东 华 大 学

毕业设计（论文）任务书

课 题 名 称 ： 针对季节性服饰商品的推荐系统的分析和设计

学 院 ： 旭日工商管理学院

专 业 ： 信息管理与信息系统

姓 名 ： 孔庆璇

学 号 ： 160610132

指 导 教 师 ： 范 宏

二 零 二 一 年 一 月 五 日

一、毕业设计（论文）的目的与要求：

1.培养学生灵活运用所学的各门课程的专业知识，尤其是软件工程、管理信息系统、数据库等核心课程以及统计学、数据结构、运筹学等基础学科的知识。培养分析和解决工程技术问题的工作能力；

2.巩固、深化学生所学基本理论、基本知识和基本技能，在此基础之上鼓励学生思考，进行创新；

3.综合训练学生获取有效信息的能力，例如通过问卷调查、和实际访问、座谈会等形式了解研究现状，明确用户的需求；训练学生的信息检索能力，例如，通过图书馆查阅书面文献，通过网络技术查阅数据文献包含期刊、专著、论文等等收集有效资料；

4.培养学生的理论分析的能力，能从得到的信息中分析出有效的信息，例如，目前存在的问题以及发展的愿景，并能根据所得信息进行可行性的分析。另一方面，从获得的文献知识中进一步检索出自己需要的信息；

5.进一步培养学生制定设计或试验方案的能力；实验、研究能力；技术经济分析和组织工作能力；总结提高、撰写论文和设计说明书的能力等等；

6.培养学生参与社会实践和实验室科研的兴趣。培养学生的创新能力和团队精神，树立良好的学术思想和工作作风。

二．毕业设计（论文）的内容：

1.绪论，分析研究背景和意义、研究现状和研究内容；

2.模型理论基础，阐述传统推荐模型理论，包括协同过滤方法、基于内容的推荐方法和混合推荐方法，以及本文模型搭建所需要的Item2vec、Bi-LSTM和self-

attention算法；

3. 数据分析与预处理，介绍本文使用的数据情况，并分析其中的商品和用户数据；

4. 模型构建，提出本文的由项目嵌入和用户偏好组成的模型框架，并阐述本文的输入准备、算法流程和推荐生成；

5. 模型验证，阐述推荐系统的评估指标，并分析本文的模型效果；

6. 系统设计与实现，阐述本文的总体设计、数据库设计、模块设计、输入输出设计、界面设计和系统测试；

7. 结论与展望，总结本文实现的推荐系统，对未来优化方向进行展望。

三、毕业设计（论文）课题应完成的工作：

1. 拟定论文方向及题目

2. 查找相关参考文献

3. 选定外文文献并翻译

4 阅读参考文献，撰写文献综述

5. 在文献综述的基础上撰写开题报告

6. 论文任务书的撰写和修改

7. 系统的设计和分析

8. 数据的收集和清洗

9. 模型的搭建

10. 模型的评估

11. 系统的实施、测试和维护

12. 论文的撰写

四．毕业设计（论文）进程的安排：

序 号	设计(论文)各阶段名称	日　　期
1	论文开题	2020 年 11 月 01 日~ 2021 年 01 月 05 日
2	系统设计和分析	2021 年 01 月 06 日~2021 年 01 月 09 日
3	数据收集和清洗	2021 年 01 月 10 日~2021 年 01 月 16 日
4	推荐算法建模	2021 年 01 月 17 日~2021 年 02 月 06 日
5	推荐算法优化	2021 年 02 月 07 日~2021 年 03 月 20 日
6	系统实施	2021 年 03 月 21 日~2021 年 04 月 17 日
7	系统测试和维护	2021 年 04 月 18 日~2021 年 05 月 01 日
8	论文撰写	2021 年 04 月 18 日~2021 年 05 月 10 日

五．应收集的资料及主要参考文献：

[1] Adomavicius G, Tuzhilin A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions[J]. IEEE transactions on knowledge and data engineering, 2005, 17(6): 734-749.

[2] Billsus D, Pazzani M J. Learning collaborative information filters[C]//Icml. 1998, 98: 46-54.

[3] Breese J S, Heckerman D, Kadie C. Empirical analysis of predictive algorithms for collaborative filtering[J]. arXiv preprint arXiv:1301.7363, 2013.

[4] Deshpande M, Karypis G. Item-based top-n recommendation algorithms[J]. ACM Transactions on Information Systems (TOIS), 2004, 22(1): 143-177.

[5] Huang Z, Chen H, Zeng D. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering[J]. ACM Transactions on Information Systems (TOIS), 2004, 22(1): 116-142.

[6] Pazzani M J. A framework for collaborative, content-based and demographic filtering[J].

Artificial intelligence review, 1999, 13(5): 393-408.

[7] Pazzani M, Billsus D. Learning and revising user profiles: The identification of interesting web sites[J]. Machine learning, 1997, 27(3): 313-331.

[8] Robertson S, Walker S. Threshold setting in adaptive filtering[J]. Journal of documentation, 2000.

[9] Resnick P, Iacovou N, Suchak M, et al. Grouplens: An open architecture for collaborative filtering of netnews[C]//Proceedings of the 1994 ACM conference on Computer supported cooperative work. 1994: 175-186.

[10] Sarwar B, Karypis G, Konstan J, et al. Application of dimensionality reduction in recommender system-a case study[R]. Minnesota Univ Minneapolis Dept of Computer Science, 2000.

[11] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms[C]//Proceedings of the 10th international conference on World Wide Web. 2001: 285-295.

[12] Shardanand U, Maes P. Social information filtering: Algorithms for automating "word of mouth" [C]//Proceedings of the SIGCHI conference on Human factors in computing systems. 1995: 210-217.

[13] Somlo G L, Howe A E. Adaptive lightweight text filtering[C]//International Symposium on Intelligent Data Analysis. Springer, Berlin, Heidelberg, 2001: 319-329.

[14] Yu K, Xu X, Tao J, et al. Instance selection techniques for memory-based collaborative filtering[C]//Proceedings of the 2002 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, 2002: 59-74.

[15] Zhang Y, Callan J. Maximum likelihood estimation for filtering thresholds[C]//Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval. 2001: 294-302.

[16] Zhang Y, Callan J, Minka T. Novelty and redundancy detection in adaptive filtering[C]//Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval. 2002: 81-88.

[17] Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality[J]. arXiv preprint arXiv:1310.4546, 2013.

[18] Barkan O, Koenigstein N. Item2vec: neural item embedding for collaborative filtering[C]//2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP). IEEE, 2016: 1-6.

[19] Graves A, Schmidhuber J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures[J]. Neural networks, 2005, 18(5-6): 602-610.

[20] Goldberg D, Nichols D, Oki B M, et al. Using collaborative filtering to weave an information tapestry[J]. Communications of the ACM, 1992, 35(12): 61-70.

[21] Schafer J B, Frankowski D, Herlocker J, et al. Collaborative filtering recommender systems[M]//The adaptive web. Springer, Berlin, Heidelberg, 2007: 291-324.

[22] Pazzani M J, Billsus D. Content-based recommendation systems[M]//The adaptive web. Springer, Berlin, Heidelberg, 2007: 325-341.

[23] Burke R. Knowledge-based recommender systems[J]. Encyclopedia of library and information systems, 2000, 69(Supplement 32): 175-186.

[24] Burke R. Hybrid recommender systems: Survey and experiments[J]. User modeling and user-adapted interaction, 2002, 12(4): 331-370.

[25] Deshpande M, Karypis G. Item-based top-n recommendation algorithms[J]. ACM Transactions on Information Systems (TOIS), 2004, 22(1): 143-177.

[26] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms[C]//Proceedings of the 10th international conference on World Wide Web. 2001: 285-295.

[27] RESNICK P. AnOpen Architecture for Collaborative Filterring of Netnews[C]//Proc CSCW'94. 1994.

[28] Burke R. Hybrid web recommender systems[J]. The adaptive web, 2007: 377-408.

[29] Bellogín A, Cantador I, Díez F, et al. An empirical comparison of social, collaborative filtering, and hybrid recommenders[J]. ACM Transactions on Intelligent Systems and Technology (TIST), 2013, 4(1): 1-29.

[30] Christakou C, Vrettos S, Stafylopatis A. A hybrid movie recommender system based on neural networks[J]. International Journal on Artificial Intelligence Tools, 2007, 16(05): 771-792.

[31] Nanopoulos A, Rafailidis D, Symeonidis P, et al. Musicbox: Personalized music

recommendation based on cubic analysis of social tags[J]. IEEE Transactions on Audio, Speech, and Language Processing, 2009, 18(2): 407-412.

[32] Al-Shamri M Y H, Bharadwaj K K. Fuzzy-genetic approach to recommender systems based on a novel hybrid user model[J]. Expert systems with applications, 2008, 35(3): 1386-1399.

[33] Cao Y, Li Y. An intelligent fuzzy-based recommendation system for consumer electronic products[J]. Expert Systems with Applications, 2007, 33(1): 230-240.

[34] Tan S, Bu J, Chen C, et al. Using rich social media information for music recommendation via hypergraph model[J]. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 2011, 7(1): 1-22.

[35] McCarthy K, Reilly J, McGinty L, et al. Thinking positively-explanatory feedback for conversational recommender systems[C]//Proceedings of the European Conference on Case-Based Reasoning (ECCBR-04) Explanation Workshop. 2004: 115-124.

[36] Garfinkel R, Gopal R, Tripathi A, et al. Design of a shopbot and recommender system for bundle purchases[J]. Decision Support Systems, 2006, 42(3): 1974-1986.

[37] Lawrence R D, Almasi G S, Kotlyar V, et al. Personalization of supermarket product recommendations[M]//Applications of data mining to electronic commerce. Springer, Boston, MA, 2001: 11-32.

[38] Hsu S H, Wen M H, Lin H C, et al. AIMED-A personalized TV recommendation system[C]//European conference on interactive television. Springer, Berlin, Heidelberg, 2007: 166-174.

六、任务执行日期：

自__2020__年___11__月___11___日 起，至_2021__年__5_月__15__日 止。

学　　生（签字） _____

指导教师（签字） _____

系 主 任（签字） _____

东华大学

毕业论文

课题名称：针对季节性服饰商品的推荐系统的分析和设计

学院：旭日工商管理学院

专业：信息管理与信息系统

姓名：孔庆璇

学号：160610132

指导教师：范宏

二零二一年五月

针对季节性服饰商品的推荐系统分析与设计

摘要

在电子商务行业的蓬勃发展的同时，它也越发地面临着供求关系难以匹配的问题。尤其在面对最具季节性特征的服饰品类时，电商平台更需要考虑商品的季节相关性。而推荐系统可以从电商平台的海量数据中探索商品与商品以及用户与商品之间的关系，帮助电商平台完成更高效的供需关系的匹配。

本文聚焦于季节性特征的服饰品类的推荐问题。本文使用了亚马逊评论数据，并通过聚类方法验证数据中的季节性特征。针对这一数据特征，本文基于传统推荐算法中的基于项目的协同过滤推荐思路，提出通过 Item2vec 模型学习项目的嵌入向量表示形式从而得到项目与项目间的相似，包括项目间的季节性相似和项目间的内容相似，再通过 Bi-LSTM 和 self-attention 模型学习到用户偏好从而得到用户与项目间的相似的推荐方法。实证表明该推荐方法在召回率、准确率和平均精度上分别达到了 0.64，0.07 和 0.40。

本文将上述推荐方法应用于推荐系统的设计中，并采用了三层信息流程和 MTV 软件架构模式，通过 Django 和 Bootstrap 技术路径，实现了推荐系统的搭建。该推荐系统支持用户查看用户维度的推荐结果和商品维度的推荐结果，提供商品查找及多方式的排序功能，并且允许用户进行心愿单的管理。本系统将为用户提供的高效准确的推荐服务的准确性，给用户带来良好的系统体验。

关键词：协同过滤，项目嵌入，Item2vec，用户偏好学习，Bi-LSTM，self-attention

Recommender System Analysis and Design For Seasonal Apparel Products

Abstract

With the fast developing of e-commerce industry, the platform is facing the problem of supply and demand match. Especially for the apparel products with apparent seasonality feature, e-commerce platform needs to consider the seasonal relevance of products. Recommender system could help explore the relevance between users and items and between items themselves from the massive data on e-commerce platform, which empowers the platform to match the supply and demand more efficiently.

This paper focuses on the recommendation of apparel category with seasonal characteristic. This paper uses Amazon review data and uses clustering methods to verify the seasonal feature in the data. To deal with the seasonal feature, this paper proposes a recommendation method based on the classic item-based collaborative filtering. The method proposed is to user Item2vec model to learn the embedding vector representation of items to obtain the similarity between items, including seasonal similarity and content similarity, and then uses Bi-LSTM and self-attention model to learn user preference to obtain the similarity between users and items. The empirical evidence shows that the recommendation method has reached 0.64, 0.07 and 0.40 in recall rate, accuracy rate and average precision, respectively.

This article applies the above-mentioned recommendation method to the design of a recommendation system, adopts the three-tier information flow and MTV software architecture model, and builds the recommendation system through Django and Bootstrap. The recommendation system supports users to view user-dimensional recommendation results and product-dimensional recommendation results, provides

product search and multi-way of sorting functions, and allows users to manage wishlists. This system provides users with efficient and accurate recommendation service, and brings users a good system experience.

Key word: Collaborative Filtering, Item Embedding, Item2vec, User Preference Learning, Bi-LSTM, self-attention

目　录

1 绪论.....1
1.1　研究背景.....1
1.2　研究意义.....1
1.3　研究现状.....2
1.3.1　技术方法.....2
1.3.2　应用领域.....3
1.4　研究内容.....5
1.5　论文结构安排.....5
2　模型理论基础.....7
2.1　传统模型理论.....7
2.1.1　基于内容的推荐方法.....7
2.1.2　协同过滤方法.....9
2.1.3　混合推荐方法.....11
2.2　研究模型理论.....12
2.2.1　Item2vec 算法.....12
2.2.2　Bi-LSTM 算法.....13
2.2.3　self-attention 算法.....14
2.3 本章小结.....15
3　数据分析与预处理.....16
3.1　数据集介绍.....16
3.2　商品数据分析.....17
3.3　用户数据分析.....18
3.4　本章小结.....18
4　模型构建过程.....19
4.1　模型架构.....19
4.2　输入准备.....21
4.3　算法流程.....23
4.4　推荐生成.....25

4.5 本章小结.....28
5 模型实证分析.....29
5.1 评估指标.....29
5.1.1 决策性指标.....29
5.1.2 排序性指标.....30
5.2 效果验证.....30
5.2.1 对 Item2vec 算法的效果验证.....31
5.2.2 对 Bi-LSTM 算法的效果验证.....32
5.2.3 对 self-attention 算法的效果验证.....34
5.3 本章小结.....36
6 系统设计与实现.....37
6.1 总体设计.....37
6.1.1 系统信息流程设计.....37
6.1.2 软件架构模式设计.....38
6.2 数据库设计.....38
6.3 模块设计.....40
6.4 本章小结.....45
7 结论与展望.....46
参考文献.....47
附录.....52

1 绪论

1.1 研究背景

全球时尚电子商务行业的市场前景十分乐观。虽然在 2020 年由于受到新冠疫情的影响，市场规模有所下滑，但预计到 2023 年，市场仍将恢复，并有望达到 6720 亿美元。其中服装商品的预计市场规模都将继续增长。在中国，服装商品的复合年均增长率有望达到 14.1%。而服饰品类中的鞋类以及配件和包，也将在现有规模上进行持续的增长。

然而，对电子商务行业来说，挑战也随之而来。随着更多的用户与更多的商品会持续地进入到该平台中。如何将用户与商品构建起有效的连接，从而促成更多的交易，以获得更多的营收，是电子商务平台需要持续思考的问题。推荐系统可以在一定程度上帮助电子商务平台解决该问题。它通过当前的用户行为，去预测用户未来的喜好，有效地构建起用户和商品的连接。

而服饰商品作为电子商务平台中的一大品类，由于其特殊的季节性，推荐系统在针对服饰商品进行推荐的时候，首先需要学习商品间的季节性相似和内容相似，再对用户偏好进行学习，从而才能做出更有效的推荐。本研究针对服饰商品的特殊性，分析并设计了一个能够捕捉到商品间季节性相似并且能够对用户长短期偏好行为做出响应的推荐系统，用于解决具有季节性特征的服饰商品的推荐问题。

1.2 研究意义

本文将利用亚马逊评论数据，结合项目本身特征和用户偏好学习，构建针对具有季节性服饰商品的推荐系统。本文的研究意义在于：

（1） 为电子商务平台提供针对服饰商品的推荐系统。该推荐系统能够学习到商品间的季节性相似和内容相似，以及消费者对于商品的长短期偏好，来更有效地对用户进行推荐，从而提高平台在服饰商品品类的盈利；

（2） 为电子商务平台的买家提供更有针对性的推荐。借助该推荐系统，买家有希望在平台上花费更少的时间挑选到符合心意的商品；

（3） 为电子商务平台的卖家提供更多有价值的流量。在该系统下，真正会受到

买家喜爱的产品才会被推荐，因此卖家能够得到更多有潜在购买意愿的流量。

1.3 研究现状

在这一部分中，我们将从技术方法和应用领域两个方面来对概述当前推荐系统的研究成果。

1.3.1 技术方法

随着网络技术的发展以及个性化技术的需求，推荐系统受到了越来越多的关注。推荐系统的早期研究源于信息检索和过滤研究[20]。直到 1990 年代中期才成为一个独立的研究领域。常用的推荐技术包括协同过滤方法[21]、基于内容的推荐方法[22]和基于知识的推荐方法[23]。每一个推荐方法都有其优缺点，例如协同过滤算法的稀疏性、可扩展性和冷启动问题[1,21]以及基于内容的推荐方法的结果过于特定的问题[1]。为了解决上述问题，又提出了许多先进的推荐方法，例如基于计算智能的推荐、基于社交网络的推荐方法、基于上下文感知的推荐方法以及小组推荐方法。

传统推荐方法包括基于内容的推荐方法、协同过滤方法和混合推荐方法[24]。下面将对这三种方法的研究现状进行概述。

基于内容的推荐方法。基于内容的推荐方法是向用户推荐与其喜好的项目相似的其他项目[22]。该方法的基本原理如下。首先，通过分析用户喜好的项目信息，来确定可以用于识别这些项目的共同属性，即用户偏好特征，并将用户偏好特征存储起来。其次，通过每个项目的属性和用户偏好特征的比较，向用户推荐与其偏好高度相似的项目[22]。该方法主要使用了两种技术。一种是使用传统的统计学习和机器学习方法来学习用户兴趣。另一种是使用传统的信息检索方法（如余弦相似性）来生成推荐结果。但该方法存过推荐结果过于特定的问题。

协同过滤方法。协同过滤方法是利用与某一用户相似的其他用户对项目的意见来向用户做出推荐[25]。该方法可以进一步被分为基于用户和基于项目的协调过滤方法[26]。在基于用户的协调过滤方法中，用户将收到相似用户喜好的项目的推荐。在基于项目的协调过滤方法中，用户将收到与他们过去喜好的项目相似的项

目的推荐。用户或项目之间的相似可以通过基于皮尔森相关性的相似度[26]、基于约束的皮尔森相关性的相似度、基于余弦的相似度或基于调整后余弦相似度的计算得到。但该方法存在稀疏性、可扩展性和冷启动问题。

混合推荐方法。为了弥补传统推荐方法的不足且提升推荐表现，提出了将两种或以上的推荐技术结合起来的混合推荐方法[28]。根据 Burke 等人的研究[28]，有七种基本混合机制：加权、混合、切换、特征组合、特征增强、级联和元级别。现有的混合推荐技术中最常见的做法是将协同过滤方法与其他推荐方法结合起来，以避免冷启动、稀疏性和扩展性问题[1,29]。

优化推荐方法主要有基于计算智能的推荐方法（如人工神经网络和聚类技术等）、基于社交网络的推荐方法、基于上下文感知的推荐方法以及小组推荐方法。由于本研究在技术方法的侧重点是神经网络，下面将主要对基于计算智能的推荐方法中的人工神经网络的研究现状进行概述。

人工神经网络。受到了生物大脑结构的启发，人工神经网络是由相互连接的节点和赋有权重的连接组成的，可用于构建基于模型的推荐系统。Hsu 等人[38]使用了人工神经网络中的反向传播算法来训练一个三层神经网络，以应用于电视推荐系统。Christakou 等人[30]提出了基于内容的推荐方法和协同过滤方法的混合模型，以获得精确的电影推荐。其中系统的协同过滤方法使用了可以表示用户偏好的人工神经网络。

1.3.2 应用领域

随着推荐方法和技术的发展，越来越多的推荐系统得到开发和实现。在当今的大数据时代，应用研究仍是当前推荐系统研究的主要研究重点。推荐系统的应用已涉及电子商务、电子学习、电子图书馆和电子政务等。由于本研究在应用领域的侧重点是电子商务领域，下面将主要对电子商务领域中的推荐系统的研究现状进行概述。

在电子商务领域中，推荐系统会通过用户对项目的评分和标记来收集用户对项目的反馈信息。如在苹果商店中，客户可以通过给所购买的商品打 1 到 5 分之间的值来向苹果提供用户对商品的反馈信息。又如在电影评分网站 Movielens[27]上，用户能够使用简单的单词来给电影打上标签。对上述的评分和标签信息，协

同过滤方法[27]和社交标签分析会分别[31]或者是同时[32]使用以增强推荐表现。推荐系统除了需要获取用户对项目的反馈信息以外，还会收集用户的信息。许多大型的电子商务网站，例如 Amazon 和 eBay，已经使用推荐系统来帮助用户找到要购买的产品。这些网站会根据销量较高的卖家、买家人口统计信息或者是对买家过去购买行为的分析，来做出买家未来购买行为的预测。

在电子购物系统中，已针对不同的购物场景提出了不同的推荐系统。如用户对产品各组件有所要求的购物场景。在购买笔记本时，用户会考虑各个组件的性能，再做出最后的消费选择。曹和李[33]针对该购物需求开发了一个基于模糊的推荐系统。该推荐系统中会收集用户对各个组件的需求权重，并根据模糊相似度量模型向用户推荐最合适的项目。又如用户对推荐理由有所需求的购物场景。在线下购物时，买家在购买高价商品时会需要销售人员的推荐意见和推荐理由以支持他做出最后的购买决策。McGarthy 等人[35]开发了一个购物助手网站，利用复合评论来生成推荐理由，以满足用户对推荐理由的需求。该系统通过每个剩余案例与当前推荐案例的比较，得到它们的相对特征差异，从而生成了一组评价。其中性价比最高的商品就会最终被推荐给用户。又如捆绑购买或促销的购物场景。Garfinkel 等人[36]扩展了购物搜索引擎中的一次一件商品的搜索方法，以考虑多件商品的同时购买。该推荐系统有效地利用了在线商家提供的捆绑定价和促销交易来节约花费。

除了电子购物系统，推荐系统还被用于电子图书系统和电子音乐系统。Mooney 和 Roy 提出了基于内容的图书推荐系统。该系统使用了信息提取和机器学习算法来进行文本分类。朴素贝叶斯本文分类模型被用于训练从网页中提取到的数据，以构建图书特征和用户配置，并为目标用户找到最合适的图书。在音乐共享平台上，音乐社交社区由各种类型的音乐和用户关系组成。为了更好地利用丰富的社交信息，Tan 等人[34]提出了引入超图模型的音乐推荐方法来处理丰富的社交媒体信息。

随着移动电话和无线网络的发展，推荐系统不仅服务于网页用户，也向移动用户开放。Lawrence 等人[37]设计了移动个性化推荐系统来向超市购物者推荐新商品，并支持购物者通过该系统向商店下达订单。该系统使用了关联挖掘方法来确定产品类别之间的关系和产品的吸引力，并使用了聚类方法来根据支出记录的

相似度识别购物者群体，最后将该购物者群体限定的产品列表作为用户和产品匹配模型的输入，以生成最后对用户的推荐。

1.4 研究内容

在上一节中，我们对推荐系统的技术方法和应用领域进行了概述。在这一节中，我们将从这两方面来阐述本文研究内容。

首先，在应用领域方面，推荐系统目前已被广泛地应用于电子商务领域中。同时，在电子购物系统中，也已有针对不同购物环境所提出的不同推荐系统。但对于服饰商品的推荐，目前的推荐系统仅将与其他商品一起作为一个推荐问题处理，并未有研究单独地聚焦于服饰商品以及其所具有的季节性特征，并以模型在这一品类上的推荐效果作为评估模型有效性的标准。本文将聚焦于季节性服饰商品，并使用推荐模型的决策性和排序性评估指标来对模型效果进行评估。

其次，在技术方法方面，本文将在亚马逊评论数据上，基于传统推荐方法中的基于项目的协同过滤方法思想，使用优化推荐方法中的人工神经网络方法来计算项目与项目之间相似以及用户与项目之间的相似，从而完成推荐模型的构建。本文使用到的人工神经网络方法是 Item2vec、Bi-LSTM 和 self-attention 算法在上述三种算法中，Item2vec 算法是推荐问题中的项目相似的常用方法，而 Bi-LSTM 和 self-attention 算法常被应用于自然语言处理领域。本文将这三种算法结合，使用 Item2vec 算法来学习项目的嵌入向量表示形式从而得到项目与项目之间的相似，以及 Bi-LSTM 和 self-attention 算法来学习用户偏好从而得到用户与项目之间的相似。

1.5 论文结构安排

第一章是绪论。分析了本文的研究背景和意义、研究现状和研究内容。

第二章是模型理论基础。阐述了传统推荐模型理论，包括协同过滤方法、基于内容的推荐方法和混合推荐方法，以及本文模型搭建所需要的 Item2vec、Bi-LSTM 和 self-attention 算法。

第三章是数据分析与预处理。介绍了本文使用的数据情况，并分析了其中的

商品和用户数据。

第四章是模型构建。提出了本文的由项目嵌入和用户偏好组成的模型框架，并阐述了本文的输入准备、算法流程和推荐生成。

第五章是模型验证。阐述了推荐系统的评估指标，并分析了本文的模型效果。

第六章是系统设计与实现。阐述了本文的总体设计、数据库设计、模块设计、输入输出设计、界面设计和系统测试。

2 模型理论基础

上一章分析了推荐系统的研究现状，并提出了本文的研究内容。本文的推荐方法是基于传统推荐方法中的基于项目的协同过滤的思想展开的。首先通过 Item2vec 算法来学习项目的嵌入向量表示形式从而得到项目与项目之间的相似度，再通过 Bi-LSTM 和 self-attention 算法来学习用户偏好从而得到用户与项目之间的相似度，最终向用户推荐与其偏好最为相似的项目集。

这一章将阐述本文推荐方法中所涉及的传统推荐方法的理论基础，以及 Item2vec、Bi-LSTM 和 self-attention 算法的理论基础，为后续的模型构建做好理论准备。

2.1 传统模型理论

传统推荐方法包括协同过滤方法、基于内容的推荐方法和混合推荐方法[24]。下面将对这三种方法的模型理论进行概述。

2.1.1 基于内容的推荐方法

在基于内容的推荐方法中，用户c对项目s的喜爱程度是通过用户c对与项目s相似的物品$s_i \in S$的喜爱程度推测得到的。

基于内容的推荐方法所使用的模型思想来源于信息检索和信息过滤的领域。因此，大多基于内容的推荐方法会包含文本信息，例如文档、网页和新闻。在传统的信息检索方法外，基于内容的推荐方法会进一步包含用户口味、偏好和需求。这些档案信息可以显性地以问卷形式得到，也可以隐形地从交易行为中学习到。

基于内容的推荐方法主要适用于针对文本项目的推荐，通常使用关键词来表征文本项目内容。在信息检索中最著名的关键词指定方法是词频反比文档频率（TF-IDF）。假定 N 是可以向用户推荐的文档数，关键词k_j出现在其中的n_i个。此外，假定$f_{i,j}$是关键词k_i出现在文档d_j中的次数。那么，$TF_{i,j}$，即关键词k_i出现在文档d_j中的词频可以被定义为

$$TF_{i,j} = \frac{f_{i,j}}{max_z f_{z,j}} \quad (1)$$

其中$max_z f_{z,j}$指的是所有出现在文档d_j中的关键词k_z的词频$f_{z,j}$的最大值。然而，当一个关键词同时出现在很多文档中时，该关键词并不能很有效地用于区分相关文档和不相关文档。因此，通常将逆文档频率（IDF_i）与简单的词频（TF_i）结合起来。使用以下公式来定义关键词k_i的逆文本频率

$$IDF_i = log\frac{N}{n_i} \quad (2),$$

对于关键词k_i在文档d_j中的 TF-IDF 权重定义如下

$$w_{i,j} = TF_{i,j} \times IDF_I \quad (3)$$

对文档d_j的定义如下

$$Content(d_j) = (w_{1j}, \dots, w_{kj}) \quad (4)$$

在基于内容的推荐方法中，$ContentBasedProfile(c)$是用户c的档案信息。该档案信息可以通过分析用户之前看过或是评论过的项目得到。$ContentBasedProfile(c)$的表示形式是$(w_{c1}, \dots, w_{ck})$的权重向量，其中权重w_{ci}代表的是关键词k_i对用户c的重要性。

基于内容的推荐方法中的用户喜好函数$u(c,s)$通常被定义如下

$$u(c,s) = score(ContentBasedProfile(c), Content(s)) \quad (5)$$

其中，用户的$ContentBasedProfile(c)$和项目s的$Content(s)$可以用 TF-IDF 向量$\vec{w}_c$和$\vec{w}_s$表示。此外，$u(c,s)$可用向量$\vec{w}_c$和$\vec{w}_s$的余弦相似性得到。

尽管信息检索领域并不是专门地针对推荐进行研究，很多信息检索方法仍可以被用于推荐方法。如自适应过滤[13, 16]和阈值设置[8, 15]。自适应过滤[13, 16]通过在连续的文档流中一一观察文档来准确地逐步识别相关文档。阈值设置[8, 15]研究文档与给定序列的匹配程度来决定该文档是否与用户相关。在传统的基于信息检索的启发式方法以外，基于内容的推荐方法还会采用贝叶斯分类[7]和聚类、决策树和人工神经网络[7]等机器学习方法。他们与基于信息检索的方法的不同之处在于他们通过使用统计学习和机器学习方法在数据中学习到的一个模型来进行偏好预测。

基于内容的推荐方法在某些场景下无法做出有效的推荐。首先，它需要基于推荐对象的特征表示来进行推荐。为了获得充分的特征信息，推荐内容必须以计算机能够理解并自动处理的形式存在，或者需要用人工方式来获得每个内容的特征。然而，并不是所有领域的内容特征都可以被自动解析，而人工的方式也并不实用。其次是特征表示的不可区分性。如果两个不同的物品是通过相同的特征向量进行表示，那么将无法区分这两个特征向量。因此，基于内容的推荐方法可能无法决定是向用户推荐一个值得推荐的项目和一个不值得推荐的项目。另一点是推荐项目的单一性。由于基于内容的推荐方法会向用户推荐与其有过高评分的项目相似的其他项目，这样的推荐是非常单一的，缺乏多样性。最后一点是冷启动问题。基于内容的推荐方法需要用户已经有一定的行为记录，才能够理解用户的喜好，再向用户做出合理的推荐。因此，对一个没有或者拥有较少的行为记录的新用户而言，系统可能无法向其做出推荐。

2.1.2 协同过滤方法

与基于内容的推荐方法不同，协同过滤推荐方法通过其他用户对项目所做出的评分来预测特定用户对项目的偏好程度。更准确地说，用户c对项目s的偏好程度$u(c,s)$是在其他与用户c相似的用户$c_j \in C$对项目s的偏好程度$u(c_j,s)$上推测得到的。

协同过滤方法的算法可以被分为基于记忆的方法和基于模型的方法。

基于记忆的方法[3, 9, 12]基本上是基于用户之前有过评分的项目的全部集合来进行用户的评分预测。用户c对项目s的未知评分$r_{c,s}$是由其他用户对相同项目s的评分集合而计算得到的

$$r_{c,s} = aggr_{c' \in \hat{C}}^{r_{c',s}} \tag{6}$$

其中$\hat{C}$表示与用户c相似的并且对项目s有过评分的N个用户集。一些聚合函数如下

$$(a)\ r_{c,s} = \frac{1}{N}\sum_{c' \in \hat{C}} r_{c',s} \tag{7}$$

$$(b)\ r_{c,s} = k \sum_{c' \in \hat{C}} sim(c, c') \times r_{c',s} \tag{7}$$

$$(c)\ r_{c,s} = \bar{r_c} + k \sum_{c' \in \hat{C}} sim(c, c') \times \left(r_{c',s} - \bar{r}_{c'}\right) \tag{7}$$

其中 k 是归一化因子，通常被表示为$k = \frac{1}{\sum_{c' \in \hat{C}} |sim(c,c')|}$，而用户$c$的平均评分$\bar{r_c}$定义为

$$\bar{r_c} = \left(\frac{1}{|S_c|}\right) \sum_{s \in S_c} r_{c,s} \tag{8}$$

其中$S_c = \{s \in S | r_{c,s} \neq \emptyset\}$.

最常用于计算用户相似性的两种方法是相关性和余弦相似性。相似性计算是基于用户x与用户y都有过评分的项目集合S_{xy}上进行的，即$S_{xy} = \{s \in S | r_{x,s} \neq \emptyset \ \& \ r_{y,s} \neq \emptyset\}$. 在相关性方法中，皮尔森相似性会被用于计算相似性[9, 12]

$$sim(x, y) = \frac{\sum_{s \in S_{xy}} \left(r_{x,s} - \bar{r_x}\right)\left(r_{y,s} - \bar{r_y}\right)}{\sqrt{\sum_{s \in S_{xy}} \left(r_{x,s} - \bar{r_x}\right)^2 \left(r_{y,s} - \bar{r_y}\right)^2}} \tag{9}$$

在余弦相似性方法中，用户x和用户y被表示为两个 m 维空间中的向量，其中$m = |S_{xy}|$. 之后，用角的余弦来计算向量相似性

$$sim(x, y) = \cos(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{\left|\left|\vec{x}\right|\right|_2 \times \left|\left|\vec{y}\right|\right|_2} = \frac{\sum_{s \in S_{xy}} r_{x,s} r_{y,s}}{\sqrt{\sum_{s \in S_{xy}} r_{x,s}^2} \sqrt{\sum_{s \in S_{xy}} r_{y,s}^2}} \tag{10}$$

其中$\vec{x}, \vec{y}$表示向量$\vec{x}$和向量$\vec{y}$的点积。

在系统实现的时候，不同的推荐系统会使用不同的方法来尽可能地有效地实现用户相似计算和评分预测。常见的策略是提前计算所有的用户相似$sim(x, y)$，再定时地重新计算以进行相似性的更新。无论何时用户向系统进行推荐请求时，系统会使用已经计算好的相似结果来进行高效地推荐。

在相关性和余弦相似性以外，提出了很多能够改进性能的修改，如默认投票、反用户频率、案例放大和加权多数预测等。Sarwar[11]将相似性计算的两种方法应用于项目之间的相似性，而不再是用户之间的相似性。这个思想后被拓展用于

top-N 项目推荐[4]中。

而基于模型的方法在评论历史记录上进行模型的学习，再将该模型用于评分预测。例如[3]提出了一个概率方法

$$r_{c,s} = E(r_{c,s}) = \sum_{i=0}^{n} i \times \Pr(r_{c,s} = i | r_{c,s'}, s' \in S_c) \tag{11}$$

假定评分区间位于 0 到 n，考虑给定用户评分历史下的用户c对项目s做出某一评分的概率。[3]将聚类模型和贝叶斯网络用于估计该概率。在第一个模型中，相似的用户被聚为一个类。给定用户的类别，用户评分可以被假定为独立的。在第二个模中，每个项目被表示为贝叶斯网络中的一个节点，而每个节点的状态对应于该项目的可能评分值。此外，[2]提出了一个在机器学习框架中的协同过滤方法。

另一种提高协同过滤方法的表现的方式是通过技术方法来将用户的评分集进行挑选[14]，以达到去除噪音，去重和利用评分数据的稀疏性的目的。实证表明该方法能够提高推荐方法的准确性和有效性，也能够处理大规模数据集的问题。

由于协同过滤系统使用其他用户的评分来完成推荐，它不再有基于内容的推荐方法的部分缺点。它可以用于处理多种内容的推荐，也可以向用户推荐与其历史交互不同的项目。然而，协同过滤系统还是有其局限。首先是冷启动问题。由于新用户和新项目没有历史交互记录，系统无法对新用户进行项目的推荐，也无法将新项目推荐给用户。将基于内容的推荐方法和协同推荐方法进行结合的融合推荐系统可以解决带问题。其次是稀疏性问题。系统中已获得的评分数和需要预测的评分数相比是非常少的。同时，也依赖于一定数量的用户人群。一种解决方法是在进行用户相似性的计算时，考虑用户画像信息。[6]在使用了用户的性别、年龄、地区、教育和雇佣信息。另一种解决方法是通过应用关联检索框架和相关的扩展激活算法来探索历史交易和反馈数据中的消费者之间的传递性关联[5]，以解决稀疏性问题。还有一种方法是将降维技术[2,10]，即奇异值分解（SVD），用于减少稀疏等级矩阵的维数。

2.1.3 混合推荐方法

将协同过滤方法和基于内容的方法进行结合的融合推荐系统可以在一定程度上避免协同过滤方法和基于内容的方法的限制。有以下几种方法将上述两种方

法进行结合：

（1） 分别实现协同过滤方法和基于内容的方法，再结合两种方法的预测结果；

（2） 将基于内容的特征融入协同过滤方法中；

（3） 将协同过滤的特征融入基于内容的方法中；

（4） 构建一个将协同过滤方法和基于内容的方法同时纳入的通用模型。

2.2 研究模型理论

本文所构建的推荐方法是在传统推荐方法中的基于项目的协同过滤的基础上，使用 Item2vec、Bi-LSTM 以及 self-attention 算法来计算项目与项目之间的相似以及用户与项目之间的相似。下面将对这三种算法的模型理论进行概述。

2.2.1 Item2vec 算法

Item2vec 使用的是自然语言处理领域中 Word2vec 的模型框架，再在原有模型的基础上进行修改，使得修改后的模型能够用于计算协同过滤推荐算法中的项目之间的相似性。Item2vec 算法常被用于处理基于项目的协同过滤推荐系统中的项目表示。对于项目集合$S=\{S_1,S_2,S_3,\dots,S_m\}$，其中$S_i$表示的是用户$i$的项目序列$S_i=\{I_1,I_2,I_3,\dots,I_n\}$，Item2vec 算法的作用就是使用低维空间的项目向量来表示每一个项目。

Item2vec 算法所基于的 Word2vec 算法是由 Mikolov 等人[17]首先提出。该模型通过学习能够捕捉到同一个句子中的词和它邻近词的关系的词表示。$(w_i)_{i=1}^{k}$是词库$W=(w_i)_{i=1}^{w}$中的一个词序列，word2vec 旨在最大化以下公式

$$\frac{1}{K}\sum_{i=1}^{K}\sum_{-c\le j\le c,j\ne 0}\log p(w_{i+j}|w_i) \tag{12}$$

其中c是窗口大小，$p(w_{i+j}|w_i)$的原始形式是 softmax 函数。但由于$\nabla p(w_j|w_i)$的大小通常在 10^5 至 10^6，考虑到计算的复杂性，word2vec 通过以下方式来计算$p(w_j|w_i)$：

$$p(w_j|w_i) = \sigma(u_i^T v_j) \prod_{k=1}^{N} \sigma(-u_i^T v_k) \quad (13)$$

其中$\sigma(x) = 1/1 + \exp(-x)$是 sigmoid 激活函数，而$N$是在一个正样本中进行负样本的次数。同时为了解决词频的不平均问题，[17]还提出了子采样方法。给定一个词序列时，我们剔除掉概率为$p(discard|w) = 1 - \sqrt{\frac{\rho}{f(w)}}$的词$w$，其中$f(w)$是词$w$的频率，$\rho$是预先定义的阈值。子采样方法可以加快算法的学习过程，同时也提高了低频词的表示能力。最后，通过对公式（13）运用随机梯度上升的方式得到公式（13）中的u, v的估计值。

将 Word2vec 算法用于协同过滤推荐算法中时，Item2vec 算法[18]首先将每个项目看作是一个单词，同时将用户的交互项目序列看作是一个句子。之后，每个项目都会被嵌入到固定维度的向量中，而每个用户所拥有的交互项目序列都会互不相同。最后，通过对每个用户的交互项目序列进行嵌入的方式，得到一个固定维度的项目序列。距离上越接近的向量在嵌入空间中更为相似。通过词嵌入，我们得到以低维向量形式表示的项目序列。

2.2.2 Bi-LSTM 算法

双向长短期记忆网络（Bi-LSTM）是基于长短期记忆网络（LSTM）的一种双向循环神经网络。其中 LSTM 算法是一种用于处理时间序列的循环神经网络。而双向循环神经网络是有 Schuster 和 Paliwal 于 1997 年发明的。该神经网络将方向相反的两个隐藏层连接到同一输出，使得输出层可以同时从过去状态和将来状态获取信息。因其对于上下文序列信息的获取，双向递归神经网络常被用于语音识别、翻译、手写识别、蛋白质结构预测、词性标注、依赖解析、实体提取等任务中。因此 Bi-LSTM 算法适用于处理关注上下文信息的时间序列，常被用于自然语言处理中。

这里对 Bi-LSTM 结构进行具体阐述。首先定义一个拥有两个隐藏层的模型，其中下层会向上层提供信息。在网络结构中，前一个时间步生成一组参数，并在随后的时间步中将这组参数传递给同一 Bi-LSTM 层中的中间神经元。同时，中间神经元需要在时间步t从 Bi-LSTM 隐藏层的上一层接受两组相关的参数。模型中

每个隐藏层均从两个方向，即从左至右和从右至左，来进行序列的输入。如公式（14）所示，在同一时间步t，位于$r-1$层的 Bi-LSTM 层的输出都会作为位于r层中间神经元的输入。在模型训练的每个时间步，通过连接所有的输入参数，由隐藏层传播产生结果。最后的隐藏层产生最终输出P。

$$(a)\ \overrightarrow{h_t}^{(r)} = f\left(\vec{A}^{(r)}\overrightarrow{h_t}^{(r-1)} + \vec{B}^{(r)}\vec{h}_{t-1}{}^{(r)} + \vec{z}^{(r)}\right) \tag{14}$$

$$(b)\ \overleftarrow{h_t}^{(r)} = f\left(\overleftarrow{A}^{(r)}\overleftarrow{h_t}^{(r-1)} + \overleftarrow{B}^{(r)}\overleftarrow{h}_{t+1}{}^{(r)} + \overleftarrow{z}^{(r)}\right) \tag{14}$$

$$(c)\ \vec{p} = concat\left(\overrightarrow{h_t}^{(r)}, \overleftarrow{h_t}^{(r)}\right) \tag{14}$$

其中$\vec{A}^{(r)}$，$\vec{B}^{(r)}$和$\vec{z}^{(r)}$是权重矩阵，且和在模型r层正向传播中生成的向量发生偏移；$\overleftarrow{A}^{(r)}$，$\overleftarrow{B}^{(r)}$和$\overleftarrow{z}^{(r)}$也是权重矩阵，且和在模型r层反向传播中生成的向量发生偏移；$\vec{p}$是输出向量；$\overrightarrow{h_t}^{(r)}$和$\overleftarrow{h_t}^{(r)}$分别是对过去和将来的中间表示，用于区分输入向量。

模型r层的传播是基于前一层当前时刻的隐藏状态$\overrightarrow{h_t}^{(r-1)}$和当前层前一时刻的隐藏状态$\vec{h}_{t-1}{}^{(r)}$来计算当前层这一时刻的隐藏状态$\overrightarrow{h_t}^{(r)}$的正向更新。相反，反向传播需要当前层的隐藏状态$\overleftarrow{h_t}^{(r-1)}$当前层的未来状态$\overleftarrow{h}_{t+1}{}^{(r)}$来计算隐藏状态$\overleftarrow{h_t}^{(r)}$的反向更新。之后，使用用来连接前向和反向隐藏表示的$concat(\overrightarrow{h_t}^{(r)}, \overleftarrow{h_t}^{(r)})$来计算每个隐藏表示。最后，通过全连接层，由隐藏层的最后一层输出目标向量。

2.2.3 self-attention 算法

注意力机制（attention）由 DeepMind 为处理图像分类提出。该机制可以让神经网络在执行预测任务时可以更多地关注输入中的相关部分，而更少关注不相关的部分。而自注意力机制（self-attention）是注意力机制的变体，其减少了对外部信息的依赖，更擅长捕捉数据或特征的内部相关性。目前被广泛地应用于自然语言处理领域。

这里对 self-attention 结构进行具体阐述。首先，对于一个给定的目标项目，该机制会遍历所有编码器的状态，并将源项目的状态与目标项目的状态进行

比较，即源项目和目标项目的关系，从而在编码器中对每个状态进行打分。之后，在给定目标项目的状态后，使用 softmax 函数对所有分数进行均一化处理后以得到一个概率分布。最后，我们可以从该分布中得到项目权重。

在上述过程中，项目特征表示会从d维空间映射到z维空间。关系映射如下所示：

$$(a)\ I_z = f_{Relu}(WI_d + b) \tag{15}$$

$$(b)\ A = softmax(I_z W(I_d)^T) \tag{15}$$

$$(c)\ I_z = AI_z \tag{15}$$

其中W是权重矩阵，b是偏差，I_z是项目特征嵌入后得到的特征表示向量。公式（15a）表示项目特征从d维空间映射到z维空间。公式（15b）在d维空间中计算所有项目对在z维空间的每一个项目的贡献权重。该模型在训练过程中使用损失函数对权重矩阵W进行自动调整。公式（15c）通过 softmax 函数均一化矩阵A。在z维空间中的项目都被赋予了权重。加权后，z维空间中每个项目的特征表示都由其自身和与之相关的所有项目共同表示。最后输出的I_z是通过 self-attention 机制加权后的项目特征。

2.3 本章小结

本章节首先从概念和理论两方面阐述了传统推荐方法中的基于内容的推荐方法、协同过滤方法以及混合推荐方法。之后，阐述了本文要用于计算协同过滤方法中项目与项目之间的相似和用户与项目之间的相似的 item2vec，Bi-LSTM 和 self-attention 算法。本章节为第四章的模型构建过程提供了理论基础。

3 数据分析与预处理

上一章阐述了本文推荐方法中所涉及的理论基础。这一章将分析本文所使用的亚马逊评论数据集，为下一章的模型构建做好数据准备。

3.1 数据集介绍

亚马逊评论数据集是由加利福利亚大学圣迭戈分校管理的线上公开数据集。该数据集收录了 1996 年至 2018 年的商品评论信息，并由商品数据和评论数据两部分组成。其中商品数据提供了商品的详细信息，包括标题、图片、价格、描述、品牌、特征、品类、排名等信息。而评论数据提供了评论的打分、时间、用户、商品、详细评论和评论摘要的信息。本文使用商品数据中的标题、图片、价格、描述、品牌、品类和排名信息以及完整的评论数据来构建系统。此外，为了聚焦季节性商品的推荐系统的分析与设计，本文使用该数据集中的 Amazon Fashion 子类别中的服装、鞋子和珠宝品类，共计 17 万条商品数据和 75 万条评论数据。

在对上述数据进行探索时，我们发现商品数据和评论中的用户数据存在一定的数据问题。商品数据在某些数据列上具有很高的数据缺失率，如表 3-1 所示。评论中的用户名称有近 12%为“Amazon Customer”，但仍可以以用户 ID 区分这些用户。由于模型的训练并不依赖于上述提到的数据列，因此该数据问题不会影响到模型的效果，但在系统的实现环节无法对商品信息做出完整的展示。

表 3-1 商品数据的缺失问题

数据列	缺失量	缺失率
商品 ID	0	0%
商品名称	0	0%
商品图片	0	0%
商品品牌	45018	25%
商品品类	0	0%
商品排名	0	0%
商品描述	164559	92%
商品价格	161611	91%

3.2 商品数据分析

本文将使用服装、鞋子和珠宝品类的商品数据来完成推荐系统的搭建。该品类的一大特点就是季节性特点。用户对服装的偏好会明显地随着季节的变化而改变。我们将从商品的评论时间分布的角度来验证数据的季节性特征。

首先从评论数据中提取到每个商品的评论总数目。商品的评论数目分布如图 3-1 所示。80%的商品评论数据小于等于 3 条。

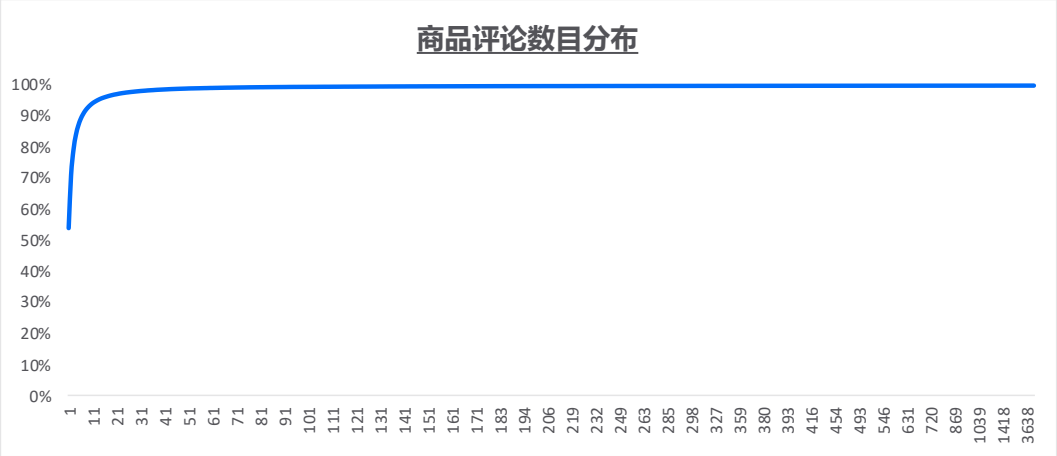

图 3-1 商品评论数目分布

之后统计商品在四个季度下的评论数目，并在该数据上训练聚类模型。这里使用的是 K-means 聚类算法。我们通过肘部法则选择出最合适的聚类数目。如图 3-2 所示，我们应将聚类数目定为 4。

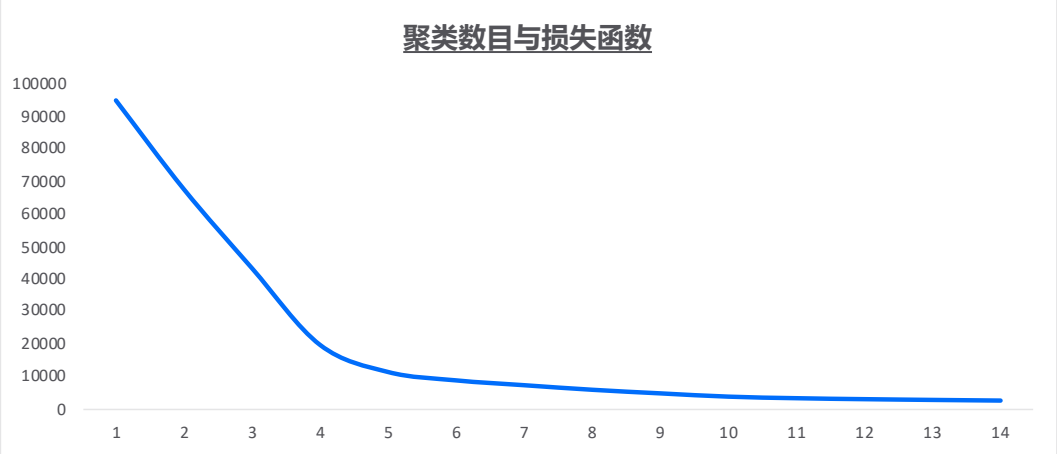

图 3-2 聚类数目与损失函数

最后，我们通过聚类模型的聚类中心来找到每类的商品评论分布的特征，从而说明每类商品的不同的季节性特征。四个类在各季度的评论数目分布如表 3-2

所示。聚类 1、聚类 2、聚类 3 和聚类 4 的评论分别集中在冬季、夏季、春季、秋季，具有非常明显的季节性特征。

表 3-2 类的评论数目分布

聚类	第一季度	第二季度	第三季度	第四季度
1	10%	3%	8%	79%
2	10%	72%	11%	8%
3	86%	3%	7%	4%
4	3%	3%	91%	3%

3.3 用户数据分析

这里我们将对数据集中的用户数据进行分析。我们从评论数据中提取到每个用户的评论总数目。用户的评论数目分布如图 3-3 所示。由于过少的交互行为无法提供充分的用户偏好，在下面的模型搭建过程中，我们将基于评论数目大于等于 3 的用户交互序列来进行模型的搭建。

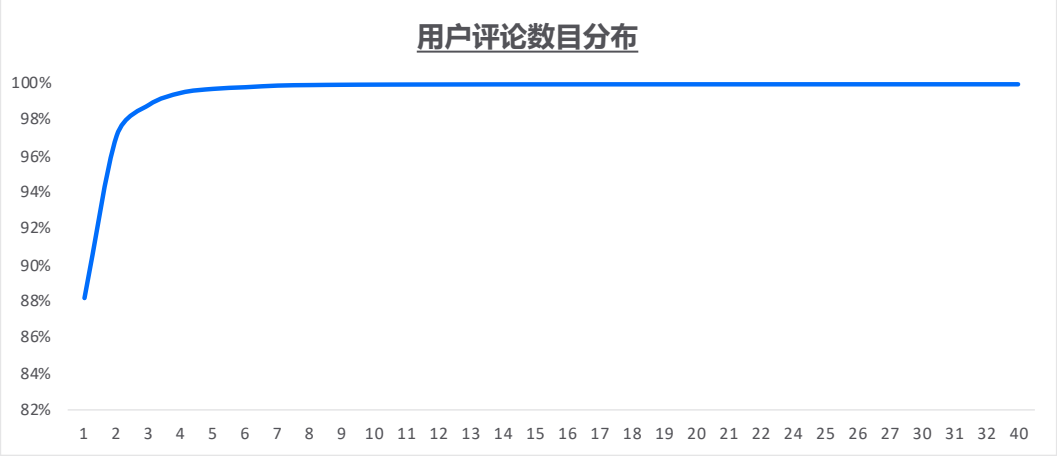

图 3-3 用户评论数目分布

3.4 本章小结

本章节首先介绍了数据集，并阐明了其中的数据缺失问题。之后分析了数据集中的商品和用户数据，阐明了数据集中的商品季节性和用户评论分布。本章节为第四章的模型构建过程提供了数据基础。

4 模型构建过程

在第二章的模型理论基础和第三章的数据分析与预处理的基础上，我们将在这一章提出本文的模型框架，并对模型构建过程中的输入准备、算法流程和推荐生成环节所做工作进行说明。

4.1 模型架构

本文的推荐方法是基于传统推荐方法中的基于项目的协同过滤的思想展开的。首先通过 Item2vec 算法来学习项目的嵌入向量表示形式从而得到项目与项目之间的相似度，再通过 Bi-LSTM 和 self-attention 算法来学习用户偏好从而得到用户与项目之间的相似度，最终向用户推荐与其偏好最为相似的项目集。

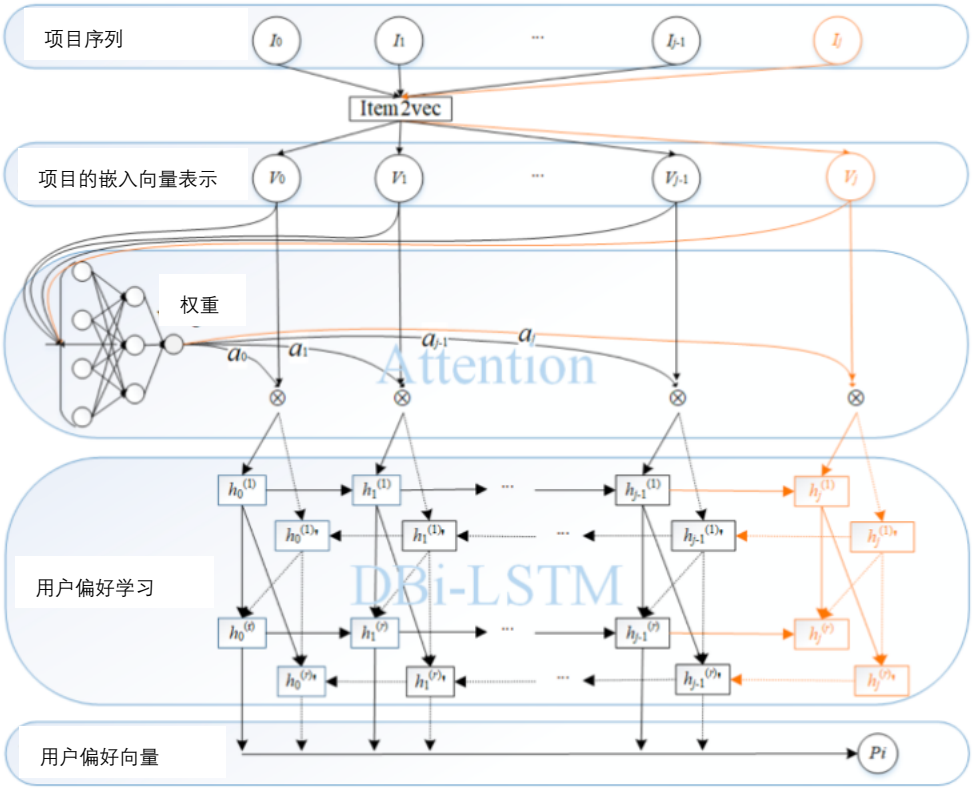

图 4-1 研究模型结构

该推荐方法的实现将通过图 4-1 所示的模型架构完成。模型分为项目嵌入模型和用户偏好模型两部分。

第一部分是使用 Item2vec 算法的项目嵌入模型。该模型将用于学习项目与项目之间的相似。项目嵌入模型接受到项目序列后，使用 Item2vec 算法来学习

该项目序列中某一项目与其邻近项目的关系，最后输出每个项目的低维嵌入向量表示。距离上越接近的向量在嵌入空间中更为相似。

第二部分是使用 Bi-LSTM 和 self-attention 算法的用户偏好模型。该模型用于学习用户与项目之间的相似。用户偏好模型接受到由上一层项目嵌入模型输出的项目的低维嵌入向量表示后，首先使用 self-attention 算法来对输入中的相关部分给予更多的关注而对不相关的部分给予更少的关注。之后将上一层项目嵌入模型输出的项目嵌入向量表示和 self-attention 模型输出的项目权重输入至 Bi-LSTM 算法，并通过该算法来学习用户对项目序列中的项目的偏好程度。最后根据学习到的用户偏好向量来完成用户对项目的偏好预测。

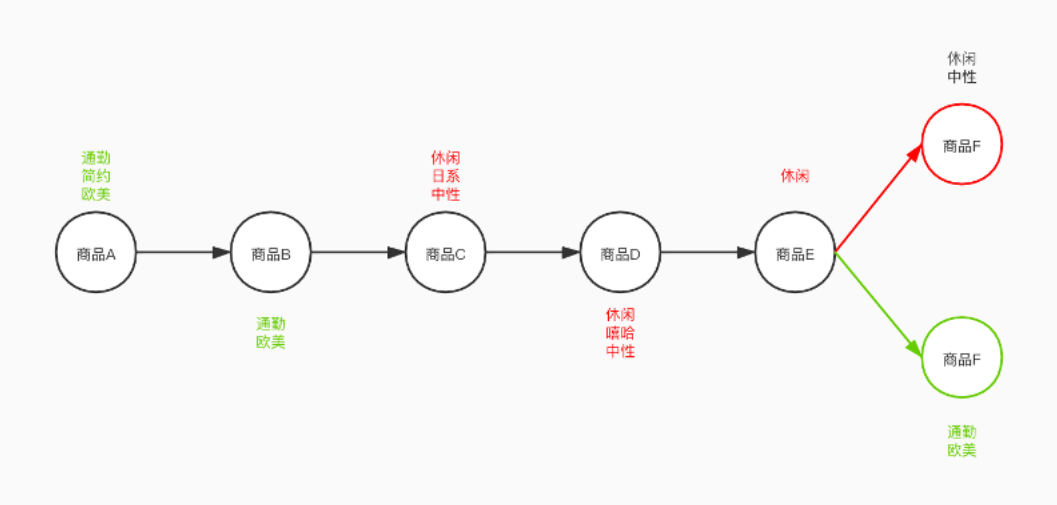

图 4-2 商品序列推荐

在第二部分的用户偏好模型中，Bi-LSTM 算法是实现用户偏好学习的主要算法，因为其所具有的双层结构使其适合处理关注上下文信息的序列信息。而 self-attention 的加入可以使模型更多地学习到用户偏好的权重转移，从而提升模型表现。在现实生活中，用户偏好不是固定的。当用户在一段时期内的交互行为集中在某类产品时，并不代表用户之后的偏好仍会遵从这一时期的偏好行为。如图 4-2 所示的序列推荐，某一用户的历史交互项目序列是“商品 A、商品 B、商品 C、商品 D、商品 E”。推荐算法目标是预测用户的下一件喜好商品。当算法将过多的重点放在用户近期的交互项目上时，很大概率会向用户推荐休闲中性风格的服饰商品，例如商品 F。然而，实际的交互记录显示用户选择购买了商品 G。这样的选择可能是受到了出现在序列起始位置的商品 A 和商品 B 的影响，而与商

品 C、商品 D 和商品 E 并不相关。因此，推荐算法需要学习一个序列中相关信息和不相关信息，并对相关信息给予更多的关注，而对不相关信息给予更少的关注。self-attention 算法的加入可以帮助用户偏好模型完成这一学习目标。

4.2 输入准备

这一节将具体说明第一节模型框架中的数据输入部分。本系统要求的输入形式是与用户有过交互的项目序列。为了满足这一要求，需要从亚马逊评论数据集中的评论信息中获取到用户项目序列。

表 4-1 用户 Tanya C 的评论列表

评论时间	评论用户	评论项目	项目标签
2012-04-20	Tanya C	Great Jewelry Silver Plated Snake Chain Necklace Italian Style Shimmering High Polish, 24-Inches	项链，镀银，蛇链
2015-02-07	Tanya C	Jelinda Lady Crochet Knitted Lace Trim Boot Cuffs Knee Caps Socks	袜子，及膝，蕾丝边
2015-02-28	Tanya C	Jelinda Lady Crochet Knitted Lace Trim Boot Cuffs Knee Caps Socks	袜子，及膝，蕾丝边
2017-02-28	Tanya C	Jelinda Lady Winter Crochet Knitted Lace Trim Boot Cuffs Knee Caps Socks (White)	护膝袜，蕾丝边
2017-05-03	Tanya C	Fedi Apparel Women's Strip Bodycon Dress V Neck Striped Loose Shirt Plus Size	女装，连衣裙，衬衫，V 领

这里我们将以用户 Tanya C 的用户项目序列生成方式为例来说明本文所做的用户项目序列获取工作。表 4-1 为用户 Tanya C 的评论对商品的按时间先后排序的评论数据。将评论数据转换为用户项目序列形式后，用户 M.Cruz 所对应的项目序列为{Great Jewelry Silver Plated Snake Chain Necklace Italian Style Shimmering High Polish, 24-Inches，Jelinda Lady Crochet Knitted

Lace Trim Boot Cuffs Knee Caps Socks，Jelinda Lady Crochet Knitted Lace Trim Boot Cuffs Knee Caps Socks，Jelinda Lady Winter Crochet Knitted Lace Trim Boot Cuffs Knee Caps Socks (White)，Fedi Apparel Women's Strip Bodycon Dress V Neck Striped Loose Shirt Plus Size}。

此外，为了验证模型在不同长度的用户项目序列上的表现，本文还按照项目序列的长度对数据集进行分组。对于项目序列长度小于某一组所要求的项目序列长度的数据，我们会对该序列进行填充。对于项目序列长度大于某一组所要求的项目系列长度的数据，我们会选择序列前段的满足组别长度要求的子序列。在训练模型时，模型的目标是在项目序列长度为i的组中，使用每个序列前段长度为$i-1$的项目序列来预测用户与第i个项目的交互行为。模型效果评估是在原始项目序列中第i个项目后的序列段上开展的。

这里我们将举例说明本文所做的数据分组工作。如表 4-2 所示，数据集中现有 6 个用户的项目序列信息，其中用户 A 和 B 的项目序列长度为 3，用户 C 和 D 的项目序列长度为 4，用户 E 和 F 的项目序列长度为 5。

表 4-2 用户项目序列表

用户	项目序列
A	A,B,C
B	A,C,D
C	A,B,C,D
D	A,B,D,F
E	A,B,C,D,E
F	A,B,C,D,F

按照项目序列长度分组后的数据集如表 4-3 所示。首先是序列长度为 3 的数据组。由于用户 A 和 B 项目序列的原始长度等于 3，因此直接将其所对应的项目序列放入数据组。由于其余用户的项目序列的原始长度大于 3，因此选择序列前段长度为 3 的子序列放入数据组中。其次是序列长度为 4 的数据组。由于用户 A 和 B 项目序列的原始长度小于 4，因此需要对序列进行填充后再放入数据组。由于其余用户的项目序列的原始长度均大于等于 4，因此按照上一组的处理逻辑将序列放入数据组中。最后是序列长度为 5 的数据组。按照上两组的处理逻辑将序

列放入数据组中。

表 4-3 用户项目序列数据组

项目序列长度	数据组
3	{A：A,B,C}, {B:A,C,D}, {C:A,B,C}
	{D:A,B,D}, {E:A,B,C}, {F:A,B,C}
4	{A：0,A,B,C}, {B:0,A,C,D}, {C:A,B,C,D}
	{D:A,B,D,F}, {E:A,B,C,D}, {F:A,B,C,D}
5	{A：0,0,A,B,C}, {B:0,0,A,C,D}, {C:0,A,B,C,D}
	{D:0,A,B,D,F}, {E:A,B,C,D,E}, {F:A,B,C,D,F}

通过上述操作，我们生成了用户项目序列组，并将其输入到后续的算法中。

4.3 算法流程

这一节将具体说明第一节模型框架中的模型算法部分。如表 4-4 所示，系统共有两层模型，分别是项目嵌入模型和用户偏好学习模型。项目嵌入模型将通过 Item2vec 算法来学习项目的嵌入向量表示形式从而得到项目与项目之间的相似度。用户偏好学习模型将通过 Bi-LSTM 和 self-attention 算法来学习用户偏好从而得到用户与项目之间的相似度，最终向用户推荐与其偏好最为相似的项目集。下面将分别说明本文在这两个模型层所做的工作。

表 4-4 模型算法流程

模型层	模型目的	算法名称	算法实现
项目嵌入	学习项目与项目之间的相似	项目嵌入	Item2vec
用户偏好学习	学习用户与项目之间的相似	偏好权重更新	self-attention
		用户偏好学习	Bi-LSTM

首先是项目嵌入模型。在输入准备环节中，系统已从用户数为m的用户评论数据中得到用户项目序列$S = (S_1, S_2, S_3, \dots, S_m)$，其中$S_i$代表第$i$个用户的项目序列。通过 Item2vec 算法，模型会学习到每个项目的低维嵌入向量表示$\vec{V}_i$，并将用户项目序列更新为$S_i = \{\vec{V}_1, \vec{V}_2, \vec{V}_3, \dots, \vec{V}_n\}$，其中$n$表示为与用户$i$有过交互行为的

项目数为n个，$\vec{V}_j$表示第j个项目的低维嵌入向量。本系统所使用的 Item2vec 算法参数如表 4-5 所示。

表 4-5 Item2vec 算法参数设置

参数名称	参数意义	参数值
vector_size	单词向量的维度	32
window	句子中当前词和预测词之间的最大距离	3
min_count	忽略出现次数低于此阈值的所有单词	2
workers	参与模型训练的进程数	12
sg	1 为 skip-gram，否则为 CBOW	1
hs	1 为 softmax；0 且参数 negative 为非零时为负采样	0
negative	>0 为负采样，数值表示多少噪声词会被提取；0 为非负采样	5
sample	用于决定对于高于何种频率的词进行随机下采样的阈值	1e-3
iter	循环次数	1000

其次是用户偏好学习模型。系统已从上一层项目嵌入模型中学习到以嵌入向量形式表示的项目序列。通过 self-attention 算法，模型会学习到项目序列中的相关部分和不相关部分，并输出项目权重。再通过 Bi-LSTM 算法，模型会从项目序列和项目权重中学习到用户偏好向量。本系统所使用的用户偏好模型摘要如表 4-6 所示。

表 4-6 用户偏好模型摘要

模型层	输出形状
Embedding	(None, None, 32)
self-attention	(None, None, 64)
Bidirectional	(None, None, 64)
Dense	(None, None, 6377)
flatten	(None, None)
activation	(None, None)

4.4 推荐生成

这一节将具体说明第一节模型框架中的推荐生成部分。最后系统的推荐生成是基于完整的由项目嵌入和用户偏好构成的模型。但为了说明每一模型层对最后推荐结果所起的作用，同时也为下一章模型实证分析做下铺垫，这一节将首先说明如何使用单独的项目嵌入来生成推荐结果，再说明如何使用项目嵌入模型层和只包含 Bi-LSTM 算法的用户偏好模型层来生成推荐结果，最后说明如何使用完整的由项目嵌入模型层以及包含 Bi-LSTM 和 self-attention 算法的用户偏好模型层来生成推荐结果。

表 4-7 项目嵌入对用户 Tanya C 的推荐结果

推荐序列	推荐项目	项目标签
1	eVogues Plus Size Glitter Layered Look Poncho Top Brown - 1X Shimmering High Polish, 24-Inches	雨披，闪粉
2	Paprika Ruler! By Lace Up Hidden Platform High Heel Ankle Bootie and Top Cushioning, black faux suede, 7.5 M	踝靴，蕾丝，鹿皮绒
3	Polka Dot Chic Womens Pleated Loose Blouses Tops Crewneck Shirts M White	女士，衬衫，圆点，百褶
4	Red Kap Men's Double Knee No-Scratch Shop Pants	男士，裤，无痕
5	Zacr Lady Winter Crochet Kanitted Lace Trim Boot Cuffs Knee Caps Leg Warmers Socks (Black)	护膝袜，蕾丝边

首先是单独使用项目嵌入方法的推荐结果生成方式。项目嵌入方法在得到用户项目序列作为模型的输入后，通过 Item2vec 模型学习到每个项目的低维向量表示。在得到以低维向量表示的项目信息后，本文将计算两个向量间的余弦相似性，从而得到项目与项目的相似矩阵。另外，根据用户项目序列，我们还可以得到用户对项目的实际偏好矩阵。基于用户对项目的交互矩阵和项目与项目的相似

矩阵，我们可以对上述两个矩阵进行点积，最终得到用户对项目的预测偏好矩阵。最终，通过对该矩阵的遍历，得到对每个用户而言偏好程度从高到底的且与用户还没有产生过交互记录的前 k 个项目，即为最终向用户推荐的项目列表。

这里我们将以用户 Tanya C 为例来查看单独使用项目嵌入方法生成的推荐结果。Tanya C 在历史交互序列中的项目标签如表 4-1 所示，为{项链，镀银，蛇链；袜子，及膝，蕾丝边；袜子，及膝，蕾丝边；护膝袜，蕾丝边；女装，连衣裙，衬衫，V 领}。单独使用项目嵌入方法对用户 Tanya C 的推荐结果如表 4-7 所示。推荐结果中具有较强解释性的推荐为 Paprika Ruler! By Lace Up Hidden Platform High Heel Ankle Bootie and Top Cushioning, black faux suede, 7.5 M（踝靴，蕾丝，鹿皮绒），Polka Dot Chic Womens Pleated Loose Blouses Tops Crewneck Shirts M White（女士，衬衫，圆点，百褶），Zacr Lady Winter Crochet Kanitted Lace Trim Boot Cuffs Knee Caps Leg Warmers Socks (Black)（护膝袜，蕾丝边）。但其仍与用户交互序列中的标签的重合度较低，且存在其他解释性较低的推荐出现在推荐结果中。

表 4-8 加入 Bi-LSTM 算法后对用户 Tanya C 的推荐结果

推荐序列	推荐项目	项目标签
1	Leg Avenue Women's Athletic Thigh High Socks	女士，高筒袜，运动
2	Leg Avenue Women's Nylon Striped Stockings	女士，袜子，尼龙，条纹
3	Cnlinkco Women's High Waisted Double Slits Maxi Skirt	女士，长裙，高腰，开衩
4	Jelinda Lady Crochet Knitted Lace Trim Boot Cuffs Knee Caps Socks	袜子，及膝，蕾丝边
5	Jelinda Lady Crochet Knitted Lace Trim Boot Cuffs Knee Caps Socks	袜子，及膝，蕾丝边

其次是使用项目嵌入模型层和只包含 Bi-LSTM 算法的用户偏好学习模型层的推荐结果生成方式。加入只包含 Bi-LSTM 算法的用户偏好学习模型后的算法流程是将项目嵌入方法学习到的项目低维向量表示作为模型的输入，再通过 Bi-LSTM 算法学习到用户偏好向量。根据训练后的用户偏好学习模型对于用户项目

序列中的下一个可能出现的项目的预测，即可得到推荐结果。其中，模型的预测为下一个项目属于项目集中各个项目的概率。在该概率分布中，概率最大的且与用户还没有产生过交互记录的前 k 个项目，即为最终向用户推荐的项目列表。

这里我们仍以用户 Tanya C 为例来查看使用项目嵌入模型层和只包含 Bi-LSTM 算法的用户偏好学习模型层生成的推荐结果。该推荐结果如表 4-8 所示。推荐结果中的项目标签与该用户的历史项目序列中的项目标签有很高的重合度。但受到用户频繁购买袜子的影响，推荐结果也多为袜子产品，缺乏对用户以往偏好的学习。

表 4-9 完整模型对用户 Tanya C 的推荐结果

推荐序列	推荐项目	项目标签
1	iecool New Sexy Romper Jumpsuit Deep V - Neck Floral Prints Club Trousers Small	连体裤，性感，花卉
2	ZANZEA Women Lace Slim Sheer Long Sleeve Slim Fit Chiffon Tops Blouse Shirt US 4 White, Medium	女士，衬衫，蕾丝
3	Jelinda Women Peacock Feather Bridal Wedding Hair Clip Headpiece Hair Accessory	女士，发饰
4	HuntGold 1X Artificial Rhinestone Diamond Crystal Pendant Chain Necklace Lovers Gift(purple)	项链，水晶
5	Long Way Women Silver Plated Snake Chain Pink Flower Maruno Glass Bead Heart Clasp Charm Bracelet	女士，手链，镀银，蛇链

最后是使用完整的由项目嵌入模型层以及包含 Bi-LSTM 和 self-attention 算法的用户偏好模型层的推荐结果生成方式。加入 self-attention 算法后，用户偏好模型层将首先使用 self-attention 算法学习到项目序列中各项目的权重，再将该项目权重与从项目嵌入模型层中获取到的以嵌入向量形式表示的项目序列输入到 Bi-LSTM 模型中，最后通过 Bi-LSTM 算法学习到用户偏好向量。这一部分的推荐结果生成方式与上一部分相同。

这里我们继续以用户 Tanya C 为例来查看完整的由项目嵌入模型层以及包含 Bi-LSTM 和 self-attention 算法的用户偏好模型层生成的推荐结果。该推荐结果如表 4-9 所示。推荐结果中的项目标签与该用户的历史项目序列中的项目标签有很高的重合度，并且在用户对袜子的偏好以外，还学习到了用户前期对镀银蛇链项链的偏好以及近期对 V 领连衣裙的偏好。

4.5 本章小结

本章节首先确定了本文的模型框架是由 Item2vec 算法构成的项目嵌入层以及由 Bi-LSTM 和 self-attention 算法构成的用户偏好层组成的，其次明确了模型的输入形式为从评论数据中提取的用户项目序列，之后阐明了项目嵌入层将从用户项目序列中学习项目的低维向量表示而用户偏好层将从由用户的低维向量表示构成的用户序列中学习用户的偏好向量，最后定义了模型的输出是由用户偏好层对于用户项目序列中的下一个可能出现的项目的概率预测得到的。

5 模型实证分析

上一章阐述了本文的模型框架以及在各环节所做工作。这一节将使用推荐系统的决策性和排序性评估指标对模型的推荐效果进行评估。

5.1 评估指标

推荐系统的质量既取决于推荐结果与用户的相关性，也取决于推荐结果的新颖性。如果推荐结果与用户高度相关，但推荐结果并不能使用户产生兴趣，那么这样的推荐结果是不能让用户与项目有后续的交互行为的发生。反之也会是一样的结果。因此推荐系统的评估指标常被分为决策性指标和排序性指标。

5.1.1 决策性指标

决策性指标主要用于评估推荐系统如何帮助用户做出更好的决定，例如是否可以帮助用户选择好的项目而避免选择坏的项目。这里我们使用召回率和准确率来对推荐系统的决策性质做出评估。

召回率的计算方式如公式（16）所示。它关注的是推荐系统是否向用户推荐正确的项目，可以用于反映推荐系统的查全率。

$$召回率 = \frac{被推荐的项目数}{所有相关的项目数} \tag{16}$$

例如，我们向用户 A 推荐了 5 个项目。推荐结果为项目 A、项目 B、项目 C、项目 D、项目 E，其中项目 A、项目 C、项目 E 为与用户 A 相关的项目。而与用户 A 的相关项目共有 10 个。此时召回率为 0.3.

准确率的计算方式如公式（17）所示。它关注的是推荐系统是否向用户推荐相关的项目，可以用于反映推荐系统的查准率。

$$准确率 = \frac{相关的项目数}{所有被推荐的项目数} \tag{17}$$

沿用在召回率中所举的例子。在向用户 A 推荐的 5 个项目中相关的项目有 3 个。此时准确率为 0.6.

5.1.2 排序性指标

排序性指标主要用于评估系统的排序质量。系统需要把与用户相关的项目放在列表的前列。排序性指标主要可以反映推荐系统两个方面的能力，即对推荐项目的位置选择能力以及对用户相对偏好的学习能力。这里我们使用平均精度来对推荐系统的排序性指标做出评估。

在对推荐系统的排序质量进行评估时，我们需要对处在推荐列表排序靠前的项目的推荐质量给予更多的权重。因此我们需要一个能够随着项目在推荐列表中所处位置而做出权重调整的误差权重值来处理误差信息。该误差权重值会随着项目在列表中位置的降低而逐步降低权重。

平均精度就是基于上述目标提出的一个评估指标。首先它在每一位用户的推荐列表上，从头到尾逐渐扩大子列表，并在子列表上计算推荐结果的准确率。之后根据子列表上计算得到的准确率，得到用户维度的平均准确率，最后得到系统维度的平均准确率，即平均精度。例如用户 A 的推荐列表为{商品 A，商品 B，商品 C，商品 D，商品 E}，其中商品 A、商品 C、商品 D 为相关商品。对于商品 A 而言，$P = 1/1$；对于商品 B 而言，$P = 2/3$；对于商品 C 而言，$P = 3/4$. 用户 A 的推荐效果的平均精度为$\frac{1+\frac{2}{3}+\frac{3}{4}}{3} = 0.8$. 另一位用户 B 的推荐列表为{商品 B,商品 D，商品 G，商品 H，商品 J}，其中商品 H、商品 J 为相关商品。对于商品 H 而言，$P = 1/1$；对于商品 D 而言，$P = 1/4$；对于商品 J 而言，$P = 2/5$. 用户 B 的推荐效果的平均精度为$\frac{1+\frac{2}{5}}{2} = 0.325$. 假设推荐系统中只有这两位用户，那么系统的推荐效果的平均精度为$\frac{0.8+0.325}{2} = 0.56$.

5.2 效果验证

为了说明各模型层中各算法对系统推荐效果的作用，这一节将首先验证使用单独的项目嵌入下的系统推荐效果，再验证使用项目嵌入模型层和只包含 Bi-LSTM 算法的用户偏好模型层下的系统推荐效果，最后验证使用完整的由项目嵌入模型层以及包含 Bi-LSTM 和 self-attention 算法的用户偏好模型层下的系统推荐结果。

5.2.1 对 Item2vec 算法的效果验证

项目嵌入模型层的目的是通过 Item2vec 算法来学习到项目与项目之间的相似度。对于本系统所聚焦的季节性服饰商品而言，我们需要 Item2vec 模型学习到季节性的相似。对此，我们首先对某一商品的推荐结果中的商品是否与该商品的季节性特征相同做了验证。如图 5-1 所示，43%的推荐结果中有半数及以上的商品与该商品具有相同的季节性特征。仅 6%的推荐结果中没有任何与该商品有相同季节性特征的商品。由于推荐系统除了学习季节性的相似以外，仍需对项目内容的相似进行学习，因此这里模型的季节性学习表现是合理的。

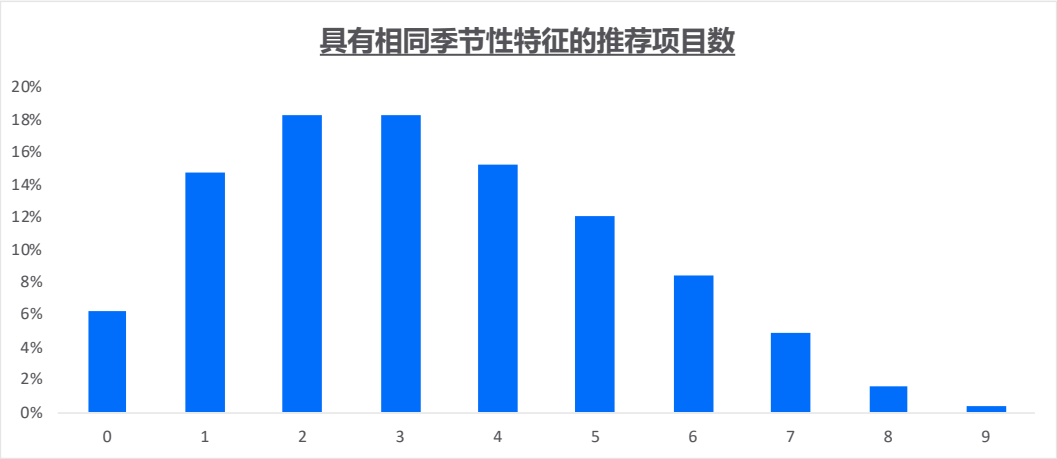

图 5-1 Item2vec 算法对于季节性特征的学习能力

其次，在推荐模型评估指标下，我们比较了模型在不同的交互项目序列长度下的表现。如图 5-2 所示。随着项目序列长度的增加，模型在这三个评价指标上的表现越差。在项目序列长度为 3 的时候，模型在召回率、准确率和平均精度上能够达到的值分别为 0.05，0.01，0.03。这里模型表现不良的原因在于项目嵌入仅学习了项目与项目之间的相似，而没有再进行用户与项目之间相似的学习。

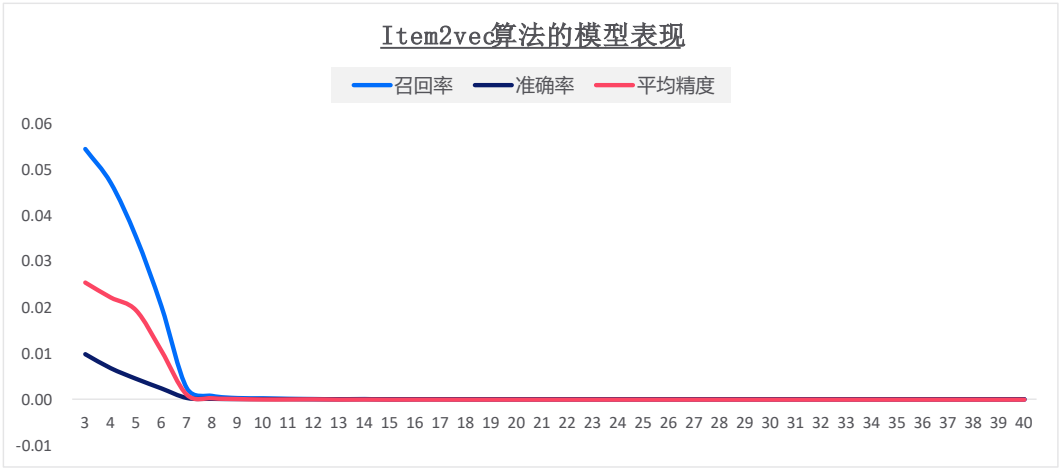

图 5-2 Item2vec 算法的模型表现

5.2.2 对 Bi-LSTM 算法的效果验证

Bi-LSTM 算法的加入使得系统在用以学习项目与项目之间相似的项目嵌入模型层的基础上，再增加了一层用以学习用户与项目之间相似的用户偏好模型层。

图 5-3 是上述模型在不同的交互项目序列长度下的表现。用户偏好学习模型的表现在各个项目序列长度下都比较稳定。模型在召回率、准确率和平均精度上分别达到了 0.81，0.09 和 0.54.

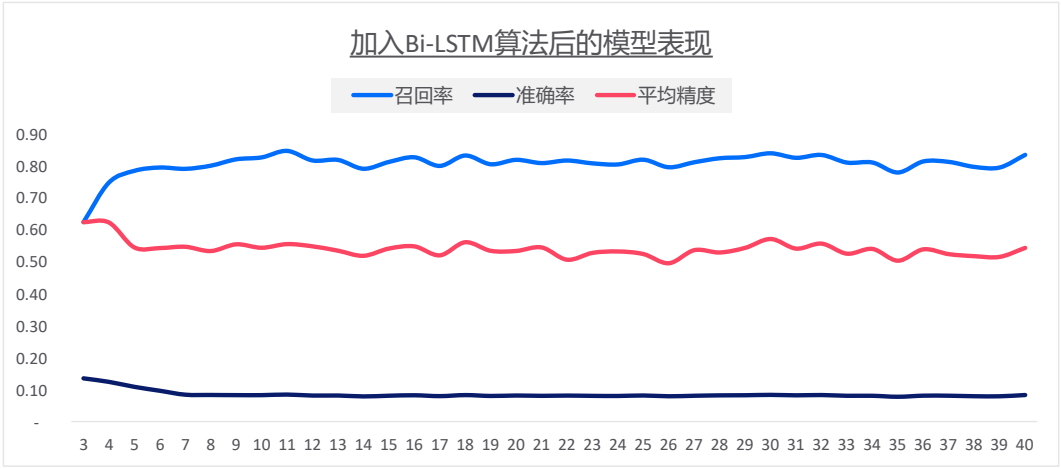

图 5-3 加入 Bi-LSTM 算法后的模型表现

与使用单一的项目嵌入模型所能达到的效果相比，由于加入了 Bi-LSTM 算法的模型能够学习到用户偏好，从而使得模型在各个评价指标上的表现都比单一的项目嵌入模型的表现更为出色。

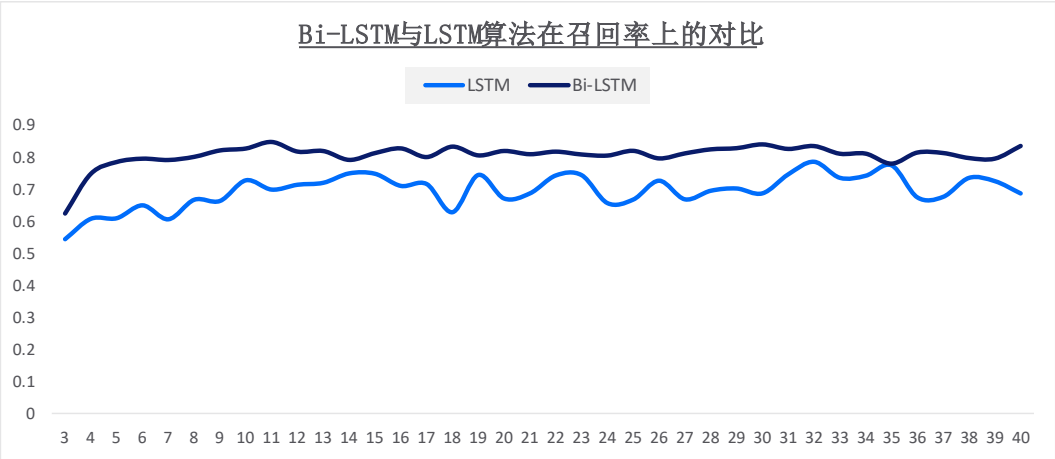

图 5-4 Bi-LSTM 与 LSTM 算法在召回率上的对比

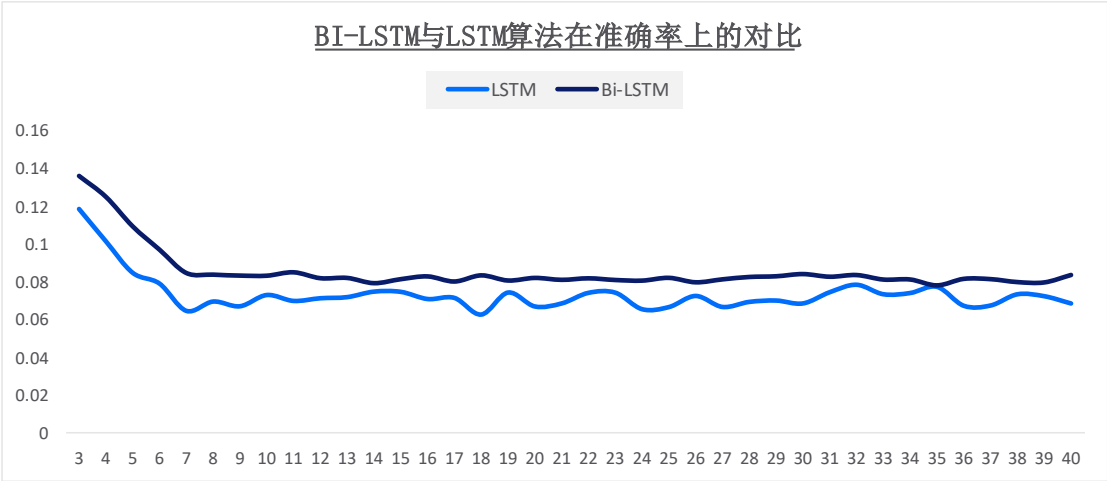

图 5-5 Bi-LSTM 与 LSTM 算法在准确率上的对比

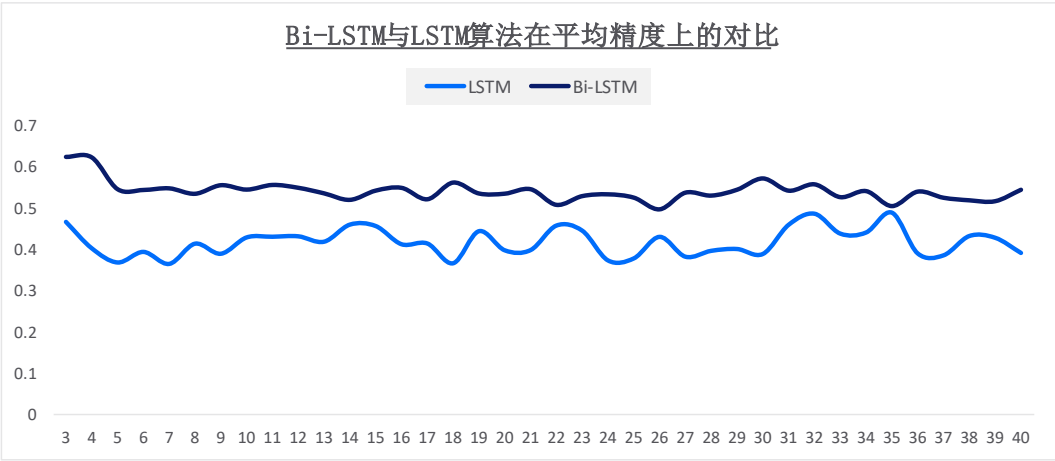

图 5-6 Bi-LSTM 与 LSTM 算法在平均精度上的对比

此外，在第二章的模型理论基础部分，我们提到了 Bi-LSTM 算法是基于 LSTM 算法的一种双向循环神经网络。在这一部分，我们还将 Bi-LSTM 算法与 LSTM 算

法的模型表现进行了对比。图 5-4、图 5-5 和图 5-6 分别从召回率、准确率以及平均精度的评价指标出发比较使用了 LSTM 层的模型和使用 Bi-LSTM 的模型的表现。使用了 Bi-LSTM 的模型在各个评价指标上的表现均优于 LSTM 模型，分别提升了 16%，16%和 0.3%。这表明了 Bi-LSTM 相较于 LSTM 能够通过上下文的序列学习从而更为准确地学习到用户偏好。

5.2.3 对 self-attention 算法的效果验证

self-attention 算法的加入使得系统在学习用户与项目之间相似的过程中，能够对相关信息给予更多的关注，而对不相关信息给予更少的关注，从而更准确地学习用户偏好。

图 5-7 是模型在不同的交互项目序列长度下的表现。偏好权重更新模型的表现在各个项目序列长度下略有波动但总体仍比较稳定。模型在召回率、准确率和平均精度上分别达到了 0.64，0.07 和 0.40。

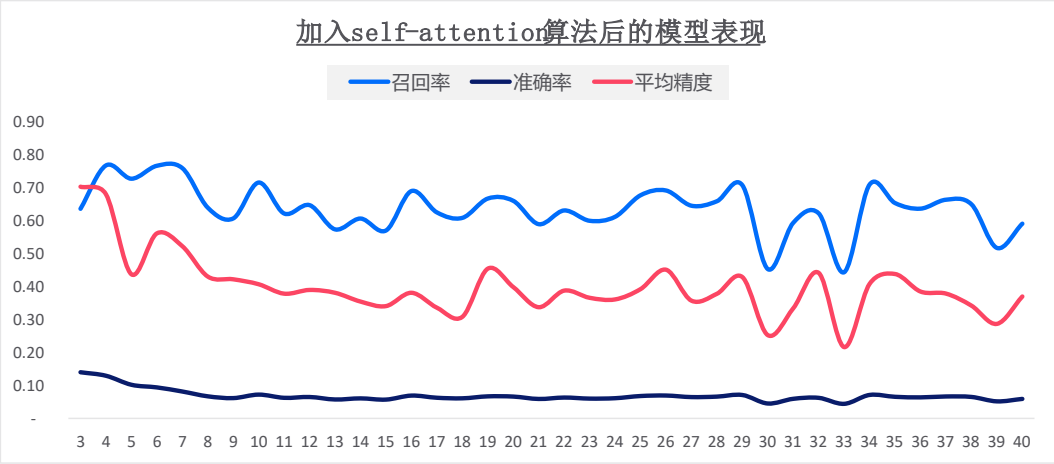

图 5-7 加入 self-attention 算法后的模型表现

图 5-8、图 5-9 和图 5-10 分别从召回率、准确率以及平均精度的评价指标出发比较加入 self-attention 算法前后的模型表现。加入了 self-attention 算法的模型在序列长度小于等于 4 的数据集下的表现优于项目嵌入和用户偏好学习的融合模型，分别在召回率、准确率和平均精度上提高了 2%，2%和 10%。

加入偏好权重更新的模型在序列长度小于等于 4 的情况下表现更优的原因在于模型在学习用户与项目之间相似的过程中，更多地关注到了与用户未来便好相关的信息，从而对用户做出了更有效的推荐。而在序列长度大于 4 的情况下表

现更差的原因在于用户过早的交互记录对用户未来偏好的影响不大。因此过多的关注历史交互记录而忽略了近期交互记录使得模型没有很好地学习到用户偏好。由此，本文最后将使用序列长度为 4 的项目序列数据组进行推荐模型的训练。

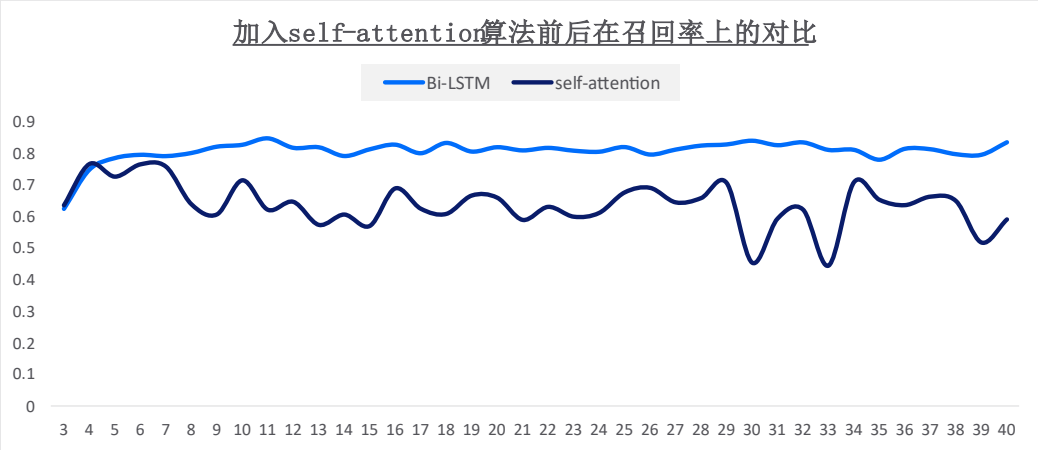

图 5-8 加入 self-attention 算法前后在召回率上的对比

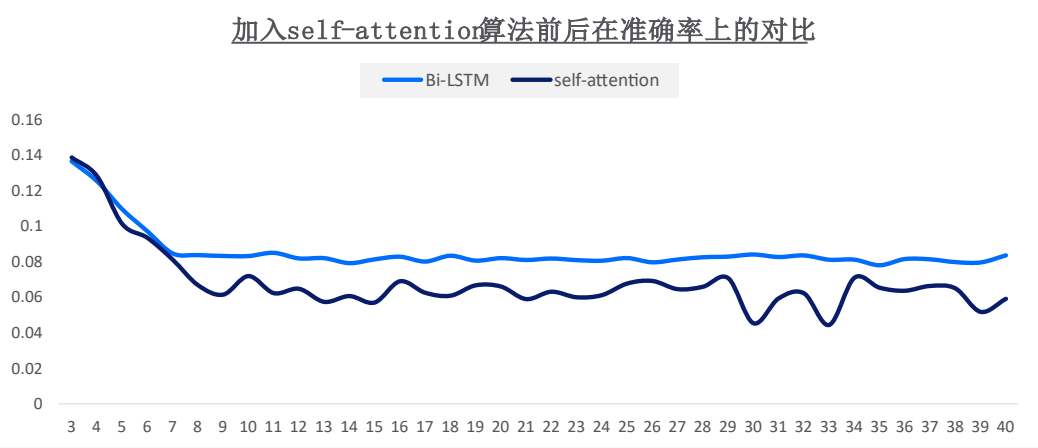

图 5-9 加入 self-attention 算法前后在准确率上的对比

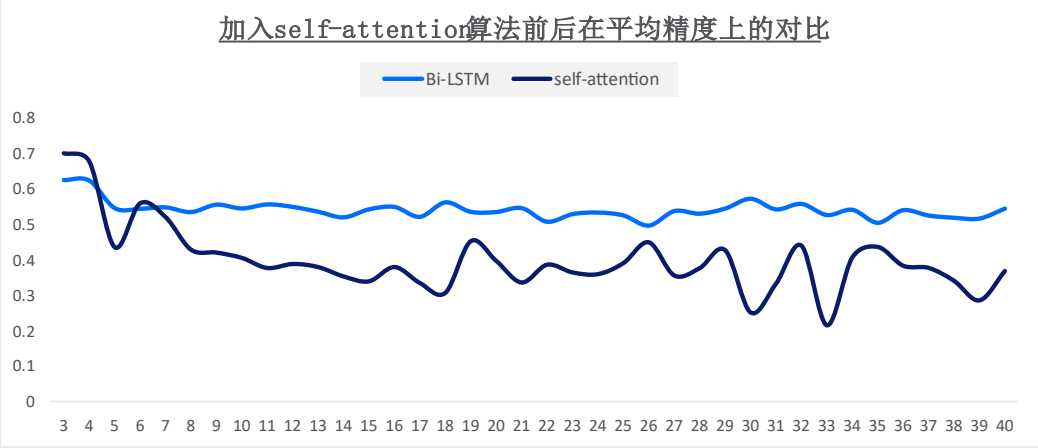

图 5-10 加入 self-attention 算法前后在平均精度上的对比

5.3 本章小结

本章节首先定义了用于评估推荐系统效果的决策性指标和排序性指标，其次验证了模型的推荐效果。验证表明：

（1）Item2vec 算法能够学习到项目之间的季节性相似，但由于未加入用户偏好学习模型层使得模型的推荐效果较差；

（2）Bi-LSTM 算法能够学习到用户偏好，从而使模型的推荐效果在召回率、准确率和平均精度上分别达到了 0.81，0.09 和 0.54，并且与 LSTM 算法相比，分别在上述三个指标下达到了 16%，16%和 0.3%的效果提升；

（3）self-attention 算法能够更多关注到影响用户未来偏好的相关信息，从而使模型的推荐效果在召回率、准确率和平均精度上再次提高了 2%，2%和 10%；

（4）完整模型在召回率、准确率和平均精度上分别达到了 0.64，0.07 和 0.40。

6 系统设计与实现

本文在前序章节中阐明了本推荐系统的模型理论基础、模型构建过程和模型实证分析。在本章节中，将说明本推荐系统的系统设计与实现过程，包括总体设计、数据库设计、模块设计、输入输出设计、界面设计以及系统测试。

6.1 总体设计

在总体设计中，我们会分别阐述本系统的系统信息流程设计和软件架构模式设计。在系统信息流程设计中，将对模型算法层、结果缓存层以及在线服务层进行说明。在软件架构模式设计中，将对 MTV 设计模式进行说明。

6.1.1 系统信息流程设计

如图 6-1 所示，本系统在信息流程上分为三层，分别是模型算法层、结果缓存层以及在线服务层。

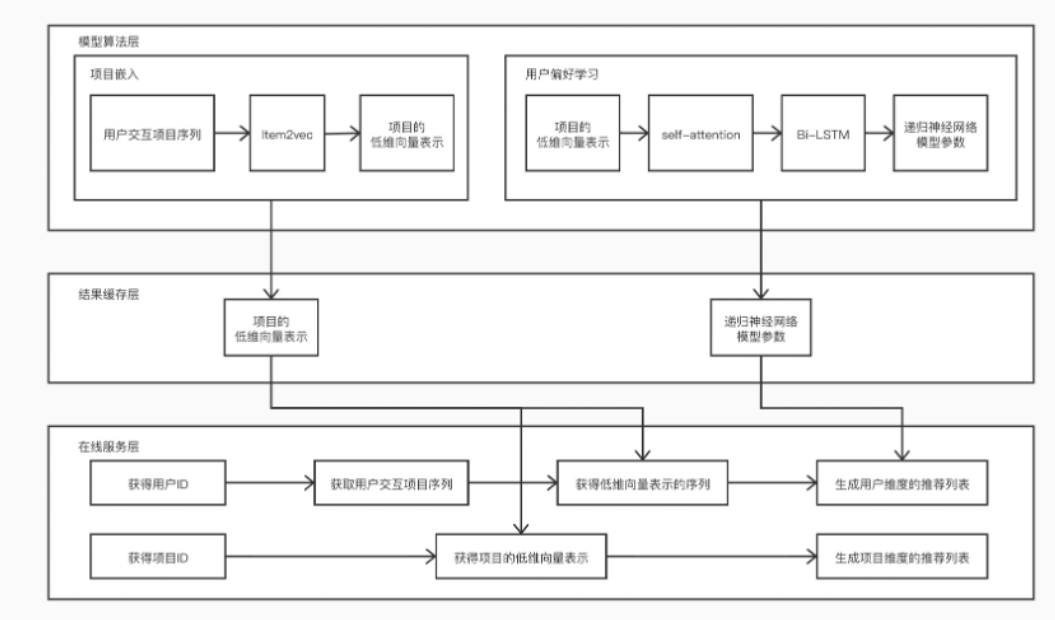

图 6-1 信息流程图

模型算法层是由项目嵌入和用户偏好学习两个子模块构成。项目嵌入会首先利用系统的用户评论信息来构建用户交互项目序列，之后在 Item2vec 模型中学习到每个项目的低维向量表示。用户偏好学习会将项目嵌入学习到的项目向量表示作为循环神经网络的输入，并通过 self-attention 层和 Bi-LSTM 层学习到用

户偏好的向量表示。

结果缓存层是将模型算法层中的已训练好的模型缓存下来以高效地支持系统的调用。对于项目嵌入，系统将缓存模型学习到的项目的低维向量表示。对于用户偏好学习，系统将缓存训练后的循环神经网络的参数设置。

在线服务层是在用户对系统有交互请求时，通过对缓存好的项目向量和循环神经网络的调用来生成系统当前用户的推荐列表，并将推荐列表返回在商品详情页上。

6.1.2 软件架构模式设计

本系统使用的软件架构模式是 Model-Template-View（MTV）模式。如图 6-2 所示，MTV 模式由客户端、服务器和数据库一起协同工作。

对 MTV 设计模式的请求与响应过程描述如下。用户在客户端向服务器发出请求时，由服务器的视图首先做出响应，之后视图再向模型和模版发出请求。模型会从数据库返回给视图相应的数据。模版会通过调用前端页面返回给视图相应的页面。视图层在收到由模型返回的数据和由模版返回的页面后，将相应的数据赋值给模版，完成渲染工作，再将渲染后的模版返回给客户端。

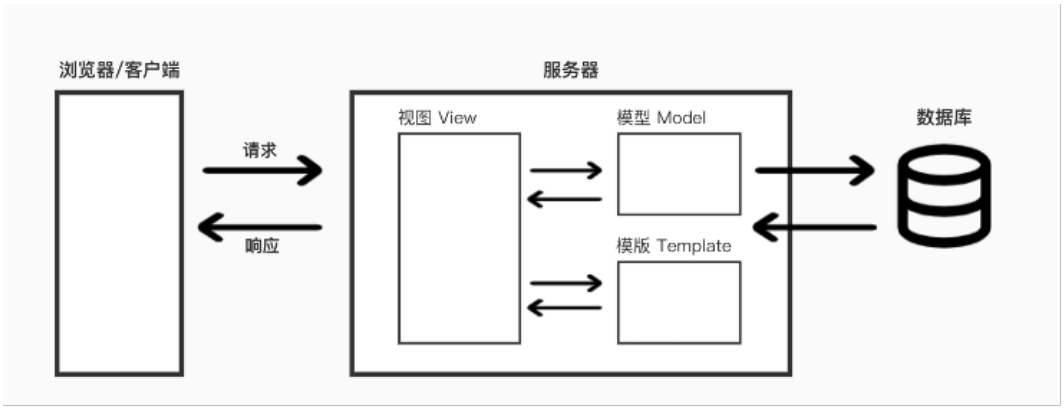

图 6-2 软件架构模式

6.2 数据库设计

本系统主要涉及到用于构建推荐模型的商品评论信息以及用于完善网站功能的用户行为信息。对此，共设计了四张数据表，分别是用户表、商品表、评论表以及心愿单表。

表 6-1 用户表设计

列名	数据类型	允许空值	主键/外键
用户编号	TextField	否	主键
用户名称	TextField	是	

表 6-2 商品表设计

列名	数据类型	允许空值	主键/外键
商品编号	TextField	否	主键
商品名称	TextField	是	
商品图片	ImageField	是	
商品品牌	TextField	是	
商品品类	TextField	是	
商品排名	IntegerField	是	
商品描述	TextField	是	
商品价格	FloatField	是	

表 6-3 评论表设计

列名	数据类型	允许空值	主键/外键
用户编号	TextField	否	主键/外键
商品编号	TextField	否	主键/外键
评分	IntegerField	是	
验证与否	TextField	是	
评论时间	DateTimeField	是	
评论摘要	TextField	是	
评论内容	TextField	是	
赞成数	IntegerField	是	

表 6-4 心愿单表设计

列名	数据类型	允许空值	主键/外键
用户编号	TextField	否	主键/外键
商品编号	TextField	否	主键/外键

6.3 模块设计

本系统面向电商平台用户开放。如下图 6-3 所示，系统支持用户查看用户维度的推荐、查找商品以及管理用户信息。

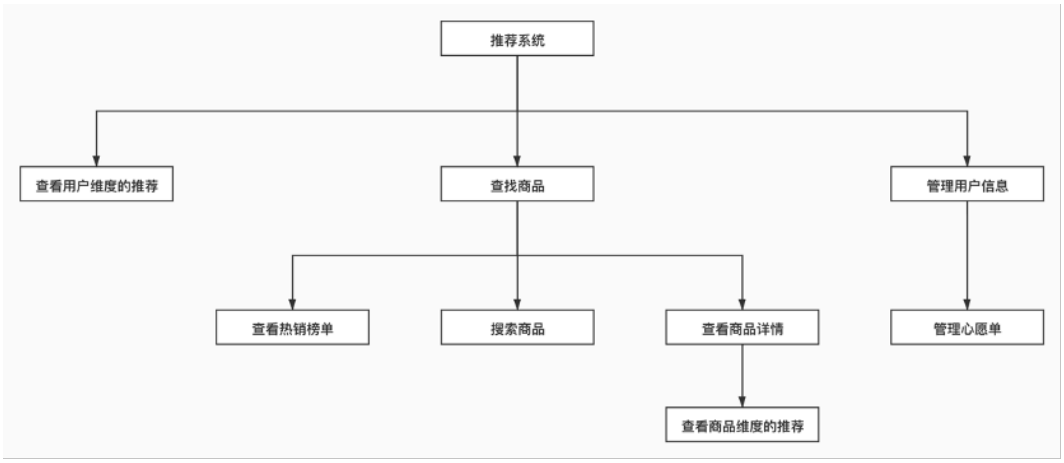

图 6-3 系统模块设计

在查看用户维度的推荐的模块中，如图 6-4 所示，已登陆且有足够的历史交互项目序列支持推荐模型输出结果的用户将看到系统针对该用户生成的个性化推荐列表。如图 6-5 所示，其余用户将看到平台的热销榜单。

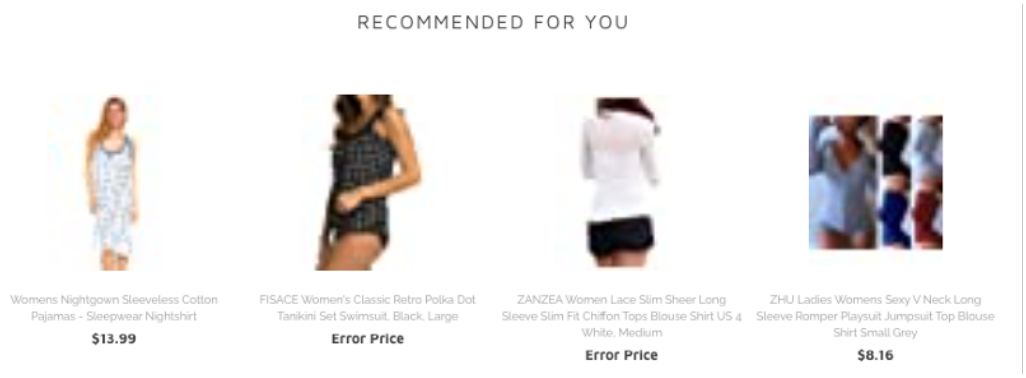

图 6-4 由推荐模型生成的用户维度推荐结果

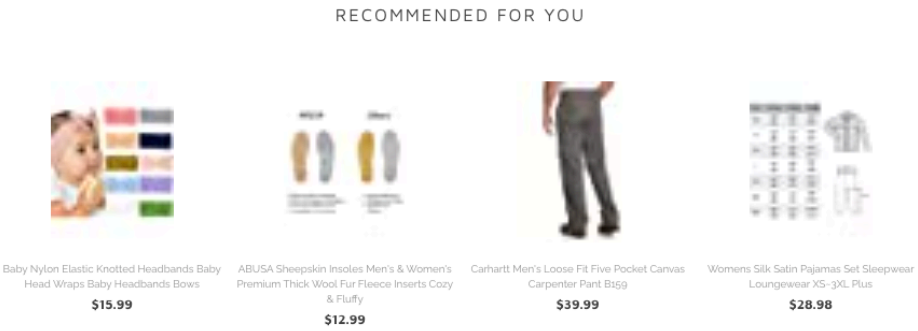

图 6-5 由热销榜单生成的用户维度推荐结果

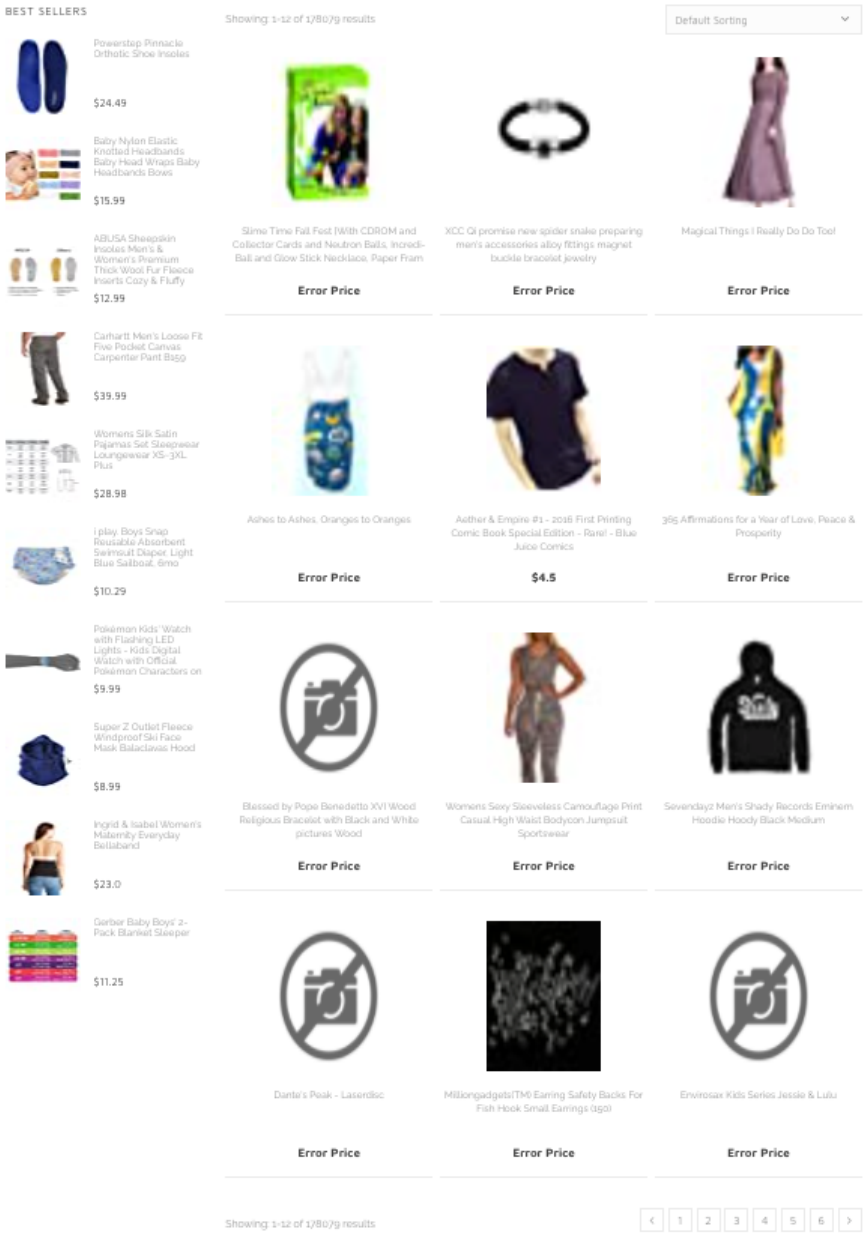

图 6-6 商品概览界面

在查找商品的模块中，用户可以进行热销榜单的查看、商品的搜索、商品详情的查看以及商品维度的推荐查看。如图 6-6 所示，热销榜单是系统根据商品的评论数目以及评论分数而统计生成的。搜索商品支持用户通过向系统输入商品名称关键词的方式来快速找到用户感兴趣的商品，也同时支持用户以价格高低和商品受欢迎程度对搜索结果进行排序。如图 6-7 所示，商品详情的查看提供商品图片、商品标题、商品价格、商品介绍以及用户评论等信息。商品维度的推荐是系统根据项目与项目之间的相似度对每个商品生成的相似商品列表。如图 6-8 所示，针对还未与用户有足够交互记录而无法通过项目嵌入模型获得相似商品列表的商品，系统将以热销榜单替代。

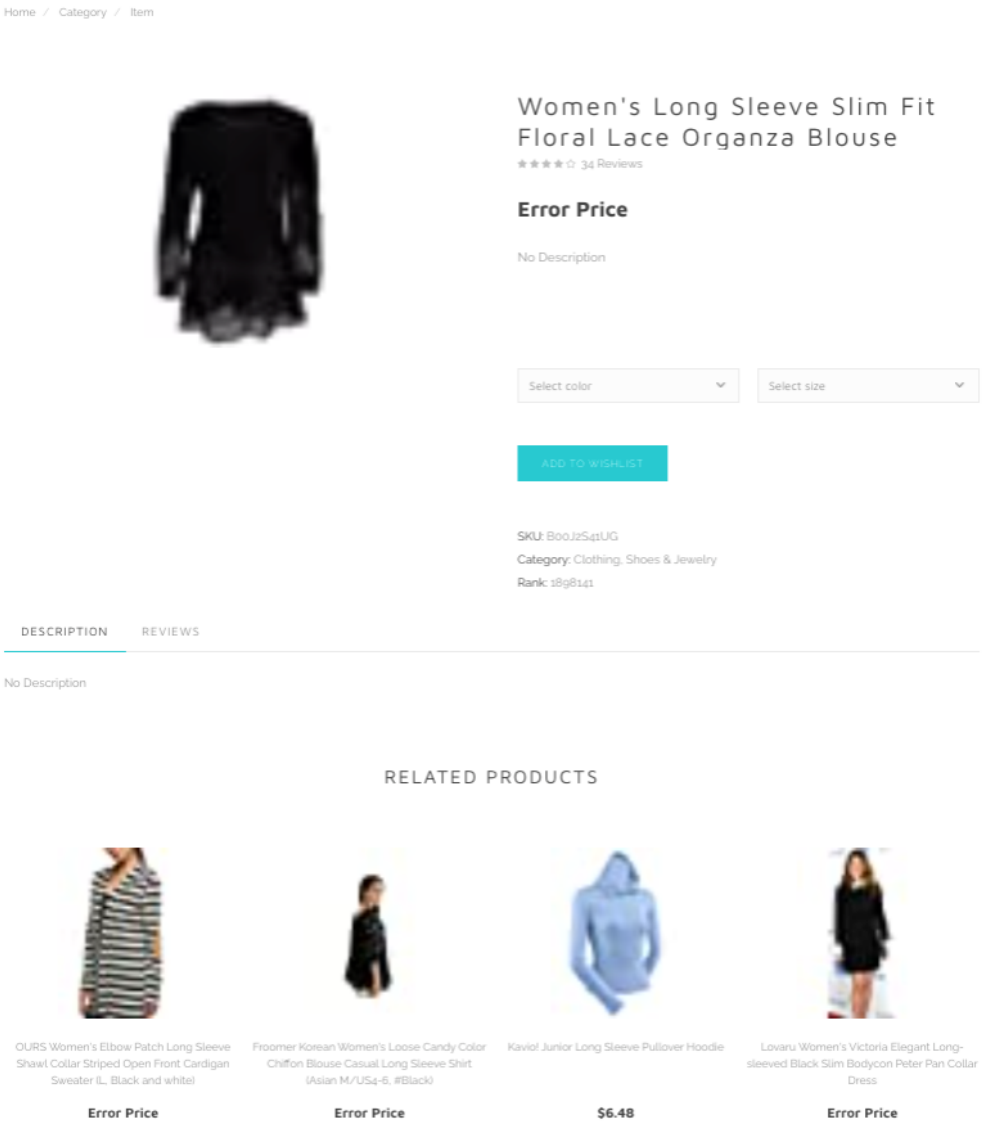

图 6-7 由推荐模型生成的商品详情页面

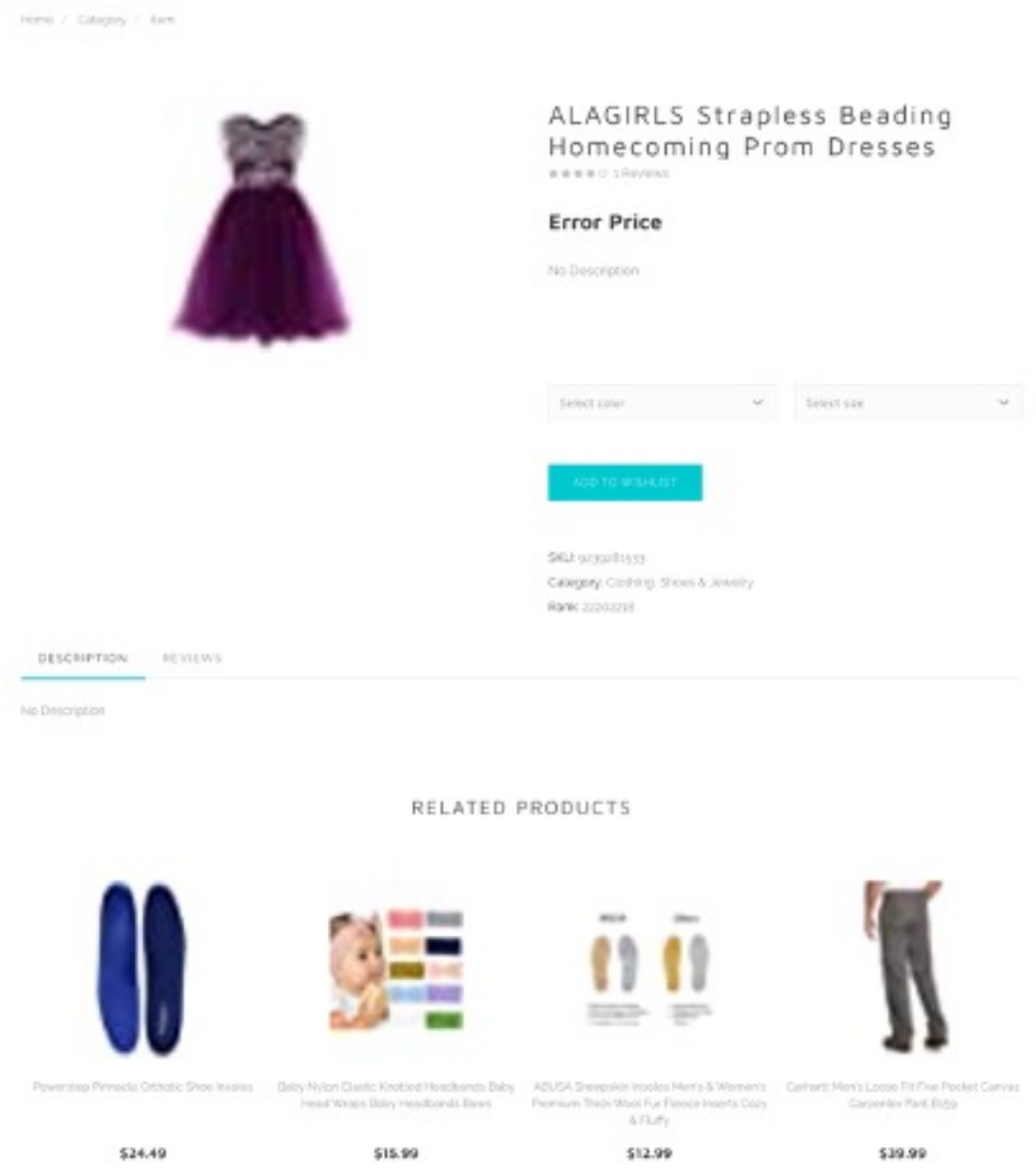

图 6-8 由热销榜单生成的商品详情页面

在查看用户信息的模块中，用户可以进行心愿单的管理。用户在上述的推荐查看和商品搜索的过程中，如图 6-9 和图 6-10 所示，可以随时将商品加入个人的心愿单中，也可以随时将商品移出心愿单。最终，如图 6-11 所示，用户将在心愿单上看到最新状态下仍被用户标记的心愿商品，并可以进行心愿商品的删除操作。

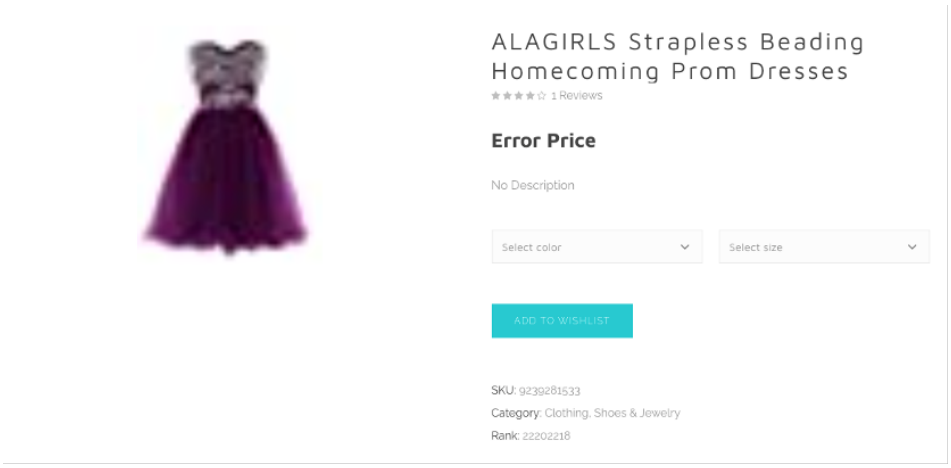

图 6-9 商品的加入/移除心愿单操作

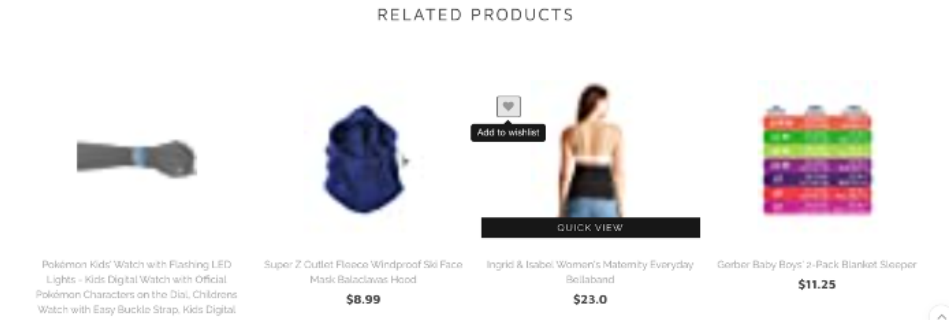

图 6-10 相似商品的加入/移除心愿单操作

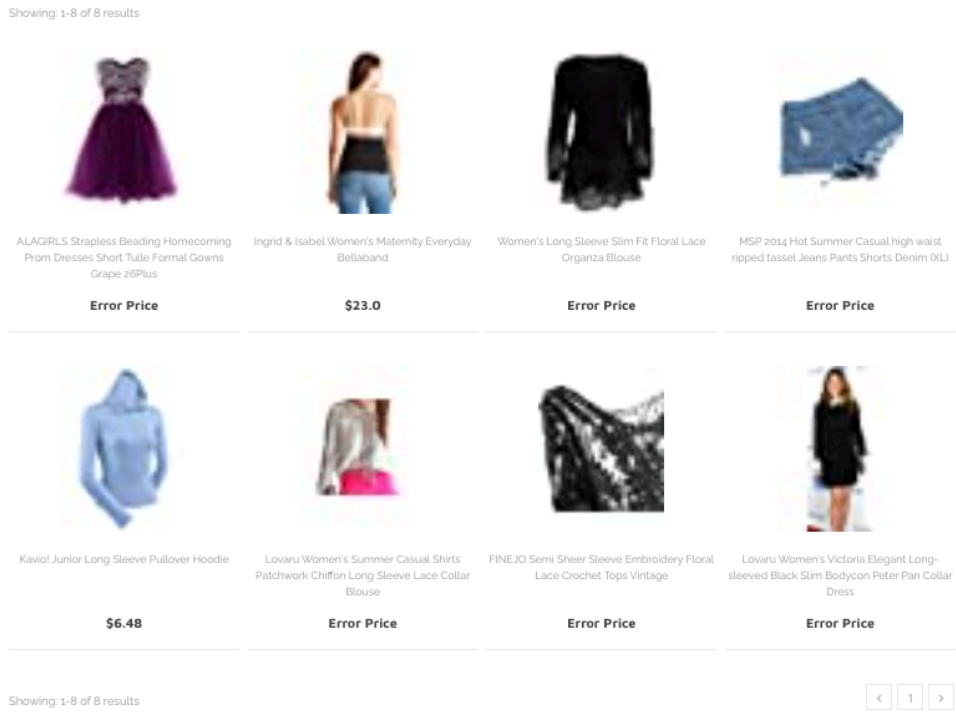

图 6-11 心愿单页面

6.4 本章小结

本章节首先说明了本推荐系统所采用的是由模型算法层、结果缓存层以及在线服务层构成的系统信息流程以及 MTV 软件架构设计模式，其次定义了系统所使用的用户表、商品表、评论表以及心愿单表，最后阐述了系统所支持的用户查看用户维度的推荐、查找商品以及管理用户信息的模块。

7 结论与展望

随着电子商务行业的发展，如何更高效地完成供需关系的匹配受到了广泛的关注。本文聚焦于电子商务行业中具有季节性特征的服饰品类，在传统推荐算法的启发下，使用优化推荐方法中的人工神经网络方法，完成了商品推荐问题。本文的主要内容总结如下：

（1）本文提出的推荐模型由项目嵌入层和用户偏好学习层组成。在项目嵌入层中，使用 Item2vec 模型学习项目的嵌入向量表示形式从而得到项目与项目间的相似，包括项目间的季节性相似和项目间的内容相似。在用户偏好学习层中，使用 Bi-LSTM 和 self-attention 模型学习到用户偏好从而得到用户与项目间的相似。

（2）项目嵌入层中 Item2vec 算法能够学习到项目之间的季节性相似，用户偏好学习层中的 Bi-LSTM 算法能够学习到用户偏好而 self-attention 算法能够更多关注到影响用户未来偏好的相关信息，使得推荐模型在召回率、准确率和平均精度上分别达到了 0.64，0.07 和 0.40。

（3）本文使用 Django 和 Bootstrap 技术路线将上述推荐模型应用于推荐系统中，实现了用户维度和商品维度的推荐生成，并支持用户完成商品的搜索、排序和心愿单管理，向用户提供了良好的系统体验。

本文所实现的推荐模型仍有待进一步的完善：

（1）本文在计算项目与项目之间的相似度时，仅依赖于用户的历史交互序列，并未利用项目本身属性，例如商品标题、描述和评论等信息。如何处理并利用此文本信息有待进一步的研究。

（2）本文在计算用户与项目之间的相似度时，依赖于用户完整的历史交互序列，再通过 self-attention 算法更加关注于会影响到用户未来偏好的项目。但其中用户在很久以前所做出的交互行为可能是不需要关注的。如何截取有效的历史交互序列有待进一步的探索。

参考文献

[1] Adomavicius G, Tuzhilin A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions[J]. IEEE transactions on knowledge and data engineering, 2005, 17(6): 734-749.

[2] Billsus D, Pazzani M J. Learning collaborative information filters[C]//Icml. 1998, 98: 46-54.

[3] Breese J S, Heckerman D, Kadie C. Empirical analysis of predictive algorithms for collaborative filtering[J]. arXiv preprint arXiv:1301.7363, 2013.

[4] Deshpande M, Karypis G. Item-based top-n recommendation algorithms[J]. ACM Transactions on Information Systems (TOIS), 2004, 22(1): 143-177.

[5] Huang Z, Chen H, Zeng D. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering[J]. ACM Transactions on Information Systems (TOIS), 2004, 22(1): 116-142.

[6] Pazzani M J. A framework for collaborative, content-based and demographic filtering[J]. Artificial intelligence review, 1999, 13(5): 393-408.

[7] Pazzani M, Billsus D. Learning and revising user profiles: The identification of interesting web sites[J]. Machine learning, 1997, 27(3): 313-331.

[8] Robertson S, Walker S. Threshold setting in adaptive filtering[J]. Journal of documentation, 2000.

[9] Resnick P, Iacovou N, Suchak M, et al. Grouplens: An open architecture for collaborative filtering of netnews[C]//Proceedings of the 1994 ACM conference on Computer supported cooperative work. 1994: 175-186.

[10] Sarwar B, Karypis G, Konstan J, et al. Application of dimensionality reduction in recommender system-a case study[R]. Minnesota Univ Minneapolis Dept of Computer Science, 2000.

[11] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms[C]//Proceedings of the 10th international conference on World Wide Web. 2001: 285-295.

[12] Shardanand U, Maes P. Social information filtering: Algorithms for automating

“word of mouth”[C]//Proceedings of the SIGCHI conference on Human factors in computing systems. 1995: 210-217.

[13] Somlo G L, Howe A E. Adaptive lightweight text filtering[C]//International Symposium on Intelligent Data Analysis. Springer, Berlin, Heidelberg, 2001: 319-329.

[14] Yu K, Xu X, Tao J, et al. Instance selection techniques for memory-based collaborative filtering[C]//Proceedings of the 2002 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, 2002: 59-74.

[15] Zhang Y, Callan J. Maximum likelihood estimation for filtering thresholds[C]//Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval. 2001: 294-302.

[16] Zhang Y, Callan J, Minka T. Novelty and redundancy detection in adaptive filtering[C]//Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval. 2002: 81-88.

[17] Mikolov T, Sutskever I, Chen K, et al. Distributed representations of words and phrases and their compositionality[J]. arXiv preprint arXiv:1310.4546, 2013.

[18] Barkan O, Koenigstein N. Item2vec: neural item embedding for collaborative filtering[C]//2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP). IEEE, 2016: 1-6.

[19] Graves A, Schmidhuber J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures[J]. Neural networks, 2005, 18(5-6): 602-610.

[20] Goldberg D, Nichols D, Oki B M, et al. Using collaborative filtering to weave an information tapestry[J]. Communications of the ACM, 1992, 35(12): 61-70.

[21] Schafer J B, Frankowski D, Herlocker J, et al. Collaborative filtering recommender systems[M]//The adaptive web. Springer, Berlin, Heidelberg, 2007: 291-324.

[22] Pazzani M J, Billsus D. Content-based recommendation systems[M]//The adaptive web. Springer, Berlin, Heidelberg, 2007: 325-341.

[23] Burke R. Knowledge-based recommender systems[J]. Encyclopedia of library and information systems, 2000, 69(Supplement 32): 175-186.

[24] Burke R. Hybrid recommender systems: Survey and experiments[J]. User modeling and user-adapted interaction, 2002, 12(4): 331-370.

[25] Deshpande M, Karypis G. Item-based top-n recommendation algorithms[J]. ACM Transactions on Information Systems (TOIS), 2004, 22(1): 143-177.

[26] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms[C]//Proceedings of the 10th international conference on World Wide Web. 2001: 285-295.

[27] RESNICK P. AnOpen Architecture for Collaborative Filterring of Netnews[C]//Proc CSCW'94. 1994.

[28] Burke R. Hybrid web recommender systems[J]. The adaptive web, 2007: 377-408.

[29] Bellogín A, Cantador I, Díez F, et al. An empirical comparison of social, collaborative filtering, and hybrid recommenders[J]. ACM Transactions on Intelligent Systems and Technology (TIST), 2013, 4(1): 1-29.

[30] Christakou C, Vrettos S, Stafylopatis A. A hybrid movie recommender system based on neural networks[J]. International Journal on Artificial Intelligence Tools, 2007, 16(05): 771-792.

[31] Nanopoulos A, Rafailidis D, Symeonidis P, et al. Musicbox: Personalized music recommendation based on cubic analysis of social tags[J]. IEEE Transactions on Audio, Speech, and Language Processing, 2009, 18(2): 407-412.

[32] Al-Shamri M Y H, Bharadwaj K K. Fuzzy-genetic approach to recommender systems based on a novel hybrid user model[J]. Expert systems with applications, 2008, 35(3): 1386-1399.

[33] Cao Y, Li Y. An intelligent fuzzy-based recommendation system for consumer electronic products[J]. Expert Systems with Applications, 2007, 33(1): 230-240.

[34] Tan S, Bu J, Chen C, et al. Using rich social media information for music recommendation via hypergraph model[J]. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 2011, 7(1): 1-22.

[35] McCarthy K, Reilly J, McGinty L, et al. Thinking positively-explanatory feedback for conversational recommender systems[C]//Proceedings of the European Conference

on Case-Based Reasoning (ECCBR-04) Explanation Workshop. 2004: 115-124.

[36] Garfinkel R, Gopal R, Tripathi A, et al. Design of a shopbot and recommender system for bundle purchases[J]. Decision Support Systems, 2006, 42(3): 1974-1986.

[37] Lawrence R D, Almasi G S, Kotlyar V, et al. Personalization of supermarket product recommendations[M]//Applications of data mining to electronic commerce. Springer, Boston, MA, 2001: 11-32.

[38] Hsu S H, Wen M H, Lin H C, et al. AIMED-A personalized TV recommendation system[C]//European conference on interactive television. Springer, Berlin, Heidelberg, 2007: 166-174.

致谢

首先，感谢我在东华大学度过的五年时光。我完成了专业学科的学习，也成为了更加成熟的自己。其次，感谢我的指导老师范宏老师在我本科学习期间与论文设计期间对我的关心与指导。老师的专业知识与见解使我得以顺利地完成我的毕业论文。此外，感谢所有学科老师对我的帮助。我在专业知识上的学习道路离不开你们对我的悉心教导。最后，感谢所有与我一同成长的同学们。你们的优秀与努力鞭策着我与你们一同进步。

附录

附录一 推荐算法

```
import os
import sys

sys.path.insert(0, '/Users/qingxuankong/PycharmProjects/amazon/recommender_system')

import django

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'recommender_system.settings')
os.environ["DJANGO_ALLOW_ASYNC_UNSAFE"] = "true"
os.environ['TF_XLA_FLAGS'] = '--tf_xla_enable_xla_devices'
django.setup()

import operator
import pandas as pd
import numpy as np
import itertools
from gensim.models import Word2Vec
from keras.preprocessing import sequence
from keras.layers.recurrent import LSTM
from keras.layers.embeddings import Embedding
from keras.models import Sequential
from keras.layers import Dense, Activation, Bidirectional, Flatten
from keras_self_attention import SeqSelfAttention
from item_recommend.models import Items, Customers, Ratings, User_Recommendations,
Item_Recommendations


def recommend_items(len_thres, top_k):
    print('preparing item dict...')
    item_dict = create_item_dict()

    print('preparing user item dict...')
    user_item_dict = create_user_item_dict(len_thres)
    total_users_list = list(user_item_dict.keys())
    total_items_list = list(set(itertools.chain.from_iterable(user_item_dict.values())))

    print('training item2vec model...')
    item2vec_model = train_item2vec_model(user_item_dict.values(), total_items_list)

    print('building user-item matrix...')
```

```
    user_item_matrix = create_user_item_matrix(user_item_dict, total_users_list, total_items_list)

    print('building item-item matrix...')
    item_item_matrix = create_item_item_matrix(item2vec_model, total_items_list)

    print('calculating new user-item matrix...')
    new_user_item_matrix = user_item_matrix.dot(item_item_matrix)

    print('predict using item2vec model...')
    predict_item_sequences_item2vec = predict_item2vec_model(top_k, total_users_list,
user_item_matrix,

new_user_item_matrix, item_dict)

    print('preparing embedding layer...')
    pretrained_weights = item2vec_model.wv.vectors
    vocab_size, embedding_size = pretrained_weights.shape
    print('embedding shape:', pretrained_weights.shape)

    train_item_sequences_x, train_item_sequences_y, test_item_sequences =
split_user_item_dict(user_item_dict)

    print('preparing train user item sequence...')
    train_users, train_x_wv_array, train_y_wv_array = create_item_sequence(item2vec_model,
train_item_sequences_x,

train_item_sequences_y)

    print('training bi-lstm model...')
    bi_lstm_model = train_bi_lstm_model(train_x_wv_array, train_y_wv_array, vocab_size,
embedding_size,
                                        pretrained_weights)

    print('predict using bi-lstm model...')
    predict_item_sequences_bilstm = predict_bi_lstm_model(bi_lstm_model, item2vec_model,
train_users, train_x_wv_array,
                                                          top_k, item_dict)

    print('fetch top selling items list...')
    top_selling_items = fetch_top_selling_items(top_k)

    print('creating user recommendation list...')
    predict_item_sequences = create_user_recommendations(total_users_list,
```

```
predict_item_sequences_bilstm,

predict_item_sequences_item2vec, top_selling_items)

    print('creating item recommendation list...')
    similar_item_sequences = create_items_recommendations(total_items_list, item2vec_model,
top_selling_items, top_k)

    print('inserting user recommendation list...')
    insert_user_recommendations(predict_item_sequences)

    print('inserting item recommendation list...')
    insert_item_recommendations(similar_item_sequences)

    return predict_item_sequences, similar_item_sequences


def create_item_dict():
    df = pd.DataFrame(list(Items.objects.all().values()))
    df_shrt = df[['asin', 'title']]
    item_dict = dict(zip(df_shrt.asin, df_shrt.title))

    return item_dict


def create_user_item_dict(len_thres):
    df = pd.DataFrame(list(Ratings.objects.all().values()))

    # read raw rating file
    df_shrt = df[['reviewerID_id', 'asin_id', 'reviewTime']]
    df_shrt = df_shrt.drop_duplicates()
    df_shrt = df_shrt.sort_values(by='reviewTime')
    df_shrt.reset_index(inplace=True)

    # create user rating records
    user_rating_records = {}
    for i in range(df_shrt.shape[0]):
        try:
            if df_shrt.loc[i, 'reviewerID_id'] not in user_rating_records.keys():
                user_rating_records[df_shrt.loc[i, 'reviewerID_id']] = []
            user_rating_records[df_shrt.loc[i,          'reviewerID_id']].append(str(df_shrt.loc[i,
'asin_id']))
        except:
```

```
                pass

        # remove one item record
        user_rating_records_cleaned = {}
        for key, value in user_rating_records.items():
            if len(value) >= len_thres:
                user_rating_records_cleaned[key] = value

        return user_rating_records_cleaned


def train_item2vec_model(item_sentences, items_list):
        model = Word2Vec(item_sentences, size=32, window=3, min_count=2, sample=1e-3,
                         negative=5, workers=12, iter=1000, sg=1, hs=0)
        model.build_vocab(items_list, update=True)
        model.train(item_sentences, total_examples=model.corpus_count, epochs=22)
        model.init_sims()
        word_vectors = model.wv
        word_vectors.save('./data/model/item_recommendations')

        return model


def create_user_item_matrix(item_sequences, users_list, items_list):
        user_item_matrix = pd.DataFrame(np.zeros((len(users_list), len(items_list))))
        user_item_matrix.columns = items_list
        user_item_matrix.index = users_list

        for key, values in item_sequences.items():
            for value in values:
                user_item_matrix.at[key, value] = 1

        return user_item_matrix


def create_item_item_matrix(item2vec_model, items_list):
        item_item_matrix = pd.DataFrame(np.zeros((len(items_list), len(items_list))))
        item_item_matrix.columns = items_list
        item_item_matrix.index = items_list

        for i in items_list:
            for j in items_list:
                if i in item2vec_model.wv.vocab and j in item2vec_model.wv.vocab:
```

```
                item_item_matrix.at[i, j] = item2vec_model.wv.similarity(i, j)

    return item_item_matrix


def predict_item2vec_model(top_k, users_list, user_item_matrix, new_user_item_matrix, item_dict):
    user_liked = {}
    user_may_like = {}
    for user in users_list:
        user_records = user_item_matrix.loc[user, :].to_dict()
        user_liked[user] = []
        for key, value in user_records.items():
            if value == 1:
                user_liked[user].append(key)

        user_favor = new_user_item_matrix.loc[user, :].to_dict()
        user_favor = sorted(user_favor, key=user_favor.get, reverse=True)
        user_may_like[user], titles_list = [], []
        count = 0
        for item in user_favor:
            item_title = item_dict[item]
            if item not in user_liked[user] and item_title not in titles_list:
                user_may_like[user].append(item)
                titles_list.append(item_title)
                count += 1
            if count == top_k:
                break

    return user_may_like


def split_user_item_dict(user_item_dict):
    max_len = max([len(i) for i in user_item_dict.values()])
    min_len = min([len(i) for i in user_item_dict.values()])

    for i in range(5, 5 + 1):
        user_slctd = [k for k, v in user_item_dict.items() if len(v) >= i]
        item_sequences_slctd = [v for k, v in user_item_dict.items() if len(v) >= i]
        train_item_sequences_x = [j[:i - 1] for j in item_sequences_slctd]
        train_item_sequences_y = [j[i - 1:i] for j in item_sequences_slctd]
        test_item_sequences = [j[i - 1:] for j in item_sequences_slctd]
```

```
        train_item_sequences_x = dict(zip(user_slctd, train_item_sequences_x))
        train_item_sequences_y = dict(zip(user_slctd, train_item_sequences_y))
        test_item_sequences = dict(zip(user_slctd, test_item_sequences))

    return train_item_sequences_x, train_item_sequences_y, test_item_sequences


def create_item_sequence(item2vec_model, train_x, train_y):
    train_users, train_x_wv, train_y_wv = [], [], []
    max_len = 0

    for i in range(len(train_x)):
        vocab_flag_x = min([j in item2vec_model.wv.vocab for j in list(train_x.values())[i]])
        vocab_flag_y = min([j in item2vec_model.wv.vocab for j in list(train_y.values())[i]])
        vocab_flag = min(vocab_flag_x, vocab_flag_y)
        if vocab_flag == 1:
            if max_len < len(list(train_x.values())[i]):
                max_len = len(list(train_x.values())[i])
            user = list(train_x.keys())[i]
            x_wv = [item2vec_model.wv.vocab[j].index for j in list(train_x.values())[i]]
            y_wv = [item2vec_model.wv.vocab[j].index for j in list(train_y.values())[i]]
            train_users.append(user)
            train_x_wv.append(x_wv)
            train_y_wv.append(y_wv)

    train_x_wv = sequence.pad_sequences(train_x_wv, maxlen=max_len)
    train_x_wv_array = np.asarray(train_x_wv).astype('int32')
    train_y_wv_array = np.asarray(train_y_wv).astype('int32')

    return train_users, train_x_wv_array, train_y_wv_array


def train_bi_lstm_model(train_x_wv_array, train_y_wv_array, vocab_size, embedding_size,
                        pretrained_weights):
    model = Sequential()
    model.add(Embedding(input_dim=vocab_size, output_dim=embedding_size,
weights=[pretrained_weights]))
    model.add(Bidirectional(LSTM(units=embedding_size, return_sequences=True)))
    model.add(SeqSelfAttention(attention_activation='softmax'))
    model.add(Dense(units=vocab_size))
    model.add(Flatten())
    model.add(Activation('softmax'))
    model.compile(optimizer='adam', loss='sparse_categorical_crossentropy')
```

```
    model.summary()

    model.fit(train_x_wv_array, train_y_wv_array, epochs=100, verbose=2)

    model.save('./data/model/user_recommendations')

    return model


def predict_bi_lstm_model(bi_lstm_model, item2vec_model, train_users, train_x_wv_array, top_k, item_dict):
    predict_item_sequences = {}
    for i in range(len(train_x_wv_array)):
        predict_dict = {v: k for v, k in enumerate(bi_lstm_model.predict(train_x_wv_array[i])[-1])}
        predict_dict_sorted = dict(sorted(predict_dict.items(), key=operator.itemgetter(1), reverse=True))

        titles_list, user_sequences = [], []
        item_cnt = 0
        for item_wv in predict_dict_sorted.keys():
            item_asin = item2vec_model.wv.index2word[item_wv]
            item_title = item_dict[item_asin]

            if item_title not in titles_list:
                titles_list.append(item_title)
                user_sequences.append(item_asin)
                item_cnt += 1
            if item_cnt == top_k:
                break

        predict_item_sequences[train_users[i]] = user_sequences

    return predict_item_sequences


def fetch_top_selling_items(top_k):
    df = pd.DataFrame(list(Items.objects.all().values()))
    df = df.sort_values(by=['category_rank']).reset_index(drop=True)
    top_selling_items = df.loc[:top_k - 1, 'asin'].tolist()

    return top_selling_items
```

```
def create_user_recommendations(total_users_list, predict_item_sequences_bilstm, predict_item_sequences_item2vec):
    predict_item_sequences = {}
    for user in total_users_list:
        if user in predict_item_sequences_bilstm.keys():
            predict_item_sequences[user] = predict_item_sequences_bilstm[user]
        elif user in predict_item_sequences_item2vec.keys():
            predict_item_sequences[user] = predict_item_sequences_item2vec[user]

    return predict_item_sequences


def create_items_recommendations(total_items_list, item2vec_model, top_k):
    similar_item_sequences = {}
    for item in total_items_list:
        if item in item2vec_model.wv.vocab:
            similar_item_sequences[item] = []
            similar_item_cnt = 0
            for tuple in item2vec_model.wv.most_similar(item, topn=top_k + 10):
                if tuple[0] in total_items_list:
                    similar_item_sequences[item].append(tuple[0])
                    similar_item_cnt += 1
                if similar_item_cnt == top_k:
                    break

    return similar_item_sequences


def insert_user_recommendations(predict_item_sequences):
    User_Recommendations.objects.all().delete()
    for key, value in predict_item_sequences.items():
        user_recommendation = User_Recommendations()
        user_recommendation.reviewerID = Customers.objects.filter(reviewerID=key)[0]
        user_recommendation.recomm_asin_1 = Items.objects.filter(asin=value[0])[0]
        user_recommendation.recomm_asin_2 = Items.objects.filter(asin=value[1])[0]
        user_recommendation.recomm_asin_3 = Items.objects.filter(asin=value[2])[0]
        user_recommendation.recomm_asin_4 = Items.objects.filter(asin=value[3])[0]
        user_recommendation.recomm_asin_5 = Items.objects.filter(asin=value[4])[0]
        user_recommendation.recomm_asin_6 = Items.objects.filter(asin=value[5])[0]
        user_recommendation.recomm_asin_7 = Items.objects.filter(asin=value[6])[0]
        user_recommendation.recomm_asin_8 = Items.objects.filter(asin=value[7])[0]
        user_recommendation.recomm_asin_9 = Items.objects.filter(asin=value[8])[0]
```

```
        user_recommendation.recomm_asin_10 = Items.objects.filter(asin=value[9])[0]
        user_recommendation.save()


def insert_item_recommendations(similar_item_sequences):
    Item_Recommendations.objects.all().delete()
    for key, value in similar_item_sequences.items():
        item_recommendation = Item_Recommendations()
        item_recommendation.asin = Items.objects.filter(asin=key)[0]
        item_recommendation.recomm_asin_1 = Items.objects.filter(asin=value[0])[0]
        item_recommendation.recomm_asin_2 = Items.objects.filter(asin=value[1])[0]
        item_recommendation.recomm_asin_3 = Items.objects.filter(asin=value[2])[0]
        item_recommendation.recomm_asin_4 = Items.objects.filter(asin=value[3])[0]
        item_recommendation.recomm_asin_5 = Items.objects.filter(asin=value[4])[0]
        item_recommendation.recomm_asin_6 = Items.objects.filter(asin=value[5])[0]
        item_recommendation.recomm_asin_7 = Items.objects.filter(asin=value[6])[0]
        item_recommendation.recomm_asin_8 = Items.objects.filter(asin=value[7])[0]
        item_recommendation.recomm_asin_9 = Items.objects.filter(asin=value[8])[0]
        item_recommendation.recomm_asin_10 = Items.objects.filter(asin=value[9])[0]
        item_recommendation.save()


if __name__ == '__main__':
    len_thres = 3
    top_k = 10
    predict_item_sequences, similar_item_sequences = recommend_items(len_thres, top_k)
```

东华大学

毕业设计（论文）英文翻译

课 题 名 称 ： 针对季节性服饰商品的推荐系统的分析和设计

学 院 ： 旭日工商管理学院

专 业 ： 信息管理与信息系统

姓 名 ： 孔庆璇

学 号 ： 160610132

指 导 教 师 ： 范 宏

二 零 二 一 年 一 月 五 日

1. 引言

自动化推荐已经成为我们线上用户体验中普遍的特征。由于其实际重要性，推荐系统也代表了一个科学研究的活跃领域。历史上，推荐系统的两大主要方法是协同过滤和基于内容的过滤。纯粹地说，协同过滤依靠用户社区中的偏好模式的识别。相反，基于内容的过滤方法值考虑用户过去的偏好，并尝试基于推荐项目内容的特征表示来学习一个偏好模型。

由于仅考虑个体用户过去偏好的潜在局限性，过去二十年提出了很多将两种方法结合起来的混合方法。近年来，我们进一步观察到将其他辅助信息和外部知识来源并入推荐过程中的各种尝试。就基于内容的方法而言，这些辅助信息主要包含额外的关于推荐项目的知识，例如它们的特征，元数据，类别分配，和其他项目的关系，用户提供的标签和评论，或者是相关的文本或多媒体内容。

尽管纯协同过滤方法主导了研究领域（Jannach 等人 2012），考虑到内容信息（例如项目的特征）在今天的许多推荐的实际应用中是非常重要的，并且由于这些原因会变得越来越重要：

- 首先，许多研究工作表明基于内容的方法和混合方法，尽管优势在离线实验中准确性较低，但在一些应用中比纯协同过滤的方法更为高效（Jannach and Hegelich 2009; Kirshenbaum 等人 2012; Liu 等人 2010）。
- 其次，只有当项目特征被纳入考虑时，推荐系统的某些方面才能被有效实行。例如，这些方面包括推荐列表结果的多样性，解释的生成，或者是可理解的用户模型的创建。
- 第三，我们继续看到提供关于项目结构化（如 DBpedia），半结构化（如维基百科）和非结构化信息的新知识源被用于推荐过程。后一类包括所有用户产生的内容，例如评论和标签，最终产生比过去可用的更为丰富的项目代表。

本文包括近期将丰富项目描述用于推荐系统建立的论文选取。我们在部分三中列举出这些论文。在部分二中，我们简单地回顾这一领域地历史，讨论近期趋势，并概述未来潜在发展方向。

2. 基于内容的推荐系统的过去、历史和未来

2.1 历史发展

基于内容的过滤的基本思想可以追溯到 1960 年代。那时被称为信息的选择性传播（Hensley 1963）。这里的目标是将新到达的信息项与存储在用户配置文件中的接收者的假定兴趣相匹配来分发信息。另一个基于内容的过滤的根源在于信息检索领域。基于内容的方法通常所依赖的文档编码是发展于这个领域的（Salton 和 McGill 1986）。在 Web 时代，基于内容的技术后来成功被应用于不同的领域，例如感兴趣网页的个性化推荐（Pazzani 等人 1996）。

然而很快发现纯基于内容的协同方法在很多应用场景下有限制，特别是与协同过滤系统进行比较时。一个主要的问题是基于内容的过滤系统大多不考虑推荐过程中的项目质量。例如，一个基于内容的电影推荐可能会推荐和这个用户过去喜欢的低质量的电影。协同过滤方法经常考虑到这样的信息，通过明确地考虑项目排名或者是间接地考虑在隐式反馈场景中地项目受欢迎程度。

数十年来的学术研究聚焦在寻找准确性最高的最佳模型。纯基于内容的过滤方法大多看上去过时了，并且他们主要用于提升协同过滤方法的表现，例如在冷启动情况下（Melville 等人 2002）。然而，随着新知识源的可用性，包括结构化和非结构化的，例如用户生成的内容，迎来了一波利用此类信息来进行更好的预测的混合系统的新提案。此外，在一些新的应用场景下，重要的是推荐项目于相关项目的内容相似性，例如相似产品推荐（Yao 和 Harper 2018）。同样，在自动音乐广播电台中，内容特征可以用于确保电台不会从种子轨道上移开太多。最后，内容信息同样确保更好解释的产生（Gedikli 等人 2014）。这对于公平透明的推荐系统而言是越来越重要的。

2.2 最近趋势

在过去的十年间，我们观察到很多工作在将新类型的附加信息（例如元数据或用户生成的内容）用于推荐或是提出新算法来处理可用信息。

2.2.1 与数据相关的趋势

链接的公开数据。传统上，基于内容的推荐系统使用元数据，项目描述，或

者是文本项目的全文索引（Belkin and Croft 1992）。今天，链接的公开数据的介绍提供了新方法来用外部数据源进行项目描述的扩展（Di Noia 等人 2012; Musto 等人 2017b）。例如，Passant（2010）的早期工作使用了 DBpedia 作为电影推荐的外部数据源。自此，链接的公开数据被用于丰富很多应用领域的项目描述，包括电影和书本推荐（Musto 等人 2017a），电影和声音推荐（Oramas 等人 2017）。链接数据的显示语义的另一优势是它们可以被用于优化例如多样性的特点指标（Musto 等人 2017a）。此外，链接数据可以作为生成解释的基础，从而增加推荐透明性（Musto 等人 2019）。总体而言，来自 LOD 云的数据被成功应用于多个领域的推荐，并且存在将 LOD 信息和其他类型的附加信息连接起来的很多机会，如 Orasmas 等人（2017）的电影推荐。

用户生成的内容（UGC）。随着参与式 Web（Web 2.0）的兴起，各种类型的 UGC 变得可用，例如产品评论，用户标签和论坛讨论。很多尝试将此类信息用于推荐文本的早期工作聚焦在通过各种方式使用用户提供的标签，例如增强用户和项目资料，或向用户解释推荐建议（Jaschke 等人 2007; Vig 等人 2009）。在其他工作中，研究者聚焦在用户评论，并尝试从中提取各种类型的可被用于推荐过程的信息，包括语义信息或用户意见和情感。与基于标签的方法一样，基于 UGC 的推荐方法不仅使用附加信息来提高推荐质量，而且可向用户解释推荐意见。此类方法的深入评论可以在 Chen 等人的文章中找到（2015）。

视觉和多媒体特征。由于基于内容的过滤方法传统上是基于文本的，非文本对象经常用元数据描述来表示。然而，在图片和视频分析上的进步使得用从对象中提取到的特征来表示多媒体对象成为可能。许多特征实际上很难以文本形式表示，例如纹理或风格。例如 McAuley 等人(2015)在项目图片上训练了卷积神经网络来学习不同的视觉特征维度是如何在不同的产品类型间相互关联。由此产生的样式空间可用于推荐，例如搭配特定鞋子的裤子。另一个例子是 Elahi 等人(2017)从电影中提取了例如颜色，纹理，动作和光线的低级特征来搭建混合推荐系统。在 Deldjoo 等人（2018）做的研究中，他们发现这些类型特征可以比从电影中通过事先训练好的神经网络提取到的语义信息提供更好的推荐。对于未来，我们期待视觉和多媒体特征会被用于多个数据可用的领域场景的推荐，例如广告，电商和游戏。

异构信息网络。最后，一些研究旨在同时连接多种类型的附加信息。在这些方法中，可用的数据被异构信息网络（HIN）表示。其中对象之前的语义关系被网络节点中的不同类型的关系所表示。早期使用社交网络进行推荐的研究关注在关联用户间的建模信任（Golbeck 2016）。一个最近的例子是 Kleinerman 等人（2018）在相互推荐场景下，例如网上约会，使用了这样的网络来生成解释。通常，随着可用知识源的增加，基于 HIN 的方法会变得越来越重要，例如创建多维用户资料（Jannach 等人 2017）。

2.2.2 与算法相关的趋势

基于元路径的方法。从算法的角度看，HIN 中不同类型的对象间的类型化连接启发了基于元路径的算法的发展。其中元路径通过对象类型间的标记关系序列来连接两个对象（Sun 等人 2011）。例如，Shi 等人（2015）证明了基于元路径的算法在多个推荐领域对于 HIN 是有用的，像是有用户、电影、演员和导演等类型的电影推荐，或是客户、产品、企业和位置间的推荐。在另一种方法中，Yu 等人（2014）利用了基于元路径的潜在特征来从多种路径表示用户和项目之间的连接。从项目的隐式反馈中学到的用户加权的项目和其他实体间的异构关系提供了个性化推荐。随着 HIN 变得越来越丰富和常见，我们期待基于元数据的方法会在未来得到更多关注。

新元数据编码。尽管 TF-IDF 或者潜在空间表示仍然常用于基于内容的协同方法中的文本项目描述的编码，近年来出现的嵌入成为了一个可被用于识别文本中术语间潜在联系的很有希望的新编码方式。例如，Meta-Prod2Vec（Vasile 等人 2016）是一个在使用音乐播放列表和收听数据作为协同过滤的交互的混合模型中对于基于序列的项目推荐来计算项目元数据的低维嵌入的一种方法。另一方面，Kula（2015）结合了用户和项目的元数据在时尚领域训练了用于生成推荐的嵌入。在他们的应用场景下，专门使用了嵌入来使得转移学习能够改进基于内容的推荐中的用户和项目的冷启动问题。由于嵌入的灵活性，我们期待看到更多结合元数据和文本和非文本内容特征的复杂模型。

深度学习。深度学习（DL）方法提供了一个灵活的框架来涵盖现存的项目和用户数据中的框架，包括外部数据源（Musto 等人 2018）和非文本特征（Deldjoo

等人 2018），从而学习更好的特征表示。深度神经网络的另一个优势是其对于时间和顺序方面的建模能力，使得他们适用于电商（Hidasi 等人 2016）和新闻（Song 等人 2016）等基于会话的和序列感知的推荐任务。此外，他们可以被训练以从用户评论（Lu 等人 2018）中产生丰富的解释。最近，Zhang 等人（2017）证明了基于越来越多的文本和多媒体内容特征的异构来源、外部知识源和交互数据的联合表示学习的潜力。另一个最近的研究方向是提高深度学习模型的可解释性，以实现透明可解释的推荐（Seo 等人 2017b）。

2.3 未来趋势

在推荐系统中，我们预见了许多研究内容、元数据和附加信息作用的研究方向。

透明推荐。近年，随着复杂机器学习模型使用的增加，我们观察到人们对可解释 AI 给予了更多的兴趣，尤其是在问责和解释是重要属性的复杂人类决策的场景下，例如健康。在推荐系统中，对解释下的探索已经持续了多年（Nunes 和 Jannach 2017；Tintarev 和 Masthoff 2008）。多种类型的附加信息也已经被用于现有的解释方法中。尽管现有的方法经常使用附加信息来突出推荐项目的特征（Gedikli 等人 2014；Vig 等人 2009），未来工作的一大领域是用附加信息来解释类似黑盒算法的内部机制。例如，可以标记派生模型的内部成分（如神经网络的特定层或分解矩阵的潜在因子），或者通过使用描述性特征来解释推荐算法的整体行为。在这种情况下，一个开放的研究问题涉及用于解释过程的特征选择和可能的组合，如描述性特征（项目属性和精选元数据）、用户生成的内容（文本评论和标签）、协同信息和其他来源。

新算法和任务。由于其对复杂文本内容的依赖，基于内容的推荐传统上收到计算语言学和自然语言处理技术（NLP）领域的启发。例如，词嵌入在许多 NLP 任务中的成功使用，使得我们看到它在推荐系统中被大范围的应用。我们期待这样方法和技术的转移在今后会持续。例如，上下文嵌入在某些 NLP 任务上的表现由于最新技术（Seo 等人 2017a），并且可以看出它们被应用于基于内容的推荐算法的前景。除了这样算法的进步，另一项富有成果的研究方向是调查内容信息在推荐任务的替代类型上除了标准化项目评分和评分预测的作用（Jannach 和

Adomavicius 2016)。例如，生成相关或相似产品的推荐（Yao 和 Harper 2018），或推荐项目的组合或排序。

新内容类型。一些领域富含非文本内容。这是目前仍然很难高效利用的信息，例如多媒体内容。到目前为止，多媒体推荐大多依靠隐式和显示的用户反馈。但我们期待深度学习算法的快速发展可以在未来带来更结构化和更高效的方法以提取和使用语义和风格特征，如同 Messina 等人（2018）所做的。公开研究问题这里包括从多媒体内容中如何提取和提取哪些特征，以及如何将它们和用户生成内容以及用户偏好数据相结合来提供更相关的推荐。一个对于跨领域推荐的更加全面的理解，即一个领域的内容如何帮助另一领域的推荐，在这里也很重要。另见 Hernandez-Rubio 等人（2018）。

3. 本期论文

本期论文延续了上面讨论的许多近期趋势。它们使用各种附加信息包括用户生成内容和链接公开数据以及从多媒体对象中自动获取的特征。从算法的角度，这些论文经常依赖深度学习方法来进行特征提取或推荐。许多应用领域都被考虑到，包括传统的音乐或新闻推荐，和新颖的小说或艺术品推荐。

论文基于内容的艺术品推荐：整合具有神经和手动设计的视觉特征的绘画元数据（Messina 等人 2018）介绍了一种新颖的艺术品推荐方法。作者结合了多种信息源，包括项目元数据和通过深度神经网络提取到的低级视觉特征，并且通过线上艺术商店提供的真实世界的实验数据显示了此结合方法的优势。

论文电影基因组：减轻电影推荐中的新项目冷启动问题（Deldjoo 等人 2019）采用了类似的想法，并在电影推荐系统中结合了各种特征，包括元数据以及低级视觉和听觉特征。他们进一步将基于内容的模型与协同信息结合在一个混合模型中。线下的分析和初步的用户研究都确认了该模型针对新项目冷启动问题的有效性。

论文在线新闻行业的情感推荐系统：情绪如何影响阅读选择（Mizgajski 和 Morzy 2018）研究了情绪在推荐过程中的角色。基于一组情感项目特征，提出了一个用于新闻项目推荐的多维情绪模型。实验揭露了考虑情绪反应的潜力，并在他们的案例中增强的推荐方法带来了更高的点击率。

论文基于从用户评论中提取的项目意见的推荐系统的比较分析（Hermandez-Rubio 等人 2018）关注用户生成内容。它对利用从用户评论中提取到的信息的推荐系统进行了深入的研究，并旨在确定每一步的最佳技术。此外，这篇论文列出了一些基于评论的推荐系统的未来方向，并向研究者贡献了很多有用的资源，包括特定领域的词汇和词典。

与数据相关的方面和链接公开数据的使用是论文的用跨领域协同过滤处理用户冷启动问题：利用矩阵分解中的项目元数据的重点（Fernandez-Tobias 等人 2019）。在他们的工作中，作者专门研究了基于链接公开领域的元数据的使用，来改进跨领域协同过滤的矩阵分解模型。他们的实验强调了考虑此类知识源的潜在价值，不仅做出了更精确的推荐，尤其是在冷启动情况下，还平衡了准确性和多样性。

论文连接主义的野蛮推荐：关于用于个性化课程指导的神经网络的实用性、可编码性和可扩展性（Pardos 等人 2019）是依靠嵌入去编码元数据的研究示例。作者提出 Word2Vec 的改进 Course2Vec 来建立大学课程的向量空间表示。他们的最终目标是向学生推荐课程序列。他们的实验不仅展示了方法的有效性，还证明了推荐是可理解的。这是现实应用的主要前提。

最后，论文用于自动音乐播放列表连续的特征组合混合推荐系统（Vall 等人 2019）对音乐推荐做出了贡献。作者介绍了两种特征组合的混合推荐系统。系统将精选音乐列表的协同信息和歌曲特征结合在一起。他们的实验表示提出的方法与协同过滤系统相比带来了有竞争性的结果。当训练数据稀疏的时候，混合方法的表现会优于协同过滤系统。因此这项研究为冷启动下的内容信息的价值提供了证据。