分类号_____ 密级_____

UDC _____ 学号_____

太原理工大学

毕业设计（论文）

论文题目 隐语义模型的图书推荐

Thesis Topic A book recommendation system based Latent Factor Model

学 生 姓 名	刘港
学 号	2015003588
所 在 院 系	物联网工程
专 业 班 级	1502
导师姓名职称	林福平 讲师
完 成 日 期	2019 年 6 月 17 日

2019 年 6 月 17 日

隐语义模型的图书推荐

摘　要

推荐系统是一种考虑用户差异，为用户提供个性化服务的信息过滤技术，它能够很好地缓解信息过载问题所带来的挑战。以个性化推荐技术为代表的推荐系统能够提供一种有效的机制，使用户可以高效获取需要的信息。

隐语义模型推荐算法由于自适应性能好、预测准确性高，在评分预测、TopN 推荐等两种典型推荐算法应用场景下都展示了可观的应用前景。本文针对隐语义模型推荐算法在图书推荐领域展开进一步研究，主要内容包括：

1、为明确本文的研究内容和研究思路，全面介绍了隐语义模型的发展背景和研究现状，介绍了推荐系统邻域常用的评测指标，比较分析了推荐系统常见推荐技术的优缺点，为后期研究工作的展开奠定基础。

2、详细阐述了隐语义模型的工作原理和学习算法，介绍了基本的隐语义模型 LFM、引入偏置项的隐语义模型 Bias-SVD、融合邻域的隐语义模型 SVD++和考虑隐式反馈的隐语义模型 AsymmetricSVD 等算法，并在 MovieLens 数据集上比较分析了几种模型的推荐效果，以及受不同参数的影响情况，总结各个算法的特点。

3、对隐语义模型的训练速度做出改进，提出了改进学习率的梯度下降法和自适应学习率 Adam，并使用 MovieLens 数据集训练进行比较。

4、针对图书推荐的特殊性，对隐语义模型也进行了相应的改进，提出了根据隐语义隐类 F 对图书候选标签的扩展 Tag 算法，对图书推荐排名提出了根据时间上下文对图书推荐的顺序做出调整的排序算法。最后设计了一个简单的基于隐语义模型的图书推荐系统。

关键词： 图书推荐；隐语义模型；SVD++

A book recommendation system based Latent Factor Model

Abstract

The recommendation system is an information filtering technology that considers user differences and provides personalized services for users. It can well alleviate the challenges brought by information overload problems. The recommendation system represented by personalized recommendation technology can provide an effective mechanism for users to efficiently obtain the required information.

The implicit semantic model recommendation algorithm has a good application prospect under the two typical recommendation algorithm application scenarios, such as scoring prediction and TopN recommendation, due to its good adaptive performance and high prediction accuracy. In this paper, the implicit semantic model recommendation algorithm is further studied in the book recommendation neighborhood. The main contents include:

In order to clarify the research content and research ideas of this paper, the development background and research status of the hidden semantic model are introduced comprehensively. The evaluation indexes commonly used in the neighborhood of recommendation system are introduced. The advantages and disadvantages of common recommendation techniques of recommendation system are compared and analyzed. The foundation of the research work is laid.

The working principle and learning algorithm of the implicit semantic model are expounded in detail. The basic implicit semantic model LFM, the implicit semantic model Bias-SVD which introduces the bias term, the implicit semantic model SVD++ considering implicit feedback and the fusion neighborhood are introduced. Implicit semantic model AsymmetricSVD and other algorithms, and comparative analysis of the recommended effects of several models on the MovieLens dataset, as well as the impact of different parameters, summarizing the characteristics of each algorithm.

The improvement of the training speed of the implicit semantic model is proposed. The gradient descent method with improved learning rate and the adaptive learning rate Adam are

proposed and compared using the MovieLens data set training.

According to the particularity of book recommendation, the implicit semantic model is also improved accordingly. The Tag algorithm based on the book label to explain the implicit semantic implicit class F is proposed. The book recommendation ranking is based on the time context. The adjusted sorting algorithm. Finally, a simple book recommendation system based on implicit semantic model is designed.

Key words: Books recommended; Latent Factor Model; SVD++

图目录

表目录

目录

1. 绪论

1.1 选题背景

随着信息技术和互联网时代的发展，网络中的信息也指数级地极速增长，每天都会有大量的信息产生在网络中，形成了浩瀚的信息海洋。这些大量复杂的信息远远超过了人类处理信息的能力，我们把这种现象称之为“信息过载“。人们从信息匮乏的时代逐步走入了信息过载(information overload)的时代。在这个时代，无论对信息的生产者和信息的消费者都提出了更多的挑战：作为信息的生产者，如何让自己生产的信息脱颖而出，更好地受到广大用户的关注，这是信息生产者在大量信息的网络中难以解决的问题。而对于信息的消费者来说，最重要的如何在大量的信息中找到自己感兴趣的信息。

对于这种信息过载的现象，有无数的科学家和工程师提出了很多优秀的解决办法。其中最被人们熟识的解决方案是搜索引擎。以谷歌为代表的搜索引擎可以让用户能通过关键词来找到自己的所需要的信息，这在一定程度上解决了信息过载和用户的需求。但是，搜索引擎一方面需要用户主动地提供准确的关键词来寻找信息，面对用户无法提供准确的关键词的时候，搜索引擎也束手无策。另一方面，搜索引擎面对相同的关键词，用户得到的搜索结果是一样的，而不同的用户对相同的关键词显然有不同的需求。用户也需要在大量的返回信息中进行筛选，这大大加大了用户寻找所需要信息的时间，所以搜索引擎也无法满足用户个性化的需求。

在这种情况下，推荐系统就是解决这一问题的重要方法。与搜索引擎一样，推荐系统也是一种帮助用户快速找到需要信息的方法。但与搜索引擎不一样的是，推荐系统不需要用户提供主动明确的关键词，而是根据用户的历史行为数据和信息本身的内容进行建模，从而向用户推荐他们需要和感兴趣的信息。

而近年来，随着大数据和云计算技术的兴起，推荐系统已经越来越被应用到人们的日常生活中。如电商的商品推荐、社交软件的好友推荐、舆论网站的新闻推荐等，推荐系统正在悄然改变着我们生活的方方面面，具有广大的发展和研究前景。

1．2　研究现状

1．2．1　推荐算法的研究现状

在推荐系统中，最核心的推荐算法。推荐算法基于用户的历史行为数据或其他附加信息，建立一种推荐模型，对用户可能感兴趣的物品进行推荐。推荐算法的应用方式主要分为两种：评分预测和 Top-N 推荐。评分预测是预测用户对没有行为的物品可能的评分，是推荐系统研究的热点。而 Greg Linde 指出推荐的目的是找到用户最有可能感兴趣的物品，而不是预测用户对该物品产生行为后会给出怎样的评分，因此 Top-N 推荐更符合实际的需求。

目前主流的推荐算法根据不同的数据来进行设计，常见的算法有基于用户行为数据的算法、基于用户标签数据的算法、基于上下文信息的算法、基于社交网络数据的算法。

基于用户行为数据的算法是推荐系统中最热门的算法，学术界把一般称为协同过滤算法。协同过滤的思想在很多利用用户行为数据的算法中得以充分体现，如基于邻域的算法、隐语义模型、基于图的模型等。基于邻域的算法主要分为两种：基于用户的协同过滤算法（UserCF）和基于物品的协同过滤算法（ItemCF），目前对这两种算法的研究已趋于成熟，业界也以这两种算法作为推荐系统的基础。隐语义模型是 2006 年由 Simon Funk 提出，利用机器学习的方法，解决了当时 SVD 分解算法的高时空复杂度。基于图的模型算法是因为用户的行为数据很容易用二分图来表示，将用户行为转化为二分图模型后，研究者采用了基于随机游走的 PersonalRank 算法来计算图中顶点的相关性来获得推荐。

学术界在协同过滤算法趋于成熟后便寻找使用更多的附加信息来提高推荐系统的预测准确率，推荐系统除了用户行为数据外还有很多附加的信息，如标签信息、时间上下文信息、社交网络信息等。如何将这些附加信息融入现有的推荐模型，以此来提高推荐系统的准确度，是目前业界研究的方向，利用附加信息来改进推荐系统也具有广泛的研究前景。

1．2．2　隐语义模型的研究现状

自 2006 年 Netflix Prize 比赛举办以来，隐语义模型逐渐成为推荐系统领域最热门

的研究话题。该算法最早在文本挖掘领域被提出，用于寻找文本的隐含语义。它的核心思想是通过隐含特征来联系用户和物品。隐语义模型是一种有效的隐含语义分析技术，其在 Top-N 推荐中的算法过程分为三个部分：将物品映射分为 K 个隐类、计算用户对隐类的兴趣度、选取用户兴趣最高的隐类的物品推荐给用户。[1]目前，研究界对隐语义模型进行了充分的研究，其中研究的方向集中在以下的几个方面：

(1).矩阵分解的优化

隐语义模型的基础之一是矩阵分解(Matrix Factorization，简称 MF)，它是利用用户的行为历史数据信息，输入一个由 m 个用户对 n 个物品的评分数据所构成的 m×n 的用户评分矩阵。而大多数实际情况是所输入的评分矩阵维度太高，给计算和存储都带来了巨大的困难。而矩阵分解技术能对输入的评分矩阵做降维处理，将用户和物品的信息映射到 K 维的因子空间中，从而建立用户和物品之间的联系，并且能对这一空间做出很好的解释。[2]

矩阵分解技术是一种重要的机器学习技术，其中在推荐算法的应用中，最常用的是奇异值分解（Singular Value Decomposition，简称 SVD），它在早期被用于信息检索中的潜在语义模型[3]，而在 2006 年 Netflix 举办的推荐算法竞赛中，Simon Fun 提出了 Funk-SVD 算法，该算法被称为隐语义模型算法。[4]其在一定程度上解决了传统的 SVD 算法在时间复杂度和空间复杂度上的弊端，但其本质还是奇异值分解。奇异值分解需要将稀疏的评分矩阵的缺失值进行补全后进行计算，但由于受高维矩阵的影响，奇异值分解的时间、空间复杂度都变得异常高，无法在大规模的数据集上使用。针对这一问题，隐语义模型使用了机器学习的方法，采用优化目标函数(如最小化均方根误差、平均绝对误差等)来学习隐语义模型中的参数，其中的学习方法主要分为两种：梯度下降法(Gradient Descent，SGD)和交替最小二乘法(Alternating Least Squares，ALS)。[5]

(2).利用上下文信息

目前在推荐系统方面主要集中研究如何联系用户兴趣和物品，将最符合用户兴趣的物品推荐给用户，但很多算法都忽略了一点，就是用户所处的上下文(context)。这些上下文包括了用户所处的时间、地点、心情等，这些信息对于推荐系统是非常重要的。研究者将时间上下文信息、地点上下文信息等额外信息融入使用的推荐系统模型，大大地提高了推荐系统的预测能力。

时间是一种重要的上下文信息，对用户兴趣有着深入广泛的影响。一般来说，时间信息对用户的兴趣的影响表现在用户兴趣变化、物品存在生命周期和季节效应等。研究者在隐语义模型中利用时间信息的方法分为两种，一种是将时间信息应用到基于邻域的模型中，另一种是将时间信息应用到矩阵分解模型中。在基于矩阵分解模型引入时间信息后，用户评分矩阵不再是一个二维矩阵，而是变成了一个三位矩阵。研究者仿照二维矩阵的学习方法，加入相应的时间变量，使用梯度下降法进行相应的优化。

在时间上下文信息的使用中，Koren 充分考虑了时间效应的特点，一方面对物品进行时间的处理，主要是对时间进行切片来获得物品在不同时期的评分差异；另一方面是对用户的处理，主要是对用户评分时间来衡量用户的评分时间效应。Koren 等将时间信息与隐语义模型结合，构建了引入时间上下文的隐语义模型 TSVD。[6]

(3).利用物品本身附属信息

目前主流的协同过滤算法是基于用户行为数据进行设计的，该算法不需要使用物品本身的信息，直接使用用户行为数据就能为用户提供推荐，但这也会导致冷启动问题，即在用户无行为的初始状态下无法给出好的推荐。并且物品在推荐系统中拥有多种属性特征，如价格、标签、评价等，这些属性对推荐系统的推荐也有着重要的作用。

目前在隐语义模型的研究中，研究者发现了附加属性对推荐结果有着很大的帮助。Ahmed[7]等提出了 LFUM 算法，该算法结合了物品属性特征和隐语义模型，能够同时处理物品固有属性和变化属性问题，采用 BPR 排序算法实现用户偏好矩阵和 LFM 的混合推荐。Zhan Chenyi 等利用主题模型 LDA 提取文献的主题分布，提出了内容+属性的隐语义模型，实现了隐语义模型在异构学术网络中的推荐。

隐语义模型由于其算法的限制，无法很好地解释训练出来的隐类，但可以利用物品的附属信息来进行解释，比如说物品的标签信息。目前融合物品附加信息的隐语义模型正在逐渐发展，物品的附加信息对基于隐语义模型算法的预测准确度有着重要的提升，其对推荐系统的研究有着良好的促进作用。

(4).利用隐性反馈行为信息

大多数推荐算法都是通过用户的行为数据进行分析设计的，用户行为数据在推荐系统中一般分为两种：显式反馈行为(explicit feedback)和隐式反馈行为(implicit feedback)。显性反馈行为包括用户明确表示对物品喜好的行为，如评分。隐语义模型在显式反馈数

据中能达到很好的预测效果。

而隐语义模型在隐式反馈数据中关键的问题是如何给每个用户生成负样本（用户对什么物品不感兴趣）。对于这个问题，Rong Pan 进行了深入的探讨，对负样本的采样提出四种方法。其中效果最好的是对于一个用户随机采样没有过行为的物品作为负样本，并保证每个用户的正负样本数数目相当。在之后 2011 年雅虎的推荐比赛中，研究者发现对每个用户采样负样本时，还应选取哪些很热门而用户没有行为的物品。

Koren 在 Netflix 评分数据中不仅使用了用户评分数据进行隐语义模型训练，还提出了用户是否对物品进行过行为的隐式行为，并将评分矩阵转成一个二值矩阵进行训练，并且进行模型融合提高了评分预测的准确性。[8]Rendle 等人也提出了贝叶斯排序模型，该模型将用户行为矩阵转为低阶稠密矩阵，基于贝叶斯后验估计量为目标函数训练模型，实现了隐式反馈二值矩阵表达方式的改进。[9]

在目前来看，利用隐式反馈行为信息的算法还在逐渐发展中，大部分以布尔模型、BPR 模型来进行体现，隐语义模型在利用隐式反馈行为信息中也有着很大的发展潜力。

(5).模型融合

目前在一个实用的推荐系统中，如果仅使用一个推荐模型，那么推荐效果是不明显的。Netflix Prize 大赛的获胜队伍都是融合了上百个模型的结果，模型融合对提高评分预测的精度至关重要。目前模型融合有很多不同的技术，比如线性融合、利用人工神经网络进行融合等，模型融合问题是一个典型的回归问题，大量的回归算法都可以用于模型融合。

常见的模型融合有模型级联融合和模型加权融合等。模型级联融合是在已经有一个预测器的情况下，在此预测器的基础上设计下一个预测器来最小化损失函数。级联融合与 Adaboost 算法类似，该方法将产生的新模型按照一定的参数加到旧模型上，从而使训练集误差达到最小，该方法被大量用于简单的预测器。模型加权融合是存在 K 个不同的预测器，将他们组合起来达到最小的预测误差。最简单的模型加权融合算法就是线性融合，即最终的预测器就这 K 个预测器的线性加权。

模型融合在目前得到了广泛应用。Bell 等在 Netflix Prize 竞赛中提出了隐语义模型与 KNN 模型的模型融合，KNN 模型只关心用户之间和物品之间的相关度，而不考虑用户-物品之间的联系，选取相关度高的物品或者用户进行近邻推荐，具有良好的局部作

用。而隐语义模型将用户-物品联系起来，考虑了数据的全局作用。[10]但是隐语义模型对局部的小规模数据集的处理存在不足，所以两者进行模型融合可以避免各自的不足来提高模型预测的准确性。

1．3　机器学习与 tensorflow

机器学习是一门人工智能的科学，它致力于研究如何通过计算的手段、利用经验来改善系统自身的性能。机器学习的研究的主要内容是关于在计算机上从数据中产生模型的算法，有了学习算法就能产生模型，模型会给我们提供相应的判断。由于传统的 SVD 分解的高时空复杂性，Simon Funk 提出了使用机器学习的方法来解决矩阵分解的高复杂性。Simon 提出既然使用 RMSE 来作为评测指标，如果使用机器学习能找到合适的矩阵来最小化训练集的预测误差，那么也可以最小化测试集的预测误差。这样引入了机器学习的方法大大降低了模型生成的复杂度。

TensorFlow 是一个基于数据流编程（dataflow programming）的符号数学系统，被广泛应用于各类机器学习算法的编程实现，其前身是谷歌的神经网络算法库 DistBelief。Tensorflow 拥有多层级结构，可部署于各类服务器、PC 终端和网页并支持 GPU 和 TPU 高性能数值计算，被广泛应用于各领域的科学研究。本文使用 Tensorflow 的 API 对隐语义模型进行训练，并将结果使用 Tensorboard 进行展示。

1．4　本文研究内容

本文在了解推荐系统和主流的推荐算法后，针对隐语义模型在评分预测和 Top-N 推荐进行深入的研究，并对隐语义模型的预测准确度和训练速度做出改进。由于本文研究的是基于隐语义模型的图书推荐系统，所以针对图书推荐的特殊性，对隐语义模型也进行了相应的改进，提出了根据图书标签解释隐语义隐类 F、根据时间上下文对图书推荐的顺序做出调整。最后设计了一个简单的基于隐语义模型的图书推荐系统。

全文分为 5 章，每一章的写作内容如下：

第一章：介绍了推荐系统的研究背景和研究现状，着重阐述了隐语义模型的研究现状和研究的几大方向，同时给出了本文主要研究内容和章节安排。

第二章：对推荐系统和推荐算法的常用评测指标做出简单的介绍，之后对主流的推

荐算法做出了分析介绍，并阐述了算法的优缺点。

第三章：针对第二章的隐语义模型在预测准确度和学习效率两方面进行改进，从隐语义模型的预测准确度角度提出了改进的隐语义模型算法，包括 BiasLFM 算法、SVD++算法、ASVD 算法，并通过实验讨论了参数 F 对算法的影响和各算法的性能比较。从隐语义模型学习效率的角度提出了改进学习率的方法和 Adam 优化方法，并在数据集上实验分析各学习方法的学习效率。

第四章：针对基于隐语义模型的图书推荐系统提出两方面的现实设想，包括利用图书标签解释隐语义模型的隐类，利用时间上下文信息优化 Top-N 推荐的推荐排序。最后设计了一个基于隐语义模型的图书推荐模块。

第五章：对本文进行总结和对进一步研究方向的展望。

2. 推荐系统与推荐算法

2.1 推荐系统简介

推荐系统是为了解决信息过载问题而出现的一种自动联系用户和物品的工具，它能够在信息过载的时代帮助用户发现令他们感兴趣的信息，也能将信息推送给对它们感兴趣的用户。与搜索引擎不同，推荐系统依赖用户的行为数据，并根据用户的行为数据做出推荐。所以一般推荐系统都是由前端的页面、后台的日志系统和推荐算法系统组成。

一个完整的推荐系统存在 3 个参与方：用户、推荐系统网站和物品的提供者。一个好的推荐系统不仅能预测用户的行为，而且能够扩展用户的视野，帮助用户发现那些可能会感兴趣，但却不那么容易发现的物品。根据推荐系统应用的方向不同，可分为评分预测系统和 Top-N 推荐。[11]

评分预测系统是根据用户对物品进行评分的行为数据，通过推荐算法或者模型，来预测用户对未评分的物品的评分，从而来衡量用户是否对物品感兴趣，发现用户的偏好。

Top-N 推荐系统是根据用户的行为数据向用户推荐一定长度的物品列表，该推荐系统使用一定的推荐算法和排序方法，把用户最感兴趣的物品排在前面，来方便用户找到他们偏好的物品，尽管研究人员在离线测试中更多地关注评分预测，但 Top-N 推荐更具有现实意义。

2．2　推荐系统常用评测指标

2．2．1　用户满意度

用户满意度是推荐系统最重要的指标，但是用户满意度无法通过离线的方法计算，一般情况用户满意度是通过用户调查和在线实验获得。

在很多在线推荐系统中，用户满意度主要是通过一些用户的行为信息来进行统计计算，比如用户的购买记录、点击率、用户滞留时间等。还有一些网站设计了用户反馈界面来了解用户的满意度，比如豆瓣网络电台中的反馈按钮，通过统计按钮点击的情况就可以度量系统的用户满意度。

2．2．2　预测准确度

预测准确度的度量是推荐算法最重要的离线评测指标。该指标体现了一个推荐系统或者推测算法预测用户行为的能力，绝大多数研究者所讨论的推荐算法都必须评测算法的预测准确度。在计算该指标时需要有一个离线的数据集，该数据集包含了用户的行为数据，将这一数据集分为训练集和测试集，通过在训练集上建立兴趣模型，最后将计算预测行为与测试集中的实际行为的重合度作为预测准确度。

离线的推荐算法一般有两个主要的研究方向：评分预测和 Top-N 推荐。不同的研究方向对于预测准确度的表述方式不同。

评分预测的预测准确度一般通过均方根误差（RMSE）和平均绝对误差（MAE）来计算。在一个测试集中，对于用户 u 和物品 i，令 r_{ui} 是用户 u 对物品 i 的实际评分，$\hat{r}_{ui}$ 是推荐算法对于用户 u 对物品 i 的预测评分，Tset 表示测试集，那么 RMSE 的定义为：

$$\mathrm{RMSE}=\frac{\sqrt{\sum_{u,i\epsilon Test}\left(r_{ui}-\hat{r}_{ui}\right)^2}}{|Test|} \tag{2-1}$$

MAE 采用绝对值来预测误差，MAE 定义为：

$$\mathrm{MAE}=\frac{\sqrt{\sum_{u,i\epsilon Test}\left|r_{ui}-\hat{r}_{ui}\right|}}{|Test|} \tag{2-2}$$

对于一个基于评分预测的算法来说，RMSE 与 MAE 各有优缺点。Netflix 认为 RMSE 加大了对预测物品不准的惩罚（平方项惩罚），所以 RMSE 对系统的评测更加严格。而在基于整数评分的评分系统中，训练的模型如果取整的话，会降低 MAE 的误差。

Top-N 推荐是对于一个用户直接给出了个性化推荐列表，Top-N 推荐的预测准确率由准确率(precison)和召回率(recall)来度量。令 R(u)是根据用户在训练集上的行为给用户作出的推荐列表，而 T(u)是用户在测试集上的行为列表，U 表示用户集合，那么召回率 Recall 定义为：

$$\text{Recall} = \frac{\sum_{u \in U} \left| R(u) \cap T(u) \right|}{\sum_{u \in U} \left| R(u) \right|} \tag{2-3}$$

Top-N 推荐的准确率 Precison 定义为：

$$\text{Precison} = \frac{\sum_{u \in U} \left| R(u) \cap T(u) \right|}{\sum_{u \in U} \left| T(u) \right|} \tag{2-4}$$

Top-N 推荐中推荐列表长度 N 对召回率和准确率有着很大的影响，研究者一般会采用不同长度的推荐列表 N 来计算召回率和准确率，最后画出召回率和准确率的变化曲线。

2．2．3　覆盖率

覆盖率(Coverage)是一个推荐系统对物品长尾的发掘能力，这个评测指标是物品的提供者会很关注的指标。一个好的推荐系统不仅应该给用户提供用户感兴趣的推荐，而且也能让物品更好地受到关注，从物品的角度出发，推荐系统应该更好地发掘物品的长尾(long tail)。

覆盖率有不同的定义方法，其中最简单的是推荐系统给出的所有推荐物品占总物品集合的比例。令用户集合为 U，推荐系统给每个用户推荐的物品列表为 R(u)，总物品集合为 I，那么覆盖率定义为：

$$\text{Coverage} = \frac{\left| \cup_{u \in U} R(u) \right|}{\left| I \right|} \tag{2-5}$$

但上述的覆盖率表述过于简单，对于推荐物品出现的次数无法给出很好的解释，为了更好地描述推荐算法对长尾的挖掘能力，研究者采用信息论和经济学中著名的两个指

标来定义覆盖率：信息熵和基尼系数。这样可以通过物品在推荐列表中出现的次数来描述对挖掘长尾的能力。

令在推荐列表中出现的物品总数为 n，p(i)表示物品 i 的流行度除以所有物品流行度，那么信息熵的定义为：

$$\mathrm{H}=-\sum_{i=1}^{n}p(i)\log p(i) \tag{2-6}$$

对于基尼系数，令i_j为按照物品流行度从小到大排序的物品列表中的第 j 个物品，那么基尼系数的定义为：

$$\mathrm{G}=\frac{1}{n-1}\sum_{j=1}^{n}(2j-n-1)\,p(i_j) \tag{2-7}$$

如果推荐系统的流行度很平均，那么基尼系数会很小，说明推荐系统不仅覆盖率高，而且对长尾的挖掘能力强。

2．3　推荐算法概述

2．3．1　基于内容的推荐算法

基于内容的推荐算法最早在信息检索中提出，之后在引入用户描述信息形成了基于内容的推荐算法。基于内容的推荐算法主要关键分为三点：建立物品特征模型、建立用户兴趣模型、计算用户特征信息和物品特征信息的相似度并进行推荐[12]。

对于物品特征模型的建立，一般是分析每个物品的特征，将物品的特征来表示这一物品。一般基于文本信息的物品，研究者通常使用文本内容的关键词来描述物品的特征，使用 TF-IDF 方法来提取文本中的关键词，并且将若干个关键词组成物品的特征属性 Content(s)。对于用户兴趣模型的建立，研究者通过该用户的行为数据来分析用户的兴趣模型，其分析的过程包含关键词分析技术，从而得到用户兴趣模型 ContentBasedProfile(c)。最后将用户兴趣模型和物品特征模型进行相似度计算，其相似度函数被定义为[13]：

$$\mathrm{sim}(\mathrm{c},\mathrm{s})=\mathrm{score}\big(\mathrm{ContentBasedProfile}(\mathrm{c}),\mathrm{Content}(\mathrm{s})\big) \tag{2-8}$$

其中相似度的计算可以使用余弦相似性来进行计算，最后将结果进行 Top-N 排序

推荐。

基于内容的推荐算法是根据该用户的行为数据和物品的特征属性来做出推荐，不需要其他用户的行为数据进行计算，所以相比于基于协同过滤的推荐算法来说，大规模的用户不会对基于内容的推荐算法造成计算复杂度，每一个用户和每个物品都是相对独立，用户之间的行为不会对推荐结果有所影响。基于内容的推荐算法的另一优势体现在新物品的冷启动上，对于协同过滤算法来说，新物品的没有任何用户评分，所以无法做出推荐，但对于基于内容的算法来说，只要分析了新物品的特征模型，就可与用户兴趣模型比较从而做出相应的推荐。

基于内容的推荐的缺点是特征信息的提取很困难，无法准确地提取一些主观性很强或者多媒体的内容特征。此外该算法只停留在用户当前的行为，并没有挖掘用户可能感兴趣的方面，这会带来内容更多的同质化，降低用户的满意度。

2．3．2　基于邻域的推荐算法

基于邻域的算法是推荐系统最基本的算法，该算法不仅得到了深入地研究，而且还在业界广泛地使用。基于邻域的推荐算法一般分为两类：基于用户的协同过滤算法和基于物品的协同过滤算法。

2．3．2．1　基于用户的协同过滤（UserCF）

基于用户的协同过滤算法是通过分析用户的行为数据寻找到与该用户兴趣相似的用户集合，并将该用户集合内喜欢的并且目标用户没有对该物品有行为的物品推荐给目标用户。基于用户的协同过滤算法的关键是找到相似的用户集合，即计算两个用户之间的相似度。

用户之间的相似度可以采用 Jaccard 公司或者余弦相似度来计算，令用户 u 与用户 v 的相似度为 Sim_{uv}，N(u)表示用户 u 喜欢物品的集合，其中利用余弦相似度计算的定义为：

$$Sim_{uv}=\frac{\left|N(u)\cap N(v)\right|}{\sqrt{\left|N(u)\right|\left|N(v)\right|}} \tag{2-9}$$

若采用两两用户都进行相似度的计算，则时间复杂度为$\mathrm{O}\left(\left|U\right|^2\right)$，这在大量用户时是非常耗时的，所以我们可以建立物品-用户的倒排表，每个用户下保存对它产生过行为的用户，对每个倒排表进行扫描，得到稀疏矩阵$\mathrm{C}[u][v]=\left|N(u)\cap\left|N(v)\right|\right.$，如图 2-1，从而降低算法的时间复杂度。

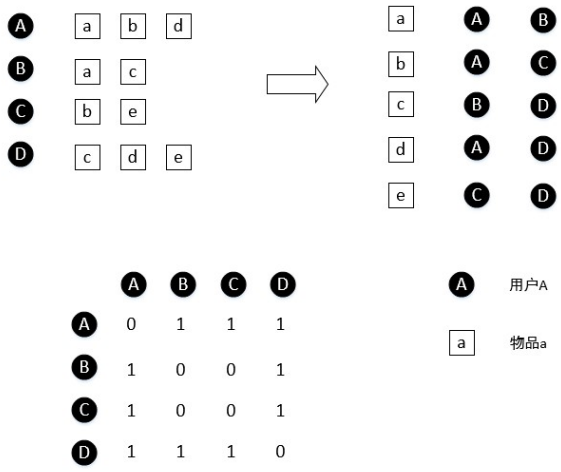

图 2-1 用户-物品倒排图(user-item inversion graph)

在得到用户的相似性后，找出与目标用户最相似的 K 个用户集合是 S(u,K)，计算目标用户对 K 个用户中没有行为的物品的兴趣度 p(u,i)，令 Sim_{uv} 为用户 u 与用户 v 的相似度，r_{vi} 表示用户 v 对物品 i 的兴趣程度，则用户 u 对物品 i 的兴趣度定义为：

$$\mathrm{p}(\mathrm{u},\mathrm{i})=\sum_{v\in S(u,K)\cap N(i)} Sim_{uv}r_{vi} \tag{2-10}$$

在得到用户对物品集合的兴趣度后，取兴趣度高的物品进行推荐。

2．3．2．2　基于物品的协同过滤（ItemCF）

基于物品的协同过滤算法是业界使用最多的算法，它与基于用户的协同过滤算法类似，但该算法考虑的是物品之间的相似度。该算法不是利用物品本身的属性内容来计算物品的相似性，而是利用用户的行为数据，其认为如果很多用户都对某两个物品都感兴趣，那么这两个物品的相似性就更高。令 Sim_{ij} 为物品 i 和物品 j 的相似度，为了避免热门物品的影响和更好地挖掘长尾，物品之间的相似度的定义为：

$$Sim_{ij}=\frac{\left|N(i)\cap N(j)\right|}{\sqrt{\left|N(i)\right|\left|N(j)\right|}} \tag{2-11}$$

与 UserCF 算法相似，ItemCF 算法计算物品相似度时也可以首先建立用户-物品倒排表，即对每个用户都建立一个他喜欢物品矩阵，然后将用户的矩阵全部相加得到物品相似度矩阵。

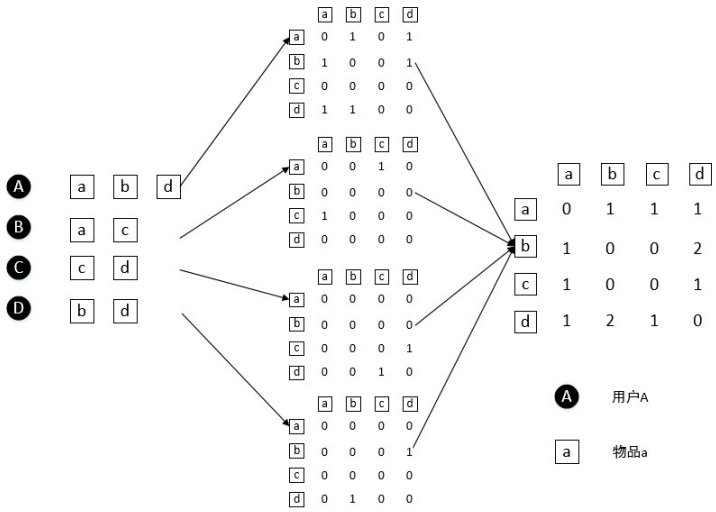

图 2-2 物品-用户倒排表(item-user inversion graph)

该算法通过同时喜欢物品 i,j 的用户个数来表示其相似度，但是有些用户非常活跃，覆盖了系统的大部分领域的书籍，然而他买这些书并不是出于自身需求，所以为了减少用户活跃度带来的影响，John S.Breese 提出了 IUF(Inverse User Frequence)修正物品相似度的计算，修正的物品相似度公式为：

$$Sim_{ij}=\frac{\sum_{u\in N(i)\cap N(j)}\frac{1}{\log\left(1+\left|N\left(u\right)\right|\right)}}{\sqrt{\left|N\left(i\right)\right|\left|N\left(j\right)\right|}} \tag{2-12}$$

在得到物品的相似度后，用户 u 对物品 j 的兴趣定义为：

$$\mathrm{p}\left(\mathrm{u},\mathrm{j}\right)=\sum_{i\in N(u)\cap S(j,K)}Sim_{ij}r_{ui} \tag{2-13}$$

协同过滤算法是推荐系统最普遍的推荐算法，其具有较高的预测准确率和覆盖率，在业界广泛使用，但是它也存在很多问题。该算法在面对稀疏的评分矩阵时，很难计算实际的用户或物品的相似度，也不能有效地预测用户的兴趣。由于该算法是基于用户行为数据的算法，在面对推荐系统冷启动时，没有用户的行为数据，所以也无法做出推荐。

2．3．3　基于图的推荐算法

基于图的推荐算法是把用户和物品之间的联系转换成二分图，该图有表示用户行为数据的一系列二分组(u,i)组成，二分组表示用户 u 对物品 i 产生的行为。该算法的主要思想是计算用户顶点与和该用户顶点没有直连的物品顶点之间的相关性，相关性越高表示用户对该物品越感兴趣。

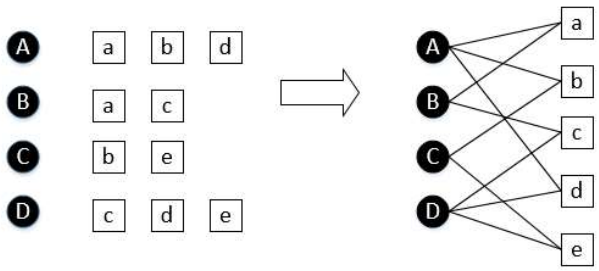

图 2-3 二分图(User-item bipartite graph)

基于图的推荐算法最关键的是计算顶点之间的相关性，对于顶点之间的相关性可以通过顶点之间的路径数、路径长度和路径经过的顶点特征来度量。其中最常见的是基于随机游走的 PersonalRank 算法。

PersonalRank 算法从用户 u 对应的节点V_u在二分图上进行随机游走，当游走在任意节点时，以概率 a 决定是否继续游走，如果选择继续游走，则以均匀分布随机选择下一个节点，如果停止则返回起始节点V_u重新游走。这样经过多次游走每个物品被访问到的概率会收敛到一个数，即为该用户访问物品的概率，概率越高用户对物品越感兴趣。

基于图的推荐算法有着良好的理论解释，但是它有着很高的时间复杂度，对于每一个用户都需要在整个二分图上进行迭代，这不仅不能提供实时的推荐，而且即使在离线推荐是也非常耗时。

2．3．4　基于隐语义模型的推荐算法

基于隐语义模型的推荐最早在 Netflix 大赛中被提出，基于物品的协同过滤是推荐给用户他们有过行为的物品的相似物品，而隐语义模型是将物品进行分类，然后在用户

的兴趣分类中选取物品给出推荐。

1. **传统的 SVD 分解**

在推荐系统中，评分预测一直是一个热门的方向，如何对一个稀疏的评分矩阵进行预测补全，历史上有很多研究者进行了研究。在初始阶段，人们试着从矩阵的特征值入手，使补全后的评分矩阵对于稀疏的评分矩阵来说扰动最小，即两个矩阵的特征值相差不大。最早的矩阵分解是从奇异值(SVD)分解开始，假设存在 m 个用户和 n 个物品，稀疏的评分矩阵 $R^{m\times n}$，将 R 的缺失值进行简单地初始补全，得到的矩阵 R' 使用 SVD 分解得到以下形式[14]：

$$R' = P^T SQ \tag{2-14}$$

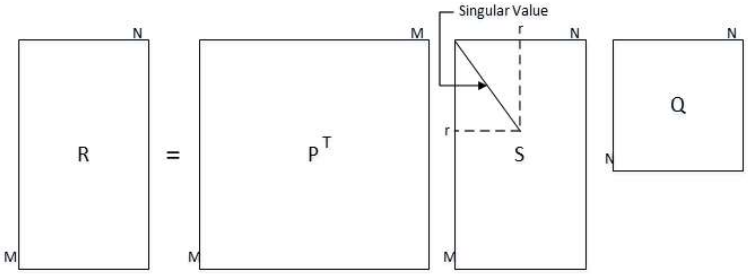

图 2-4 矩阵分解(Traditional matrix decomposition diagram)

其中 P、Q 分别为 m×m，n×n 的正交矩阵，S 为 m×n 的对角矩阵，对角线上每个元素都是矩阵的奇异值，且都是从大到小进行排列。由于奇异值的下降速度很快，前 1%的奇异值就可描述 99%的矩阵特征，所以为了对 R' 矩阵进行降维处理，选取最大的 k 个奇异值组成对角阵 S_K，如图 2-5，得到降维后的评分矩阵为：

$$R_K^{'} = P_K^T S_K Q_K \tag{2-15}$$

其中 $R_K^{'}$ (u,i)即为预测后的用户 u 对物品 i 的评分预测值。

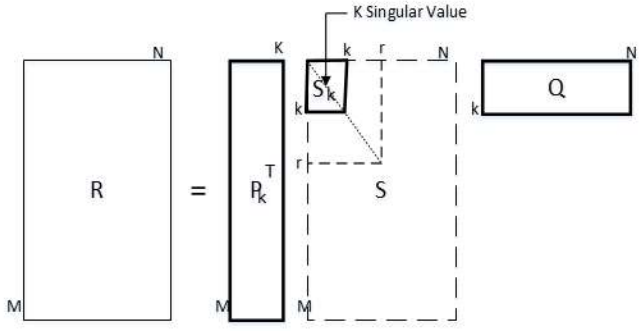

图 2-5 奇异值分解(singular value decomposition diagram)

SVD 分解是早期的推荐系统的矩阵分解方法，但在矩阵分解中，计算 P,Q 矩阵所耗费的时间、空间复杂度极高，实际的系统由于大量的用户和物品，使该算法不再实用。

2. 隐语义模型算法（Latent Factor Model 简称为 LFM）

由于 SVD 分解算法的高时空复杂度，使得它在推荐系统领域未得到广泛关注。直到 2006 年的 Netflix 大赛，Simon Funk 公布了基于机器学习的矩阵分解算法，受到了学术界的极大关注，Koren 将其称为隐语义模型。

隐语义模型是将用户与物品的联系映射到矩阵的 K 维空间，这 K 维空间具有很好的解释性，即将物品进行自动分类，K 维空间表示 K 个不同的物品隐类，如图 2-6。它的核心思想是通过隐含特征来联系用户的兴趣和物品，其采用了基于用户行为数据进行自动聚类，对分类的粒度、维度和类别权重有着很好的解释性[15]。

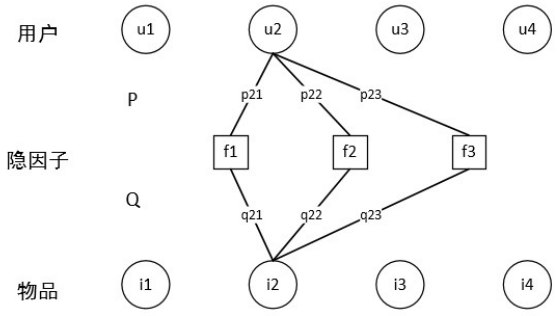

图 2-6 隐含特征解释(implicit characteristic explanation diagram)

(1) 评分预测的隐语义模型

在评分预测的推荐系统中，隐语义模型利用用户评分这种显式反馈数据来预测用户的评分。对于 m 个用户和 n 个物品的评分矩阵，将物品分为 f 个隐类，其将评分矩阵分解为两个低维矩阵相乘表示为：

$$\hat{R} = P^T Q \tag{2-16}$$

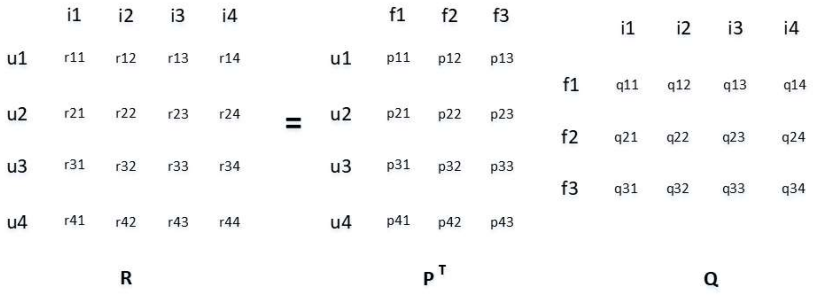

图 2-7 隐语义模型的评分矩阵分解(scoring matrix decomposition of LFM diagram)

其中 P、Q 分为 f×m、f×n 的两个降维后的矩阵。P、Q 矩阵具有很好的解释性，P 矩阵表示用户对不同隐类的感兴趣程度，Q 矩阵表示物品占不同隐类的权重。对于用户 u 对物品 i 的预测值 $\hat{R}(u,i) = \hat{r}_{ui}$ 可通过矩阵相乘计算得知：

$$\hat{r}_{ui} = \sum_{k=1}^{f} p_{uk} q_{ik} \tag{2-17}$$

Simon Funk 认为既然可以用 RMSE 来作为评测指标，如果 P、Q 矩阵能最小化训练集的预测误差，那么也能最小化测试集的预测误差，其定义的损失函数为：

$$\mathrm{C(p,q)} = \sum_{(u,i)\in Train} \left(r_{ui} - \hat{r}_{ui}\right)^2 = \sum_{(u,i)\in Train} \left(r_{ui} - \sum_{k=1}^{f} p_{uk} q_{ik}\right)^2 \tag{2-18}$$

为了避免 P、Q 矩阵的过拟合，令λ为正则化参数，修改后的损失函数定义为：

$$\mathrm{C(p,q)} = \sum_{(u,i)\in Train} \left(r_{ui} - \sum_{k=1}^{f} p_{uk} q_{ik}\right)^2 + \lambda\left(\left\|p_u\right\|^2 + \left\|q_i\right\|^2\right) \tag{2-19}$$

为了最小化损失函数，研究者采用了随机梯度下降法，对于损失函数中的两组参数 $p_{uk} q_{ik}$，对它们分别求偏导，可得：

$$\frac{\partial C}{\partial p_{uk}} = -2\left(r_{ui} - \hat{r}_{ui}\right) q_{ik} + 2\lambda p_{uk} \tag{2-20}$$

$$\frac{\partial C}{\partial q_{ik}} = -2\left(r_{ui} - \hat{r}_{ui}\right) p_{uk} + 2\lambda q_{ik} \tag{2-21}$$

对于每一次的学习迭代，参数都沿着最快速的下降方向推进，参数的迭代公式为：

$$p_{uk} = p_{uk} + \alpha\left(\left(r_{ui} - \hat{r}_{ui}\right)q_{ik} - \lambda p_{uk}\right) \tag{2-22}$$

$$q_{ik} = q_{ik} + \alpha\left(\left(r_{ui} - \hat{r}_{ui}\right)p_{uk} - \lambda q_{ik}\right) \tag{2-23}$$

其中α表示学习速率，在实际的环境中，为了找到全局的最优解和加快收敛速度，初始时对α赋予一个较大的值，对每一步的学习速率α进行衰减，α衰减定义为：

$$\alpha = \alpha * 0.9 \tag{2-24}$$

在评分预测中，首先对 P、Q 两个矩阵进行初始化，初始化矩阵一般以随机数进行填充，根据研究者的经验来说，随机数的值需要与$\frac{1}{\sqrt{f}}$成正比。之后通过随机梯度下降法进行迭代最终得到 P、Q 两个预测矩阵。

(2) Top-N 推荐的隐语义模型

隐语义模型在基于 Top-N 的推荐中，它可以利用隐式反馈数据来对物品进行聚类并给用户进行推荐。隐式反馈数据并没有用户对物品的评分，而是只有用户对物品产生行为的记录。研究者把用户对物品产生过行为的数据成为正样本，否则为负样本。在隐式反馈数据中，LFM 的首要问题是解决如何生成每一个用户的负样本。

对于负样本的采集通常优先选取热门的但用户没有行为的物品，而且负样本采样的数量应该与正样本的数量相平衡。通过实验表明，负样本/正样本比例对 LFM 在 Top-N 中的性能影响最大。固定 f=100，α=0.02，λ=0.01，由表 2-1 可知，随着负样本数目的增加，LFM 的召回率和预测准确率都有很大的提升，但推荐系统的覆盖率随着下降，流行度不断增加，说明负样本/正样本的比例表示了该算法挖掘长尾的能力。

表 2-1 负正样本比例对隐语义模型测评的影响

(influence of negative and positive sample proportion on evaluation of the LFM)

radio	准确率	召回率	覆盖率	流行度
1	20.85%	10.30%	52.75%	6.536
2	23.92%	11.65%	53.57%	6.585
3	25.15%	12.59%	50.55%	6.682
5	26.54%	13.10%	43.28%	6.716
10	27.55%	13.50%	32.97%	6.915
20	27.27%	13.25%	24.96%	7.186

3. 隐语义模型的改进

隐语义模型虽然在推荐系统得到了很好地应用，但仍有很多不足。本章将对隐语义模型进行改进，从隐语义模型的准确度和学习效率出发，提出了改进的隐语义算法和改进的学习方法。

3．1 改进的隐语义算法

3．1．1 BiasSVD 模型

上述的隐语义模型通过隐类将用户和物品联系起来，但该模型只考虑了用户、物品与隐类的关联，而有些属性是用户或者物品所固有的，该属性只影响用户或者物品。比如一个用户习惯对物品比较严格，所以他给的评分偏低，有些物品本身质量很好，所以它的评分普遍都很高。所以为了提供推荐系统的预测准确度，研究者提出了另一个 LFM，我们称之为 BiasSVD 模型，其预测值的定义如下：

$$\hat{r}_{ui} = \mu + b_u + b_i + \sum_{k=1}^{f} p_{uk} q_{ik} \tag{3-1}$$

相应的损失函数为：

$$\mathrm{C}(\mathrm{p},\mathrm{q},b_u,b_i) = \sum_{(u,i)\in Train} \left(r_{ui} - \hat{r}_{ui}\right)^2 + \lambda\left(\left\|p_u\right\|^2 + \left\|q_i\right\|^2 + {b_u}^2 + {b_i}^2\right) \tag{3-2}$$

该模型相比于隐语义模型，增加了 3 个参数。其中μ参数表示训练集中所有评分的全局平均值，这是为了避免整个推荐系统评分差异带来的影响，有些网站用户评分总体都很低，而有的却很高，所以μ参数就可以表示系统本身给用户评分的影响。

b_u 参数被称为用户偏置项，该参数表示了用户自身的习惯对评分没有关系的因素，有些用户对于物品比较严格，他习惯对物品打低分，而有些用户则习惯打高分，所以该参数就避免了用户自身因素对评分的影响。

b_i 参数被称为物品偏置项，该参数表示了物品属性对评分没有关系的因素，有些物品的质量很好，它本身就能得到很好的评分，而有些质量不好的物品也会都被打低分，

所以该参数就避免了物品自身质量对评分的影响。

该模型这三个参数，其中μ参数可以直接得出，而b_u和b_i需要通过学习得到，对损失函数分别求它们的偏导得：

$$\frac{\partial C}{\partial b_u} = -2\left(r_{ui} - \hat{r}_{ui}\right) + 2\lambda b_u \tag{3-3}$$

$$\frac{\partial C}{\partial b_i} = -2\left(r_{ui} - \hat{r}_{ui}\right) + 2\lambda b_i \tag{3-4}$$

对于每一次的学习迭代，参数的迭代公式为：

$$b_u = b_u + \alpha\left(\left(r_{ui} - \hat{r}_{ui}\right) - \lambda b_u\right) \tag{3-5}$$

$$b_i = b_i + \alpha\left(\left(r_{ui} - \hat{r}_{ui}\right) - \lambda b_i\right) \tag{3-6}$$

在 BiasSVD 模型初始时，可令b_u和b_i全为 0。该算法经过研究表明，预测准确率比 LFM 模型有着很大的提升。

3．1．2　SVD++模型

在 BiasSVD 模型中，充分考虑了用户、物品和系统自身属性对评分的影响，但这所有以的考虑都是基于用户的显式反馈数据，模型并没有利用用户的隐式反馈数据，没有考虑用户的历史行为对评分的影响。Koren 提出将基于邻域的方法设计成基于学习的方法，对于基于物品的方法来说，研究者提出可以将 ItemCF 算法改成如下形式：

$$\hat{r}_{ui} = \frac{\sum_{j \in N(u)} w_{ij}}{\sqrt{\left|N(u)\right|}} \tag{3-7}$$

其中w_{ij}表示物品 i 与物品 j 的相似度，但是这里的w_{ij}不是基于 ItemCF 算法中利用余弦相似性等来计算相似度，而是把w_{ij}看成待学习的参数，它可以像隐语义模型一样进行优化学习，其使用的损失函数如下：

$$\mathrm{C}(\mathrm{w}) = \sum_{(u,i) \in Train} \left(r_{ui} - \sum_{j \in N(u)} w_{ij} r_{uj} \right)^2 + \lambda {w_{ij}}^2 \tag{3-8}$$

其中$\lambda {w_{ij}}^2$和隐语义模型一样表示偏置防止过拟合，r_{uj}表示用户 u 对物品 i 的相似

物品 j 的兴趣程度。如果存在 n 个物品，那么该模型的参数个数则为n^2个，在大量的物品情况下，显然该模型的参数数目过多，而且存储它们需要占用很大的空间。所以 Koren 提出对 w 矩阵进行分解，将模型的参数个数下降到 2nf 个，该模型的定义如下：

$$\hat{r}_{ui}=\frac{\sum_{j\in N(u)}x_i^T y_j}{\sqrt{|N(u)|}}=\frac{1}{\sqrt{|N(u)|}}x_i^T\sum_{j\in N(u)}y_j \tag{3-9}$$

其中x_i、y_j是 w 矩阵分解的类似于 P、Q 的矩阵，这样大大减少了参数的个数和存储空间。将上述基于邻域的模型与 BiasSVD 模型进行融合，可得 SVD++模型[：

$$\hat{r}_{ui}=\mu+b_u+b_i+\sum_{k=1}^{f}p_{uk}q_{ik}+\frac{1}{\sqrt{|N(u)|}}x_i^T\sum_{j\in N(u)}y_j \tag{3-10}$$

为了更好地减少参数个数，令 x=q，得到 SVD++模型的定义：

$$\hat{r}_{ui}=\mu+b_u+b_i+q_i^T\left(p_u+\frac{1}{\sqrt{|N(u)|}}\sum_{j\in N(u)}y_j\right) \tag{3-11}$$

对于的损失函数的定义为：

$$\mathrm{C}(\mathrm{p},\mathrm{q},y,b_u,b_i)=\sum_{(u,i)\in Train}(r_{ui}-\hat{r}_{ui})^2+\lambda\left(\|p_u\|^2+\|q_i\|^2+\|y_j\|^2+b_u{}^2+b_i^2\right) \tag{3-12}$$

令$e_{ui}=r_{ui}-\hat{r}_{ui}$，将损失函数对每个学习参数求偏导数，可得：

$$\frac{\partial C}{\partial p_{uk}}=-2e_{ui}q_{ik}+2\lambda p_{uk} \tag{3-13}$$

$$\frac{\partial C}{\partial q_{ik}}=-2e_{ui}(p_{uk}+\frac{1}{\sqrt{|N(u)|}}y_{jk})+2\lambda q_{ik} \tag{3-14}$$

$$\frac{\partial C}{\partial y_{ik}}=-2e_{ui}\frac{1}{\sqrt{|N(u)|}}q_{ik}+2\lambda q_{ik} \tag{3-15}$$

$$\frac{\partial C}{\partial b_u}=-2e_{ui}+2\lambda b_i \tag{3-16}$$

$$\frac{\partial C}{\partial b_i}=-2e_{ui}+2\lambda b_i \tag{3-17}$$

令偏导为 0，可得各参数的迭代公式为：

$$p_{uk} = p_{uk} + \alpha\left(e_{ui}q_{ik} - \lambda p_{uk}\right) \tag{3-18}$$

$$q_{ik} = q_{ik} + \alpha\left(e_{ui}(p_{uk} + \frac{1}{\sqrt{|N(u)|}}y_{jk}\right) - \lambda q_{ik}) \tag{3-19}$$

$$y_{ik} = y_{ik} + \alpha\left(e_{ui}\frac{1}{\sqrt{|N(u)|}}q_{ik} - \lambda y_{ik}\right) \tag{3-20}$$

$$b_u = b_u + \alpha\left(e_{ui} - \lambda b_u\right) \tag{3-21}$$

$$b_i = b_i + \alpha\left(e_{ui} - \lambda b_i\right) \tag{3-22}$$

3．1．3　Asymmetric-SVD 模型

SVD++模型将用户的兴趣描述映射为用户的评分特征矩阵 p 和用户未评分但存在行为的隐式反馈矩阵 y，其实质是对用户特征矩阵 p 进行了扩展，加入了用户历史行为数据对预测评分的影响。但在实际的推荐系统中，存储所有用户评分的矩阵 p 会占大量的空间和资源，所以研究者提出直接使用用户评过分的物品和用户没有评过分但是存在浏览记录的物品属性来表示用户的属性。这个模型被称为非对称奇异值分解模型(Asymmetric-SVD 简称 ASVD[16])。

将用户的评分特征矩阵 p 替换成用户评分物品的特征矩阵 x，令 R(u)表示用户 u 评过分的物品集合,N(u)表示用户 u 没有评过分但有隐式反馈行为的物品集合, x_j、y_j 分别表示相对应集合的物品属性，那么 ASVD 模型定义如下：

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T\left(\frac{1}{\sqrt{|R(u)|}}\sum_{j\in R(u)}\left(r_{ui} - \mu - b_u - b_i\right)x_j + \frac{1}{\sqrt{|N(u)|}}\sum_{j\in N(u)}y_j\right) \tag{3-23}$$

对于 ASVD 模型的学习，采用随机梯度下降法进行学习，损失函数为 RMSE 加上正则化项。相比于其他的隐语义模型，ASVD 具有以下优点：

（1）模型复杂度较低。在用户数量比物品数量多很多的情况下，相比于 SVD++模型，ASVD 模型减少了训练的参数个数，降低了时空复杂度和模型的复杂度。

（2）冷启动处理。对于其他模型来说，一个新用户在初始时是没有对物品进行评分的，所以系统无法给出合适的推荐，但 ASVD 在初始时可以不需要用户进行评分，只要用户产生了隐式反馈行为，就可以进行推荐。

（3）综合考虑用户行为。ASVD 模型综合考虑了用户的显式反馈数据和隐式反馈数据，当用户的显式反馈数据较多时，则|R(u)|比较大，显式反馈数据的作用明显。当用户隐式反馈数据较多时，则隐式反馈数据作用明显。

3．1．4　改进隐语义算法实验与结果分析

3.1.4.1　实验集和参数

实验采用 Netflix 的 Movielens 数据集，该数据集记录了 MovieLens 网站上用户对电影的评分，包括用户 ID 信息、电影 ID 信息和用户对电影的评分等信息。该数据集存在不同规模，本文采用的是记录了 10 万条评分信息的数据集，数据集介绍如下表：

表 3-1 MovieLens 数据集(MovieLens data set)

数据集	用户数	电影数	评分数	数据量
D(10 万)	943	1682	100000	1.9M

本实验将数据集的数据进行随机划分，训练集和测试集的比例为 4:1，实验结果选取 RMSE 和 MAE 的平均值作为测试的衡量标准。实验选取了 LFM、BiasLFM、SVD++、ASVD 四种算法进行对比评估，四种算法都涉及 3 个重要的参数：

(1)、隐因子个数 f：隐语义模型的核心就是将用户和物品的联系映射到 f 维的隐空间，该空间的解释就是对物品进行聚类，所以隐因子个数就是物品聚类的个数。它控制着物品分类的粒度，隐因子个数 f 越大，分类的粒度就会越细，保留的信息就会越多，但相应的时空复杂度就越高。隐因子个数的选择不仅要考虑实际情况，还要考虑模型的特点。

(2)、学习速率α：学习速率决定着模型训练收敛的速度和精度。如果学习速率过大，模型训练的收敛速率会比较快，但是有可能跳过最优解陷入局部最优，如果学习速率过大。模型训练收敛速度会比较缓慢，训练所耗费的时间会增大，但精度有所提升。

(3)、正则化系数λ：正则化系数是为了避免模型过拟合而引入，不同模型需要根据

自身来选择适合的正则化系数。如果该系数选择过大，那么会欠拟合，如果选择过小则会陷入过拟合。

3.1.4.2 隐因子个数 f 对各算法预测误差的影响

隐因子个数决定了物品分类粒度和模型降维程度，实验将探讨不同的 f 对各算法的影响。实验中设置相同的学习速率$\alpha = 0.002$和正则化系数$\lambda = 0.01$，实验迭代次数为 500 次。在 MovieLens 数据集中，不同的 f 值得到的预测误差 RMSE 和 MAE 如表 3-2，各算法迭代过程中的 RMSE 变换如图 3-1，各算法迭代过程中的 MAE 变换如图 3-2。

表 3-2 隐因子对隐语义算法中的影响误差(influence errors of hidden factors on LFM algorithm)

	LFM		BiasLFM		SVD++		ASVD	
F	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
20	0.99546	0.79513	0.95706	0.75119	0.94004	0.74551	0.92581	0.73193
40	0.99543	0.79510	0.95725	0.75108	0.94017	0.74549	0.92525	0.73155
60	0.99540	0.79515	0.95716	0.75118	0.94092	0.74548	0.92501	0.73106
80	0.99545	0.79516	0.95725	0.75109	0.94019	0.74552	0.92574	0.73172
100	0.99548	0.79516	0.95716	0.75107	0.94095	0.74551	0.92516	0.73172
150	0.99548	0.79517	0.95737	0.75108	0.94085	0.74552	0.92509	0.73143
200	0.99545	0.79516	0.95726	0.75109	0.94086	0.74549	0.92568	0.73193
500	0.99548	0.79515	0.95715	0.75108	0.94083	0.74549	0.92558	0.73129

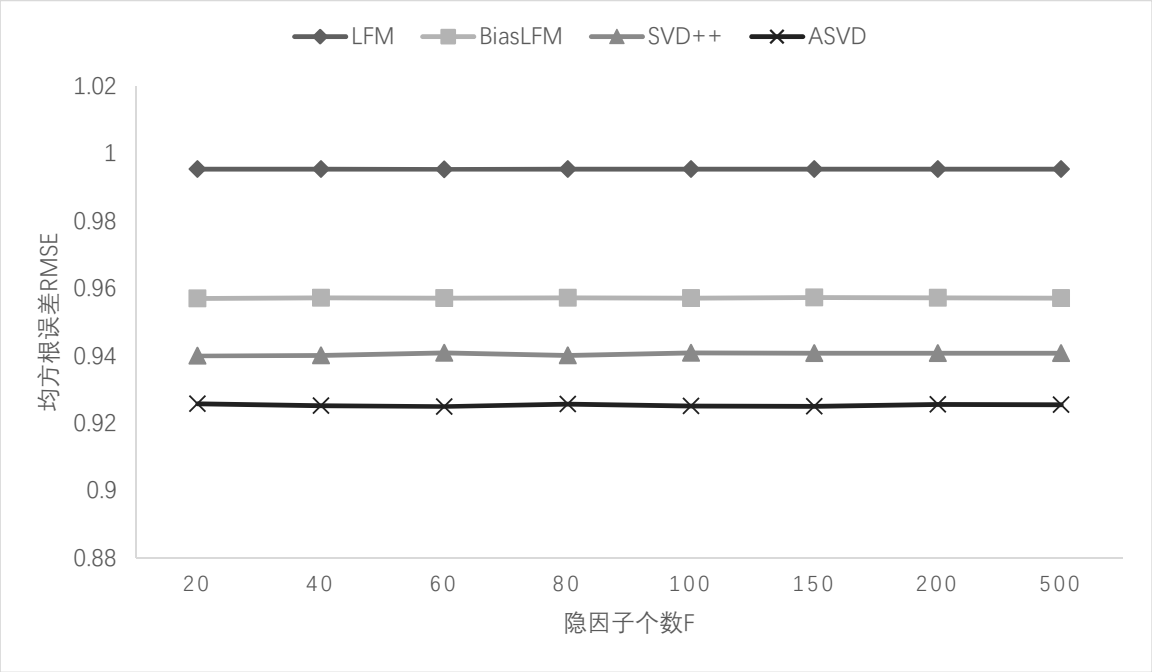

图 3-1 隐语义算法 RMSE 随隐因子个数变换图

(RMSE transformation diagram of LFM algorithm with the number of hidden factors)

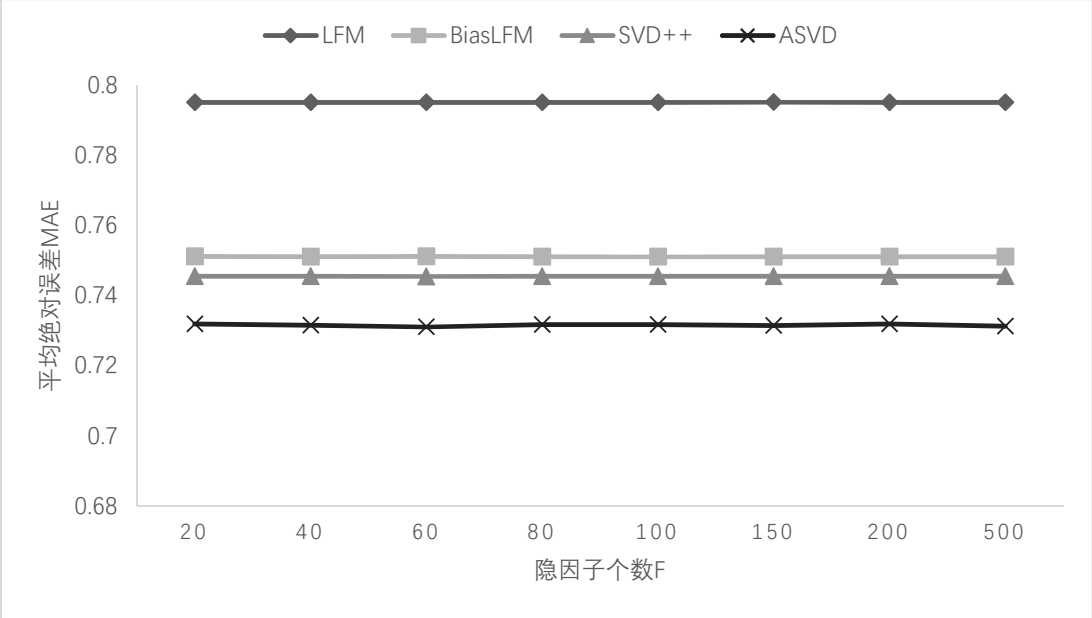

图 3-2 隐语义算法 MAE 随隐因子个数变换图

(MAE transformation diagram of LFM algorithm with the number of hidden factors)

由表 3、图 8、图 9 可知，隐因子个数不断增加，但 LFM、BiasLFM、SVD++、ASVD 模型对于 f 的变化很不敏感，均方根误差 RMSE 和平均绝对误差 MAE 的变化都非常小。从图中可知，这四种模型的误差随着 f 的变化却保持一种平稳的状态，由此可知隐因子个数 f 对模型预测精度影响很小。所以在隐因子个数的选择上，可以根据实际情况

进行选择，减少考虑其对模型的影响。

3.1.4.3　各隐语义算法的比较

上述实验讨论的隐因子个数 f 对各算法的影响，由上述结果可知，在隐因子个数 f=50、学习速率 $\alpha = 0.002$ 、正则化系数 $\lambda = 0.01$ 、迭代次数 n=500 的情况下，得到各算法的 RMSE 和 MAE 如表 3-3：

表 3-3 隐语义算法预测误差对比表(Implicit semantic algorithm prediction error comparison table)

	RMSE	MAE
LFM	0.99548	0.79516
BiasLFM	0.94716	0.75107
SVD++	0.94019	0.74552
ASVD	0.92516	0.73172

在上述参数情况下，各算法在 MovieLens 数据集迭代过程中 RMSE 如表 3-4：

表 3-4　隐语义算法迭代过程中 RMSE 和 MAE 的变化表

(Change table of RMSE and MAE in the iterative process of implicit semantic algorithm)

迭代次数	LFM	BiasLFM	SVD++	ASVD
20	1.1465	1.10694	1.04136	1.11462
50	1.06829	1.05694	0.98241	1.03641
80	1.01745	0.98346	0.95356	0.97664
100	0.99704	0.96329	0.94836	0.95642
200	0.99581	0.95834	0.94493	0.94863
300	0.99578	0.95713	0.94019	0.92924
500	0.99548	0.95716	0.94019	0.92516

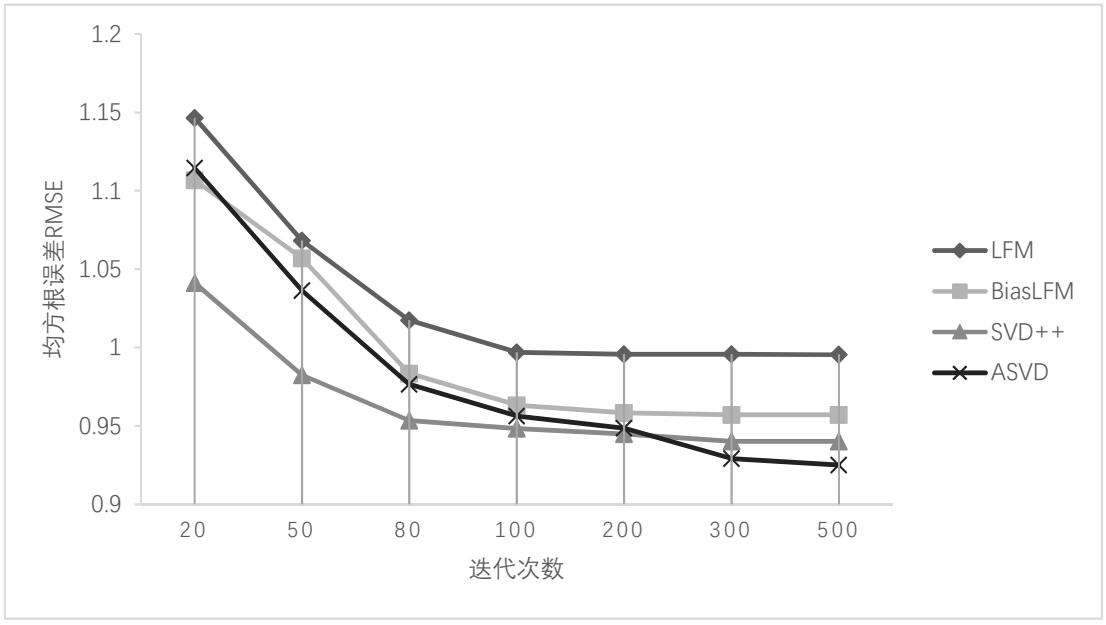

图 3-3 隐语义算法预测准确率比较图

(Implicit semantic algorithm prediction accuracy comparison chart)

由图 3-3 可知，SVD++模型和 ASVD 模型有着较低的预测误差，具有良好的预测性。LFM 模型、BiasLFM 和 SVD++算法都能在迭代次数为 200 时 RMSE 保持平稳，而 ASVD 在近 500 次的迭代次数时才趋于平稳，说明 ASVD 模型比前三个模型所需要更多的迭代时间。总的来说，SVD++在本实验虽然在预测精度上不如 ASVD，但所耗费的迭代时间比 ASVD 少很多，所以 SVD++在性能上优于 ASVD，而 BaisLFM 不如 SVD++但优于 LFM。

3．2　隐语义模型学习方法的改进

3．2．1　SGD 学习率的改进

梯度下降法[17]是一种常用的优化方法， 该方法通过沿着目标函数 $J(\theta)$ 的参数 θ 的一阶导数相反方向 $-\nabla\theta J(\theta)$ 不断地更新模型的参数，逐渐得到目标函数的极小值点，以此来使目标函数收敛，如图 3-4。对于求一个目标函数 f(x)的极小值，首先应该对优化的参数 x 进行求一阶导，将一阶导的反方向作为迭代的搜索方向，之后需要确定一个学习率 α 作为搜索的步长进行迭代，以此使 f(x)收敛。

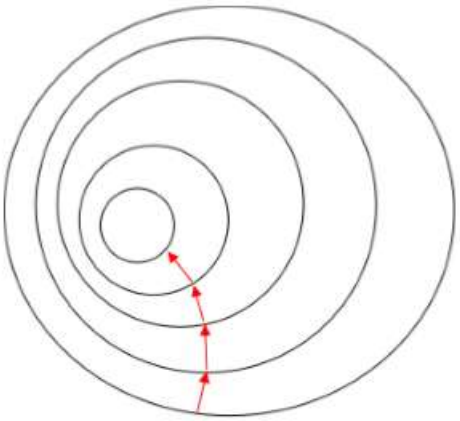

图 3-4 梯度下降法过程图(process diagram of gradient descent method）

在隐语义模型的训练阶段，学习率α的值对训练的时间和结果都有很大的影响。如果学习率α选择过大，虽然能快速迭代到最优解附近，但是有可能陷入局部最优，无法计算出全局最优，对训练的预测结果准确度产生很大的影响。如果学习率α选择过小，虽然可以迭代出的结果精度高，但由于迭代的步长小，使得迭代会耗费很长的时间。

之前的隐语义模型都采用了固定的学习率，分析了固定学习率$(\alpha=0.001)$的学习的训练过程后，发现该模型在前期的收敛速度太慢，耗费了很多的时间。之后增加了学习率$(\alpha=0.01)$并采用学习率递减$(\alpha=0.9\alpha)$的方式进行训练，虽然误差变小，但迭代也会耗费很长的时间。研究者又通过指数函数来调节学习率的大小，其公式[18]为：

$$\alpha(t)=\alpha_0*\alpha_1^t \tag{3-24}$$

其中 t 为迭代次数，α_0和α_1分别是控制学习率的参数，其中α_1应小于 1 以此来保证该函数是减函数。该函数满足了学习率逐渐减小，一定程度上满足了初期大学习率和后期小学习率的需求，但该函数可控函数不够灵活，在应用中改进效果不明显。

为了进一步使迭代时间缩短和训练误差变小，本节进一步对梯度下降法的学习率进行改进，该学习率的表达式为：

$$\alpha(t)=\frac{\alpha_0}{\alpha_1^t+\alpha_2*t} \tag{3-25}$$

其中 t 为迭代次数，α_0、α_1、α_2为学习率的三个参数，用来灵活控制学习率的变化。学习率改变公式应该是一个减函数，其中应满足$\alpha_0>0,\alpha_1>1,\alpha_2>0$，该表达式在传

统采用指数改变学习率($\alpha(t)=\alpha_0*\alpha_1^t$)的基础上进行改进，能灵活控制在初期有着较大的初始，加快收敛速度，而在后期迭代过程中控制更小的步长寻找到更精确的极值点。实验证明，$\alpha_0=0.09,\alpha_1=1.9,\alpha_2=0.01$时模型具有很好的训练效果。

3．2．2　自适应学习率的 Adam 方法

Adam 是一种可以替代传统随机梯度下降的一阶优化算法。Adam 算法和的随机梯度下降不同，随机梯度下降保持单一的学习率更新所有的权重，学习率在训练过程中并不会改变。而 Adam 通过计算梯度的一阶矩估计和二阶矩估计为不同的参数自适应性学习率。Adam 算法在实践中性能优异，相对于其他种类的随机优化算法具有很大的优势，Adam 算法的迭代过程如图 3-5。

Algorithm 1: *Adam*, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. g_t^2 indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t.

Require: α: Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
Require: $f(\theta)$: Stochastic objective function with parameters θ
Require: θ_0: Initial parameter vector
$m_0 \leftarrow 0$ (Initialize 1st moment vector)
$v_0 \leftarrow 0$ (Initialize 2nd moment vector)
$t \leftarrow 0$ (Initialize timestep)
while θ_t not converged **do**
$t \leftarrow t + 1$
$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
$\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
$\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
end while
return θ_t (Resulting parameters)

图 3-5 Adam 算法过程图(Adam algorithm process diagram)

我们可以对隐语义模型使用 Adam 方法来进行训练，迭代的过程为：

(1)、对参数求偏导得∇f

(2)、计算参数上一步的梯度

$$g_t=\nabla f\left(\theta_{t-1}\right) \tag{3-26}$$

(3)、计算一阶矩向量

$$m_t = \beta_1 * m_{t-1} + (1-\beta_1) * g_t \tag{3-27}$$

(4)、计算二阶矩向量

$$v_t = \beta_2 * v_{t-1} + (1-\beta_2) * g_t^2 \tag{3-28}$$

(5)、纠正一阶矩向量

$$\hat{m}_t = m_t / (1-\beta_1^t) \tag{3-29}$$

(6)、纠正二阶矩向量

$$\hat{v}_t = v_t / (1-\beta_2^t) \tag{3-30}$$

(6)、更新参数

$$\theta_t = \theta_{t-1} - \alpha * \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon) \tag{3-31}$$

以 SVD++模型为例，其参数 p_{uk} 迭代更新的过程为：

$$\frac{\partial C}{\partial p_{uk}} = -2e_{ui}q_{ik} + 2\lambda p_{uk} \tag{3-32}$$

$$g_{t(p_{uk})} = -2e_{ui}q_{ik} + 2\lambda p_{uk} \tag{3-33}$$

$$m_{t(p_{uk})} = \beta_1 * m_{t-1(p_{uk})} + (1-\beta_1) * g_{t(p_{uk})} \tag{3-34}$$

$$\hat{v}_{t(p_{uk})} = v_{t(p_{uk})} / (1-\beta_2^t) \tag{3-35}$$

$$\hat{m}_{t(p_{uk})} = m_{t(p_{uk})} / (1-\beta_1^t) \tag{3-36}$$

$$\hat{v}_{t(p_{uk})} = v_t / (1-\beta_2^t) \tag{3-37}$$

$$p_{uk} = p_{uk} - \alpha * \hat{m}_{t(p_{uk})} / (\sqrt{\hat{v}_{t(p_{uk})}} + \epsilon) \tag{3-38}$$

其中默认 $m_0 = 0$， $v_0 = 0$， $t = 0$， $\beta_1 = 0.9$， $\beta_2 = 0.999$， $\alpha = 0.001$， $\epsilon = 10^{-8}$。使用 Adam 算法的默认参数就可以解决绝大多数训练模型，不需要调节学习率。

3．2．3　实验与结果

本实验采用 MovieLens 数据集，对提出的隐语义模型改进的学习方法进行测试比较，其中选取 f=30，$\lambda = 0.01$，对固定学习率、改进的学习率和改进的学习方法在 SVD++

模型上进行比较，令固定学习率$\alpha = 0.001$，改进学习率的$\alpha_0 = 0.09, \alpha_1 = 1.9, \alpha_2 = 0.01$，Adam 为默认参数，三种方法的 RMSE 在不同的迭代次数中如图 3-6：

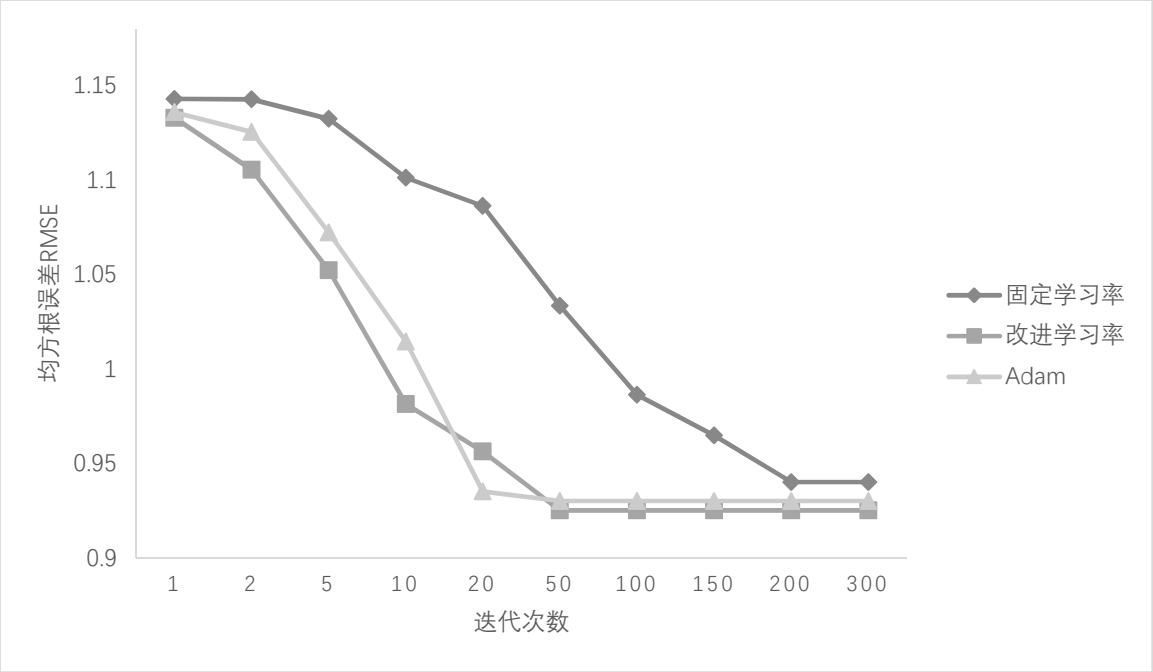

图 3-6 不同学习率或学习方法训练效率图

(training efficiency of different learning rates or learning methods)

由图的曲线可知，固定学习率的迭代在 200 次时才稳定，而且 RMSE 比改进学习率和 Adam 学习方法较高，所以固定学习率的学习效率较低。改进后的学习率迭代约 50 次找到最优点，其在前期具有很快的收敛速度，但在中期收敛速度逐渐减慢，到了后期能很好地找到最优解。改进后的学习率与 Adam 学习方法相比，Adam 算法在迭代 20 次时就趋于稳定，在前期收敛速度不如改进学习率，在中后期收敛速度比改进后的学习率快，预测精度和改进的学习率相差无几。尽管从迭代的次数上来看，Adam 比改进的学习率方法更快，但迭代时间和使用的空间却比改进的学习率方法多得多。总的来看，对于隐语义模型来说，使用改进学习率的随机梯度下降法的效果最好。

4. 基于隐语义模型的图书推荐

4．1 隐语义模型对图书标签候选集的扩展

推荐系统的目的是联系用户的偏好和物品，这种联系需要依赖一些媒介。GroupLens

提出目前推荐系统基本上通过三种方式进行联系，第一种是基于用户的算法，对与用户有相似偏好的用户喜欢的物品进行推荐。第二种是基于物品的算法，对用户喜欢的物品的相似物品进行推荐。第三种是利用一些特征把用户和物品进行联系，这里的特征有很多不同的表现方式，在本文中表现为隐语义向量的方式。

在图书推荐系统中，很多系统都支持用户对物品进行打标签[19]，比如国内的豆瓣图书。隐语义模型将用户和图书的联系都映射到了 f 维空间，为了对这 f 维空间进行解释，我们采用图书的标签对 f 维空间做出隐类解释。

隐语义模型不仅能预测用户对物品的评分，还可以利用隐式反馈行为来对物品进行聚类。对用户 u 产生过行为的物品 i 标记为正样本，在矩阵中记为 $r_{ui}=1$，否则为 $r_{ui}=0$，对于正负样本的采样在 2.3.4 中已经讨论过这里不赘述。这里采用 LFM 模型对物品进行聚类。

我们使用 Book-Crossing 数据集对 LFM 模型计算出了用户兴趣矩阵 P 和物品矩阵 Q，对于每个隐类来说，最简单的方式就是取该隐类权值最大的 S 个物品集合，将物品打过相同标签的次数进行相加，取打标签次数最多的前三个标签作为该隐类的聚类解释。令 $n_{b,i}$ 表示物品 i 被打过标签 b 的次数，N(q,s)表示物品集合中的标签集合，该算法被定义为：

$$\text{Simple\_Tag}=\text{sort}\left(\sum_{i\in(Q,S),b\in N(q,s)} n_{b,i}\right) \tag{4-1}$$

上述的算法标签的权重是通过标签次数确定的，这样给热门的标签过大的权重，从而不能反映隐类真正的解释。而且这里的隐类仅仅通过聚类后的图书进行解释，没有考虑用户对隐类的也存在解释，用户对一个类图书感兴趣他也极有可能对该类打了很多标签。所以基于以上的问题，本文对热门问题做出惩罚，对上述算法进行改进：

$$\text{Tag}=\text{sort}\left(\sum_{i\in(Q,S),u\in(P,S)} \frac{1}{\log\left(1+n_b^{(u)}\right)} n_{u,b}p_{u,k}+\frac{1}{\log\left(1+n_i^{(u)}\right)} n_{i,b}p_{i,k}\right) \tag{4-2}$$

其中 u 表示该隐类中权值最高的 S 个用户集合中的一个用户，i 表示该隐类中权值最高的 S 个物品集合中的一个物品，$n_b^{(u)}$ 表示标签 b 被多少不同的用户使用过，$n_i^{(u)}$ 表

示物品 i 被多少不同的用户打过标签，$\frac{1}{\log\left(1+n_b^{(u)}\right)}$是热门标签的惩罚，$\frac{1}{\log\left(1+n_i^{(u)}\right)}$是热门物品的惩罚，$n_{u,b}$表示用户 u 打过标签 b 的次数，$p_{u,k}$表示用户标签的权重，$n_{b,\mathrm{i}}$表示物品 i 被打过标签 b 的次数，$p_{i,k}$表示物品标签的权重。

经过测试表明，Tag 算法比 Simple_Tag 算法对图书推荐的隐类解释更加准确。在实际意义中，算法得出的标签的隐类解释可以扩展用户对图书打标签的候选集，如图 4-1。如该图书在 f2 类所占的权重最大，在用户对图书打标签的候选标签里，候选标签可以是图书已有标签+f2 类解释标签，这大大丰富了候选标签，也能提高用户满意度。

图 4-1 豆瓣标签示意图(douban tags)

4．2 融合图书时间信息的隐语义模型

时间在推荐系统中是一种重要的上下文信息，用户、物品会随着时间的变化而发生变化，相应的推荐也会发生变化。给定时间信息后，推荐系统从一个静态的系统变成一个时变的系统，用户行为数据也成为一个时间序列(u,i,t)，下面将利用时间上下文信息改进隐语义模型，对针对特定的图书推荐做出改进。

4．2．1 TSVD 模型

在有了时间信息[20]后，用户的行为数据变成一个三元组(u,i,t)，二维矩阵变成了三维矩阵，融入了时间信息后可以分析用户的兴趣变化做出更精确的推荐。Koren 在 BiasLFM 模型的基础上引入了时间效应，该模型称为 TSVD，其对评分的预测值定义为：

$$\hat{r}_{uit}=\mu+b_u+b_i+b_t+p_u^T\bullet q_i+x_u^T\bullet y_t+s_i^T\bullet z_t+\sum_{k=1}^{f}g_{u,k}h_{i,k}l_{t,k} \tag{4-3}$$

其中b_t表示系统平均分随时间的影响，$x_u^T\bullet y_t$表示用户平均分随时间变化的影响，

$s_i^T \bullet z_t$ 表示物品平均分随时间变化的影响，$\sum_{k=1}^{f} g_{u,k} h_{i,k} l_{t,k}$ 表示用户偏好随时间的影响，该模型同样也可以采用随机梯度下降法来进行训练。

4．2．2　Top-N 推荐中的图书排名优化

由于图书推荐更多的是面向学习者，而学习是一种循序渐进的过程，图书阅读的次序对学习者来说也是很重要的。比如一个已经阅读过《java 高级编程》的用户，如果你给他推荐《java 入门》这本书，虽然这两本书相似度很高，而且隐语义模型对《Java 入门》的预测评分也很高，但是这显然不能满足用户的兴趣。所以由于图书推荐的特殊性，图书推荐不仅应该推荐用户预测评分高的图书，而且还能根据用户当前状态，对 Top-N 推荐列表的图书推荐进行排序[21]优化，以此来提高用户满意度。

为了解决图书本身属性之间阅读顺序的问题，本节提出了两种方法：基于统计的图书阅读顺序和基于隐语义模型的图书阅读顺序。

(1)基于统计的图书阅读顺序

为了找出图书次序的相关信息，我们可以仿照 ItemCF 算法中物品相关性计算的方法，对每个用户扫描他的行为数据，如图 4-2。由于用户的行为数据是按照时间进行排列的，所以一定程度上反映了图书之间的相对次序。假设用户对图书 i 产生行为的时间早于图书 j，那么图书顺序矩阵 C[i][j]++，将所有用户的次序矩阵相加得到最终的图书顺序矩阵 C。若 C[i][j]>C[j][i]，那么可认为图书 i 阅读的时间应该比图书 j 阅读的时间更早。

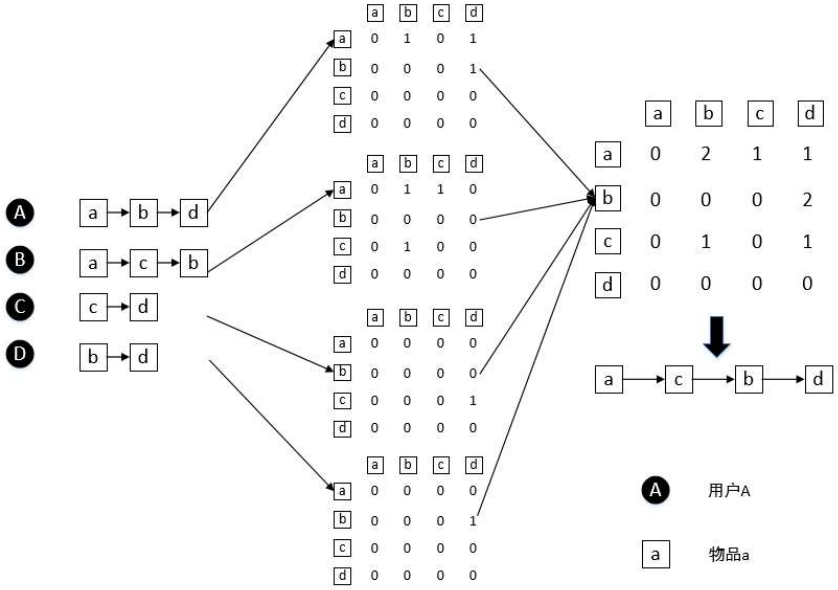

图 4-2 图书顺序矩阵过程图(book order matrix process diagram)

(2)基于隐语义模型的图书阅读顺序

在上文我们谈到隐语义模型可以将用户和物品的联系映射到 f 维空间来预测用户对物品的评分，那么是否可以根据用户对物品产生过行为的历史时间来预测用户对物品产生行为的时间，然后根据时间的相对大小来判断图书的相对次序。

假设从 2000 年 1 月 1 日进行计时，初始时间数为 0，之后的每一天对时间数加 1，如(u,i,2118)表示用户 u 对物品 i 产生行为的时间在起始时间的第 2118 天。将所有的用户行为时间数据转换为上述形式形成了用户-物品-时间矩阵 T'。经过比较我们采用了 BiasSVD 模型对该矩阵进行训练，BiasSVD 模型预测的时间 $\hat{t}_{ui}$ 定义为：

$$\hat{t}_{ui} = \mu + b_u + b_i + p_u^T q_i \tag{4-4}$$

相应的损失函数为：

$$\mathrm{C}(\mathrm{p},\mathrm{q},b_u,b_i) = \sum_{(u,i)\in Train} \left(t_{ui} - \hat{t}_{ui}\right)^2 + \lambda\left(\left\|p_u\right\|^2 + \left\|q_i\right\|^2 + {b_u}^2 + {b_i}^2\right) \tag{4-5}$$

同样我们可以采用随机梯度法来对损失函数进行训练，最终得到用户预测物品行为时间矩阵 T。若 T[u][i]>T[u][j]，则用户 u 认为图书 i 的阅读顺序比图书 j 的更迟。之后对所有的用户进行询问，得到最终的图书顺序矩阵 C。

(3)Top-N 推荐的排序调整：

对于根据隐语义模型得出的长度为 K 的推荐列表，我们给予推荐列表中的物品一个顺序权重 s，由原来的根据预测评分 p 排序改进为根据 p*s 进行排序。顺序权重 s 的取值范围为(0.8,1.2)，令 C[i][j]/C[j][i]为 r，则顺序权重 s 的定义为：

$$s = \frac{14}{25e^{r}} + \frac{4}{5} \tag{4-6}$$

推荐列表根据 p*s 进行排序，既实现了用户评分偏好，又对图书阅读顺序做出来调整，有效地提高了用户满意度。

4．3　基于隐语义模型的图书推荐实例

4．3．1　图书推荐外围架构

图书推荐由 UI、日志系统。用户行为日志存储系统和推荐系统组成，如图 4-3。UI 界面负责与用户进行交互，并将用户行为数据向日志系统反馈。日志系统通过用户行为数据的类型来判断数据存储的位置，日志可能存储在数据库、缓存或者文件系统中。推荐系统通过各种推荐引擎分析用户的行为日志，形成初始的推荐结果，并通过过滤模块、排名模块等形成最终的推荐结果发送给用户，并通过 UI 界面进行展示。

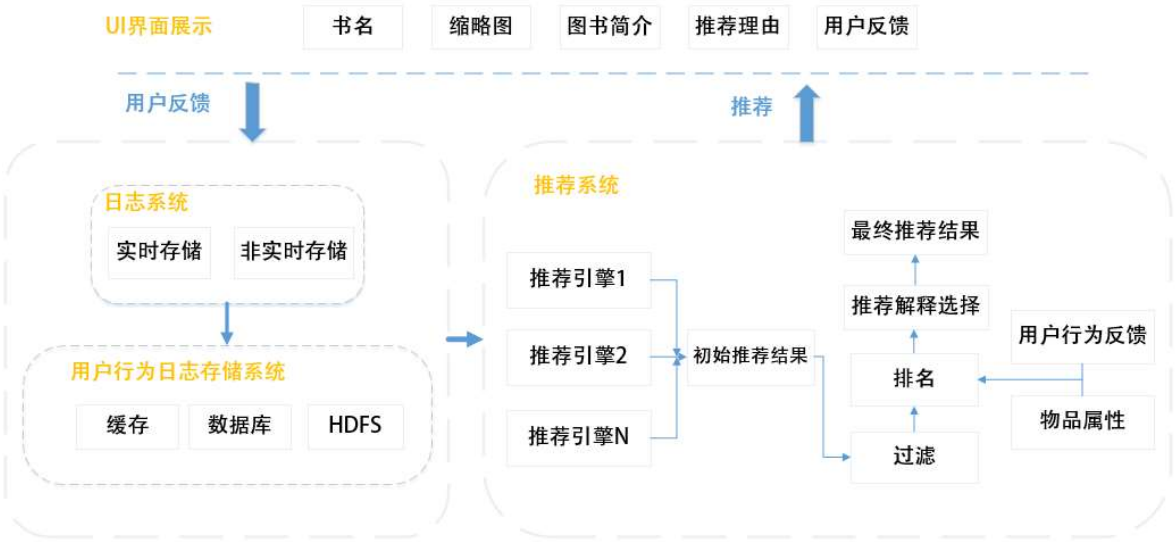

图 4-3 图书推荐系统的外围架构图(peripheral architecture diagram of book recommendation system)

4．3．2　数据收集和存储

用户行为数据是推荐系统分析用户的数据基础，用户行为数据的收集由日志系统负责，用户行为数据的存储由用户行为日志存储系统负责。在图书推荐中，存在很多不同类型的用户行为数据，比如浏览、收藏、评论等，而对于不同的用户行为数据存储的方式也不同。

日志系统能判别不同的用户行为数据的存储方式。比如收藏、评论、评分、分享等用户行为是需要实时存取的，因为只要用户有了这些行为，UI 界面就需要立即体现出来。而对于浏览网页和搜索等用户行为就不需要进行实时存储。

根据用户行为的类型觉得是否需要实时存储，不同的数据应该存储在不同的媒介中。一般来说，需要实时存储的数据存储在缓存和数据库中，而不需要实时存储的数据存储在文件系统中，如 HDFS。

4．3．3　推荐系统架构

本文讨论的是基于隐语义模型的图书推荐，所以采用一种基于隐类特征的推荐系统架构，如图 4-4。那么推荐系统的核心就分为了两个部分：如何给用户生成特征和如何把特征与物品联系起来。

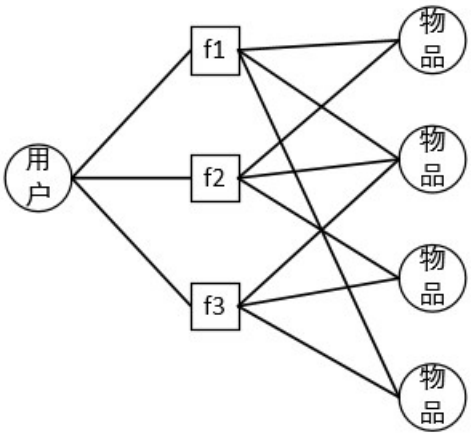

图 4-4 用户-特征-物品联系图(user - feature - item contact diagram)

要解决推荐系统核心的两个问题，就需要使用不同的推荐引擎。因此推荐系统需要由多个推荐引擎组成，每一个推荐引擎完成一类特征。这样一方面能提高推荐系统的扩展性，另一方面可以实现推荐引擎级别的用户反馈。推荐引擎主要分为三个部分：生成

用户特征向量、物品-特征关联推荐和过滤排名模块，如图 4-5。

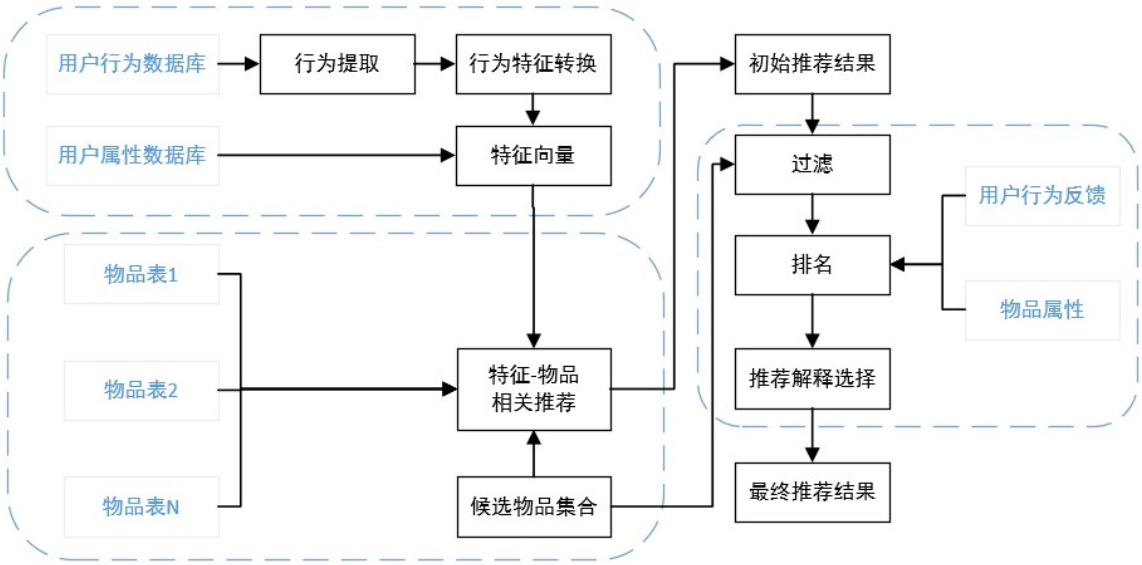

图 4-5 推荐引擎架构图(recommendation engine architecture diagram)

生成用户特征向量模块负责从数据库或者缓存中取出用户行为数据，通过用户行为的种类、用户行为产生时间、用户的行为次数等形成用户特征向量。

特征-物品关联推荐模块是通过离线的相关表与用户的特征向量进行关联，物品表不止一个，可以是隐语义模型所训练的预测评分表，也可以是各种推荐算法在离线时所产生的表。除了离线表之外，还有一个候选物品集合，候选物品集合可以包括新图书，以此来解决物品冷启动带来的问题。

过滤和排名模块负责对推荐物品进行过滤和排名。在过滤的阶段，系统对初始的推荐列表过滤掉用户已经产生过行为的图书和质量很差的图书。排名模块可以更好地提高用户的满意度，一般排名模块需要很多不同的排名子模块。排名模块可以根据图书的新颖性、多样性、时间多样性和用户反馈等进行排名。

结论

本文在深入了解推荐系统和推荐算法后，一方面对隐语义模型在预测准确度方面提出了 BiasSVD、SVD++、ASVD 算法，实验结果证明改进后的算法具有更好的预测准确度；另一方面提出了隐语义模型改进的学习方法，实验证明能有效减少隐语义模型迭代次数和提高预测准确度。最后对图书推荐的特点对隐因子做出标签化的解释和优化图书推荐排名，设计了一个基于隐语义模型的图书推荐系统。本文对隐语义模型和图书推荐的研究都具有很好的现实意义。

致谢

本人在论文期间都是在林福平老师全面、具体指导下完成进行的。林老师渊博的学识、敏锐的思维、民主而严谨的作风使我受益非浅，并终生难忘。

感谢林老师在毕业设计工作中给予的帮助。

感谢我的学友和朋友对我的关心和帮助。

参考文献

[1] YEHUDA K. Factor in the Neighbors：Sealable and Accurate Collaborative Filtering[J]. ACM Transactions on Knowledge Discovery from Data(TKDD)，2010，4(1)：1-17.

[2] YEHUDA K，ROBERY B，CHRIS V. Matrix Factorization Techniques for Recommender System[J]. IEEE Computer Society，2009，42(8)：42-49.

[3] THOMAS H. Latent Semantic Models for Collaborative Filtering[J]. ACM Transactions on Information Systems，2004，22(1)：89. 115.

[4] ROBERT M. BELL，Y K，CHRIS V. The BellKor solution to the Netflix Prize. Tech. Rep(2007)：1. 15.

[5] 江雪琴.推荐系统中的隐语义模型算法研究[D].北京:中国科学技术信息研究所,2015.

[6] YEHUDA K. Collaborative Filtering with Temporal Dynamics[J]. Communications of the ACM，2010，53(4)：89. 97.

[7] AHMED A，BHARGAV K，Sandeep Pandey. Latent Factor Models with Additive and Hierarchically. smoothed User Preferences[C]. Proceedings of 6th ACM International Conference on Web Search and Data Mining (WSDM)，2013.

[8] YEHUDA K，ROBERY B，CHRIS V. Matrix Factorization Techniques for Recommender System[J]. IEEE Computer Society，2009，42(8)：42-49.

[9] 1STEEN R，CHRISTOPH F，ZENO G’et al。BPR：Bayesian Personalized Ranking from Implicit Feedback[J1. In UA12009，2009：452. 461.

[10] 1ROBERT M。BELL，Y K，CHRIS V. The BellKor 2008 Solution to the Netflix Prize[C]. Tech. Rep(2008)：1.21.

[11] 项亮. 推荐系统实践[M]. 北京：人民邮电出版社，2012

[12] 鲍晨阳.主题引导推荐系统[D].四川:电子科技大学,2016.

[13] 李新良.基于隐语义的混合推荐算法研究[D].福建:厦门大学,2014.

[14] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender

systems, Computer, 2009, 42(8), 30-37.

[15] 巫可,战荫伟,李鹰.融合用户属性的隐语义模型推荐算法[J].计算机工程,2016,42(12):171-175.

[16] 李春春,李俊.基于 ASVD 的协同过滤推荐算法[J].小型微型计算机系统,2018,39(6):1286-1290.

[17] YEHUDA K, ROBERY B, CHRIS V. Matrix Factorization Techniques for Recommender System[J]. IEEE Computer Society, 2009, 42(8): 42-49.

[18] 王燕,李凤莲,张雪英, 等.改进学习率的一种高效 SVD++算法[J].现代电子技术,2018,41(3):146-150,156.

[19] 高成.基于标签主题建模的图书推荐系统研究[D].浙江:浙江大学,2014.

[20] Yehuda Koren ,Collaborative Filtering with temporal dynamics,ACM 2009 Article ,2009

[21] 黄震华,张佳雯,田春岐, 等.基于排序学习的推荐算法研究综述[J].软件学报,2016,27(3):691-713.