基于GPU加速和非负矩阵分解的并行协同过滤推荐算法

康林瑶 唐 兵 夏艳敏 张 黎
(湖南科技大学计算机科学与工程学院 湖南 湘潭 411201)

摘 要 协同过滤(CF)已经在推荐系统中得到了广泛的应用。但是随着用户和项目规模的增大,协同过滤算法的运行效率以及结果的正确性会大大降低。针对这一问题,文中提出了一种基于GPU的非负矩阵分解(NMF)的并行协同过滤方法,充分利用NMF数据降维和特征提取的优势以及CUDA的多核并行计算模式,进行维数简化和用户的相似性计算。该算法在提高精确性的同时降低了计算耗费,可以较好地解决协同过滤推荐系统所存在的稀疏性和扩展性等问题,快速产生精确的个性化推荐结果。基于NVIDIA CUDA设备和真实的MovieLens用户评分数据集,将所设计的并行NMF协同过滤算法与传统的基于用户的和基于物品的协同过滤算法进行了比较,实验结果表明,所设计的并行NMF协同过滤算法达到了较快的处理速度以及较高的推荐准确率。

关键词 协同过滤,非负矩阵分解,GPU,推荐算法
中图法分类号 TP391 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2019.08.017

GPU-accelerated Non-negative Matrix Factorization-based Parallel Collaborative Filtering Recommendation Algorithm

KANG Lin-yao TANG Bing XIA Yan-min ZHANG Li
(School of Computer Science and Engineering, Hunan University of Science and Technology, Xiangtan, Hunan 411201, China)

Abstract Collaborative filtering (CF) is widely used in recommendation systems. However, with the increase of user and item number, the efficiency of collaborative filtering algorithm and the correctness of the result will be greatly reduced. To solve this problem, this paper proposed a GPU-accelerated non-negative matrix factorization (NMF)-based parallel collaborative filtering algorithm. By utilizing the advantages of data dimensionality reduction and feature extraction of NMF, as well as the multi-core parallel computing mode of CUDA, dimension reduction and user similarity are realized. The proposed algorithm improves the recommendation accuracy and also reduces the computational cost, which can better solve the sparseness and scalability of CF-based recommendation system, and generate accurate and personalized recommendations quickly. The new algorithm was evaluated on a NVIDIA CUDA device using real MovieLens datasets. Experimental results show that, NMF-based collaborative filtering outperforms traditional User-based and Item-based CF with higher processing speed and higher accuracy recommendations.

Keywords Collaborative filtering, Non-negative matrix factorization, GPU, Recommendation algorithm

协同过滤(Collaborative Filtering,CF)推荐系统的原理是通过已有的历史兴趣相近群体的偏好来预测当前用户的兴趣喜好。其基本思想是将当前用户的数据与其他用户的数据进行比对,具体地,即是通过计算两个用户数据之间的相似性,来说明两个用户之间的相似程度[1]。协同过滤是目前使用最广泛的推荐算法,但是随着数据量以及数据复杂性的增加,数据的存储和计算问题逐渐凸显,这时就需要对推荐算法进行改进。非负矩阵分解(Nonnegative Matrix Factorization,NMF)可以有效地处理大数据。NMF对所有元素均为非负数的矩阵进行分解,得到两个低秩矩阵,可以实现对数据的降维和特征的提取[2]。与PCA(主成分分析)、ICA(独立成分分析)、SVD(奇异值分解)、VQ(矢量量化)等矩阵分解不同,NMF克服了传统矩阵分解的很多问题,其结果具有很直观的语义解释。因此,本文利用NMF进行数据处理。除了优化算法,还须对运行平台进行改变,以往以单机形式运行的协同过滤算法的运行速度较慢,随着数据的增加,也越来越不能满足应用的需求。因此,本文将运行平台移至GPU(Graphic Processing Unit),从而产生了基于NMF并行的协同过滤算

到稿日期:2018-09-15 返修日期:2018-11-29 本文受国家自然科学基金项目(61602169),湖南省自然科学基金项目(2018JJ2135),湖南省教育厅科研项目(18A186)资助。
康林瑶(1995-),女,硕士生,主要研究方向为大数据,E-mail:kanglinyao@qq.com;**唐 兵**(1982-),男,博士,副教授,主要研究方向为分布式计算与大数据,E-mail:btang@hnust.edu.cn(通信作者);**夏艳敏**(1995-),女,硕士生,主要研究方向为服务计算与大数据;**张 黎**(1981-),女,硕士,讲师,主要研究方向为大数据。

法。实验结果表明，NMF 的并行化有效地提高了计算效率以及准确率。

GPU 是一种应用于图形图像处理的多核处理器，是为可并行化计算密集型的任务而设计的，拥有非常强的计算处理能力和非常好的数据吞吐量。在科研和实际应用中，往往是将系统中可并行化并且少逻辑的处理计算任务模块移植到 GPU 上执行，这样通常能够获取大幅度的性能提升。

虽然传统的协同过滤在推荐系统中极其成功，但是随着数据的增多，算法会遇到各种问题，例如严重的扩展性问题、冷启动问题、矩阵稀疏性问题，推荐系统的可信性也会随之降低。为了解决协同过滤存在的问题，本文提出了基于矩阵分解的协同过滤方法，其在数据规模很大的情况下也能够产生比较精确的推荐结果，并且将矩阵分解并行化移植到 GPU 实现，有效地提升了计算效率，极大地减少了系统的运行时间，可以更快地更新推荐结果数据，使系统的实时性得到提高，对实际应用有很大的实用性[3-4]。

1　非负矩阵分解

NMF 的思想自 Lee 等于 1999 年在 *Nature* 上提出开始，就迅速得到了人们的重视[2]。一方面，科学研究中的很多大规模数据的分析方法需要通过矩阵形式进行有效处理，其中一个关键的步骤便是对矩阵进行分解操作，即对矩阵的维数进行削减，也可以对大量的数据进行压缩和概括，NMF 思想则为处理大规模数据提供了一种新的途径；另一方面，相较于传统的一些算法而言，NMF 分解算法具有实现上的简便性，以及分解形式和分解结果上的可解释性[5-7]。

NMF 的基本思想可描述为：任意给定一个非负矩阵 $\boldsymbol{V}=(v_{ij})_{n\times m}=[v_1,v_2,\cdots,v_m]$，寻找一个非负矩阵 $\boldsymbol{W}\in\boldsymbol{R}^{n\times r}\boldsymbol{W}\in\boldsymbol{R}^{n\times r}$ 和一个非负矩阵 $\boldsymbol{H}\in\boldsymbol{R}^{r\times m}$，满足 $\boldsymbol{V}\approx\boldsymbol{WH}$。原矩阵 $\boldsymbol{V}$ 中的一列向量可以解释为对左矩阵 $\boldsymbol{W}$ 中所有列向量（称为基向量）的加权和，而权重系数为右矩阵 $\boldsymbol{H}$ 中对应列向量中的元素。在数学上，从计算的观点看，分解结果中存在负值是正确的，但负值元素在实际问题中往往是没有意义的，也是无法解释的。NMF 通过寻找低秩，对元素都为非负值的矩阵进行非负分解（即分解出的矩阵元素值也为非负），这在现实应用中有很多例子（如图像的像素灰度值都是非负的），它是一种很有效的分解多维数据的方法。

NMF 是 NP 问题，可以转化成优化问题：通过迭代方法求解 $\boldsymbol{W}$ 和 $\boldsymbol{H}$，只能使分解误差尽可能小，定义适当的目标函数，通过最小化目标函数来找到 $\boldsymbol{V}$ 尽可能精确的分解[2,7]。通常使用两种目标函数，一种是基于欧氏距离的，另一种是基于 K-L 离散度的，它们的定义分别如下：

$$\boldsymbol{E}(\boldsymbol{V}\parallel\boldsymbol{WH})=\parallel\boldsymbol{V}-\boldsymbol{WH}\parallel_F^2=\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}(v_{ij}-\sum_{k=1}^{r}w_{ik}h_{kj})^2 \tag{1}$$

$$\boldsymbol{D}(\boldsymbol{V}\parallel\boldsymbol{WH})=\sum_{i,j}(v_{ij}\log\frac{v_{ij}}{(\boldsymbol{WH})_{ij}}-v_{ij}+(\boldsymbol{WH})_{ij}) \tag{2}$$

那么，NMF 的最优化问题即为：

$$\min_{\boldsymbol{W},\boldsymbol{H}}\boldsymbol{E}(\boldsymbol{V}\parallel\boldsymbol{WH})\text{或}\min_{\boldsymbol{W},\boldsymbol{H}}\boldsymbol{D}(\boldsymbol{V}\parallel\boldsymbol{WH})$$
$$\text{s.t. }\boldsymbol{W},\boldsymbol{H}\geqslant 0,\sum_{i=1}^{n}w_{ij}=1,1\leqslant j\leqslant r \tag{3}$$

如果采用欧氏距离，则通过下列迭代可以得到上述最优化问题的局部最优解：

$$\begin{cases}h_{ij}=h_{ij}\dfrac{(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{V})_{ij}}{(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{WH})_{ij}}\\ w_{ij}=w_{ij}\dfrac{(\boldsymbol{VH}^{\mathrm{T}})_{ij}}{(\boldsymbol{WHH}^{\mathrm{T}})_{ij}}\end{cases} \tag{4}$$

在现实应用中，可以根据实际情况设计不同形式的误差度量函数，但要能够证明目标函数的求解规则具有收敛性。如果采用不同的目标函数，则会产生不同的迭代公式。本文主要以欧氏距离作为目标函数为例，采用式(4)中的迭代原理来阐述 NMF 并行算法。

2　基于 NMF 的并行协同过滤算法

2.1　传统的协同过滤推荐算法

协同过滤推荐算法主要用来预测和推荐，是较为流行的推荐算法。协同过滤算法根据已有兴趣相近的用户群的历史行为或意见预测当前用户最可能喜欢的项目，它分为两类，分别是基于用户的协同过滤算法（User- based Collaborative Filtering）和基于物品的协同过滤算法（Item-based Collaborative Filtering）。基于协同过滤技术的推荐过程可描述为 3 个阶段[1]。

(1)收集用户偏好。将用户行为数据进行预处理之后，根据不同应用的行为分析方法，可以选择分组或者加权处理，得到一个 $n\times m$ 的“用户-物品”偏好二维矩阵 $\boldsymbol{V}$。其中，n 是用户数；m 是物品数；矩阵元素 v_{ij} 是用户 i 对物品 j 的偏好，一般是[1,5]的浮点数或 0/1 值，具体的取值与物品的内容相关。如果物品是电子商务中的货品，则 v_{ij} 表示用户是否购买，例如 1 代表购买，0 代表没有购买；如果物品是电影种类，则 v_{ij} 表示观看与否或者兴趣度的高低。

(2)找到相似的用户或物品。在“用户-物品”偏好矩阵中，将一个用户对所有物品的偏好作为一个向量来计算用户之间的相似度，得到相似度矩阵 ***sim***。对于某个用户 u，从系统中剩下的 $n-1$ 个用户中（根据其与用户 u 的相似度值从大到小排列）选取用户 u 的 k 个相似度最大的近邻用户，组成最近邻用户集 $N=\{n_1,n_2,\cdots,n_k\}$。

如果是基于物品的协同过滤算法，则需要把所有用户对某个物品的偏好作为一个向量来计算物品之间的相似度。通常，计算相似度的方法有 3 种，即欧几里得距离、皮尔逊相关系数（Pearson Correlation Coefficient）、余弦相似度（Cosine Similarity）。本文以皮尔逊相关系数为例[8-10]。采用皮尔逊相关系数时，用户之间的相似性计算公式如下：

$$sim(u_1,u_2)=\frac{m\sum_{j=1}^{m}v_{u_1j}v_{u_2j}-\sum_{j=1}^{m}v_{u_1j}\sum_{j=1}^{m}v_{u_2j}}{\sqrt{m\sum_{j=1}^{m}v_{u_1j}^2-(\sum_{j=1}^{m}v_{u_1j})^2}\sqrt{m\sum_{j=1}^{m}v_{u_2j}^2-(\sum_{j=1}^{m}v_{u_2j})^2}} \tag{5}$$

其中，u_1 和 u_2 是两个用户，$sim(u_1,u_2)$ 是用户 u_1 和用户 u_2 的相似度，v_{u_1j} 和 v_{u_2j} 是用户 u_1 和用户 u_2 对物品 j 的评分。

(3)产生评分预测矩阵和 Top-N 推荐数据集。

1)产生评分预测矩阵。基于最近邻以及最近邻对物品的评分，利用相似度加权平均值计算用户对特定物品的评分。

用户 u 的最近邻集合 $N=\{n_1,n_2,\cdots,n_k\}$，用户 u 对某个物品 i 的预测评分为 v_{ui}，则 v_{ui} 的计算公式如式(6)所示：

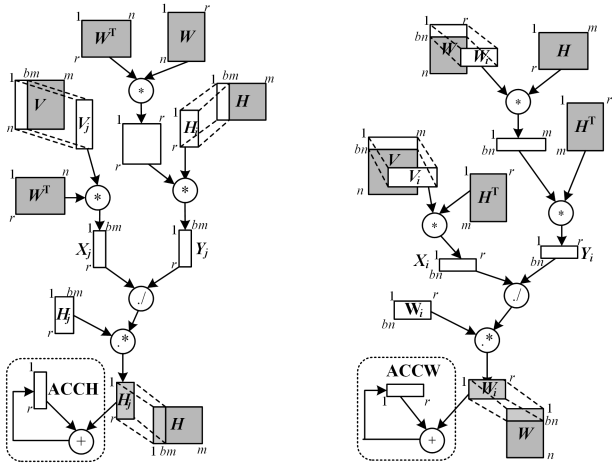
$$v'_{ui}=\bar{u}+\frac{\sum_{r\in N} sim(u,r)\times(v_{ri}-\bar{r})}{\sum_{v\in N} sim(u,v)} \tag{6}$$

其中，$\bar{u}$ 为用户 u 对物品的平均评分，$sim(u,r)$ 是用户 u 和用户 r 之间的相似度，v_{ri} 为用户 r 对物品 i 的评分，$\bar{r}$ 为用户 r 对物品的平均评分。

2)产生 Top-N 推荐数据集。把用户 i 没有评分或者购买的物品按照预测分值的高低进行排序，取前 N 个物品作为 Top-N 物品推荐集推荐给用户 i。

2.2 基于 GPU 的并行 NMF 算法

与式(4)中矩阵 $\boldsymbol{H}$ 和 $\boldsymbol{W}$ 的迭代公式相对应，图 1 给出了基于 GPU 的非负矩阵分解中迭代更新的基本原理。图中的圆圈代表 GPU 内核函数，矩阵乘法采用的是 CUDA 中的 CUBLAS 库的 cblas_sgemm 函数，此外需要利用 C 语言编程实现 2 个内核函数：点除(./)和点乘(.*)。



(a)$\boldsymbol{H}$ 矩阵迭代 (b)$\boldsymbol{W}$ 矩阵迭代

图 1 基于 GPU 的 NMF 中 $\boldsymbol{W}$ 和 $\boldsymbol{H}$ 矩阵迭代更新的原理

Fig. 1 Iteration update principle of matrix $\boldsymbol{H}$ and $\boldsymbol{W}$ in NMF based on GPU

2.3 基于 NMF 的协同过滤算法

本文提出的基于 NMF 的协同过滤算法可以分为两个过程：矩阵分解降维和协同过滤。

(1)矩阵分解降维

Step1 采用基于 GPU 的 NMF 将大规模用户偏好矩阵 $\boldsymbol{V}$ 分解成两个矩阵相乘 $\boldsymbol{W}\times\boldsymbol{H}$，基矩阵 $\boldsymbol{W}$ 是物品特征矩阵，投影矩阵 $\boldsymbol{H}$ 是用户特征矩阵，分解时设置的秩 r 代表着主题数；

Step2 根据矩阵 $\boldsymbol{W}$，可以计算出目标用户评分向量 $\boldsymbol{V}_i$ 对基矩阵 $\boldsymbol{W}$ 的投影向量为 h_i。

(2)协同过滤

Step1 基于 GPU 用户相似度计算，为每个用户分配一个线程，设计一个内核函数用于计算皮尔逊相关系数，采用并行的方式计算 $\boldsymbol{h}_i$ 与投影矩阵 $\boldsymbol{H}$ 各列之间的相似度，得到用户相似度矩阵 $\boldsymbol{sim}$；

Step2 将相似度最高的 k 个用户组成用户 u_i 的最近邻集合 N；

Step3 用最近邻集合中的 u_i 的邻居与 $\boldsymbol{V}$ 中的相应评分进行加权计算，得到评分预测矩阵 $\boldsymbol{p}$；

Step4 排序得到 Top-N 推荐数据集，据此进行推荐。

3 算法结果分析

3.1 实验条件

实验使用 GroupLens 项目研究组所提供的 MovieLens 数据集，该数据集记录了 7120 个用户对 130642 部电影的评分记录，从中随机选取了不同大小的数据集进行实验，每个用户至少对 20 部电影进行了评分，评分范围为 1～5，分值越高表示越满意。

在实验中，将评分数据转化为评分矩阵，如果某用户对某物品没有评分，则所对应的矩阵元素值为 0，评分矩阵是典型的稀疏矩阵。在本文实验中，随机选取了部分数据集，整个实验需要将随机选取的评分数据集 $\boldsymbol{V}$ 按照 7∶3 的比例进一步划分为训练集 $\boldsymbol{VT}$ 和测试集 $\boldsymbol{T}$，矩阵 $\boldsymbol{VT}$ 和矩阵 $\boldsymbol{T}$ 的大小与矩阵 $\boldsymbol{V}$ 的大小一致。在实验中考虑了下列 5 种规模的评分数据：400×800，400×1600，800×1600，800×3200，1200×3200。用户数从 400 到 1200，物品数从 800 到 3200。

在实验中采用了 Dell T3610 图形工作站，其硬件配置是 Intel Xeon 6 核 E5-1650 V2 处理器，32GB DDR3 1886 内存，NVIDIA Quadro K4000 GPU 显卡(768 CUDA cores)，并采用 CUDA 编程方法进行 GPU 程序的开发。

3.2 算法的评价指标

为了精确衡量算法之间的差别，除了算法运行时间外，还考虑了预测评分和物品推荐结果。对于预测评分的准确性，本文采用均方根误差(RMSE)和平均绝对误差(MAE)进行衡量。对于推荐结果的准确性，一般采用正确率(Precision)和召回率(Recall)来度量，但正确率和召回率指标会出现矛盾的情况，这时用 F-Measure 来综合考虑[1]。

RMSE 根据预测分值和用户实际评分之间的均方根误差来度量预测的准确性。RMSE 越小，说明预测结果越准确，推荐算法的质量越高。通过训练集得到预测评分集合 $p=\{p_1,p_2,\cdots,p_n\}$，测试集中实际的用户偏好评分集合 $T=\{t_1,t_2,\cdots,t_k\}$，则 RMSE 的定义为：

$$RMSE=\sqrt{\sum_{i=1}^{n}(p_i-t_i)^2/n} \tag{7}$$

MAE 根据预测分值和实际用户评分之间的均值偏差来度量预测的准确性。MAE 的公式定义为：

$$MAE=\frac{\sum_{i=1}^{n}|p_i-t_i|}{n} \tag{8}$$

从未购买或未评分的物品中选择预测评分最高的 N 个物品组成 Top-N 推荐列表。定义 R_u 是为用户 u 进行推荐的物品集合，T_u 为测试集中用户 u 实际喜欢的物品集合。准确率表示推荐的物品中相关的物品所占的比例，简单来说即为推荐的命中率(命中，即所推荐的物品在测试集中有评分，且评分超过一定的阈值)，亦即推荐的物品中确实是用户所喜欢的物品所占的比例。定义 U 为所有用户的集合，则推荐的正确率 Precision 的定义为：

$$Precision=\frac{\sum_{u\in U}|R_u\cap T_u|}{\sum_{u\in U}|R_u|} \tag{9}$$

召回率 Recall 表示推荐的相关物品占所有相关物品的比例，定义如下：

$$Recall=\frac{\sum_{u\in U}|R_u\cap T_u|}{\sum_{u\in U}|T_u|} \tag{10}$$

F-Measure 是 Precision 和 Recall 的加权调和平均值，定义如下：

$$F\text{-}Measure=\frac{Precision\times Recall\times 2}{Precision+Recall} \tag{11}$$

3.3　测试结果及分析

实验中对传统的基于用户的协同过滤算法（User-based CF）、传统的基于物品的协同过滤算法（Item-base CF）、本文所提的基于 NMF 的协同过滤算法（NMF）这 3 种算法的性能进行了比较。选取 5 组固定矩阵（矩阵的规模分别为 400×800，400×1 600，800×1 600，800×3 200，1 200×3 200）进行算法测试，将 NMF 分解的秩 r 设置为 4，迭代次数设为 100，为每个用户选取 50 个物品作为 Top-50 物品推荐集。

（1）预测评分的准确性比较

3 种算法预测评分的 *RMSE* 以及 *MAE* 的对比如图 2 和图 3 所示。可以看出，在 5 种不同大小的评分矩阵情况下，不论是 *RMSE* 还是 *MAE*，NMF 算法明显优于 User-CF 以及 Item-CF 算法；基于 NMF 的协同过滤推荐算法的 *RMSE* 以及 *MAE* 都是最小的，而 Item-CF 算法的预测评分误差最大，预测评分的效果最差。当矩阵的规模为 400×800 时，与 Item-CF 算法相比，NMF 算法的 *RMSE* 降低了 31.64%，*MAE* 降低了 28.5%。

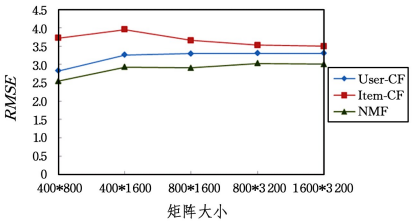


图 2　*RMSE* 的比较

Fig. 2　Comparison of *RMSE*

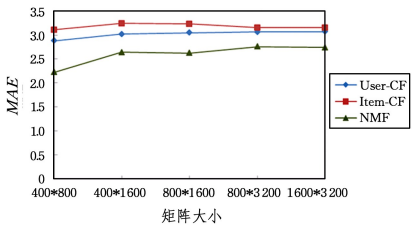


图 3　*MAE* 的比较

Fig. 3　Comparison of *MAE*

（2）推荐物品的准确性比较

3 种算法预测评分的 *Precision*，*Recall* 以及 *F-Measure* 分别如图 4－图 6 所示。可以看出，在 5 种不同大小的评分矩阵情况下，随着矩阵规模的增大，3 种算法的这 3 个指标都有所下降，但不论是 *Precision*，*Recall* 还是 *F-Measure*，NMF 算法均优于 User-CF 算法以及 Item-CF 算法，说明了基于 NMF 的协同过滤推荐的质量最好。随着矩阵规模 $n\times m$ 的增大，特别是 m 的增大（即商品数目增多），由于我们在协同过滤中均只推荐了 50 个商品，因此在计算 *Precision* 和 *Recall* 这两个指标时，会与 30%的测试集进行比对，推荐物品的命中率会较低，NMF 的优势会越来越小；当矩阵大小为 1 600×3 200 时，NMF 与 User-CF 算法的结果几乎相同；而 Item-CF 算法推荐的效果一直是最差的。与 Item-CF 算法相比，NMF 算法的推荐正确率 *Precision* 和召回率 *Recall* 提高了大约 3 倍。

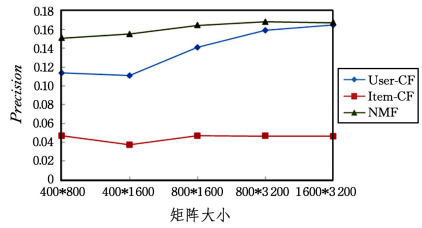


图 4　正确率 *Precision* 的比较

Fig. 4　Comparison of *Precision*

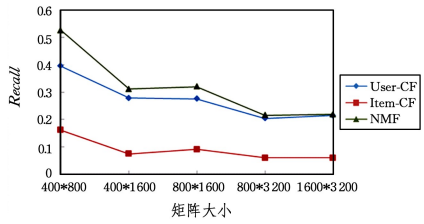


图 5　召回率 *Recall* 的比较

Fig. 5　Comparison of *Recall*

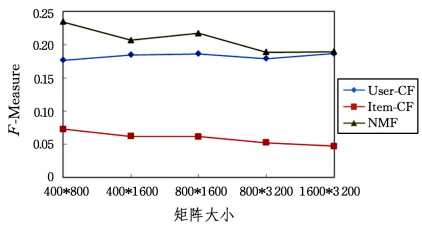


图 6　*F-Measure* 的比较

Fig. 6　Comparison of *F-Measureg*

（3）算法运行时间的比较

首先，仅仅测试非负矩阵分解算法，对比基于 CPU 的非负矩阵分解（称之为 NMF-串行）和基于 CPU＋GPU 的非负矩阵分解（称之为 NMF-并行）。同样在 5 种矩阵大小的情况下对比 NMF 串行算法与 NMF 并行算法的运行时间，结果如图 7 所示。从图中可以看出，在采用了 GPU 加速后计算时间明显缩短，且随着矩阵规模的增大，并行效率越来越高，加速效果越来越好。当矩阵规模为 1 600×3 200 时，由于利用了 GPU，NMF-并行比 NMF-串行快了约 20 倍，这也验证了 GPU 对非负矩阵分解的加速性能。

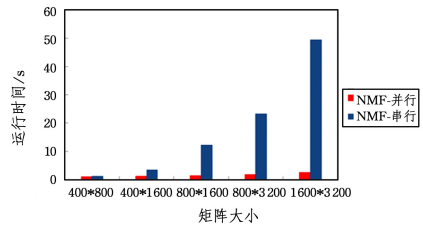


图 7　NMF-串行算法与 NMF-并行算法的时间比较

Fig. 7　Time comparison of serial and parallel NMF

最后，对 3 种算法的运行时间进行对比。3 种算法均采用 GPU 进行皮尔逊相关系数的计算，并且 NMF 协同过滤算法中采用 GPU 进行 NMF 的计算。3 种算法的运行时间比较结果如图 8 所示。从图中可以看出，随着矩阵规模的增大，各个算法的运行时间都有所增加，基于 NMF 的协同过滤推荐算法的运行时间明显少于 User-CF 算法以及 Item-CF 算法的运行时间。当矩阵规模为 1600×3200 时，与 Item-CF 算法相比，NMF 算法的运行时间缩短了 45.0%；与 User-CF 算法相比，NMF 算法的运行时间缩短了 28.3%。

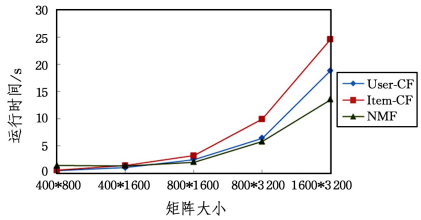


图 8 3 种协同过滤推荐算法的运行时间比较

Fig. 8 Comparison of three CF recommendation algorithms

总体来看，与传统协同过滤相比，基于 NMF 的协同过滤推荐算法加入了评分矩阵 $\boldsymbol{V}$ 分解为 $\boldsymbol{W}$ 和 $\boldsymbol{H}$ 的过程，这看似是个耗时的操作，但在后续计算相关系数时会节省很多时间。由于 $\boldsymbol{W}$ 矩阵的大小为 $n\times r$，r 反映的是用户特征数或者主题数，而 r 的取值通常是非常小的（r 一般取值为 2～4），即 $\boldsymbol{W}$ 矩阵规模很小，因此本文所提出的推荐算法计算相关系数所花费的时间比 User-CF 算法或 Item-CF 算法要短得多。基于 NMF 的协同过滤推荐算法在增加了准确率的基础上，利用 GPU 并行运行，所用时间仍然是最短的。

结束语 推荐系统被广泛应用于生活中的多个领域，但随着用户规模以及数据复杂性的增加，推荐系统的推荐速度变慢，推荐准确率降低。本文实现了基于非负矩阵分解的协同过滤算法，即将非负矩阵分解和传统的协同过滤方法相结合，将原始评分数据分解为基空间以及投影矩阵，并且在 GPU 上并行运行。对不同规模矩阵的测试实验表明，本文所设计的并行 NMF 协同过滤推荐算法不仅提高了预测以及推荐准确率，而且还极大地提高了计算的效率，大大节省了时间。

参 考 文 献

[1] DENG A L, ZHU Y Y, SHI B L. Collaborative filtering Recommendation algorithm based on project score prediction [J]. Journal of Software, 2003, 14(9): 1621-1628. (in Chinese)
邓爱林，朱扬勇，施伯乐. 基于项目评分预测的协同过滤推荐算法[J]. 软件学报，2003，14(9)：1621-1628.

[2] LEE D D H, SEUNG S. Learning the parts of objects by non-negative matrix factorization [J]. Nature, 1999, 401(6755): 788-791.

[3] LIU J F, GUO L. Comparison and analysis of several matrix multiplications on CPU and GPU [J]. Computer Engineering and Applications, 2011, 47(19): 9-11, 23. (in Chinese)
刘进锋，郭雷. CPU 与 GPU 上几种矩阵乘法的比较与分析[J]. 计算机工程与应用，2011，47(19)：9-11，23.

[4] CHENG H, ZHANG Y Q, ZHANG X Y, et al. Implementation and performance analysis of CPU-GPU parallel matrix multiplication [J]. Computer Engineering, 2010, 36(13): 24-26, 29. (in Chinese)
程豪，张云泉，张先轶，等. CPU-GPU 并行矩阵乘法的实现与性能分析[J]. 计算机工程，2010，36(13)：24-26，29.

[5] LIU W X, ZHENG N N, YOU Q B. Non-negative matrix factorization and its application in pattern recognition [J]. Chinese Science Bulletin, 2006, 51(3): 241-250. (in Chinese)
刘维湘，郑南宁，游屈波. 非负矩阵分解及其在模式识别中的应用[J]. 科学通报，2006，51(3)：241-250.

[6] WANG K J, ZUO C T. Research progress of feature extraction techniques for nonnegative matrix factorization [J]. Application Research of Computers, 2014, 31(4): 970-975. (in Chinese)
王科俊，左春婷. 非负矩阵分解特征提取技术的研究进展[J]. 计算机应用研究，2014，31(4)：970-975.

[7] CHEN Y H, REGE M, DONG M, et al. Non-negative matrix factorization for semi-supervised data clustering [J]. Knowledge and Information Systems, 2008, 17(3): 355-379.

[8] KOREN Y, BELL R, VOLINSKY C. Matrix factorization techniques for recommender systems [J]. IEEE Computer, August 2009, 42(8): 40-49.

[9] YANG Y, XIANG Y, XIONG L. Collaborative filtering recommendation algorithm based on matrix decomposition and user neighbor model [J]. Computer Application, 2012, 32(2): 395-398. (in Chinese)
杨阳，向阳，熊磊. 基于矩阵分解与用户近邻模型的协同过滤推荐算法[J]. 计算机应用，2012，32(2)：395-398.

[10] LI G, LI L. Collaborative filtering algorithm based on matrix decomposition[J]. Computer Engineering and Applications, 2011, 47(30): 4-7. (in Chinese)
李改，李磊. 基于矩阵分解的协同过滤算法[J]. 计算机工程与应用，2011，47(30)：4-7.