Modeling and Simulation 建模与仿真, 2023, 12(3), 1807-1819
Published Online May 2023 in Hans. https://www.hanspub.org/journal/mos
https://doi.org/10.12677/mos.2023.123167

Hans汉斯

融合用户信任关系和神经网络的推荐模型

于佳玄[1]，丁德锐[2*]

[1]上海理工大学理学院，上海
[2]上海理工大学控制科学与工程系，上海

收稿日期：2023年2月24日；录用日期：2023年5月1日；发布日期：2023年5月8日

摘　要

矩阵分解(Matrix Factorization, MF)是构建推荐系统的有效方法之一。然而，传统的矩阵分解模型在应用于大规模稀疏数据场景时依然存在推荐准确率较差、训练速度较慢等严重不足。本文从提高推荐准确率和减少时间花销这两方面考虑，提出一种基于神经网络(Neural Network, NN)的融合用户信任关系(Trust)的矩阵分解(NN-MF-TR)模型。一方面，通过用户间潜在的信任与被信任关系充分挖掘用户的潜在偏好，从而提高推荐准确率。另一方面，在训练模型时引入神经网络，减少因多次迭代所造成的巨大时间花销。最后，通过选取四个有代表性的对比模型并在四个真实世界数据集上进行了实验验证。实验结果表明，本文提出的NN-MF-TR模型能够在保证提高推荐准确率的同时大大减少时间花销。

关键词

矩阵分解，信任关系，被信任关系，神经网络

A Recommendation Model Integrating User Trust Relationship and Neural Network

Jiaxuan Yu[1], Derui Ding[2*]

[1]College of Science, University of Shanghai for Science and Technology, Shanghai
[2]Department of Control Science and Engineering, University of Shanghai for Science and Technology, Shanghai

Received: Feb. 24th, 2023; accepted: May 1st, 2023; published: May 8th, 2023

Abstract

Matrix factorization is one of the effective methods to construct recommendation system. However, the traditional matrix factorization model still faces the problems of both poor recommenda-

*通讯作者。

文章引用: 于佳玄, 丁德锐. 融合用户信任关系和神经网络的推荐模型[J]. 建模与仿真, 2023, 12(3): 1807-1819.
DOI: 10.12677/mos.2023.123167

tion accuracy and low training speed when applied to the case with large-scale sparse data. In order to improve the recommendation accuracy while reducing the time cost, this paper proposes an NN-based matrix factorization model combining user trust relationships (NN-MF-TR). First, the latent preference of users is fully mined through the latent trust and trusted relationships between users, so as to improve the recommendation accuracy. Then, a neural network is introduced to train the constructed model to reduce the huge time cost caused by multiple iterations. Finally, four representative comparison models are selected and tested on four real-world data sets. The results show that the NN-MF-TR model proposed in this paper can greatly reduce the time cost while ensuring the recommendation accuracy.

Keywords

Matrix Factorization, Trust Relationship, Trusted Relationship, Neural Network

1. 引言

推荐系统[1] [2]可以从互联网上收集有关用户的偏好等信息，旨在为客户提供个性化推荐。推荐系统需要在较短时间内处理海量不同且稀疏的数据，矩阵分解[3] [4] [5] [6]技术已被证明对于实现推荐系统是有用且可靠的。随着网络的发展使得用户和条目激增，导致评分矩阵中数据极度稀疏，使得大多数现有的推荐模型准确性较差。在现实世界中，我们总会向信任的朋友寻求推荐，所以信任关系可以作为推荐的辅助信息来缓解评分信息的稀疏[6] [7] [8]。然而，很少有研究从信任与被信任信息这一角度去探讨用户间的关联。因此，本文基于矩阵分解方法，利用用户间潜在的信任与被信任信息，来提高推荐准确率。

在如今这个快节奏时代中，效率一词成为大家共同追求的目标，模型的时间效率已逐渐成为判断其性能是否优秀的标准之一。目前绝大多数的模型都通过随机梯度下降算法[9] [10] [11]等迭代学习算法[12]进行求解。尽管这种算法被广泛采纳，但是它需要通过多次迭代才会收敛，在大规模的数据集中，它的时间效率不尽如人意，用非迭代算法替换迭代算法是至关重要的。基于神经网络的训练过程是非迭代的，会大大提高模型的时间效率。

综上，本文提出的 NN-MF-TR 模型的主要贡献如下：

1) 利用用户间的信任与被信任关系提高推荐系统的性能。在提取用户的潜在信任与被信任关系时，利用图拉普拉斯正则化，不仅保留了用户间的结构信息，还提高了模型的泛化能力。

2) 利用神经网络对模型进行训练，用非迭代算法替换迭代算法，大大提高模型的时间效率。

3) 四个真实世界的数据集上的实验结果可以表明，本文提出的模型在提高推荐准确率的同时大大提高了时间效率。

2. 相关工作

2.1. 矩阵分解

基于推荐系统的矩阵分解[3] [4]定义如下：假设 $\boldsymbol{R}=\left[r_{i,j}\right]_{M\times N}$ 为评分矩阵，$r_{i,j}$ 代表用户 i 对条目 j 给出的评分，M 为用户的数量，N 为条目的数量。$\boldsymbol{R}$ 可分解成两个低秩矩阵的近似乘积，即用户特征矩阵

$\boldsymbol{P}=\{p_i\}\in\mathbb{R}^{M\times k}$ 和条目特征矩阵 $\boldsymbol{Q}=\{q_j\}\in\mathbb{R}^{k\times N}$ 的乘积，损失函数为：

$$L=\frac{1}{2}\sum_{r_{i,j}\in\Lambda}\left(r_{i,j}-p_iq_j\right)^2+\frac{\lambda}{2}\left(\sum_i\|p_i\|_F^2+\sum_j\|q_j\|_F^2\right) \tag{1}$$

其中，Λ 表示已知评分，λ 是可以防止模型过拟合的正则化系数。

2.2. 自编码神经网络

自编码神经网络[5] [13] [14]相比于矩阵分解较为新颖，它通过捕获非线性特征为推荐系统提供了有竞争力的结果。基于自编码的模型由一个三层神经网络构成，即输入层、隐藏层和输出层[5] [13] [14]，图 1 是它的结构模型。自编码神经网络由编码和解码两阶段建立，通过重构输入的数据，使输出数据 y 与输入数据 x 相近。其损失函数可由欧式距离构建，具体如下：

$$J=\frac{1}{2}\sum_{i=1}^{m}\|x_i-y_i\|_F^2 \tag{2}$$

其中，m 为输入层数据的个数。

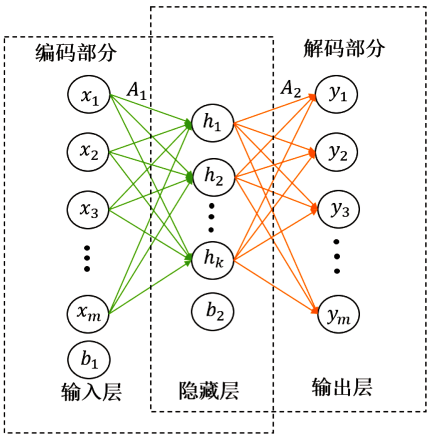


Figure 1. Autoencoder neural network model
图 1. 自编码神经网络模型

3. NN-MF-TR 模型

3.1. 提取用户潜在的信任与被信任关系

基于分解评分矩阵的方法[3] [4]同样可以分解信任矩阵，$\boldsymbol{T}=\left[t_{ij}\right]_{M\times M}$ 为用户的信任矩阵，已知 $\boldsymbol{T}$ 为对称 0，1 矩阵，1 代表用户间存在信任关系，0 代表用户间不存在信任关系，则可以将 $\boldsymbol{T}$ 分解成潜在的信任特征矩阵 $\boldsymbol{Z}$ 和潜在的被信任特征矩阵 $\boldsymbol{W}$ 的乘积。为了提高信任特征与被信任特征的准确性，利用图拉普拉斯正则化[15] [16]可将用户间的结构关系保留到潜在的信任与被信任特征之中，同时防止模型过拟合并提高了模型的泛化能力。具体提取用户潜在信任与被信任特征公式如下：

$$L=\frac{1}{2}\|\boldsymbol{T}-\boldsymbol{ZW}\|_F^2+\frac{\lambda_Z}{2}tr\left(\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{L}_Z\boldsymbol{Z}\right)+\frac{\lambda_W}{2}tr\left(\boldsymbol{WL}_W\boldsymbol{W}^{\mathrm{T}}\right) \tag{3}$$

其中，λ_Z、λ_W 分别为信任特征矩阵和被信任特征矩阵的正则化参数，$\boldsymbol{L}_Z$、$\boldsymbol{L}_W$ 分别为信任特征矩阵和被信任特征矩阵的拉普拉斯矩阵。

3.2. NN-MF-TR 算法

本文的模型基于矩阵分解[3] [4]，利用用户的潜在信任与被信任特征作为辅助信息来缓解目前评分信息的大规模缺失所导致的预测效果较不理想等问题，根据式(3)可以得到用户的潜在信任特征和潜在被信任特征，具体公式如下：

$$J(\boldsymbol{U},\boldsymbol{V})=\sum_{r_{i,j}\in\Lambda}\left(r_{i,j}-\left(u_{i,\cdot}+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^{+}(i)\right|^{-\frac{1}{2}}\sum_{s\in T^{+}(i)}w_{s,\cdot}\right)v_{j,\cdot}\right)^{2}+\lambda_{U}\left\|u_{i,\cdot}\right\|+\lambda_{V}\left\|v_{j,\cdot}\right\| \tag{4}$$

其中，$u_{i,\cdot}$表示潜在因子矩阵$\boldsymbol{U}$中的第i行元素，$v_{j,\cdot}$表示潜在因子矩阵$\boldsymbol{V}$中的第j行元素，$T(i)$表示用户i信任的用户集合，$T^{+}(i)$表示信任用户i的用户集合，$\sum_{s\in T(i)}z_{s,\cdot}$表示用户$i$潜在信任特征矩阵的第$s$行，$\sum_{s\in T^{+}(i)}w_{s,\cdot}$表示用户$i$潜在被信任特征矩阵的第$s$行，$\lambda_{U}$、$\lambda_{V}$为正则化参数。

3.3. 基于神经网络进行参数训练

为了使模型实现更高的计算效率，本文利用神经网络对模型的参数进行训练。假设模型中神经网络的层数为n，则过程如下：

1) 当$n=1$时，即神经网络的层数为单层。训练评分矩阵$\boldsymbol{R}$，将矩阵$\boldsymbol{R}$的每一行作为输入，然后训练潜在因子矩阵$\boldsymbol{U}^{1}$，通过$\boldsymbol{U}^{1}$可得到$\boldsymbol{V}^{1}$。

2) 当$n\geq 2$时，即神经网络的层数为多层。第一层的训练过程和单层神经网络模型相同；从第二层开始，每次训练都是将上一层训练出的潜在因子矩阵$\boldsymbol{U}^{n-1}$的每一行作为下一层的输入，从而训练出当前层潜在因子矩阵$\boldsymbol{U}^{n}$，通过$\boldsymbol{U}^{n}$可得到$\boldsymbol{V}^{n}$。

接下来，将基于梯度法对潜在因子矩阵$\boldsymbol{U}^{n}$，$\boldsymbol{V}^{n}$进行求解。

1) 当$n=1$时，研发的求解步骤如下：

首先，将评分矩阵$\boldsymbol{R}$中的已知评分作为输入，通过随机学习算法[17] [18] [19]生成权重矩阵$\boldsymbol{A}^{k\times M}$和偏置向量$\boldsymbol{b}^{1\times k}$，然后可得到初始化后的潜在因子矩阵$\boldsymbol{U}^{1}$：

$$\boldsymbol{U}^{1}=\begin{bmatrix}u_{1}^{1}\\u_{2}^{1}\\\vdots\\u_{M}^{1}\end{bmatrix}=\begin{bmatrix}f\left(a_{1,\cdot}r_{1,\cdot}^{\mathrm{T}}+b_{1}\right)&\cdots&f\left(a_{k,\cdot}r_{1,\cdot}^{\mathrm{T}}+b_{k}\right)\\\vdots&\ddots&\vdots\\f\left(a_{1,\cdot}r_{M,\cdot}^{\mathrm{T}}+b_{1}\right)&\cdots&f\left(a_{k,\cdot}r_{M,\cdot}^{\mathrm{T}}+b_{k}\right)\end{bmatrix} \tag{5}$$

其中，$a_{1,\cdot},\cdots,a_{k,\cdot}$分别为权重矩阵$\boldsymbol{A}^{k\times M}$的第$1,\cdots,k$行元素，$b_{1},\cdots b_{k}$分别为偏置向量$\boldsymbol{b}^{1\times k}$中的第$1,\cdots,k$个元素。本文设置激活函数为$f(x)=\left(1-\mathrm{e}^{-x}\right)\Big/\left(1+\mathrm{e}^{-x}\right)$ [20]，该方法与其他激活函数兼容。

接下来，利用初始化后的潜在因子矩阵$\boldsymbol{U}^{1}$求解潜在因子矩阵$\boldsymbol{V}^{1}$，具体算法如下：

$$\frac{\partial J}{\partial v_{j,\cdot}}=-2\sum_{i\in\Lambda(j)}\left(u_{i,\cdot}^{1}+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^{+}(i)\right|^{-\frac{1}{2}}\sum_{s\in T^{+}(i)}w_{s,\cdot}\right)^{\mathrm{T}}\times\left(r_{i,j}-\left(u_{i,\cdot}^{1}+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^{+}(i)\right|^{-\frac{1}{2}}\sum_{s\in T^{+}(i)}w_{s,\cdot}\right)v_{j,\cdot}^{1}\right)+2\left|\Lambda(j)\right|\lambda_{V}v_{j,\cdot}^{1} \tag{6}$$

其中，$i\in\Lambda(j)$矩阵每行中已知元素集合。

令 $\partial J/\partial v_{j,\cdot}^1=0$，可得到：

$$\begin{aligned}&\left(\sum_{i\in\Lambda(j)}\left(u_{i,\cdot}^1+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^+(i)\right|^{-\frac{1}{2}}\sum_{s\in T^+(i)}w_{s,\cdot}\right)^{\mathrm{T}}\right.\\&\left.\times\left(u_{i,\cdot}^1+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^+(i)\right|^{-\frac{1}{2}}\sum_{s\in T^+(i)}w_{s,\cdot}\right)+\left|\Lambda(j)\right|\lambda_V\boldsymbol{E}\right)v_{j,\cdot}^1\\&=\sum_{i\in\Lambda(j)}\left(u_{i,\cdot}^1+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^+(i)\right|^{-\frac{1}{2}}\sum_{s\in T^+(i)}w_{s,\cdot}\right)^{\mathrm{T}}r_{i,j}\end{aligned}\tag{7}$$

其中，$\boldsymbol{E}$ 是一个 $k\times k$ 维的单位矩阵。

整理可得到更新公式如下：

$$\begin{aligned}v_{j,\cdot}^1=&\left(\sum_{i\in\Lambda(j)}\left(u_{i,\cdot}^1+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^+(i)\right|^{-\frac{1}{2}}\sum_{s\in T^+(i)}w_{s,\cdot}\right)^{\mathrm{T}}\right.\\&\left.\times\left(u_{i,\cdot}^1+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^+(i)\right|^{-\frac{1}{2}}\sum_{s\in T^+(i)}w_{s,\cdot}\right)+\left|\Lambda(j)\right|\lambda_V\boldsymbol{E}\right)^{-1}\\&\times\left(\sum_{i\in\Lambda(j)}\left(u_{i,\cdot}^1+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^+(i)\right|^{-\frac{1}{2}}\sum_{s\in T^+(i)}w_{s,\cdot}\right)^{\mathrm{T}}r_{i,j}\right)\end{aligned}\tag{8}$$

最后，再一次对潜在因子矩阵 $\boldsymbol{U}^1$ 进行更新。因为最初的 $\boldsymbol{U}^1$ 具有随机性，为了使结果更近似于评分矩阵 $\boldsymbol{R}$，则对 $\boldsymbol{U}^1$ 再次更新。$\boldsymbol{U}^1$ 的更新方法与 $\boldsymbol{V}^1$ 相同，具体算法如下：

$$\frac{\partial J}{\partial u_{i,\cdot}^1}=-2\sum_{j\in\Lambda(i)}\left(r_{i,j}-\left(u_{i,\cdot}^1+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^+(i)\right|^{-\frac{1}{2}}\sum_{s\in T^+(i)}w_{s,\cdot}\right)v_{j,\cdot}^1\right)\left(v_{j,\cdot}^1\right)^{\mathrm{T}}+2\left|\Lambda(i)\right|\lambda_U u_{i,\cdot}^1\tag{9}$$

令 $\partial J/\partial u_{i,\cdot}^1=0$，可得到：

$$u_{i,\cdot}^1=\left(\sum_{j\in\Lambda(i)}r_{i,j}\left(v_{j,\cdot}^1\right)^{\mathrm{T}}\right)\left(\sum_{j\in\Lambda(i)}v_{j,\cdot}^1\left(v_{j,\cdot}^1\right)^{\mathrm{T}}+\left|\Lambda(i)\right|\lambda_U\boldsymbol{E}\right)^{-1}\tag{10}$$

2) 当 $n\geq 2$ 时，需要注意的是，每层的输入数据为上一层的潜在因子矩阵 $\boldsymbol{U}^{n-1}$ 中的元素，具体算法如下：

$$\boldsymbol{U}^n=\begin{bmatrix}u_{1,\cdot}^n\\\vdots\\u_{M,\cdot}^n\end{bmatrix}=\begin{bmatrix}f\left(c_{1,\cdot}^n\left(u_{1,\cdot}^{n-1}\right)^{\mathrm{T}}+d_1\right)&\cdots&f\left(c_{k,\cdot}^n\left(u_{1,\cdot}^{n-1}\right)^{\mathrm{T}}+d_k\right)\\\vdots&\ddots&\vdots\\f\left(c_{1,\cdot}^n\left(u_{M,\cdot}^{n-1}\right)^{\mathrm{T}}+d_1\right)&\cdots&f\left(c_{1,\cdot}^n\left(u_{M,\cdot}^{n-1}\right)^{\mathrm{T}}+d_k\right)\end{bmatrix}\tag{11}$$

其中，$c_{1,\cdot},\cdots,c_{k,\cdot}$ 分别为权重矩阵 $\boldsymbol{C}^{k\times M}$ 的第 $1,\cdots,k$ 行元素，$d_1,\cdots d_k$ 分别为偏置向量 $\boldsymbol{D}^{1\times k}$ 中的第 $1,\cdots,k$ 个元素。

接下来，利用梯度法更新潜在因子矩阵 $\boldsymbol{V}^n$，得到公式如下：

$$v_{j,\cdot}=\left(\sum_{i\in\Lambda(j)}\left(u_{i,\cdot}^{n}+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^{+}(i)\right|^{-\frac{1}{2}}\sum_{s\in T^{+}(i)}w_{s,\cdot}\right)^{\mathrm{T}}\right.$$
$$\times\left(u_{i,\cdot}^{n}+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^{+}(i)\right|^{-\frac{1}{2}}\sum_{s\in T^{+}(i)}w_{s,\cdot}\right)+\left|\Lambda(j)\right|\lambda_{V}\boldsymbol{E}\Bigg)^{-1} \tag{12}$$
$$\left.\times\left(\sum_{i\in\Lambda(j)}\left(u_{i,\cdot}^{n}+\left|T(i)\right|^{-\frac{1}{2}}\sum_{s\in T(i)}z_{s,\cdot}+\left|T^{+}(i)\right|^{-\frac{1}{2}}\sum_{s\in T^{+}(i)}w_{s,\cdot}\right)^{\mathrm{T}}r_{i,j}\right)\right.$$

最后，再一次对潜在因子矩阵 $\boldsymbol{U}^{n}$ 进行更新，可得到：

$$u_{i,\cdot}^{n}=\left(\sum_{j\in\Lambda(i)}r_{i,j}\left(v_{j,\cdot}^{n}\right)^{\mathrm{T}}\right)\left(\sum_{j\in\Lambda(i)}v_{j,\cdot}^{n}\left(v_{j,\cdot}^{n}\right)^{\mathrm{T}}+\left|\Lambda(i)\right|\lambda_{U}\boldsymbol{E}\right) \tag{13}$$

3.4. 算法设计和复杂度分析

基于以上公式，本文设计模型的算法流程如下：

输入：信任矩阵 $\boldsymbol{T}$，评分矩阵 $\boldsymbol{R}$ 中的已知评分数据集合 Λ，k，n，正则化参数 λ_{Z}，λ_{W}，λ_{U}，λ_{V}

输出：潜在信任特征矩阵 $\boldsymbol{Z}$，潜在被信任特征矩阵 $\boldsymbol{W}$，潜在因子矩阵 $\boldsymbol{U}^{n}$ 和 $\boldsymbol{V}^{n}$

Step 1：根据式(3)求解潜在信任特征矩阵 $\boldsymbol{Z}$，潜在被信任特征矩阵 $\boldsymbol{W}$

Step 2：根据式(11)求解潜在因子矩阵 $\boldsymbol{U}^{n}$

Step 3：根据式(12)求解潜在因子矩阵 $\boldsymbol{V}^{n}$

Step 4：根据式(13)更新潜在因子矩阵 $\boldsymbol{U}^{n}$

显然，本文所提算法的时间开销主要来源于计算公式(12)和(13)中的逆矩阵。通过计算得到公式(12)和(13)的时间复杂度均为 $\Theta\left(k^{3}\right)$，其中 k 表示潜在因子维数。由此可以看出，潜在因子维数越小，则时间花销越少。此外，神经网络层数 n 的取值也对时间花销有着重要的影响，层数越多，则训练模型的时间成本也越大。

4. 实验及结果分析

4.1. 实验数据集及评估指标

为了验证本文所提模型的高效性，本文选取四个公开数据集作为实验数据，具体如表 1：

Table 1. Information of datasets
表 1. 数据集的信息

数据集	用户数量	条目数量	评分数量
MovieLens 1M	6040	3706	1,000,209
Film Trust	1508	2071	35,497
Epinions (Extended)	120,492	755,760	13,668,320
Book-Crossing	92,107	271,379	1,031,175

为了客观公正的对模型进行评估，本文使用 5 折交叉验证用于训练和测试数据集，其中 80%的数据用来做训练集，20%的数据用来做测试集，最终结果取 20 次实验后的平均值。

本文使用平均绝对误差(Mean Absolute Error, MAE)和均方根误差(Root Mean Squared Error, RMSE)这两个指标来衡量模型的质量。具体公式如下：

$$\mathrm{MAE}=\frac{\sum_{i,j}\left|\hat{r}_{i,j}-r_{i,j}\right|}{|\Omega|} \tag{14}$$

$$\mathrm{RMSE}=\sqrt{\frac{\sum_{i,j}\left|\hat{r}_{i,j}-r_{i,j}\right|^{2}}{|\Omega|}} \tag{15}$$

其中，$|\Omega|$ 是测试数据集中的评分个数，$r_{i,j}$ 是测试数据集中的评分，$\hat{r}_{i,j}$ 是预测的评分。

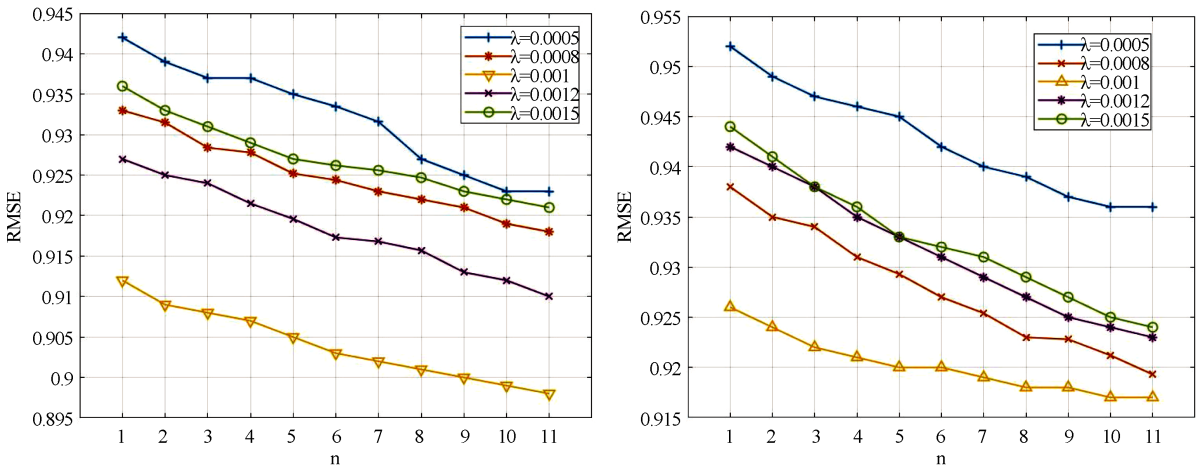
4.2. 模型的参数实验

为了综合评估本文所提模型的性能，需对相关参数进行灵敏度实验，为简便后续操作，预设 $\lambda_Z=\lambda_W=\lambda_T$，$\lambda_U=\lambda_V=\lambda$。其中正则化系数 λ、λ_T，潜在因子维数 k 和神经网络层数 n 这四个参数对实验结果的影响较大。本文利用控制变量法对这三个参数分别进行实验，具体训练过程如图 2~5 所示。

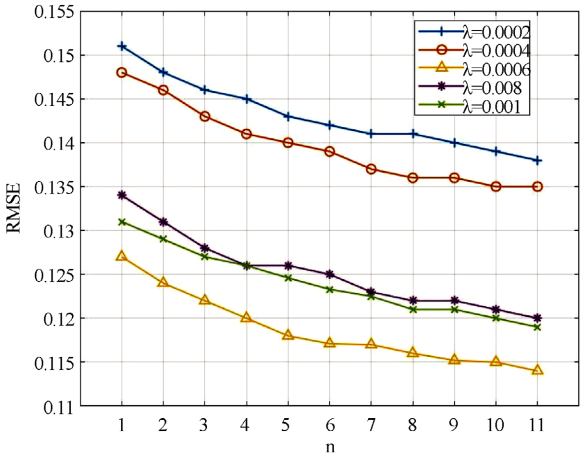
图 2 是针对正则化系数 λ 的灵敏度实验，其中潜在因子维数 k 预设为 30，图 2(a)~(d)分别展示了在数据集 D1~D4 上的实验结果。可以看出，当潜在因子维数 k 固定时，正则化系数 λ 在不同的数据集上最优取值不同，且变化的神经网络层数 n 对正则化系数 λ 值有一定的影响。通过实验，本文选取的最优正则化系数 λ 值在数据集 D1~D4 上分别为 0.001，0.001，0.0006 和 0.0002。正则化系数 λ_T 的实验与 λ 类似，在数据集 D1~D4 上的最终取值为 0.047，0.062，0.013 和 0.02。

图 3 是针对潜在因子维数 k 的灵敏度实验，其中正则化系数 λ 和 λ_T 分别预设为 0.001，0.04，图 3(a)~(d)分别展示了在数据集 D1~D4 上的实验结果。同样可以看出，当正则化系数固定时，潜在因子维数 k 的取值随着网络层数 n 的变化而变化。本文进一步考虑时间花销，图 4 展示了不同潜在因子维数 k 的时间花销，不难看出 k 越大导致时间花销越多，综合考虑后选取潜在因子维数 k 在数据集 D1~D4 上分别为 30，20，50，40。

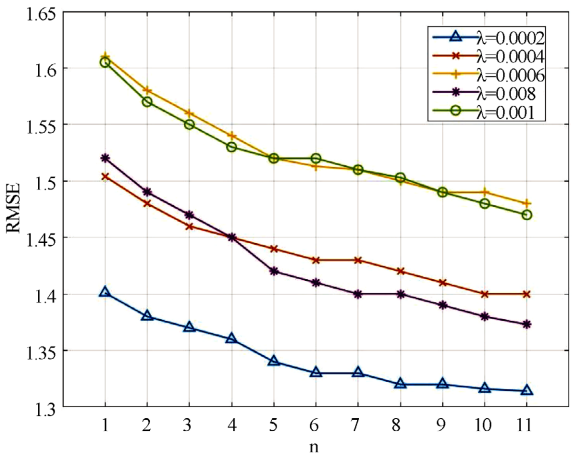
图 5 同样是考虑时间花销的实验，它展示了不同神经网络层数 n 的时间花销，从图中可以看出，时间花销随着 n 的变大而增加。同时考虑预测精度与时间花销后，本文的选取的最佳神经网络层数 n 在 D1~D4 数据上分别为 5，3，5，5。



(a) D1　　(b) D2

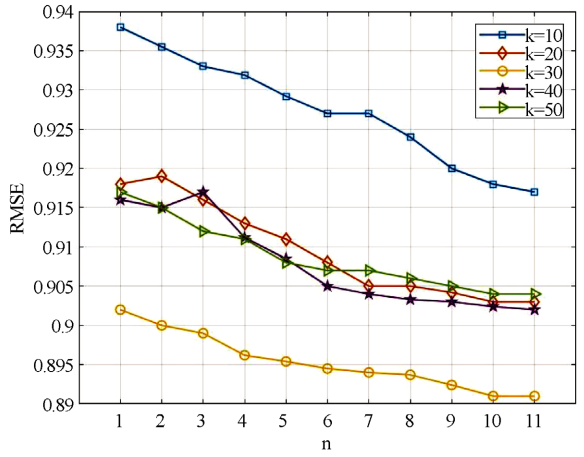


(c) D3

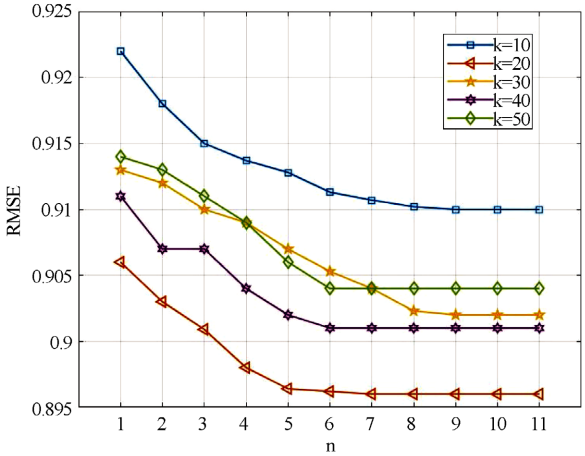


(d) D4

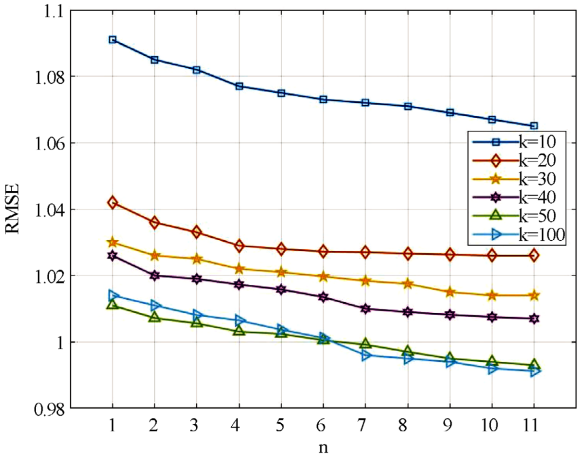
Figure 2. Different values of λ on datasets affect the RMSE
图 2. 数据集上不同 λ 值对 RMSE 的影响



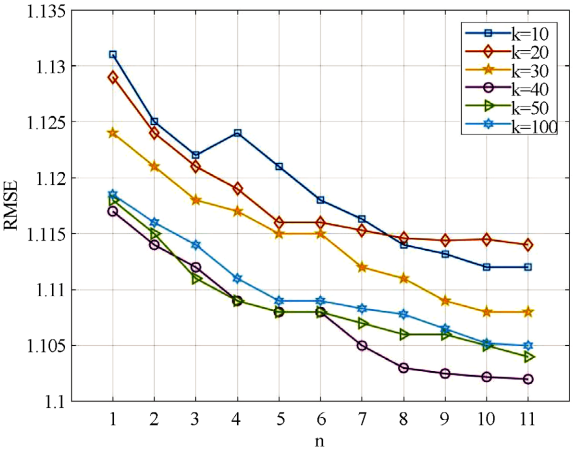
(a) D1



(b) D2



(c) D3



(d) D4

Figure 3. Different values of k on datasets affect the RMSE
图 3. 数据集上不同 k 值对 RMSE 的影响

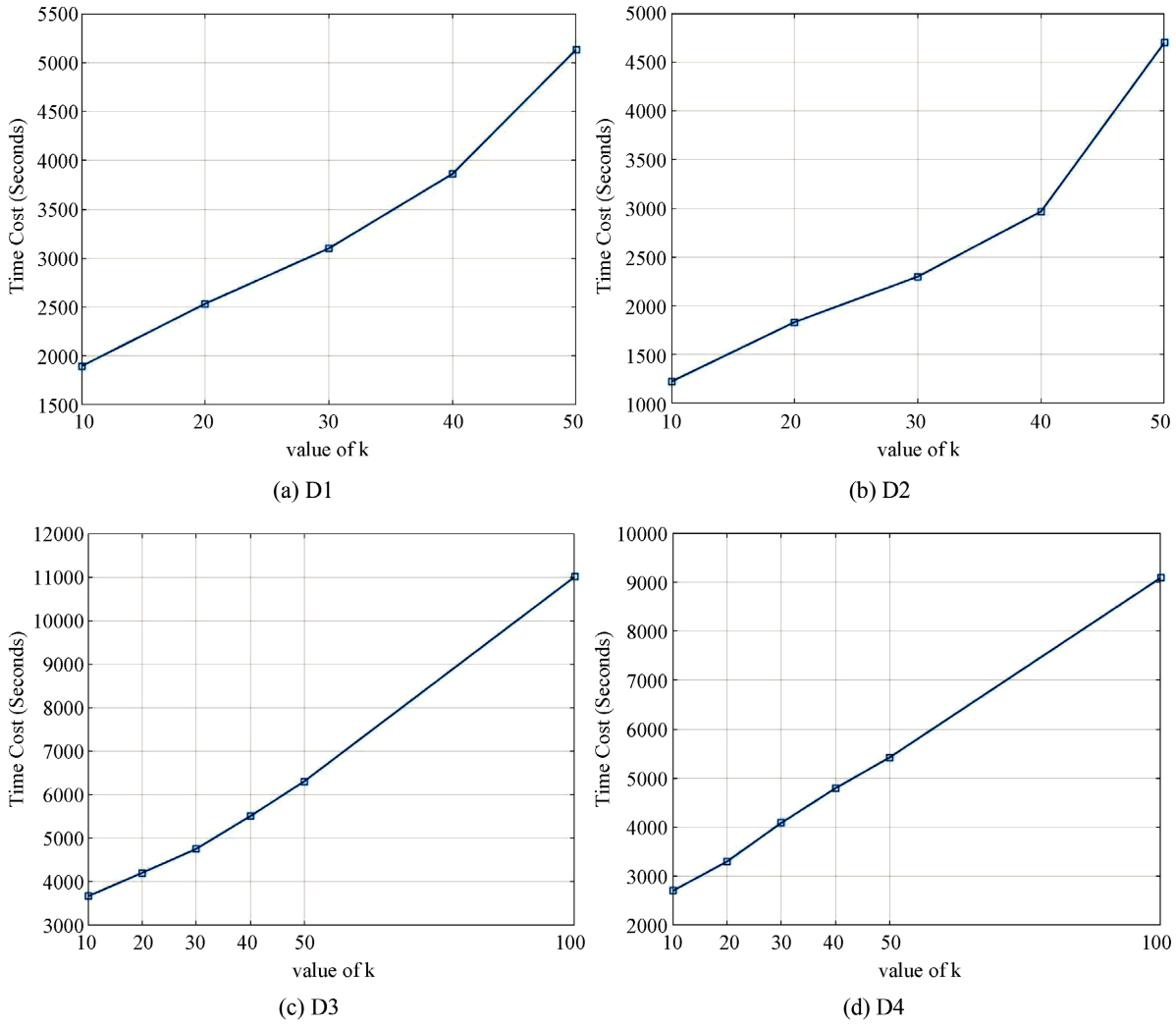
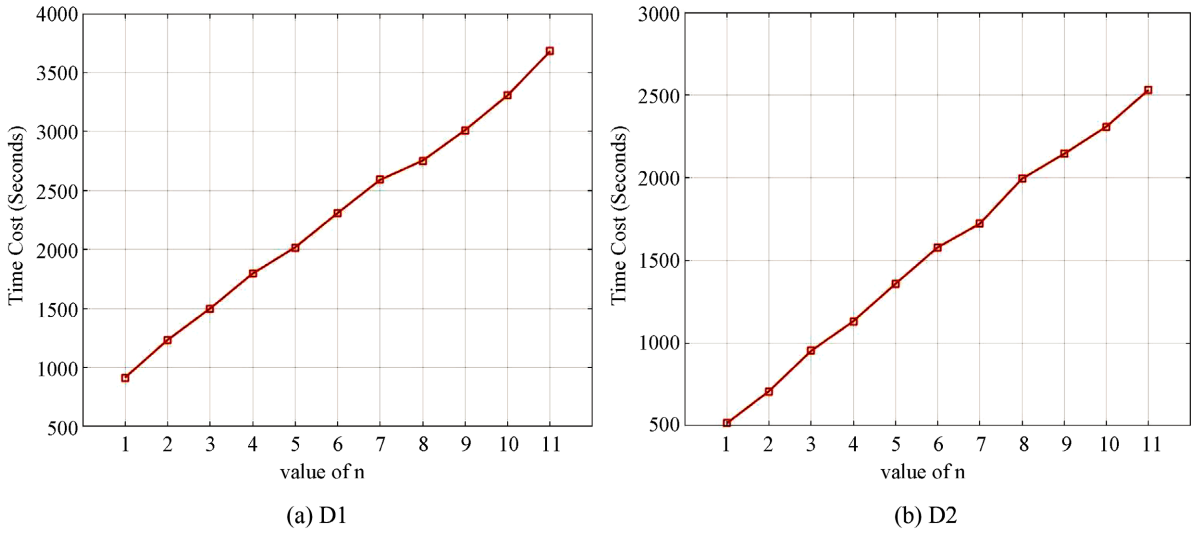
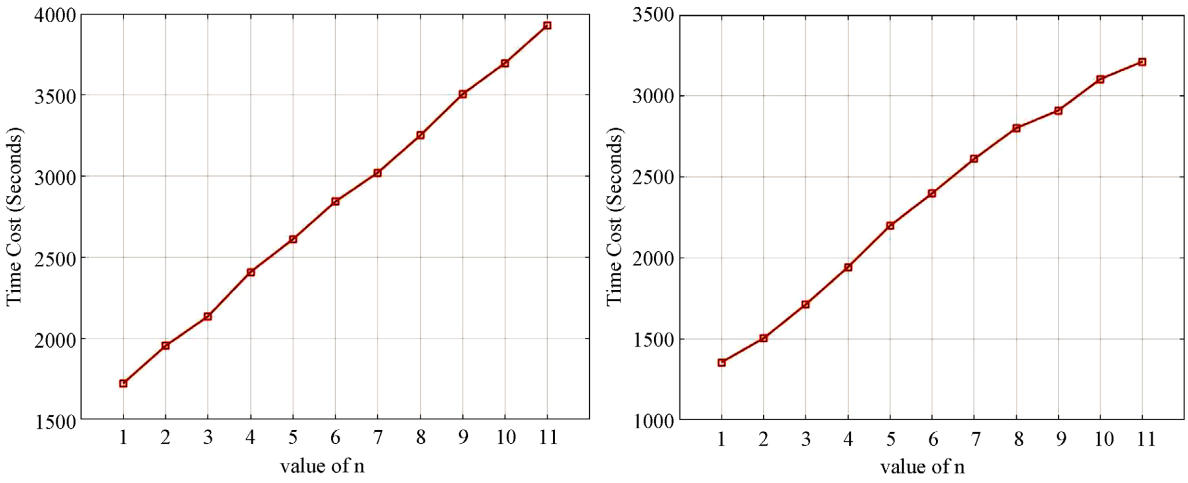


Figure 4. Different values of k on datasets affect the time cost (Seconds)
图 4. 数据集上不同 k 值的时间花销(秒)





(c) D3　　(d) D4

Figure 5. Different values of *n* on datasets affect the time cost (Seconds)
图 5. 数据集上不同 *n* 值的时间花销(秒)

4.3. 对比模型实验

为了判断 NN-MF-TR 模型的有效性，本文选取了两个经典社交推荐模型和两个较新颖的推荐模型，本文涉及的实验模型共有 5 个，具体信息如下表 2：

Table 2. Comparison models
表 2. 对比模型

编号	名称	描述
M1	Sorec [7]	基于矩阵分解对信任关系进行分解的推荐模型
M2	TrustSVD [8]	利用用户信任的隐含影响和显式影响的推荐模型
M3	FDAE [14]	基于一种快速深度自编码器的推荐模型
M4	MLF [17]	基于随机算法的潜在因子模型
M5	NN-MF-TR	融合用户信任关系和神经网络的推荐模型

基于上文参数的确定，将本文模型与所选四个对比模型在四个数据集上进行实验，实验结果如下表所示。其中图 6 是模型 M1~M5 在数据集 D1~D4 上关于 MAE 和 RMSE 这两个指标的表现，图 7 是模型 M1~M5 在数据集 D1~D4 上的关于最小 RMSE 值的时间花销。

从图 6 可以看出，模型 M5 与 M1、M3、M4 相比 RMSE 和 MAE 有明显的优势。基于 MAE 值来说，模型 M5 至少提升了 3.51%。基于 RMSE 值来说，模型 M5 至少提升了 0.93%。这说明考虑用户的信任与被信任关系挖掘了用户的潜在偏好，缓解了矩阵稀疏的问题，可以获得较好的推荐结果。

从图 7 可以看出，模型 M5 的时间花销远少于模型 M1~M3，尤其在大型数据集上，如模型 M3 在数据集 D3 上取得的最小 RMSE 的时间花销是 29,537 s，而模型 M5 在数据集 D3 上取得的最小 RMSE 的时间花销是 3485 s，这说明在训练模型时引入神经网络实现了更高的计算效率。

模型 M5 与模型 M2 相比较 MAE 和 RMSE 值比较相近，但是模型 M5 的时间花销少于模型 M2。模型 M4 的时间花销小于模型 M5，但是模型 M5 预测评分的精度却高于模型 M4。在综合考察衡量推荐准确率和时间效率后，模型 M5 被证明可以在保证预测评分精度的同时减少时间花销，所以模型 M5 的整体表现优于其余四个对比模型。

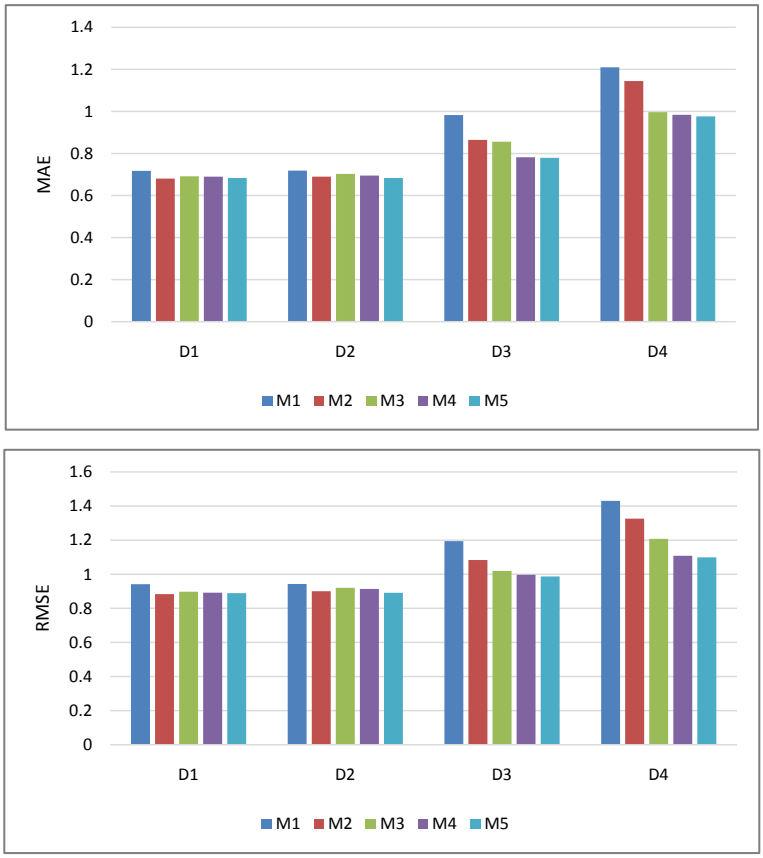


Figure 6. Lowest MAE and RMSE of models M1~M5 on datasets D1~D4
图 6. 模型 M1~M5 在数据集 D1~D4 上的最小 MAE 和 RMSE

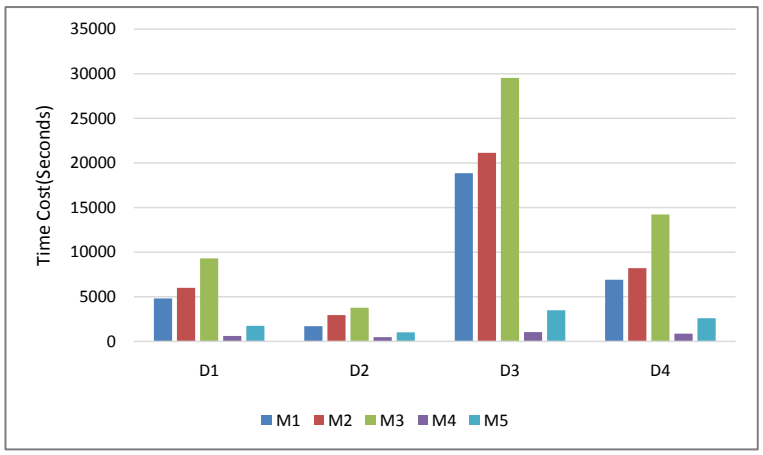


Figure 7. Time cost by the models M1~M5 on the datasets D1~D4 regarding the lowest RMSE (Seconds)
图 7. 模型 M1~M5 在数据集 D1~D4 上关于最小 RMSE 的时间花销(秒)

5. 结论

如何提高推荐准确率和时间效率已经逐渐成为推荐系统中的研究热点。本文综合考察衡量准确率和效率这两个方面。一方面，利用用户潜在的信任与被信任信息来挖掘出用户的潜在偏好，在一定程度上

缓解了评分矩阵稀疏所导致的推荐准确率低等问题。另一方面，在训练模型时引入神经网络，其训练过程大大提高了其在稀疏数据上的表示学习能力，能够在保证推荐准确率的前提下减少训练模型的时间花销。最后，本文在四个真实世界数据集进行实验，NN-MF-TR 模型与对比模型相比较综合性能最优。通过实验结果可证明引入神经网络对模型进行训练且利用用户间潜在的信任与被信任关系作为推荐模型的辅助信息可提高推荐系统的性能，在提高模型准确率的同时也提升了训练速度。

基金项目

本研究获得国家自然科学基金面上项目资助，项目编号为 61973219。

参考文献

[1] 刘君良, 李晓光. 个性化推荐系统技术进展[J]. 计算机科学, 2020, 47(7): 47-55.

[2] 于蒙, 何文涛, 周绪川, 等. 推荐系统综述[J]. 计算机应用, 2022, 42(6): 1898-1913.

[3] Cao, B., Zhang, Y.T., Zhao, Z.W., Liu, X., Skonieczny, Ł. and Lv, Z. (2022) Recommendation Based on Large-Scale Many-Objective Optimization for the Intelligent Internet of Things System. *IEEE Internet of Things Journal*, **9**, 15030-15038. https://doi.org/10.1109/JIOT.2021.3104661

[4] Liu, H., Zheng, C., Li, D.C., *et al*. (2022) EDMF: Efficient Deep Matrix Factorization with Review Feature Learning for Industrial Recommender System. *IEEE Transactions on Industrial Informatics*, **18**, 4361-4371. https://doi.org/10.1109/TII.2021.3128240

[5] 吴正洋, 汤庸, 刘海. 个性化学习推荐研究综述[J]. 计算机科学与探索, 2022, 16(1): 21-40.

[6] 刘华锋, 景丽萍, 于剑. 融合社交信息的矩阵分解推荐方法研究综述[J]. 软件学报, 2018, 29(2): 340-362.

[7] Ma, H., Yang, H., Lyu, M.R. and King, I. (2008) SoRec: Social Recommendation Using Probabilistic Matrix Factorization. *Proceedings of the* 17*th ACM Conference on Information and Knowledge Management*, Napa Valley, 26-30 October 2008, 931-940. https://doi.org/10.1145/1458082.1458205

[8] Guo, G., Zhang, J. and Yorke-Smith, N. (2015) TrustSVD: Collaborative Filtering with Both the Explicit and Implicit Influence of User Trust and of Item Ratings. *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence*, Palo Alto, 22 February-1 March 2022, 123-129. https://doi.org/10.1609/aaai.v29i1.9153

[9] Luo, X., Zhou, M.C., Xia, Y.N. and Zhu, Q.S. (2014) An Efficient Non-Negative Matrix-Factorization-Based Approach to Collaborative Filtering for Recommender Systems. *IEEE Transactions on Industrial Informatics*, **10**, 1273-1284. https://doi.org/10.1109/TII.2014.2308433

[10] Luo, X., Zhou, M.C., Li, S., You, Z., Xia, Y. and Zhu, Q. (2016) A Nonnegative Latent Factor Model for Large-Scale sparse MATRICES in Recommender Systems via Alternating Direction Method. *IEEE Transactions on Neural Networks and Learning Systems*, **27**, 579-592. https://doi.org/10.1109/TNNLS.2015.2415257

[11] Song, Y., Li, M., Luo, X., Yang, G. and Wang, C. (2020) Improved Symmetric and Non-Negative Matrix Factorization models for Undirected Sparse and Large-Scaled Networks: A Trip Factorization-Based Approach. *IEEE Transactions on Industrial Informatics*, **16**, 3006-3017. https://doi.org/10.1109/TII.2019.2908958

[12] Luo, X., Liu, Z.G., Li, S., Shang, M. and Wang, Z. (2021) A Fast Non-Negative Latent Factor Model Based on Generalized Momentum Method. *IEEE Transactions on Systems*, *Man and Cybernetics*: *Systems*, **51**, 610-620. https://doi.org/10.1109/TSMC.2018.2875452

[13] Zhang, G.J., Liu, Y. and Jin, X.N. (2019) A Survey of Autoencoder-Based Recommender Systems. *Frontiers of Computer Science*, **14**, 430-450. https://doi.org/10.1007/s11704-018-8052-6

[14] Jiang, J.J., Li, W.L., Dong, A.N., Gou, Q.H. and Luo, X. (2020) A Fast Deep AutoEncoder for High-Dimensional and Sparse Matrices in Recommender Systems. *Neurocomputing*, **412**, 381-391. https://doi.org/10.1016/j.neucom.2020.06.109

[15] Luo, X., Liu, Z., Shang, M.S., *et al*. (2021) Highly-Accurate Community Detection via Pointwise Mutual Information-Incorporated Symmetric Mon-Negative Matrix Factorization. *IEEE Transactions on Network Science and Engineering*, **8**, 463-476. https://doi.org/10.1109/TNSE.2020.3040407

[16] 钱冲, 常冬霞. 图拉普拉斯正则化稀疏变换学习图像去噪算法[J]. 计算机工程与应用, 2022, 58(5): 232-239.

[17] Yuan, Y., He, Q., Luo, X. and Shang, M.S. (2022) A Multilayered-and-Randomized Latent Factor Model for High-Dimensional and Sparse Matrices. *IEEE Transactions on Big Data*, **8**, 784-794. https://doi.org/10.1109/TBDATA.2020.2988778

[18] Guliyev, N.J. and Ismailov, V.E. (2015) A Single Hidden Layer Feedforward Network with Only One Neuron in the Hidden Layer Can Approximate Any Univariate Function. *Neural Computation*, **28**, 1289-1304. https://doi.org/10.1162/NECO_a_00849

[19] Bacciu, D., Colombo, M., Morelli, D. and Plans, D. (2018) Randomized Neural Networks for Preference Learning with Physiological Data. *Neurocomputing*, **298**, 9-20. https://doi.org/10.1016/j.neucom.2017.11.070

[20] Yi, B.L., Shen, X.X., Liu, H., *et al*. (2019) Deep Matrix Factorization with Implicit Feedback Embedding for Recommendation System. *IEEE Transactions on Industrial Informatics*, **15**, 4591-4601. https://doi.org/10.1109/TII.2019.2893714