武漢理工大學

（申请工学硕士学位论文）

基于知识图谱的个性化职位推荐算法研究

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：李传龙
指 导 教 师：刘洪星教授

2020 年 5 月

分类号_____ 密　　级_____

UDC _____ 学校代码 10497

武漢理工大學

学　　位　　论　　文

题　目 基于知识图谱的个性化职位推荐算法研究

英　文 Research on Personalized Job Recommendation

题　目 Algorithm Based on Knowledge Graph

研究生姓名 李传龙

指导教师 姓名 刘洪星 职称 教授 学位 博士

单位名称 计算机科学与技术学院 邮编 430063

申请学位级别 硕士 学科专业名称 计算机科学与技术

论文提交日期 2020.4 论文答辩日期 2020.5

学位授予单位 武汉理工大学 学位授予日期 2020.6

答辩委员会主席 _____ 评阅人 教育部学位与研究生教育发展中心盲审

2020 年 5 月

摘 要

随着信息技术的飞速发展，网络上的信息量激增。在求职领域中，传统的线下招聘也逐步被网络招聘所取代，出现了诸如 58 同城、智联招聘的大量求职网站系统，用户可以在网站上投递简历、进行面试，但是大量的职位信息可能让用户难以抉择。为了从海量的求职信息中发现用户的兴趣，满足用户个性化的信息需求，对求职推荐的准确性要求更加严格。

传统求职推荐系统多采用协同过滤算法，协同过滤算法存在冷启动和数据稀疏问题，解决这些问题的一个常见思路是在推荐算法中额外引入一些辅助信息作为输入。知识图谱包含了丰富的语义信息，可以为推荐系统提供潜在的辅助信息，近几年引起了越来越多的研究人员的关注。针对当前职位推荐系统的不足，本课题旨在结合知识图谱相关技术，构建职位知识图谱，并使用职位图谱中的语义信息提高职位推荐系统的性能，为在线求职者提供个性化、精准化的推荐。本文的主要研究内容如下：

1）分析求职领域信息，构建了求职领域知识图谱。在对各大网站职位信息进行详细分析后，构建了职位领域本体，然后从各大求职网站上采集职位数据，根据本体指导进行知识抽取。针对求职领域的知识图谱构建中不同数据源数据冗余问题，改进了一种基于属性嵌入的实体对齐算法，该算法将处理后的三元组数据分为结构数据和属性数据，基于 TransE 算法思想对两个训练模型进行训练，最终得到职位实体的向量表示，计算实体向量相似度得到对齐的实体对。相比于传统的基于字符串相似度实体对齐算法和基于知识表示的实体对齐算法，能够提高实体对齐的效率。最后通过 Neo4j 图数据库存储职位知识图谱，并进行可视化的展现。

2）分析用户对职位属性的喜好，构建了用户兴趣模型。通过分析用户对职位信息中薪资、工作地点、工作经验、学历、公司、行业六个属性的喜好程度，使用知识表示将职位的属性三元组向量化，计算出各个属性的权重，加权求和得到用户兴趣向量。在构建用户兴趣模型中加入了用户对属性的爱好，希望能够提高推荐的个性化。

3）提出了一种改进的基于协同过滤的推荐算法。通过知识图谱学习得到的语义信息和用户历史评价信息，计算得到职位相似度矩阵；同时使用之前构建

的用户兴趣向量，计算用户相似度矩阵，分别通过协同过滤方式计算预测评分，取评分高的前 N 个项目，得到推荐列表。

通过对比实验验证，结果表明，本文提出的基于知识图谱的个性化职位推荐算法相比传统协同过滤算法，推荐的准确率和召回率都有了一定程度的提升。

关键词：知识图谱；实体对齐；用户偏好；个性化推荐

Abstract

With the rapid development of information technology, the amount of information on the network has surged. In the field of job search, traditional offline recruitment is gradually being replaced by online recruitment. A large number of job search website systems such as 58 city and Zhilian recruitment have appeared, Users can post resumes and conduct interviews on the website, but a lot of job information may make it difficult for users to choose. In order to find the user's interest from the massive job search information and meet the user's personalized information needs, the results of job recommendation should be more accurate.

Traditional job search recommendation systems mostly use collaborative filtering algorithms. collaborative filtering algorithms have cold start and data sparse problems. A common idea to solve these problems is to introduce some auxiliary information as input in the recommendation algorithm. knowledge graph contains rich semantic information, which can provide potential auxiliary information for the recommendation system. In recent years, it has attracted more and more researchers' attention. Aiming at the shortcomings of the current job recommendation system, this thesis aims to combine technologies of knowledge graph to construct a job knowledge graph, and use the semantic data in the knowledge graph of job to improve the performance of the job recommendation system.The main contents of this thesis are as follows:

1) By analyzing the data in the job search field, a knowledge graph of job based on ontology is constructed. First, the ontology of the job field was build, Then job data was collected on major job search websites and knowledge was extracted according to the ontology guidance. Aiming at the problem of data redundancy of different data sources in the construction of knowledge graph in the job search field, an entity alignment algorithm based on attribute embedding was proposed. Compared with the traditional string alignment based entity alignment algorithm and the knowledge representation based entity alignment algorithm, efficiency of entity alignment was improved. Finally, through the Neo4j graph database, the job knowl--edge graph was displayed visually.

2) By analyzing the user's preference for each attribute in the job information, the user interest vector was constructed. Considering users' preference for the attributes of the position, the user's attribute preference vector was constructed based on the knowledge graph, attribute factors was added to the recommendation process to improve the personalization of the recommendation.

3) An improved collaborative filtering-based recommendation algorithm was proposed.Semantic information and user preference information was obtained from knowledge graph learning, then, Position similarity matrix and user preference similarity matrix was constructed, Finally the recommendation list was obtained.

The results of comparison experiments show that the personalized job recommendation method based on knowledge map proposed in this thesis has improved the accuracy and recall rate of recommendation to a certain extent compared with the traditional collaborative filtering recommendation method.

Keywords:knowledge graph; entity alignment; user preference;personalized recommendation

目 录

第 1 章 绪论

1.1 研究背景及意义

互联网技术飞速发展，网络上的知识也越来越多。传统的搜索引擎系统需要用户在搜索时输入关键字，然后在海量数据中进行检索，而结果往往因为用户搜索词不明确而不尽人意。为了满足用户个性化的需求，推荐系统(Recommender Systems)应运而生。推荐系统通过分析用户的搜索、浏览、评价等行为，发现用户的兴趣，然后将用户感兴趣的信息展示给用户。相比于搜索引擎，推荐系统可以解决信息过载的问题并进行个性化的推荐[1]。但是传统推荐系统存在冷启动和数据稀疏的问题，往往影响了推荐系统的准确性，如何解决这些问题是当前推荐领域研究的重点。

如今网络上存在大量的职位招聘信息，传统线下招聘由于覆盖率低、效率低、成本高等原因逐渐被线上招聘所取代，国内也出现了大量的求职招聘网站，如智联招聘、58 同城、拉勾网等。企业招聘人员通过招聘网站投发了大量招聘信息，面对海量的职位信息用户往往难以抉择，可能找不到适合自己的职位。而目前大部分的推荐系统采用协同过滤的推荐算法，存在冷启动和数据稀疏问题，比如对于新发布的职位，系统中没有用户的反馈信息，所以也无法将这个职位推荐出去，而用户没有历史行为记录，无法通过用户历史行为找到相似用户群体，也就很难进行推荐。

知识图谱（Knowledge Graph, KG）的出现为解决传统求职推荐系统的问题提供了新的思路，知识图谱中包含丰富的语义信息，能够一定程度上解决协同过滤算法中的数据稀疏问题，通过引入更多的语义关系，可以更深层次的发现用户的个人兴趣，具有重要的研究意义和实用价值，因此基于知识图谱的推荐研究逐渐成为推荐系统研究领域最为活跃的分支之一[1]。

知识图谱技术已经在旅游、医学等领域得到了广泛使用，但是现阶段大部分职位推荐还是使用传统协同过滤算法进行推荐，知识图谱在职位推荐领域应用较少，因此本文拟在求职领域构建职位知识图谱，并使用该知识图谱的语义信息作为辅助信息，进行职位推荐。这样就能通过知识图谱的语义信息，计算

出与用户喜好相近的职位，通过将知识图谱语义信息和传统协同过滤推荐相融合，使得职位推荐更个性化、更准确。

为了提高推荐的效果，就需要构建高质量的知识图谱。在构建过程中，需要进行实体对齐（Entity Alignment），实体对齐是实现多个数据源数据融合的重要步骤，一种好的实体对齐算法能够解决不同数据源中实体的冗余问题，提高知识图谱的构建质量，从而支持语义搜索、问答系统、个性推荐等上层应用。知识图谱多使用基于字符串相似度的实体算法进行实体对齐，这种算法需要研究者针对不同的领域设计相似度比较函数，往往需要研究者对相关领域有一定的了解，比较耗费人力。同时这种算法忽略了属性因素的影响，也没有考虑实体之间的关系，对齐效果不佳。

本文将对职位领域知识图谱的构建和基于知识图谱的职位推荐方法进行深入研究。拟构建职位领域知识图谱，并针对职位知识图谱，改进一种实体对齐算法，提高实体对齐效果。在推荐系统中引入知识图谱，希望相比传统推荐算法，能得到更好的推荐效果，并解决冷启动和数据稀疏问题。

1.2 研究现状

本文拟研究构建一个职位领域知识图谱，然后在此基础上进行个性化推荐。因此，下面从知识图谱、推荐技术、基于知识图谱的推荐方法三个方面分析与本文相关的研究现状和存在的问题。

1.2.1 推荐技术研究现状

推荐技术作为一种信息过滤技术，不需要像搜索引擎那样需要用户主动进行筛选，推荐主要通过用户的历史行为，挖掘出用户个人兴趣偏好，然后向用户提供满足需求的个性化、精准化信息。协同过滤推荐技术是发展最快、应用最广泛的个性化推荐技术，它利用了“群体智慧”的思想，使用用户对物品的历史评分计算用户或物品的相似度，将相似度高的物品推荐给相关用户。但是如果用户没有进行过评分或者物品没有被用户评价，就很难得到准确的推荐结果。为解决这个问题，在职位推荐领域，通常引用社会关系、用户或物品属性和上下文信息来作为补充信息辅助推荐系统进行精准推荐。

针对协同过滤中数据稀缺带来的推荐效果不佳的问题，已经有不少学者提出了针对求职领域的推荐算法。国内方面，针对基于传统的职位搜索与浏览的推荐现状在实际实施过程中面临的问题，路小瑞[43]通过对求职者的 Web 行为日志进行分析，融合基于内容的过滤、基于人口统计过滤与协同过滤推荐技术，实现了一种基于行为分析的个性化职位推荐系统。侯丽娟[26]对用户隐式信息进行分析，使用了行为分析的思想，将求职者的行为划分为注册、申请和浏览三种行为，运用了“行为类似的求职者在求职时有相似的求职偏好”的思想，在推荐中加入兴趣相似，提出了求职者行为-兴趣模型。国外方面，文献[44]提出了一个可扩展的基于项目的在线工作推荐系统，通过利用由表示各种行为和上下文相似性信号的多边连有向图来克服稀疏性和可伸缩性的问题。文献[45]实现了一种基于动态用户配置文件的工作推荐系统。根据求职者的行为更新用户配置文件，使用用户配置文件计算用户偏好，完成工作的推荐。

在职位推荐领域多采用协同过滤算法，但存在稀疏性问题和冷启动问题。在职位推荐系统中，往往包含数以万计的用户和职位，而用户只会对部分职位进行评价，导致用户-项目评分矩阵稀疏，影响了相似用户集的计算；如果一个新用户从未对系统中的职位进行评价，或者一个新职位从未被评价，那么这个用户或职位就永远得不到推荐。解决稀疏性和冷启动问题的一个常见思路是在推荐算法中额外引入一些辅助信息（side information）作为输入，如社交网络信息、用户或者物品的属性、上下文信息等。辅助信息可以丰富系统中用户和物品的信息描述，很好地补充系统中交互信息的稀疏，使得推荐效果更加个性化。而知识图谱本身拥有丰富的语义信息，可以作为辅助信息引入推荐系统中，解决稀疏性和冷启动问题，提高推荐的准确性。

1.2.2 知识图谱研究现状

2012 年 5 月 17 日，Google 提出了知识图谱，最初是为了提高搜索引擎的能力，增强搜索质量,优化用户体验。目前，随着网络信息技术的飞速发展，知识图谱在搜索、个性化推荐、智能问答等领域都取得了一定的研究成果[2]。

知识图谱是一种结构化的语义网络，通过符号形式描述现实中的概念及其关系。知识图谱在逻辑上可分为本体层和模式层，模式层在数据层之上，在实际运用中，通常使用本体库来管理模式层。在数据层中，知识通过三元组的形式存储事实。

按覆盖范围可把知识图谱可以分为通用知识图谱和领域知识图谱。通用知识图谱存储常识知识，多用于搜索等通用场景，国外不少学者已经进行了研究，并构建了诸如 DBpedia[20]、Yago[21]、NELL[22]等的高质量通用知识图谱。国内方面，近些年也出现了搜狗的知立方[68]、百度的知心[69]等用于搜索引擎的知识图谱。

另一方面，领域知识图谱主要针对单一领域或行业，构建的成本更小，也取得不少研究成果。例如，文献[24]针对煤矿领域，使用自顶向下的构建流程，构建了煤矿领域的本体信息，并在本体的指导下构建了煤矿知识图谱。而文献[25]提出一种高效的知识图谱构建方法，并通过提出的方法构建了高质量的学科知识图谱;文献[77]针对求职领域，利用知识图谱为求职者进行职位推荐。总体来看，国内领域知识图谱已经逐渐引起了国内学者的注意，在旅游、文物管理、音乐推荐等多个领域都有研究应用，但是在国内职位推荐领域，还没有学者构建高质量的职位知识图谱。

国内的知识图谱研究虽然已有一定成果，但是还处于发展阶段，很多领域知识图谱都还有待构建，除此以外，知识图谱的构建方法也有待进一步优化，实体对齐也可称为实体匹配，作为知识图谱构建的主要步骤，是当前知识图谱研究的热点。实体对齐目的是在异构数据源寻找属于现实世界中的同一实体，对重复、冗余的实体进行清理合并，解决知识图谱中实体复用问题，进而从顶层创建一个大规模的统一的知识图谱，并在知识图谱的语义主持下优化搜索、推荐等应用。

实体对齐算法可分为基于字符相似性的方法和基于知识表示的方法。由于计算简单，起初构建知识图谱时，实体对齐多使用字符串相似性。文献[14]基于字符串相似性匹配原则，根据待对齐实体属性特征的字面量判定实体匹配与否。基于字符串相似性的实体对齐虽然在一定程度上能够进行冗余实体的清除，但是在不同数据源中，实体的名称（字符串序列）可能存在很大的差异，这时就不能通过字符串进行相似度比较，而需要考虑实体潜在的语义信息。

随着 TransE[71]、TransR[55]等知识表示的嵌入模型的提出，嵌入模型的进步为实体对齐的研究带来了新的契机。知识表示学习（Knowledge Graph Embedding）是将知识图谱中的实体和关系映射到低维空间，学习得到实体和关系的向量表示。通过知识表示学习得到的向量表示蕴含了知识图谱内在的结构信息及实体和关系的属性特征。目前，应该有不少学者在基于嵌入的实体对齐

方向进行了研究，文献[15]将待对齐的两个知识图谱分别转化为向量表示形式(称为知识表示);然后，基于得到的知识表示，根据先验对齐数据学习知识图谱间实体对的映射关系。文献[50]提出了一种基于知识表示和机器学习的实体对齐方法，利用实体属性中包含的语义信息，迭代训练找到语义对齐的实体对。这类算法仅仅使用了知识表示的语义信息，忽视了结构信息，同时还需要大量的标记好的实体对齐数据。

通过分析知识图谱构建流程中的实体对齐研究现状，发现目前构建知识图谱一般还是采用传统的基于字符串对齐算法，这种对齐算法对齐效果一般，在不同的领域还需要构建特定的对齐公式，步骤繁琐并且需要构建者有一定的相关领域知识。而基于知识表示的对齐算法大部分没有考虑属性的因素，这些问题都是本文待解决的问题。

1.2.3 基于知识图谱的推荐方法研究现状

已经有不少学者将知识图谱运用于电影、新闻、景点、音乐等推荐领域。在国外，文献[39]最早在音乐推荐领域使用了知识图谱，利用知识图谱丰富语义信息来加强音乐的推荐。之后，文献[40]将音乐推荐分为两个部分。通过在Songfacts3 和 Last.fm4 采集用户反馈信息以及歌曲数据集来进行音乐家的推荐；通过基于实体的近邻项目匹配、基于路径的近邻项目匹配进行音乐推荐;文献[38]中利用基于知识图谱的推荐方法为求职者根据自身情况生成职位推荐列表。近年来，国内知识图谱也得到了广泛应用，针对电影领域，文献[41]使用自顶向下的构建方法构建了电影领域知识图谱，首先分析了电影领域相关知识特征,构建了电影领域本体库，在本体库的指导下构建了电影领域知识图谱。最后知识图谱的向量化表示方法和协同过滤算法结合，利用从知识图谱中提取的物品语义信息,解决了协同过滤算法没有考虑语义信息的问题,完成了电影的个性化推荐。文献[42]通过对旅游知识图谱中不同标签的属性子图独立建模，利用深度学习模型挖掘游客及景点等图节点语义特征，利用知识表示学习得到游客和景点特征向量，最后通过计算游客和景点向量相关性进行景点推荐。

目前，基于知识图谱的推荐方法主要包括基于表示学习（representation learning）的嵌入方法和基于元路径（meta path）的方法。基于表示学习的嵌入方法使用知识表示学习对实体、关系进行向量化处理，从而得到实体和关系的低维稠密向量表示。使用得到的向量能够很好与传统的协同过滤进行融合计算，

得到推荐列表。例如文献[66]使用知识表示学习得到了实体、关系的向量表示，然后将学习得到的实体嵌入合并到推荐框架中。深度知识感知网络[30]将实体嵌入和字嵌入视为不同的渠道，然后设计 CNN 框架将它们组合在一起以进行新闻推荐。文献[67]提出了一种结合协同过滤模型的知识库嵌入推荐方法 CKE，通过将知识图谱中的实体向量、词向量和图向量同时嵌入到贝叶斯框架中，作为项目的补充信息，并联合协同过滤进行推荐。基于嵌入的方法在利用 KG 辅助推荐系统方面表现出很高的灵活性，但这些方法中采用的知识图谱嵌入(Knowledge Graph Embedding)算法通常更适用于链路预测等图形内应用而不是推荐。

基于路径的方法通过探索知识图谱中各项目之间的联系模式，为推荐提供额外的指导。例如，个性化实体推荐[64]和基于元图[65]的推荐将 KG 视为异构信息网络（heterogeneous information netword，HIN），异构信息网络包含多样的节点类型和边的类型，社交网络就是一个典型的例子，通过构建用户或物品间的元路径（meta-path）来进行推荐。基于路径的方法充分的利用了知识图谱的网络结构，通过挖掘节点之间的路径找到实体间的潜在关系，但元路径需要进行手动设计，在实践中难以达到最优效果。另一方面，在某些领域，例如新闻推荐，它们的实体和关系不在一个域内，是不可能人工设计元路径的[16]。

综合来看，基于知识图谱的推荐算法一定程度上解决了协同过滤算法的问题，提高了推荐效果，并且在旅游、医学等领域得到了应用，但是在职位推荐领域还未见相关研究。本文拟在求职领域引入知识图谱，并考虑到基于表示学习的嵌入方法多考虑的是引入实体向量作为补充信息，而求职领域中用户往往关注职位中的属性，因此将在推荐过程中考虑属性因素，来提高推荐效果。

1.3 研究内容

通过对知识图谱和推荐技术的研究现状分析，现在的推荐系统多采用协同过滤的推荐算法，在职位推荐上，目前通常引用社会关系、用户或物品属性和上下文信息作为补充信息，解决协同过滤的弊端。知识图谱作为一种语义网络，蕴含丰富的语义信息，在推荐系统中引入知识图谱作为辅助信息，能够很好的解决数据稀疏问题。因此本文拟采用在职位推荐中引入知识图谱来提高推荐准确性，同时为了提高推荐效果，在知识图谱构建中优化实体对齐过程，构建高质量的职位知识图谱，利用职位知识图谱中的语义信息和属性信息，挖掘用户

的深层喜好，通过协同过滤和知识图谱相融合，目的是缓解数据稀疏问题，提高职位推荐系统的性能。

本文研究内容包括如下三个方面:

（1）职位知识图谱构建

首先分析领域知识图谱的构建流程，构建职位领域的本体。根据各大求职网站中职位数据，采集多个数据源的职位信息，并根据本体从数据源中抽取实体和关系，进行异构数据源的知识融合。最后，将抽取的职位知识保存在 Neo4j 图数据库中。现有的基于知识表示学习的对齐方法依赖大量标注数据，且并未利用知识图谱中属性等结构化信息，限制了实体对齐的效果。本文拟使用一种基于属性嵌入的实体对齐算法，将采集的三元组数据分为结构数据和属性数据两部分，其中结构数据表示实体与实体的关系三元组，属性数据表示实体与属性的三元组。构造基于结构嵌入表示的实体对齐模型和基于属性嵌入表示的实体对齐模型，训练两个视角的实体对齐模型，通过训练得到属性嵌入的向量约束结构嵌入训练得到的向量，最终训练得到职位实体的向量表示，完成实体对齐，提高职位知识图谱的构建质量。

（2）基于知识图谱的用户兴趣挖掘

用户兴趣模型的构建是进行职位推荐的重要步骤，因此本文将基于知识图谱对职位属性信息进行分析，构建用户兴趣模型。针对求职领域，考虑到用户求职时往往对职位的属性有一定偏好，分析了职位本体中各个属性对用户求职选择的影响，通过计算各个属性在用户兴趣中的权重，建立基于知识图谱的用户兴趣向量。希望通过在推荐过程中加入属性因素，提高推荐的个性化和准确性。

（3）基于知识图谱的推荐算法

传统的协同过滤推荐算法存在数据稀缺、冷启动等问题，进行推荐时没有考虑物品的语义信息。本文拟提出一种基于知识图谱的推荐算法，采集用户面试评价信息，通过知识图谱学习职位的语义信息，使用职位的语义信息计算职位相似度，使用用户兴趣模型计算用户相似度，分别计算在用户相似度和职位相似度下的预测评分，融合计算结果并完成推荐。

1.4 组织结构

本文的内容组织结构如下：

第一章：绪论。分析了求职推荐领域面临的问题，分析了知识图谱在推荐领域的应用，考虑在职位推荐领域构建知识图谱，提高推荐效率。

第二章：职位领域知识图谱的构建。采用自顶向下的构建方式，首先构建了职位领域本体，在本体的指导下完成了知识抽取。进行实体对齐，最终构建了职位知识图谱，并通过图数据库进行可视化展现，完整的构建了求职领域的知识图谱，并提出了一种基于知识表示的对齐算法，相比与传统的基于字符串的对齐算法提高了对齐效果。

第三章：用户兴趣建模。使用第二章得到的职位三元组向量，通过对职位的各个属性分析，计算属性权重，并进行加权和处理，得到用户-属性向量。

第四章：基于知识图谱的职位推荐。针对求职领域，提出了一种基于知识图谱和协同过滤的推荐算法，使用第三章得到用户-属性向量计算得到用户偏好相似度矩阵，使用用户偏好相似度矩阵和职位相似度矩阵计算得到推荐结果。

第五章：实验验证。介绍实验环境和实验数据，给出相应的评价指标和分析对比算法。整理实验结果数据，对实验结果进行了对比分析。最后构建了一个推荐系统原型。

第六章：总结。总结了全文的工作，提出了不足之处。

第 2 章 求职领域知识图谱构建

知识图谱是实体的属性关系网，能够很好的表达实体之间的关系。通过知识图谱，推荐系统可以很好的进行关联内容的推荐，知识图谱的语义信息越丰富，推荐效果越好。本章主要介绍职位知识图谱的构建过程，采用领域知识图谱的通用构建方法，自顶向下地构建职位知识图谱，然后对多个数据源的知识进行融合，最终建成的知识图谱保存在图数据库中。

2.1 知识图谱构建流程

知识图谱是一种揭示实体间联系的语义网络，通过符号形式描述现实中的概念或实体及其关系。实体（Entity）指的是具有现实世界中的某一具体概念，例如具体的某一个人、某一个城市。概念（Concept）指的是具有同一种性质的实体集合，如城市、书籍。

按覆盖范围可把知识图谱分为通用知识图谱和领域知识图谱。通用知识图谱通常以互联网开放数据为基础，逐步扩大规模，强调知识覆盖的广度，对知识抽取的质量有一定容忍度，以知识融合提升数据质量，多用于智能搜索等领域，例如搜狗的知立方[68]、百度的知心[69]；领域知识图谱以领域或企业内部的数据为主要来源，具有特定的行业意义，对知识抽取的质量要求很高，从领域或企业内部的结构化、非结构化以及半结构化数据进行联合抽取，需要依靠人工进行审核校验，来保证质量，除了用于搜索问答，还用于决策分析、业务管理等，对推理的要求更高，要求有较强的可解释性要求。

在逻辑上，知识图谱可分为模式层和数据层。在数据层，事实通常使用（实体，关系，实体）或（实体，属性，属性值）三元组的形式来存储，其中，实体是知识图谱的最基本的元素，指具体的人、时间、物品等。模式层在数据层之上，模式层一般由本体组成，它是整个知识图谱的架构，定义了该领域内的所有概念。知识图谱的体系架构指的是构建知识图谱的整个技术流程，如图 2-1 所示，其中虚线框内的部分为知识图谱的构建过程[3]。

知识图谱的构建方式分为自顶向下与自下向上两种方式。自顶向下的方式需要先定义本体或者模式，利用模式或者本体来约束输入的数据，完成知识抽

取，并最终构建知识图谱。这种构建方式适用于领域知识图谱的构建，因为特定领域的知识更容易定义模式，如 Freebase 项目就是采用这种方式。自下向上的构建方式则是先从开发的数据源中采集数据，从采集的数据中抽取知识，完成知识图谱的构建。这种构建方式更适用于搜索之类的通用知识图谱构建，Google 的 Knowledge Vault 和微软的 Satori 知识库采用的就是这种方法。

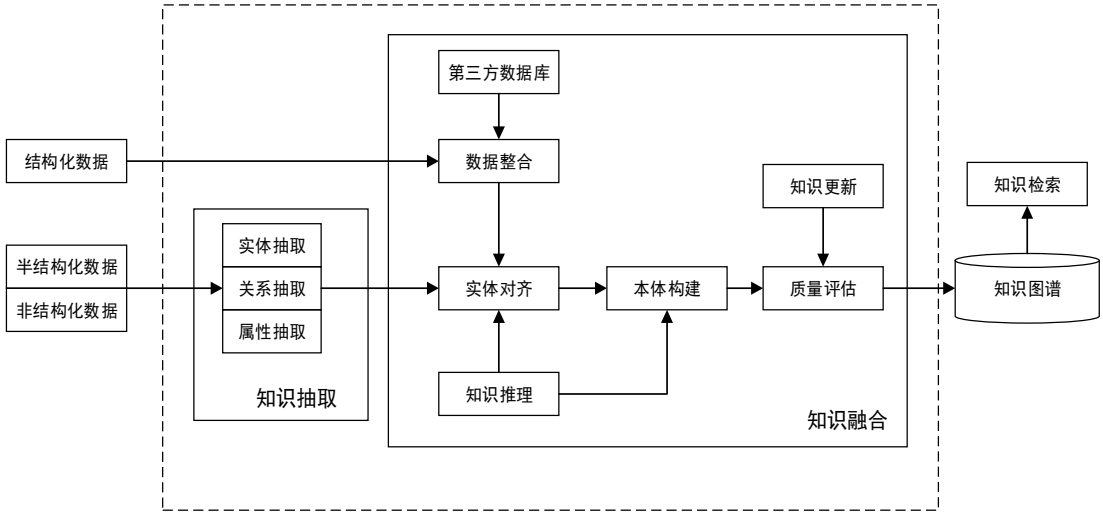


图 2-1 知识图谱的体系架构

本文要构建的职位知识图谱是一种领域知识图谱，对知识图谱的质量有一定要求。自顶向下的构建方法较好的体现了概念间层次，并且领域知识图谱中的术语和概念更加专业化，在领域专业网站和百科网站中具有结构化的数据源，适合于提取领域本体和模式，因此本文采用自顶向下的构建方式构建职位知识图谱。具体的构建过程如图 2-2 所示。

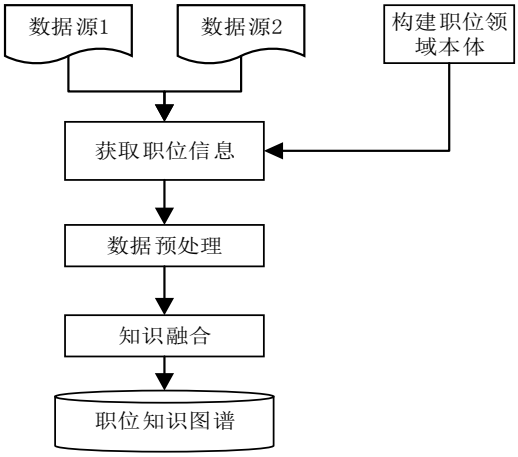


图 2-2 职位知识图谱构建过程

首先，构建职位知识图谱的模式层，定义求职领域的本体；其次，从各大求职网站上爬取职位信息数据源，根据职位本体从数据源中抽取职位知识；然后，将从不同数据源抽取的实体进行实体对齐（Entity Alignment），实现知识融合，并以三元组形式存储；最后，将知识保存在 Neo4j 图数据库中，以可视化的形式展示出来。

2.2 求职领域本体构建

2.2.1 本体的定义

本体（Ontology）最初源于哲学，用来描述世界中客观存在的事物的本质。文献[36]定义了本体为“本体是共享概念模型的形式化规范说明”[36]，其中共享（Share）指本体中体现的知识是共同认可的，概念化（Conceptualization）指将客观世界中事物进行抽象建模，明确化（Explicit）指本体中包含的概念以及它们之间的联系都被精确的定义，形式化（Formal）指本体能够被成机器理解并处理。

一个本体是由类、关系、函数、公理和实例五种元素组成。类表示领域内实体或者概念，例如电影、城市等。关系用来描述类之间的联系，如铅笔属于文具，属于就是一种关系。函数则是一种特殊的关系，例如定义 Classmateof(x,y) 表示 x 和 y 是同学。公理表示约束本体类、关系的某种事实依据，例如年龄必须大于 0。实例则表示具体某个类的实际存在，如北京是一个城市是一个实例。

构建本体是为了描述或表示一个领域的知识，通过本体的类、关系、函数等规范化的元素对知识进行表示，将抽象化的领域知识进行符号化，使得计算机能够对领域的概念进行理解，最终得到人机交流的目的。

构建领域本体常用的方法分为半自动化构建、自动化构建和人工构建三类。人工构建方法通常具有很高的质量，但会耗费大量的人力资源。自动化构建本体技术是采用各种知识获取技术和机器学习技术，自动化地从数据源处抽取概念和关系，从而形成本体。本文构建的知识图谱需要应用到推荐系统中，需要更高的可信度，因此本文使用人工构建方法。

2.2.2 求职领域本体构建流程

传统的本体构建方法分为骨架法、企业建模法和七步法[72]等，其中七步法

由斯坦福大学医学院开发，七步法的构建流程清晰简单，是领域本体构建的常用方法，本文也采用七步法来构建求职领域本体，构建过程如图 2-3 所示。构建步骤分为：确定本体的专业领域和范畴;考虑复用已有的本体；列出本体中的重要术语；定义类和类的结构体系；定义类的属性；定义属性的分面(值域)；创建实例。

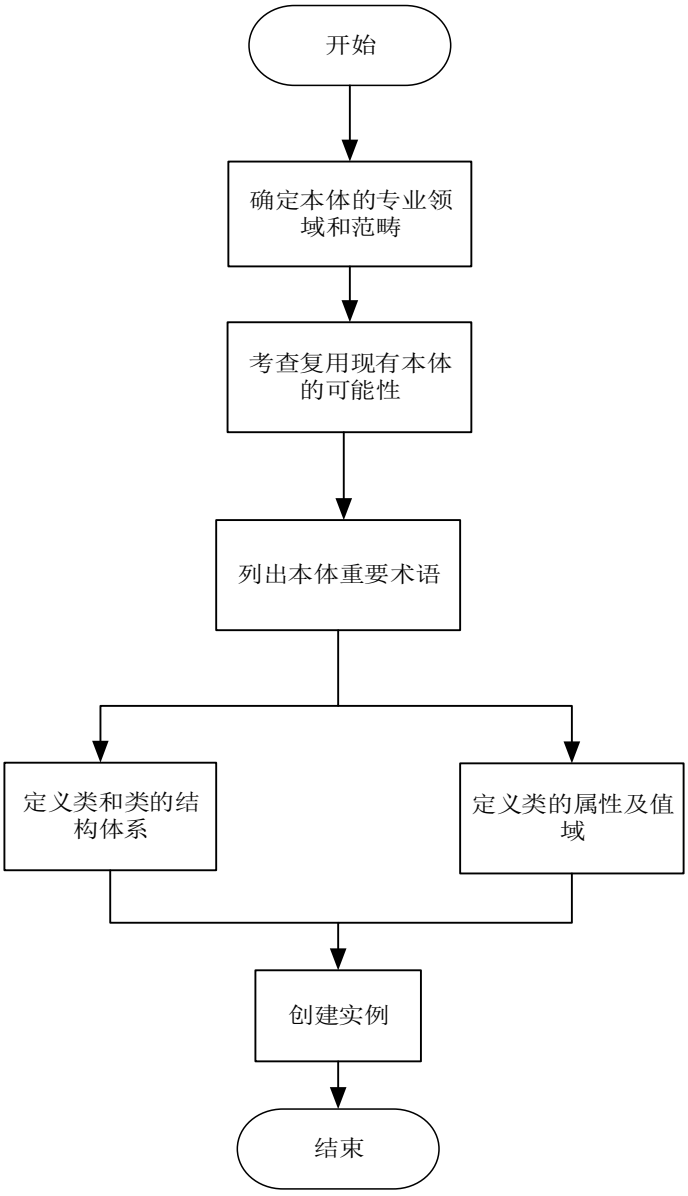


图 2-3 本体构建流程图

(1)确定本体的专业领域和范畴。本文构建求职领域本体的目的是帮助完成职位信息的知识抽取，构建职位知识图谱。

(2)考查复用现有本体的可能性。文献[73]定义了网络招聘领域本体，文献中

针对大学生求职，分析了学生信息和职位信息，抽取了教育背景、职业、行业、公司类别、职业五个概念，并对五个概念进行了属性分析。本文也是构建求职领域的本体，不同的是用户人群不仅仅是大学生，因此本文参考了该文献本体构建流程，进行求职领域本体构建。

(3)列出本体中的重要术语。为了分析求职领域包含的概念，分析了拉钩网、智联招聘、51job 上的职位信息，几个网站的主要职位属性信息如表 2-1 所示。通过对三个网站上的职位信息的分析，职位的主要属性有:薪资、工作地点、要求工作经验、要求学历、发布时间、职位描述、所属公司、公司类型、公司规模、所属行业。从中抽取“职位”，“行业类型”，“公司”，“公司类型”，“工作地点”五个概念。

表 2-1 求职招聘网站的职位属性

网站名	职位属性
智联招聘	薪资、工作地点、要求工作经验、要求学历、发布时间、职位描述、所属公司、公司类型、公司规模、职位亮点、所属行业、公司简介
51job	薪资、工作地点、要求工作经验、要求学历、发布时间、所属公司、公司类型、公司规模、所属行业、职位描述、任职要求
拉勾网	薪资、工作地点、要求工作经验、要求学历、发布时间、职位关键字、所属公司、公司类型、公司规模、所属行业、任职要求、公司网址、福利待遇

(4)定义类和类的结构体系。本文使用自顶向下法定义类及阶层，将最顶层概念作为起点，逐步抽象出子类并细化和拓展。在上一步本文抽取出了“职位”，“行业类型”，“公司”，“公司类型”，“工作地点”五个概念，将每一个概率定义为类。其中“公司类型”按求职网站上的划分，可分为民营、国企、外资、合资、上市公司、政府机关和创业公司。“工作地点”按公司发布职位时填写的具细分为省份或者具体城市。“行业类型”主要有“互联网/IT/电子/通信”、“金融业”、“建筑业”、“批发/零售/贸易”、“制造业”、“生活服务行业”、“科研教育行业”、“文化/体育/娱乐”、“交通运输/仓储/物流”等。

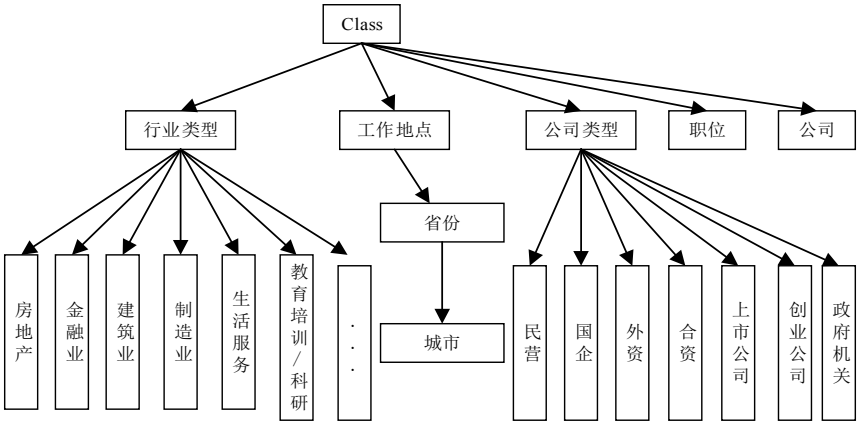


图 2-4 求职领域本体类别结构图

(5)定义类的属性及值域。属性可分为对象属性和数据属性。对象属性连接各个实体类，数据属性连接数据值和类。本文职位本体的属性如表 2-2 所示：

表 2.2 求职领域本体属性

实体类	属性名称	属性类型	属性描述	值域
职位	所属公司	对象属性	发布职位的公司信息	公司
	薪资	数据属性	职位的薪酬	String
	发布时间	数据属性	发布时间	String
	工作经验	数据属性	要求的工作年限	String
	职位描述	数据属性	职位要求，工作内容	String
	所属行业	对象属性	职位所属的行业	行业类型
	所属城市	对象属性	职位所在城市	工作地点
	学历	数据属性	职位要求的学历	String
工作地点	地点名称	数据属性	地名	String
行业类型	行业名称	数据属性	行业名称	String
公司	公司规模	数据属性	人数	String
	公司类型	对象属性	公司的类型	公司类型
公司类型	公司类型名称	数据属性	企业分类	String

2.2.3 使用 Protégé 构建 Schema

本体构建工具能够帮助完成本体形式化，方便本体的构建。常用的本体构建工具有 OntoEdit、Ontolingua、WebODE、WebOnto、OntoSaurus、Protégé 等。Protégé 软件是斯坦福大学开发的一个开源的本体编辑器，它基于 Java 语言编写，拥有可视化的用户界面 GUI（Graphical User Interface），用户通过点击相应项编辑增加类、属性、实例等元素。本文使用 Protégé 来构建求职领域的本体。具体流程如下：

1) 定义实体类别

通过对求职领域的本体分析，本文构建的类有行业类型、公司、工作地点、公司类型、职位。

2) 定义语义关系

通过对职位信息的分析，需要定义的语义关系有：职位---所属公司---公司；职位---所属行业---行业；职位---位于---工作地点；公司---属于---公司类型。

3) 定义实体属性

通过本体构建时对概念的属性分析，职位实体包括工作地点、薪资、发布时间、要求工作经验、职位描述等属性信息。公司实体包括公司名称、人数规模、公司类型。公司类型包括公司类型名称。“行业类型”主要有行业名称。最终得到的求职领域的本体结构图如图 2-5 所示：

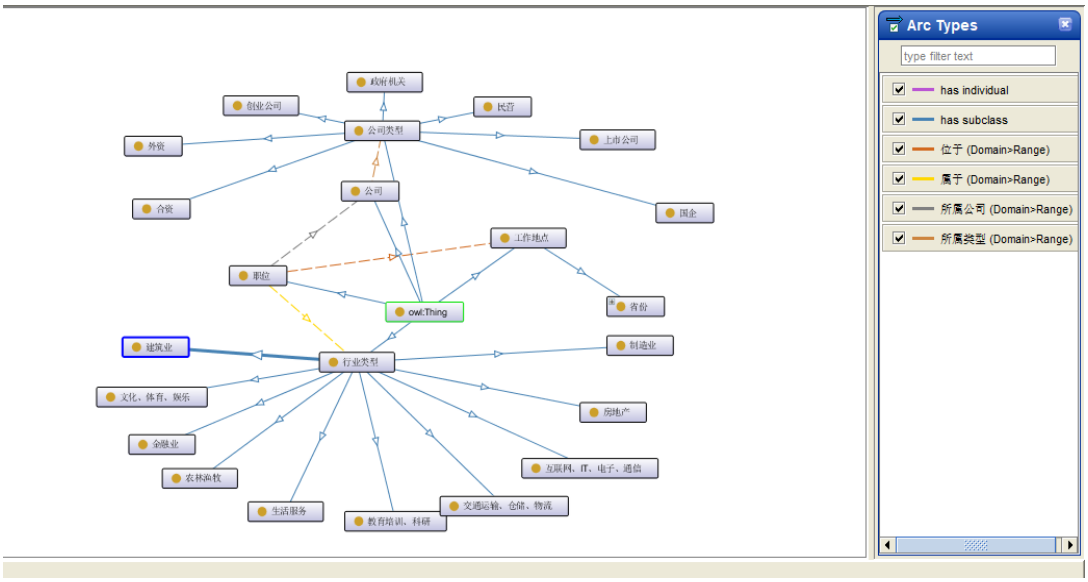


图 2-5.职位领域本体结构图实例

2.3 求职领域实体数据获取

2.3.1 数据抓取

本文的职位信息主要从拉钩网、51job、智联招聘等各大求职网站采集，这些网站都是以 html 形式展现，所以本文使用 python 的 BeautifulSoup 包对网页进行解析。具体的流程如图 2-6 所示。

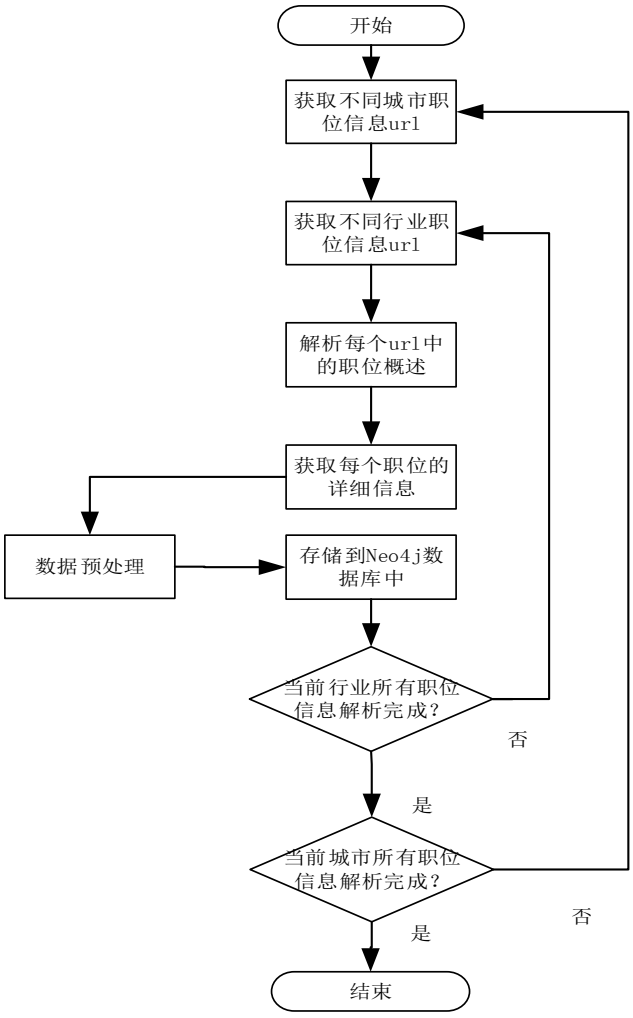


图 2-6 求职网站数据爬取流程图

因为求职网站中同一类页面的布局基本相似，本文从求职网站中通过城市、行业的筛选条件对职位界面信息进行解析，采用基于标签遍历的方法，获取其在 DOM 树中对应的节点，从而提取所需信息。最后将从各个城市各个行业中

采集得到的职位信息网页中爬取的数据。MongoDB 是一种非关系型数据库（Non-relational Databases），将数据以键值对的形式保存，存取操作非常方便。因此本文将采集到的数据保存在 MongoDB 数据库中。

2.3.2 数据处理

将爬取的数据存储到 MongoDB 数据库中，存储的数据如图 2-7 所示：

	_id ObjectId	company_financing_stage String	company_industry String	company_location String	company_name String
1	5e53a6df04cd3471f8279039	"已上市"	"计算机软件"	"上海"	"中软国际"
2	5e53a6df04cd3471f827903a	"不需要融资"	"互联网"	"北京"	"去哪儿"
3	5e53a6df04cd3471f827903b	"已上市"	"计算机软件"	"上海"	"中软国际"
4	5e53a6df04cd3471f827903c	"不需要融资"	"互联网"	"北京"	"去哪儿"
5	5e53a6df04cd3471f827903d	"已上市"	"计算机软件"	"上海"	"中软国际"
6	5e53a6df04cd3471f827903e	"不需要融资"	"互联网"	"北京"	"去哪儿"
7	5e53a6df04cd3471f827903f	"已上市"	"计算机软件"	"上海"	"中软国际"
8	5e53a6df04cd3471f8279040	"不需要融资"	"互联网"	"北京"	"去哪儿"
9	5e53a6df04cd3471f8279041	"已上市"	"计算机软件"	"上海"	"中软国际"
10	5e53a6df04cd3471f8279042	"不需要融资"	"互联网"	"北京"	"去哪儿"
11	5e53a6df04cd3471f8279043	"已上市"	"计算机软件"	"上海"	"中软国际"
12	5e53a6df04cd3471f8279044	"不需要融资"	"互联网"	"北京"	"去哪儿"
13	5e53a6df04cd3471f8279045	"已上市"	"计算机软件"	"上海"	"中软国际"
14	5e53a6df04cd3471f8279046	"不需要融资"	"互联网"	"北京"	"去哪儿"

图 2-7 爬虫数据示例

然后对存储的数据进行处理，处理步骤如下：

1) 校正发布日期：如 11-28.11:28 等统一使用 yyyy/mm/dd 表示。
2) 校正薪水以数字保存，一般的求职网站上薪资都是区间值，因此在数据库中使用最低薪资-最高薪资表示。
3) 根据工作经验年限划分招聘等级：这里工作年限一般有不限、1-3 年、3-5 年、5 年以上等，分别表示 0，3，5，10。
4) 学历要求：分为不限、本科、硕士、中专、大专等。
5) 公司类型分为民营、国企、外资、合资、上市公司等。
6) 公司人数范围分为 100 以下、1000 人以下、1000-9999 人以及 10000 以上。
7) 工作地点分为城市和具体工作地点两个字段保存，具体工作地点可为空值。
8) 缺省值分析：对于大量属性缺失的数据进行丢弃，对于部分属性缺失的数据进行填充，如没有填写工作年限则默认为 0。
9) 异常值分析和失效值排除。对于失效的数据进行丢弃处理，部分数据异

常的数据进行修正，无法识别修正则进行丢弃。

10) 清除掉数据中空格、无意字符、标点符号等与文本主要内容无关的字符。

最后使用 python 编写程序，将处理后的属性信息根据职位领域本体构建三元组信息，读取一条职位数据，将职位的属性信息依次读取，以(实体，属性，属性值)构建三元组信息，并保存到文本中；同时读取职位与其他实体的信息，以（实体，关系，实体）构建三元组信息并保存到另一文本中。实体三元组与实体属性三元组的划分按之前构建的职位领域本体进行。

2.4 求职领域的实体对齐算法研究

2.4.1 传统的实体对齐算法

由于职位数据是从多个数据源抽取得到，所以会产生知识重复的问题。知识融合就是将不同数据源中获取的知识进行融合，包括“实体对齐（Entity alignment）”和“实体消歧（Entity disambiguation）”两个方面。“实体对齐”也称为实体匹配或实体解析，是判断两个实体是否指向真实世界同一实体的过程。在不同的数据源中，同一个事物可能有不同的名称，例如同一个职位可以被发布在多个网站，在知识抽取后就会出现重复的问题，因此需要将它们进行对齐。而“实体消歧”解决的是“一词多义”的问题。同一个名称可能指代多个事物，比如“苹果”，可以表示一种水果，也可以指手机。同一个名称在不同的上下文中所表达的含义不太一样。一词多义的现象多存在于通用知识图谱中，在领域知识图谱中很少出现，因此本文没有进行实体消歧。

好的实体对齐算法能够解决知识库的复用问题，消除冗余重复的实体，提高知识图谱的质量，对齐算法设计的主要思路是利用多种实体匹配技术结合数据的特点和处理方法对知识库中指向相同对象的实例进行辨析以获得对齐结果[11]。传统的实体对齐多采用基于属性相似性的成对比较的方法，属性相似性的计算多采用距离度量算法，常见的距离度量算法有编辑距离、余弦相似度以及欧式距离等，这种方法没有考虑匹配实体间的关系，而是通过对不同属性分配权重，计算属性的相似度加权和，最后通过设置的相似度阈值判断两个实体是否相等，基于字符串相似性的实体对齐大多都局限在实体的字符串描述上，无法对图谱结构进行量化，而且难以在多领域间迁移。

随着 TransE、TransR 等知识表示(knowledge representation)的嵌入模型的提出，也有不少学者使用了基于知识嵌入的实体对齐算法。知识表示是将知识图谱中的实体和关系映射到低维空间，学习得到实体和关系的向量表示，这种低维稠密的向量表示不仅蕴含了知识图谱内在的结构信息及实体和关系的属性特征，还具有丰富的语义信息。但是大部分研究仅仅使用了知识表示的属性信息，忽视了结构信息，同时还需要大量的标记好的实体对齐数据。对于知识表示嵌入进行实体对齐的问题，文献[45]提出了融合语义和结构信息的知识图谱实体对齐，使用协同训练框架，将抽取到的信息划分为语义信息和结构信息，并在两个视角下分别训练，不断选出最可信的实体对齐结果用于辅助另一视角的训练，实现了语义和结构信息的融合，从而提升实体对齐的效果。

本文针对求职领域，在文献[16]提出的基于属性嵌入的实体对齐算法基础上进行了优化，融合实体的结构信息和属性信息，对求职领域中抽取的实体信息进行对齐，消除冗余的实体，构建职位知识图谱，相比传统实体对齐算法，能够得到更好的实体对齐效果。

2.4.2 改进的实体对齐算法

2.4.2.1 知识表示算法

知识表示就是将知识用计算机可接受的符号描述出来，即知识的符号化过程。传统的三元组知识表示形式在计算效率、数据稀疏上面临诸多问题[3]，近年来随着深度学习的发展，知识表示学习也得到了重要的进展，通过机器学习训练，从而将实体的语义信息转化为稠密的低维向量，从而把三元组计算替换为计算实体、关系之间的向量关系，提高了计算效率和结果。

Word2Vec（Word to Vector）是由 Google 的 Mikolov 等人提出的一个词向量计算模型[55]，它的特点是将所有的词表示为低维向量，使用向量来计算词与词之间的关系，解决了离散数据处理的问题，同时起到了扩充特征的作用，但是 Word2vec 没有考虑词之间的顺序问题。之后，Bordes 等人受到 word2vec 启发，利用了词向量的“平移不变现象”,提出了 TransE 模型[56]。TransE 的提出是为了解决多关系数据处理的问题。

TransE 模型如图 2-8 所示，$(\boldsymbol{h},\boldsymbol{r},\boldsymbol{t})$ 是知识图谱中的三元组，$\boldsymbol{r}$ 是实体 $\boldsymbol{h}$、$\boldsymbol{t}$ 之间的关系，TransE 模型将关系 $\boldsymbol{r}$ 看做实体 $\boldsymbol{h}$ 到尾部实体 $\boldsymbol{t}$ 的距离，通过不断调整 $\boldsymbol{h}$、$\boldsymbol{r}$、$\boldsymbol{t}$，使得 $h+r$ 尽量与 $\boldsymbol{t}$ 相等，如表达式 2-1 所示。

$$\boldsymbol{h}+\boldsymbol{r}\approx\boldsymbol{t} \tag{2-1}$$

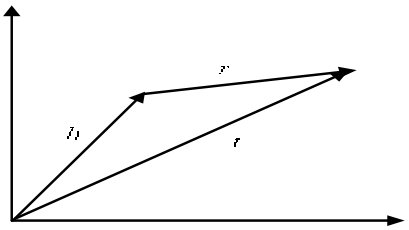


图 2-8 TransE 模型

TransE 模型参数较少，计算复杂度低，并且具有一定的可扩展性，在知识图谱的构建中得到了广泛应用。但是 TransE 模型不能处理复杂的关系，因此不少学者在之后又提出了 TransR、TransH 等翻译模型，另一方面，由于 TransE 模型的可扩展性，也有不少例如 MTransE 等 TranE 的扩展模型被提出。

2.4.2.2 基于属性嵌入的实体对齐算法

本文在文献[16]提出的基于属性嵌入的实体对齐算法基础上，针对求职领域做出了一定的调整，利用属性信息来辅助实体的向量化学习，完成实体对齐，完成了职位知识图谱的冗余消除。

通过上一章节对数据的预处理，采集到的数据属性命名格式基本一致，然后对三元组划分为属性信息和结构信息，属性信息表示实体的属性信息，结构信息表示实体与实体之间的信息。对于职位领域知识图谱，属性信息包括薪资、工作经验、职位描述等；结构信息包括公司、行业、工作地点等。然后分别进行结构嵌入和属性嵌入的训练，将属性嵌入得到的向量与结构训练得到向量进行相似度计算，使得同一个实体在不同训练模型中的向量表示接近，直至模型收敛，得到实体的向量表示。最后通过计算向量相似度，找到实体对齐对，使用相似度阈值进行过滤，消除冗余的实体信息，完成实体的对齐，流程如图 2-9 所示。

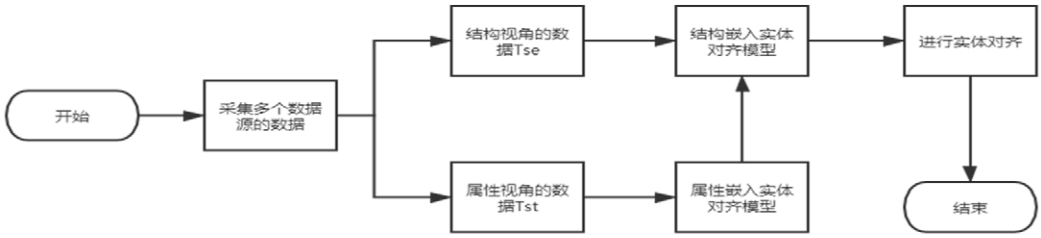


图 2-9.基于属性嵌入的实体对齐框架

传统的基于知识表示的实体对齐算法，如 TransE、TransR 一般只是用了结构信息，考虑到求职领域，职位的属性信息一般比结构信息更加相似，对于一个职位，即使它被发布在不同平台，但是它的薪资、工作地点等属性信息一般都不会修改。因此本文的向量表示学习使用了结构和属性两个方面的信息，通过属性嵌入来约束结构嵌入，使得同一实体在属性嵌入训练和结构嵌入下的向量表示接近，最终得到实体的向量表示。结构嵌入模块的知识表示算法是通过 TransE 实现，目标函数如式 2-2 所示:

$$J_{SE} = \sum_{t_r \in T_r} \sum_{t_r^{'} \in T_r^{'}} \max(0,\ \gamma + f(t_r) - f(t_r^{'})) \tag{2-2}$$

$$T_r = \{< h, r, t > | < h, r, t > \in G\} \tag{2-3}$$

$$T_r^{'} = \{< h', r, t > | h' \in E\} \cup \{< h, r, t' > | t' \in E\} \tag{2-4}$$

$$f(t_r) = \| h + r - t \| \tag{2-5}$$

其中 T_r 表示有效三元组集合，$T_r^{'}$ 表示负三元组集合，γ 是 margin 超参数。与结构嵌入的训练算法相似，实体属性嵌入的训练也使用了 TransE 的训练思路，但与结构嵌入不同的是，对于相同含义的属性值，可能在不同的数据源中表现形式存在差别。因此，本文参考了文献[16]提出的属性值组合函数。在组合函数对属性值进行编码整合之后，使得 $h + r \approx f_a(a)$，其中 $f_a(a)$ 是组合函数，表示属性值的字符串集合 $a = \{c_1, c_2, ..., c_n\}$，$c_i$ 表示属性的一个字符值。文献[16]提出了三种组合函数。

(1) 总和函数 SUM，定义为属性值的所有字符嵌入的总和。但是如果存在两个包含相同字符集且具有不同顺序的字符串，将存在具有相同的向量表示的问题。 例如，两个“50.15”和“15.05”两个数字将具有相同的向量表示。

$$f_a(a) = c_1 + c_2 + c_3 + ... + c_t \tag{2-6}$$

(2) 基于 LSTM 的组合函数（LSTM）。为了解决 SUM 的问题，提出了一种基于 LSTM 的组合函数。此函数使用 LSTM 网络将字符序列编码为单个向量。f_{lstm} 表示 LSTM 网络。

$$f_a(a) = f_{lstm}(c_1, c_2, c_3, ..., c_t) \tag{2-7}$$

(3) 基于 N-gram 的组合函数(N-gram)。基于 N-gram 的组合函数也是为了解决 SUM 问题的替代方法。其中 N 表示 N-gram 组合中使用的 n 的最大值，t 是属性值的长度。

$$f_a(a)=\sum_{n=1}^{N}(\frac{\sum_{i=1}^{t}\sum_{j=i}^{n}c_j}{t-i-1}) \tag{2-8}$$

针对求职领域，职位信息中数字信息较少，且年薪、工作年限很少出现总和函数 SUM 中的字符问题。因此采用第一种总和函数 SUM 作为属性嵌入的组合函数，本文使用的组合函数如式 2-9，属性嵌入的目标参数为式 2-10。

$$f_a(a)=c_1+c_2+c_3+...+c_t \tag{2-9}$$

$$J_{CE}=\sum_{t_\alpha\in T_\alpha}\sum_{t_\alpha^{'}\in T_\alpha^{'}}\max(0,[\gamma+f(t_\alpha)-f(t_\alpha^{'})]) \tag{2-10}$$

$$T_a=\{<h,r,a>\in G\} \tag{2-11}$$

$$T_a^{'}=\{<h^{'},r,a>|h^{'}\in E\}\cup\{<h,r,a^{'}>|a^{'}\in A\} \tag{2-12}$$

$$f(t_a)=\| h+r-f_a(a)\| \tag{2-13}$$

其中 T_a 是数据集的有效属性三元组的集合，而 $T_a^{'}$ 是被破坏的属性三元组的集合，通过用随机实体或具有随机属性值的属性替换头实体，将损坏的三元组用作负样本。a 表示职位所有属性的集合。$f(t_a)$ 是基于头实体 h 的嵌入，关系 r 的嵌入以及使用组合函数 $f_a(a)$ 计算的属性值的向量表示的似然性得分。结构嵌入的训练算法使用了 TransE 的算法思路，采用随机梯度下降（Stochastic Gradient Descent，SGD）算法对模型进行训练。流程如算法 2-1 所示。

算法 2-1：基于属性嵌入的职位知识图谱训练算法（结构嵌入训练模型）

输入：实体集 E，关系集 R，三元组集 T，间距 γ，学习率 α，向量维度 k，单批数据采样大小 B

输出：实体向量，关系向量

1 初始化：通过均一分布初始化 r∈ R， e∈ E

2 **Loop**：

3 e ←e/‖e‖

4 S_{batch} ←大小为 b 的样本(S,b)

算法 2-1：基于属性嵌入的职位知识图谱训练算法（结构嵌入训练模型）

5 T_{batch} ←初始化三元组

6 For $(h,r,t)\in S_{batch}$ do

7 $(h^{'},r,t^{'})$ ←对负例三元组采样 $S(h^{'},r,t^{'})$ //构建负样本

8 T_{batch} ← $T_{batch}\cup\{(h,r,t),(h^{'},r,t^{'})\}$

9 **End For**

10 最小化损失函数式 2-2 //SGD 优化参数

更新实体和关系的表示向量

11 **End Loop**

10 **return** 实体向量，关系向量

属性嵌入也使用了 TransE 的思想，与结构嵌入不同在于将三元组尾实体调整为组合函数，且目标函数为 2-10，具体的训练流程，如算法 2-2 所示。

算法 2-2：基于属性嵌入的职位知识图谱训练算法（属性嵌入训练模型）

输入：实体集 E，关系集 R，三元组集 T，间距 γ，学习率 α，向量维度 k，单批数据采样大小 B

输出：实体向量，关系向量，属性向量

1 初始化：通过均一分布初始化 $r\in R$， $e\in E$

2 **Loop**：

3 $e\leftarrow e/\|e\|$

4 S_{batch} ←大小为 B 的样本(S,B)

5 T_{batch} ←初始化三元组

6 For $(h,r,t)\in S_{batch}$ do

7 $(h^{'},r,t^{'})$ ←对负例三元组采样 $S(h^{'},r,t^{'})$ //构建负样本

8 T_{batch} ← $T_{batch}\cup\{(h,r,t),(h^{'},r,t^{'})\}$

9 **End For**

10 最小化损失函数式 2-10 //SGD 优化参数

更新实体、关系、属性组合函数的表示向量

11 **End Loop**

12 **return** 实体向量，关系向量

通过两个模型下的训练，得到实体的两个向量表示 h_{se} 和 h_{ce}，通过在属性嵌入下的训练帮助结构嵌入完成训练，联合训练的目标函数如下。

$$J_{Sim} = \sum_{h \in G} [1 - \|h_{se}\|_2 \bullet \|h_{ce}\|_2] \tag{2-14}$$

最终的目标函数如式 2-15。

$$J = J_{CE} + J_{SE} + J_{Sim} \tag{2-15}$$

通过上述训练过程之后，不同数据源中的实体将有相似的向量表示，通过余弦相似度 $\cos(h_1, h_2)$ 来计算获得潜在的实体对齐对，并设置阈值进行过滤，最终得到实体对齐结果。

2.5 求职领域的知识图谱构建与存储

Neo4j[60]是一种高效的 NOSQL 图形数据库，作为图数据库，Neo4j 最大的特点是关系数据的存储，例如对于职位信息，除了要存储职位信息、公司信息等基本信息，更重要的是存储职位、公司等之间的关系信息。在具有高连接关系的数据处理方面，Neo4j 的速度比关系型数据库快千倍，可以实现高效地检索数据。图数据库在处理实体之间存在复杂关系的数据具有很大的优势，并且在现有的图数据库中，Neo4j 的查询和存储性能也是优良的[40]，因此本文采用 Neo4j 图数据库作为职位知识图谱的存储载体。Neo4j 的数据由节点、边构成，节点是主要的数据元素，表示一个实体。边表示实体之间的联系。根据职位知识图谱的结构，对图数据库的结构进行如下设计：

节点：知识图谱中的实体和属性，包括薪资、工作年限、工作内容、工作地点、行业、公司、城市。

关系：职位的属性关系和对象关系。职位包含的关系有职位所在城市，职位所属公司，职位所属行业，公司所属类型等。

py2neo 是用来对接 Neo4j 的 Python 库，本文采用 python 语言的 py2neo 包将抽取的职位知识导入到图数据库中。图数据库中的节点具有类型名，关系具有关系名，通过之前的对齐处理，删除了文本中冗余三元组信息，得到最终的三元组文本信息，然后将三元组信息导入到图数据库。具体的过程如下。

(h, r, t) 表示一条职位信息的三元组，h 是职位名称，r 表示关系，t 表示属性值。数据导入的流程如下：

（1）从文本中获取一条三元组数据 (h,r,t) 。

（2）获取头节点 h，判断图数据库中 h 是否存在，不存在则创建一个新的节点，并设置节点的标签和值。

（3）然后获取 t 的值，判断图数据库中 t 是否存在，不存在则创建一个新的节点，节点的值是 v。

（4）获取关系 r 的值，判断关系 r 在图数据库中是否存在，不存在则创建一条从 s 指向 v 的关系，关系的标签是类型。

（5）重复（1）-（4）步，直到文本文件中所有的三元组都存储到图数据库。

通过上述的操作，将所有的三元组信息导入到 Neo4j 图数据库中，并以可视化的形式展示。图 2-10 展示了本文构建的职位知识图谱的局部图形。

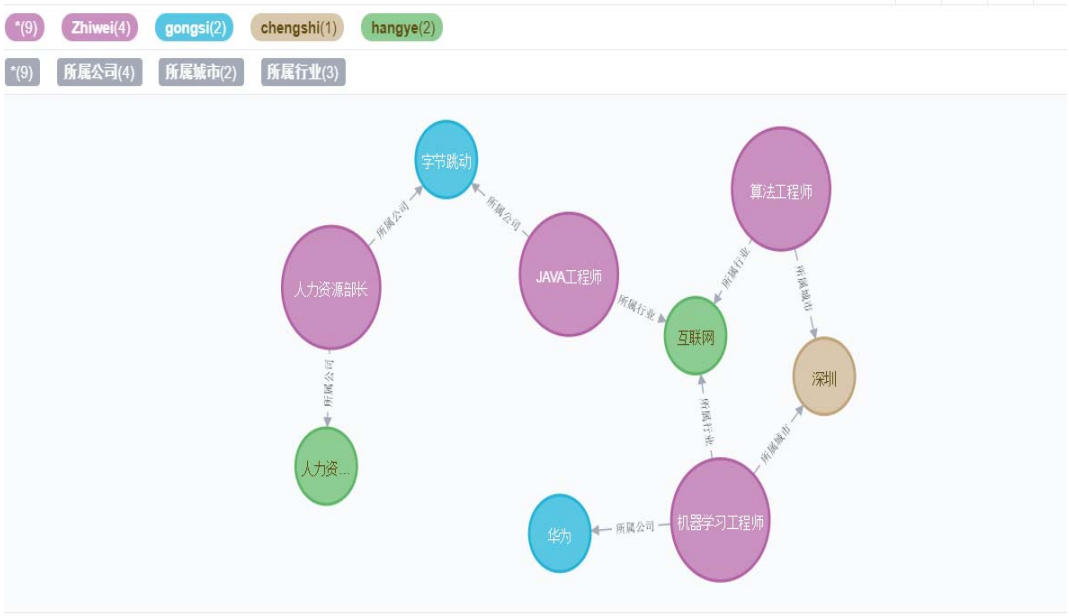


图 2-10 职位知识图谱局部示意图

通过上述示例，本文基于职位本体抽取知识，完成了职位知识图谱的构建工作。本文的职位知识图谱中既包含了职位实体，又包含了职位的属性实体以及职位与属性之间的关系。

2.6 本章小结

本章主要构建了职位知识图谱，描述了职位领域的语义信息。本文采用自顶向下的构建流程。首先构建了求职领域的本体，通过本体指导，对采集得到

的职位信息分析和处理，抽取得到了职位知识图谱的三元组信息。本章设计了一种融合属性和结构信息进行对齐的实体对齐方法，将采集的三元组数据分为结构数据和属性数据，将两部分数据进行知识表示训练，使得训练得到的职位实体向量表示相似，计算职位向量相似度进行实体对齐，最终得到对齐后的三元组。相比传统知识表示的对齐算法，本章的算法引入了属性数据，训练得到的向量在对齐效果上更佳。最终将三元组数据存储到 Neo4j 图数据库中，并进行可视化展现。

第 3 章 用户兴趣建模

通过用户反馈信息构建用户兴趣模型，是实现精准推荐的关键。用户兴趣建模是对用户偏好信息的一种表示，将用户信息表示为计算机能够识别计算的形式。本文利用知识表示技术，使用了基于向量空间的模型表示思想，利用知识图谱将用户偏好向量化，通过向量表示用户偏好，方便后续的推荐工作。

3.1 用户兴趣建模技术

用户兴趣建模是指用兴趣表示方法描述用户兴趣的过程[74]。用户兴趣建模的实现步骤为获取用户的信息，通过采集用户历史行为或者用户主动反馈信息得到用户的项目偏好，然后使用兴趣表示方法将偏好信息表示成计算机能够识别计算的形式。

获取用户的偏好信息的方法分为显式获取和隐式获取。显式获取主要是获取用户注册、使用、反馈时主动提供的信息，这种方法比较简单，用户偏好信息准确率较高，但是可能会涉及用户隐私，同时不断要求用户主动反馈信息也会加重用户负担。隐式获取主要通过用户在系统中评价、浏览等行为获取用户偏好信息，例如用户在淘宝的评价记录、用户对电影的评分等，利用用户的隐式反馈挖掘信息用户的兴趣偏好，隐式获取不需要用户主动参与，减小了用户的负担，但是由于不是用户主动提供，在计算用户兴趣偏好时中可能存在误差，影响兴趣建模的准确度。

目前，用户兴趣表示方法有主题词表示法、主题法、基于向量空间表示法等。关键词表示法使用关键词构建列表来表示用户兴趣，关键词可由用户主动提供或者通过用户历史行为隐式获取，例如用户求职时倾向北京计算机相关的职位，并进行了大量的计算机职位的申请行为，则关键词列表为{“北京”，“计算机”}，而用户在求职中填写了对“java 开发工程师”有兴趣，则关键词列表为{“北京”，“计算机”，“java 开发工程师”}。主题表示法与主题词列表法相近，但是只是使用领域关键字进行粗略的兴趣表示，例如用户喜好互联网行业，则用户兴趣可表示为{“互联网”}。基于向量空间模型表示方法（Vector Space Model，VSM）将文本信息进行向量化处理，通过计算向量的相似度度量文本相似度，

向量空间模型通常将用户兴趣表示成用户关键词向量 T 和用户对关键词的兴趣度向量 W，通过计算两者的相似度来评估用户兴趣相似度。

在职位推荐中，用户往往更关注部分属性信息，如有些用户更偏爱北上广等大城市的工作，有些用户对薪资区间有严格的要求，如果能够充分利用这些属性信息，就能够更加准确的表示用户的兴趣偏好。因此在推荐的过程中，本文引入了“属性偏好”的概念，将用户对职位的偏好细化为用户对属性的偏好。

本文运用了 VSM 的思想，采集用户对职位的评价信息，分析了职位中各个属性影响程度，计算职位中各个属性的相应权值，计算相应的加权和得到用户属性偏好向量，用来表示求职领域的用户兴趣特征。

3.2 用户兴趣向量构建

针对基于知识图谱的推荐中没有考虑实体属性的问题，在求职领域中，本文改进并使用了一种基于知识图谱中实体属性的用户兴趣建模方法。利用知识表示将职位的属性三元组向量化，计算各个属性项的权重并加权求和，得到用户兴趣模型。

由于求职领域的隐私性较强，大部分的求职网站无法获取到用户主动反馈的信息，因此采用隐式反馈信息作为用户偏好信息。同时用户历史访问记录较难获取，但是可以拉勾网、牛客等网站中获取得到用户面试评价信息，用户面试评价信息表示的是用户完成相关职位面试后的意见评价，如果用户评价了某一职位，则表明用户申请过该职位，并对该职位有一定的兴趣，因此本文选取用户职位面试评价信息作为用户的反馈信息，采集的用户评价实例如图 3-1 所示。

图 3-1 用户评价信息实例

从图 3-1 中可看出用户的面试评价信息中包括用户名称、职位、评分、评价内容等，并且可以从评价的网页中获取职位的详细数据。通过分析用户申请过职位的属性信息，将职位的属性向量化后加权处理，从而构建用户兴趣向量。

职位属性三元组是以职位知识图谱中的职位为头实体，职位属性为关系，职位属性值为尾实体的三元组。通过知识表示的训练算法得到的职位属性三元组的向量表示，计算用户属性三元组集中每一种属性的权重，将所有属性值向量加权和，便可以得到用户兴趣向量。表示学习训练得到的是语义向量，因此通过表示学习构建的用户兴趣向量也是一种语义用户兴趣向量，包含了用户对职位中属性的喜好程度。本文的用户兴趣模型如图 3-2 所示。

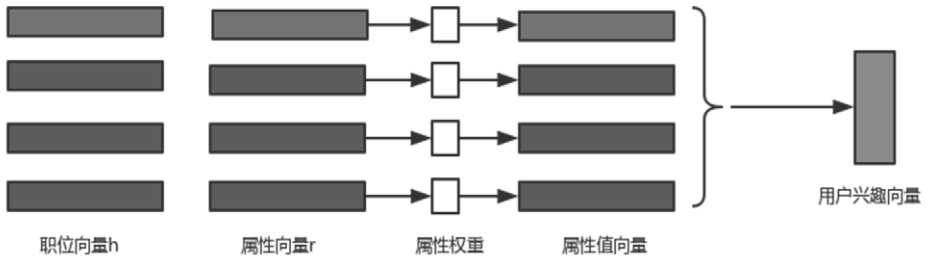


图 3-2 用户兴趣模型

3.3 基于知识图谱的用户兴趣建模

要构建用户兴趣向量，就需要分析用户更关注哪些职位属性。为了解用户的属性偏好，本文参考了 58 同城上的用户简历信息进行了分析，表 3-1 为采集的用户简历信息示例。

表 3-1 用户简历信息实例

用户用例	简历信息
刘*	刘*、女、27、中专/技校、5-10 工作经验；期待职位：淘宝客服、店员/营业员、电子商务；期待薪资：5000-8000
朱巧*	朱巧*、27、大专、5-10 工作经验；期望职位：计调；期望工作地点：宜昌；期待薪资：3000-5000

从表 3-1 中可知，用户往往对工作有期望的薪资、期望的工作地点,而且用户一般都是在自己擅长的行业领域进行求职。同时在求职过程中，用户会关注

自己的年龄、工作经验是否符合目标职位要求。从第二章采集的职位信息可知，职位实体一共包含薪资、工作地点、工作经验、学历、发布时间、职位描述、公司、行业 8 个属性，用户在求职中往往比较关注部分重要的属性，如薪资、工作内容等，而发布时间对用户求职几乎没有影响，因此综合用户简历信息和职位信息，本文主要分析薪资、工作地点、工作经验、学历、公司、行业这 6 个属性对用户求职的影响程度。下面对各种属性逐一进行分析:

1. 工作地点：用户求职时往往对工作的城市有一定要求，如部分用户喜欢北上广深这种大城市的工作，而部分用户更倾向于离家近的工作。

2. 薪资：用户对工作的薪资一般都有一定的要求，一般为范围区间。相似职位的工资区间也会比较接近。

3. 公司：部分用户更钟爱一些公司的工作，在求职时往往对公司有所偏向。

4. 行业：一般用户的技能都积累在自己的专长领域，应该寻找的工作也固定在特定行业领域。

5. 要求工作经验：用户一般都会根据本身情况选择符合要求工作经验的工作。

6. 要求学历：用户在求职时也更倾向选择满足学历要求的岗位。

用户对职位的关注往往集中在这几种属性上面。计算用户的职位偏好，也就是计算用户对这六个属性的偏好的向量表示。在第二章中本文构建了基于属性嵌入的实体对齐算法，其中将职位信息分为了结构信息和属性信息分别训练，得到向量表示，并进行了实体对齐。本文使用实体对齐完成时得到的职位实体、关系、属性的向量表示，根据得到的向量表示计算用户兴趣向量。职位属性三元组集如公式 3-1 所示。v_u 是用户 u 的历史面试记录的职位集,T_u 表示用户 u 访问的职位集包含的三元组信息。

$$T_u = \{(h,r,a) \mid h \in v_u\} \tag{3-1}$$

其中，(h,r,a) 表示用户进行评价过的职位属性三元组，h 表示职位实体，r 表示关系，t 表示职位的属性值。由第二章的基于属性嵌入的知识表示得到了所有三元组的向量表示并进行了编号标记。从得到向量表示集中获取用户评价的职位三元组的向量表示。通过职位、属性和属性值的向量表示，使用式 3-2 计算得到职位 h 中属性 a 的权重如下所示：

$$w_a = \frac{\exp(h^T r_a)}{\sum_{(h,r,t)\in T_a} \exp(h^T r)} \tag{3-2}$$

其中，h、r_a 分别为通过属性嵌入训练得到的职位和关系的向量表示。通过式 3-2 求出薪资、工作地点、工作经验、学历、公司、行业权重后，再对用户历史评价职位集中所有属性值进行加权求和，得到用户的兴趣模型，计算过程如式 3-3 所示,其中 a 为训练得到的属性值的向量。

$$c_u = \sum_{(h,r,a)\in T_u} w_a a \tag{3-3}$$

综上所示，用户兴趣建模的步骤如下：

（1）读取每一个用户，获取该用户的历史面试的职位记录，通过第二章中的基于属性嵌入的知识表示得到用户的职位属性三元组的向量表示。

（2）结合用户简历信息，分析了求职过程中影响用户求职的各个属性，并通过得到的职位知识图谱的实体向量、关系向量计算用户属性三元组中每一个属性的权重。

（3）使用计算得到的各个属性的权重，将对应用户职位的属性的向量表示加权求和，计算得到各个用户兴趣向量，即用户兴趣模型。具体过程如算法 3-2 所示。

算法 3-2：用户兴趣建模算法

输入：职位知识图谱实体集 E ，关系集 R ，用户历史面试职位集 v_u

输出：用户兴趣向量 c_u

1 **for** u **in** U **do** //获取用户

2 **for** v **in** v_u **do** //获取用户历史评价职位集 v_u

3 **for** (h,r,a) **in** T_u **do** //获取职位属性三元组集 T_u

4 使用公式 3-2 计算属性 a 的权重

5 **end if**

6 **end for**

7 **end for**

8 通过公式(3-3)计算用户 u 的兴趣向量

9 **end for** //读取完所有的用户

10 **return** c_u

3.4 本章小结

本章主要是对用户兴趣建模技术进行分析，发现在求职领域，用户往往更关注职位的属性信息，因此本文提出了一种基于知识图谱的用户兴趣建模方法，采用向量空间建模技术，利用知识图谱的知识表示得到职位三元组的向量表示，使用训练得到的向量，分析了影响用户选择的各个职位属性，计算了各个属性的权重，通过加权和计算得到了用户的兴趣向量。

第 4 章 基于知识图谱的个性化职位推荐算法

针对协同过滤算法没有考虑职位的语义信息，而知识图谱能够弥补协同过滤的数据稀疏问题，因此本章提出一种基于知识图谱的个性化职位推荐算法，使用知识图谱挖掘出用户喜好相似度和职位的语义相似度信息，在传统协同过滤中融入利用知识图谱得到的相似度信息，进而提升职位推荐系统的性能。

4.1 协同过滤算法综述

在生活中，人们往往会根据群体智慧来做出一些选择，比如人们会去听当前热门的歌曲、去买新潮流行的衣服，协同过滤推荐技术正是考虑了这种“群体智慧”的想法，“群体智慧”指的是从大量的用户行为或者项目数据中挖掘共性，协同过滤在新闻推荐、电子商务、网络购物大放异彩。

协同过滤算法的分类如图 4-1 所示，其中基于用户的协同过滤是从大量用户中找到一部分与目标用户品味相似的，然后将他们喜好的物品组织排序后进行推荐。基于用户的协同过滤能够帮助用户发现新的项目类别，但是存在用户冷启动问题，而且得到的推荐结果可能让用户难以理解，例如用户喜好苹果，而他的相似用户还喜欢手机，则可能将与苹果完全不相关的手机推荐给用户。

基于项目的协同过滤多用于网上书屋、电商系统等领域，该算法通过用户历史行为，计算职位间的相似度矩阵，将于用户偏好接近的项目推荐给用户。该算法推荐结果会随着用户行为改变，具有一定的时效性，并且推荐结果也更有解释理由，能够让用户信服，但是同样存在物品冷启动问题和计算效率问题，推荐的项目也与用户历史喜好相近，不会推荐新的项目类别。

面对日益增多的用户，系统数据量增大，基于内存的协同过滤计算效率会逐渐下降，可能影响推荐效率。而基于模型的协同过滤推荐算法将用户历史数据划分为训练集和测试集，在不同的预测模型中进行离线学习，然后使用训练得到的模型，实时计算用户对未评分项目的预测评分，常用算法包括关联算法、矩阵分解、聚类算法等。这种方法不存在计算效率问题。

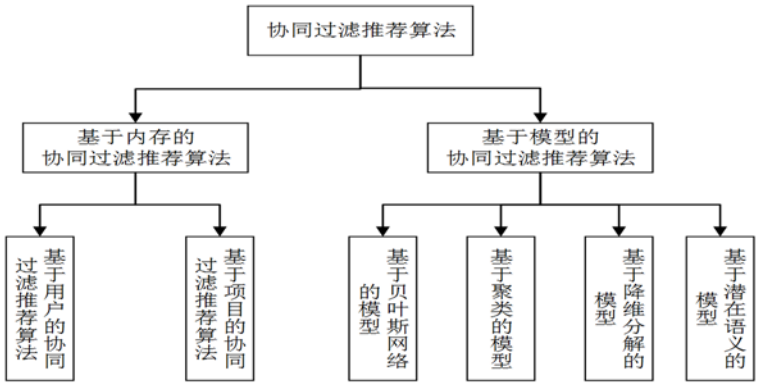


图 4-1 协同过滤算法分类

尽管协同过滤算法计算流程简单高效，在过往十几年间应用广泛，但伴随着大数据时代的来临，用户量与日俱增，协同过滤算法也暴露出了不少问题。

（1）稀疏性问题。一个推荐系统的用户或者项目都十分庞大，系统中会不断出现新的用户或者项目，系统中所有项目也不可能都被用户浏览访问过，因此相比于系统所有的用户和项目，能够获取的用户-项目交互记录是十分稀疏的。

（2）冷启动问题。对于一个初次使用系统的用户，系统中没有该用户的历史数据，那么协同过滤算法就很难进行相似度的计算，系统无法对其做出推荐；同样对于一个新项目，由于没有用户评价过，该项目就无法将推荐给用户。

（3）可扩展性问题。协同过滤推荐算法是通过用户与项目的交互记录进行计算的，随着系统长时间的运行，用户-项目的交互记录也越发庞大，推荐算法的计算量也急速上升，这些可能导致计算结果不及时，推荐效果偏移。但是现阶段主要通过提升计算机性能来缓解该问题，因此本文不考虑算法的可扩展性。

因此本文提出了一种个性化职位推荐算法。利用知识图谱三元组的实体、属性的语义信息，解决传统协同过滤在用户或职位信息量不足时无法进行推荐的问题，深层次的挖掘用户的职位喜好，得到更加准确、可解释的职位推荐结果。

4.2 基于知识图谱的职位推荐算法

本文在前两章已经构建了职位知识图谱并构建了用户兴趣模型，本章提出一种基于知识图谱的职位推荐算法，在职位推荐上，能够挖掘用户的潜在兴趣，

希望得到更好的推荐效果，并解决了冷启动和数据稀疏问题。本文提出算法具体流程如图 4-2 所示：

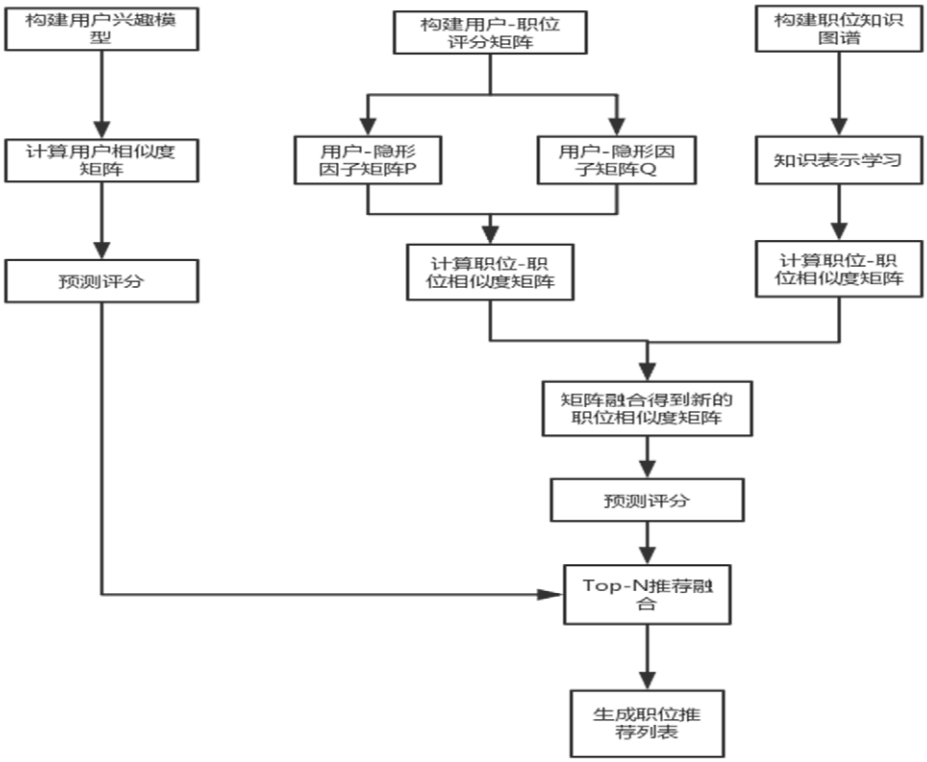


图 4-2 职位推荐算法

本文的推荐算法主要分为两个部分。首先在拉勾网上爬取的用户评价信息，使用矩阵分解计算得到构建用户评分矩阵并计算职位相似度矩阵。然后从之前构建的职位知识图谱通过知识表示得到职位的向量表示，计算职位的相似度，生成职位的语义相似度矩阵，将两者融合得到新的职位相似度矩阵，通过基于物品的协同过滤算法，选取评分高的 N 个项目。另一方面使用之前构建的用户兴趣模型，计算用户偏好相似度矩阵，通过用户相似度矩阵计算得到预测评分，选取评分高的 N 个项目得到推荐集。最终将得到的推荐集进行融合，得到推荐列表。

4.2.1 基于知识图谱职位相似度计算

为解决传统的协同过滤算法面临的冷启动问题，这里通过引入知识图谱丰富的语义信息，对于一个新项目，在用户职位评分数据不足时，通过语义相似度也能将新的项目推荐给用户，解决冷启动问题。在第二章中，本文构建了职

位知识图谱，并训练得到了实体、关系的向量表示。本文选取了采集数据中的 7 个职位以及 6 个用户的信息，对算法流程进行说明展示。通过训练得到的职位向量，然后使用训练得到的向量表示计算职位间的向量相似度。

表 4-1 职位实例信息

编号	职位名称	说明
1	Java 开发工程师	技术开发岗位
2	机器学习工程师	技术开发岗位
3	技术总监	技术、管理相关
4	UI 设计师	技术开发岗位
5	运营总监	运营、管理岗位
6	产品经理	运营、管理岗位
7	产品总监	运营、管理岗位

表 4-2 用户实例信息

编号	用户名	用户属性
1	朱巧玲	女、25、本科、无工作经验
2	杜宗艳	女、30、本科、3 年工作经验
3	赵威林	男、37、本科、5-10 年工作经验□□
4	周绍华	男、33、中专、5-10 年工作经验
5	宋传勇	男、24、硕士、2 年工作经验
6	张羽	男、23、大专、3 年工作经验

这里通过计算得到向量的距离来计算职位的相似度，有很多不同的距离的计算方法，如欧氏距离，马氏距离等。这里使用欧几里和距离来衡量职位向量的语义相似度，职位之间的距离，计算如式 4-1 所示：

$$d(I_i, I_j) = \sqrt{\sum_{k=1}^{d} (E_{ki} - E_{kj})^2} \tag{4-1}$$

$$sim_{kg}(I_i, I_j) = \frac{1}{1 + d(I_i, I_j)} \tag{4-2}$$

其中 E_{pi} 表示职位 I_i 的嵌入向量在第 p 维的值。然后通过式 4-2 将 $d(I_i, I_j)$ 的值约束在(0，1]之间，值越大表示两个职位越相似，当值接近 0 时表示两个职位相似度越小，1 表示两个职位完全相同。计算所有职位的相似值，并构建基于知

识图谱的职位相似度矩阵。最终表 4-1 中通过语义相似度计算得到的职位相似度矩阵如图 4-3 所示，矩阵中每个值表示职位 i 与职位 j 的相似度，取值在 0-1 之间，越接近 1 表示职位相似度越高，职位与本身的相似度为 1，由此可以看出 Java 开发工程师和机器学习工程师同属技术岗位，通过语义计算后相似度为 0.77，两者最为接近，而产品总监和运营总监语义相似度也比较高，达到了 0.72。

```
[[1.         0.77218734 0.46291005 0.33333333 0.10482848 0.54433105
  0.68138514]
 [0.77218734 1.         0.87287156 0.57735027 0.31807321 0.32075015
  0.6289709 ]
 [0.46291005 0.87287156 1.         0.57655666 0.46291005 0.54594868
  0.46291005]
 [0.33333333 0.57735027 0.57655666 1.         0.07647191 0.89104211
  0.72057669]
 [0.10482848 0.31807321 0.46291005 0.07647191 1.         0.54433105
  0.5       ]
 [0.54433105 0.32075015 0.54594868 0.89104211 0.54433105 1.
  0.27216553]
 [0.68138514 0.6289709  0.46291005 0.72057669 0.5        0.27216553
  1.        ]]
```

图 4-3 职位的语义相似度矩阵

4.2.2 基于用户行为的职位相似度计算

4.2.2.1 矩阵分解算法

数据稀疏是推荐算法一直存在的问题。本文通过引入矩阵分解模型对稀疏的评分矩阵进行填充，提高用户-职位评分矩阵的稠密度，通过矩阵分解算法得到了新的评分矩阵。然后使用填充过的评分矩阵进行相似度计算。

矩阵分解就是将原有的稀疏矩阵近似的分解成小矩阵的乘积。常用的矩阵分解方法包括奇异值分解（SVD）、SVD++算法和 FunkSVD 算法[52]。奇异值分解将矩阵分解成用户/项目隐含因子矩阵和对角矩阵的乘积，计算相对简单，但是 SVD 要求原矩阵是稠密的，在推荐算法中使用比较困难。BiasSVD 在计算中需要考虑其他限制条件，只适合应用于特定的场景。FunkSVD 算法是为了解决传统 SVD 计算效率问题而提出的，同时避开了数据稀疏的缺点，将原矩阵分解为两个低维矩阵，采用了线性回归的思想，在实际中应用十分广泛[52]。

FunkSVD 算法的思想是将评分矩阵分解成两个低维矩阵，表示用户潜在特征矩阵和项目潜在特征矩阵，如下所示：

$$M_{m\times n} = P^{T}_{m\times k} Q_{k\times n} \tag{4-3}$$

其中 m_{ij} 表示用户 i 对项目 j 的评分，通过 FunkSVD 进行矩阵分解，得到两

个低维矩阵，表示为$q_j^T p_i$。采用均方差做为损失函数，目标函数定义如式 4-4 所示：

$$\sum_{i,j}\left(m_{ij}-q_j^T p_i\right)^2 \tag{4-4}$$

实际运用时，为了防止过拟合，会加入一个正则化项，因此 FunkSVD 的最终优化目标函数为：

$$\sum_{i,j}\left(m_{ij}-q_j^T p_i\right)^2+\lambda\left(\|p_i\|_2^2+\|q_i\|_2^2\right) \tag{4-5}$$

其中$\|\cdot\|_F$是 F 范数，λ是正则化因子。通过随机梯度下降方法求解上式。

将上式分别对p_i，q_j求导得到：

$$\frac{\partial J}{\partial p_i}=-2(m_{ij}-q_j^T p_i)q_j+2\lambda p_i \tag{4-6}$$

$$\frac{\partial J}{\partial q_i}=-2(m_{ij}-q_j^T p_i)p_j+2\lambda q_i \tag{4-7}$$

则在梯度下降迭代时，p_i，q_j的迭代公式为：

$$p_i=p_i+\alpha((m_{ij}-q_j^T p_i)q_j-\lambda p_i) \tag{4-8}$$

$$q_j=q_j+\alpha((m_{ij}-q_j^T p_i)p_i-\lambda q_j) \tag{4-9}$$

本文将矩阵分解应用到用户评分矩阵中，采集得到职位评分数据构建用户评分矩阵后，使用 FunkSVD 对用户评分矩阵进行填充。

假设推荐系统中有 m 个用户、n 个职位，则 m 个用户对 n 个职位的原评分矩阵如式 4-10 所示,其中用户评分范围为 0-5,0 表示用户没有对相关职位进行评分。

$$R=\begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1n} \\ R_{21} & R_{22} & \cdots & R_{2n} \\ \cdots & \cdots & \cdots & \cdots, \\ R_{m1} & R_{m2} & \cdots & R_{mn} \end{bmatrix} \tag{4-10}$$

其中R_{ij}表示用户 i 对职位 j 的面试评分。通过矩阵分解可得到用户特征矩阵$P_{m\times k}$和职位特征矩阵$Q_{m\times n}$，使用式 4-8 和式 4-9 计算得到的p_i和q_j相乘得到预测评分值，得到稠密的职位评分矩阵$R_{m\times n}$。仍使用表 4-1 的实例信息，其中用户对职位的评分如式 4-11。图 4-4 则是程序通过矩阵训练得到的稠密矩阵，

图中每个值表示计算得到的预测评分，例如图中R_{13}的值 2.24 表示用户朱巧玲对技术总监这个职位的预测评分。通过矩阵分解得到了用户对未评分项目的预测评分，解决了数据稀疏的问题。

$$R = \begin{bmatrix} 5 & 3 & 0 & 1 & 0 & 2 & 3 \\ 4 & 0 & 0 & 1 & 0 & 3 & 0 \\ 1 & 1 & 0 & 5 & 1 & 0 & 1 \\ 1 & 0 & 0 & 4 & 0 & 0 & 5 \\ 0 & 1 & 5 & 4 & 5 & 2 & 0 \\ 2 & 1 & 0 & 4 & 5 & 0 & 4 \end{bmatrix} \tag{4-11}$$

```
[[4.69014085 2.95293516 2.24197375 0.73851045 0.4042431  2.47251104
  3.10450044]
 [4.2528121  2.72139796 2.61724086 1.22166079 0.94441168 2.44778302
  3.22026115]
 [0.66880066 0.61128829 2.8567806  2.50211948 2.56669215 1.24623104
  2.20221462]
 [1.54212373 1.30932851 5.25089377 4.5069957  4.59677163 2.40287448
  4.15113429]
 [0.8663315  0.8865978  4.96455102 4.43525473 4.57481802 2.05954232
  3.72926469]
 [1.62246314 1.35069934 5.16643372 4.40357515 4.4821899  2.40194981
  4.11910388]]

Process finished with exit code 0
```

图 4-4 矩阵分解后得到的稠密矩阵示例

4.2.2.2 职位相似度矩阵计算

一个用户对与历史喜好相似的职位可能更感兴趣，通过用户历史评分记录，就能对相似度较高的职位进行评分预测。本文采集了用户对职位的面试评价，评价信息包括职位的具体信息、用户名、评价时间、评分，构建用户评分矩阵，并通过矩阵分解填充稀疏矩阵，得到了新的稠密用户评分矩阵$R_{m\times n}$，利用式 4-12 可以计算得到基于用户行为的职位相似度,其中R_{kj}表示用户 k 对职位 j 的面试评价。

$$\mathrm{sim}_{cf}(I_i, I_j) = \frac{I_i \bullet I_j}{\|I_i\| \|I_j\|} = \frac{\sum_{k=1}^{m} R_{ki} \times R_{kj}}{\sqrt{\sum_{k=1}^{m} R_{ki}^2} \times \sqrt{\sum_{k=i}^{m} R_{kj}^2}} \tag{4-12}$$

计算结果越大，表明职位I_i和职位I_j基于用户行为的相似度越大，当值为 1 时，则认为两个职位相同，值为 0 则表示两个职位完全不同。算法 4-1 是职位相似度计算的流程。

算法 4-1：职位相似度计算

输入：相似性阈值 δ，用户面试评价数据集 U ,用户集 $User$，职位集 $Item$

输出：职位相似性矩阵 I

```
1  //通过用户对职位的评分构建用户-职位评分矩阵
2  for u in User do （读取每一个用户）
3   for i in Item do （读取每一个职位）
4     if(U contains (u,i,socre)) //评价集中包含用户对该职位的评分
5           R_ui = score
6   else R_ui =0
4   end for
5  end for
6  //通过矩阵分解稠密的用户-职位评分矩阵
7   for  step < steps do //进行矩阵分解的迭代次数
8      for  i  in  R  do
9          for  j  in  R  do
10             if( R_ij > 0 ) //有评分
11                 通过式 4-8 和式 4-9 计算 p_i 和 q_j
12          end for
13      end for
14     loss = 0 , num = 0  //初始化损失值 loss
15     for  i  in  R  do
16         for  j  in  R  do
17             通过式 4-5 计算得到损失函数值 loss_ij
18             loss += loss_ij
19         end for
20      end for
21     loss = loss / num
22     if（loss < 0.001）
23        break；得到用户潜在特征矩阵 P 和项目潜在特征矩阵 Q
24     end for
25     计算得到的用户潜在特征矩阵 P 和项目潜在特征矩阵 Q，通过相乘得到稠密的
```

算法 4-1：职位相似度计算

用 26　　户-职位评分矩阵 R

27　　根据公式(4-12)计算职位 i 和职位 j 的相似性 $sim_{cf}(I_i, I_j)$

28　　**if** $sim_{cf}(I_i, I_j) < \delta$ **then**

29　　$I_{ij} = 0$

30　　**else**

31　　$I_{ij} = sim_{cf}(I_i, I_j)$

32　　**end if**

33 **return** I

4.2.3 基于属性偏好相似度计算

第三章中，本文构建了用户兴趣模型，通过知识表示计算得到了用户偏好向量 C_u， C_u 如式 4-13。

$$\boldsymbol{c}_u = (e_{u1}, e_{u2}, ..., e_{uk}) \tag{4-13}$$

其中，e_{ui} 为向量在第 i 维的值。用户相似度的计算可以通过向量 C_u 的相似度计算取代。使用欧式距离来计算用户偏好相似性。计算用户偏好向量的距离如式 4-14 所示。

$$d(\boldsymbol{c}_u, \boldsymbol{c}_v) = \sqrt{\sum_{i=1}^{k} (e_{ui} - e_{vi})^2} \tag{4-14}$$

式 4-14 的值域是[0,∞]，而最终用户相似度的取值范围是[0,1]，因此使用式 4-15 将计算得到的值约束在(0，1]之间，得到用户 u 和用户 v 的相似性。

$$sim_{property}(u,v) = \frac{1}{d(\boldsymbol{c}_u, \boldsymbol{c}_v) + 1} = \frac{1}{1 + \sqrt{\sum_{i=1}^{k} (e_{ui} - e_{vi})^2}} \tag{4-15}$$

考虑到相似度较低的用户之间的影响很小，这些影响可能对推荐结果影响甚微，还影响了计算效率。因此设定了一个阈值 δ，当用户间偏好相似性高于阈值时，认为用户间的偏好是相似的，否则，认为用户的偏好不同，将用户将相似性设置为 0。最终用户 u、v 的偏好相似性表示如下：

$$s_{uv} = \begin{cases} sim_{property}(u,v), & if\ sim_{property}(u,v) \geq \delta \\ 0, & otherwise \end{cases} \tag{4-16}$$

用户 u 与用户 v 的偏好相似性表示为$\mathrm{Sim}_{\mathrm{property}}(\mathrm{v},\mathrm{u})$，用户 v 与用户 u 的偏好相似性表示为$\mathrm{Sim}_{\mathrm{property}}(\mathrm{v},\mathrm{u})$，且$\mathrm{Sim}_{\mathrm{property}}(\mathrm{v},\mathrm{u})=\mathrm{Sim}_{\mathrm{property}}(\mathrm{u},\mathrm{v})$。设置用户本身相似度为 1。最终计算得到用户相似度矩阵 S，如式 4-17 所示。

$$\boldsymbol{S}=\begin{pmatrix} 1 & s_{12} & \cdots & s_{1m} \\ s_{21} & 1 & \cdots & s_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ s_{m1} & s_{m2} & \cdots & 1 \end{pmatrix} \tag{4-17}$$

构造用户偏好相似性矩阵的算法如算法 4-2 所示。

算法 4-2：用户相似性矩阵算法

输入：相似性阈值δ，用户兴趣模型U_c，用户评价集U

输出：用户偏好相似性矩阵$\boldsymbol{S}$

```
for u in U do
  for v in U do
    从U_c中获取用户u的兴趣向量c_u和用户v的兴趣向量c_v
    根据公式(4-14)计算c_u和c_v之间的欧式距离
    根据公式(4-15)计算用户u和用户v的偏好相似性 sim_property(u,v)
    if sim_property(u,v) < δ then
      s_uv = 0
    else
      s_uv = sim_property(u,v)
    end if
  end for
end for
return S
```

4.2.4 相似度融合

在 4.2.1 节中基于知识图谱得到了职位相似度$\mathrm{sim}_{\mathrm{kg}}$，同时在 4.2.2 中使用协同过滤算法得到了职位相似度$\mathrm{sim}_{\mathrm{cf}}$，最终通过将两者融合得到最终的职位相似度。具体计算方式如式 4-18 所示：

$$\mathrm{Sim}(\mathrm{I_i},\mathrm{I_j})=\mathrm{aSim}_{\mathrm{kg}}(\mathrm{I_i},\mathrm{I_j})+(1-\mathrm{a})\ \mathrm{Sim}_{\mathrm{cf}}(\mathrm{I_i},\mathrm{I_j}) \tag{4-18}$$

其中 a 为融合因子，取值为[0,1]，表示基于知识图谱的职位相似度所占的比例。融合完之后职位的相似度以矩阵表示如下：

$$I=\begin{pmatrix} 1 & I_{12} & \cdots & I_{1n} \\ I_{11} & 1 & \cdots & I_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ I_{n1} & I_{n2} & \vdots & 1 \end{pmatrix} \tag{4-19}$$

其中 I_{ij} 表示职位 I_i 和职位 I_j 的相似度，当 i=j 是，I_{ij}=1，即职位与自身相似度为 1。

4.2.4 评分预测

通过上文得到的物品相似度矩阵和用户相似度矩阵，分别进行 Top-N 推荐并将结果进行融合。首先通过融合后得到的物品相似度矩阵，计算得到用户对职位的预测评分如式所示：

$$p_{ui}=\frac{\sum_{j\in N(u)\cap S(i,k)} sim(i,j)\times R_{ij}}{\sum_{j\in N(u)\cap S(i,k)} sim(i,j)} \tag{4-20}$$

其中，p_{ui} 表示用户 u 对职位 i 的预测评分，N(u)表示用户 u 评分过的物品集合，S(i,k)表示与物品 i 最相似的职位集合。通过将计算的预测评分排序，将前 N 个职位记录下来。通过图 4-5 和 4-6 得到的职位相似度矩阵融合，设置融合因子为 0.6，设置邻域为 1，融合因子会在后续实验部分进行讨论，使用式 4-20 就可以得到职位推荐列表，如表 4-3 所示。

表 4-3 基于职位相似度的推荐结果

用户名称	推荐列表	预测评分
朱巧玲	UI 设计师	3.4
杜宗艳	产品总监	4.2
赵威林	机器学习工程师	4.0
周绍华	机器学习工程师	2. 9
宋传勇	产品总监	4.9
张羽	产品经理	3.8

然后通过得到的用户相似度矩阵，通过式 4-21 计算用户对职位的预测评分：

$$p_{ui} = \overline{R_u} + \frac{\sum_{v \in s(u)} sim(u,v) \bullet (R_{vi} - \overline{R_v})}{\sum_{v \in s(u)} |sim(u,v)|} \tag{4-21}$$

其中，p_{ui} 表示用户 u 对职位 i 的预测评分，S(u)表示用户近邻集合 $\overline{R_v}$，$sim(u,v)$ 表示用户的偏好相似度，$\overline{R_u}$ 表示用户 u 的平均评分，$\overline{R_v}$ 表示用户 v 的平均评分。通过将计算的预测评分排序，将前 N 个职位记录下来。使用 4-21 对 4.2.3 节计算得到的用户相似度矩阵计算评分，如表 4-4 所示。

表 4-4 基于兴趣模型的推荐结果

用户名称	推荐列表	预测评分
朱巧玲	运营总监	4.1
杜宗艳	机器学习工程师	3.7
赵威林	技术总监	4.8
周绍华	运营总监	2.5
宋传勇	Java 开发工程师	3.5
张羽	技术总监	4.1

整个推荐算法流程为使用式 4-20 计算得到用户预测评分，按降序得到评分高的 N 个职位，生成推荐职位列表 L，同时使用 4-21 计算预测评分生成推荐职位列表 E，将两部分的推荐结果融合，生成最终的职位推荐列表，融合过程如算法 4-3 所示。算法主要思路为通过遍历依次将两个集合 L、E 中的物品放入 Top-N 推荐集合中，在放入推荐集合的过程中，要保证放入的对象不存在于推荐集合中，保证推荐集合中对象的唯一性。

算法 4-3：融合算法

输入：基于知识图谱和职位相似度的协同过滤近邻集：Set $L = \{L_0, ..., L_n\}$;

基于用户偏好相似度的协同过滤近邻集：Set $E = \{E_0, ..., E_n\}$

输出：推荐集 $C = \{C_0, ..., C_k\}$

```
1  for i<N do
2    If  L_i ∉ C
3      C append( L_i )
4    If Len(C)==k;break;
```

算法 4-3：融合算法

5 **If** $E_i \notin C$

6 C **append**(E_i)

7 **If** Len(C)==k;**break**;

Return Top-N 推荐集 C

4.2.5 职位推荐列表生成

具体的职位推荐流程如图 4-7 所示。

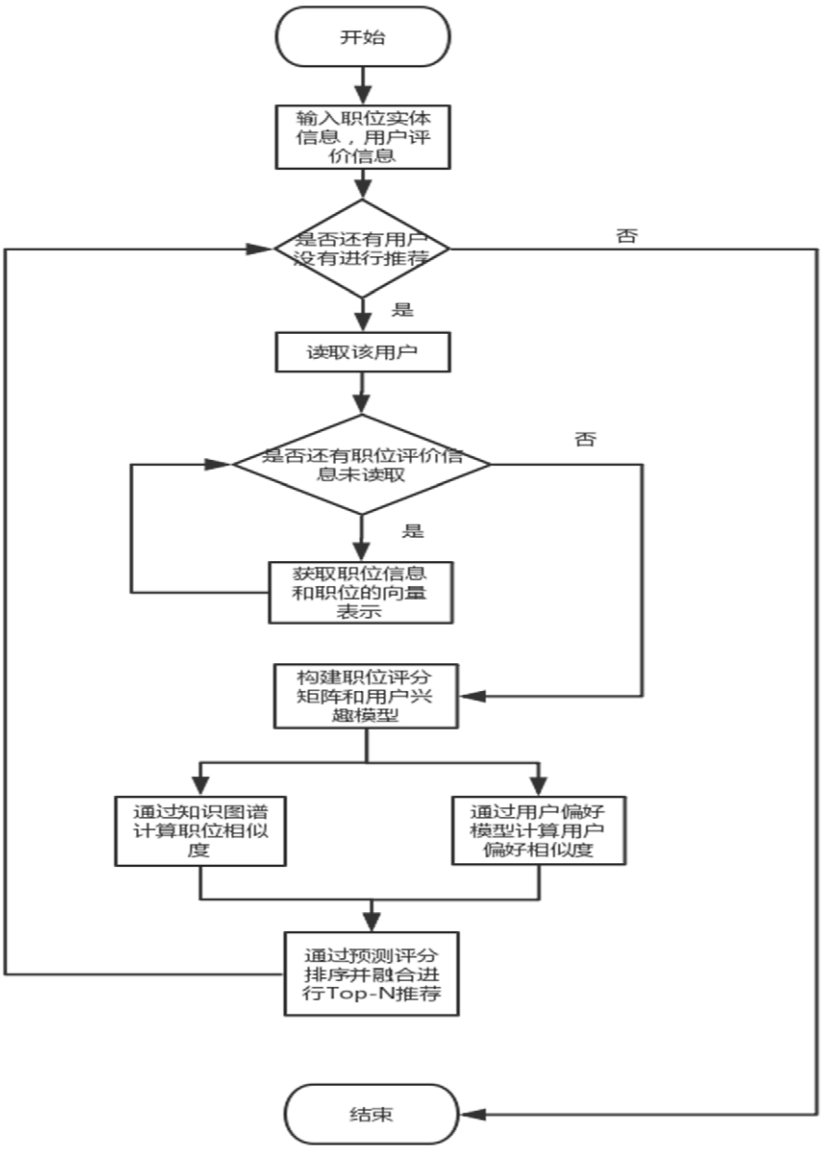


图 4-7 生成职位推荐列表流程图

在推荐过程中，采用 Top-N 推荐，读取每一个未推荐用户信息，获取该用户的兴趣模型，通过本文提出的推荐算法进行计算，为每个用户生成一个包含 N 个职位的推荐列表。按照预测评分对候选职位排序，用户对职位的评分越高，表明用户对职位越感兴趣，取评分高的 N 个项目生成推荐列表。通过该算法对得到的两个推荐列表进行融合，设置推荐个数为 2，得到最后的推荐列表 4-5。

表 4-5 最终推荐结果

用户名称	推荐列表
朱巧玲	UI 设计师、运营总监
杜宗艳	产品总监、机器学习工程师
赵威林	机器学习工程师、技术总监
周绍华	机器学习工程师、运营总监
宋传勇	产品总监、Java 开发工程师
张羽	产品经理、技术总监

4.3 本章小结

本章提出了一种基于知识图谱的职位推荐算法，将职位知识图谱的语义信息与传统协同过滤融合。算法分为两个部分，一方面使用第二章中知识图谱训练得到的向量计算职位相似度，同时基于用户历史评分计算职位相似度矩阵，并通过设置融合因子将两个矩阵融合，通过最终得到的职位相似度矩阵得到职位推荐集合；另一方面通过用户兴趣模型计算用户相似度矩阵，然后计算预测评分，取评分高的职位得到推荐集合，最终将两者的推荐结果融合得到最终的推荐结果。

第 5 章 实验验证与结果分析

本文在上面章节构建了职位知识图谱，并设计了一种新的职位推荐算法。本章分别设计了两种算法的实验流程，通过实验对比对所提出的方法进行验证和分析，验证了本文提出的算法在一定程度上能够提高推荐准确性。最后通过提出的推荐算法实现了一个职位推荐系统原型。

5.1 实验环境

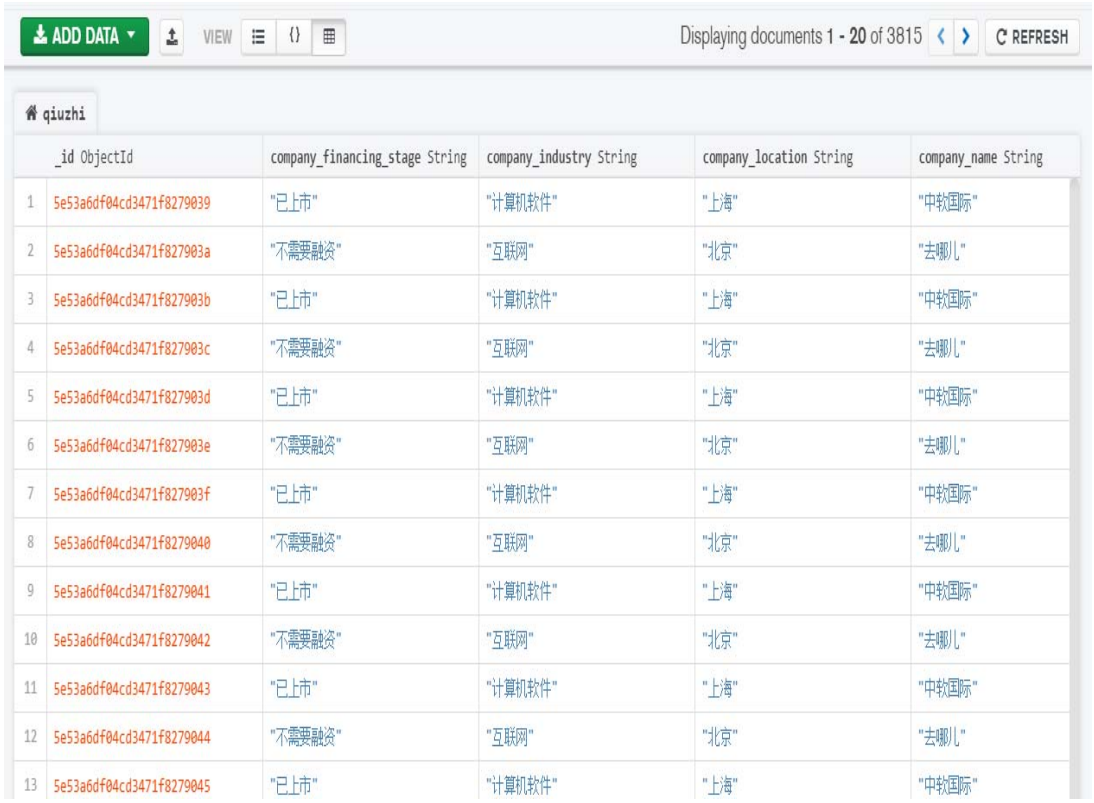
本文实验所依托的软硬件环境如表 5-1 ~~表 5-1~~所示。

表 5-1 实验环境

名称	型号说明
处理器	Intel(R) Core(TM) i5-8300CPU @2.30GHz
操作系统	Windows 10
开发语言	Python3
内存	8 GB
图数据库	Neo4j
开发工具	PyCharm 2019.3.2 x64
数据库	MongoDB

5.2 实验数据介绍与存储设计

实体对齐实验使用的数据集来自从三大求职网站采集得到的职位数据集，通过第二章中的数据处理后存储到 MongoDB 数据库中，一共采集到 76290 条职位的信息，职位信息包括职位名称、职位所属的公司、所属公司的类型、所在城市、工作地点、薪资、要求工作经验、职位所属行业等。图 5-1 为采集得到的职位信息数据节选。



	_id ObjectId	company_financing_stage String	company_industry String	company_location String	company_name String
1	5e53a6df04cd3471f8279039	"已上市"	"计算机软件"	"上海"	"中软国际"
2	5e53a6df04cd3471f827903a	"不需要融资"	"互联网"	"北京"	"去哪儿"
3	5e53a6df04cd3471f827903b	"已上市"	"计算机软件"	"上海"	"中软国际"
4	5e53a6df04cd3471f827903c	"不需要融资"	"互联网"	"北京"	"去哪儿"
5	5e53a6df04cd3471f827903d	"已上市"	"计算机软件"	"上海"	"中软国际"
6	5e53a6df04cd3471f827903e	"不需要融资"	"互联网"	"北京"	"去哪儿"
7	5e53a6df04cd3471f827903f	"已上市"	"计算机软件"	"上海"	"中软国际"
8	5e53a6df04cd3471f8279040	"不需要融资"	"互联网"	"北京"	"去哪儿"
9	5e53a6df04cd3471f8279041	"已上市"	"计算机软件"	"上海"	"中软国际"
10	5e53a6df04cd3471f8279042	"不需要融资"	"互联网"	"北京"	"去哪儿"
11	5e53a6df04cd3471f8279043	"已上市"	"计算机软件"	"上海"	"中软国际"
12	5e53a6df04cd3471f8279044	"不需要融资"	"互联网"	"北京"	"去哪儿"
13	5e53a6df04cd3471f8279045	"已上市"	"计算机软件"	"上海"	"中软国际"

图 5-1 职位信息数据节选

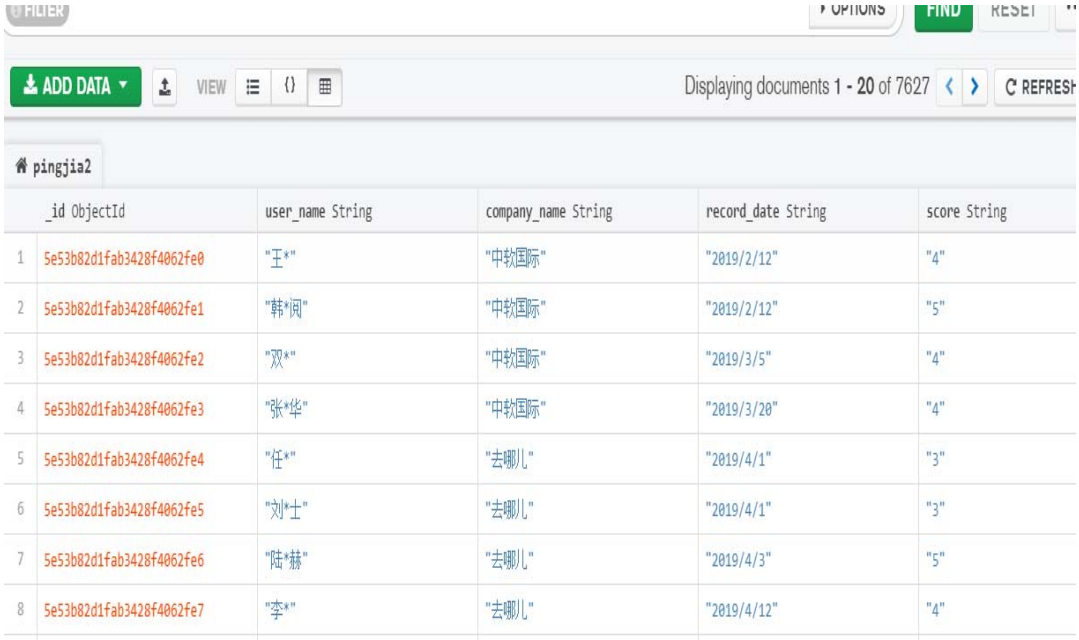
职位信息表的存储结构如表 5-2 所示，对于职位名称相同的信息，通过对职位信息加上数字编号，如“java 开发工程师”有多个，则表示为“java 开发工程师”，“java 开发工程师 2”，...，“java 开发工程师 n”。这样解决了数据项重复的问题，也方便后续的实体对齐处理。

表 5-2 职位评价表

字段	数据类型	可否为空	说明
Hiring_name	String	否	职位名称
Company_industry	String	否	所在的行业
Company_location	String	否	工作地点
Company_name	String	否	公司名
Company_people	String	是	公司人数
education	String	否	要求的学历水平
Company_stage	String	否	公司类型
Pay	String	否	薪资区间

Experiences	String	否	要求的工作经验
Position_info	String	是	职位描述
Record_date	Date	否	发布时间

推荐算法的实验数据来自采集得到的用户面试评价数据。本文从拉勾网采集了用户面试评价记录，从中抽取用户隐式反馈数据，并把抽取到的数据存储到 MongoDB 数据库中，共有 24812 名用户对 5423 个职位的 152540 条评论数据，图 5-2 是采集的评分数据节选。



	_id ObjectId	user_name String	company_name String	record_date String	score String
1	5e53b82d1fab3428f4062fe0	"王*"	"中软国际"	"2019/2/12"	"4"
2	5e53b82d1fab3428f4062fe1	"韩*闵"	"中软国际"	"2019/2/12"	"5"
3	5e53b82d1fab3428f4062fe2	"双*"	"中软国际"	"2019/3/5"	"4"
4	5e53b82d1fab3428f4062fe3	"张*华"	"中软国际"	"2019/3/20"	"4"
5	5e53b82d1fab3428f4062fe4	"任*"	"去哪儿"	"2019/4/1"	"3"
6	5e53b82d1fab3428f4062fe5	"刘*士"	"去哪儿"	"2019/4/1"	"3"
7	5e53b82d1fab3428f4062fe6	"陆*赫"	"去哪儿"	"2019/4/3"	"5"
8	5e53b82d1fab3428f4062fe7	"李*"	"去哪儿"	"2019/4/12"	"4"

图 5-2 评分数据节选

职位评价信息表的存储结构如表 5-3 所示。其中用户评分取值 0-5，0 表示用户没有对职位进行评价。

表 5-3 职位评价表

字段	数据类型	可否为空	说明
Username	String	否	用户名
Company_name	String	否	公司名
Record_date	String	否	评价的时间
Score	String	否	用户对职位的评分
Content	String	是	用户具体的评价

在推荐系统原型中，使用了职位信息表和用户信息表，如表 5-2 和表 5-4 所示。

表 5-4 用户信息表

字段	数据类型	可否为空	说明
Username	String	否	用户名
Sex	String	否	性别
Age	Int	否	年龄
Education	String	否	学历
Contact_information	String	是	联系方式
Experience	Int	否	工作经验

5.3 基于属性嵌入的实体对齐算法实验验证

5.3.1 实验设计

本文使用了 Guan 等人[76]提出的 Cocal 数据集和上一节采集得到了职位信息数据集进行实验。Cocal 数据集包括科研论文信息，共 288 个论文实体，在文献[47]中作者对其进行了划为，分为训练集 76 个，验证集 30 个，测试集 20 个。本文采集数据的三元组共有 7428 个节点，共有 35270 个三元组。

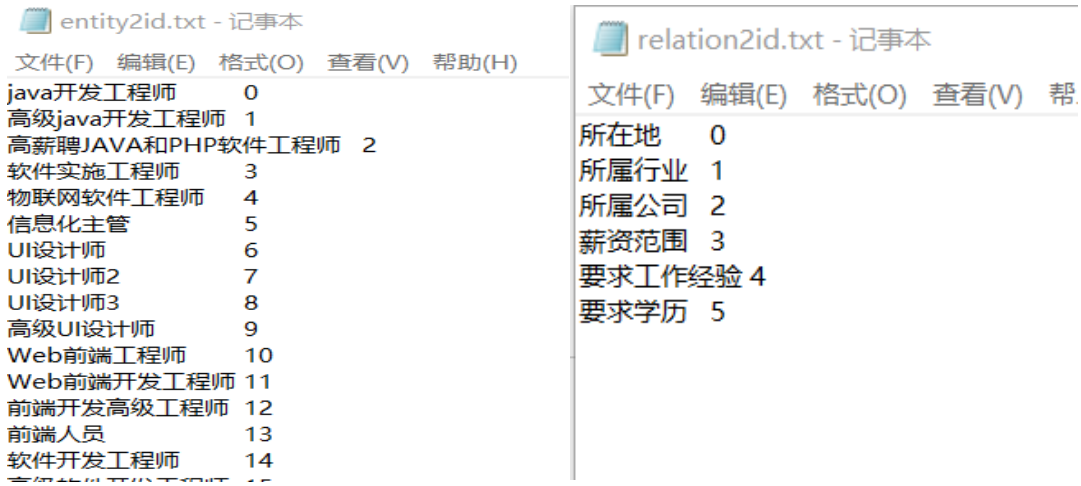


图 5-3 职位数据标记节选

职位的属性类型就是知识图谱中的关系类型,通过 2.2.2 节的分析，职位共有薪资、工作地点、工作经验、学历、发布时间、职位描述、公司、行业 8 个属性，其中发布时间和职位描述不作为实体对齐时使用的关系，因此每个实体共

有 6 种语义关系，本文采用的实体对齐算法将数据集划分为结构信息与属性信息，结构信息包括公司、行业、工作地点，属性信息包括薪资、工作经验、学历。

训练过程以三元组为单位进行，一共得到 24388 对三元组信息，为了方便训练，对节点和关系分别进行编号。如图 5-3 所示，实体包括职位、公司、工作地点、行业；关系包括所在地、所属行业、所属公司、薪资范围、要求工作经验、要求学历 6 个关系类型；属性包括薪资、工作经验、学历三个属性。在实体对齐的实验中，本文随机抽取了 500 个职位的三元组信息，人工标记实体对齐结果，然后把本文实体对齐方法与其他实体对齐算法进行对比。

为了进行实体对齐效果验证，从处理得到的三元组信息中抽取 500 个职位三元组，对抽取的职位三元组进行人工标记，标记出应该对齐的实体对，将标记好的 500 个职位实体的三元组信息按 8:1:1 的比例分割成训练集、验证集和测试集。数据统计如表 5-5 所示。

表 5-5 职位知识图谱数据统计

	三元组总量	标记职位数量	训练集	验证集	测试集
职位数据集	35270	500	2400	300	300
Coral 数据集	288	288	76	20	20

将标记好的三元组分为结构信息三元组和属性信息三元组进行训练，结构信息的训练使用了 TransE 算法，使用 python 的 init()和 norm()函数对实体、关系进行向量初始化和归一化，然后进行训练数据集的构建，从之前得到的三元组文件中读取正样本，然后随机选取正样本的三元组的第一项和第三项进行替换打碎得到负样本。得到了训练集之后，进行表示向量的更新，计算正负样本的 L1 距离范数，然后计算合页损失函数，对损失函数的 5 个参数进行梯度下降，得到向量的更新，更新时需要乘上学习速率，否则可能造成不收敛的情况。向量参数更新过程如图 5-4 所示，其中 (h,r,t) 表示正样本三元组，$(h^{'},r,t^{'})$ 表示负样本三元组，$learning\_rate$ 表示学习速率。其中 $h^{'}$ 和 $t^{'}$ 表示被打乱的头尾头尾实体。

在进行结构信息三元组训练同时，进行属性信息三元组训练。其中初始化和负样本获取基本相同，不同是尾实体是初始化每个字符的向量表示，通过 $f_a(a)=c_1+...+c_n$ 得到每个属性的向量表示，其中 c_i 表示字符 i 的向量表示。向

量更新时，对实体 h 和关系 r 更新与结构信息三元组相同，而对尾实体 t 的更新修改为对属性 a 中的所有c_i进行更新，然后计算得到新的属性向量表示，最后进行归一化处理。所有三元组训练完成后，计算损失函数式 2-17。对训练集进行 1000 次迭代训练，最终得到实体的向量表示，计算实体向量的相似度，通过相似度阈值过滤得到最终的对齐实体对。将得到的对齐实体对与标记的实体对对比，得到正确对齐的实体对数量，计算得到实体对齐算法的准确率和召回率。

5.3.2 评价指标

本文使用准确率（Precesion）、召回率（Recall）和综合指标 F1 值评价实体对齐效果，计算公式如下：

1) 准确率，表示通过实体对齐后得到的正确对齐的实体数和所有参与对齐实体数的比率

$$precision = \frac{N_c}{N_a} \tag{5-1}$$

2) 召回率，表示通过实体对齐后正确对齐的实体数和数据集中可对齐实体数的比率

$$recall = \frac{N_c}{N_m} \tag{5-2}$$

3) 综合指标 F1 值，是衡量准确率和召回率的综合指标

$$F = \frac{2 \times precison \times recall}{precision + recall} \tag{5-3}$$

其中，N_c表示正确对齐的实体数，N_a表示实际对齐的实体数，N_m表示标注的测试集中可以对齐的实体数。

5.3.3 实验对比和参数设置

为了比较本文改进的实体对齐算法的效果，本文采用基于字符串相似度的实体对齐算法、基于 TransE 的实体对齐算法[56]和本文提出的对齐模型分别训练职位知识图谱，通过实验对比三种训练模型的结果。

1) 基于字符串相似度的实体对齐算法

本文使用的基于字符串相似度的实体对齐算法采用编辑距离计算，编辑距

离[57]是指两个字符间转换的最少编辑次数，编辑包括替换、删除、插入一个字符。一般来说，编辑距离越小，两个串的相似度越大。字符串 S_1，S_2 的编辑距离相似度表示如下：

$$sim_e(s_1,s_2)=1-\frac{n_{op}}{\max(len(s_1),len(s_2))} \tag{5-4}$$

其中，$len(S_1)$ 为字符串 S_1 的长度，n_{op} 为字符串 S_1 变换到 S_2 的最小编辑次数。在计算了实体编辑距离相似度之后，设定一个阈值 t，当相似度高于这个阈值，就认为两个实体需要对齐，否则不相似。

2) 基于 TransE 的实体对齐算法

TransE 的训练过程如算法 2-3 所示，本文使用 TransE 的知识表示算法对职位实体进行向量化，对模型中的参数进行调试，采用随机梯度下降法（Stochastic Gradient Descent）进行参数的训练，使用 SGD 更新参数时，每次从样本中选择一个训练样本以获取梯度。在样本量特别大的情况下，SGD 不用训练完所有的样本，训练次数为 1000 次，最终选择训练效果最好的参数作为模型参数，设置随机梯度下降的学习率 α 的区间为{0.01,0.1,0.5}，间距 λ 的区间为{1,2,4}，单批采样数据 B 的区间为{10,50,100,150,200,500}，嵌入维度 k 的区间为{20,50,100,150,200,50,1000}，最终确定了模型参数的最佳配置为 $\alpha=0.01$，$\lambda=1$，$B=100$，$k=50$。

3) 本文提出的算法（Proposed）

本文提出的基于属性嵌入的对齐算法，使用的是 TransE 的算法思想进行实体对齐，同样通过 1000 次实验对比，优化参数配置为 $\alpha=0.01$，$\lambda=2$，$B=100$，$k=50$。

三种算法都需要设置相似度阈值来过滤对齐实体对，图 5-4、5-5 为相似度阈值对准确度、召回率的影响。从图 5-4、5-5 可以看出，随着 δ 的增加，准确度、召回率都呈现先升高后降低的趋势，说明对齐效果先逐渐变好然后又逐渐变差，在中间某一处取到了最大值。阈值设置过低，会导致相似实体对太多，可能出现错误的对齐实体对；阈值设置过高，会导致对齐实体较少，不能完全清除不同数据源中的冗余数据。

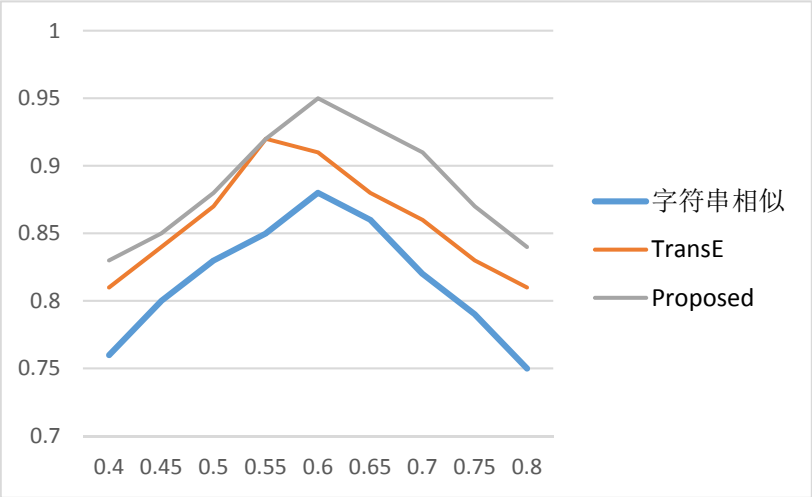


图 5-4 相似性阈值 δ 对准确度的影响

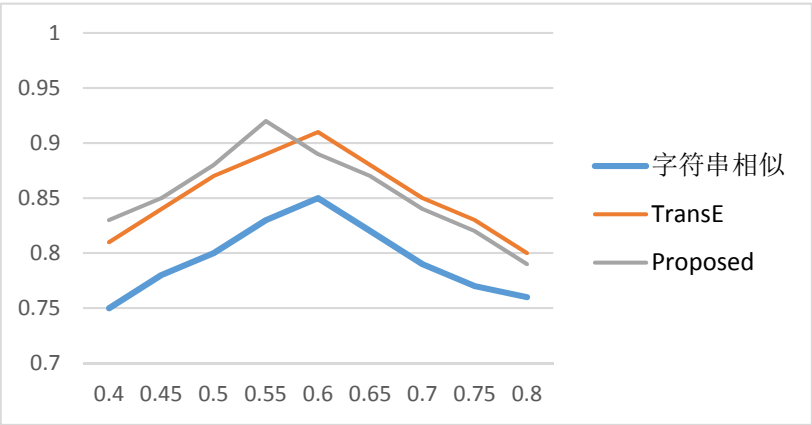


图 5-5 相似性阈值 δ 对召回率的影响

5.3.4 实验结果

通过上一节分析，取各个算法对齐效果最好时的相似度阈值，实验对齐效果如表 5-6 所示。

表 5-6 实体对齐结果

数据集类型	职位数据集			Coral 数据集		
评价指标	Precision	Recall	F1	Precision	Recall	F1
字符串相似度方法	0.88	0.85	0.86	0.86	0.64	0.73
TransE	0.92	0.88	0.90	0.87	0.63	0.73
本文算法	0.95	0.91	0.93	0.89	0.75	0.90

从表中可以看出，仅使用字符串相似度的实体对齐方法准确度、召回率和F1值都比较低，相比与基于知识表示的实体对齐算法，只比较字符串相似度的效果不佳。TransE算法将实体表示成向量，通过计算向量相似度进行实体对齐，由于使用了语义信息，相比基于字符串相似度的算法，在准确度、召回率和F1值上都有明显提升。

本文在TransE算法的基础上，针对求职领域，使用了知识图谱中的属性信息，最终在准确度、召回率和F1值要优于字符串相似度方法和TransE算法。由于引进了属性嵌入，因此，在求职领域中，本文改进的基于属性的实体对齐算法是能够提高知识图谱构建质量的。

5.4 基于知识图谱的职位推荐算法实验验证

5.4.1 实验设计

本文提出的基于知识图谱的推荐算法分为两个部分，第一部分使用了上一节训练得到的职位实体向量表示，计算职位相似度矩阵。之前训练得到的向量表示存储在entity_vector.txt文件中，并对职位进行了编号处理，如图5-6所示。

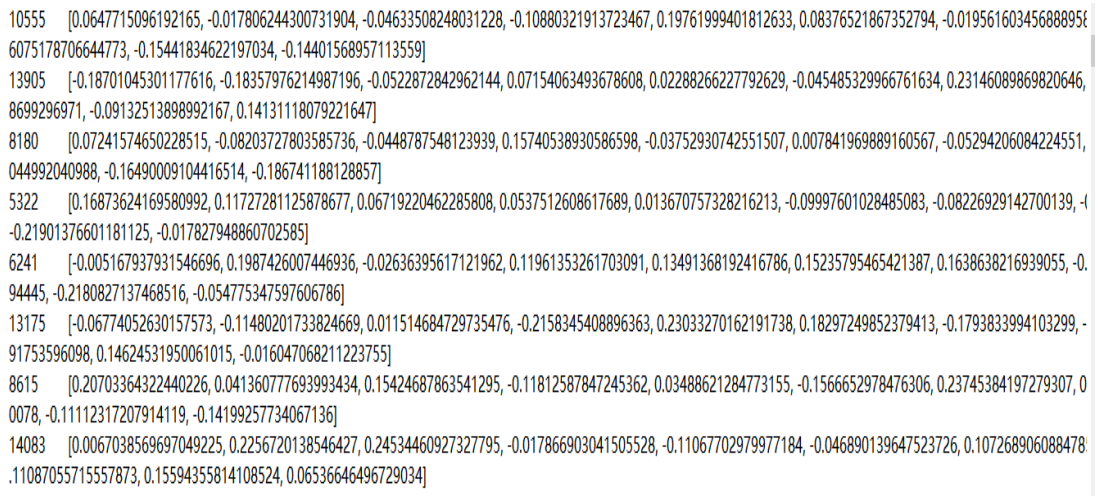


图5-6 训练得到的职位实体向量表示

通过采集的面试职位查找职位的编号，通过编号在entity_vector.txt获取职位的向量表示。

计算所有用户访问职位的相似度并构建相似度矩阵，然后使用矩阵分解用户原始评分矩阵，并计算基于用户行为的职位相似度矩阵。最后使用融合因子a将两个职位相似度矩阵融合，并使用融合的职位相似度矩阵计算预测评分，取

评分高的 N 个项目组成推荐列表。

另一方面，读取三元组的实体、关系、属性的向量表示，计算各个属性的权重，对用户评价过的职位属性向量进行加权求和，得到用户的兴趣表示，获取所有用户的兴趣表示向量，使用余弦相似度计算方法得到用户相似度矩阵，计算预测评分，取评分高的 N 个项目组成推荐列表，最后运用算法 4-3 将两部分的推荐列表融合，生成最后的推荐结果。

本文提出了融合知识图谱和协同过滤的职位推荐算法，通过实验，选取领域设置为 20，设置融合因子 $\alpha = 0.5$，相似度阈值 $\delta = 0.75$，矩阵分解维度为 $K = 10$ 在后续实验中会对实验中的参数进行分析说明。选取了几种基准算法与本文所提出的算法（Proposed）进行对比，讨论实验中参数对实验结果的影响。下面为本文选取的实验对比算法：

(1) 基于用户的协同过滤算法（User-based CF）：该算法主要通过分析用户的行为记录找到用户相似度群体，然后将相似用户群喜欢的项目推荐给目标用户。

(2) 基于项目的协同过滤算法（Item-based CF）：该算法主要通过找到与目标用户兴趣接近的群体，然后将群体喜好的项目进行筛选后推荐给目标用户。

(3) 概率矩阵分解（PMF）[55]：PMF 是一个经典的矩阵分解算法，它对 SVD 模型进行了概率扩展，仅使用用户评分数据进行推荐。

参照对比算法的文献中的参数设置，取推荐效果最优时的参数。参数的设置如表 5-7 所示。

表 5-7 参数设置

算法	参数设置
User-based CF	相似邻域设为 30
Item-based CF	相似邻域设为 30
PMF	$\lambda_U = \lambda_V = 0.001$， $\alpha = 0.005$
Proposed	$\alpha = 0.5$， $\delta = 0.75$， 相似邻域设为 30

5.4.2 评价指标

本文采用准确率(Pre@k)和召回率(Rec@k)作为评价指标，k 表示为用户推荐前 k 个职位。准确率表示正确推荐的职位占实际推荐职位的比例；召回率表

示正确推荐的职位占实际访问职位的比例。准确率和召回率越高，说明推荐算法的性能越好。用 L_u 表示用户实际访问的职位列表，L_r 表示系统推荐的职位列表。准确率和召回率的定义如下：

$$\mathrm{Pre@k} = \frac{|L_u \cap L_r|}{k} \tag{5-5}$$

$$\mathrm{Rec@k} = \frac{|L_u \cap L_r|}{|L_u|} \tag{5-6}$$

5.4.3 参数分析

（1）参数 δ 的影响

相似度阈值 δ 是计算用户偏好相似度时的主要参数。固定参数 $\alpha = 0.5$，$\mathrm{K} = 10$，本文用 Pre@5 值的变化趋势说明参数 δ 对推荐系统的影响。从图 5-7 可以看出，随着 δ 的增加，Pre@5 值呈现先升高后降低的趋势，说明推荐的效果先逐渐变好然后又逐渐变差，在中间某一处取到了最大值，可见相似性阈值 δ 的值设为 0.75 时推荐系统的效果最佳。本文的推荐方法通过阈值确定相似用户。阈值设置过低，会导致相似用户过多，无法确定最佳的相似用户；阈值设置过高，会导致相似用户过少，丢失引入偏好相似性的意义。

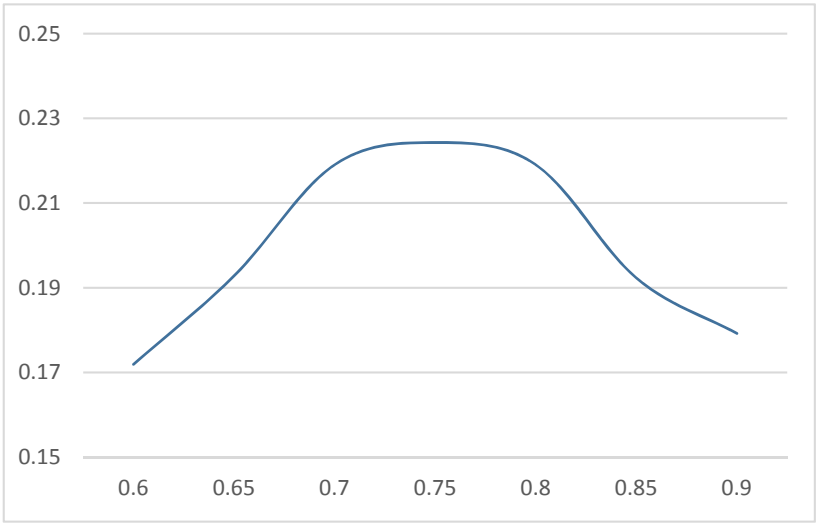


图 5-7 相似性阈值 δ 对 Pre@5 的影响

（2）参数 α 的影响

融合因子 α 是在计算职位相似度矩阵时，融合基于知识图谱的职位相似度

矩阵和基于用户行为的职位相似度矩阵的参数，对推荐结果有中烟的影响。这里设置$\delta=0.75$，$K=10$，通过实验来观察融合因子α对推荐系统的影响，同样，在 Pre@5 值上进行观察，选取了[0.1,1]之间的点进行对比，发现$\alpha=0.5$时推荐系统的性能最高，此时，算法能够最佳地平衡知识图谱职位相似度和协调过滤的职位相似度之间的关系，得到的推荐结果更好。实验结果如图 5-8 所示。

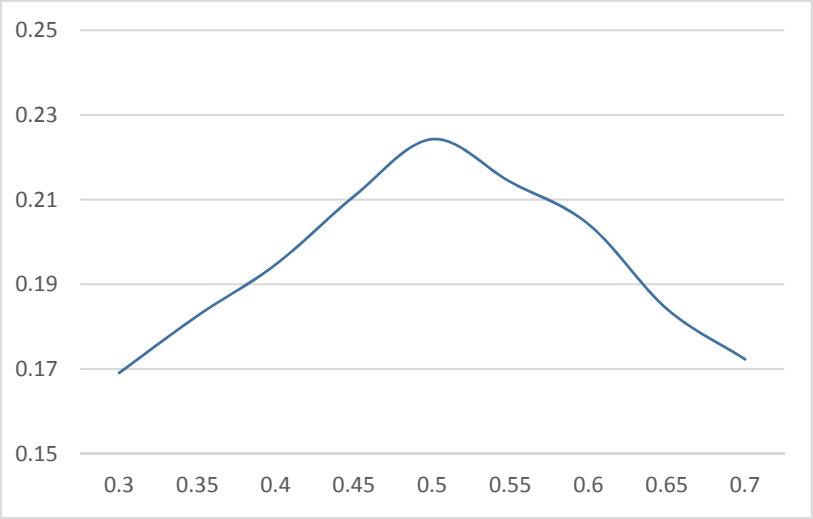


图 5-8 参数α对 Pre@5 的影响

（3）参数 K 的影响

固定设置参数$\delta=0.75$，$\alpha=0.5$，调整潜在特征向量维度 K，观察 Pre@5 值在数据集上的变化情况，实验结果如图 5-9 所示。从图 5-9 可以看出，随着维度 K 的增加，Pre@5 的值呈现先上升后下降的变化，当$K=10$时，Pre@5 的值最大。

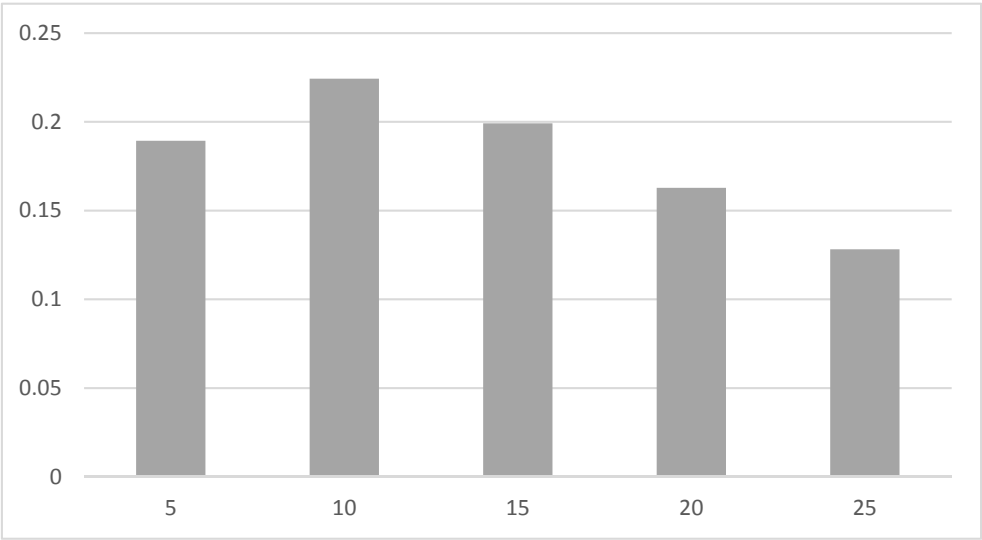


图 5-9 参数 K 对 Pre@5 的影响

5.4.4 实验结果

设置矩阵分解方法的潜在特征维度为 $K=10$，本文算法的相似度阈值为 $\delta=0.75$，知识图谱职位相似度融合因子 $\alpha=0.5$。在数据集上进行 10 次实验，观察推荐职位的个数 k 值对实验结果的影响，所有算法在数据集上的实验结果如图 5-10 和 5-11 所示。

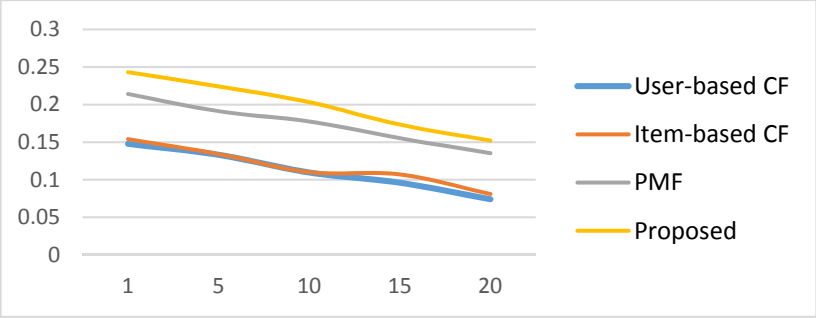


图 5-10 准确率

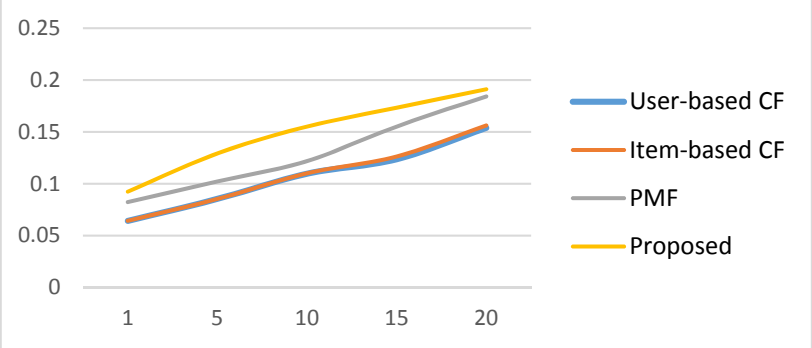


图 5-11 召回率

在图 5-10 和 5-11 中，横坐标表示职位数量 k，纵坐标表示准确率、召回率。从上面两图中可以比较看出，在所有的对比算法中，User-based CF 和 Item-based CF 的结果比基于矩阵分解的 PMF 算法效果差，说明在数据较为稀疏的情况下，矩阵分解模型的性能总体上比基于内存的协同过滤推荐算法好。本文提出的推荐算法相比基于矩阵分解的 PMF 算法，准确率和召回率都有略微提升，这是因为在推荐系统中增加了用户偏好数据和知识图谱的语义信息，而其它方法仅使用隐式反馈数据进行推荐，辅助数据的加入提升了推荐系统的性能。由此可见，本文融合了知识图谱的推荐算法有不错的推荐效果。

本文的算法主要的优势是在推荐过程中引入了知识图谱的属性和实体语义信息，发现用户潜在的求职需求。利用职位知识图谱中存在的职位属性信息和语义信息，建立用户兴趣模型，构建用户偏好相似性矩阵，有助于发现用户之间的关联。同时通过融合知识图谱深层语义得到的职位相似度，使用用户相似度和职位相似度共同计算得到职位推荐列表，提高了个性化职位推荐的性能。

5.5 推荐系统原型设计与展示

本节主要展现基于知识图谱推荐算法实现的推荐系统原型。使用 MongoDB 存储用户信息、职位信息以及用户的评价信息，数据表的设计已经在 5.2 节进行了介绍。本文使用 python 编写了服务端程序。

5.5.1 需求分析

本文设计的职位推荐系统主要是基于用户的历史信息，在职位数据集中为用户推荐可能感兴趣的职位。系统中主要包含普通求职用户与企业用户。普通用户可以在系统中查找各个职位的信息，进行申请、收藏等操作；企业用户可以在系统中发布职位信息。具体的功能模块如图 5-12 所示。

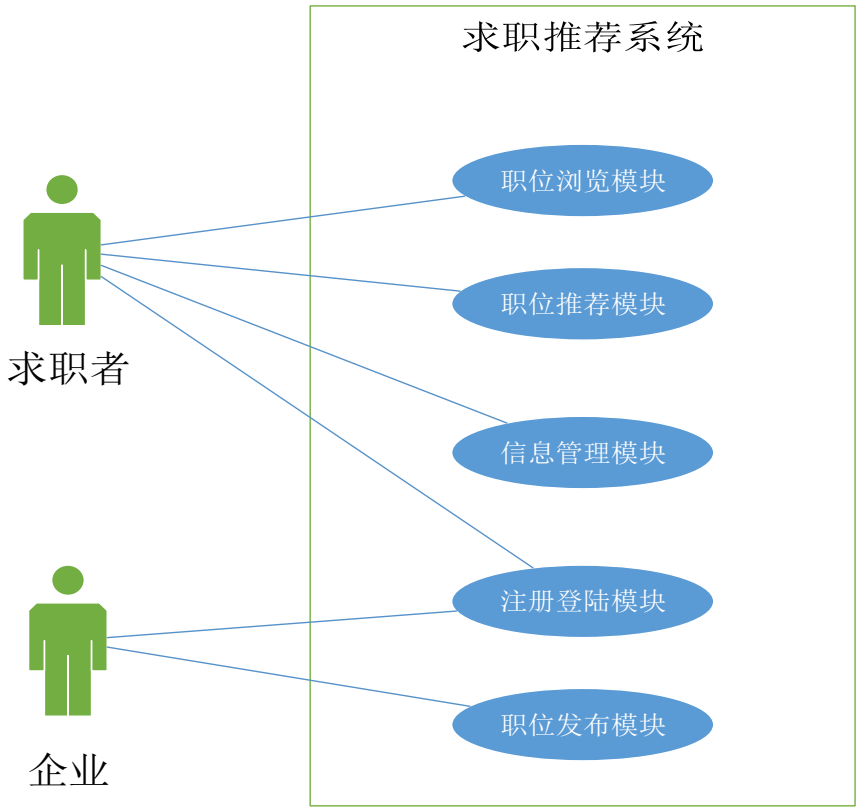


图 5-12 系统用例图

本文设计的原型系统分为五个模块，包括信息管理模块、注册登陆模块、职位浏览模块、职位推荐模块和职位发布模块。注册登陆模块提供了个人用户或企业注册账号并进行登陆；信息管理模块展现了用户信息，用户可以对个人信息进行修改；企业用户可以在职位发布模块中进行职位的发布；职位浏览模块模块提供了职位搜索功能，帮助用户找到目标职位，并展现职位的详细信息，用户可以对职位进行申请、收藏等操作；通过数据库中的职位信息与用户的历史访问信息，职位推荐模块会自动的查询与用户喜好相似度较高的职位，将匹配度高的 10 个职位推荐给用户。

5.5.2 系统实现

本文使用 python 构建了一个轻量级的职位推荐系统原型，考虑到数据传输安全，使用 HTTPS 协议进行数据传输，并使用了 MD5 进行了传输数据加密，保证了系统数据的安全性，系统主要实现了用户注册登陆、职位浏览、职位推

荐、个人信息修改等功能，使用的职位数据为第二章中采集得到的职位信息，存储在 MongoDB 数据库中。

图 5-13 为系统的入口界面，用户可以选择注册或者登陆系统，查看职位信息，对职位进行评价和申请。登陆分为求职用户和企业用户。企业用户登陆系统后，可以看到自己发布的职位信息，并发布新的职位信息。

图 5-13 推荐系统登陆界面

企业用户登陆后如图 5-14 所示，企业可以查看企业信息，进行职位发布并查看已发布职位。发布职位时要填写职位名称、薪资、工作地点、工作经验、学历要求和具体的职位描述。

图 5-14 企业用户主界面

求职用户登录系统后进入推荐系统主界面。主界面由搜索框、搜索条件和搜索结果组成，搜索条件包括工作地点、行业、薪资、公司类型四部分，用户

通过选择搜索条件，可以对搜索结果进行筛选。图 5-15 中通过 java 开发关键字进行搜索，设置搜索的行业为互联网，薪资在 10k 以上，搜索结果展现在搜索框下的列表中。以第一条结果为例，该职位名称为 java 研发工程师，工作地点在武汉光谷，薪资为 10k-20k，要求求职者工作经验为 3-5 年、学历至少本科，发布职位的公司是盛趣时代。

图 5-15 推荐系统搜索界面

通过点击一条搜索结果，可以进入职位的详细界面，如图 5-16 所示，职位详细界面包含了职位的基本信息、职位要求、职位描述、工作地点以及其他用户的评价信息。用户在职位详细信息界面可对职位进行申请、收藏、评价等操作。

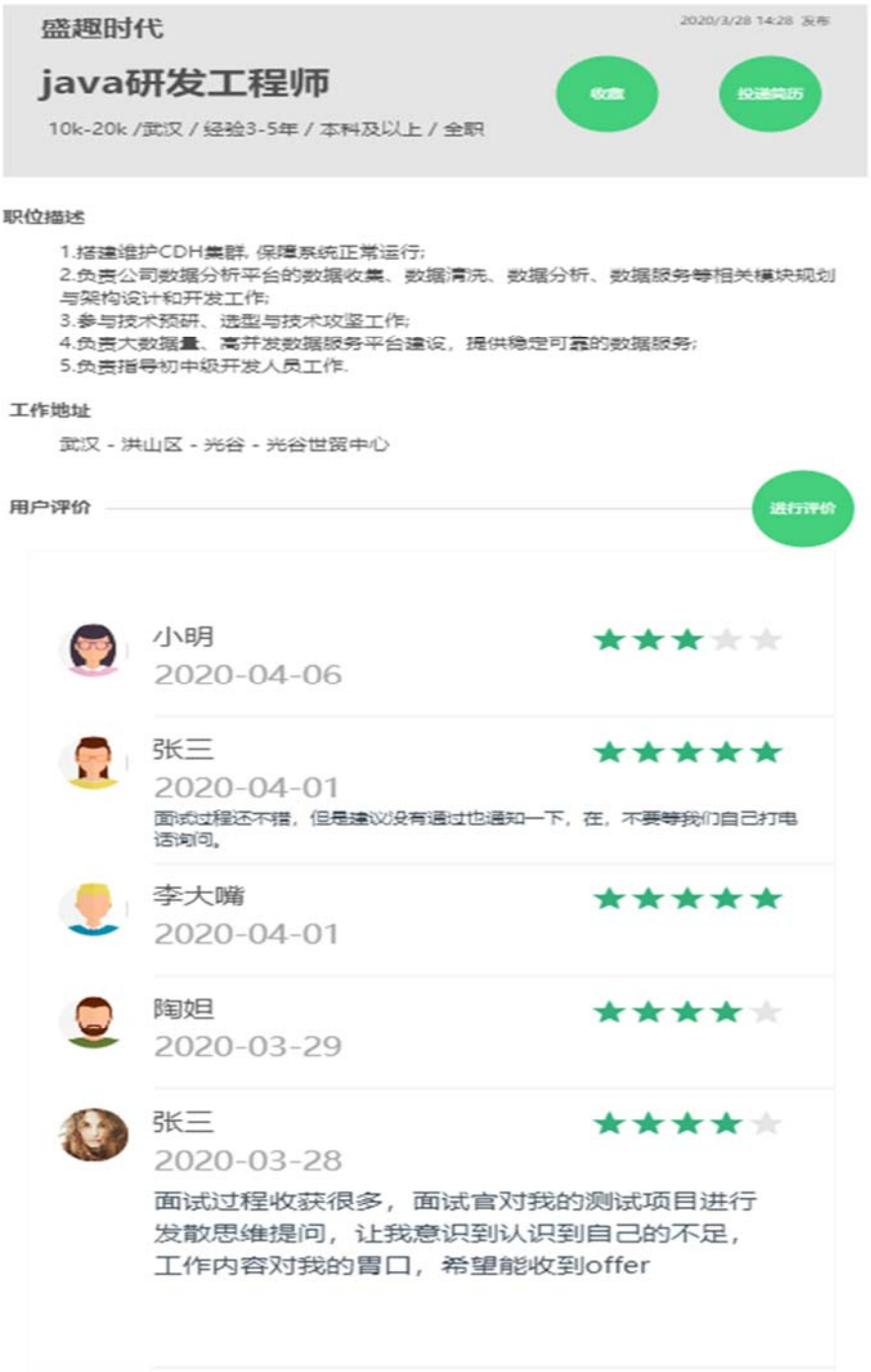


图 5-16 职位详细信息界面

点击图 5-16 系统主界面上的个人中心按钮，进入用户个人中心。本文以数据库中的一个用户小张为例，描述推荐系统的推荐过程。表 5-8 为用户小张的个人信息。

表 5-8 用户个人实例

字段	数据类型	数据值
Username	String	小张
Sex	String	男
Age	Int	24
Education	String	本科
Experience	Int	2

个人信息界面包括用户信息、职位评价记录、用户申请记录、推荐职位以及跳转职位搜索五部分。用户信息展现了用户的个人基本信息，用户可在该界面进行信息编辑修改。图 5-17 为小张登陆后的用户信息界面。图 5-18 为小张的职位评价记录，可以看到小张对 java 开发工程师、Web 开发工程师等职位进行了评价，其中职位信息只显示了职位名称，通过点击职位名称可跳转到职位详细信息界面。用户的评价信息包括了用户名、职位名称、评分（1-5）、评价内容、评价时间等。

图 5-17 个人信息界面

图 5-18 用户职位评价界面

通过读取小张的职位评价，使用本文提出职位推荐算法在服务端进行计算，将得到的推荐列表展现在用户推荐界面中，如图 5-19 所示。图 5-19 包括推荐职位的基本信息、预测的评分值，评分值通过星号展现，值域为 1-5，评分愈高表示用户更可能喜欢该职位。

图 5-19 职位推荐界面

5.6 本章小结

本章在推荐算法实验中，介绍了实验的数据集来源并进行了数据预处理，采用准确率和召回率两个指标评价算法的性能。然后，通过对比实验说明了本文所提出的推荐算法的优势，并分析了各个参数对推荐效果的影响，表明本文提出算法在推荐效果上相比于矩阵分解推荐算法和传统协同过滤算法有一定的提升，推荐的召回率和准确率更高。最后实现了一个推荐系统原型。

第 6 章 总结

6.1 主要工作总结

在职位推荐中，常用的方法是根据用户历史信息或简历信息进行推荐，推荐算法多采用基于用户的协同过滤算法，往往会存在冷启动和数据稀疏问题，为了解决这些问题，大部分的研究多是引用社会关系、用户或物品属性和上下文信息来作为补充信息辅助推荐系统进行精准推荐。本文针对这个问题，从知识图谱中将职位属性信息引入到了推荐系统中，通过知识图谱丰富的语义信息来提高推荐效果。具体的研究工作有以下几个方面：

（1）构建了职位知识图谱。分析了求职领域的信息，构建了求职领域的本体；从各大网站爬取了职位信息，在本体的约束下，通过处理后提取了职位信息，并使用了本文提出的实体对齐算法进行了实体融合，构建了职位知识图谱用于职位推荐。最后，将职位知识存储在了 Neo4j 图数据库中，可视化的展示出来。

（2）设计了基于知识图谱中实体属性的用户兴趣模型。通过之前的知识图谱向量学习，分析了职位各个属性的影响，计算了各个属性的权值，通过加权和计算得到了用户属性偏好向量。

（3）提出了一种改进的基于知识图谱的联合推荐算法。针对传统协同过滤的问题，通过基于用户行为和基于知识图谱的职位相似度，得到了推荐列表。然后使用了用户对职位属性的偏好信息，通过用户相似度矩阵得到了另一个推荐列表，最后将两个推荐列表融合得到最后的推荐结果。在协同过滤中，引入了知识图谱的语义信息和属性信息，相比传统协同过滤推荐算法提高了推荐的准确性和多样性，最后通过提出的推荐算法构建了一个职位推荐系统原型。

6.2 展望

本文首先提出了一种基于属性嵌入的知识图谱间实体对齐方法，利用了知识图谱中大量的属性三元组信息，使其帮助不同 KG 中的实体映射到同一空间中，提高了对齐效果，并构建了职位知识图谱用于推荐，提出了一种融合知识图谱和用户隐式反馈的推荐方法，提高了推荐的准确率和召回率。但是，本文的研究也有诸多不足，需要进一步研究。具体有以下几个方面：

（1）可以进一步提高知识图谱构建质量。它的质量关系着推荐系统的效果。在实体对齐上，需要尝试更加优化的方法，如深度学习的方法，构建知识更加完备的职位知识图谱。

（2）在实际的推荐问题中，用户对职位兴趣会随着时间的推进而改变，在求职领域，用户之前的记录可以失效，因此在构建用户兴趣模型时，因考虑到时间因素。本文的研究没有考虑时间因素对用户兴趣的影响，下一步的研究工作是将时间因素融入到矩阵分解模型中，提高推荐系统的时效性。

（3）在知识图谱和推荐模型相结合时，采用的是先从知识图谱中学习实体的向量表示，然后将实体向量融入到推荐系统的方法，这种依次学习方法可能会产生级联误差，降低推荐的性能，下一步的研究工作是将知识图谱的表示模型和推荐模型进行联合学习，降低级联误差的影响。

致 谢

时光荏苒，光阴如梭，一转眼三年的研究生生涯就结束了。在三年的学习和生活中，我收获了很多，不仅仅是知识的积累，在为人处事方面也受益匪浅，这对我未来的工作和学习都是一比宝贵的财富。在论文完成之际，我要向给过我帮助的人致以最衷心感谢。

首先，我要感谢我的导师刘洪星教授。从入学开始，刘老师就教导我们要有计划的学习和工作，指导我制定了培养计划和研究方向，并在以后的学习中不断给予我帮助，在生活上也无微不至，对我的学习和生活都有很大的帮助。在论文的选题和撰写过程中，刘老师一直悉心的指导我，提出了很多宝贵的意见，让我在写作过程中收益甚多。刘老师严谨的治学态度，精益求精的工作精神和科学的工作方法，深深地影响了我。在此，谨向刘老师表达我最衷心的感谢。

同时，感谢科研项目组的所有老师和同学们。杨青教授、姚寒冰副教授、李勇华副教授和吴业福副教授在项目上的专业技术和经验，给了我很大的帮助，他们在我确定研究方向和修改论文的过程中，都给出了宝贵的意见。感谢实验室的所有同学们，他们在我的实验、研究遇到问题时，积极与我探讨，帮助我解决了很多难题。

然后我也要感谢学姐学长们的帮助，在我的论文研究上提出建议，毫不吝啬的分享自己的研究成果，对我的论文帮助很大。最后感谢我的家人，是你们的关心和鼓励给了我前进的动力。

最后，感谢评审老师百忙之中抽出时间审阅我的论文，感谢你们提出的宝贵意见！

参考文献

[1] 常亮,张伟涛,古天龙等.知识图谱的推荐系统综述[J].智能系统学报.2019(02):207-216

[2] Singhal, Amit. Introducing the Knowledge Graph: Things, Not Strings[EB/OL]. [2018-05-17].https://google-blog.blogspot.co.uk/2012/05/introducing-knowledge-graph-things-not.html

[3] 徐增林,盛泳潘,贺丽荣等.知识图谱技术综述[J].电子科技大学学报.20 16,45(04):589-606

[4] 孙镇,王惠临.命名实体识别研究进展综述[J].现代图书情报技术.2010(6):42-47.

[5] 杨博,蔡东风,杨华.开放式信息抽取研究进展[J].中文信息学报.2014(4):1-11

[6] DOMINGOS P, WEBB A. A tractable first-order probabilistic logic[C]. Proceedings of the 26th AAAI Conference on Artificial Intelligence. San Francisco, CA: AAAI, 2012: 1902-1909

[7] Bordes A, Usunier N, Garcia-Duran A, et al.Translating embeddings for modeling multirelational data [C]. Proc of NIPS. Cambridge, MA:MIT Press, 2013:2787-2795

[8] Wang Zhen, Zhang Jianwen, Feng Jianlin, et al. Knowledge graph embedding by translating on hyperplanes[C]. Proc of AAAI Conference on Artificial Intelligence.Menlo Park. 2014: 1112-1119

[9] Lin Yankai, Liu Zhiyuan, Sun Maosong, et al. Learning entity and relation embeddings for knowledge graph completion[C]. Proc of AAAI Conference on Artificial Intelligence. Menlo Park. 2015:2181-2187

[10] Ji Guoliang, He Shizhu, Xu Liheng, et al. Knowledge graph embedding via dynamic mapping matrix[C]. Meeting of the Association for Computational Linguistics & the International Joint Conference on Natural Language Processing. 2015:687-696

[11] 庄严,李国良,冯建华.知识库对齐技术综述[J].计算机研究与发展.2016(1):165-192

[12] Christen P. Automatic training example selection for scalable unsupervised record linkage [C]. Proceedings of the 12th Pacific-Asia conference on Advances in knowledge discovery and data mining. Berlin: Springer.2008(5012).511-518

[13] Juanzi L, Zhichun W, Xiao Z, et al. Large scale instance matching via multiple indexes and candidate selection[J]. Knowledge-Based Systems.2013, 50(complete): 112-120

[14] Lacoste-Julien S, Palla K, Davies A, et al. Sigma:simplegreedy matching for aligning large knowledge bases[C]. Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.Chicago, 2013:572-580

[15] 朱继召,乔建忠,林树宽.表示学习知识图谱的实体对齐算法[J].东北大学学报(自然科学版).2018.39 (11): 1535-1539

[16] Bayu Distiawan T, Jianzhong Q, et al. Entity Alignment between Knowledge Graphs Using Attribute Embeddings[C]. Proceedings of AAAI Conference on Artificial Intelligence.2019

[17] Wong W, Liu Wei, Bennamoun M. Ontology learning from text: a look back and into the future[J]. ACM Computing Surveys. 2011, 44(4):1-36.

[18] 刘峤,李杨,段宏,刘瑶.知识图谱构建技术综述[J].计算机研究与发展.2016,53(3):582-600

[19] Fader A, Soderland S, Etzioni O. Identifying relations for open information extraction[C]. Proc of the Conf on Empirical Methods in Natual Language Processing. Stroudsburg. 2011:1535-1545

[20] Lehmann, Jens, et al. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia[J]. Semantic Web, 2015: 167-195.

[21] Hoffart J, Suchanek F M, Berberich K, et al. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia[J]. Artificial Intelligence, 2013, 194:28-61.

[22] Santos F A, Nascimento F B D, Santos M S, et al. Training Neural Tensor Networks with the Never Ending Language Learner[J]. Information Technology-New Generations. Springer, Cham, 2018: 19-23.

[23] 曹红姣.基于情境感知的大学生就业推荐系统的设计与实现[D].华中师范大学.2014

[24] 潘理虎,张佳宇,张英俊等.煤矿领域知识图谱构建[J]. 计算机应用与技术,2019(8): 851-862.

[25] 杨玉基,许斌,胡家威等. 一种准确而高效的领域知识图谱构建方法[J].软件学报，2018, 29(10): 39-55.

[26] 侯丽娟.基于行为分析的个性化职位推荐[D].河北师范大学.2014

[27] 王超.基于社交关系的职位推荐系统的架构与实现[D].华中科技大学.2015

[28] 汪澎洋.个性化求职信息推荐系统的研究与设计[D].北京邮电大学.2017

[29] 陈朝冲.基于高校毕业生与招聘企业双选的推荐系统[D].西南工业大学.2017

[30] Miao J,YiXie F, ming H, et al. User click prediction for personalized job recommendation[J]. World Wide Web. 2019. 1(22):325-345

[31] Hongwei W, Fuzheng Z, Min H, Xing Xie, et al. Shine: Signed heterogeneous information network embedding for sentiment link prediction[C]. In Procee- -dings of the Eleventh ACM International Conference on Web Search and Data Mining. 2018: 592-600

[32] Hongwei W, Fuzheng Z, Xing X, et al. DKN: Deep Knowledge Aware Network for News Recommendation[C]. In Proceedings of the 2018 World Wide Web Conference on World Wide Web. International World Wide Web Conferences Steering Committee. 2018: 1835-1844

[33] Fuzheng Z, Nicholas Jing Y, Defu L, et al, and Wei-Ying Ma. Collaborative knowledge base embedding for recommender systems[C]. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016: 353- 362

[34] Quan W, Zhendong M, Bin W, et al. Knowledge graph embedding: A survey of approaches and applications[C].IEEE Transactions on Knowledge and Data Engineering. 2017: 2724-2743.

[35] Xiao Y, Xiang R, Yizhou S,et al. Personalized entity recommendation:A heterogeneous information network approach [C]. In Proceedings of the 7th ACM International Conference on Web Search and Data Mining. 2014: 283-292

[36] Gruber T R. A translation approach to portable ontology specifications[J]. Knowledge Acquisition, 1993, 5(2): 199-220.

[37] Hongwei W, Fuzheng Z, JiaLin W, et al. RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems[C].27th ACM International Conference on Information and Knowledge Management. 2018: 417-426

[38] BENJAMIN H, CONOR H. Using Linked Data to Build Open Collaborative Recommender Systems[C]//16th International Conference on Electronic Commerce and Web Technologies. 2010. (239):30-41

[39] Noia T D, Mirizzi R, Ostuni V C, et al. Linked open data to support content-based recommender systems[C]// International Conference on Semantic Systems. Graz, Austria, ACM, 2012: 1-8.

[40] Oramas S, Ostuni V C, Noia T D, et al. Sound and Music Recommendation with Knowledge

Graphs[J]. Acm Transactions on Intelligent Systems & Technology, 2016, 8(2):21

[41]王一鸣.基于知识图谱的推荐技术研究及应用[D].电子科技大学.2018

[42]贾中浩,古天龙,宾辰忠等.旅游知识图谱特征学习的景点推荐[J].智能系统学报.2019.3(14):430-437

[43]路小瑞.基于 Hadoop 平台的职位推荐系统的设计与实现[D].上海交通大学.2014

[44]Shalaby W, AlAila B, Korayem M, et al. Help Me Find a Job: A Graph-based Approach for Job Recommendation at Scale[C]. IEEE Internat- -ional Conference on Big Data. 2017: 1544-1554

[45]Wenxing H, Siting Z, Huan W.Dynamic User Profile-Based Job Recommender System [C]. 8th International Conference on Computer Science and Education. 2013: 26-28

[46]漆桂林,高桓,吴天星.知识图谱研究进展[J].东南大学计算机科学与工程学院.情报工程.2017(01):4-25

[47]苏佳林,王元卓,靳小龙等.融合语义和结构信息的知识图谱实体对齐[J].山西大学学报(自然科学版).2019(01):23-30

[48]朱文跃,刘炜,刘宗田.基于事件本体的新闻个性化推荐[J].计算机工程.2019.45(6):262-272

[49]刘超然.面向招聘领域的互惠推荐算法研究[D].燕山大学.2012

[50] 谷楠楠,冯筠,孙霞,赵妍等;中文简历自动解析及推荐算法[J].计算机工程与应用.2017.53(18)：141-148

[51]李成铭.基于文本特征提取技术的在线人职匹配研究及应用[D].电子科技大学.2017

[52]Guan S, Jin X, Jia Y, et al. Self-Learning and Embedding Based Entity Alignment[C]. //Knowledge and Information Systems. 2018(24). 1-26

[53]Nickel M. Rosasco L. and Poggio T. Holo-graphic embeddings of knowledge graphs[J]. In Proceedings of AAAI Conference on Artificial Intelligence. 2016. 1955–1961

[54]郝雅娴,孙艳蕊.K 近邻矩阵分解推荐系统算法[I].小型微型计算机系统.2018.39(4):755-758.

[55]Lin Y, Liu Z, Sun M, et al. Learning Entity and Relation Embeddings for Knowledge Graph Completion[C]. Twenty-ninth Aaai Conference on Artificial Intelligence, AAAI Press, 2015, 15: 2181-2187.

[56]Wang Z, Zhang J, Feng J, et al. Knowledge Graph Embedding by Translating on Hyperpla-

-nes[C]. Twenty-eighth Aaai Conference on Artificial Intelligence, AAAI Press, 2014, 14: 1112-1119.

[57] Singhal, Amit. Introducing the Knowledge Graph: Things, Not Strings[EB/OL]. [2018-05-17]. https://googleblog.blogspot.co.uk/2012/05/introducing-knowledge-graph-things-not.html.

[58] Santos F A, Nascimento F B D, Santos M S, et al. Training Neural Tensor Networks with the Never Ending Language Learner[J]. Information Technology-New Generations. Springer, Cham, 2018: 19-23.

[59] 刘济源.旅游知识图谱的构建及应用研究[D].浙江大学.2019

[60] Webber J, Robinson I. A programmatic introduction to neo4j[M]. Addison-Wesley Professio-al, 2018.

[61] 姜华,韩安琪,王美佳等.基于改进编辑距离的字符串相似度求解算法[J].计算机工程，2014, 40(1): 222-227.

[62] Jamali M, Ester M. A matrix factorization technique with trust propagation for recommendation in social networks[C]. Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September, ACM, 2010: 135-142.

[63] Braunhofer M, Elahi M, Ricci F. User personality and the new user problem in a context-aware point of interest recommender system[M]. Information and Communication Technologies in Tourism 2015, Springer, Cham, 2015: 537-549.

[64] Xiao Y, Xiang R, Yizhou S, etal, Urvashi Khandel wal, Brandon Norick, and Jiawei Han.Personalized entity recommendation: A heterogeneous information network approach [C]. In Proceedings of the 7th ACM International Conference on Web Search and Data Mining. 014: 283-29

[65] Huan Z, Quanming Y, Jianda L, etal. Meta-graph based recommendation fusion over heterogeneous information networks[C]. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2017: 635-644.

[66] Hongwei W, Fuzheng Z, etal. DKN: Deep Knowledge Aware Network for News Recommendation[C]. In Proceedings of the 2018 World Wide Web Conference on World Wide Web. International World Wide Web Conferences Steering Committee. 2018: 1835-1844

[67] Fuzheng Z, Nicholas Jing Y, Defu L, etal. Collaborative knowledge base embedding for recommender systems[C].s In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016: 353- 362

[68] 百度百科.百度知心[EB/OL].http://baike.baidu.com/view/10972128.html.2020-2

[69] 搜狗百科.知立方[EB/OL].http://baike.sogou.com/h66616234.html.2020-2

[70] 复旦大学.知识工厂[EB/OL].http://kw.fudan.edu.cn/.2020-2

[71] Bordes A, Usunier N, Garcia-Duran A, et al. Translating embeddings for modeling multi-relational data[C]. Advances in neural information processing systems, 2013:2787-2795.

[72] Information Extraction: A multidisciplinary approach to an emerging Information Techno logy[M]. Heidelberg: Springer. 1997

[73] 宋晴晴.基于本体的网络招聘个性化推荐系统用户建模研究[D].南京航空航天大学.2009

[74] 美国兰德公司(RAND).[EB/OL].http: //www.rand.org.2010

[75] 冯子威.用户兴趣建模的研究[D].哈尔滨工业大学.2010

[76] Guan S, Jin X, Jia Y, etal. Self-Learning and Embedding Based EntityAlignment, Big Knowledge(ICBK)[C]. IEEE International Conferenceon.2017.33-40

[77] Kethavarapu K. Concept based dynamic ontology creation for job recommendation system [J]. Procedia computer science, 2016, 85: 915 – 921.

攻读学位期间获得与学位论文相关的科研成果目录

发表论文：

[1] ChuanLong Li,HongXing Liu, FangRong Zhang. Research on Entity Recognition Method in Knowledge Base Question Answering[C]. International Symposium on Distributed Computing and Applications To Business, Engineering and Science. (本文作者为第一作者)

科研竞赛：

[1] 第十届蓝桥杯全国软件和信息技术专业人才大赛湖北赛区 JAVA 软件开发大学三等奖 （2019.3.1）

参与的主要科研项目：

[1] 重庆江津港件散货码头生产业务管理系统，企业委托，2017.9-2020.1