Modeling and Simulation 建模与仿真, 2025, 14(4), 699-715
Published Online April 2025 in Hans. https://www.hanspub.org/journal/mos
https://doi.org/10.12677/mos.2025.144322

Hans汉斯

融合网络社团特征与矩阵分解的推荐算法

艾　均，李贵平，苏　湛，苏　杭

上海理工大学光电信息与计算机工程学院，上海

收稿日期：2025年3月22日；录用日期：2025年4月15日；发布日期：2025年4月23日

摘　要

为解决推荐系统在处理超大规模数据集时计算复杂度高、资源消耗大，以及推荐性能与可扩展性之间的矛盾，本文设计了一种融合用户相似性网络社团特征的快速并行梯度下降矩阵分解算法。首先，通过计算用户相似性，基于高相似度用户关系构建相似性网络，并采用模块度优化算法将用户划分为若干社团。同时，设计了一种用户补偿机制，在社团划分中引入指定数量的活跃用户，降低基于用户相似性社团划分的用户群内部数据的过度相似性。随后，针对每个用户群，采用并行梯度下降矩阵分解算法学习用户与物品的隐因子，以预测用户对未知物品的评分。该模型通过社团分组和并行计算，不仅减少了计算复杂度和资源消耗，还有效提升了推荐系统的准确性和算法学习效率。与现有领先算法相比，该模型在多个数据集上有效提高了推荐精度和召回率，降低了均方根误差和平均绝对误差。

关键词

用户相似性网络社团发现，梯度下降矩阵分解，可扩展性，数据稀疏，推荐算法，社团检测算法

Recommendation Algorithm Based on Fast Parallel Stochastic Gradient Descent Matrix Factorization and User Similarity Network Community Features

Jun Ai, Guiping Li, Zhan Su, Hang Su

School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai

Received: Mar. 22nd, 2025; accepted: Apr. 15th, 2025; published: Apr. 23rd, 2025

Abstract

To address the challenges of high computational complexity and resource consumption in recom-

文章引用: 艾均, 李贵平, 苏湛, 苏杭. 融合网络社团特征与矩阵分解的推荐算法[J]. 建模与仿真, 2025, 14(4): 699-715.
DOI: 10.12677/mos.2025.144322

mendation systems when handling ultra-large datasets, this study proposes a fast parallel gradient descent matrix factorization algorithm that integrates user similarity network community features. First, by calculating user similarities, a similarity network is constructed based on highly similar user relationships, and a modularity optimization algorithm is employed to partition users into several communities. Additionally, a user compensation mechanism is designed to introduce a specified number of active users into the community division, thereby reducing the excessive similarity of data within user groups based on similarity community partitioning. Subsequently, for each user group, a parallel gradient descent matrix factorization algorithm is utilized to learn the latent factors of users and items, enabling the prediction of user ratings for unknown items. This model reduces computational complexity and resource consumption through community grouping and parallel computation, while effectively enhancing the accuracy of the recommendation system and the efficiency of algorithm learning. Compared to existing leading algorithms, this model significantly improves recommendation precision and recall across multiple datasets, while also lowering root mean square error and mean absolute error.

Keywords

User Similarity Network Community Discovery, Gradient Descent Matrix Factorization, Scalability, Data Sparsity, Recommendation Algorithm, Community Detection Algorithm

1. 引言

推荐系统能够有效应对科技快速发展和信息爆炸带来的信息过载问题[1] [2]。其是一种主动提供个性化信息的服务，它通过用户与内容的交互历史建模用户的偏好，进而为用户推荐感兴趣的内容，满足信息过滤需求，提升用户获取信息的效率[3]。

协同过滤(CF)是推荐系统中应用最为广泛的一种方法，而矩阵分解(MF)是其中最为常用的技术之一，在解决可扩展性问题方面取得了显著进展[4]。矩阵分解算法将用户对物品的评分矩阵分解为用户的潜在因子矩阵和物品隐因子矩阵，用户对物品的评分预测值由用户潜在因子矩阵和物品潜在因子矩阵的内积求得[5]。

矩阵分解技术在推荐系统领域广受关注和应用，目前流行趋势主要体现在以下三个方面：

在矩阵分解推荐算法中引入时间因素在提高推荐的精准度方面发挥了重要作用。例如，时间权重的矩阵分解算法(Matrix Factorization with Time Weight Collaborative Filtering, MFTWCF) [6]通过时间衰减函数调整信息权重，基于 OTT 视频的个性化时间感知推荐算法(Time-Aware Matrix Factorization, TAMF) [7]则利用矩阵分解分析观看时间对用户兴趣的影响。这些算法结合时间信息，能更准确地捕捉用户随时间变化的偏好。然而，引入时间因素也显著增加了计算复杂度，对处理大规模数据集的实时性和可扩展性提出了挑战[8]。

跨领域知识迁移与矩阵分解结合。为克服数据稀疏和冷启动问题，将迁移学习与矩阵分解技术相结合成为一种有效策略[9]，这类方法通过引入用户的内容偏好或地理位置等信息，提升推荐精度，扩展了矩阵分解的应用范围。然而，有效迁移往往依赖于高质量的跨领域数据，其获取成本较高。此外，不同领域间的数据分布差异可能导致模型适应性降低，甚至引入噪声，最终影响推荐系统的性能[10]。

为了提升大规模数据集上的计算效率和模型精度，研究者们采用并行化、自适应学习率和混合计算

架构来优化矩阵分解算法。例如，快速并行随机梯度下降算法(FPSG) [11]-[13]减少线程锁竞争，CCD++算法[14]高效处理稀疏数据，混合 CPU 和 GPU 架构[15] [16]提升计算效率。然而，单机或非分布式架构在面对持续增长的数据规模时存在资源瓶颈，因此分布式架构成为提升推荐系统可扩展性和实时性的关键[17]。

尽管基于矩阵分解的推荐算法研究已经取得一定进展，但目前仍面临诸多挑战：

1) 单一随机梯度下降算法(SGD)的内存和计算成本显著增加[18]。

2) 共享内存环境中存在锁竞争，导致计算资源浪费。

3) 当数据规模过大或数据过于稀疏时，计算成本急剧增加，模型的预测准确性也会下降，这在大规模推荐系统、用户个性化推荐和实时数据分析中尤为明显，显著影响系统的整体性能和用户体验。

深度学习在推荐系统中应用广泛，研究集中于提高效率、准确性和可扩展性。例如，LightGCN [19]简化图卷积网络，SGL [20]利用自监督学习提升性能，基于深度强化学习(DRL)模型[21]通过异质信息网络捕捉复杂协同信号。虽然这些模型有所改进，但由于结构复杂、参数众多，训练时间长且缺乏增量计算能力，每次更新需重新训练整个模型。相比之下，矩阵分解算法具有增量计算优势，可有效降低成本并提升效率。

因此，本文对基于矩阵分解的推荐模型进行研究，针对模型在可扩展性和分类准确性上的不足，本文做出了如下贡献：

1) 本文构建了一个基于用户相似性的复杂网络，并采用模块度优化的社团检测算法将用户分组到社团中，形成小的用户群落。这一分组机制有效降低了每个社团的用户处理规模，使得用户群体能够在不同计算设备上实现分布式并行处理，从而显著提升了算法在大规模数据集上的处理效率，并增强了系统的可扩展性。

2) 本文设计了一种用户补偿机制，通过在每个基于社团划分的用户群体中引入指定数量的活跃用户，增加了数据的丰富性，从而缓解了数据稀疏性问题。

3) 本文进一步设计了一种结合用户相似性网络社团检测的快速并行梯度下降的矩阵分解推荐模型，进一步提升了推荐系统的可扩展性，并提高了推荐结果的准确性。

本文第 2 节将介绍与本文研究相关的重要工作、原理及应用场景；第 3 节提出基于用户相似性社团检测的快速并行矩阵分解模型；第 4 节通过实验对模型进行分析与讨论，并与现有领先算法进行比较；第 5 节总结本文工作并提出未来展望。

2. 相关工作

本文采用复杂网络、Louvain 社团发现以及快速并行矩阵分解算法(Fast Parallel Stochastic Gradient Descent, FPSG)等技术，研究在用户相似性网络中进行社团划分后，如何在每个社团内进行快速分布式并行矩阵分解，以提升模型在大规模数据下的可扩展性。

复杂网络的社团结构可以标识网络中节点之间的内在联系和相似性特征，从而揭示节点之间的距离特征。这些特性在推荐系统中尤为重要，因为推荐系统本身可以被视为一个复杂网络，其中节点代表用户或物品，边则代表用户与物品之间的交互关系。通过对推荐系统进行复杂网络建模，本文可以深入研究推荐系统中用户间的网络结构特征，从而提升推荐的准确性和个性化程度[20]。

文献[22]提出基于模块度优化的社团发现算法(Fast Newman-Girvan Algorithm, FN)。这是一种凝聚式社团发现算法，首先将每个节点初始化为一个社团，算法的每一步按使模块度函数值增加最多或减少最少的方法合并两个社团，重复上述步骤直到划分出最后的社团。尽管 FN 算法降低了复杂度，但在处理大规模复杂网络时仍然需要消耗大量时间。为了解决这一问题，Blondel 等[23]提出了基于模块度优化的社

团发现算法(Louvain)。该算法基于层次聚类和局部优化算法，每个节点只需考虑其邻居的社团，从而减少计算量，显著提高计算效率，尤其在处理大规模网络时表现尤为出色。本文采用 Louvain 社团发现算法对用户相似性网络进行划分，形成较小规模的用户群体，并将其用于后续的预测阶段，从而提升模型的计算效率和推荐性能。

快速并行矩阵分解算法[24]，这是一种专为共享内存系统设计的高效并行随机梯度下降算法，用于矩阵分解任务。FPSG 算法引入了部分随机方法，在块内顺序选择评分实例，块间随机选择评分块，保证了内存访问的连续性，同时确保算法快速收敛。这种方法显著减少了缓存未命中率，提升计算效率。此外，FPSG 采用灵活的任务调度机制，动态分配任务给空闲线程，避免线程锁定的问题。调度器根据块的更新次数和当前空闲状态，选择适当的块分配给线程，确保所有线程始终保持忙碌。FPSG 通过将评分矩阵分割成较小的块，并在块间随机选择，避免了线程间的负载不平衡问题。这种设计使得 FPSG 能够高效利用多核 CPU 的计算能力，减少线程等待时间。因为该算法能够快速收敛、优化内存访问，并显著缩短训练时间，本文选择了此矩阵分解算法用于后续的学习和预测任务。

总之，本文通过构建复杂网络对用户相似性关系进行建模，利用 Louvain 算法对用户相似性网络进行社团划分，从而将大规模用户网络分解为多个更具相似性的社团。最后，在每个社团内使用 FPSG 算法对评分矩阵进行分解与预测，从而提升整体推荐系统的可扩展性和效率。

3. 模型

3.1. 问题定义

推荐系统规模的持续增长使得在处理超大规模数据集时，面临深度学习分批训练和矩阵分解同步学习隐因子的问题，消耗大量时间和资源。这种情况下，如何将用户－物品评分数据集 D 合理地分割为若干较小的用户数据子集 $D_1, D_2, \cdots, D_n$ 成为用户群，并将其分布在多台计算机上进行训练和预测，以确保每个子集 D_i 训练顺序有序，并在降低数据稀疏性的同时提升预测准确性，是本研究需要解决的关键问题。

本文所用到的符号及说明如表 1 所示。

Table 1. Key symbols and their descriptions
表 1. 关键符号及其说明

符号	说明
D	整个数据集
$D_1, D_2, \cdots, D_n$	将数据集 D 分割成的若干较小的子集
C	整个社团
$C_1, C_2, \cdots, C_k$	将数据集 C 分割成的若干较小的子集
A_{uv}	用户 u 和用户 v 之间的边权重
T	由相似性网络提取的最小生成树
$Q(u,v)$	模块度函数
R	评分矩阵
U_{active}	最活跃的用户群体
p_u	用户 u 的潜在特征向量
q_v	用户 v 的潜在特征向量
w_{uv}	社团 C_u 与社团 C_v 之间边的权重

续表

(u,v)	用户对
pcc_{uv}	用户 u 和用户 v 之间的相似值
k_u	用户 u 的度数
m	网络中的总权重数

3.2. 算法整体结构

本文提出的算法结构如图 1 所示。首先，在分布式计算模块中，算法将完整数据集分割为若干较小的数据子集，模拟实际环境中的多台服务器处理。尽管所有计算均在单台机器上模拟完成，但这种方式能近似实现分布式系统的工作效果，确保各子集的独立处理与实际情况相符，从而验证算法并保证分布式系统各节点的计算一致性。详细结构如图 2 所示。

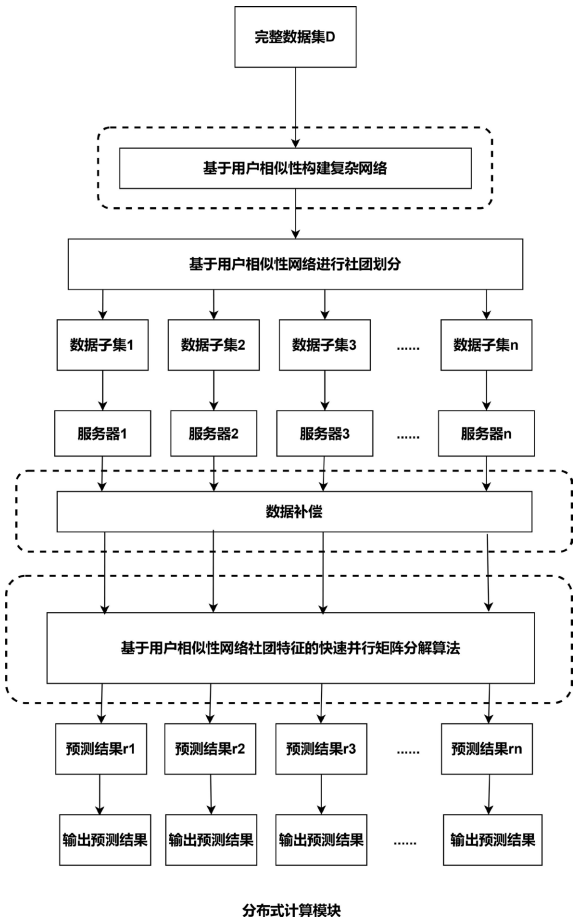

Figure 1. Overall algorithm structure
图 1. 整体算法结构图

首先，用户相似性建模模块通过计算用户之间的相似性，构建加权邻接矩阵，并提取最小生成树，以简化网络结构，同时保留关键用户连接，最终形成用户相似性网络。

然后，基于相似性网络，社团划分模块使用 Louvain 算法，将用户相似性网络划分为多个社团，形成

用户组群。尽管这些用户之间的品味和偏好各不相同，以这些用户为核心进行隐因子学习和预测更为有效，但此阶段生成的用户评分数据仍存在稀疏性问题。

为进一步减少数据稀疏性，数据补全模块挑选全体数据中最活跃的用户作为共享数据，分配到每个用户群体中，增加评分数据的数量。

最后，在每一个补全数据后的用户种群中通过快速并行矩阵分解模块，进行用户和物品的隐因子特征学习。通过快速并行随机梯度下降(FPSG)算法，该模块在多台服务器上并行执行整个算法的处理过程。在理论上应用任何类型的矩阵分解类算法在多台服务器上进行学习。

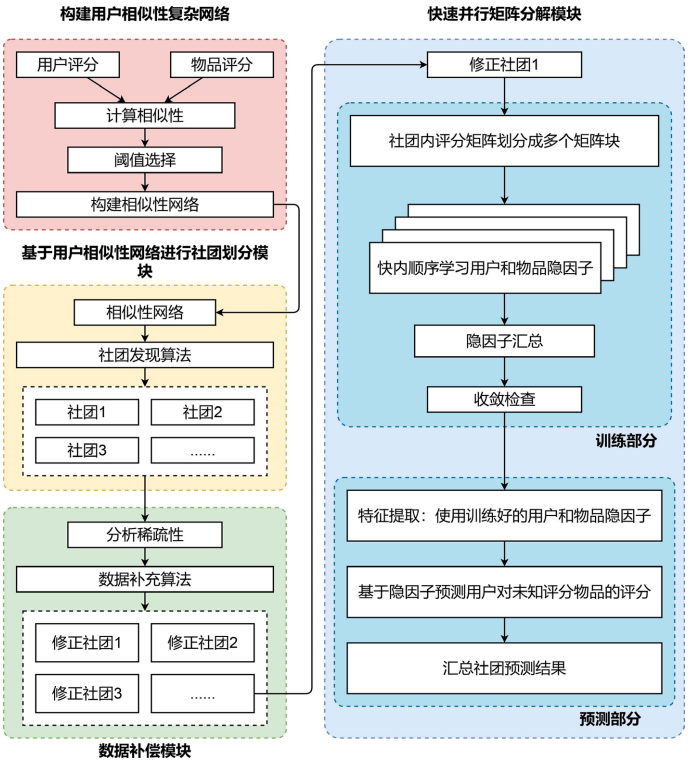

Figure 2. Structure of fast parallel matrix factorization based on user similarity network community features

图 2. 基于用户相似性网络社团特征的快速并行矩阵分解结构图

3.3. 基于用户相似性的社团网络发现方法

3.3.1. 构建用户相似性网络

用户对相同物品的行为产生相似性，评分物品越多，相似性越高。一般来说，用户相似性越大，评分相似性越强，其邻居在预测中的重要性越高。低相似性邻居价值有限，且会干扰其他用户评分。

度量相似性的方法众多，本文选择了皮尔逊(Pearson)相似性度量方式。其公式如式(1)：

$$PCC(u,v) = pcc_{uv} = \frac{\sum_{i\in I_{uv}} (r_{ui} - \overline{r}_u)(r_{vi} - \overline{r}_v)}{\sqrt{\sum_{i\in I_{uv}} (r_{ui} - \overline{r}_u)^2} \sqrt{\sum_{i\in I_{uv}} (r_{vi} - \overline{r}_v)^2}} \tag{1}$$

其中 I_{uv} 集合代表用户 u 和用户 v 共同评过分的物品集合，r_{ui} 和 r_{vi} 分别表示用户 u 和用户 v 对物品 i 的评分，$\overline{r}_u$ 和 $\overline{r}_v$ 分别表示用户 u 和用户 v 对所有相关物品的平均评分。

根据计算得到的 Pearson 相关系数，构建加权网络的邻接矩阵 S，每个元素 S_{uv} 表示用户 u 和用户 v 之间的边权重：

$$S_{uv}=\begin{cases} pcc_{uv}, & \text{if nodes } u \text{ and } v \text{ are connected} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

其中 pcc_{uv} 是用户 u 和用户 v 之间的相似值。基于该矩阵，可以构建反映用户相似性关系的复杂网络，为后续的社团划分和矩阵分解提供基础。

本文选择过滤掉权重小于等于 0 的边，仅保留权重大于 0 的边，一方面为了简化网络结构，另一方面负权重在用户相似性网络中无法有效反映正相关性和推荐价值。其公式如(3)：

$$S_{uv}=\begin{cases} pcc_{uv} & \text{if } \mathrm{pcc}_{uv}>0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

由于构建的相似性网络过于密集，信息冗杂，因此需要对其进行简化处理。在这种情况下，最小生成树作为复杂网络模型的一种基本结构，可以有效过滤冗余信息，简化网络结构。因此，本文通过采用 Prime 算法从相似性网络中提取最小生成树，其步骤如下：

1) 选择任意一个起始用户节点，将其加入到最小生成树(MST)集合 T 中，初始化一个空的边集合 E_T，用来存储 MST 边，使用一个最小堆来存储当前可用的边，按边的权重从大到小排。

2) 从最小堆中取出权重最小的边 (u,v) 加入到 MST 集合中，即 $T=T\cup\{v\}$ 和 $E_T=E_T\cup\{(u,v)\}$，将用户节点 v 的所有的边加入最小堆中，只保留连接到树外用户节点的边，更新最小堆，继续选择下一条边。

3) 重复步骤 2，知道所有用户节点都被包含在 MST 中，即 $|T|=n$，其中 n 图中用户总数。

3.3.2. Louvain 社团发现

为了能够更好地挖掘用户间的紧密度，本文将对得到的最小生成树进行社团划分。Louvain 算法是基于模块度优化的社团发现算法，其优化目标是最大化社团网络的模块度函数值[25]。模块度函数被用来定量描述网络中的社团结构，衡量网络社团结构划分的质量，其定义如(4)：

$$Q(u,v)=\frac{1}{2m}\sum_{uv}\left(S_{uv}-\frac{k_u k_v}{2m}\right)\delta(C_u,C_v) \tag{4}$$

式中模块度函数 $Q(u,v)$ 的值大小反应了社团结构的质量，$Q(u,v)$ 的值越接近 1 说明网络的社团结构越明显。S_{uv} 是邻接矩阵元素，表示用户节点 u 和用户节点 v 之间的边权重，k_u 和 k_v 分别表示用户节点 u 和用户节点 v 的度数，m 表示网络中的总边权重，C_u 表示用户节点所在的社团，若 u 和 v 在同一个社团中，则 $\delta(C_u,C_v)=1$，否则 $\delta(C_u,C_v)=0$。其定义如(5)：

$$\delta(C_u,C_v)=\begin{cases} 1, & \text{if } C_u=C_v \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

该算法主要分为顶点移动和社团聚合两个阶段，逐步优化社团划分，并最终得到模块度最大的社团结构。第一阶段，针对上文构建的相似性网络 S，Louvain 算法首先将每个用户节点视为单独的社团，初始化时每个社团只包含一个顶点，并初始化模块度 $Q(u,v)$。接着对每一个用户节点 u，计算将其从当前社团 C_u 移动到邻居社团 C_v 时的模块度增量 ΔQ，对所有邻居社团计算出的 ΔQ 进行比较，选择最大的邻居社团，若 ΔQ 大于 0，则将用户节点 u 移动到该邻居社团 C_v，否则，用户节点继续保持在当前社团 C_u，其定义如(6)：

$$f(u)=\begin{cases} C_v, & \text{if } \max(\Delta Q)>0 \\ C_u, & \text{if } \max(\Delta Q)\le 0 \end{cases} \tag{6}$$

$f(u)$ 表示用户节点 u 最终所在的社团。重复上述过程，直到所有用户节点不再发生移动，第一阶段停止。其中模块度增量 ΔQ 计算公式如(7)：

$$\Delta Q = \left\{ \frac{1}{2m}\left(\sum_{\text{in}} + k_{u,\text{in}} \right) - \left[\frac{1}{2m}\left(\sum_{\text{tot}} + k_u \right) \right]^2 \right\} - \left\{ \frac{1}{2m}\sum_{\text{in}} - \left[\frac{1}{2m}\sum_{\text{tot}} \right]^2 - \left(\frac{k_u}{2m} \right)^2 \right\} \tag{7}$$

其中 $\sum_{\text{in}}$ 表示社团 C_v 内部边权重之和，$k_{u,in}$ 表示用户节点 u 与社团 C_v 中其他用户节点的边权重之和，$\sum_{tot}$ 表示社团 C_v 的所有边权重之和，k_u 表示用户节点 u 的度数。

第二阶段，将第一阶段得到的每个社团视为一个新的用户节点，此时新用户节点的权重为社团内部用户节点权重之和，社团内部用户节点之间的边的权重之和作为新用户节点的环的权，社团间的连边的权转化为两个新用户节点之间的边的权，通过上述步骤可以得到一个新的网络，重复上述两个过程直到整个社团模块度不再发生变化，算法结束。其中社团 C_u 与社团 C_v 之间边的权重 w_{uv} 的计算公式如(8)：

$$w_{uv} = \sum_{u \in C_u} \sum_{v \in C_v} A_{uv} \tag{8}$$

其中 $\sum_{u \in C_u} \sum_{v \in C_v}$ 表示对所有属于社团 C_u 的用户节点 u 和所有属于社团 C_v 的用户节点 v 进行求和，S_{uv} 表示原始网络中，用户节点 u 和用户节点 v 之间的边权重。图 3 展示了 Louvain 算法实现具体流程。

算法步骤如下：

1) 首先将图中每个用户节点都看成独立的社团，此时社团的数目与用户节点的个数相同。

对每个用户节点 u 计算将其移动到邻居社团 C_v 后的模块度增量 ΔQ，并记录 ΔQ 最大的邻居用户节点。如果 $\max \Delta Q > 0$，则将 i 移动到增量最大的社团，否则保持不变。

2) 重复迭代 2 过程，直到所有结点的社团划分稳定。

3) 将同一社团内的用户节点压缩为新用户节点，并将社团内和社团间的权重更新为 W_{uu} 和 W_{uv}。

在压缩后的图上重复上述步骤，直到整个图的模块度 $Q(u,v)$ 不再发生变化。

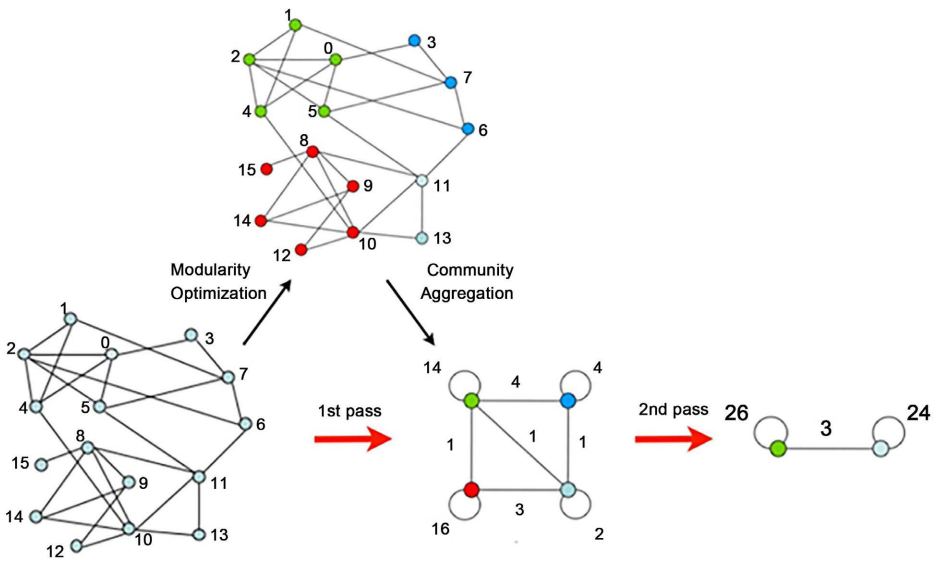

Figure 3. Flowchart of the Louvain algorithm [26]
图 3. Louvain 算法流程图[26]

3.3.3. 数据补偿模块

经过上述用户相似性建模和 Louvain 社团划分之后，各个社团 C_j 内可能仍然存在稀疏性问题，为了

解决这一问题，本文提出了数据补偿模块，通过选择数据集中评分次数最多的活跃用户，按数据集大小调整选取比例，平均分配到各社团中，以填补稀疏部分，提升数据完整性和模型预测性能。

首先，对每个用户 u 在评分矩阵 R 中的评分次数 count_u 进行统计。接着，根据数据集的大小选择最活跃的用户群体 U_{active}。具体而言，当数据集总用户数 $N \geq x_1$ 时，选择评分次数最多的前 $y_1\%$ 用户；而当 $N \leq x_2$ 时，选择前 $y_2\%$ 的用户作为活跃用户。这些用户集合 U_{active} 的定义如下(9)：

$$U_{\text{active}} = \begin{cases} \{u \mid \text{count}_u \geq \text{前} y_1\% \text{高频用户}\}, & \text{如果} N \geq x_1 \\ \{u \mid \text{count}_u \geq \text{前} y_2\% \text{高频用户}\}, & \text{如果} N \leq x_2 \\ \{u \mid \text{count}_u \geq \text{前} y_3\% \text{高频用户}\}, & \text{如果} x_1 > N > x_2 \end{cases} \tag{9}$$

其中 x_1 和 x_2 数据集中用户数量阈值，y_1、y_2 和 y_3 表示活跃用户选择比例。

然后，将活跃用户评分数据合理分配到各个社团 C_j。为确保每个社团获得足够补充，分配比例主要依据社团大小确定：用户数量较多的社团分配更多活跃用户评分数据，反之则较少，从而均衡满足各社团需求，缓解稀疏性问题。分配公式如(10)所示：

$$B_j = \frac{|C_j|}{\sum_{k=1}^{m} |C_k|} \tag{10}$$

其中，B_j 表示分配给社团 C_j 的比例，$|C_j|$ 表示社团 C_j 的用户数量，$\sum_{k=1}^{m} |C_k|$ 表示所有社团的用户数量总和，m 表示社团的总数量。

最后，经过上述步骤，更新原来的社团集合，生成了包含补充数据社团集合 C。

3.4. 基于用户相似性网络社团划分的快速并行矩阵分解模型

为描述使用快速并行矩阵分解模型训练过程。在上一小节中，本文已经构建了社团网络 C，其中 C 代表所有社团的集合。这一节中将详细介绍如何在每个划分出来的社团上进行快速并行矩阵分解。该模型分为选取社团、定义优化目标、随机梯度下降优化(SGD)四个阶段。在选择目标社团时，本文从划分出的社团集合中选择一个社团 C_i，并构建该社团的邻接矩阵 S_i。矩阵 S_i 是一个维度为 $|C_i| \times |C_i|$ 的方阵，用于表示社团 C_i 内所有用户节点之间的连接权重。邻接矩阵 S_i 的每个元素 pcc_{uv} 表示社团 C_i 内用户节点 u 和用户节点 v 之间的相似度。

接下来，在邻接矩阵 S_i 上建立快速并行矩阵分解模型，目标是找到两个低秩矩阵 P_i 和 Q_i，使得 S_i 可以近似表示为 $P_i^{\top} Q_i$，其公式如(11)：

$$S_i \approx P_i^{\top} Q_i \tag{11}$$

其中 k 表示预先指定的潜在因子数，通常选择 $k \ll |C_i|$ 以减少模型的复杂性。矩阵 P_i 和 Q_i 分别表示社团中每个用户节点的潜在特征。在模型初始化时，通常使用随机初始化的方法为 P_i 和 Q_i 的每个元素赋初值。为了防止模型过拟合，需要在目标函数中加入正则化项，正则化参数 λ_P 和 λ_Q 用于限制模型的复杂性。

在定义优化目标时，本文的目标函数旨在最小化模型的预测值与实际观测值之间的误差，同时引入正则化项来防止过拟合。具体形式为(12)：

$$\min_{P_i, Q_i} \sum_{(u,v) \in C_i} \left(\left(pcc_{uv} - p_u^{\top} q_v \right)^2 + \lambda_P \left\| p_u \right\|^2 + \lambda_Q \left\| q_v \right\|^2 \right) \tag{12}$$

其中，pcc_{uv} 表示社团 C_i 内用户节点 u 和用户节点 v 之间的相似度，而 $p_u^{\top} q_v$ 是社团中对用户节点 u 和用户

节点 v 预测的相似度。误差项 e_{uv} 表示实际相似度与模型预测值之间的差异，定义为(13)：

$$e_{uv} = pcc_{uv} - p_u^{\top} q_v \tag{13}$$

此外，正则化项 $\lambda_P \|p_u\|^2 + \lambda_Q \|q_v\|^2$ 可以防止模型在训练过程中对数据过度拟合，从而确保模型在未见数据上的稳定表现。

为了进一步提高计算效率和模块收敛速度，采用了无锁调度和部分随机方法实现 SGD 算法对模块进行优化。在处理社团 C_i 的矩阵分解时，将评价矩阵 S_i 划分成多个数据块，并将这些快分配给不同的线程进行并行处理。每个线程处理数据块时，通过无锁调度机制，确保计算任务高效分配。同时，采用部分随机方法，在块的选择上引入随机性，以保证模型收敛的稳定性。而在块内 SGD 按顺序选择样本 (u,v) 进行梯度计算，以优化内存访问效率。对于每个块中的样本，逐步优化目标函数的公式如(14)：

$$\begin{aligned} p_u &\leftarrow p_u + \gamma\left(e_{uv} q_v - \lambda_P p_u\right) \\ q_v &\leftarrow q_v + \gamma\left(e_{uv} p_u - \lambda_Q q_v\right) \end{aligned} \tag{14}$$

其中，γ 是学习率，决定每次迭代中参数调整的步长。选择合适的学习率能够确保模型的快速收敛。对于社团 C_i 中所有用户对执行上述更新操作，直到达到预设的最大迭代次数或收敛条件。每次迭代的迭代过程都会逐步减小目标函数的值，从而逐步逼近最优解。在完成对一个社团的矩阵分解后，对其他社团 $C_1, C_2, \cdots, C_k$ 依次执行上述所有步骤，确保每个社团都能通过矩阵分解得到其内部的用户和物品潜在特征。所有社团的预测结果将被汇总，计算整体的 RMSE 和 MAE，以评估整个模型在整个数据集上的表现。

算法步骤：

遍历社团集合 C，为每一个社团 C_i 建立邻接矩阵 S_i，并计算用户节点之间的相似度。为每个社团的矩阵分解模型随机初始化潜在因子矩阵 P_i 和 Q_i。

将邻接矩阵 S_i 划分成多个数据块，并使用无锁调度机制将这些块分配给不同的线程进行并行处理。

每个线程在分配到的数据块上执行随机梯度下降优化，更新潜在因子矩阵 P_i 和 Q_i。

在模型上进行预测评分，计算均方根误差(RMSE)和平均绝对误差(MAE)来评估模型。

保存训练好的模型。

3.5. 评分预测

通过对每个划分出来的社团进行独立的快速并行矩阵分解，不仅能提取用户和物品的潜在特征，还能深入捕捉到社团内部的复杂关系，提高预测评分的准确性，其评分公式如：

$$\hat{r}_{ui} = p_u^T q_i \tag{15}$$

其中 $\hat{r}_{ui}$ 表示用户 u 对物品 i 的预测评分，p_u 是用户 u 在模型中的潜在特征向量，q_i 是物品 i 在模型中的特征向量。

4. 实验设计

4.1. 数据集

本文实验数据集为 MovieLens 25M、Movies_and_TV、TripAdvisor、Yelp。完整数据集信息如表 2 所示。

MovieLens 25M。该数据集是由 GroupLens Research 实验室提供的一个大规模电影评分数据集，包含大约 162,000 个用户、62,000 部电影以及 2500 万条评分记录。

Movies_and_TV。该数据集是 Amazon Product Review 数据集中提取的一个子集，主要包含用户对电

影和电视节目的评分和评论数据，包含 7506 个用户、7360 部电影或电视以及 441,783 条评分记录。

TripAdvisor。该数据集是关于旅游数据的数据集，包含 9765 个用户，6280 个酒店、餐厅或景点以及 320,023 条评分记录。

Yelp。包含用户对各种商家的评分，包含 27,147 个用户、20,266 个商家以及 1,293,247 条评分。

为了提高处理效率，本文从每个数据集中随机抽取了 3000 个用户的完整评分记录，作为本次实验的抽样数据集。这一抽样策略不仅有效降低了计算复杂度，还确保了实验的代表性和可靠性。此外，本文采用了折五验证方法，以验证所提算法的科学性和有效性。抽样数据集的统计信息如表 3 所示。

Table 2. Complete dataset statistics
表 2. 完整数据集统计信息

数据集	用户数量	物品数量	评分数量
MovieLens 25M	162,000	62,000	2500 万
Movies_and_TV	7506	7360	441,783
TripAdvisor	9765	6280	320,023
Yelp	27,147	20,266	1,293,247

Table 3. Sample dataset statistics
表 3. 抽样数据集统计信息

数据集	用户数量	物品数量	评分数量
MovieLens 25M	3000	36672	484,422
Movies_and_TV	3000	7360	180,163
TripAdvisor	3000	6280	97,828
Yelp	3000	19,766	142,265

4.2. 对比算法

本文通过对比实验评估 CDMF 模型的计算成本，关注其处理大规模数据集时的效率和资源利用，以及其在大规模稀疏数据处理上的表现，验证其在提高覆盖率和减少稀疏性方面的效果。因此，选择推荐系统中广泛应用且研究深入的经典算法 LIBMF、概率矩阵分解(Probabilistic Matrix Factorization, PMF)和奇异值分解扩展 SVD++ (Singular Value Decomposition Plus Plus, SVD++)作为基准算法。以下是几种基准算法的介绍。

PMF [27]：一种概率矩阵分解方法。PMF 假设用户－物品评分矩阵可以近似表示为两个潜在因子矩阵的乘积，基于高斯噪声假设，通过最大化后验概率来求解潜在因子矩阵。

SVD++ [28]：一种经典 SVD 方法的扩展算法。它在传统 SVD 的基础上增加了对隐式反馈信息的建模能力，从而改进预测性能。

LIBMF [13]：一种优化的并行随机梯度下降算法，能够在多线程环境下高效训练模型。通过优化数据存储结构和计算流程，LIBMF 显著减少了内存访问的开销，提升模型的收敛速度。

4.3. 时间性能分析

为保证统计结果稳健可靠，本文选取五折交叉验证中每折计算的最大运行时间作为最终统计值。这更贴近真实应用用户体验，反映系统应对高负载情况，并避免低估算法时间消耗，确保性能分析的全面

性和准确性。

表 4 展示了四种算法在四个不同数据集上的运行时间和相对于 CDMF 算法的性能表现。CDMF 的运行时间显著低于其他算法，尤其在 Movielens25m 上优势明显。CDMF 相比 LIBMF 缩短了 20%到 78.26%的运行时间。PMF 运行时间约为 CDMF 的 5 到 48 倍，性能提升 95%到 97.5%。SVD++运行时间远高于 CDMF，在大规模数据集上是 CDMF 的数百倍，性能提升达 96.31%至 99.7%。PMF 和 SVD++计算复杂度高，LIBMF 虽表现较好但仍不及 CDMF。

Table 4. Algorithm running times on various datasets
表 4. 各数据集上的算法运行时间

	Movielens25m	Yelp	TripAdvisor	Movies_and_TV
LIBMF	120 s (+77.5%)	37 s (+75.68%)	23 s (+78.26%)	10 s (+20%)
PMF	540 s (+95%)	260 s (+96.54%)	200 s (+97.5%)	240 s (+96.67%)
SVD++	150min (+99.7%)	35 s (+74.29%)	22 s (+77.27%)	217 s (+96.31%)
CDMF	**27 s (3 个社团)**	**9 s (7 个社团)**	**5 s (8 个社团)**	**8 s (4 个社团)**

4.4. 误差分析

平均绝对误差(Mean Absolute Error, MAE)和均方根误差(Root Mean Square Error, RMSE)是评价推荐系统预测准确性的两个常用指标，用来衡量预测评级与实际评级的距离。

$$\mathrm{MAE}=\frac{1}{|\mathrm{test}|}\sum_{r_{u,i}\in N}\left|r_{u,i}-r\right| \tag{16}$$

$$\mathrm{RMSE}=\sqrt{\frac{1}{|\mathrm{test}|}\sum_{r_{u,i}\in N}\left(r_{u,i}-r\right)^{2}} \tag{17}$$

其中$|\mathrm{test}|$为测试集的长度；r 为用户的真实评分。MAE 为预 测值与实际值的真实误差，RMSE 则是预测值与真实值偏差的平方，更加关注较大误差间的差距。

评分预测结果如表 5 所示。CDMF 在三个数据集上的 MAE 和 RMSE 指标优于其他算法，尤其在 Movielens25m 上表现出色。LIBMF 和 PMF 的 MAE 和 RMSE 值接近但略逊 CDMF。在 TripAdvisor 上，CDMF 的 RMSE 为 0.8768，LIBMF 为 0.8403，差距较小。SVD++在 TripAdvisor 上 RMSE 最低，但在其他数据集上不如 CDMF，尤其在 Yelp 上 RMSE 较高(1.1569)。

Table 5. MAE and RMSE evaluation results
表 5. MAE、RMSE 评估结果

	Movielens25m		Yelp		TripAdvisor		Movies_and_TV	
	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓
LIBMF	0.6403	0.8521	0.8840	1.1364	0.6454	0.8403	0.7433	0.9988
PMF	0.6628	0.8720	0.8870	1.1364	0.6377	0.8285	0.7556	0.9876
SVD++	0.6450	0.8562	0.9036	1.1569	**0.6269**	**0.8166**	0.7436	1.0096
CDMF	**0.6388**	**0.8499**	**0.8258**	**1.0559**	0.6744	0.8768	**0.7384**	**0.9811**

4.5. 准确性分析

准确率(Accuracy)、精确率(Precision)和召回率(Recall)是常用的分类指标之一。当对用户进行物品推荐时，TP 说明推荐了和用户喜好相关的物品；FP 说明推荐了与用户不相关的物品；FN 组表示系统未推荐给用户但是和用户喜好相关的物品；TN 组表示未将与用户喜好不相关的物品推荐给用户，表 6 显示了四种指标在混淆矩阵的情况。

Table 6. Classification of preference prediction results
表 6. 分类结果混淆矩阵

用户喜好	系统推荐	系统不推荐
相关	True-Positive (TP)	False-Negative (FN)
不相关	False-Positive (FP)	True-Negative (TN)

其中 TP 表示真阳性，TN 表示真阴性，FP 表示假阳性，FN 表示假阴性。

准确性(Accuracy)是指分类正确的样本数量与总样本数量的比例。准确性衡量了推荐系统成功推荐给用户感兴趣物品并避免不相关物品的能力。准确性越高，表示推荐效果也好。其公式如(18)所示：

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{18}$$

表 7 展示了四种算法在不同数据集上的准确率(Accuracy)表现。CDMF 在 Movielens25m 和 Movies_and_TV 上准确率较高。LIBMF 表现稳定，但在 Movielens25m 和 Movies_and_TV 上略低于 CDMF 和 SVD++。PMF 略高于 LIBMF，但在 Yelp 和 TripAdvisor 上不如 CDMF。SVD++在 TripAdvisor 上准确率最高(0.7866)。

Table 7. Accuracy evaluation metrics
表 7. Accuracy 评价指标

	Movielens25m	Yelp	TripAdvisor	Movies_and_TV
	Accuracy ↑	Accuracy ↑	Accuracy ↑	Accuracy ↑
LIBMF	0.7050	0.6661	0.7698	0.7678
PMF	0.7107	0.6715	0.7755	0.7711
SVD++	0.7171	0.6542	**0.7866**	0.7676
CDMF	**0.7174**	**0.6947**	0.7532	**0.7741**

精确率(Precision)衡量推荐列表中用户真正感兴趣的物品比例，越高说明推荐越精准，用户满意度越高。公式如(19)所示：

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{19}$$

在表 8 中，Pre@5 表示在推荐列表的前 5 个物品中，有多少比例是用户感兴趣的。同理，Pre@10 表示在推荐列表的前 10 个物品中，有多少比例是用户感兴趣的。

从表 8 中可以看出 CDMF 在 Yelp、TripAdvisor 和 Movies_and_TV 数据集上表现出色，尤其在 Yelp 数据集上 Re@5 和 Re@10 分别达到 0.7603 和 0.7617，表明能有效覆盖用户感兴趣的物品。SVD++在

Movielens25m 数据集上 Re@5 最高(0.8478)，召回能力强，但在其他数据集上不如 CDMF。PMF 在 Movielens25m 数据集上 Re@10 最高(0.8720)。LIBMF 在所有数据集上 Recall 均未达到最高值，尤其在 Yelp 数据集上 Re@10 最低(0.7519)，表明覆盖用户兴趣范围窄。

Table 8. Precision evaluation metrics
表 8. Precision 评价指标

	Movielens25m		Yelp		TripAdvisor		Movies_and_TV	
	Pre@5 ↑	Pre@10 ↑	Pre@5 ↑	Pre@10 ↑	Pre@5 ↑	Pre@10 ↑	Pre@5 ↑	Pre@10 ↑
LIBMF	0.8336	0.8238	0.7504	0.7466	0.8850	0.8695	0.8850	0.8695
PMF	0.8466	**0.8317**	0.7568	0.7426	0.8752	0.8626	0.8752	0.8626
SVD++	**0.8472**	0.8303	0.7468	0.7436	0.8830	0.8638	0.8830	0.8638
CDMF	0.8383	0.8254	**0.7601**	**0.7563**	**0.8914**	**0.8745**	**0.8914**	**0.8745**

召回率(Recall)衡量推荐系统覆盖用户实际兴趣范围的能力，越高说明推荐系统越能全面地满足用户需求。公式如(20)所示：

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{20}$$

在表 9 中，Re@5 表示在推荐列表的前 5 个物品中，成功推荐给用户的相关物品占用户所有感兴趣物品的比例。同理，Re@10 则表示在推荐列表的前 10 个物品中，成功推荐给用户的相关物品占用户所有感兴趣物品的比例。

从表 9 中数据可以看出，CDMF 在 Yelp、TripAdvisor 和 Movies_and_TV 数据集上表现突出，尤其在 Yelp 上 Re@5 和 Re@10 分别高达 0.7603 和 0.7617，意味着推荐列表能有效覆盖用户兴趣。SVD++在 Movielens25m 数据集上 Re@5 最高(0.8478)，召回能力强，但在其他数据集上不如 CDMF。PMF 在 Movielens25m 数据集上 Re@10 表现最佳(0.8720)。LIBMF 在所有数据集上的 Recall 均未达最高，尤其在 Yelp 上 Re@10 仅为 0.7519，用户兴趣覆盖范围较窄。

Table 9. Recall evaluation metrics
表 9. Recall 评价指标

	Movielens25m		Yelp		TripAdvisor		Movies_and_TV	
	Re@5 ↑	Re@10 ↑	Re@5 ↑	Re@10 ↑	Re@5 ↑	Re@10 ↑	Re@5 ↑	Re@10 ↑
LIBMF	0.8343	0.8521	0.7504	0.7519	0.8850	0.8704	0.8225	0.8231
PMF	0.8475	**0.8720**	0.7571	0.7479	0.8753	0.8635	0.8367	0.8351
SVD++	**0.8478**	0.8562	0.7469	0.7492	0.8830	0.8649	0.8204	0.8224
CDMF	0.8333	0.8500	**0.7603**	**0.7617**	**0.8914**	**0.8756**	**0.8393**	**0.8357**

4.6. 参数分析

本文涉及的所有相关算法在相同的实验背景下进行，采用相同的编程语言。在参数设置上，本文尽量保持与原文中对比方法的设置一致。在本文提出的模型中，图 4 展示了不同参数下 CDMF 算法准确度的表现，由于在不同的数据集中不同参数下 MAE 和 RMSE 呈现的趋势一样，本文展示了在 MovieLens25m

数据集下的趋势图。从图 4(a)中可以看出，随着迭代次数的增加，RMSE 和 MAE 都呈现出逐渐下降的趋势，这表明模型的误差在不断减少，模型的预测能力在不断提升。在迭代达到 5000 次后，误差趋于平稳，因此设置迭代次数 iter 为 5000。图 4(b)展示了 RMSE 和 MAE 在潜在因子 k 值为 20 到 50 之间保持稳定，几乎没有明显变化，因此本文选择将潜在因子 k 设置为 35。图 4(c)展示了学习率从 0.005 增加到 0.010 时，RMSE 和 MAE 误差明显下降，并在 0.01 处趋于平稳，因此学习率 eta 设置为 0.01。在图 4(d)中展示了随着补充高频用户的数量从 10 增加到 50 时，RMSE 随之缓慢上升，趋于平稳，大致维持在 0.85 附近。而 MAE 则保持相对较低的值，约为 0.65，没有显著的波动，尽管 RMSE 随着用户数量增加有所上升，考虑到误差控制的需要，本文选择了最活跃的前 25%用户作为高频用户群体。这一选择在一定范围内平衡了用户数量的增加和误差的稳定性，确保模型在可控误差范围内，能够从高频用户中获取有价值的信息。上述所有参数值均通过网格搜索确定，达到实验的最佳效果。

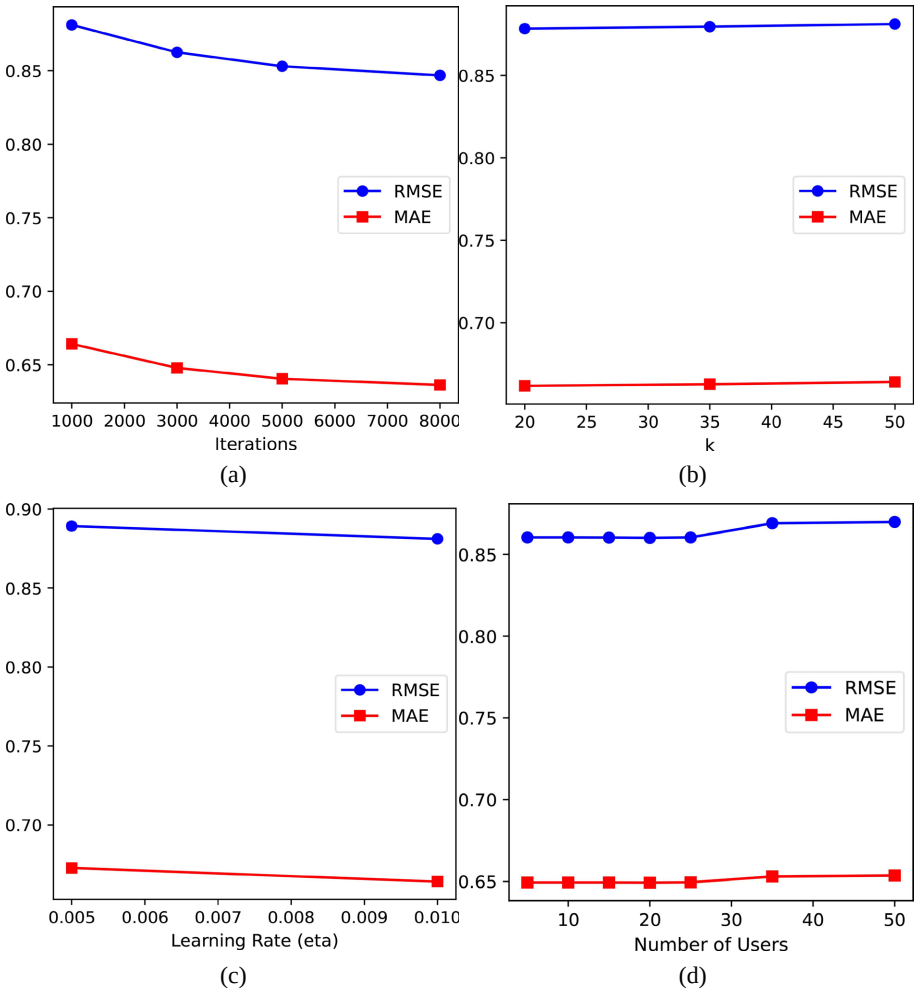

Figure 4. MAE and RMSE of CDMF under different parameters
图 4. CDMF 不同参数下的 MAE 和 RMSE

5. 总结

针对现实中大规模数据集处理中的复杂度和资源消耗问题，以及深度学习模型在这一场景下的挑战，本文设计了一种基于用户相似性网络的快速并行梯度下降矩阵分解算法(CDMF)。该方法首先构建用户相

似性复杂网络，然后采用模块度优化的社团检测算法对用户网络分组，将高相似度用户划分到相同社团，最后在社团网络中采用并行梯度下降算法进行矩阵分解，从而提升推荐系统的可扩展性和准确性。

实验结果表明，CDMF 算法在多个关键指标上表现突出：1) 在推荐精度方面，CDMF 在大部分数据集上表现最佳，成功覆盖更多用户感兴趣的物品。2) 在 RMSE 和 MAE 方面，CDMF 在 Yelp 和 TripAdvisor 数据集上误差最低，预测准确性更高。3) 在准确率方面，CDMF 在 Movielens25m 和 Movies_and_TV 数据集上同样表现出色。4) 在可扩展性方面，处理大规模数据集时，CDMF 的运行时间显著优于其他算法。

未来，随着推荐系统广泛普及，处理海量数据的需求日益增长。基于本文提出的融合用户相似性网络社团特征与梯度下降矩阵分解的推荐算法，未来研究可以进一步探讨以下方向：

一是结合复杂网络中的中心性信息，更准确地识别具有关键影响力的用户，优化社团划分和用户相似度度量，提升分组的合理性与精确度，进而提高推荐结果的个性化和准确性。二是融合深度学习技术，结合神经网络、注意力机制等方法，捕捉更复杂的用户行为和偏好模式，提升算法在高维数据和非线性关系处理中的自适应性，实时调整推荐策略。

基金项目

国家自然科学基金项目，基于复杂网络时空一体化模型的节点中心性研究(61803264)。

参考文献

[1] Shi, X., He, Q., Luo, X., Bai, Y. and Shang, M. (2020) Large-Scale and Scalable Latent Factor Analysis via Distributed Alternative Stochastic Gradient Descent for Recommender Systems. *IEEE Transactions on Big Data*, **8**, 420-431. https://doi.org/10.1109/tbdata.2020.2973141

[2] Khojamli, H. and Razmara, J. (2021) Survey of Similarity Functions on Neighborhood-Based Collaborative Filtering. *Expert Systems with Applications*, **185**, Article ID: 115482. https://doi.org/10.1016/j.eswa.2021.115482

[3] Wei, W., Huang, C., Xia, L. and Zhang, C. (2023) Multi-Modal Self-Supervised Learning for Recommendation. *Proceedings of the ACM Web Conference* 2023, Austin, 30 April-4 May 2023, 790-800. https://doi.org/10.1145/3543507.3583206

[4] Ashokan, A. and Haas, C. (2021) Fairness Metrics and Bias Mitigation Strategies for Rating Predictions. *Information Processing & Management*, **58**, Article ID: 102646. https://doi.org/10.1016/j.ipm.2021.102646

[5] Yu, R., Liu, Q., Ye, Y., Cheng, M., Chen, E. and Ma, J. (2020) Collaborative List-And-Pairwise Filtering from Implicit Feedback. *IEEE Transactions on Knowledge and Data Engineering*, **34**, 2667-2680. https://doi.org/10.1109/tkde.2020.3016732

[6] Li, H., Li, K., An, J. and Li, K. (2022) An Online and Scalable Model for Generalized Sparse Nonnegative Matrix Factorization in Industrial Applications on Multi-GPU. *IEEE Transactions on Industrial Informatics*, **18**, 437-447. https://doi.org/10.1109/tii.2019.2896634

[7] Shi, H., Sun, T., Li, S., *et al*. (2019) A Matrix Factorization Recommendation Algorithm with Time and Type Weight. *Journal of Chongqing University*, **42**, 79-87.

[8] Wu, Z., Zhou, Y., Wu, D., Chen, M. and Xu, Y. (2019) TAMF: Towards Personalized Time-Aware Recommendation for Over-the-Top Videos. *Proceedings of the* 29*th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, Amherst, 21 June 2019, 43-48. https://doi.org/10.1145/3304112.3325606

[9] Guo, G., Yang, E., Shen, L., Yang, X. and He, X. (2019) Discrete Trust-Aware Matrix Factorization for Fast Recommendation. *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, Macao, 10-16 August 2019, 1380-1386. https://doi.org/10.24963/ijcai.2019/191

[10] Lei, C., Dai, H., Yu, Z. and Li, R. (2020) A Service Recommendation Algorithm with the Transfer Learning Based Matrix Factorization to Improve Cloud Security. *Information Sciences*, **513**, 98-111. https://doi.org/10.1016/j.ins.2019.10.004

[11] Jafri, S.I.H., Ghazali, R., Javid, I., Mahmood, Z. and Hassan, A.A.A. (2022) Deep Transfer Learning with Multimodal Embedding to Tackle Cold-Start and Sparsity Issues in Recommendation System. *PLOS ONE*, **17**, e0273486. https://doi.org/10.1371/journal.pone.0273486

[12] Guo, R., Zhang, F., Wang, L., Zhang, W., Lei, X., Ranjan, R., *et al*. (2021) Bapa: A Novel Approach of Improving Load

Balance in Parallel Matrix Factorization for Recommender Systems. *IEEE Transactions on Computers*, **70**, 789-802. https://doi.org/10.1109/tc.2020.2997051

[13] Liu, X., Ho, C., Zheng, S. and Yuan, S. (2019) Comprehensive Evaluation of Large-Scale Parallel Matrix Factorization Algorithms. 2019 *IEEE* 21*st International Conference on High Performance Computing and Communications*; *IEEE* 17*th International Conference on Smart City*; *IEEE* 5*th International Conference on Data Science and Systems* (*HPCC/SmartCity/DSS*), Zhangjiajie, 10-12 August 2019, 811-818. https://doi.org/10.1109/hpcc/smartcity/dss.2019.00119

[14] Chin, W., Zhuang, Y., Juan, Y. and Lin, C. (2015) A Fast Parallel Stochastic Gradient Method for Matrix Factorization in Shared Memory Systems. *ACM Transactions on Intelligent Systems and Technology*, **6**, 1-24. https://doi.org/10.1145/2668133

[15] Yu, H., Hsieh, C., Si, S. and Dhillon, I. (2012) Scalable Coordinate Descent Approaches to Parallel Matrix Factorization for Recommender Systems. 2012 *IEEE* 12*th International Conference on Data Mining*, Brussels, 10-13 December 2012, 765-774. https://doi.org/10.1109/icdm.2012.168

[16] Lee, H. and Kim, Y. (2020) Stochastic Gradient Descent for Matrix Completion: Hybrid Parallelization. *Concurrency and Computation*: *Practice and Experience*, **32**, e5662.

[17] Li, Q., Xiong, D. and Shang, M. (2022) Adjusted Stochastic Gradient Descent for Latent Factor Analysis. *Information Sciences*, **588**, 196-213. https://doi.org/10.1016/j.ins.2021.12.065

[18] Chen, J. and Wang, L. (2020) Parallel Fractional Stochastic Gradient Descent with Adaptive Learning. *IEEE Transactions on Neural Networks and Learning Systems*, **31**, 4053-4065.

[19] Liu, Y., Wang, S,. Li, X., *et al*. (2024) A Meta-Adversarial Framework for Cross-Domain Cold-Start Recommendation. *Data Science and Engineering*, **9**, 238-249.

[20] Zhang, C., Yang, Y., Zhou, W. and Zhang, S. (2022) Distributed Bayesian Matrix Decomposition for Big Data Mining and Clustering. *IEEE Transactions on Knowledge and Data Engineering*, **34**, 3701-3713. https://doi.org/10.1109/tkde.2020.3029582

[21] Hansel, A.C., Adrianus, Pradana, L., Suganda, A., Girsang, and Nugroho, A. (2022) Optimized LightGCN for Music Recommendation Satisfaction. 2022 6*th International Conference on Information Technology, Information Systems and Electrical Engineering* (*ICITISEE*), Yogyakarta, 13-14 December 2022, 449-454. https://doi.org/10.1109/icitisee57756.2022.10057831

[22] Wu, J., Wang, X., Feng, F., He, X., Chen, L., Lian, J., *et al*. (2021) Self-Supervised Graph Learning for Recommendation. *Proceedings of the* 44*th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 11-15 July 2021, 726-735. https://doi.org/10.1145/3404835.3462862

[23] 甘宏. 基于异质信息网络与图注意力的深度学习推荐算法研究[J]. 江西科学, 2023, 41(4): 788-793.

[24] Su, Z., Lin, Z., Ai, J. and Li, H. (2021) Rating Prediction in Recommender Systems Based on User Behavior Probability and Complex Network Modeling. *IEEE Access*, **9**, 30739-30749. https://doi.org/10.1109/access.2021.3060016

[25] Newman, M.E.J. (2004) Fast Algorithm for Detecting Community Structure in Networks. *Physical Review E*, **69**, Article ID: 066133. https://doi.org/10.1103/physreve.69.066133

[26] Blondel, V., Guillaume, J.L. and Lambiotte, R. (2023) Fast Unfolding of Communities in Large Networks: 15 Years Later. arXiv:2311.06047. https://arxiv.org/abs/2311.06047

[27] 柯建坤, 许忠好. Louvain 算法与 K 均值聚类算法的比较研究[J].应用概率统计, 2022, 38(5): 780-790.

[28] Xu, S., Zhuang, H., Sun, F., Wang, S., Wu, T. and Dong, J. (2021) Recommendation Algorithm of Probabilistic Matrix Factorization Based on Directed Trust. *Computers & Electrical Engineering*, **93**, Article ID: 107206. https://doi.org/10.1016/j.compeleceng.2021.107206