向量维度一致，参数 $w$ 的学习将在下一节详细阐述。

$$\pi(a_t|s_t; w) = \frac{\exp\{w^T u_{m(a_t)}\}}{\sum_{a \in A(S_t)} \exp\{w^T u_{m(a)}\}} \quad \text{式 (4-3)}$$

### 4.2.3 策略梯度学习

早期的强化学习算法几乎都是“行动-价值”方法，这些方法首先学习每个行动在特定状态的价值，之后在所有状态下根据每个动作的估计价值进行选择。这种方法可以看成是一种“间接”的方法，因为强化学习的目标是如何决策，此类方法以每个动作的价值作为指标来辅助决策，是一种直观的、易理解的思维方式。但对于那些拥有高维度或连续状态空间来说，制定策略时需要比较各种行为对应的价值大小，从中比较得出一个有最大价值函数的行为是比较难的，因此适合于使用基于策略的学习。这种方法的核心思想在于直接学习决策，将决策过程视作为决策函数，拟合决策函数则成为了关键性问题。近年来深度学习的流行使得策略学习算法得到广泛应用，基于梯度的策略学习也成为了主流。

上一节内容提到一个重要参数 $w$ ，该参数决定了智能体在任意一个状态下执行某个动作的概率。本文使用策略梯度算法<sup>[49]</sup>来学习此参数，式（4-4）为性能函数，策略梯度算法的目标是最大化这个性能函数，即智能体希望通过最优策略获取最大的回报：

$$J(w) = \mathbb{E}_{Z \sim \pi_w} [G(Z)] \quad \text{式（4-4）}$$

$$Z = \{u_m(a_0), u_m(a_1), \dots, u_m(a_{|C|})\} \quad \text{式（4-5）}$$

其中， $Z$ 为模型预测的专家列表， $|C|$ 为当前问题的真实回答者列表的长度。 $G(Z)$ 为此次智能体预测专家列表的总回报，使用式（4-6）表示，其中 $r_t = R(s_{t-1}, a_{t-1})$ ，为第 $t$ 步时获得的回报， $\gamma$ 为折损因子。

$$G(Z) = \sum_{t=1}^{|C|} \gamma^{t-1} r_t \quad \text{式（4-6）}$$

梯度定义如式（4-7）所示：

$$\nabla_w J(w) = G_t \nabla_w \log \pi_w(a_t | s_t; w) \quad \text{式（4-7）}$$

由于强化学习的特性就是考虑长期回报而不只是短时间内的回报，所以定义第 $t$ 步时的长期回报为：

$$G_t = \sum_{k=1}^{|C|-t} \gamma^{k-1} r_{t+k} \quad \text{式（4-8）}$$



可以这么说， $G_t$ 会使参数 $w$ 趋于执行能取得最大长期回报的动作。其中，第 $t$ 步时，参数 $w$ 的梯度计算方式如式（4-9）所示：

$$\begin{aligned}\nabla_w \log \pi_w(a_t|s_t; w) &= \frac{\nabla_w \pi(a_t|s_t; w)}{\pi(a_t|s_t; w)} \\ &= u_{m(a_t)} - \frac{\sum_{a \in A_t} U_{m(a)} \exp\{w^T u_{m(a)}\}}{\sum_{a \in A_t} \exp\{w^T u_{m(a)}\}}\end{aligned}\quad \text{式 (4-9)}$$

下面给出策略梯度学习算法过程：

---

算法 4-1：策略梯度学习算法

---

输入：训练集 $D = \{q^{(n)}, C^{(n)}, U\}$ ，学习率 $\eta$ ，回报函数 $R$ ，折损因子 $\gamma$

输出： $w$

- 1: 使用随机数初始化参数  $w$
  - 2: 初始化 $\nabla_w$ 为 0
  - 3: for each  $(q, C, U)$  in  $D$ :
  - 4:     初始化经验池 $E$ ，令 $s_0 = [0, Z_0, U_0]$
  - 5:     for  $t = 0$  to  $|C|$  do
  - 6:         随机选择一个可执行的动作 $a_t \in A(s_t) \sim \pi(a_t|s_t; w)$
  - 7:         计算即时回报 $r_{t+1}$
  - 8:         保存经验元组 $(s_t, a_t, r_{t+1})$ 至 $E$ 中
  - 9:         更新剩余用户集 $U_t \leftarrow U_{t-1} - u_{m(a)}$
  - 10:         将用户 $u_{m(a_t)}$ 添加至末尾 $Z_t \leftarrow Z_{t-1} + u_{m(a_t)}$
  - 11:         更新环境状态 $s_{t+1} \leftarrow [t + 1, Z_t, U_t]$
  - 12:     end for
  - 13:     for  $t = 0$  to  $|C|$  do
  - 14:         计算未来总回报 $G_t$
  - 15:         更新 $\nabla_w$ ， $\nabla_w \leftarrow \nabla_w + \gamma^t G_t \nabla_w \log \pi_w(a_t|s_t; w)$
  - 16:     end for
  - 17: end for
  - 18: 更新 $w$ ，令 $w \leftarrow w + \eta \nabla_w$
  - 19: 重复 2 到 18 的过程直至 $w$ 收敛
  - 20: **Return**  $w$
-

## 4.3 实验设计及结果分析

### 4.3.1 数据集和数据预处理

Stack Exchange 问答网站集包含着各种领域的问答网站，每个站点有着相应领域内的大量问题。网站中的每个问题都有所属的若干个标签，这些标签由提问者创建，用来描述问题所属的领域。浏览者可以对问题下的回答表示赞同或反对，以此表明该回答是否有用。

本章使用的数据集来源于 Stack Exchange 问答网站集的子站点“Super User”。同样地，根据时间段将数据切分训练集和测试集：取 2010 年 01 月到 2016 年 12 月之间的数据作为训练集，取 2017 年 01 月到 2018 年 07 月之间的数据作为测试集。

对于每一个问题，本文根据回答获得的“赞同”数量从大到小对该问题下的所有回答进行排序，获得“赞同”数量较多的回答会排在前列，这些回答获得了广大用户的认可，因此可以认为该回答者提供了高质量的答案。有些问题由于提问者的疏忽，没有设置“最佳回答”或者在一些不再设置“最佳回答”的网站中，使用“赞同”量来评估回答是否是有用是很有必要的，获得“赞同”量最多的回答将会被排在首位。此外，剔除了获得“赞同”数少于 5 的回答，这些回答获得的认可度不高，本文认为该回答没有获得认可。最后，回答数少于 2 的问题也被过滤掉，回答数太少不适合本文模型的训练。

为了区分出专业度不同的用户，设 $N$ 为用户至少回答的问题数，本文以 $N \geq 10, 15, 20$ 作为依据构造 3 个用户集 $U_N$ ，如表 4-1 所示。训练集和测试集中问题的回答者都属于这三个用户集。例如，回答了至少 20 个问题的用户有 1326 位，训练集有 6567 个问题，测试集有 1178 个问题，这些问题的回答者都属于这 1326 位用户。

表 4-1 用户集、训练集和测试集

| 用户集      | 用户数  | 训练集问题数 | 测试集问题数 |
|----------|------|--------|--------|
| $U_{10}$ | 3920 | 19781  | 3287   |
| $U_{15}$ | 2053 | 10240  | 1709   |
| $U_{20}$ | 1326 | 6567   | 1178   |

### 4.3.2 结果评价

#### 4.3.2.1 评价标准

为了评价本文提出的专家发现方法的效果，本章使用 MAP 以及 S@N 作为实验结果的评价指标。

(1) 平均准确率 (Mean Average Precision, MAP)

MAP 用来衡量系统预测的专家列表命中的用户的排名情况，命中的用户越多、排名越靠前，则专家发现模型的效果越好。计算方式如式 (4-10)：

$$MAP = \frac{\sum_{q=1}^T AP(q)}{T} \quad \text{式 (4-10)}$$

其中  $T$  为问题数， $AP(q)$  为问题  $q$  下的平均准确率。在信息检索中，单纯地用准确率 *precision* 和召回率 *recall* 都不能全面的评价算法，于是就有了 AP 值的概念。在本章中，AP 的计算公式如下：

$$AP = \frac{\sum_{t=1}^{|U|} \frac{t}{rank_{U_t}}}{L} \quad \text{式 (4-11)}$$

其中， $U$  为命中的用户集， $L$  为问题的真实回答者数量， $rank_{U_t}$  为用户  $U_t$  在系统预测的用户列表的位置。例如，某个问题有 4 个回答者，系统预测的专家列表中命中了其中 3 个回答者，这三个回答者在预测列表中的位置分别是 1, 3, 6，因此其平均准确率为  $(1/1 + 2/3 + 3/6)/4 = 0.54$ 。

(2) S@N

如果问题下的获得“赞同”最高的回答者出现在预测列表的前  $N$  个用户中，那么此次预测是成功的，即 S@N 的值为 1。这种评价排序质量的方法被称为 S@N，S@N 值越大表明预测效果越好。在本次实验中，将使用 S@1、S@3 和 S@5 作为评价指标。

#### 4.3.2.2 对比方法

在本章的实验中，提出的方法与其他几种方法在数据集“Super User”上进行了对比实验：

(1) STM<sup>[15]</sup>：一种基于分段主题模型的专家发现方法，利用分段主题模型从文本中挖掘潜在主题，并比较问题文本和用户文档的相似性，按照相似度大小生成专家列表。

(2) RankingSVM<sup>[23]</sup>: 一种基于 RankingSVM 的专家发现方法。根据用户文档和问题的相关度分数从大到小对候选专家列表进行排序。

(3) QR-DSSM<sup>[30]</sup>: 基于 DSSM 的专家发现方法。通过 Trigram 表示词向量, 使用 DSSM 模型提取语义特征, 即从用户文档、问题文本中分别提取用户特征、问题特征, 根据这两者的余弦相似度从高到低得到候选专家列表。

(4) EF-DL: 第 3 章提出的一种基于深度学习的专家发现方法。实验证明了使用 LINE 迭代“用户-标签”网络能有更好的结果, 因此本次使用 LINE 迭代网络生成用户向量。

(5) EF-LRTR: 本章提出的方法。其中实验过程中用到的参数设置为, 学习率  $\eta = 0.01$ , 折损因子  $\gamma = 0.9$ 。

### 4.3.3 实验结果与分析

本次实验的结果均在内存为 16.00GB、处理器为 *Inter(R) Core(TM) i7-6700HQ CPU@2.60GHz* 的计算机得出。在本次实验中, EF-LRTR 与五种不同的方法进行了对比实验, 本节将从平均准确率和 S@N 这两个方面验证本文提出的方法的有效性。首先是平均准确率, 实验结果如表 4-2 所示:

表 4-2 使用 MAP 评价预测结果

| 用户集      | STM    | RankingSVM | QR-DSSM | EF-DL  | EF-LRTR       |
|----------|--------|------------|---------|--------|---------------|
| $U_{10}$ | 0.0401 | 0.0442     | 0.0588  | 0.0591 | <b>0.062</b>  |
| $U_{15}$ | 0.0465 | 0.0543     | 0.0671  | 0.0664 | <b>0.0695</b> |
| $U_{20}$ | 0.056  | 0.0616     | 0.0785  | 0.0786 | <b>0.0812</b> |

从表 4-2 可以看出, RankingSVM 要优于基于 STM 的方法, 这是因为传统的主题模型方法无法克服短文本难以提取潜在主题信息的缺点, 而基于机器学习的方法对于大数据集能充分发挥其自身优势, 数据量越大则训练的效果越好。

后三种方法 QR-DSSM、EF-DL 和 EF-LRTR 的平均准确率远远超过前两种较为传统的方法, 提升幅度超过 30%。其中, EF-LRTR 在三种用户集上均取得了最好的结果。与 QR-DSSM 相比, EF-LRTR 提升了大约 3% 的准确率, 由此能说明 EF-LRTR 在预测问题的若干个回答者上能取得最好的效果。

表 4-3 使用 S@1、S@3 和 S@5 评价预测结果

| 用户集      | Metric | STM    | RankSVM | QR-DSSM | EF-DL         | EF-LRTR       |
|----------|--------|--------|---------|---------|---------------|---------------|
| $U_{10}$ | S@1    | 0.0912 | 0.097   | 0.1081  | <b>0.1097</b> | 0.1085        |
|          | S@3    | 0.1623 | 0.1655  | 0.177   | <b>0.182</b>  | 0.1816        |
|          | S@5    | 0.2236 | 0.2279  | 0.2334  | <b>0.2375</b> | 0.237         |
| $U_{15}$ | S@1    | 0.0971 | 0.1002  | 0.1113  | <b>0.1143</b> | 0.1142        |
|          | S@3    | 0.187  | 0.1912  | 0.1932  | <b>0.1977</b> | 0.1943        |
|          | S@5    | 0.2467 | 0.248   | 0.2501  | 0.2566        | <b>0.2568</b> |
| $U_{20}$ | S@1    | 0.1087 | 0.1104  | 0.1186  | <b>0.1206</b> | 0.1203        |
|          | S@3    | 0.2166 | 0.2189  | 0.2205  | 0.2245        | <b>0.2253</b> |
|          | S@5    | 0.288  | 0.2896  | 0.2956  | <b>0.3015</b> | 0.298         |

MAP 指标能评价方法预测相关专家列表的能力，但不能评价预测列表中的第一个专家是否是最相关的，因此本文通过 S@1、S@3 和 S@5 这三种标准评估了这五种方法在预测最相关的回答者方面上的表现，实验结果如表 4-3 所示，分析结果如下：

(1) 与传统的基于 STM 的专家发现方法还有 RankingSVM 相比较，后面三种方法取得了较高的 S@N 值，在专家发现任务上能预测更准确的专家。

(2) 如图 4-5 所示，首先分析用户集  $U_{10}$ ，即至少回答了 10 个问题的用户所构成的集合。其中，横坐标为 S@1、S@3 和 S@5，纵坐标为 S@N 值。可以看出，EF-DL 取得了最好的结果，EF-LRTR 的结果稍低。分析其原因可知，EF-DL 以单个问题和单个用户作为输入训练模型，而 EF-LRTR 以单个问题和多个用户作为输入训练模型，故在预测最相关的回答者上 EF-DL 能表现更优。

(3) 如图 4-6 所示，横坐标为三个用户集，纵坐标为 S@5 值。可以看出，EF-LRTR 在用户集  $U_{20}$  上的结果最好。用户集根据用户回答过的问题数量而细分，随着门槛的提高，用户集越来越小， $U_{20}$  含有 1326 位用户，每一位都是回答过至少 20 个问题的专家。由此说明，在“小”而专业的用户集上系统能预测出最好的结果。

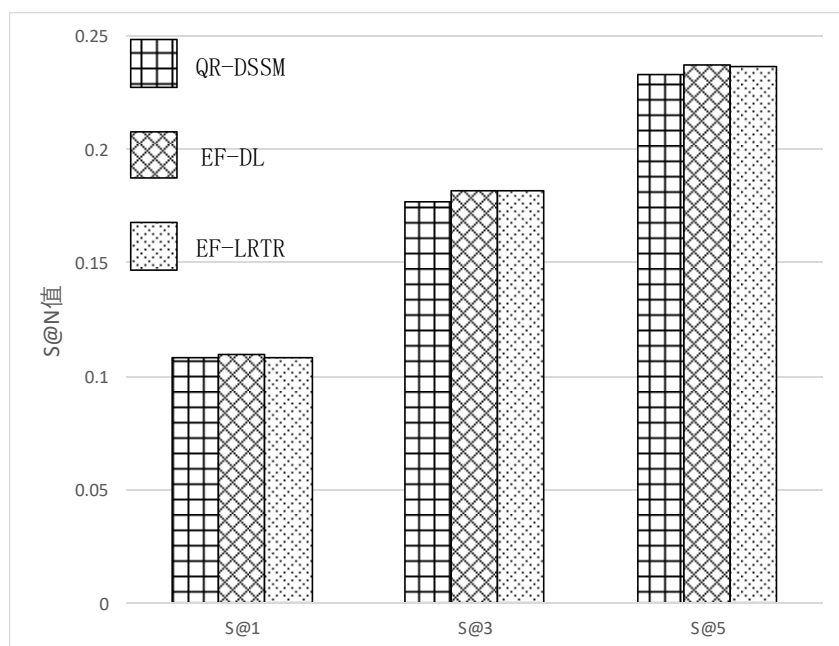


图 4-5 用户集 $U_{10}$ 下的 S@1、S@3 和 S@5 结果

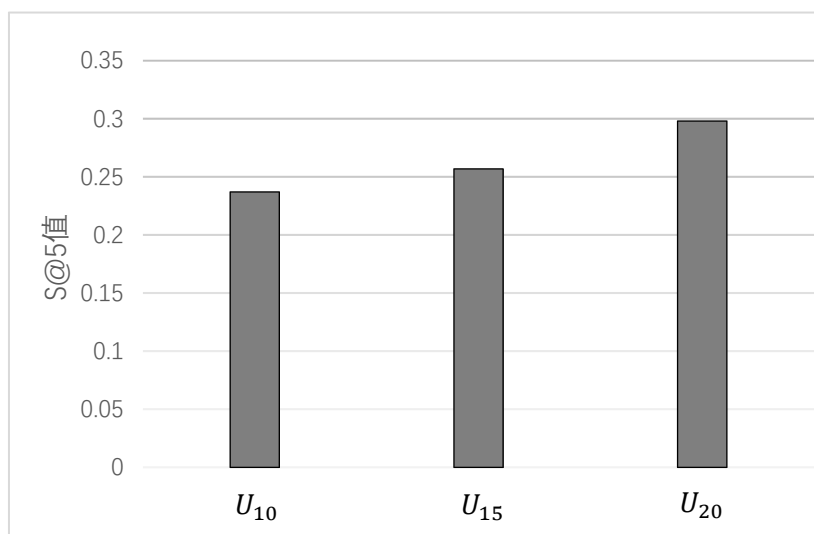


图 4-6 EF-LRTR 在三个用户集上的 S@5 值

最后，从训练时间方面比较各算法，结果如图 4-7 所示。其中，基于 STM 的方法花费了最长的时间，20 个小时；QR-DSSM 以及 RankingSVM 的训练时间比较短，分别是 10 个小时和 12 个小时；EF-RLTR 则花费了 13 个小时，EF-DL

花费了 18 个小时，由此可知，本文提出方法在训练时间方面没有取得优势。

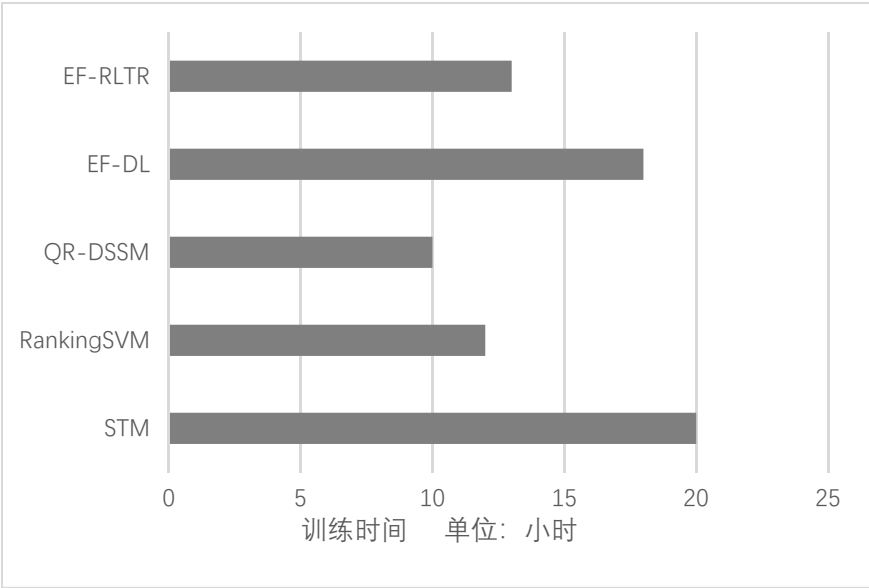


图 4-7 各方法的训练时间

综上所述，本章从  $S@N$ 、MAP 以及训练时间指标上对提出的方法进行了全方面的对比。其中，EF-RLTR 的训练时间较长，在  $S@N$  指标上取得了不错的效果，在 MAP 指标上均要优于其他方法。

#### 4.4 本章小结

本章从排序的角度对问答社区中的专家发现方法进行了研究，提出了一种基于强化排序学习的专家发现方法（EF-RLTR）。首先将专家发现任务形式化成马尔科夫决策过程，然后利用策略梯度方法学习模型的参数。最后在 Stack Exchange 问答数据集上对 EF-RLTR 进行了全方面的实验对比，评价指标包括 MAP、 $S@N$  以及训练时间，各项指标都证明了 EF-RLTR 的有效性。

## 第 5 章 总结与展望

### 5.1 总结

问答社区具备着搜索引擎所没有的提供高质量回答的能力，因此取得了快速的发展，逐渐演变成了一种提供问答服务的知识共享平台。问答社区主要包含用户、问题和回答这三个基本要素，其中用户又可细分为提问者、回答者和浏览者。在问答社区中提问者提出自己的问题，回答者给出解答，浏览者浏览问题并对回答给出自己的评价。这种互动式的问答模式受到了用户的喜爱，因此吸引了大量用户的踊跃参与。问答社区的用户来自各个行业，专业知识领域各不相同，水平也有所差别，既有大量的“小白”用户，也有能提供高质量回答的专家用户。这些专家用户具有高度专业知识，能快速地解决问题，同时给出的高质量答案也能极大的提高社区问答数据质量，也能给浏览者带来极大的阅读满足感。

本文分析了 2009 年到 2015 年之间的问答数据，发现随着问题的逐渐增长，问题被回答的百分比却在加速下降。这无疑会影响问答社区的问答质量，也会给提问者产生“不信任”感，导致用户的流失。因此为了社区的长足发展，问答社区需要一种方法将问题及时推送给具有丰富的专业知识且能提供高质量答案的专家，此类方法被称为专家发现方法。本文针对给新提出的问题寻找能提供高质量答案的专家这一任务进行了研究，具体工作如下：

#### （1）“用户-标签”网络。

问答社区中问题有着丰富的标签信息，由于提问者在创建问题时难以获得完美的统一，导致了标签过于细化问题和多命名问题。本文提出一种标签相似度衡量的方法，结合马尔可夫聚类算法融合相似的标签，该标签聚类方法的有效性在 Stack Exchange 数据集上进行了实验验证。然后进一步探讨用户与标签之间的关系，构建出一种以用户和标签作为节点的网络，并应用网络嵌入方法迭代网络学习出用户向量，使得用户向量能同时含有标签信息和结构信息。

#### （2）基于深度学习的专家发现方法。

针对问题设有“最佳回答”的特点，本文提出一种基于深度学习的专家发现方法。首先将问题的标题、正文和标签组合起来建立问题文本，经过若干数据清



洗步骤得到了仅由重要词干组成的词序列，利用 word2vec 词向量模型得到问题向量。然后构造两个结构相同、参数不同的 DNN 分别从用户向量和问题向量中提取特征，并根据余弦相似度的大小排序以预测专家列表。最后在 Stack Exchange 数据集上进行实验，证明了提出的算法的有效性。

### （3）基于强化排序学习的专家发现方法。

近期越来越多的问答社区采用开放的问答模式，不再设置“最佳回答”以结束问题的讨论，而是通过浏览者对问答进行评价的形式鼓励更多用户尽可能地深入探讨问题。在这种问答模式下，问题拥有若干个专家提供的高质量答案，基于深度学习的专家发现方法在准确寻找这若干个专家上表现出了不足。因此本文从专家排序的角度提出一种基于强化排序学习的专家发现方法，首先将专家发现任务形式化为马尔科夫决策过程，然后利用策略梯度方法学习模型的参数。最后在 Stack Exchange 问答数据集上进行了全方面的实验对比，评价指标包括 MAP 和 S@N，各项指标都表明了提出的算法的有效性。

## 5.2 展望

问答社区中针对问题的专家发现方法是一项复杂的研究，本文的工作还具有一定的不足，有很多的地方值得改进或者需要进一步的研究：

（1）标签相似等级的判定根据经验确定，这种人工的判定方法有时候不能找到一个完美的阈值，因此在如何寻找最合适的阈值以区分相似标签和非相似标签上需要做进一步的研究。

（2）专家回答问题具有极大的随机性，会受到情绪或者个人时间的影响。可能在一段时间内，专家即使收到了系统的问题推送，也不会去回答问题。因此需要综合分析专家的近一段时间内的活跃程度，以此评估专家目前主动回答问题的概率。

（3）问题的难易程度不同，有些专家倾向于回答高难度问题同时对简单问题不感兴趣，另一些专家则倾向于为简单问题解答。因此在未来的工作中，需要区分问题的难易度，以分析专家的难易倾向。

（4）本文提出的专家方法均立足于专家的历史表现，如果一个专家来到了一个新的问答社区，尽管他在上一个社区中被认定为权威，但在新的社区中由于没有任何回答记录，会被视作为新用户。因此，针对迁移专家的认定问题需要做进一步的研究。

## 致谢

秋去春来，转眼间在武汉理工大学三年的研究生求学生涯即将画上句号，感慨良多。无论是在学习上还是在生活中，都有太多的人给予了我极大的帮助，我才能度过一次次难关。借这次撰写论文的机会，表达对他们的感激之情。

首先我要感谢我的导师刘永坚教授，三年以来，刘老师不仅在学术科研方面对我进行耐心指导，而且在生活方面给予我无微不至的关怀和帮助。让我印象最深的是刘老师严于律己的态度以及张弛有度的处事风格，这将是人生道路上的指明灯。其次感谢解庆老师，解老师在我的论文选题、写作、修改过程中一直耐心指导，并给了我许多改进的意见以及新的研究思路。虽然在论文撰写的过程中遇到了很多的挫折和困难，但在两位老师的帮助下，我始终坚持了下来。当论文撰写完毕，我感觉一切的付出都是值得的。在此祝愿刘老师、解老师身体健康，桃李满天下。

此外，感谢数字传播中心的全体同学以及各位前辈们。在这里，我们一起研究工程界的难题，也共同交流学术上的各种问题，让我深深地体会到团队的力量。在中心的每一天我都收获颇丰，不仅极大的提高了科研水平，同时工程技术也得到了全面的锻炼。

还要感谢一直在背后默默付出并给予我支持的家人，他们为我提供了良好的生活条件和无私的关爱，使我能够没有顾虑地追求自己的学业。感谢我可爱的室友，当分别即将来临，因每天朝夕相处而显得无趣的日子才显得深刻，因重复太多次而显得平常的关怀才难以忘却。

最后感谢能在百忙之中抽出时间审阅论文的评委老师们。

## 参考文献

- [1] 姜雯, 许鑫. 在线问答社区信息质量评价研究综述[J]. 数据分析与知识发现, 2014, 30(6): 41-50.
- [2] 贾佳, 宋恩梅, 苏环. 社会化问答平台的答案质量评估——以“知乎”、“百度知道”为例[J]. 信息资源管理学报, 2013, 3(2): 19-28.
- [3] 延霞, 范士喜. 基于问答社区的海量问句检索关键技术研究[J]. 计算机应用与软件, 2013, 30(7): 315-317.
- [4] 李丹. 中美网络问答社区的对比研究——以 Quora 和知乎为例[J]. 青年记者, 2014(26): 19-20.
- [5] Adamic L A, Zhang J, Bakshy E, et al. Knowledge sharing and yahoo answers: everyone knows something[C]. Proceedings of the 17th international conference on World Wide Web, 2008: 665-674.
- [6] 刘佩, 林如鹏. 网络问答社区“知乎”的知识分享与传播行为研究[J]. 图书情报知识, 2015(6): 109-119.
- [7] Lin S, Hong W, Wang D, et al. A survey on expert finding techniques[J]. Journal of Intelligent Information Systems, 2017, 49(2): 255-279.
- [8] Yuan S, Zhang Y, Tang J, et al. Expert Finding in Community Question Answering: A Review[J]. ArXiv: Information Retrieval, 2018.
- [9] 刘高军, 马砚忠, 段建勇. 社区问答系统中“问答对”的质量评价[J]. 北方工业大学学报, 2012, 24(03): 31-36.
- [10] Blei D M, Ng A Y, Jordan M I. Latent dirichlet allocation[J]. Journal of Machine Learning Research Archive, 2003, 3: 993-1022.
- [11] 熊大平, 王健, 林鸿飞. 一种基于 LDA 的社区问答问句相似度计算方法[J]. 中文信息学报, 2012, 26(5): 40-46.
- [12] 国琳, 左万利. 基于兴趣图谱的用户兴趣分布分析及专家发现[J]. 电子学报, 2015, 43(8): 1561-1567.
- [13] 林鸿飞, 王健, 熊大平, 等. 基于类别参与度的社区问答专家发现方法[J]. 计算机工程与设计, 2014, 35(1): 333-338.
- [14] 李科霖. 一种融合话题和行为的在线问答社区领域专家发现方法[J]. 计算机与现代化, 2018, 0(09).

- [15] Riahi F, Zolaktaf Z, Shafiei M, et al. Finding expert users in community question answering [C]. Proceedings of the 21st International Conference on World Wide Web. New York, NY, USA: ACM, 2012: 791-798.
- [16] Rosen-Zvi M, Griffiths T, Steyvers T, et al. The author-topic model for authors and documents [C]. Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence. Banff, Canada: AUAI Press Arlington, 2004: 487-494.
- [17] Yan X, Guo J, Lan Y, et al. A biterm topic model for short texts[C]. Proceedings of the 22nd international conference on World Wide Web, WWW 2013: 1445-1456.
- [18] Wang Y, Liu J, Qu J, et al. Hashtag Graph Based Topic Model for Tweet Mining[C]. Proceedings of the 2014 IEEE International Conference on Data Mining, 2014: 1025-1030.
- [19] Manandhar S. Tag-based expert recommendation in community question answering[C]. IEEE/ACM International Conference on Advances in Social Networks Analysis & Mining. IEEE, 2014: 960-963.
- [20] Cheng X, Zhu S, Chen G, et al. Exploiting user feedback for expert finding in community question answering[C]. Proceedings of 2015 IEEE International Conference on Data Mining Workshop (ICDMW). Washington, DC, USA: IEEE Computer Society, 2015: 295-302.
- [21] Cheng X, Zhu S, Su S, et al. A Multi-Objective Optimization Approach for Question Routing in Community Question Answering Services[J]. IEEE Transactions on Knowledge and Data Engineering, 2017, 29(9): 1779-1792.
- [22] 何清, 李宁, 罗文娟, 等. 大数据下的机器学习算法综述[J]. 模式识别与人工智能, 2014, 27(04): 327-336.
- [23] Ji Z, Wang B. Learning to rank for question routing in community question answering[C]. Proceedings of the 22nd ACM International Conference on Information & Knowledge Management. San Francisco, California, USA: ACM, 2013: 2363-2368
- [24] Yan Z, Zhou J. Optimal answerer ranking for new questions in community question answering[J]. Information Processing & Management, 2015, 51(1): 163-178.
- [25] Lecun Y, Bengio Y, Hinton G. Deep learning[J]. Nature, 2015, 521(7553): 436.
- [26] 张军阳, 王慧丽, 郭阳, 等. 深度学习相关研究综述[J]. 计算机应用研究, 2018, 35(07): 1921-1928.
- [27] Zhou G, Zhou Y, He T, et al. Learning semantic representation with neural networks for community question answering retrieval[J]. Knowledge-Based Systems, 2016, 93: 75-83.
- [28] Zhang K, Wu W, Wang F, et al. Learning Distributed Representations of Data in Community Question Answering for Question Retrieval[C]. ACM International Conference on Web Search and Data Mining. 2016: 533-542.

- [29] Hsu W N, Zhang Y, Glass J. Recurrent Neural Network Encoder with Attention for Community Question Answering[J]. *Computation and Language*, 2016.
- [30] Azzam A, Tazi N, Hossny A. A Question Routing Technique Using Deep Neural Network for Communities of Question Answering [C]. *International Conference on Database Systems for Advanced Applications*. Cham: Springer, 2017: 35-49
- [31] Huang P S, He X, Gao J, et al. Learning deep structured semantic models for web search using clickthrough data[C]. *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*. New York, NY, USA: ACM, 2013: 2333–2338
- [32] Wang J, Sun J, Lin H, et al. Convolutional neural networks for expert recommendation in community question answering[J]. *Science China Information Sciences*, 2017, 60(11): 19-27
- [33] Zhou G, Lai S, Liu K, et al. Topic-sensitive probabilistic model for expert finding in question answer communities[C]. *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 2012: 1662-1666.
- [34] Shen J, Shen W, Fan X. Recommending Experts in Q&A Communities by Weighted HITS Algorithm [C]. *International Forum on Information Technology and Applications*. 2009: 151-154.
- [35] Yang L, Qiu M, Gottipati S, et al. CQArank: jointly model topics and expertise in community question answering[C]. *Conference on information and knowledge management*, 2013: 99-108.
- [36] Zhao Z, Yang Q, Cai D, et al. Expert finding for community-based question answering via ranking metric network learning [C]. *Proceedings of the 25th International Joint Conference on Artificial Intelligence*. New York, USA: AAAI Press, 2016: 3000–3006
- [37] Perozzi B, Al-Rfou R, Skiena S. Deepwalk: Online learning of social representations [C]. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA: ACM, 2014: 701-710.
- [38] Liu Z, Zhang Y. Structures or Texts? A Dynamic Gating Method for Expert Finding in CQA Services [C]. *Database Systems for Advanced Applications*. Cham: Springer International Publishing. 2018: 201-208
- [39] Socher R, Chen D, Manning C D, et al. Reasoning with neural tensor networks for knowledge base completion [C]. *Advances in Neural Information Processing Systems* 26. 2013: 926–934
- [40] 任鹏飞. 问答社区中问题响应时间预测方法的研究[D]. 上海大学, 2017.
- [41] Van Dongen S M. Graph clustering by flow simulation[J]. *ACM Transactions on Information Systems*, 2000.
- [42] Chen H, Perozzi B, Alrfou R, et al. A Tutorial on Network Embeddings[J]. *arXiv: Social and*

Information Networks, 2018.

- [43] 涂存超, 杨成, 刘知远, 等. 网络表示学习综述[J]. 中国科学: 信息科学, 2017(08): 32-48.
- [44] Tang J, Qu M, Wang M, et al. LINE: Large-scale Information Network Embedding[C]. International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, 2015: 1067-1077.
- [45] Grover A, Leskovec J. Node2vec: scalable feature learning for networks[C]. Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016: 855-864.
- [46] Kaelbling L P, Littman M L, Moore A W. Reinforcement Learning: An Introduction[J]. IEEE Transactions on Neural Networks, 2005, 16(1): 285-286.
- [47] Wei Z, Xu J, Lan Y, et al. Reinforcement Learning to Rank with Markov Decision Process[C]. Proceedings of the 40th International ACM SIGIR Conference. ACM, 2017: 945-948.
- [48] Xia L, Xu J, Lan Y, et al. Adapting Markov Decision Process for Search Result Diversification[C]. Proceedings of the 40th International ACM SIGIR Conference. ACM, 2017: 535-544.
- [49] Williams R J. Simple statistical gradient-following algorithms for connectionist reinforcement learning[J]. Machine Learning, 1992, 8(3-4): 229-256.

## 研究生期间发表论文

- [1] 黄辉, 刘永坚, 解庆. 一种基于“用户-标签”的专家发现方法[J]. 计算机工程  
(录用待发表).