附件5C：译文

指导教师评定成绩

(五级制)：

指导教师签字：

基于矩阵分解的列表排序学习协同过滤

摘要

我们提出了一种用于协同过滤的排名方法ListRank-MF,这种算法将矩阵分解技术（MF）结合到了排序学习算法中。通过MF排序模型，我们最小化了代表训练列表与输出列表之间的不确定性的损失函数，获得了排序的项目列表。ListRank-MF具有低复杂度的优势，并且与给定用户-物品评分的矩阵成线性关系。我们还通过将其性能与基于项目的协作推荐和相关的最新协作排名方法（CoFiRank）进行比较，通过实验证明了ListRank-MF的有效性。

领域与主题

H.3.3 [信息存储和检索]

一般项

算法，性能，实验

关键词

推荐系统，协同过滤，排序学习，矩阵分解，推荐

1. 概述

推荐系统受到了研究者的关注，因为它在处理空前的大量的数据方面十分重要，例如，它能在线给用户推荐电影、音乐、书籍。而协同过滤算法，则是公认的当前最成功的推荐技术之一。它的基本思想是：如果与A用户有相似爱好的用户B喜欢某个物品，则A用户也会喜欢这个物品。推荐系统的最终目的是为用户提供排名或推荐列表，且这个目标比评分预测的准确性更为重要[1][5][7][13]。出于这个原因，我们在本文中关注的是排名和推荐（也被称为前N推荐），而不是评分预测。

我们提出了一种ListRank-MF方法，将矩阵分解方法扩展到协同过滤。这个方法利用排序学习方法对用户的每个项目进行排名，其中用户和项目表示为用矩阵分解（MF）学习的潜在特征。我们的方法主要有两个贡献：与现有的最先进的矩阵分解方法相比，它改进了推荐性能；这个方法复杂度与给出的用户-物品矩阵中的已有的评分数目呈线性关系，这意味着它可以扩展以用于非常大的集合。

排序学习（LTR）是受监督的机器学习方法，它自动从训练数据构建排序模型或函数[10]。LTR最近一直被深入研究，例如：在雅虎内部就有个LTR Challenge挑战赛。LTR研究既有益于信息检索（IR）也有利于推荐，这两个领域都聚焦于相同任务，即响应信息需求（明确表示为查询（IR）或隐含在用户数据中（推荐）），返回物品排名列表。虽然在响应查询而返回文档的系统的LTR技术方面，人们已经进行了大量的工作[10]，但很少有人致力于在推荐系统中利用LTR，或者特别是在协同过滤中。

将LTR应用到CF这一尝试会面临许多挑战。首先，用户数据和推荐设置中的项目不容易用显式特征来表示。如此一来，推荐就和查询（IR）形成了对比，后者可以明确地表示查询和文档，例如，可以使用从出现频率导出的术语权重。推荐系统的研究者经常将用户和物品表示为它们的评分向量，这就必然使得它与信息检索和文档表示不相同，因为后者与用户和项目特征没有直接关系。为了解决这个问题，我们利用矩阵分解，通过潜在特征来表示用户和物品。其次，并不是所有的LTR方法都与CF应用具有内在适用性。刘[10]等人将LTR分为逐点、成对和列表方式。逐点方法预测单个文档中的排名分数，因此在概念上可视为与CF相同的评分预测问题。根据这些分数对文档进行排序就产生了一个有序表。如文献中所讨论的，评级预测的准确性度量难以解释为排名的度量质量[11][13]。成对LTR对最终表中每对与其相对排序有关的文档进行预测。这种方法计算复杂度较高，这种缺点使得它不能很好地扩展到CF场景的典型大数据集合。在概念上类似于成对LTR的CF的方法已经在文献[11][12][14]中提出，但是计算复杂性问题的可扩展解决方案仍亟待解决。出于这个原因，我们转向了列表方式LTR。

本文剩余结构如下：在下一节，我们总结了相关工作，并对我们的方法进行了定位。然后，我们展示了ListRank-MF算法并实验验证了它。最后，我们总结了ListRank-MF的关键方面，并简要提及了未来工作的方向。

1. 相关工作

本节简要总结了现有协同过滤和相关LTR方法。

* 1. 协同过滤

一些关于协同过滤方法的全面介绍可以在[1][7][18]中找到。协同过滤可以基于记忆，也可以基于模型。通常，基于记忆的方法根据用户（user-based）之间的相似性[6]或者根据物品（item-based）之间的相似性[3][16]做出建议。进一步的修改和改进都致力于改进基于用户的方法[4][17][22]，或基于项目的方法[21]。基于模型的方法首先基于训练数据拟合预测模型，然后使用该模型来预测用户对项目的偏好，例如潜在语义模型[8]。由于其可扩展性和准确性[9][15]的优势，矩阵分解（MF）技术引起了很多研究的关注，特别是对于大规模数据，如Netflix竞赛。通常，MF技术从用户物品矩阵中观察到的评分中学习用户和项目的潜在特征，这些特征又进一步用于预测未观察到的评级。

最近，CF领域的研究关注已经从评级预测问题转移，并且更加关注推荐系统产生的排名或推荐列表的质量[13]。很多方法已经被研究，包括调查用户项目之间的成对偏好，例如EigenRank[11]，概率潜在偏好分析[12]和贝叶斯概率排序[14]。这些方法都因为高昂的计算代价，限制了其可扩展性。相反，本文中提出的ListRank-MF具有低复杂度，即复杂度与给出的用户-物品矩阵中的已有的评分数目呈线性关系。

* 1. 排序学习

关于LTR的详细介绍可以在[10]中找到。LTR文献的累积量很大，在这里我们关注列表方式的LTR方法，这与我们的ListRank-MF方法最相关。在列表方法下，单个训练样本是整个项目列表，而不是单个项目或项目对[2]。列表式LTR将损失函数公式化，以反映参考列表与排序模型的输出列表之间的距离，用各种算法来学习最优或局部最优排序模型。我们提出置换概率来表示排序列表，这样可以进一步简化为最高概率[2]。在本文中，我们还采用顶级概率的概念来表示推荐列表，使我们的工作最接近列表方式的LTR。CoFiRank[20]被提议直接优化协同排序为的排名有效性指标，这主要由LTR推动，也接近本文所讲述的内容。

1. 算法

在本节中，我们将介绍使用矩阵分解进行排名的列表学习（ListRank-MF）。我们首先介绍与ListRank-MF相关的两个关键成分，即概率矩阵分解（PMF）框架和top-one概率。然后我们将ListRank-MF的表达式作为损失函数，并为局部最优解开发相应的学习过程。

* 1. 概率矩阵分解

PMF框架在[15]中提出，其中矩阵分解是根据观察到的评级的条件分布的概率推断，用户评级先验和项目评级先验来制定的。最终框架的表述如下：

假设物品-用户矩阵R由M个用户和N个物品组成，PMF寻求用两个低秩矩阵和表示用户-物品评分矩阵R。我们使用的表示的用户i的d维列特征向量，表示项目j的d维列特征向量。表示用户i对物品j的评分。是一个指示函数，当时等于1，否则为0。最后，和是正则化系数。为简单起见，我们通常设置。是一个逻辑函数，用于约束的范围，例如，可取。

请注意：PMF和一些其他的矩阵分解方法相似，都是面向评分预测的。尽管我们可以用预测的评分去对物品排序，但是排序的质量和PMF目的（最小化评分误差）并不是直接相关的。类似地，PMF相当于逐点排排序模型，而不是直接建模排序。

* 1. Top-one 概率

如[2]所述，用户i的排序列表（例如有K个物品）中物品评分矩阵的Top-one概率可以表示为：

其中可以是任何单调递增且严格正函数。为简单起见，我们采用与[2]中使用的相同的形式，即指数函数作为。Top-one概率表示物品在给定排名列表的最高位置的概率。

* 1. ListRank-MF

我们的ListRank方法通过使用训练示例列表中的项目的top-one概率的交叉熵和来自排序模型（MF）的排名列表作为损失函数来实现。损失函数如下

训练样本列表由每个用户数据的训练集项组成。通过生成的值对集合中的物品进行降序排序，推荐模型的输出是针对每个用户i的推荐列表。用户i的数据中的训练样例中已包含的项目将被删除。

该损失函数反映了训练列表与排名模型的输出列表之间的不确定性，即MF。正则项用于减少过拟合。最佳排名模型应该在训练评分列表和输出列表排序预测列表中，表现出最小的不确定性。请注意，此处MF未针对评分预测进行优化，而是根据用户列表中物品的排名进行优化。由于损失函数不是在U和V上共同凸起，我们选择使用具有交替固定的U和V的梯度下降，从而可以获得局部最小值。损失函数L(U，V)相对于U和V的梯度可以计算如下：

其中表示的导数

* 1. 复杂度分析

利用用户项矩阵的稀疏性，计算公式中的损失函数。（3）具有复杂度，其中S表示给定用户项矩阵中观察到的评分的数量。方程（4）和（5）中的梯度分别是是O(2dS + dM)和O(dS + pdS + dN)，其中p表示每个用户的已平均评分项目数，并且与协作过滤中相比,S通常是非常小的值。考虑到经常具有S >> M,N的情况，一次迭代中的总复杂度是O(dS + pdS)，其在矩阵中观察到的评分的数量是线性的。 该分析表明ListRank-MF计算高效，可应用于大规模案例。

1. 实验和评估

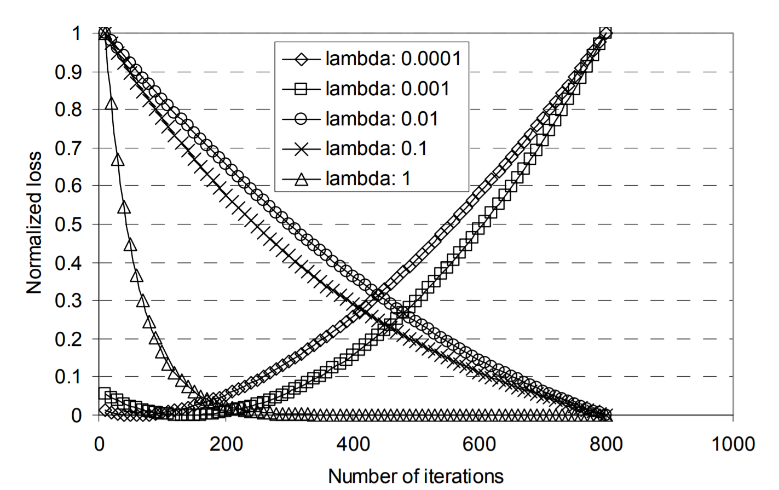
在本节，我们进行了一些初步的实验来评估ListRank-MF。我们首先观察了正则化参数对ListRank-MF的影响,并进一步验证了优化损失函数对排序效果影响的有效性。最后，我们论证了ListRank-MF不仅在学习方面有效，而且在性能方面也很出色，它可以改进最先进的CoFiRank方法。

* 1. 实验准备

我们的实验是在MovieLens数据集上进行的，该数据集由943个用户给1682个项目[6]的10万个评分（分数1-5）组成。我们采用了在[20]中CoFiRank评估的 “弱泛化”的评估协议。我们为每个用户随机选择10,20和50个项目进行训练，并使用用户数据中的其余已评分项目进行测试。对应每种训练方式，少于20,30或60个评分项目的用户将被删除，以确保我们可以评估每个用户至少10个评级项目。我们报告了所有用户的平均性能以及此过程的十次运行。我们选择评估指标作为标准化折扣累积增益（NDCG），它对排名较高项目的相关性更敏感。与CoFiRank [20]一样，我们专注于NDCG@10。 请注意，在所有实验中，我们的ListRank-MF使用了潜在特征维度为5和学习率为0.01进行实现。

* 1. 正则化系数λ的影响

ListRank-MF中的正则化系数影响着损失函数的收敛和过度拟合的控制。为了研究的影响，我们使用一个数据折叠，在此情况下。每个用户的10个随机选择的评分项目中被选择用于训练。图1说明了与损失之间的关系，并表明当低于0.001的水平时有出现过度拟合的风险。注意，为了说明的目的，已经针对的每个设置对损失函数进行了归一化。由于λ等于或大于0.01时没有明显的过度拟合效应，因此为了一致性，我们在以下所有实验中将λ设置为0.01。

图一 正则化系数对学习过程中损失收敛的影响

* 1. 有效性和效率

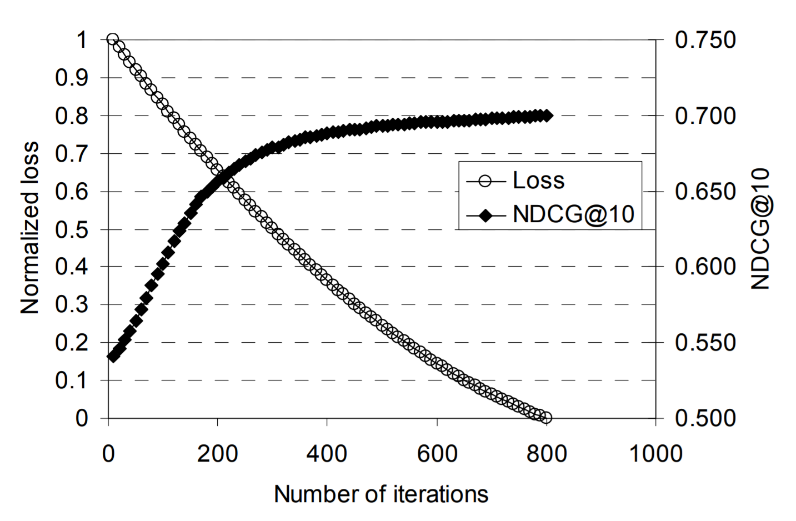
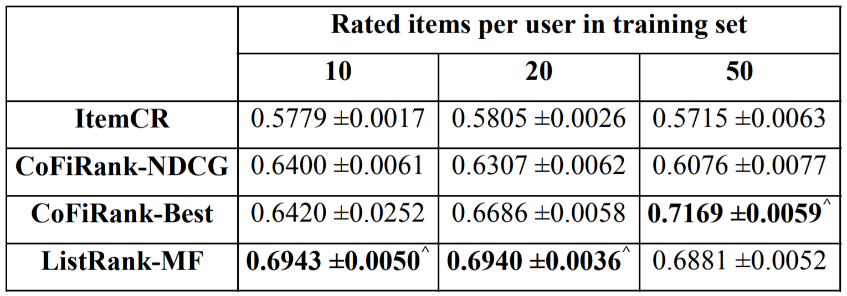
损失函数的优化使得损失最小化。但是，减少损失能否带来一个良好的排名？这个调查仍是必要的，例如本文中使用的NDCG@10。因此，我们在优化迭代期间同时展示了损失和NDCG@10的发展，如图2所示。可以看到在优化损失函数时，排序性能变得最佳和收敛都是有效的。此外，它表明ListRank-MF是有效的，因为NDCG@10在大约250次迭代后变得近似最佳。

图2 ListRank-MF通过最小化损失实现最佳NDCG @ 10的有效性

* 1. 性能比较

在本节中，我们将ListRank-MF的性能与众所周知的基于物品的协作推荐（ItemCR）[3]和最先进的CoFiRank [20]进行比较。由于我们的实验协议与CoFiRank [20]完全相同，我们可以直接比较结果，即[20]中的CoFiRank-NDCG和CoFiRank-Best所述。请注意，CoFiRank-NDCG作为损失函数代表了一种直接优化NDCG的方法，CoFiRank-Best代表了CoFiRank在各种条件下获得最佳结果的选项。如表1所示，ListRank-MF实现了性能的提升，比ItemCR高出约15%，也比CoFiRank-DNCG高出10%。根据Wilcoxon符号秩检验，p <0.05，这个改善对所有情况都很显着。在每个用户10个和20个评分项目进行训练的实验条件下，ListRank-MF也达到了相比CoFiRank最优情况下5％的提高（十分显著），而对于每个用户50个评分项目进行培训的情况则略微优于其他条件。

表1 项目CR，CoFiRank和ListRankMF之间的NDCG@10（10次运行的平均值±标准差），“^”表示相对于其他的显着改进。

请注意，由于我们在每次运行中没有为每个用户获得CoFiRank的性能，因此涉及CoFiRank的显着性检验是通过从高斯分布生成10个样本并且在每个条件下具有相应的均值和偏差来进行的，这进一步与在10次运行中ListRank-MF和ItemCR的10个平均性能进行比较。

1. 讨论与总结

在本文中，我们引入了ListRank-MF，这种方法利用列表对学习技术进行排序，以便实现对用于推荐任务的最新矩阵分解技术的性能的改进。实验证明，在大多数情况下，ListRank-MF优于基于迭代的协作推荐和最先进的CoFiRank。我们还分析了ListRank-MF的计算复杂度，并发现它与给定用户-物品矩阵中观察到的评分的数量是线性的。因为ListRank-MF的计算代价不高，所以它可以扩展用于大型真实世界的集合。

未来的工作涉及几个有意思的方向。首先，本文中的ListRank-MF基于前一概率的概念。从推荐系统的角度来看，top项目是关键的，这提供了列表的合理表示。但是，我们想研究是否可以通过反映整个列表中信息的表示来实现性能改进，而不过度增加计算复杂性。其次，与大多数当前CF推荐算法一样，提议的ListRank-MF方法可以被视为推荐方法的变式，其中共同评估度量与该模型不直接相关，例如平均倒数排名，平均精度和NDCG。CoFiRank[20]首次尝试解决这一问题，并且可以进一步利用LTR研究[10][19]中该领域的最新发展，通过直接优化评估指标来提高推荐性能。

1. 感谢

以上研究是在PetaMedia卓越网络实验室内进行的，并且根据拨款协议编号216444获得了欧盟委员会第7框架计划的资助。

1. 参考文献

[1] Adomavicius G., and Tuzhilin, A., 2005. Toward the next generation of recommender systems: A survey of the state-of-theart and possible extensions. IEEE TKDE, 17, 6, 734-749.

[2] Cao, Z., Qin, T., Liu, T.-Y., Tsai, M.-F. and Li, H., 2007. Learning to rank: From pairwise approach to listwise approach. Technical Report, MSR-TR-2007-40, Microsoft Research.

[3] Deshpande, M., and Karypis, G., 2004. Item-based top-N recommendation algorithms. ACM TOIS, 22, 1, 143-177.

[4] Ding, S., Zhao, S., Yuan, Q., Zhang, X., Fu, R. and Bergman, L., 2008. Boosting collaborative filtering based on statistical prediction errors. In RecSys ′08, 3-10.

[5] Gunawardana A., and Shani, G., 2009. A survey of accuracy evaluation metrics of recommendation tasks. JMLR, 10, 2935- 2962.

[6] Herlocker, J., Konstan, J., Borchers, A., and Riedl, J., 1999. An algorithmic framework for performing collaborative filtering. In SIGIR ′99, 230-237.

[7] Herlocker, J., Konstan, J., Terveen, L. G., and Riedl, J. 2004. Evaluating collaborative filtering recommender systems. ACM TOIS, 22, 1, 5-53.

[8] Hofmann, T., 2004. Latent semantic models for collaborative filtering. ACM TOIS, 22, 1, 89-115.

[9] Koren, Y., Bell, R., and Volinsky, C., 2009. Matrix factorization techniques for recommender systems. IEEE Computer, 42, 8, 30-37.

[10] Liu, T. -Y., 2009. Learning to rank for information retrieval. Foundations and Trends in Information Retrieval, 3, 3, 225- 331.

[11] Liu, N. N., and Yang, Q., 2008. EigenRank: a ranking-oriented approach to collaborative filtering. In SIGIR ′08, 83-90.

[12] Liu, N. N., Zhao, M., and Yang, Q., 2009. Probabilistic latent preference analysis for collaborative filtering. In CIKM ′09, 759-766.

[13] McNee, S. M., Riedl, J., and Konstan, J. A., 2006. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In extended abstracts of CHI ′06, 1097-1101.

[14] Rendle, S., Freudenthaler, C., Gantner, Z., and SchmidtThieme L., 2009. BPR: Bayesian personalized ranking from implicit feedback. In UAI ′09, 452-461.

[15] Salakhutdinov, R., and Mnih, A., 2008. Probabilistic matrix factorization. In NIPS ’08, 20.

[16] Sarwar, B., Karypis, G., Konstan, J., and Reidl, J., 2001. Itembased collaborative filtering recommendation algorithms. In WWW ′01, 285-295.

[17] Shi, Y., Larson, M., and Hanjalic, A., 2009. Exploiting user similarity based on rated-item pools for improved user-based collaborative filtering. In RecSys ′09, 125-132.

[18] Su, X. and Khoshgoftaar, T. M., 2009. A survey of collaborative filtering techniques. Advances in Artificial Intelligence, no. 421425, 19 pages.

[19] Volkovs, M. N., and Zemel, R. S., 2009. BoltzRank: learning to maximize expected ranking gain. In ICML ′09, 1089-1096.

[20] Weimer, M, Karatzoglou, A., Le, Q., and Smola, A., 2007. CoFi rank-maximum margin matrix factorization for collaborative ranking. In NIPS ′07, 20, 1593-1600.

[21] Yildirim, H., and Krishnamoorthy, M. S., 2008. A random walk method for alleviating the sparsity problem in collaborative filtering. In RecSys ′08, 131-138.

[22] Zhang, J. and Pu, P., 2007. A recursive prediction algorithm for collaborative filtering recommender systems. In RecSys ′07, 57-64.

译文原文出处：Shi Y，Larson M，Hanjalic A．List-wise learning to rank

with matrix factorization for collaborative filtering[C]//

Proceedings of the Fourth ACM Conference on Recommender Systems．Barcelona：ACM，2010：269-272．