



**基于知识图谱的电影推荐系统**

**Movie Recommender Systems Based on Knowledge Graph**

**Major:** Computer Science

**Student′ Name (In Pin-Yin):** LIN, Yunqi

**Student′ Name (in Chinese):** 林韫奇

**Student′ ID No.**： 18722016

**Supervisor′ Name**： YUAN, Jidong

7 May 2022

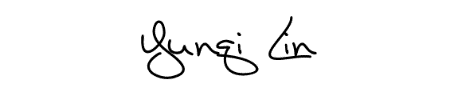
**Declaration**

I certify that the material contained in this dissertation is my own work and does not contain unreferenced or unacknowledged material.

Regarding the electronically submitted version of this submitted work, I consent to the work being stored electronically and copied for assessment purposes, including the School’s use of plagiarism detection systems in order to check the integrity of assessed work.

I agree to my dissertation being placed in the public domain, with my name explicitly included as the author of the work.

Date: 29 April 2022



Signed:

学士论文版权使用授权书

本学士论文作者完全了解北京交通大学有关保留、使用学士论文的规定。特授权北京交通大学可以将学士论文的全部或部分内容编入有关数据库进行检索，提供阅览服务，并采用影印、缩印或扫描等复制手段保存、汇编以供查阅和借阅。



学位论文作者签名： 指导教师签名：

签字日期：2022年04月15日 签字日期： 年 月 日

详细中文摘要

详细中文摘要应将论文的内容要点简短明了地表达出来，约800字左右。字体与字号的规定是，中文采用普通小四号宋体，英文采用普通12 points的Times New Roman字体。内容应包括工作目的、研究方法、成果和结论。要突出本论文的创新点，语言力求精炼。为了便于文献检索，应在本页下方另起一行注明论文的关键词（3-5个）。特别注意：详细中文摘要比英文摘要需要更详尽，详细中文摘要的篇幅一般是英文摘要的一倍。

**关键词：**

**Abstract**

英文摘要应将论文的内容要点简短明了地表达出来，约400字左右。采用普通12 points的Times New Roman字体。内容应包括工作目的、研究方法、成果和结论。要突出本论文的创新点，语言力求精炼。为了便于文献检索，应在本页下方另起一行注明论文的关键词（3-5个）。特别注意：详细中文摘要比英文摘要需要更详尽，详细中文摘要的篇幅一般是英文摘要的一倍，英文摘要不能与中文摘要的内容完全一致。

**KEYWORDS：**

**Contents**

[中文摘要 III](#_Toc414268225)

[ABSTRACT IV](#_Toc414268226)

[目 录 V](#_Toc414268227)

[1 引言 错误!未定义书签。](#_Toc414268228)

[2 （小四、黑体、左对齐、空1字符） 错误!未定义书签。](#_Toc414268229)

[2.1 （小四、宋体、首行缩进1字符） **错误!未定义书签。**](#_Toc414268230)

[2.1.1 （小四、宋体、首行缩进2字符） **错误!未定义书签。**](#_Toc414268231)

[3 错误!未定义书签。](#_Toc414268232)

[3.1 **错误!未定义书签。**](#_Toc414268233)

[3.1.1 **错误!未定义书签。**](#_Toc414268234)

[4 错误!未定义书签。](#_Toc414268235)

[4.1 **错误!未定义书签。**](#_Toc414268236)

[4.1.1 **错误!未定义书签。**](#_Toc414268237)

[5 结论 错误!未定义书签。](#_Toc414268238)

[参考文献 41](#_Toc414268239)

[致 谢 44](#_Toc414268240)

[附 录 44](#_Toc414268240)

【针对威海校区计科专业说明：】

无论是在正文之中，还是在目录页之上，都只能有3级目录。学生可以采用“更新域”方式更新目录，但必须进行格式修正与控制。并且，保持上面示例的行距。这3级目录之中的标题格式规定如下：

(1) 1级标题：中文采用黑体，英文采用Times New Roman字体；都是12 points的大小；“左对齐”且左侧空1字符。

(2) 2级标题：本级之中一般没有中文，但是若有中文，则采用宋体；英文都是采用Times New Roman字体。字号也都是12 points的大小；左侧再缩进1字符。

(3) 3级标题：本级之中一般没有中文，但是若有中文，则采用宋体；英文都是采用Times New Roman字体。字号也都是12 points的大小；左侧再缩进2字符。

注意需要2行的标题，不仅设置行距为单倍行距，且每行都是一个“段落”。

1 Introduction

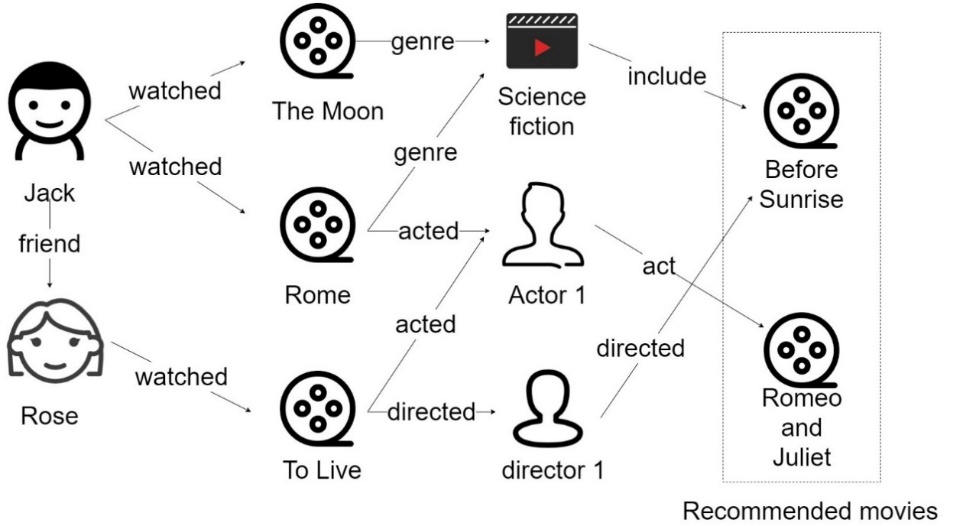
This study aims to build a movie recommender system with the knowledge graph used as auxiliary information of collaborative filtering algorithm by using the film data information obtained from the Internet to enhance the user experience. The chapter is divided into two parts, which are research background and the significance, and the organizations.

1.1 Research Background and Significance

Because the Internet has the features such as openness of information transmission and two-way communication, the amount of information rapidly expands and increases by exponents, the problem of information overload worsens in the course of time. The key to solving these problems is to transform the Internet from passively accepting requests from viewers to proactively sensing information requests from viewer (Resnick et al, 1997), which is how recommendation techniques emerge and become the core technique concerned by many scholars and internet users.

There are many categories of recommender system. Collaborative filtering algorithm is one that used widely in the domain of recommendation (Ma et al, 2009). It mainly relies on the wisdom of the crowd and relies only to a low extent to professional knowledge, which is why it's popular. Moreover, after continuous research and improvement of researchers，it can achieve good performance. Collaborative filtering algorithm is a process in which the system predicts the items that users may like according to their historical behavior information. The more the historical trial the user left on a certain kind of items, the higher the probability of this kind of items being recommended to the user by the recommender system would be.

However, due to the fact that in reality, the amount of data on the Internet is too large, users tend not to visit the types and quantities of data sufficiently, which results in sparse traces left by users on the network. Also, because the problem of cold start, the effect of traditional recommendation system is often not ideal in the start-up stage, and the performance of recommendation algorithm is not satisfactory due to the sparse historical data of users. In recent times, Knowledge-Graph based recommendation system (KGRS) has aroused wide interest among researchers (Guo et al, 2020). It mainly integrates knowledge graph as auxiliary information into recommendation systems, which brings to two advantages. The first advantage is to raise the accuracy of recommendation system. The knowledge graph can be used to represent relations between entities. The item and its attribute information can be mapped into knowledge graph to comprehend relationship between items. In addition, user and auxiliary information of user can be integrated into knowledge graph to capture the relationship between user and item as well as preferences of user more accurately.



**Figure 1** A KG-based recommender system example

Figure 1 shows a KG-based recommender system, which contains entity points such as movies, users, actors, directors and genres. Through this knowledge graph, the system recommends to Jack two movies: *Before Sunrise* and *Romeo and Juliet*. The figure indicates that there are different potential relationships between movies and users, which helps to increase the accuracy of recommendation. Another advantage of KG-based recommendation is that the recommended results are interpretable. In figure1, obeying relationship sequences in the knowledge graph shows the reason why these two movies are recommended to Jack. For example, the reason why *Before Sunrise* is recommended is that *Before Sunrise* shares the same genre with *The Moon*, of which Jack has watched before.

1.2 Research Objectives

Although the traditional collaborative filtering algorithm can recommend information that users are interested in from the point of view of users and items, its recommendation effect is not ideal due to the data sparsity. The amount of user and types of items are large, but users’ rating on item is few, most items have no rating records, thus the accuracy of similarity is not enough. Due to the fact that knowledge graph contains abundant information about user and item, these features can provide more intuitive and targeted explanations for the generation of recommended items. In addition, the knowledge graph diverges different types of relationships and historical records, which improves the diversity and interpretability of recommendation results. Therefore, in this paper, knowledge graph is used as the auxiliary information of items in collaborative filtering algorithm to supplement the semantic information of items and a recommender system with higher prediction ability is developed.

1.3 Organizations

2 Related Work

This section discusses the related work on recommender system, knowledge graph and knowledge graph-based recommender system.

Conventional recommender systems can be divided into collaborative filtering recommendation, content-based recommendation and hybrid recommendation. Sarwar et al (2001) put forward an item-based prediction algorithm, which constructed pre-calculation model of item similarity and enhanced online scalability of recommended system changes. Liu et al (2015) proposed an association mining technique to calculate the similarity among cited papers used for collaborative filtering from the context of papers. Content-based recommendation can well solve the problem of sparse user behavior data and cold start of new users. By using vector space model, linear classification, linear regression and other methods to model user's interest characteristics and item characteristics, users can be recommended items similar to the content they are interested in.

Knowledge graph is a concept put forward by Google in 2012. It’s originally used to enhance query performance. Knowledge graphs are usually represented as triples and can be used to describe entities, concepts, and associations in the physical world. Knowledge graph aims to bring the computer's language of description closer to that of human natural language. Knowledge graph can be divided into domain knowledge graph and open domain knowledge graph according to different application scope (Yu an et al, 2020). Domain knowledge graph covers only a small amount of knowledge categories. It mainly refines knowledge in a certain field, expands the depth of knowledge and provides users with professional knowledge. The open domain knowledge graph covers a wide range of knowledge, but it is not in-depth. It mainly expands the breadth of knowledge and provides users with common knowledge. At present, the research of knowledge graph is in full swing, and many fields are trying to use knowledge graph as auxiliary information to further study the refinement problems of their fields.

Knowledge graph can obtain rich semantic information, which can make up for the data sparsity problem of recommendation system to some extent. Li Hao et al (2020) proposed an algorithm combining knowledge graph and collaborative filtering. The knowledge graph and cyclic neural network are fused to form cyclic knowledge graph, and semantic modeling is carried out through entities, so that entities and relations can be embedded into low-dimensional space through learning. In addition, some additional auxiliary information is introduced as input to enrich the association between entities and improve the performance of the recommender system.

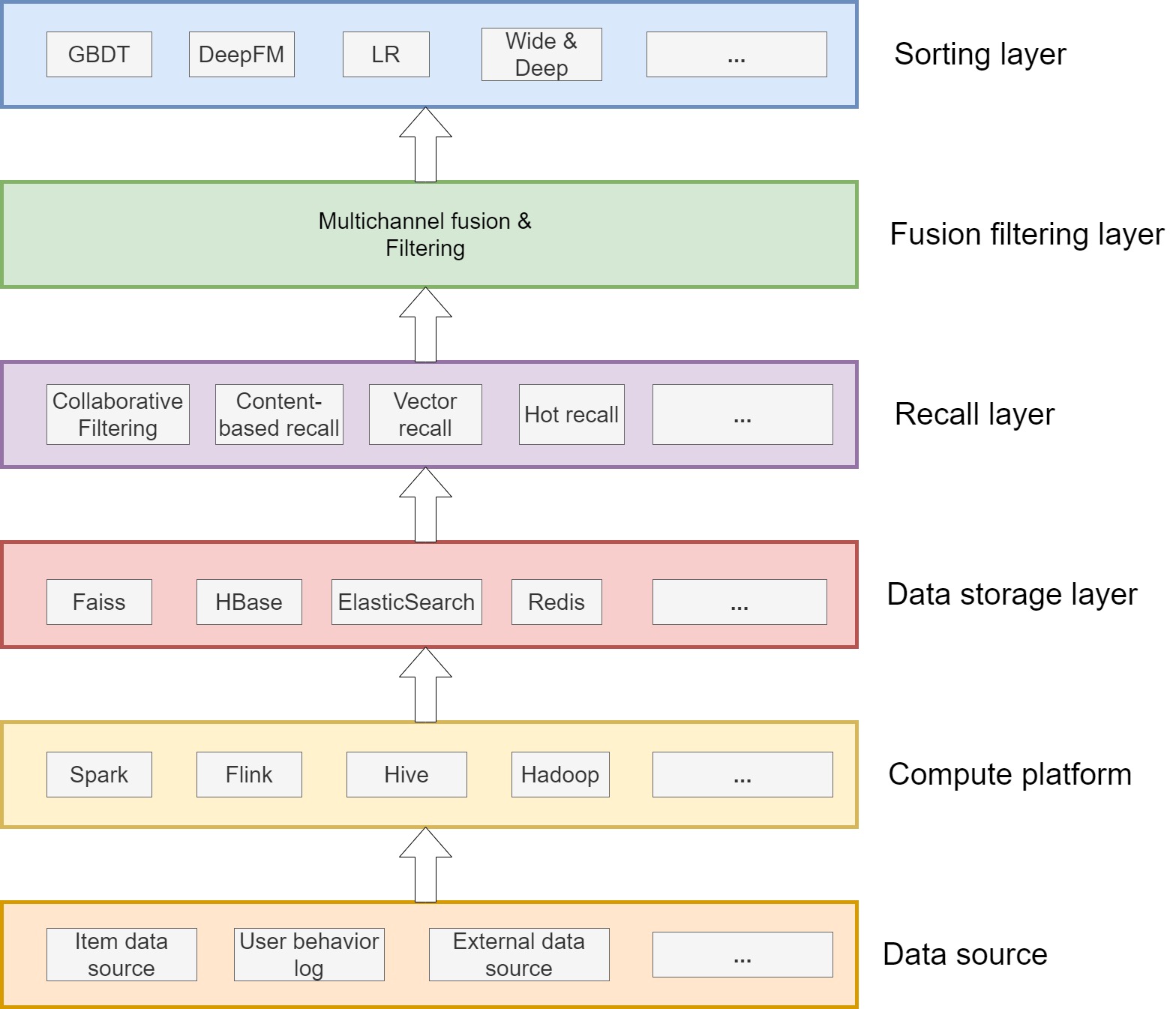
At present, the research on the combination of knowledge graph and recommendation algorithm is still in its infancy, and further research is still needed to obtain more excellent combination effects.

3 Methodology

This section declares the detailed theories of collaborative filtering algorithm, similarity measurement, knowledge graph and techniques related to my method.

3.1 Architecture of recommender system

The figure below shows the architecture of recommender system. There are layers from the bottom up, the main effect of each layer is as follow.



**Figure 2** Typical recommender system architecture

Data source layer: Recommendation algorithms rely on a variety of data sources, including item data, user data, behavior logs, other available business data, and even data from outside the company. Compute platform: Responsible for cleaning, processing, off-line calculation and real-time calculation of various underlying heterogeneous data. Data storage layer: The data processed by the computing platform can be stored in different storage systems as required. For example, Redis can store user feature and user portrait data, ES can be used to index object data, and Faiss can store user or object vector, etc. Recall layer: It includes various recommendation strategies or algorithms, such as classic collaborative filtering, content-based recall, vector-based recall, and popular recommendation for bottom support. To cope with high concurrent traffic online, recall results are usually calculated, indexed and cached. Fusion filtering layer: Trigger multi-way recall, because each recall source of recall layer will return a candidate set, so this layer needs to be fused and filtered. Machine learning or deep learning models and richer features are used for reordering, and smaller and more accurate recommendation sets are selected and returned to the upper business.

From the data storage layer to the recall layer, then to the fusion filtering layer and sorting layer, the candidate set is reduced layer by layer, but the accuracy requirement is getting higher and higher, so it also brings the layer by layer increase of computational complexity, which is the biggest challenge of the recommendation system.

3.2 Collaborative filtering algorithm

Collaborative filtering algorithm is the most frequently used algorithm in existence. Collaborative filtering algorithm does not need a lot of professional knowledge and can use the wisdom of the crowd, which is a kind of algorithm with relatively small restrictions, so it has been studied effectively (xxxx). There are two types of collaborative filtering algorithms (xxxx). One is memory-based collaborative filtering algorithm, which includes user-based collaborative filtering algorithm and object-based collaborative filtering algorithm. The other is model-based collaborative filtering algorithm, which includes matrix decomposition, Bayesian classification model, clustering model, graph model and other algorithms. Among collaborative filtering recommendation algorithms, the memory-based collaborative filtering algorithm mainly relies on the user's historical rating data of the item. The user rating data can be represented by a mn matrix A(m, n), row m represents m users, column n represents n items. The *i*st row and *j*st column’s element represents user i’s rating on item j. The user rating matrix is shown in Figure 2, where the scores in the matrix indicate the user's interest in a certain item.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Ite | … | Ite | … | Ite |
| Use |  | … |  | … | / |
| … | … | … | … | … | … |
| Use |  | … | / | … |  |
| … | … | … | … | … | … |
|  | / | … |  | … |  |

Figure Matrix of user rating data

3.2.1 User-based collaborative filtering

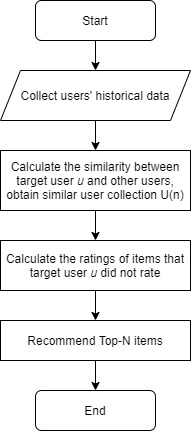
User-based collaborative filtering algorithm (xxxx) is a kind of nearest neighbor algorithm, which is widely used. It recommends items mainly through the interest preferences of neighborhood users. Its working principle is through technically counting users who share the same preferred items with target user and calculate N neighborhood users who have the highest similarity. Then recommend items to the target user that the target user has not accessed but that these N users have.

Figure 4 User-based collaborative filtering algorithm flow chart

The flow steps are as follow. Firstly, collect historical information about users by accessing historical behavior data. Secondly, it mainly calculates the similarity between the target user and other users through the access behavior and selects the user set U(n) with similar behavior to the target user. The key to this step is to calculating user similarity. Thirdly, from items visited by similar sets of users, the items not visited by the target user are identified and a rating prediction is made for the unrated items. Lastly, top-n recommendation is made for items with predicted scores that are not accessed by target users, and the N items with the highest score are recommended.

Figure 3 indicates that the essence of user-based collaborative filtering is actually a history tracking reasoning problem. The algorithm collects similar user sets by counting the likes of other users and target user for certain kinds of feeding items. Based on the preferences of users in similar user sets, recommend items that similar users like and that the target users have not visited.

3.3 Conventional similarity measurement method

In the application algorithm of this paper, the most central and important step is the calculation of similarity. In order to finding the closest neighbors to the target user, the similarity between users must be measured. The several users with the highest similarity would be considered the closest neighbor of target user. Whether the closest neighbors are accurate directly influences the recommendation quality of the whole recommender system.

The user rating data matrix is already shown in figure2. The method to measure the similarity between user *i* and user *j* is firstly, obtain every item user *i* and user *j* rated and then calculate the similarity between user *i* and user *j* with different similarity measurement method, denoted as *sim*(*i, j*).

There are many ways to measure similarity between users, mainly includes three methods: cosine similarity, correlation similarity and adjusted cosine similarity.

Cosine similarity means the user rating is considered as the vector of n-dimension item space, if user has no rating on the item, then the user’s rating on the item is set to 0. The similarity between user is measured by cosine angle between vectors. Set the ratings of user *i* and user *j* on n-dimension item space to be and , then the similarity *sim*(*i, j*) between user *i* and user *j* is

(1)

Where the numerator is inner product of two user rating vector, denominator is module product of two user vector.

Correlation similarity is that set as item collection scored by both user *i* and user *j*. Then, the similarity *sim*(*i, j*) between user *i* and user *j* which is measured by Pearson relativity coefficient is

(2)

Where represents the rating of user *i* to item *c* and and represent the average rating of user *i* and user *j* to the item.

Adjusted cosine similarity is that in cosine similarity measurement method, problem of different users’ rating scale is not taken into consideration. The adjusted cosine similarity can improve such disadvantages by subtracting users’ average rating on items. Set as item collection scored by both user *i* and user *j*, and represent the ratings of user *i* and user *j* on item collection, respectively. Then, the similarity *sim*(*i, j*) between user *i* and user *j* is

(3)

Where represents the rating of user *i* to item *c* and and represent the average rating of user *i* and user *j* to the item.

3.4 The construction of knowledge graph

Knowledge graph consists of ontology layer and instance layer. Ontology layer describes abstract notion and attributes. It makes data specification of instance layer. Instance layer contains large amount of data and facts. Knowledge graph can be divided into domain knowledge graph and universal knowledge graph. As for this paper, domain knowledge graph is used. Figure 4 shows construction procedure.

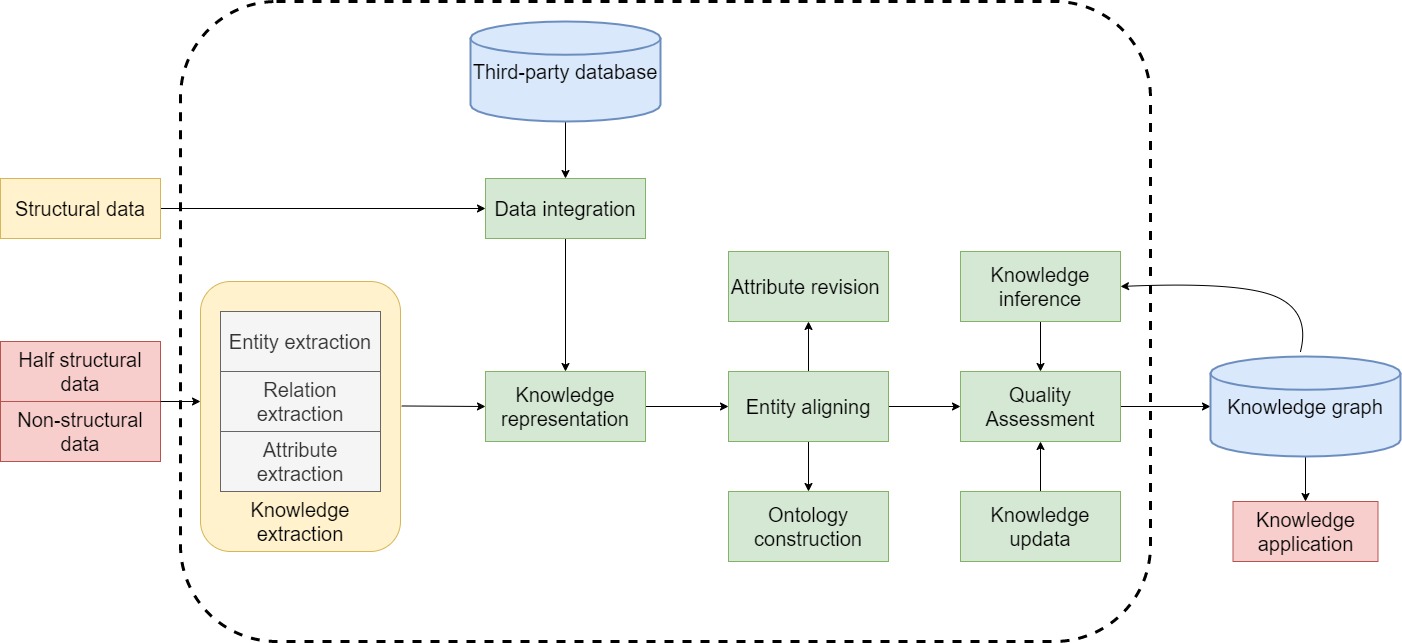


Figure Knowledge graph construction procedure

3.4.1 Knowledge modeling

Knowledge modeling is also known as domain ontology modeling. It is similar to the definition of table structure in relational database. Good definition on pattern can better organize knowledge, better describe relations between knowledge, decrease data redundancy and increase application efficiency.

According to Jia et al (2016), knowledge modeling often has two methods: top to down and down to top. Whichever way can produce a data pattern with good layer structure. These two methods have their advantages respectively. The quality of knowledge model would directly influence the knowledge graph constructed in the later phases. The most commonly used knowledge representative language and framework of knowledge modeling are RDF, RDFS and OWL.

3.4.2 Knowledge storage

After the completion of constructing knowledge graph, rational manner is needed to store knowledge graph. The function of knowledge storage is to design bottom layer storage manner. Good knowledge storage manner can increase efficiency of query. There are three most commonly used knowledge storage manners.

The first one is storage solution based on relational database. Relational database is so popular among researchers and developers that it is thought about in the first place to be used to store knowledge graph. The triple table is used to construct in database a table with three columns, which consists of subject, predicate and object. Each row of data in the table is a piece of triple. Triple table is simple and direct.

The second one is RDF triple database-oriented database storage solution. RDF is a kind of data format that the representation and distribution of knowledge graph cannot live without.

The third one is original graph database-based storage solution. Comparing to relational database and RDF triple, graph database is not only outstanding on correlation depth query and graph calculation engine, but also better in data flexibility and development swiftness than the above two manners. The most commonly used graph database include Neo4j, which is used in the experiment of this paper, and OrientDB, JanusGraph and so on.

3.4.3 Knowledge extraction

Knowledge is an important technique to realize automatic construction of knowledge graph. It helps to extract entity, attributes and relations between entities from all kinds of data sources. It contains three techniques: entity extraction, relation extraction and attribute extraction. Entity extraction is to detect name entity (some nouns which have certain direction) from text data and classify it into defined categories. After entity extraction, relation extraction is needed. The goal of relation extraction is to correlate name entities that are obtained after entity extraction and to gain retiform knowledge structure. Attribute extraction means recognizing and extracting attribute information of certain entity from data source. For example, for movie *Deception*, a series of attribute information such as director, actors, release date can be recognized and extracted from encyclopedia data.

3.4.4 Knowledge integration

Knowledge extraction results in information that is illogical and not hierarchical, confusing and of low quality. There may even exist large amount of replicated and abnormal information. Knowledge integration is to combine knowledge from different data source, mark the knowledge with a global and universal way to prevent the situation where a single entity has different marks. Knowledge integration mainly includes two parts: entity linking and knowledge merging.

Entity linking is to link extracted entity to correct entity in knowledge base. It contains entity disambiguation and coreference resolution. Entity disambiguation is a technique to resolve ambiguation problem caused by different entities having same name. In reality, same name different people and same name different items are not rarely seen. Coreference resolution, also known as entity aligning, is a technology to judge whether multiple items point to a same entity. Its goal is to merging those pointed items to a only confirmed correct entity.

Knowledge merging is that during the construction of knowledge graph, situations such as short of knowledge amount may occur. By then, knowledge can be obtained from third-party knowledge graph or existing structural data and be integrated into local knowledge graph. Such operations are called knowledge merging. Merging external knowledge graph is also called knowledge graph integration. When applying knowledge graph to practical production environment, different people would construct different entities according to their comprehension. Even in the same domain, the entities constructed by different organizations may result in large difference. This is called ontology heterogeneity. When constructing or applying knowledge graph, knowledge is often obtained from different knowledge graphs from the same domain. However, ontology heterogeneity will cause knowledge in different knowledge graphs to not interact properly. This is when knowledge graph integration is needed to combine knowledge from different sources. Knowledge graph integration includes two aspects: the integration of the ontology layer and the integration of the instance layer. The integration method of ontology layer is ontology integration and ontology mapping (Chang et al, 2019). The purpose of ontology integration and ontology mapping is to eliminate ontology heterogeneity and realize the interoperability of heterogeneous ontology. The fusion matching of the instance layer is similar to the ontology layer, but in practical applications, the strength estimation of the knowledge graph is often large, so the integration of the instance layer is often a large-scale data processing problem, and it is necessary to consider the time in the integration and matching process.

3.4.5 Knowledge calculation

Knowledge calculation represents graphical digging, calculation and knowledge inference. Through these two manners, knowledge graph can give full play to their own capabilities, change the traditional application, improve the quality of service. In some circumstances we can infer unknown facts or relations according to known facts or relations, this is called knowledge inference. Since the essence of knowledge graph is a graph, many related algorithms which based on graph theory can function on knowledge graph. Some conventional application scenarios, such as movie recommendation, can perform distinctively when combining with the graph calculation.

3.4.6 Knowledge application

Knowledge application is the last phase in the knowledge graph construction life circle. It’s to combine the constructed knowledge graph with practical business. In this paper, it’s to combine the movie knowledge graph with collaborative filtering algorithm and make recommendations to specified users. Like ontology modeling, knowledge graph construction also has two methods: top-down and bottom-up. According to different fields and different data conditions, different methods need to be adopted. For those mature fields with a complete knowledge system and a wide coverage, the domain knowledge graph can be constructed only by the top-down method; but for those fields where the data is not systematic, the bottom-up method needs to be used for the construction of the knowledge graph. Combining the two methods is a commonly used method in practical work.

4 Construction of movie knowledge graph

According to the work requirement of this paper, a movie knowledge graph needs to be constructed from scratch. The content of above chapter implies that the logic layer of knowledge graph can be divided into schema layer and data layer. Schema layer is mainly managed by ontology database, the quality of which will directly influence the quality of knowledge graph. Next, this chapter will analyze content elements involved in the movie knowledge graph, provide detailed methods to ontology library construction of knowledge graph, obtain knowledge graph data and store the data.

4.1 The overview of movie knowledge graph

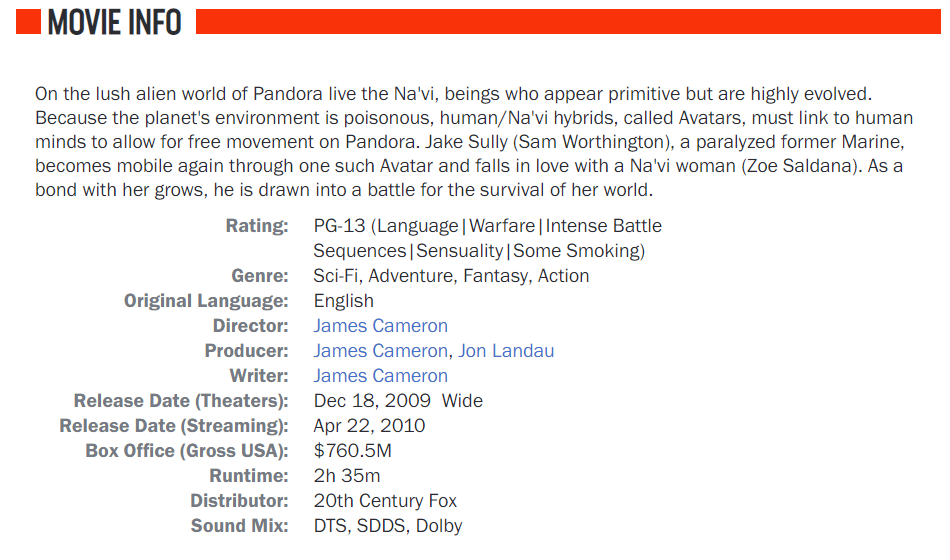
The knowledge graph constructed in this paper is intended to be used in the collaborative filtering recommendation algorithm as an auxiliary semantic information in the subsequent work. The data used in this paper comes from the field of film, so the knowledge graph constructed belongs to the domain knowledge graph. For the construction of knowledge graph in the field of film, first of all, knowledge elements involved in this paper need to be determined.

Figure Basic elements a movie contains

As shown in figure 5, elements a movie involves contains movie title, actor, director, writer, release data, rating, genre and so on. The elements of the movie are classified abstractly according to the characteristics of conceptual abstraction of things in ontology library. Since the ontology library will not represent the specific attributes of things, this paper will only study and analyze the main conceptual elements in the field of film.

4.2 Ontology modeling of schema layer

After confirming basic elements in knowledge graph of film domain, the next step is to construct ontology library in schema layer. The construction of ontology library is very important. The previous chapter has shown that there are two ways to construct ontology library. Since this paper is primarily in the movie domain, choose top-down method to construct knowledge graph. This paper choose Protégé as ontology construction tool.

4.2.1 Knowledge graph and ontology

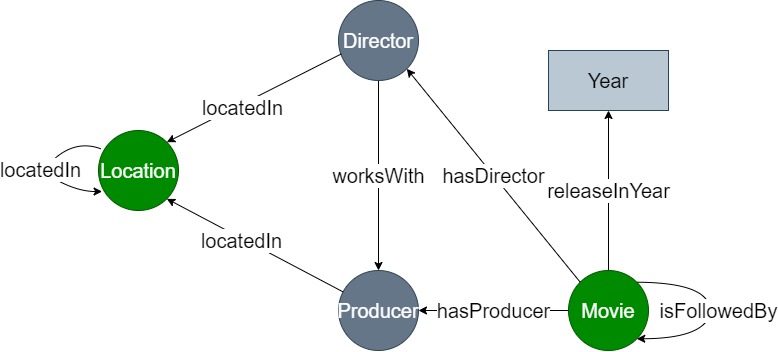
Ontology is a modeling tool that can describe the conceptual model of information system at the semantic and knowledge levels (xxxx), and it belongs to the schema layer of knowledge graph. Ontology is primarily the realization of an abstract representation of things in the real world. By summarizing and expressing the concept and relation of the real thing, the facts expressed by it can be associated and reasoned effectively. In short, ontology is a data set used to describe a domain, and it is the skeleton of knowledge.

Figure 7 Movie ontology

There are several principles about ontology design. The terms ontology define must be explicit and objective. The terms ontology define must be semantically intact. The terms defined by the ontology must be content consistent with the reasoning. The main procedure of constructing ontology library is as follow. Determine the goal of constructing domain knowledge graph. Categorical representation of represented entities. Constraints on attributes and range of entities. Restrict other constraints.

4.2.2 Ontology description language

Ontology description language can help users to write clear and formal concept description for domain model, so it should meet the requirements such as well-defined syntax, well-defined semantics, effective inference support, sufficient expression ability and convenience of expression and so on. According to the demand of detail preference in user modeling, researchers have developed a variety of ontology description languages with different bias, such as RDF, RDFS, Loom, OWL and so on. According to the detailed direction of different ontology description languages and the description requirements of this paper, it has obvious advantages to adopt OWL as the ontology description language of this paper. OWL (Web Ontology Language) is a vocabulary expansion on RDF, which includes not only lexical representations of classes and instances, but also representations of more relationships. According to (xxxx), OWL has the following advantages: first, OWL has strong expression ability and rich knowledge representation and reasoning ability. Second, OWL introduces Boolean operators, builds complex classes recursively, and provides the representation of existing value constraints, arbitrary value constraints and quantity-value constraints. Third, OWL can define special properties of attributes, including TransitiveProperty, SymmetricProperty, FunctionProperty and InverseFunctionProperty. Last, OWL is one of the cores of the Semantic Web technology stack, it provides fast, flexible data modeling capabilities and efficient automatic reasoning.

In conclusion, compared with other ontology description languages, OWL has a better modeling effect on the ontology library in the field of film.

4.2.3 Ontology library construction procedure

Based on what has been described in chapter 4.2.1, the main work of this chapter is completing the ontology construction of movie domain schema layer. The implementation procedure is as follow:

First, determine the purpose of ontology library construction. According to the application field of this paper, a corresponding ontology library of the film domain has to be established according to the information elements involved in the film domain, and realize the abstract representation of things at the conceptual level, which lays a foundation for the construction of the domain knowledge graph.

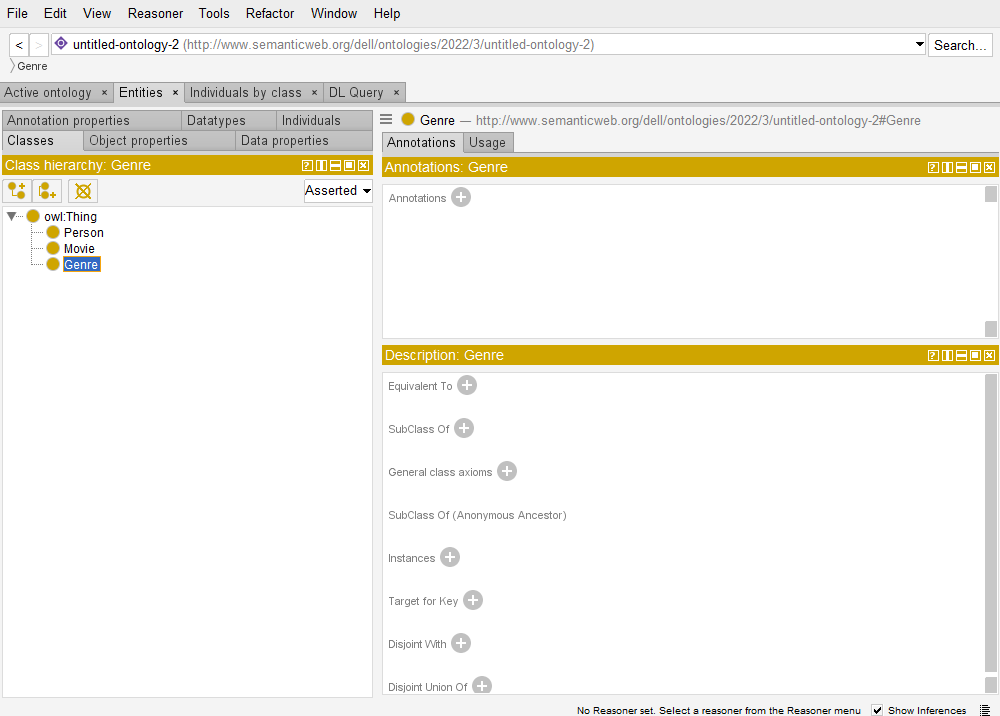
 Second, categorical representation of represented entities. Through analysis on knowledge of movie domain, it turns out that the domain contains information such as movie, actor, director, writer and genre, and the basic information of actor, director and writer share the same content. Therefore, they can be abstracted to a concept, Person. In this chapter, there are three concepts abstracted from ontology library, which are Movie, People and Genre. When constructing ontology in Protégé, firstly fill new ontology resources’ IRI in Ontology IRI. IRI can be filled according to the users’ own requirement. As show in figure 7, create new class in the Class tab, *Thing* is the largest class in ontology library, which represents everything. The other abstracted concepts are its sub-classes. This paper applies the below three concepts: Movie, Person and Genre. The definition description of OWL on classes is shown in figure 8.

Figure 8 Movie domain entity class definition



Figure 9 Ontology classes defined by OWL

Third, apply constraints on attributes and range of entities. After classifying entities conceptually, restrict the attributes and range of these classes accordingly. According to the different types of object attributes and data attributes in ontology library, the number of their restrictions is also different. In this paper, the object attributes defined in ontology library includes *hasActedIn, hasActor, hasGenre* and so on, 7 attributes in total.

**Table 1** Movie class attribute set

|  |  |  |  |
| --- | --- | --- | --- |
| element | attribute | symbol | domain |
| title | data | m\_title | String |
| genre | object | hasGenre | Genre |
| director | object | hasDirector | Person |
| writer | object | hasWriter | Person |
| actor | object | hasActor | Person |
| tag | data | m\_tag | String |
| language | data | m\_language | String |
| rate | data | m\_rate | String |
| time length | data | m\_length | String |
| release date | data | m\_release | String |
| release company | data | m\_produce | String |

**Table 2** Person class attribute set

|  |  |  |  |
| --- | --- | --- | --- |
| element | attribute | symbol | domain |
| name | data | p\_name | String |
| birth place | data | p\_nation | String |
| birthday | data | p\_birthday | String |
| movie acted in | object | hasActedIn | Movie |
| movie directed | object | hasDirectedIn | Movie |
| movie wrote | object | hasWritedIn | Movie |

**Table 3** Genre class attribute set

|  |  |  |  |
| --- | --- | --- | --- |
| element | attribute | symbol | domain |
| name | data | g\_name | String |

Fourth, Restrict other constraints. Once the attributes of the entity class have been qualified, then need to qualify the class in other ways. In ontology library, there are some inversible attributes between Movie and Person. This paper defined 7 object attributes above, which contains three pairs of mutually inversible attribute, which are *hasActor-hasActedIn, hasDirector-hasDirectedIn, hasWriter-hasWritedIn*. For these type of data, this paper use *Inverse of* in Protégé to restrict.

4.3 Data acquisition of knowledge graph

In the previous chapter, it is known that there are three data sources of knowledge graph, and the movie data used in this paper belongs to structured data. Structured data can be stored using D2RQ to transform data from relational databases into RDF triples of knowledge graphs. Chapter 4.2 has described the information content involved in the construction process of ontology library to realize the concept layer in the movie domain knowledge graph. The realization of ontology database construction lays a foundation for the establishment of the knowledge graph. Then need to fill the data layer of the knowledge graph, carry out instantiation operation and import data according to the classes in the ontology library and the realization purpose of this paper. Knowledge graph is a kind of graph structure data, which needs to select graph database to store its data. Currently, graph database mainly has three categories: First, open source database: RDF4j, gStore and so on. Second, commercial database: Virtuoso, AllegroGraph, Stardog and so on. Third, Original graph database: Neo4j, OrientDB, Titan and so on. After comparing the advantages and disadvantages of each graph database, this paper will choose Neo4j as the data storage tool.

4.3.1 Data sources

The movie field data for this article is provided by the website The MovieDatabase (TMDb). Launched in 2008, TMDb aims to help the media center community provide high-resolution posters and fan art. Later, it gradually developed into an open film data sharing platform, similar to Baidu Baike, Douban Film, IMDB and so on. TMDb, which is used by millions of people every week, is considered the best place to get movie data. It is also the first and only site to offer a free movie database API.

For ratings of movie, this paper uses Netflix Prize Data. Netflix Prize Data mainly records the movie score records of nearly one trillion anonymous users in Netflix movie community from 1988 to 2005. This paper mainly uses this data set to collect and obtain users' behavior information on movies. Therefore, the field information of basic title, ID and ratings of movies in the data set by corresponding user ID is extracted for the construction of knowledge map in the next stage.

4.3.2 Data acquisition

The TMDB 5000 Movie Dataset is acquired from Kaggle. Kaggle was founded by co-founder and chief executive Anthony Goldbloom in Melbourne in 2010 as a platform for developers and data scientists to host machine learning competitions, host databases, and write and share code. The platform has attracted the attention of 800,000 data scientists, and these users may be the main attraction for Google.

The Dataset records nearly 5000 netflix movie related information, such as staff, budget amount, the style of the film, theme and so on. In this paper, when constructing the knowledge graph of film knowledge base, the actual business scenes are combined. Several information that users are concerned about in actual scenarios are selected to improve information extraction efficiency and save computation, which includes genres, keywords, production companies, original title. In essence, fields useful to this experiment are extracted from this data set and stored in a new CSV file for the construction of the movie knowledge graph. The field information of the new table includes the name of the field to be extracted and the title of the movie.

4.3.3 Data cleaning

There are many noise in the downloaded source data. For example, some data types have serious data loss and inconsistent data formats. The methods to deal with these problems are as follow.

First, format conversion. In newly downloaded data, the information in person is all json format. Actors, directors, and screenwriters need to be read out in string format. The title, genre, production\_companies are json format as well and need to be transformed to string. The json of the person, movie and genre datasets are parsed respectively.

Second, Merge data. Whether the fields in person and movie are the same, if so, delete the same fields in movie. For example, if person is the same as the name in movie, delete the name in movie. The resulting fields are: movie\_id, title, director, genre, and so on.

4.3.4 Data Storage

After cleaning the data, begin to store it. Instead of a common relational database, this paper will choose a graph database to store the data. Currently, the most commonly used graph database includes Neo4j and Apache Jena. After overall consideration, this paper chose Neo4j to store the data. Neo4j is an online graph storage tool, after activated on computer, enter the database through website “http://localhost:7474”. Neo4j has three categories: node, relation and attribute. The knowledge graph view represents the relationship between entities through nodes and edges. The schema in Neo4j is shown in the table below.

Table 4 Element representation in neo4j

|  |  |  |
| --- | --- | --- |
| name | content | corresponding ontology library class |
| node | Movie, Person, Genre | class |
| edge | hasActor, hasActedIn  hasDirector, hasDirectedIn  hasWriter, hasWritedIn, hasGenre | object attribute |
| attribute | m\_title, m\_tag\_m\_language  m\_length, m\_released, m\_rate  m\_rating, m\_produce, p\_name  p\_nation, p\_birthday, g\_name | data attribute |

4.3.5 Data flow framework

In Section 4.2, the construction of movie domain ontology library has been realized, laying a foundation for the construction of knowledge graph. This section mainly describes the subsequent process of knowledge graph construction, as shown in figure 8. After obtaining the movie dataset, this section will read the movie data in TMDB dataset based on the movie name, director and release time. When all the movie data of the target query is completed, the obtained movie data will be cleaned and formatted. Finally, it is stored in Neo4j to realize the construction of knowledge graph.

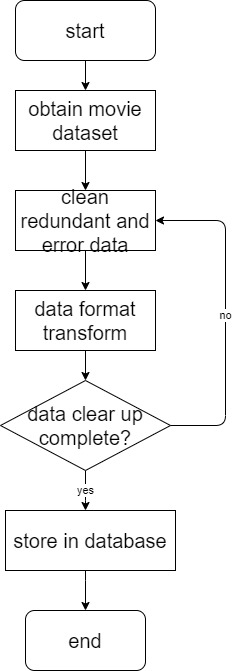


Figure 10 Knowledge graph construction flow chart

4.4 Experiment results

This chapter demonstrates the constructed ontology library and knowledge graph.

4.4.1 Realization of ontology library of schema layer

Section 4.2.2 classifies entities conceptually and restrict the attributes of classes. In the ontology library, the relationship between each class can be intuitively seen through view, but the attributes and range of each class cannot be visually displayed in view. All the classes constructed are subclasses of Thing. View the structure figure x of the ontology library that has been built by clicking Ontograf in Protege.

4.4.2 Realization of knowledge graph

After the data was stored in Neo4J, we constructed entities, their attributes and semantic relations between entities, completing the construction of knowledge map in the field of film. Entities in the knowledge graph include movies, people, and movie genres.

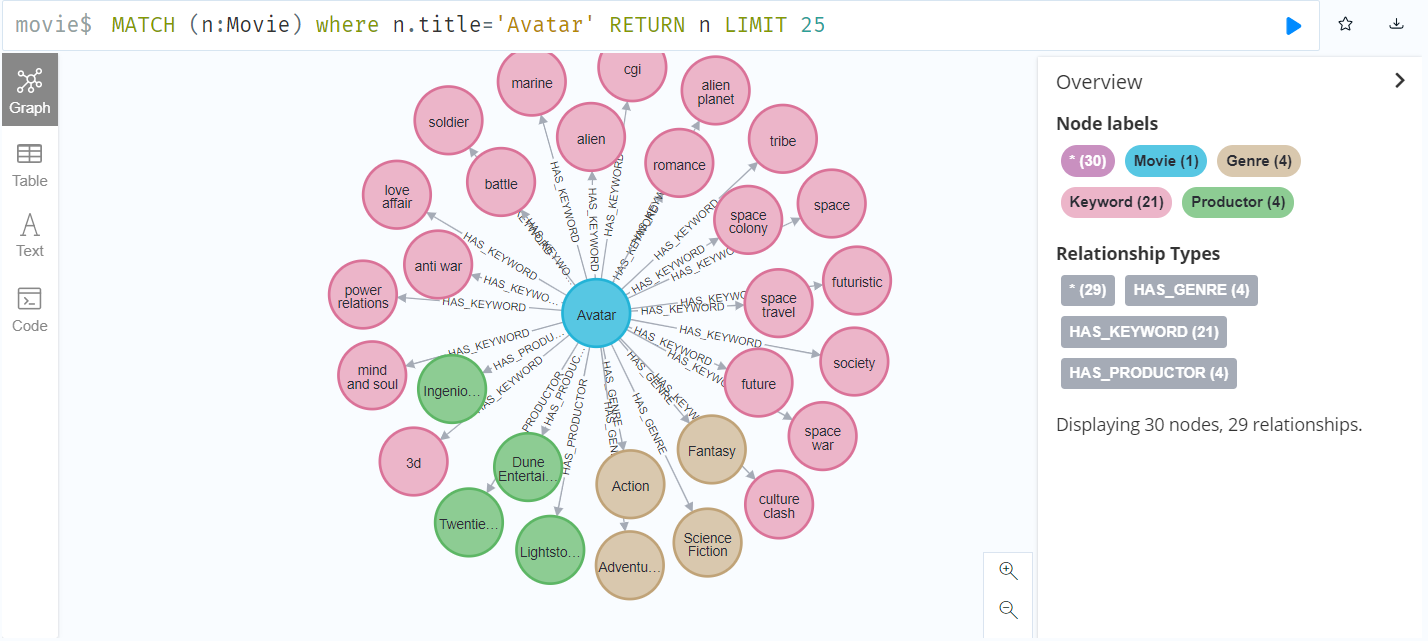


Figure 11 Single movie entity and its relation between object attribute entity

Figure 10 takes movie *Avatar* as example, demonstrates this movie and its relation between object attributes. According to the relations between entities represented by each edge, it turns out that blue, orange, pink, green nodes represent Movie, Genre, Keyword and Productor respectively.

Expands from movie genre *Adventure*, figure11 shows the movie genre contains all the movie nodes in the usage data range.

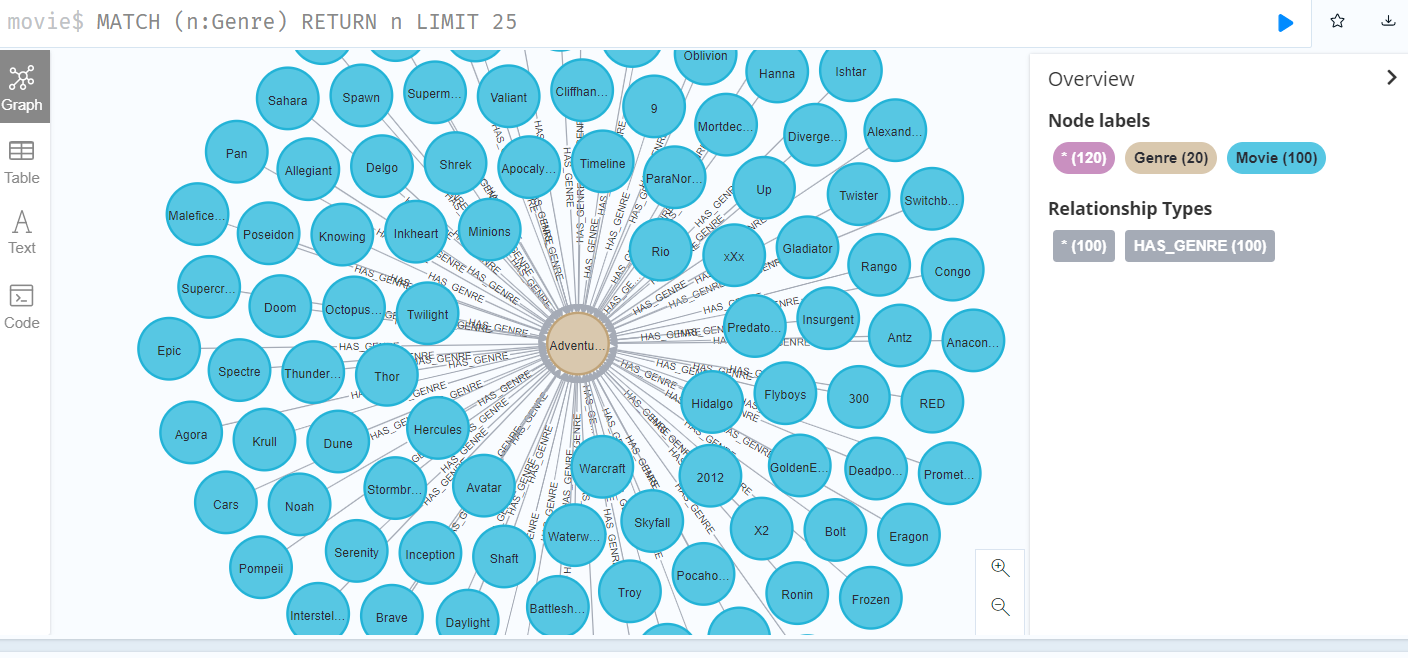


Figure 12 Multiple relation example of single entity in knowledge graph

According to the above design ideas of ontology library, node types and relationship types are embodied in Neo4j. The knowledge graph constructed based on Neo4j has a visual structure diagram, which can more intuitively show the relationship between entities. Table 5 shows the entities and relationships involved after the construction of the film knowledge graph.

Table 5 The amount of relations and entities in knowledge graph

|  |  |  |
| --- | --- | --- |
| Knowledge type | Knowledge name | Knowledge amount |
| entity | Movie | 18205 |
| entity | User | 177 |
| entity | Genre | 20 |
| entity | Productor | 1616 |
| entity | Keyword | 3728 |
| relation | hasGenre | 2359 |
| relation | hasKeyword | 7264 |
| relation | hasProductor | 2836 |
| relation | rated | 7445 |

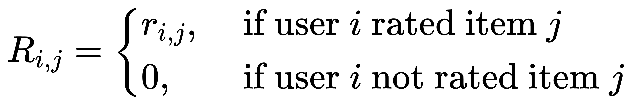
There is large amount of movie information in TMDB. According to the statistics in the table, 20018 entities and 19904 relationships are extracted. At this point, the construction of the knowledge graph has been completed.

5 Experiment on KG-based collaborative filtering algorithm

When using the recommender system, collaborative filtering recommendation algorithm is generally selected, but due to the user's few historical behaviors, the data will be sparse, which leads to poor recommendation effect and cold start problem. Therefore, this paper proposes an improved algorithm using knowledge graph as auxiliary information tool of the item in the collaborative filtering algorithm.

5.1 Analysis on conventional similarity measurement method

Although the conventional collaborative filtering algorithm is simple and easy to operate when making recommendations, its recommendation effect is affected by the historical behavior data of users, and the quality of the data determines the effect of the recommendation system. In practical applications, the amount of data in the database is very large, and there are many data users cannot access, which inevitably leads to data sparsity. Under such circumstances, conventional similarity measurement method has some disadvantages.

 In the cosine similarity measurement method, the scores of all the items that users do not score are assumed to be 0, and the scores of user *i* on item *j* is . Then, when constructing the user rating data matrix A(m,n), for any item *j*, the score of user *i* on any item is

(n)

The advantage of this is that it can effectively improve the computing performance, but in the case of extremely large number of items and extremely sparse number of user ratings, the credibility of the above assumption is not high, because users assume 0 for all unrated items. In fact, the degree of user to unscored items would not be exactly the same, ratings on these items could not be exactly the same, which is zero. Therefore, under the extreme condition that the user rating data are sparse, cosine similarity cannot be effectively measuring the similarity between the user, the adjusted cosine similarity has the same problems.

In the correlation similarity measurement method, the item set rated by user A is represented by , so when calculating the similarity between user *i* and user *j*, the intersection of item set rated by user *i* and user *j* needs to be calculated first:

(n)

Then the similarity between user *i* and user *j* is calculated on item set by using the correlation similarity measure. Common sense indicates that it is more certain of similarity between users only if they score similarly across a larger number of items. A collection of items jointly rated by two users in the case of extremely sparse user rating data is smaller and usually has only one or two items. Even if users score very similar on such a small set of items, it cannot be sure that the similarity between them is very high according to common sense. Therefore, in the case of extremely sparse user rating data, the relevant similarity measurement method also has certain drawbacks.

To sum up, the conventional similarity measurement methods cannot effectively measure the similarity between users in the case of extremely sparse user rating data, which makes the nearest neighbor of the calculated target user inaccurate and leads to a sharp decline in the recommendation quality of the whole recommendation algorithm.

5.2 Collaborative filtering based on item score prediction

Due to the extreme sparsity of user rating data, the traditional similarity measurement method cannot effectively calculate the nearest neighbor of the target user, and the recommendation quality of collaborative filtering recommendation system is difficult to guarantee (xxxx). To solve the extreme sparsity of user rating data, the simplest solution is to set user ratings for unrated items to a fixed default value, which usually is set to medium value of rating domain, such as 3 in 5-grading system, or set it to the average user rating. Experiments show that this improved method can effectively improve the recommendation accuracy of collaborative filtering recommendation system.

Obviously, users' ratings of unrated items cannot be identical, so the above improved method cannot fundamentally solve the problems existing in traditional similarity measurement methods when user rating data is extremely sparse. This paper calculates the similarity between items and predict users’ rating on unscored items by users’ rating on similar items to increase commonly scored items between user, which can effectively solve the shortage of conventional similarity measurement under extreme condition that user rating data are sparse and make the calculated nearest neighbors of target users relatively accurate.

Next, this paper will introduce collaborative filtering algorithm based on item rating in detail. The algorithm is mainly divided into two steps: search for the nearest neighbor and generate recommendations.

5.2.1 Search for nearest neighbors

In order to effectively solve the problems existing in conventional similarity measurement methods when user rating data is extremely sparse, this paper proposes that when calculating the similarity between user *i* and user *j*, first calculate the union set of item set rated by user *i* and user *j*, set item set rated by user A as , then is

(n)

The unrated items of user *i* and user *j* in item set are predicted by user's ratings of similar items, and then the similarity between user *i* and user *j* is calculated on item set. This method can not only effectively solve the problem that there is less common user rating data in the relevant similarity measurement method, but also effectively solve the problem that the scores of all unrated goods in the cosine similarity measurement method and the modified cosine similarity measurement method are the same, which is zero. This makes the nearest neighbor of the calculated target user more accurate, thus effectively improving the recommendation quality of the recommendation algorithm.

Predicting user *i*'s rating of unrated items in item set is the key to collaborative filtering algorithm based on item similarity. Let the set of unscored items of user *i* in item space be represented by

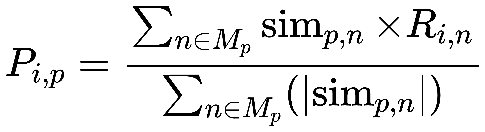
(n)

For any item , use the method below to predict user *i*’s rating on item *p* (xxxx).

First, compute the similarity between item *p* and other items. Similar to computing the similarity between users, it first requires obtaining all user ratings for items *i* and *j*, then compute the similarity between items *i* and *j* through the various similarity measures introduced in Section 3.3.

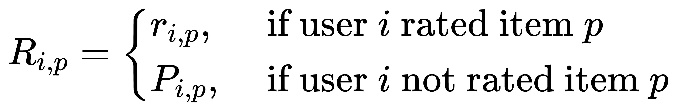
Second, the items with the highest similarity are regarded as the set of neighbor items of item *p*, that is, finding the set of items in the whole project space , which makes *p*, meanwhile, similarity of item and item *p*, which is *sim*(*p*, ) reaches highest and similarity of item and item *p*, which is *sim*(*p*, ) second to it, the rest can be done in the same manner.

Third, after obtaining , predict user *i*’s rating on item *p*, which is , according to the method introduced by xxxx (xxxx)



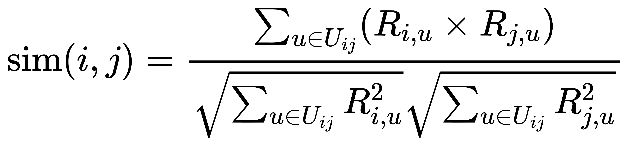
(n)

After processing by the above method, user *i* and user *j* have scores for all items in the item set , that is, for any project , user *i*'s score for project *p* is

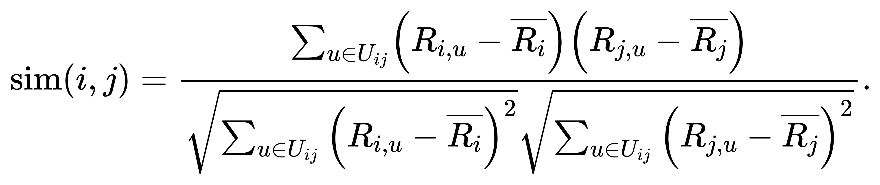


(n)

Then, based on the item set , the similarity between user *i* and user *j* is calculated through the three similarity measures introduced in Section 3.3. In the method proposed in this paper, the correlation similarity is consistent with the adjusted cosine similarity, which is as follows:



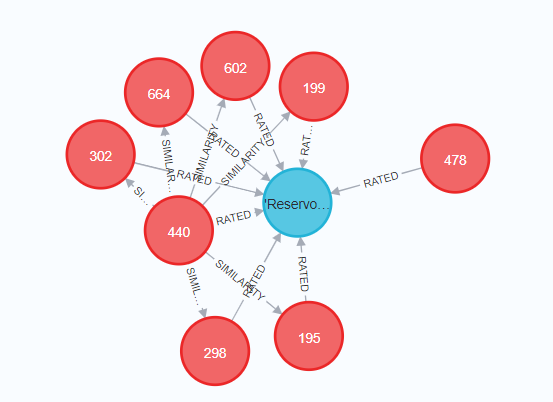
(n)



(n)

Formula n is cosine similarity and formula n is correlation similarity (adjusted cosine similarity). The goal of finding the nearest neighbor is to find the user set C={,,…,}, making *u*C, and the similarity *sim*(*u*,) between and *u* is the highest, followed by the similarity *sim*(*u*,) between and *u*, and so on.

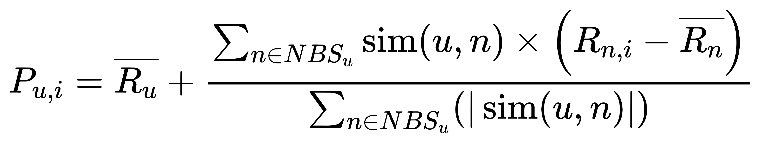
In section 4.4, the movie knowledge graph has been stored in graph database Neo4j. Apply the above methods on the movie data in the database. Take user 440 as example, the red nodes are user id and the blue point is the movie. It can be seen that the nearest neighbors 302, 664, 602, 199, 298, 195 are calculated.

****

**Figure 13** Nearest neighbors of user 440

5.2.2 Generate recommendation

The similarity measurement method proposed in this paper is used to obtain the nearest neighbor of the target user, and the next step is to generate corresponding recommendations. Assume that user *u*'s nearest neighbor set is represented by , then user *u*'s predicted score of item *i*, which is , can be obtained by user *u*'s score on items in user *u*'s nearest neighbor set, . According to Breese et al (1998) the calculation method is as follows.



(n)

*sim*(*u*,*n*) indicates the similarity between user *u* and user *n*. indicates the score of user *n* on item *i*. indicates the average score of user *u* and user *n* on item *i* respectively.

5.3 Experiment result and analysis

This section will make a comprehensive arrangement of the experimental algorithms proposed in this chapter.

5.3.1 Dataset

This paper adopts Netlix prize data dataset provided by Kaggle (https://www.kaggle.com/). The dataset gives approximately 480189 user and 17770 movie ratings. This paper selects 6000 rating data from the user rating database as the experimental data set, which contains 145 users and 805 movies, of which each user has rated at least 20 movies.

The whole experimental dataset needs to be further divided into training set and test set. For this purpose, this paper introduces variable *x* to represent the percentage of training set in the whole data set. For example, *x*=0.8 means that 80% of the entire dataset is the training set and 20% is the test set. In all experiments in this paper, *x*=0.8 was adopted as the experimental basis.

The experiment hardware is Intel(R) Core(TM) i7-8550U CPU 1.80GHz 1.99 GHz, the memory is 8 GB. The experiment environment is Spyder in Anaconda, with Python 3.9.

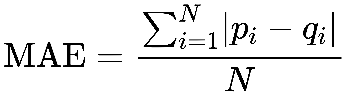
To measure the sparsity of the entire dataset, this paper introduces the concept of sparsity level, which is defined as the percentage of unrated items in the user rating data matrix. The sparsity level of the movie dataset this paper selected is

1-6000/(145805)=0.9486.

5.3.2 Metric

The measurement standards for evaluating recommendation quality of recommendation system mainly include statistical accuracy measurement method and decision support accuracy measurement method (xxxx). The Mean Absolute Error (MAE) in statistical accuracy measurement method is easy to understand and can intuitively measure recommendation quality. MAE is the most common recommendation quality measurement method. In this paper, MAE is adopted as the measurement standard. MAE measures the accuracy of the prediction by calculating the deviation between the predicted user ratings and the actual user ratings. The smaller the MAE, the higher the recommendation quality.

Let the set of predicted user ratings be represented as { }, the corresponding actual user score set is { }, then MAE is defined as

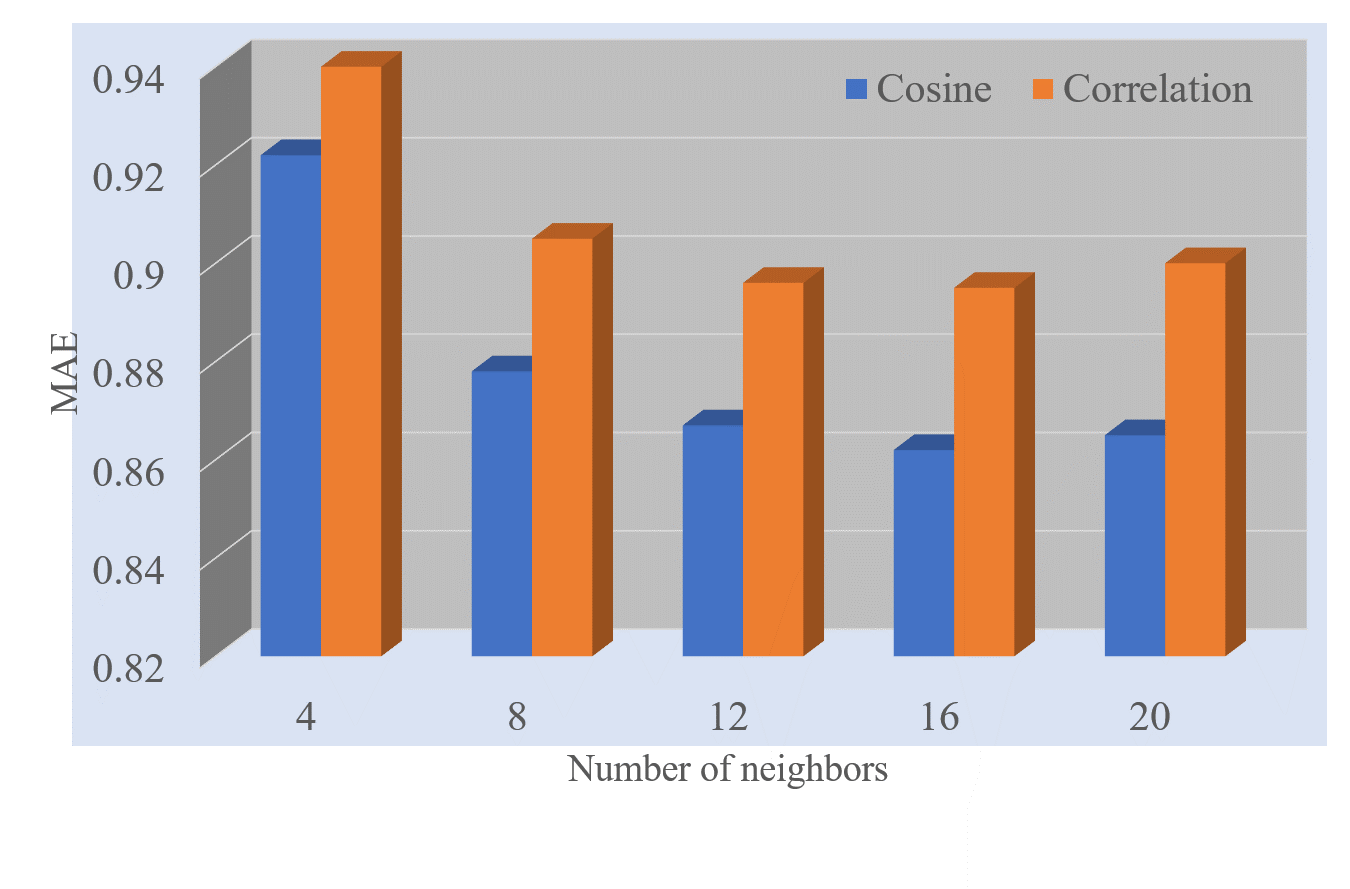


(n)

5.3.3 Comparison of similarity measures

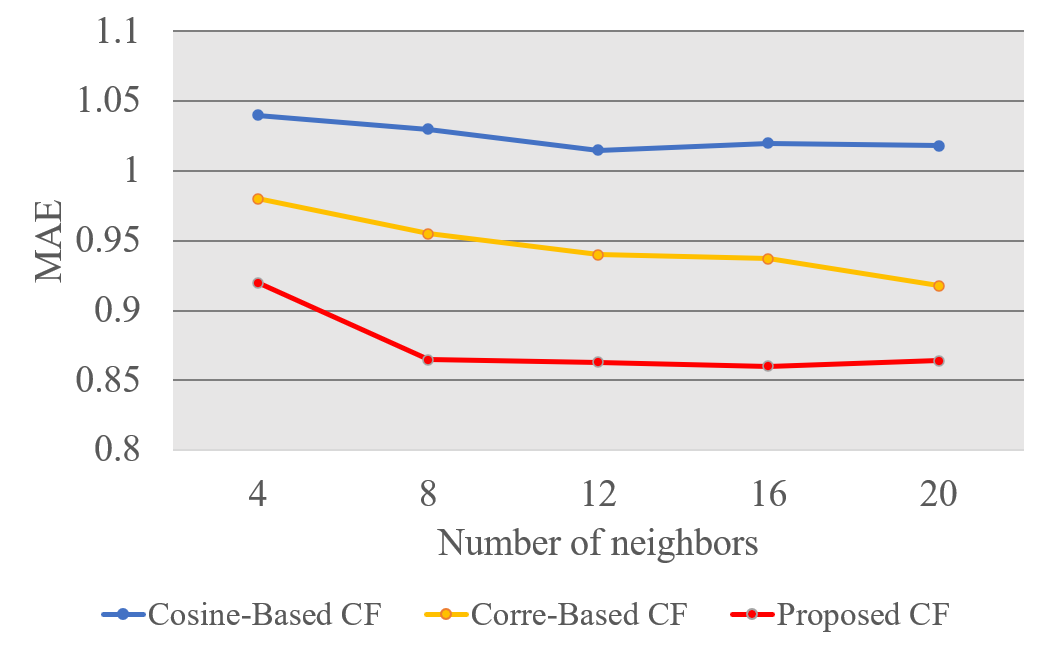
In the calculation of similarity between items, cosine similarity is chosen as the measure standard of similarity because this method is easy to implement, the prediction speed is relatively fast, and the prediction accuracy is relatively high.

In the collaborative filtering recommendation algorithm based on item score prediction, the modified cosine similarity measure method is consistent with the correlation similarity measure method, which is collectively called correlation similarity. The original cosine similarity is called cosine similarity. The two similarity measurement methods are used to conduct experiments on the dataset respectively, and MAE is calculated. The number of neighbors increase from 4 to 20 with an interval of 4. The experimental results are shown in figure below. Figure 2 shows that under various experimental conditions, the MAE of cosine similarity measurement method is lower than that of correlation similarity measurement method. Therefore, in subsequent experiments, cosine similarity is selected as the measurement method of similarity between users in collaborative filtering recommendation algorithm based on item score prediction.



**Figure 14** Comparison of similarity measure methods

5.3.4 Comparison of recommendation algorithm performance

In order to test the effectiveness of the algorithm proposed in this paper, take the traditional collaborative filtering recommendation algorithm as the contrast. In the traditional collaborative filtering recommendation algorithm, cosine similarity and correlation similarity are respectively used as similarity measurement standards to calculate MAE. The number of neighbors increases from 4 to 20, and the interval is 4. Then it is compared with the collaborative filtering recommendation algorithm based on item score prediction proposed in this paper. The experiment result is as follows.

**Figure 15** Comparison of accuracy of recommendation algorithms

授权页包括英文授权页与中文授权页，都不得修改。也无需填写任何内容。

引言（或绪论）简要说明研究工作的目的、范围、相关领域的前人工作和知识空白、理论基础和分析、研究设想、研究方法和实验设计、预期结果和意义等。应言简意赅，不要与摘要雷同，不要成为摘要的注释。一般教科书中有的知识，在引言中不必赘述。

**最重要的规定与阅读提示**：

(1) 本模板的本章（仅限本章）此处“最重要的规定与阅读提示”之前的内容是现行的北京交通大学本科生毕业设计论文模板之中的内容；从现在开始，直至本章结束的所有内容都是针对威海校区计算机科学与技术专业学生的。

(3) 本模板是基于现行的北京交通大学本科生毕业设计论文模板（以下简称为“原始模板”）进行修改的。

(4) 所有学生都应该从此处“最重要的的规定与阅读提示”开始顺序阅读。

(5) 在原始模板第1章之中，我们给出正文之前的所有页面的撰写规定，所有页码都是指本Microsoft Office Word文件的自然页码，而非页面之上所标注的页码。

(6) 在原始模板第2章中，我们首先给出了关于论文“正文”与论文的3级“标题”所需要注意的问题，随后，将原始北京交通大学本科生毕业设计论文模板之中所给出的格式要求内容，分别单列在第2章的各个大节之中。没有任何原则性修改，完全与我课堂上给大家讲授的内容一致。

(7) 学生需要撰写的地方一律采用英文，除非有明确说明。

本文件的第1页是封面。

“设计（论文）题目”为中文，其中若包括英文单词或缩写，这采用Times New Roman加粗字体；字号都是16 points，单倍行距，居中排版。

“设计（论文）英文题目”当然全部采用英文；采用Times New Roman加粗字体，字号仍为16 points，单倍行距，居中排版。注意，学生需要自行控制多行的换行位置。

封面其他部分给出了实例，较为明确，不再叙述。

注意，所有下划线需要保持长度一致。

不要修改日期。

保证封面为一整页，不得变成两页，保证“日期”为最后一行。

详细中文摘要应将论文的内容要点简短明了地表达出来，约800字左右。

字体与字号的规定是，中文采用普通小四号宋体，英文采用普通12 points的Times New Roman字体。

内容应包括工作目的、研究方法、成果和结论。要突出本论文的创新点，语言力求精炼。特别注意，详细中文摘要比英文摘要需要更详尽，详细中文摘要的篇幅一般是英文摘要的一倍。

应在摘要之后另起一行注明论文的关键词，一般为3至5个专业常见的实词。关键词之间使用分号，最后一个关键词之后没有标点符号。

英文摘要应将论文的内容要点简短明了地表达出来，约400字左右。

采用普通12 points的Times New Roman字体。

有关关键词是一样的，但关键词之间需要使用西文分号，最后一个关键词之后也没有标点符号。

1.5 目录

无论是在正文之中，还是在目录页之上，都只能有3级目录。学生可以采用“更新域”方式更新目录，但必须进行格式修正与控制。并且，保持上面示例的行距。这3级目录之中的标题格式规定如下：

(1) 1级标题：中文采用黑体，英文采用Times New Roman字体；都是12 points的大小；“左对齐”且左侧空1字符。

(2) 2级标题：本级之中一般没有中文，但是若有中文，则采用宋体；英文都是采用Times New Roman字体。字号也都是12 points的大小；左侧再缩进1字符。

(3) 3级标题：本级之中一般没有中文，但是若有中文，则采用宋体；英文都是采用Times New Roman字体。字号也都是12 points的大小；左侧再缩进2字符。

注意需要2行的标题，不仅设置行距为单倍行距，且每行都是一个“段落”。

1.5.1

1.6 图示与表格索引等

许多学位论文在目录之后，正文之前给出了其文中的图示（figures）与表格（tables）索引。一般可以将这两部分内容分别列出，也可以合并列出（List of Figures and Tables）。另外，采用的数学符号较为复杂的学位论文，也可以在此处列出文中基本数学符号的含义与用途。

这里不再详细叙述，有需要的同学可以直接问我。

1.7 重要提醒

请大家特别注意，如果在某一页面的最后有“分节符（下一页）”标识，就需要特别留心，**务必不能删去**！

2 正文与标题

在原始的北京交通大学包括毕业设计论文的模板中，此处给出了许多问题的详细规定。在现在修订的适用于威海校区计算机科学与技术专业的本模板中，我们将这些内容分别单列为第2章的各个大节之中。没有任何原则性修改，完全与我课堂上给大家讲授的内容一致。现在只谈两个基本问题：正文与标题。

关于正文的主要问题如下：

(1) 字体与字号：一般的，中文采用普通小四号（即12 points）宋体，英文采用12 points的Times New Roman字体。

(2) 行距：采用“固定值”为20磅（points）。但是，不适用于图示与表格，以及需要单独成行的数学公式等。

(3) 标点符号：一律采用西文标点符号。注意两个问题，其一是标点符号之后的空格问题，其二是若有中文，则也采用“西文标点符号”加“空格”的方式。

关于3级标题的主要问题如下：

(1) 字体与字号：所有这3级标题的字体与字号都是严格按照示例所规定的形式。所有英文都采用Times New Roman粗体（bold）。

(2) 行距：所有这3级标题的行距都是“单倍行距”（single space），即按照示例所表明的行距。注意，大家应该设法避免任何一级标题为两行。这样自动生成目录时也不太好控制。

(3) 空行问题：在Lancaster University所给的以前学生论文样例（samples）中，两个自然段落之间大多有半个空行。我们希望学生自己手工控制这一问题。并非所有这样的空行都是必须的。

(4) 段落首行缩进：强烈建议每一小节的第1自然段采用首行缩进形式，这一点与Lancaster University所给的样例不同；其他自然段落可以不采用首行缩进形式，也可以采用。

(5) 段落对齐方式：一律采用两端对齐方式。Lancaster University所给的论文样例也是这样的。

2.1 图示

学位论文为了需要反映出作者确已掌握了坚实的基础理论和系统的专门知识，具有开阔的科学视野，对研究方案作了充分论证，因此，有关历史回顾和前人工作的综合评述，以及理论分析等，可以单独成章，用足够的文字叙述。正文是学位论文的核心部分，占主要篇幅，可以包括：调查对象、实验和观测方法、仪器设备、材料原料、实验和观测结果、计算方法和编程原理、数据资料、经过加工整理的图表、形成的论点和导出的结论等。

由于研究工作涉及的学科、选题、研究方法、工作进程、结果表达方式等有很大的差异，对正文内容不能作统一的规定。但是，必须实事求是，客观真切，准确完备，合乎逻辑，层次分明，简练可读。

**图：**

图包括曲线图、构造图、示意图、框图、流程图、记录图、地图、照片等。

图应具有“自明性”。

图应有编号。图的编号由“图”和从“1”开始的阿拉伯数字组成，图较多时，可分章编号。

图宜有图题，图题即图的名称，置于图的编号之后。图的编号和图题应置于图下方。

照片图要求主题和主要显示部分的轮廓鲜明，便于制版。如用放大缩小的复制品，必须清晰，反差适中。照片上应有表示目的物尺寸的标度。

图片示例如下（**全文统一按顺序编号**）：

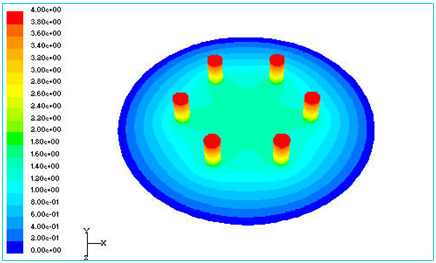


图1 太合金多炭钢铁产品柱扭曲局部受力分析示意图

2.2 表格

**表：**

表应具有“自明性”。

表应有编号。表的编号由“表”和从“1”开始的阿拉伯数字组成，表较多时，可分章编号。

表宜有表题，表题即表的名称，置于表的编号之后。表的编号和表题应置于表上方。

表的编排，一般是内容和测试项目由左至右横读，数据依序竖读。

表的编排建议采用国际通行的三线表。

如某个表需要转页接排，在随后的各页上应重复表的编号。编号后跟表题（可省略）和“（续）”，置于表上方。

续表均应重复表头。

表格示例1（**全文统一按顺序编号**）：

表1 国际单位制的基本单位

|  |  |  |
| --- | --- | --- |
| 量的名称 | 单位名称 | 单位符号 |
| 长度 | 米 | m |
| 质量 | 千克(公斤) | kg |
| 时间 | 秒 | s |
| 电流 | 安[培] | A |
| 热力学温度 | 开[尔文] | K |
| 物质的量 | 摩[尔] | mol |
| 发光强度 | 坎[德拉] | cd |

2.3 公式

**公式：**

论文中的公式应另行起，并缩格书写，与周围文字留足够的空间区分开。

如有两个以上的公式，应用从“1”开始的阿拉伯数字进行编号，并将编号置于括号内。公式的编号右端对齐，公式与编号之间可用“…”连接。公式较多时，可分章编号。

公式示例如下（**全文统一按顺序编号**）：

 (1)

 (2)

式中 —— 多孔质材料的平均粒子直径(m)；

—— 孔隙度（孔隙体积占总体积的百分比）；

—— 特征渗透性或固有渗透性，与材料的结构性质有关(m2)。

较长的公式需要转行时，应尽可能在“＝”处回行，或者在“+”、“－”“×”、“/”等记号处回行。

公式中分数线的横线，其长度应等于或略大于分子和分母中较长的一方。

如正文中书写分数，应尽量将其高度降低为一行。如将分数线书写为“/”，将根号改为负指数。

公式示例2：

将  写成 1/ 或 

2.4 其他

**引文标注**

论文中引用的文献的标注方法遵照GB/T 7714－2005，可采用顺序编码制，也可采用著者－出版年制，但全文必须统一。

**注释**

当论文中的字、词或短语，需要进一步加以说明，而又没有具体的文献来源时，用注释。注释一般在社会科学中用得较多。

应控制论文中的注释数量，不宜过多。

由于论文篇幅较长，建议采用文中编号加“脚注”的方式。最好不用采用文中编号加“尾注”。

3

表2 中文字号和西文磅值之间的对应关系

Table 2 Relationship between the Chinese character size and the point size

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序号** | **中文字号** | **西文磅值** | **公制高度** | **备注** |
| 1 | 初号 | 42磅 | 14.82毫米 |  |
| 2 | 小初 | 36磅 | 12.70毫米 |  |
| 3 | 一号 | 26磅 | 9.17毫米 |  |
| 4 | 小一 | 24磅 | 8.47毫米 |  |
| 5 | 二号 | 22磅 | 7.76毫米 |  |
| 6 | 小二 | 18磅 | 6.35毫米 | 本模板使用 |
| 7 | 三号 | 16磅 | 5.64毫米 |  |
| 8 | 小三 | 15磅 | 5.29毫米 | 本模板使用 |
| 9 | 四号 | 14磅 | 4.94毫米 |  |
| 10 | 小四 | 12磅 | 4.23毫米 | 最常用的字号 |
| 11 | 五号 | 10.5磅 | 3.70毫米 | 最常用的字号 |
| 12 | 小五 | 9磅 | 3.18毫米 |  |
| 13 | 六号 | 7.5磅 | 2.56毫米 |  |
| 14 | 小六 | 6.5磅 | 2.29毫米 |  |
| 15 | 七号 | 5.5磅 | 1.94毫米 |  |
| 16 | 八号 | 5磅 | 1.76毫米 |  |

4

4.1

4.1.1

5 Conclusions

论文的结论是最终的、总体的结论，不是正文中各段的小结的简单重复。结论应该准确、完整、明确、精练。如果不可能导出应有的结论，也可以没有结论而进行必要的讨论。可以在结论或讨论中提出建议、研究设想、仪器设备改进意见以及尚待解决的问题等。

**References**

【威海校区同学：前半部分内容为原始北京交通大学本科生毕业设计论文模板之上的内容，但是我们将字号从原来的10.5 points修改成了11 points；都半部分为建议格式的示例。】参考文献是文中引用的有具体文字来源的文献集合。按照GB 7714《文后参考文献著录规则》的规定执行。

参考文献以文献在整个论文中出现的次序用[1]、[2]、[3]……形式统一排序、依次列出。

参考文献的表示格式为:

**著作:**

[序号] 作者.译者.书名.版本.出版地.出版社.出版时间.引用部分起止页

**期刊:**

[序号] 作者.译者.文章题目.期刊名.年份.卷号(期数).引用部分起止页

**会议论文集：**

[序号]作者.译者.文章名.文集名 .会址.开会年.出版地.出版者.出版时间.引用部分起止页

**学位论文：**

[序号]作者.题名[学位论文]（英文用[Dissertation]）.保存地点.保存单位.年份.引用部分起止页

**专利:**

[序号] 专利申请者.题名.国别.专利文献种类.专利号.发布日期.引用部分起止页

**技术标准:**

[序号] 起草责任者.标准代号.标准顺序号—发布年.标准名称.出版地.出版者.出版年份.引用部分起止页

【威海校区同学：以下为建议格式的示例】

表7-2 参考文献列表的例子2

|  |  |
| --- | --- |
| [1] | Hong. J, and Mao, C. Incremental discovery of rules and structure by hierarchical and parallel clustering. In: Piatetsky-Shapiro G, Frawley W J (eds.) Knowledge Discovery in Databases. Menlo Park, California: AAAI Press / The MIT Press, 1991, 177-194. |
| [2] | Godin R, Missaoui R, Alaui H. Incremental concept formation algorithms based on Galois (concept) lattices. Computational Intelligence, 1995, 11(2): 246-267. |
| [3] | Pawlak Z. Rough Sets: Theoretical Aspects of Reasoning about Data. Dordrecht: Kluwer Academic Publishers, 1991. |
| [4] | Quinlan, J. R. C4.5: Programs for Machine Learning. San Mateo, CA: Morgan Kaufmann, 1993. |
| [5] | Bai Hong-Yi, Jiang He-Lun, et al. Research Report on Fake News in 2018. Journalist, 2019, 431(01):6-16) (in Chinese).  （白红义, 江海伦等. 2018年虚假新闻研究报告[J]. 新闻记者, 2019, 431(01): 6-16).） |
| [6] | Ramamohanarao, K. and Bailey, J. Discovery of emerging patterns and their use in classification. In: Gedeon, T. D. and Fung, L. C. C. (eds.), The Proceedings of Artificial Intelligence 2003, LNAI 2903, 2003. pp. 1-12. |

【威海校区同学：这里我们只说明一个问题】

不允许仅出现中文形式的参考文献著录条目。如果一篇参考文献是中文的，则也必须以英文著录为主，并在英文著录之后，另起一行再列出中文著录形式。参见上面的第5篇参考文献。也请注意，**中文参考文献**著录之中的标点符号都是使用英文的标点符号，参见我们在第2章“正文与标题”中，“关于正文的主要问题”说明的第(3)条款。

**Acknowledgements**

放置在参考文献页后，对象包括：1）国家科学基金，资助研究工作的奖学金基金，合同单位，资助或支持的企业、组织或个人。2）协助完成研究工作和提供便利条件的组织或个人。3）在研究工作中提出建议和提供帮助的人。4）给予转载和引用权的资料、图片、文献、研究思想和设想的所有者。5）其他应感谢的组织和个人。

【只允许1页】

**Appendixes**

**Appendix A:** 程序代码

附录是作为论文主体的补充项目，并不是必须的。

论文的附录依序用大写正体英文字母A、B、C……编序号，如：附录A。

**Appendix B:** 工程图纸