

文章编号: 2095-2163(2022)06-0020-08

中图分类号: TP391

文献标志码: A

# 融合迁移学习和神经网络的潜在因子模型

吴静<sup>1</sup>, 宋燕<sup>2</sup>

(1 上海理工大学理学院, 上海 200093; 2 上海理工大学光电信息与计算机工程学院, 上海 200093)

**摘要:** 潜在因子(Latent Factor, LF)模型是解决大规模稀疏数据缺失信息填补的有效方法之一。但是,传统的LF模型及其大部分变种考虑的目标数据集形式单一,且模型训练迭代次数多,收敛速度慢,很难满足快速响应的实时性需求。针对这些问题,本文综合考虑了模型预测精度和训练花费时间这两个方面,创新地提出一种融合迁移学习(Transfer Learning, TF)和神经网络(Neural Network, NN)的潜在因子(TL-NNLF)模型。一方面,通过迁移不同评分模式间的共享潜在信息来充分挖掘已知信息之间的关系,更好地进行特征的表征学习;另一方面,引入神经网络来减缓由于引入额外的评分领域所造成的模型训练负担,在保证满意的预测精度的同时提高模型的训练速度。最后,在真实数据集上的实验结果表明,和其他经典LF模型相比,本文提出的TL-NNLF模型能够在预测精度有所保证的前提下大大减少了模型训练时间。

**关键词:** 潜在因子模型; 矩阵分解; 大规模稀疏数据; 迁移学习; 神经网络

## Latent Factor model based on transfer learning and neural networks

WU Jing<sup>1</sup>, SONG Yan<sup>2</sup>

(1 College of Science, University of Shanghai for Science and Technology, Shanghai 200093, China; 2 School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China)

**[Abstract]** Latent Factor (LF) model is one of the effective methods to solve the problem of filling missing information in large-scale sparse data. However, the traditional LF model and most of its variants only consider a single form of target data set, and the model has many training iterations and slow convergence speed, which makes it difficult to meet the real-time requirements of rapid response. To solve these problems, this paper innovatively proposes a so-called transfer learning incorporated and neural network LF (TL-NNLF) model that integrates transfer learning and neural networks, which comprehensively considers the prediction accuracy and the training time of the model. On the one hand, the relationship between known information can be fully explored by transferring the shared potential information under different rating patterns, so as to carry out better learning of feature representation. On the other hand, neural network is introduced to alleviate the model training burden caused by the introduction of additional rating domain, which can improve the model training speed while ensuring satisfactory prediction accuracy. Finally, experimental results on real data sets show that compared with other classical LF models, the TL-NNLF model proposed in this paper can greatly reduce the model training time on the premise of guaranteeing the prediction accuracy.

**[Key words]** Latent Factor model; matrix decomposition; large-scale sparse data; transfer learning; neural networks

## 0 引言

随着信息时代的迅猛发展,数据呈现爆炸式增长且常以高维矩阵的形式被存储<sup>[1-2]</sup>。然而,由于天然限制、技术不足等原因使得高维稀疏矩阵普遍存在<sup>[3]</sup>。并且当面临这样的大规模数据缺失的场景时,传统潜在因子(Latent Factor, LF)模型的性能往往不尽如人意。

实际上,现实应用中总是可以使用多种评分模式来描述两两实体之间的关系<sup>[4-6]</sup>,并且不同评分模式之间存在一定的关联性,即实体之间的关系并

不会因为评分模式的改变而改变。也就是说,当某位用户需要对商品进行评价时,如果该用户在二分制评分模式下点击了喜欢按钮,那么在五分制评分模式中有很大概率会给出3~5分这样的高分。因此,基于不同评分模式之间的关联性,可以通过迁移学习将辅助领域中的潜在信息迁移到目标领域,以此缓解其中的数据稀疏问题<sup>[7]</sup>。

此外,时间效率逐渐成为评判模型性能好坏的原因之一。现有的学习方法,如随机梯度下降法等,往往需要多次迭代达到模型收敛,从而造成模型训练效率低<sup>[8-10]</sup>。

**基金项目:** 国家自然科学基金(62073223); 中央军委装备发展部航天飞行动力学技术国防科技重点实验室资助(6142210200304)。

**作者简介:** 吴静(1998-),女,硕士研究生,主要研究方向:推荐系统、数据挖掘; 宋燕(1979-),女,博士,教授,博士生导师,CCF高级会员(No.93073SM),主要研究方向:大数据算法、图像处理、预测控制。

**通讯作者:** 宋燕 Email: songya@usst.edu.cn

收稿日期: 2021-12-09

综上,本文提出的 TN-NNLF 模型主要贡献如下:

- (1) 基于传统的 LF 模型,建立了一种新的融合迁移学习和神经网络的潜在因子模型。
- (2) 基于不同评分模式之间的关联性,通过迁移学习来迁移不同评分模式之间的共享潜在信息,以便充分挖掘已知信息之间的关系。
- (3) 利用神经网络进行模型参数训练,减缓由于额外的评分领域的引入所造成的模型训练负担。
- (4) 在 2 个真实工业应用数据集上的实验结果表明,本文提出的 TL-NNLF 模型能够在保证满意的预测精度的前提下大大加快了模型训练速度。

### 1 相关工作

#### 1.1 迁移学习

与经典的机器学习方法不同的是,迁移学习主要基于不同领域之间所存在的潜在联系,能够通过从相关的辅助领域中迁移共享的知识,以此来改进模型在目标领域中的学习效果<sup>[11]</sup>。

具体地,基于迁移内容的不同,可以将迁移学习分为以下 3 种类型:

- (1) 基于实例的迁移,即从辅助领域中挑选出有助于目标领域学习的实例。
- (2) 基于特征的迁移学习,即提取并迁移辅助领域和目标领域之间所共享的特征表示。
- (3) 基于模型的迁移学习,又称为基于共享参数的迁移,通过找到辅助领域与目标领域的空间模型的共享参数以达到迁移的目的<sup>[12-13]</sup>。

### 1.2 自编码神经网络

一个简单的自编码神经网络模型主要包含 3 层结构:输入层、隐藏层、输出层<sup>[14-16]</sup>。主要工作原理是:在隐藏层将输入数据映射到维度更低的特征空间中以提取重要潜在信息,再对特征空间中所包含的信息进行重构并作为输出层数据。

一般情况下,通常使输入层和输出层样本数据维度相同,从而可以更好地重构原始数据。这里假设输入层包含  $m$  个样本  $x_1, x_2, \dots, x_m$ , 编码层包含  $f$  个神经元,对应的输出层数据为  $y_1, y_2, \dots, y_m$ 。研究中可利用欧式距离建立重构误差损失函数,得到具体形式如下:

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

其中,  $m$  表示输入层样本个数。

## 2 模型研究设计

### 2.1 TL-NNLF 模型

本文提出的模型考虑的场景是:基于五分制和二分制这 2 种不同的评分模式来描述相同项目集合中两两实体之间的关系。并且将五分制评分模式领域作为目标领域,二分制评分模式领域作为辅助领域。

一般地,若定义目标矩阵  $X = [x_{i,j}]^{I \times |J|} \in \{1, 2, 3, 4, 5, ?\}$ , 辅助矩阵  $Y = [y_{i,j}]^{I' \times |J|} \in \{0, 1, ?\}$ , 已知值集合  $\Delta$ , 其中  $I$  和  $J$  分别表示不同模式评分矩阵所包含的项目集合,“?”表示缺失值。具体的模型结构如图 1 所示。

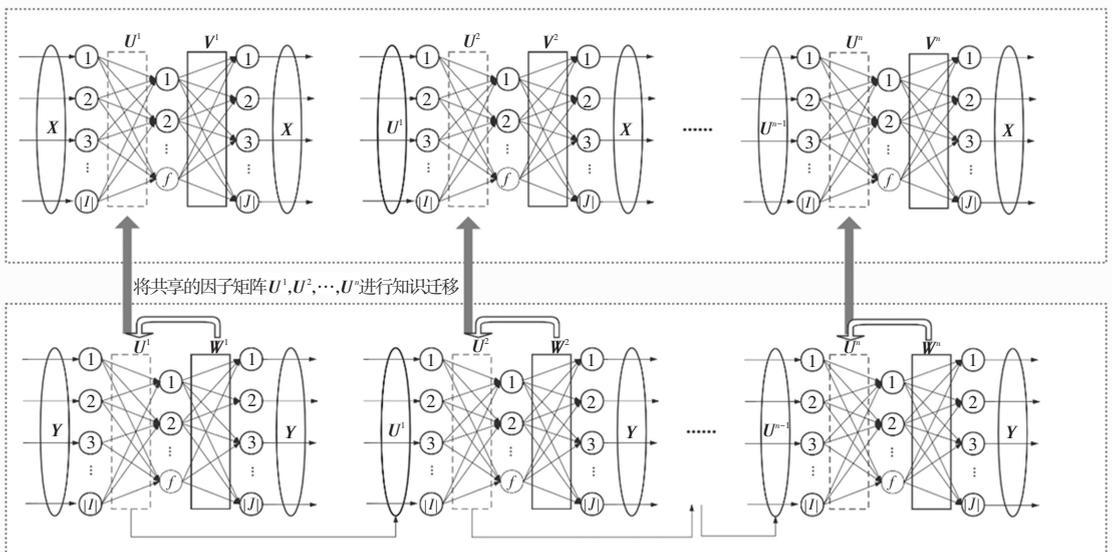


图 1 TL-NNLF 模型结构图

Fig. 1 Structure of an TL-NNLF model

首先进行目标函数的构造,为了避免模型训练过程中出现过拟合的情况而引入正则化项<sup>[8-10]</sup>,得到具体形式为:

$$J_1(\mathbf{U}, \mathbf{V}) = \sum_{(i,j) \in \Delta} ((x_{i,j} - \mathbf{u}_{i,\cdot} \mathbf{v}_{j,\cdot})^2 + \lambda_U \|\mathbf{u}_{i,\cdot}\|_F^2 + \lambda_V \|\mathbf{v}_{j,\cdot}\|_F^2) \quad (2)$$

$$J_2(\mathbf{U}, \mathbf{W}) = \sum_{(i,j) \in \Delta} ((y_{i,j} - \mathbf{u}_{i,\cdot} \mathbf{w}_{j,\cdot})^2 + \lambda_U \|\mathbf{u}_{i,\cdot}\|_F^2 + \lambda_W \|\mathbf{w}_{j,\cdot}\|_F^2) \quad (3)$$

其中,  $x_{i,j}, y_{i,j}$  分别表示目标矩阵  $\mathbf{X}$ , 辅助矩阵  $\mathbf{Y}$  中的元素;  $\mathbf{u}_{i,\cdot}$  表示共享因子矩阵  $\mathbf{U}$  的第  $i$  行元素;  $\mathbf{v}_{j,\cdot}, \mathbf{w}_{j,\cdot}$  分别表示第  $n$  层神经网络中因子矩阵  $\mathbf{V}, \mathbf{W}$  的第  $j$  行元素;  $\lambda_U, \lambda_V, \lambda_W$  分别表示正则化系数。

## 2.2 基于神经网络进行参数训练

根据图1,若假设 TL-NNLF 模型中包含  $n$  层神经网络结构,则可得到模型训练过程如下:

(1) 当  $n = 1$ 、即模型为单层神经网络结构时。首先训练辅助矩阵  $\mathbf{Y}$ , 将矩阵  $\mathbf{Y}$  的每一行作为输入数据,并依次训练因子矩阵  $\mathbf{U}^1$  和  $\mathbf{W}^1$ 。其次将因子矩阵  $\mathbf{U}^1$  迁移到目标矩阵  $\mathbf{X}$  的训练过程中,此时将矩阵  $\mathbf{X}$  的每一行作为输入数据,并训练因子矩阵  $\mathbf{V}^1$ 。

(2) 当  $n \geq 2$ 、即模型包含多层神经网络结构时。其第一层训练过程和单层神经网络模型类似;而从第二层开始,则是将上一层训练好的因子矩阵  $\mathbf{U}^{n-1}$  作为下一层的输入,这也有利于潜在信息的进一步提取。最后依次训练因子矩阵  $\mathbf{U}^n, \mathbf{W}^n$ 。

接下来,基于梯度法来求解对应的因子矩阵  $\mathbf{U}^n, \mathbf{V}^n, \mathbf{W}^n$ 。若仅仅考虑单层神经网络结构、即当  $n = 1$  时的情况,研究可得到的具体求解步骤分述如下。

(1) 第一步,求解因子矩阵  $\mathbf{U}^1$ 。将辅助矩阵  $\mathbf{Y}$  中的已知值集合作为输入,且通过权重矩阵  $\mathbf{A}^{f \times |I|}$  和偏置向量  $\mathbf{b}^{1 \times f}$  对输入层数据进行编码,得到初始化后的因子矩阵  $\mathbf{U}^1 \mathbf{X}$ , 即:

$$\mathbf{U}^1 = \begin{bmatrix} \mathbf{u}_{1,\cdot}^1 \\ \vdots \\ \mathbf{u}_{|I|,\cdot}^1 \end{bmatrix} = \begin{bmatrix} h(\mathbf{a}_{1,\cdot} \mathbf{y}_{1,\cdot}^T + b_1) & \cdots & h(\mathbf{a}_{f,\cdot} \mathbf{y}_{1,\cdot}^T + b_f) \\ \vdots & \ddots & \vdots \\ h(\mathbf{a}_{1,\cdot} \mathbf{y}_{|I|,\cdot}^T + b_1) & \cdots & h(\mathbf{a}_{f,\cdot} \mathbf{y}_{|I|,\cdot}^T + b_f) \end{bmatrix} \quad (4)$$

其中,  $\mathbf{a}_{1,\cdot}, \dots, \mathbf{a}_{f,\cdot}$  分别表示权重矩阵  $\mathbf{A}^{f \times |I|}$  的每一行元素,  $b_1, \dots, b_f$  则分别表示偏置向量  $\mathbf{b}^{1 \times f}$  中的每一个元素。

通常情况下需要使用反向传播算法训练权重矩阵  $\mathbf{A}$  和偏置向量  $\mathbf{b}^{[17]}$ , 然而该算法容易加重模型训练负担。所以为了加快模型训练速度,本文使用随机学习算法来确定权重矩阵以及偏置向量<sup>[18-20]</sup>。

(2) 第二步,求解因子矩阵  $\mathbf{W}^1$ 。根据初始化后的因子矩阵  $\mathbf{U}^1$ , 并利用梯度法求解目标函数。研究中首先求偏导,可以得到:

$$\frac{\partial J_2}{\partial \mathbf{w}_{j,\cdot}^1} = -2 \sum_{i \in I(j)} (\mathbf{u}_{i,\cdot}^1)^T (\mathbf{y}_{i,j} - \mathbf{u}_{i,\cdot}^1 \mathbf{w}_{j,\cdot}^1) + 2 |\Delta(j)| \lambda_W \mathbf{w}_{j,\cdot}^1 \quad (5)$$

其中,  $I(j)$  表示矩阵每行的已知元素集合,  $|\Delta(j)|$  表示辅助矩阵  $\mathbf{Y}$  中第  $j$  列的已知值集合中的元素个数。

接着,令  $\partial J_2 / \partial \mathbf{w}_{j,\cdot}^1 = 0$  有:

$$\left( \sum_{i \in I(j)} (\mathbf{u}_{i,\cdot}^1)^T \mathbf{u}_{i,\cdot}^1 + |\Delta(j)| \lambda_W \mathbf{E} \right) \mathbf{w}_{j,\cdot}^1 = \sum_{i \in I(j)} (\mathbf{u}_{i,\cdot}^1)^T \mathbf{y}_{i,j} \quad (6)$$

其中,  $\mathbf{E}$  表示一个维度为  $f \times f$  的单位矩阵;且  $\sum_{i \in I(j)} (\mathbf{u}_{i,\cdot}^1)^T \mathbf{u}_{i,\cdot}^1$  是一个维度为  $f \times f$  的完全矩阵。

最终可以得到因子矩阵  $\mathbf{W}^1$  中每一行元素的更新公式为:

$$\mathbf{w}_{j,\cdot}^1 = \left( \sum_{i \in I(j)} (\mathbf{u}_{i,\cdot}^1)^T \mathbf{u}_{i,\cdot}^1 + |\Delta(j)| \lambda_W \mathbf{E} \right)^{-1} \left( \sum_{i \in I(j)} (\mathbf{u}_{i,\cdot}^1)^T \mathbf{y}_{i,j} \right) \quad (7)$$

(3) 第三步,更新因子矩阵  $\mathbf{U}^1$ 。为了使其更好地近似辅助矩阵  $\mathbf{Y}$ , 需要进一步更新因子矩阵  $\mathbf{U}^1$ 。

类似地,利用梯度法,首先求偏导得到:

$$\frac{\partial J_2}{\partial \mathbf{u}_{i,\cdot}^1} = -2 \sum_{j \in J(i)} (\mathbf{y}_{i,j} - \mathbf{u}_{i,\cdot}^1 \mathbf{w}_{j,\cdot}^1) (\mathbf{w}_{j,\cdot}^1)^T + 2 |\Delta(j)| \lambda_U \mathbf{u}_{i,\cdot}^1 \quad (8)$$

其中,  $J(i)$  表示矩阵每一列的已知元素集合。

并令  $\partial J_2 / \partial \mathbf{u}_{i,\cdot}^1 = 0$ , 得到其最终参数更新形式为:

$$\mathbf{u}_{i,\cdot}^1 = \left( \sum_{j \in J(i)} \mathbf{y}_{i,j} (\mathbf{w}_{j,\cdot}^1)^T \right) \left( \sum_{j \in J(i)} \mathbf{w}_{j,\cdot}^1 (\mathbf{w}_{j,\cdot}^1)^T + |\Delta(j)| \lambda_U \mathbf{E} \right)^{-1} \quad (9)$$

其中,  $\sum_{j \in J(i)} \mathbf{w}_{j,\cdot}^1 (\mathbf{w}_{j,\cdot}^1)^T$  是一个维度为  $f \times f$  的完全矩阵。

(4) 第四步,求解因子矩阵  $\mathbf{V}^1$ 。此时将目标矩阵  $\mathbf{X}$  作为输入,并借助更新后的因子矩阵  $\mathbf{U}^1$  进行参数训练。类似地,可以得到因子矩阵  $\mathbf{V}^1$  的具体参数更新形式如下:

$$\mathbf{v}_{j,\cdot}^1 = \left( \sum_{i \in I(j)} (\mathbf{u}_{i,\cdot}^1)^T \mathbf{u}_{i,\cdot}^1 + |\Delta(j)| \lambda_V \mathbf{E} \right)^{-1} \cdot$$

$$\left( \sum_{i \in I(j)} (\mathbf{u}_{i,\cdot}^1)^\top \mathbf{x}_{i,j} \right) \quad (10)$$

通过上述求解过程,可以得到当  $n = 1$  时的单层 TL-NNLF 模型中各个因子矩阵中元素的具体训练规则。

其次考虑一般情况,也就是当  $n \geq 2$ 、即模型包含多层网络结构时的情况。从图 1 可以看出,和第一层神经网络模型不同的是,此时将上一层的因子矩阵、即  $\mathbf{U}^{n-1}$  中的元素作为下一层输入。也就是说,此时仅仅考虑潜在因子矩阵的信息提取。并且由于潜在因子的矩阵规模远远小于原始矩阵,这不仅减轻了模型负担,同时也有助于进一步挖掘原始矩阵中所包含的潜在信息。

因此,当  $n \geq 2$  时,首先通过随机学习算法初始化权重矩阵和偏置向量对上一步训练完成的因子矩阵  $\mathbf{U}^{n-1}$  进行编码,并得到因子矩阵  $\mathbf{U}^n$  的具体更新规则如下:

$$\mathbf{U}^n = \begin{bmatrix} \mathbf{u}_{1,\cdot}^n \\ \vdots \\ \mathbf{u}_{|I|,\cdot}^n \end{bmatrix} = \begin{bmatrix} h(\mathbf{z}_{1,\cdot}^n, (\mathbf{u}_{1,\cdot}^{n-1})^\top + d_1) & \cdots & h(\mathbf{z}_{f,\cdot}^n, (\mathbf{u}_{1,\cdot}^{n-1})^\top + d_f) \\ \vdots & \ddots & \vdots \\ h(\mathbf{z}_{1,\cdot}^n, (\mathbf{u}_{|I|,\cdot}^{n-1})^\top + d_1) & \cdots & h(\mathbf{z}_{f,\cdot}^n, (\mathbf{u}_{|I|,\cdot}^{n-1})^\top + d_f) \end{bmatrix} \quad (11)$$

其中,  $\mathbf{z}_{1,\cdot}^n, \dots, \mathbf{z}_{f,\cdot}^n$  分别表示第  $n$  层神经网络模型的权重矩阵  $\mathbf{Z}^n$  中的每一行元素,  $d_1, \dots, d_f$  则分别表示偏置向量  $\mathbf{d}^n$  的每一个元素。

其次,类似地,根据梯度法,得到因子矩阵  $\mathbf{W}^n$  的更新公式:

$$\mathbf{w}_{j,\cdot}^n = \left( \sum_{i \in I(j)} (\mathbf{u}_{i,\cdot}^n)^\top \mathbf{u}_{i,\cdot}^n + |\Delta(j)| \lambda_w \mathbf{E} \right)^{-1} \left( \sum_{i \in I(j)} (\mathbf{u}_{i,\cdot}^n)^\top \mathbf{y}_{i,j} \right) \quad (12)$$

并根据求解后的因子矩阵  $\mathbf{W}^n$ , 进一步更新因子矩阵  $\mathbf{U}^n$ , 得到如下更新公式:

$$\mathbf{u}_{i,\cdot}^n = \left( \sum_{j \in J(i)} \mathbf{y}_{i,j} (\mathbf{w}_{j,\cdot}^n)^\top \right) \left( \sum_{j \in J(i)} \mathbf{w}_{j,\cdot}^n (\mathbf{w}_{j,\cdot}^n)^\top + |\Delta(j)| \lambda_u \mathbf{E} \right)^{-1} \quad (13)$$

最后,则通过将对应神经网络层训练好的因子矩阵  $\mathbf{U}^n$  进行知识迁移,得到因子矩阵  $\mathbf{V}^n$  的具体更新公式如下:

$$\mathbf{v}_{j,\cdot}^n = \left( \sum_{i \in I(j)} (\mathbf{u}_{i,\cdot}^n)^\top \mathbf{u}_{i,\cdot}^n + |\Delta(j)| \lambda_v \mathbf{E} \right)^{-1} \left( \sum_{i \in I(j)} (\mathbf{u}_{i,\cdot}^n)^\top \mathbf{x}_{i,j} \right) \quad (14)$$

至此,文中全面论述了利用神经网络进行 TN-

NNLF 模型中参数更新的具体过程。最终可以通过  $\hat{\mathbf{X}}^n = \mathbf{U}^n \mathbf{V}^n$ ,  $\hat{\mathbf{Y}}^n = \mathbf{U}^n \mathbf{W}^n$  得到完整的目标矩阵、辅助矩阵所对应的估计矩阵。

### 2.3 算法设计和复杂度分析

基于以上参数更新规则,研发设计了 TL-NNLF 模型算法,算法的步骤流程可见如下。

#### 算法 TL-NNLF 模型算法

输入: 已知值集合  $\Delta$ ,  $f, n$ , 正则化系数  $\lambda$

输出: 因子矩阵  $\mathbf{U}^n, \mathbf{V}^n$

Step 1: 当  $n = 1$  时根据式(4)求解因子矩阵  $\mathbf{U}^1$ , 当  $n \geq 2$  时根据式(11)求解因子矩阵  $\mathbf{U}^n$ ;

Step 2: 根据式(12)求解因子矩阵  $\mathbf{W}^n$ ;

Step 3: 根据式(13)更新因子矩阵  $\mathbf{U}^n$ ;

Step 4: 根据式(14)求解因子矩阵  $\mathbf{V}^n$ 。

不难发现,该算法主要时间开销来自于元素更新公式(12)~(14)中逆矩阵的计算。具体地,对于每个逆矩阵的求解需要花费的时间复杂度为  $\Theta(f^3)$ , 其中  $f$  表示因子空间的维度。也就是说,时间复杂度往往和  $f$  呈线性关系,  $f$  值越小、时间开销越少。另外,神经网络层数  $n$  的设置也十分重要,这是因为  $n$  越大、表示层数越多,模型训练所需花费的时间也越多。

## 3 实验结果与分析

### 3.1 评估指标

为了有效衡量提出的模型算法的精确度,采用均方根误差(RMSE)和平均绝对误差(MAE)作为实验评估指标<sup>[9,21-24]</sup>,具体公式如下:

$$RMSE = \sqrt{\frac{\sum_{(i,j)} (x_{i,j} - \hat{x}_{i,j})^2}{|T|}} \quad (15)$$

$$MAE = \frac{\sum_{(i,j)} |x_{i,j} - \hat{x}_{i,j}|_{abs}}{|T|} \quad (16)$$

其中,  $T$  表示测试集;  $|T|$  表示测试集中元素个数;  $x_{i,j}$  表示真实值;  $\hat{x}_{i,j}$  表示根据测试集  $T$  训练得到的估计值。并且,  $RMSE$  和  $MAE$  的值越小,表示模型预测精度越高。

### 3.2 实验数据集及预处理

本文选取了 2 个真实工业应用数据集作为实验数据集来验证提出模型的性能。对此数据集拟做阐释分述如下。

(1)  $D_1$  (MovieLens 1 M)<sup>[25]</sup>: 该数据集由 GroupLens 研究组根据 MovieLens 网站收集的电影

评分数据,其中共包含6 040位用户对3 883部电影的评分数据,且评分区间为 $[1,5]$ 。该数据集密度为4.26%。

(2)  $D_2$ (Douban)<sup>[3]</sup>:该数据集来自中国最大的在线图书、电影和音乐供应商-豆瓣。其中包含了129 490位用户对58 541个项目的评分,评分区间为 $[1,5]$ 。该数据集密度仅为0.22%。

考虑到目前没有适合本文这种场景、即用二分值和五分制两种不同的评分模式来描述相同项目集中两两实体之间的关系,因此在实验过程中参考文献[5-7,21]中的方法处理实验相关数据集。具体地,对数据集 $D_1$ ,将目前大规模稀疏的五分制评分矩阵作为目标矩阵 $X$ ,并通过公式(17)得到对应的辅助矩阵 $Y$ :

$$y_{i,j} = \begin{cases} 0 & x_{i,j} \leq 3 \\ 1 & x_{i,j} > 3 \end{cases} \quad (17)$$

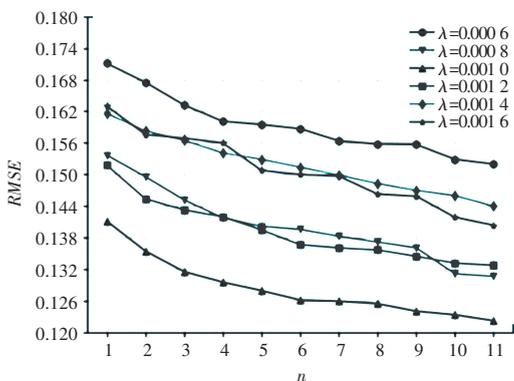
另外为了保证模型训练过程中的客观、公正,需要将实验数据集中所有元素进行归一化;并将所有的实验均采用80%-20%的训练集-测试集实验来设置,进行5折交叉验证;在此基础上重复进行10

次实验,再取10次结果的平均值作为最终结果、且记录下来。以上所有实验都是用Pycharm社区版实现,在Intel Core CPU 3.30 GHz,16 GB内存的计算机上运行。

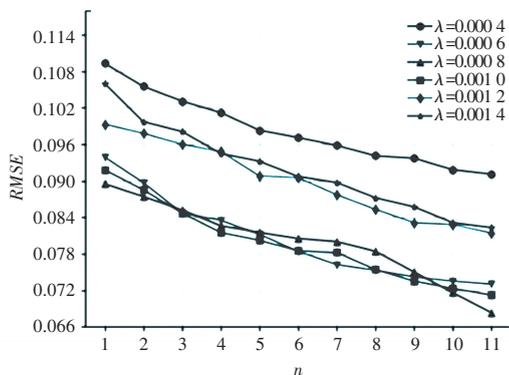
### 3.3 模型参数设置

在进行对比实验之前,需要事先通过参数灵敏度实验。首先将正则化系数 $\lambda_U, \lambda_V, \lambda_W$ 预设为 $\lambda_U = \lambda_V = \lambda_W = \lambda$ 以降低训练成本。因此,可以明确实验中需要训练的参数包括:正则化系数 $\lambda$ 、潜在因子维度 $f$ 、神经网络层数 $n$ 。具体的参数训练结构如图2~图5所示。

图2展示了当潜在因子维度 $f = 40$ 时,不同神经网络层数下正则化系数 $\lambda$ 对模型的影响。可以发现,正则化参数 $\lambda$ 的最优值随数据集的变化而变化,并且正则化参数 $\lambda$ 的最优值还会受到神经网络层数 $n$ 的影响。具体地从图2(a)中可以发现,在数据集 $D_1$ 中,当正则化参数 $\lambda$ 的值为0.001,此时若神经网络层数 $n = 1$ 时有 $RMSE$ 最小值为0.141 2,而当 $n = 5$ 和 $n = 11$ 时, $RMSE$ 最小值分别为0.128 0和0.122 3。



(a)  $D_1$



(b)  $D_2$

图2 正则化系数 $\lambda$ 对 $RMSE$ 的影响

Fig. 2 Influence of regularization coefficient  $\lambda$  on  $RMSE$

图3展示了当正则化系数 $\lambda = 0.001$ 时,不同神经网络层数下潜在因子维数 $f$ 对模型的影响。类似地可以发现,潜在因子维数 $f$ 会随着神经网络层数 $n$ 的变化而变化。具体地从图3(a)中可以发现,在数据集 $D_1$ 中,若潜在因子维数 $f = 40$ ,此时当神经网络层数 $n = 1$ 时有 $RMSE$ 最小值为0.126 2,而当 $n = 5$ 和 $n = 11$ 时, $RMSE$ 最小值则分别为0.125 3和0.105 4。

另外,由于时间效率也是判断模型性能好坏的关键因素,因此还记录了不同潜在因子维数 $f$ 和神

神经网络层数 $n$ 下的时间花销,具体结果如图4、图5所示。

从图4(a)、图4(b)中发现,在数据集 $D_1$ 上,当潜在因子维数 $f = 10$ 时、模型训练时间仅为1 634 min,而当 $f = 160$ 时、模型训练过程则需要花费37 250 min。尽管潜在因子维数越大,模型的特征表示能力越强,但同时也大大增加了实验负担。类似地,从图5(a)、图5(b)中可以发现,无论是在数据集 $D_1$ 、还是 $D_2$ 中,模型训练时间都会随着神经网络层数 $n$ 的增加而增加。

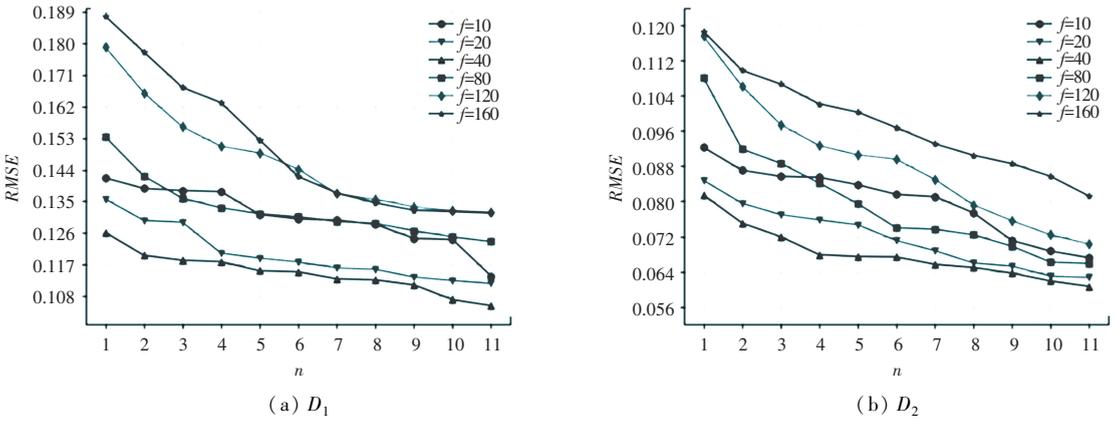


图 3 潜在因子维数  $f$  对 RMSE 的影响

Fig. 3 Influence of Latent Factor dimension  $f$  on RMSE

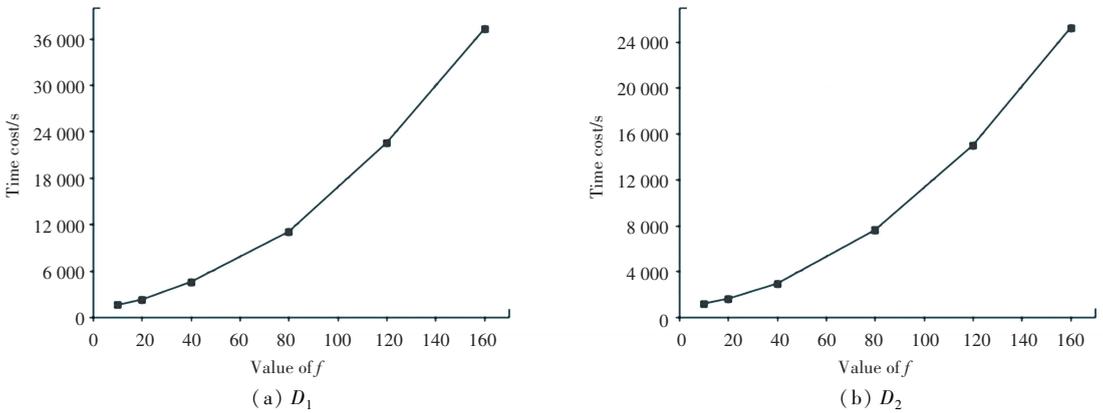


图 4 不同潜在因子维数  $f$  的时间花费

Fig. 4 Time costs with different Latent Factor dimension  $f$

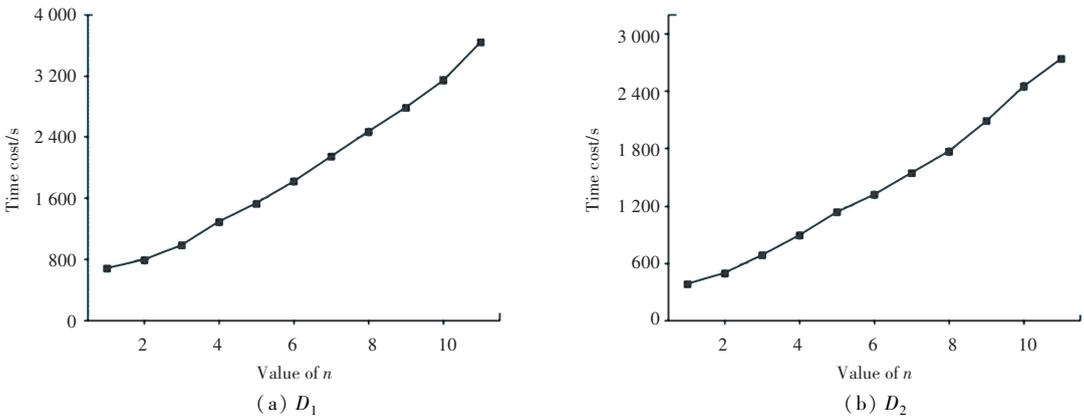


图 5 不同神经网络层数  $n$  的时间花费

Fig. 5 Time costs with different layers of neural networks

综上所述,本次研究中设置 TL-NNLF 模型中参数为:在  $D_1$  中,正则化系数  $\lambda = 0.001$ ,潜在因子维数  $f = 40$ ,神经网络层数  $n = 5$ ;在  $D_2$  中,正则化系数  $\lambda = 0.0008$ ,潜在因子维数  $f = 40$ ,神经网络层数  $n = 5$ 。

### 3.4 实验结果分析

最后为了评估本文所提出模型的有效性,共选

择了 3 种经典的 LF 模型与本文提出模型进行对比实验,具体如下:

- (1)  $M_1$ : NLF<sup>[26]</sup>。NLF 是一种目前应用较广泛的基于单一目标领域的非负 LF 模型。
- (2)  $M_2$ : CMF<sup>[7]</sup>。CMF 是一种跨领域的联合矩阵分解。
- (3)  $M_3$ : MLF<sup>[20]</sup>。MLF 是一种基于随机算法

的 LF 模型。

(4)  $M_4$ : TL-NNLF。TL-NNLF 是本文提出的一种新颖的 LF 模型。

接着根据上述参数设置对模型  $M_1 \sim M_4$  进行对

比实验,最终结果见表 1。表 1 中,每个括号中的值表示该模型在某一指标下的排序值。这里,仅考虑了本文提出模型在目标领域中的预测误差和训练时间,值得注意的是,该模型同样适用于辅助领域。

表 1 模型  $M_1 \sim M_4$  在  $D_1 \sim D_2$  的最小预测误差和相应时间开销

Tab. 1 Lowest prediction error and the corresponding time costs achieved by models  $M_1 \sim M_4$  on  $D_1 \sim D_2$

	$D_1$					$D_2$				
	RMSE	Time	MAE	Time	Mean Ranking	RMSE	Time	MAE	Time	Mean Ranking
$M_1$	0.124 5(3)	2 689.59(3)	0.109 3(4)	2 862.25(3)	3.25	0.081 2(3)	2 084.96(3)	0.075 8(3)	2 076.91(3)	3
$M_2$	0.108 3(1)	4 156.62(4)	0.089 2(1)	4 268.21(4)	2.5	0.071 1(2)	3 499.62(4)	0.063 0(2)	3 512.83(4)	3
$M_3$	0.126 7(4)	869.43(1)	0.106 2(3)	869.64(1)	2.25	0.083 8(4)	683.96(1)	0.081 2(4)	679.78(1)	2.5
$M_4$	0.112 1(2)	1 563.79(2)	0.091 5(2)	1 569.98(2)	<b>2</b>	0.067 2(1)	1 052.81(2)	0.059 2(1)	1 032.13(2)	<b>1.5</b>

研究中遵循的具体排序规则如下:首先将 4 个模型基于评价指标和花费时间值的大小分别进行排序,其中 1 表示最好,4 表示最差;其次分别计算每个模型的平均排名,例如在数据集  $D_1$  中,模型  $M_1$  的每项排名分别为:3,3,4,3,所以模型  $M_1$  的整体平均排名为  $(3+3+4+3)/4=3.25$ 。此处的平均排名值越低,表示模型整体性能越好。

至此,由表 1 中可以得到以下结论:

(1) 模型  $M_4$  在 2 个真实数据集上关于评价指标  $RMSE$  和  $MAE$  的整体表现明显优于模型  $M_1$  和  $M_3$ 。具体地,在数据集  $D_1$  中,模型  $M_1$  和  $M_3$  分别取得最小  $RMSE$  值为 0.124 5 和 0.126 7,而引入了迁移学习的  $M_4$  所取得的最小  $RMSE$  值为 0.112 1,较  $M_1$  和  $M_3$  相比分别提升了 1.24%,1.26%。类似地,基于评价指标  $MAE$ ,模型  $M_4$  的表现也有明显提升。

(2) 模型  $M_4$  在模型训练时间上的花费远远小于模型  $M_2$ 。具体地,在数据集  $D_1$  中,模型  $M_2$  取得的最小  $RMSE$  值为 0.108 3,而其在训练过程中需要花费 4 156.62 min;尽管模型  $M_4$  取得的最小  $RMSE$  值为 0.112 1,较  $M_1$  相比降低了 0.38%,而模型训练仅花费 1 563.79 min。

(3) 模型  $M_4$  的整体排名都优于其他传统的 LF 模型。也就是说,模型  $M_4$  能够在保证模型预测精度的前提下大大减少了模型训练时间。因此,综合考虑模型预测精度和训练花费时间这 2 个方面,模型  $M_4$  性能优于其他经典的 LF 模型。

## 4 结束语

如何有效缓解数据稀疏问题逐渐成为一大研究热点。本文综合考虑模型预测精度和训练花费时间

两个方面,一方面基于不同模式下评分数据之间的关联性,引入迁移学习将二评分数据的潜在知识迁移到五分制评分领域中以缓解其存在数据稀疏问题;另一方面,利用神经网络进行模型训练以加快模型训练速度,并且通过合理地设置实验参数,能够在保证较好的预测精度的前提下大大缩短了模型训练所花费的时间。最终在 2 个真实数据集上的实验结果表明,本文提出的模型整体性能优于其他传统 LF 模型。最后,如何进一步充分挖掘不同评分模式之间的公共知识将成为未来的重要研究方向。

## 参考文献

- [1] SONG Yan, LI Ming, LUO Xin, et al. Improved symmetric and non-negative matrix factorization models for undirected sparse and large-scaled networks: A trip factorization-based approach[J]. IEEE Transactions on Industrial Informatics, 2020, 16(5): 3006-3017.
- [2] 刘鑫鹏,栾悉道,谢毓湘,等. 迁移学习研究和算法综述[J]. 长沙大学学报, 2018, 32(05): 28-31,36.
- [3] LUO X, YUAN Y, ZHOU M C, Liu Z G, Shang M S. Nonnegative latent factor model based on  $\beta$ -divergence for recommender systems[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2019, 51(8): 4612-4623.
- [4] ZHANG Sheng, WANG Weihong, FORD J, et al. Learning from incomplete ratings using non-negative matrix factorization[C]// Proceedings of the 2006 SIAM International Conference on Data Mining. Bethesda, MD, USA: SIAM, 2006: 549-553.
- [5] HUANG Linqing, LIU Zhunga, PAN Quan, et al. Evidential combination of augmented multi-source of information based on domain adaptation[J]. Science China Information Sciences, 2020, 63: 210203.
- [6] JIANG Shuhui, DING Zhengming, FU Yun. Heterogeneous recommendation via deep low-rank sparse collective factorization [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020, 42: 1097-1111.
- [7] PAN Weike, MING Zhong. Interaction-rich transfer learning for

- collaborative filtering with heterogeneous user feedback[J]. IEEE Intelligent Systems, 2014, 29: 48-54.
- [8] ZHU Y. Review on heterogeneous transfer learning [J]. Telecommunications Science, 2020,36(3):100-110.
- [9] LUO Xin, ZHOU Mingchu, XIA Yunni, et al. An efficient nonnegative matrix factorization based approach to collaborative filtering for recommender systems [J]. IEEE Transactions on Industrial Informatics, 2014, 10(2): 1273-1284.
- [10] LUO Xin, ZHOU Mengchu, LI Shuai, et al. A nonnegative latent factor model for large scale sparse matrices in recommender systems via alternating direction method[J]. IEEE Transactions on Neural Networks and Learning Systems, 2016, 27(3): 579-592.
- [11] 朱应钊. 异构迁移学习研究综述[J]. 电信科学, 2020,36(03): 100-110.
- [12] LUO Xin, LIU Zhigang, LI Shuai, et al. A fast non-negative latent factor model based on generalized momentum method[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2021,51(1): 610-620.
- [13] WANG H. A review of transfer learning research[J]. Computer Knowledge and Technology, 2017,13(32):203-205.
- [14] WEISS K, KHOSHGOFTAAR T, WANG Dingding. A survey of transfer learning[J]. Journal of Big Data, 2016, 3: 9.
- [15] 冯文, 陈志国, 傅毅, 等. 增强碰撞体算法优化的自编码神经网络[J]. 小型微型计算机系统, 2019, 40(04): 721-725.
- [16] PAN Weike. A survey of transfer learning for collaborative recommendation with auxiliary data[J]. Neurocomputing, 2016, 177: 447-453.
- [17] 梁鹏, 黎绍发, 林智勇, 等. 共享域特征的深度神经网络异常检测方法[J]. 小型微型计算机系统, 2018, 39(06): 1234-1238.
- [18] SEDHAIN S, MENON A K, SANNER S, et al. AutoRec: autoencoders meet collaborative filtering[C]// Proceedings of the 24<sup>th</sup> International Conference on World Wide Web. Florence, Italy: Association for Computing Machinery, 2015: 111-112.
- [19] FENG Wen, CHEN Zhiguo, FU Yi, Wang Kai-yu. Autoencoder neural network optimized by enhanced colliding bodies optimization[J]. Journal of Chinese Computer Systems, 2019, 40(4): 721-725.
- [20] JIANG Jiajia, LI Weiling, DONG Ani, et al. A fast deep autoencoder for high - dimensional and sparse matrices in recommender systems [J]. Neurocomputing, 2020, 412: 381 - 391.
- [21] PAN Weike, YANG Qiang. Transfer learning in heterogeneous collaborative filtering domains [J]. Artificial Intelligence, 2013, 197(1): 39-55.
- [22] 李依桐, 黄岳, 石川, 等. 基于矩阵分解的异质信息网络聚类分析研究[J]. 小型微型计算机系统, 2014, 35(10): 2256-2261.
- [23] LIANG Peng, LI Shaofa, LIN Zhiyong, et al. Deep neural networks anomaly detection method via shared domain feature[J]. Journal of Chinese Computer Systems, 2018, 39(6): 1234-1238.
- [24] GULIYEV N J, ISMAILOV V E. A single hidden layer feedforward network with only one neuron in the hidden layer can approximate any univariate function [J]. Neural Computation, 2016, 28(7): 1289-1304.
- [25] BACCIU D, COLOMBO M, MORELLI D, et al. Randomized neural networks for preference learning with physiological data[J]. Neurocomputing, 2018, 298: 9-20.
- [26] LIU Kai, ZHANG Limin, ZHOU Lijun. Transfer Learning Research and Algorithm Review [J]. Journal of Changsha University, 2018, 32(5): 28-31.
- [27] LUO Xin, ZHOU Mengchu, XIA Yunni, et al. Generating highly accurate predictions for missing QoS data via aggregating nonnegative Latent factor models[J]. IEEE Transactions on Neural Networks and Learning Systems, 2016, 27(3): 524-537.
- [28] SINGH A P, GORDON G J. Relational learning via collective matrix factorization[C]//Proceedings of the 14<sup>th</sup> ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2008: 650-658.
- [29] YUAN Ye, HE Qiang, LUO Xin, et al. A multilayered-and-randomized latent factor model for high-dimensional and sparse matrices [J]. IEEE Transactions on Big Data, DOI: 10.1109/TBDATA.2020.2988778.
- [30] LI Yitong, HUANG Yue, SHI Chuan, et al. Clustering analysis in heterogeneous information network based on matrix factorization [J]. Journal of Chinese Computer Systems, 2014, 35(10): 2256-2261.
- [31] KOREN Y, BELL R, VOLINSKY C. Matrix factorization techniques for recommender systems [J]. Computer, 2009, 42(8): 30-37.
- [32] HARPER F M, KONSTAN J A. The movieLens datasets: History and context [J]. ACM Transactions on Interactive Intelligent Systems, 2015, 5(4):19: 1-19:19.

(上接第19页)

- [28] HE K, GKIOXARI G, DOLLÁR P, et al. Mask R-CNN[J]. Proceedings of the IEEE International Conference on Computer Vision. Venice:IEEE,2017: 2961-2969 .
- [29] DONG Zhikai, LIU Sicheng, ZHOU Tao, et al. PPR-Net: Point-wise pose regression network for instance segmentation and 6D pose estimation in bin-picking scenarios [C]// 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Macau, China: IEEE, 2019:1773-1780.
- [30] ZHANG C L, LUO Jianhao, WEI Xiushen, et al. In defense of fully connected layers in visual representation transfer [M]// ZENG B, HUANG Q, SADDIK E A, et al. Advances in Multimedia Information Processing - PCM 2017. PCM 2017. Lecture Notes in Computer Science. Cham: Springer, 2017, 10736:807-817.
- [31] GIRSHICK R. Fast R - CNN [J]. arXiv preprint arXiv: 1504.08083, 2015.