练集划分并随机分发至每一个客户端中.考虑到已有分布式框架良好的底层数据分发机制,C2CDF 在数据划分和分发部分,直接调用 Spark 相关操作.与此同时,C2CDF 中每个客户端节点都独立保存一份完整的模型参数,并使用分发的子数据集对相关的参数进行更新.因此,最终 C2CDF 可以得到与客户端节点相同数目的相互独立的模型.与传统的服务器-客户端结构类似,C2CDF 保留了调度节点,用以实现模型的全局调度和全局变量的保存,其中,全局变量包括每个客户端节点存储模型的标识、正在更新的模型标识和已经完成更新空闲等待的模型标识等.在模型预测过程中,C2CDF 利用已经学习得到的多个模型对测试集进行独立的预测,对同一测试数据的多个结果,采用投票的方法进行合并,多数表决得到的结果作为最终结果.

2.3 模型调度和参数更新机制

通过上述可知,C2CDF 中每个客户端节点上保存的模型其学习过程是相互独立的.但由于每个节点只保存了很少一部分数据集,数据稀疏性在这类高维数据集上越发严重,直接导致每个节点上学习得到的模型有效性差.为了进一步提高模型的学习有效性,C2CDF 设计有一套完整的模型调度策略.模型调度的基本思想是:允许每个节点不但可以更新它本身存储的模型,还可以更新不属于它的其他节点的模型.即:节点在更新完一个模型后,将模型推送至它的原始保存节点,然后通过调度函数,节点选择另一个新的从未更新过的模型,将该模型从所属的节点拉取至当前节点,继续使用分发给当前节点的子数据集对该模型进行更新.

模型的调度带来了全新的挑战:首先,在下一个模型选取时,遵循何种策略来确定应选择的模型;其次,频繁的模型调度,会导致通信代价大为增加;这里的通信代价包括模型在不同节点间的传送和调用调度函数所需的消耗;最后,若选择的下一个模型仍在被其他节点更新,当前节点就必须等待,坏的调度策略会直接导致等待时延的增加.

通过前文可知,C2CDF 中客户端节点的数目和模型的数目是相等的.因此,模型的标识符通常采用其所属的节点属性来标记,如 IP 和端口.故而在模型的调度中,C2CDF 的调度会给出下一个模型所属的节点 IP 和端口信息,通过 IP 和端口得到相应的模型.当一个模型正在被更新,其他节点无法对其进行拉取和更新.为了调度方便,C2CDF 在调度节点中维护了多个全局变量:一个空闲模型列表,用以记录所有已经完成更新的空闲模型;一个已使用模型列表,用以记录每个节点更新过的历史模型.

在模型调度机制方面,我们给出两种调度策略:第 1 种调度为随机调度,其核心思想在于,当前节点从所有模型列表中随机选择一个从未更新过的模型;第 2 种调度为先到先得,指当前节点总是会选择最早空闲的且从未更新过的模型.通过对上述两种调度机制比较可知:先到先得调度能够保证所选取的模型当前空闲,能够在调度后立即拉取并进行更新,减少等待时延,但会造成更新快的节点永远快,慢的一直慢,最终的模型训练时间由最慢节点决定.相比之下,随机调度则会在一定程度上保证各个节点执行时间上的均衡,两者各有优劣.由于模型是否空闲的判断在实现上采取被动的等待循环判断,因此,模型训练时间的长短不确定性较大.在本文的实验部分,我们采取了先到先得的策略完成全部实验.

在参数更新机制方面,由于频繁的模型调度会导致通信代价增加,为了降低通信代价,我们采用每个节点上的多次迭代为一个周期的机制[28],周期结束后,调用模型调度策略选择下一个要更新模型.即,取代服务器-客户端分布式模式下的一次迭代一次更新方式.由于模型之间相互独立,因此完全可以在每个节点上更新多次后再将更新后模型推送给模型所在节点.这里,我们记该周期为 τ.与传统的服务器-客户端模式相比,节点间的通信代价降低为原来的 $1/\tau$.同时,由于特征的高维且稀疏特性,若每次模型的拉取和推送都是整个模型,则会产生高昂的通信消耗.为进一步降低通信规模,C2CDF 采取与参数服务器框架类似的策略,节点间的模型传输仅交换当前节点所需的子模型空间.子模型空间的选取,取决于拉取该模型的节点所分配的子数据集中不为零特征的个数.

2.4 C2CDF具体实现

在 C2CDF 系统(以下简称系统)的具体实现上,我们结合了现有大数据分析平台 Spark 和消息分发系统 Akka actor[29]的优势.其中,Spark 主要用于分发存放在分布式文件系统 HDFS 上的训练和测试数据,并指派具体的计算操作给集群中每个运算单元;Akka actor 则是用于设计大规模分布式并行系统底层消息分发机制的工

具.通过利用 Spark 和 Akka actor 等成熟框架,系统能够更加专注于模型并行化的调度和参数更新机制的设计.

按照系统的模型学习步骤,我们可以将整个流程分为两个阶段:环境初始化和模型调度学习.在环境初始化阶段,由于系统建立在 Spark 和 Akka actor 的基础上,因此,首先必须初始化集群 Spark 环境及各个计算单元的 Akka actor 角色;然后,利用 Spark 将 HDFS 上数据集读入并分发至每个节点上;进一步,注册计算单元并初始化每个单元上的模型参数;最后,调度器发送模型学习消息给每个计算单元,开始进入模型调度学习阶段.具体环境初始化阶段伪代码见算法 1.

算法 1. 环境初始化.

输入:数据集 D,计算单元数 S,分解隐向量维度 k 和正则化参数$\lambda^0,\lambda^w,\lambda^v$;

1. Spark 环境和 Akka actor 角色初始化
2. Spark 读入 HDFS 数据集 D
3. Spark 分发数据集 D 至 S 个计算单元
4. 计算单元发送“注册”消息给调度器
5. 根据第 1.3 节的参数分布,初始化计算单元的模型参数 w^0,w,v
6. 调度器发送模型“学习”消息给所有计算单元

在模型调度学习阶段,集群节点可分为两类角色:计算单元和调度器.计算单元作为系统中最小的执行单元,主要负责拉取调度器分配的模型至本单元,然后对调度器分配的模型进行学习更新,最后将更新结果推送至模型初始所在单元,周而复始.调度器在集群中只有一个,主要用于维护全局参数和根据既定的模型调度策略将模型分配给请求方计算单元.具体的计算单元和调度器执行伪代码分别见算法 2 和算法 3.

算法 2. 计算单元.

输入:最大迭代数 T,内部最大迭代数τ;

```
1.   持续接收消息
2.   if 消息类型为模型“学习” then
3.      i=0
4.      While i<T do
5.         if i%τ=0 then
6.            发送“调度”消息给调度器,返回需要更新模型所在 IP 和端口
7.            “拉取”相应的局部模型参数从上述 IP 和端口对应的计算单元
8.         end in
9.         使用第 1.3 节对应的公式更新模型参数 w^0,w,v
10.        i++
11.        if i%τ=0 then
12.           “推送”更新完成的模型参数至其对应的原始计算单元
13.           发送“完成”消息给调度器,附带更新完成模型 ID
14.        end if
15.     end while
16.  else if 消息类型是“拉取” then
17.     返回对应的局部模型参数给请求计算单元
18.  else if 消息类型是“推送” then
19.     根据发送方提供的局部模型,更新属于当前计算单元的模型
20.  end if
```

算法 3. 调度器.

```
1.  持续接收消息
2.  初始化注册计算单元列表
3.  初始化空闲模型列表
4.  初始化每个计算单元的已更新模型列表
5.  if 消息类型是“注册” then
6.      添加发送方(计算单元)至注册计算单元列表
7.  else if 消息类型是“完成” then
8.      添加发送方(计算单元)携带的模型 ID 至空闲模型列表
9.      添加发送方(计算单元)携带的模型 ID 至该计算单元的已更新模型列表
10. else if 消息类型是“调度” then
11.     根据调度策略选择空闲模型 m
12.     while m 已在发送方(计算单元)的已更新模型列表中 do
13.         根据调度策略重新选择空闲模型 m
14.     end while
15.     while m 不在空闲模型列表 do
16.         等待
17.         判断 m 是否在空闲模型列表中
18.     end while
19.     返回选定的模型给调度请求方(计算单元)
20. end if
```

3 实 验

在本节,我们使用提出的 C2CDF 下的因式分解模型在真实数据集上作分类预测.

3.1 实验设置

为了说明新提出框架 C2CDF 下因式分解模型的性能,我们选取了 3 种不同规模和不同场景下的数据集分别进行实验——Epinions,Criteo,Cvotype.

- Epinions 在本实验中是一个符号网络正负关系预测数据集,通过转换原数据集中用户之间信任/不信任关系的数据处理得到的.整个数据集包含用户 131 828 个,用户之间的关系 841 372 个.每个数据实例包含 179 918 个特征维度,整个数据集实例数与关系数相等;
- Criteo 是用于广告点击预测的数据集,其目标是预测一个广告是否会被点击.每个数据实例包含 46 811 个特征维度,其中,类别相关信息 26 个,整数型相关信息 13 个.整个数据集共有 22 917 906 个数据实例;
- Cvotype 是一个从地图变量预测森林覆盖类型的数据集,每个数据实例包含 54 个维度,数据集共 581 012 个实例.

为了更直观地对 C2CDF 框架及其框架下因子分解机模型在 SGLD 方法上的优化有效性进行说明,我们共选择 4 种方法进行不同层面的对比.

- FM:因子分解机模型使用 SGLD 为优化方法,在单机环境下的实现;
- MSFM:因子分解机模型使用 SGLD 为优化方法,在传统服务器-客户端分布式架构下的实现;
- C2CFM-SGD:因子分解机模型使用随机梯度下降为优化方法,在 C2CDF 模式下的实现;
- C2CFM:本文所提出的新模型,因子分解机模型使用 SGLD 为优化方法,在 C2CDF 模式下的实现.

为了定量地评估所提出 C2CFM 方法,我们考虑使用如下的性能评估方法.

- 准确性评估:使用准确性指标来比较上述 4 种方法.准确性的定义为 $TP+TN/P+N$.其中,TP 表示正例实

例且测试结果也为正例的实例数目,*TN* 表示负例实例且测试结果也为负例的实例数目,*P* 表示为测试集正例的总数,*N* 表示为测试集负例的总数;

- 有效性评估:使用模型的训练执行时间来评估 C2CFM 与其他方法的加速比;
- 扩展性评估:使用模型的训练执行时间来评估 C2CFM 与其他方法的扩展性对比,包括同一数据的不同模型维度和不同数据集大小两类测试.

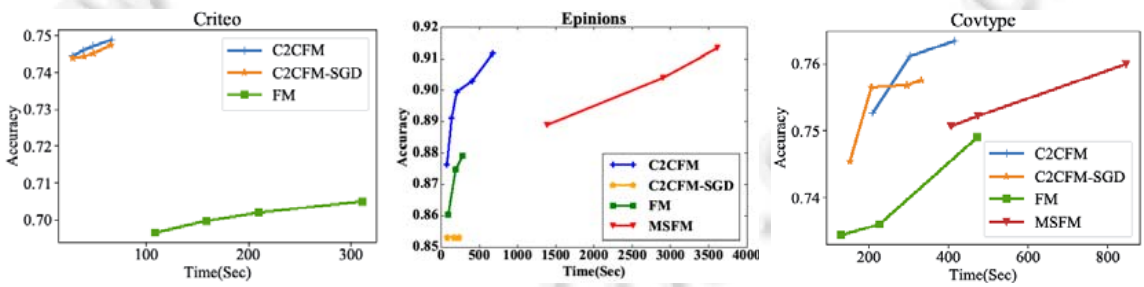
在硬件执行环境方面,MSFM、C2CFM-SGD 和 C2CFM 均执行在由 15 台相同配置的机器组成的分布式集群中,每台机器的配置为 Linux Centos 6.5,4 核 CPU,2.0GHz 频率,16G 内存及 500G 硬盘.FM 则单独执行在其中一台机器上.若无特殊说明,在 MSFM、C2CFM-SGD 和 C2CFM 算法实现方面,我们默认设置客户端节点的数量为 29(客户端节点的最小单位可为一个 CPU 核).对于 C2CFM 在每个节点上迭代周期 τ,我们默认设置为 100.

3.2 准确性和有效性比较

在上述 3 个数据集中,我们对 4 种方法的准确性和有效性进行对比.对于每一种方法,我们最多执行 Cvotype 和 Epinions 迭代 600 次,Criteo 为 300 次.在最大迭代的 1/3,2/3 和 1 倍处(若标记多于 3 处,增 1/2 倍处;若标记不等,则前 3 处分别对应 1/3,2/3 和 1 倍处)记录每个方法在相同测试集上的准确率和相应的训练执行时间.最终生成横坐标为执行时间、纵坐标为准确率的结果,如图 3 所示.从结果我们可以看出:

- 首先,所有方法的准确性随着执行时间的增加能够获得更好的提升;
- 其次,本文提出的 C2CFM 较其他 3 种方法在较少的执行次数下也能获得更好的准确率.造成该结果的主要原因可能在于,C2CFM 减少了节点间的通信且模型的多数表决机制使得其准确率更高;
- 最后,在所有 3 个数据集中,C2CFM-SGD 的准确性都略低于 C2CFM.这种结果表示我们提出的因子分解机模型高斯噪音的加入是有效的.

需要注意的是:在数据集 Criteo 中没有 MSFM 方法的结果,这是由于 MSFM 在正常模型维度大小时就已经失败了,而随着模型维度的减小,MSFM 逐渐正常工作,详见后文图 5(b).



(a) Criteo (b) Epinions (c) Cvotype

Fig.3 Accuracy and runtime comparison in all methods

图 3 所有方法的准确性及效率性能比较

在上述实现设置条件下,我们进一步对大数据集 Criteo 下 4 种方法的最终准确率和训练执行时间进行对比,结果如图 4 所示.在该图中我们可以看出:C2CFM 在获得更高准确率的同时,相比 FM 可以获得 3.3 倍~4.7 倍的加速比,相比 MSFM 可以获得 2.3 倍~3.3 倍的加速比.

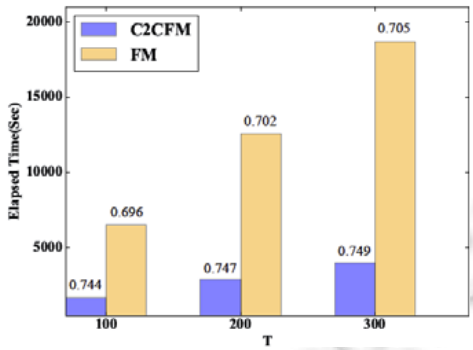


Fig.4 Accuracy and runtime comparison under the different iterations in Criteo dataset

图 4 Criteo 数据集下不同迭代次数下算法准确率和效率性能比较

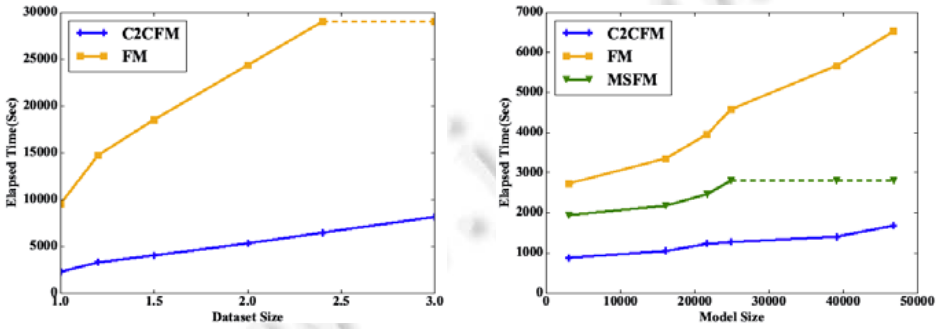
3.3 扩展性比较

在本环节实验中,Cvotype 和 Epinions 数据集由于太小,因此我们通过改变大数据集 Criteo 的模型维度和数据集大小来证明提出方法 C2CFM 是否具有良好的可扩展性.C2CFM-SGD 方法由于是用来说明 SGLD 优化的有效性,与 C2CDF 的扩展性无关,因此不予考虑.

在图 5(a)中,我们增加数据集的大小至其原始大小的 1.2 倍、1.5 倍、2.0 倍、2.4 倍和 3.0 倍,然后检验不同方法的执行时间.在本次实验中,最大迭代次数均为 150 次.从结果我们可以看出:C2CFM 随着数据集大小的增加,其运行时间增加较其他两种方法缓慢;当数据集大于 2.4 倍时,FM 算法由于单机内存无法放下整个数据集而执行失败.此外,MSFM 在原始 Criteo 数据集上就已经执行失败,其原因可能是随着数据集规模的增加,分发至每个节点上的数据量增加,导致每个节点上的模型规模增大,在集中模型交互时受带宽压力的影响而失败.

在图 5(b)中,我们减少模型的维度规模,然后检验不同方法在每个模型大小时的训练执行时间.可以看出:C2CFM 随着模型维度的增加,其运行时间的增加较其他两种方法缓慢.此外,当模型规模增至原始规模的 83%时,MSFM 就已经执行失败了.其原因可能在于模型规模的增加导致服务器和客户端节点间所需的传输带宽增加,超过最大值而导致失败.

通过对图 5 的结果进行总结可以看出:C2CDF 相比单机架构和传统服务器-客户端模式,有良好的扩展性.



(a) 不同数据规模下算法扩展性 (b) 不同模型维度下算法扩展性

Fig.5 Scalability of different methods in Criteo dataset

图 5 Criteo 数据集下算法扩展性

4 相关工作

4.1 符号网络正负关系预测

在符号网络关系的正负预测领域,研究人员从不同角度设计了多种算法.从大的思路来看,目前所有的符号

链接预测模型可分为两类[1]:基于矩阵的[1−5]和基于分类的[6−11].基于矩阵的符号链接预测算法将符号网络视为矩阵,利用信任传播模型[1]、矩阵分解[2,4,5]或矩阵填充[3]进行符号预测.基于分类的方法将符号网络中链接的正负预测转化为二值分类问题,构建特征集,利用分类算法进行符号预测.该类模型的重点在于如何构架特征集,根据特征集的选择不同,这类方法可继续细分为基于网络结构和基于网络上下文的算法.前者结合社会结构平衡理论或地位理论,利用符号网络的结构信息设计符号预测算法[6,7,9,10];后者利用符号网络的上下文信息,通过分类算法对网络中边的正负关系进行预测[6,11].

4.2 因子分解机模型

因子分解机是一种被广泛应用于预测、推荐等领域的模型(以下简称模型),特征工程和分解模型的结合,使得它通常能够获得较好的性能[13,14,30−32].然而模型独有的特征组织形式,导致它在大规模数据集下,传统的单机环境已无法满足其需求.因此,模型的扩展应运而生.目前,针对因子分解机模型的扩展研究主要集中在两个方面:其一,不改变单机环境的基础,针对模型底层数据的组织特征重新建立新的模型,既有效地降低了底层数据的存储,还提高了模型的计算效率[33];其二,利用分布式框架对原有的单机模型进行扩展和优化,例如基于参数服务器框架的模型实现[17,18]和基于 Spark 的模型扩展(https://github.com/blebreton/spark-FM-parallelSGD).

4.3 分布式框架

大数据时代,为了提高模型的学习效率,各种各样的分布式学习框架不断被提出.本节将对流行的分布式环境进行总结.Hadoop[27]作为一个简单易用的分布式平台,其底层数据和计算分发等实现机制对用户完全透明,通过提供简洁的编程接口用户即可实现高效的分布式数据流处理.同时,它非常适合于机器学习中模型的迭代学习.基于 Hadoop 平台实现的分布式机器学习算法包有 Mahout 等.Spark[34]可以看做是 Hadoop 的一次进化,相比 Hadoop,Spark 的优势在于它基于内存的处理框架能够有效地提升处理性能.MLlib 是 Spark 下比较流行的机器学习算法包,与 Hadoop 和 Spark 不同,GraphLab[35]和 Pregel[36]更加专注于图模型方面的分布式学习.为了进一步提升 Hadoop 和 Spark 的分布式性能,Li 正式提出了参数服务器框架.目前,在参数服务器模式下已有多篇工作[37−40].与 Hadoop 和 Spark 这种天然的基于同步机制的模型迭代学习相比,参数服务器架构具有以下优势: (1) 服务器端由多台节点共同构成,称为服务器组,全局模型分布由多个节点组成的服务器组;(2) 允许服务器上全局模型的异步更新;(3) 采取多种策略,保证节点间的通信规模尽可能小,提升模型的学习效率.

通过对上述提及的架构进行分析,不难看出,它们都可归类于服务器-客户端分布式系统.在这类环境中,通常是客户端从服务器端拉取参数进行更新后,再将增量送回服务器端更新全局模型.这种框架的天然缺陷在于参数的传递集中,从而导致服务器端的带宽压力较大,影响模型性能.

5 总结与展望

本文将因子分解机模型用于符号网络正负关系的预测.在模型中,我们首先使用 SGLD 方法来优化因子分解机模型.由于 SGLD 方法中高斯噪声分布的加入,能够减轻模型容易陷入局部最优解的问题.为了进一步将模型应用到大规模数据环境中,提出了端到端的分布式框架 C2CDF.C2CDF 一方面抛弃了传统服务器-客户端分布式中的服务器端,将原来存在于服务器端的全局模型分别放一份在各个客户端节点上,避免了集中频繁通信带来的压力和瓶颈;另一方面,设计提出了新的模型调度机制和参数更新机制,使得模型的学习更加有效,很好地利用了原有Spark的数据分区和分发机制.在最终的模型预测方面,创造性地将模型的多数表决机制引入.在3个不同规模的实际数据集上的实验结果显示:C2CDF 拥有良好的泛化性,不仅能应用在社交网络正负符号预测方面,也能作用于广告点击预测等其他领域.在上述数据集中,C2CDF 相比 FM 和 MSFM 取得了更好的性能,并获得了相较于 MSFM 有 2.3 倍~3.3 倍的加速比.未来将尝试 C2CDF 框架应用到更多的机器学习算法中去.

References:

[1] Guha R, Kumar R, Raghavan P, Tomkins A. Propagation of trust and distrust. In: Proc. of the WWW 2004. 2004. 403−412. [doi: 10.1145/988672.988727]

[2] Kunegis J, Schmidt S, Lommatzsch A, Lerner J, Luca E, Albayrak S. Spectral analysis of signed graphs for clustering, prediction and visualization. In: Proc. of the SDM 2010. 2010. 559−559. [doi: 10.1137/1.9781611972801.49]

[3] Hsieh CJ, Chiang KY, Dhillon IS. Low rank modeling of signed networks. In: Proc. of the SIGKDD 2012. 2012. 507−515. [doi: 10.1145/2339530.2339612]

[4] Kunegis J, Lommatzsch A, Bauckhage C. The slashdotzoo: Mining a social network with negative edges. In: Proc. of the WWW 2009. 2009. 741−750. [doi: 10.1145/1526709.1526809]

[5] Agrawal P, Garg VK, Narayanam R. Link label prediction in signed social networks. In: Proc. of the IJCAI 2013. 2013.

[6] Leskovec J, Huttenlocher D, Kleinberg J. Predicting positive and negative links in online social networks. In: Proc. of the WWW 2010. 2010. 641−650. [doi: 10.1145/1772690.1772756]

[7] Chiang KY, Natarajan N, Tewari A, Dhillon I. Exploiting longer cycles for link prediction in signed networks. In: Proc. of the CIKM 2011. 2011. 1157−1162. [doi: 10.1145/2063576.2063742]

[8] Yang SH, Smola AJ, Long B, Zha H, Chang Y. Friend or frenemy? Predicting signed ties in social networks. In: Proc. of the SIGIR 2012. 2012. 555−564. [doi: 10.1145/2348283.2348359]

[9] Ye J, Cheng H, Zhu Z, Chen M. Predicting positive and negative links in signed social networks by transfer learning. In: Proc. of the WWW 2013. 2013. 1477−1488. [doi: 10.1145/2488388.2488517]

[10] Dubois T, Golbeck J, Srinivasan A. Predicting trust and distrust in social networks. In: Proc. of the 3rd Int'l Conf. on Privacy, Security, Risk and Trust. 2012. 418−424. [doi: 10.1109/PASSAT/SocialCom.2011.56]

[11] Borzymek P, Sydow M. Trust and distrust prediction in social network with combined graphical and review-based attributes. In: Proc. of the Kes Int'l Conf. on Agent and Multi-Agent Systems: Technologies and Applications. 2010. 122−131. [doi: 10.1007/978-3-642-13480-7_14]

[12] Freudenthaler C, Schmidt-Thieme L, Rendle S. Bayesian factorization machines. 2011. https://wenku.baidu.com/view/5cf5080f581b6bd97f19ea23.html

[13] Loni B, Shi Y, Larson M, Hanjalic A. Cross-Domain collaborative filtering with factorization machines. In: Proc. of the ECIR 2011, 2014. 656−661. [doi: 10.1007/978-3-319-06028-6_72]

[14] Rendle S. Factorization machines. In: Proc. of the ICDM 2010. 2010. 995−1000. [doi: 10.1109/ICDM.2010.127]

[15] Tsai MF, Wang CJ, Lin ZL. Social influencer analysis with factorization machines. In: Proc. of the WebSci 2015. 2015. 50−50. [doi: 10.1145/2786451.2786490]

[16] Wang S, Du C, Zhao K, Li C, Li Y, Zheng Y, Wang Z, Chen H. Random partition factorization machines for context-aware recommendations. In: Proc. of the WAIM 2016. 2016. 219−230. [doi: 10.1007/978-3-319-39937-9_17]

[17] Li M, Liu Z, Smola AJ, Wang YX. Difacto: Distributed factorization machines. In: Proc. of the WSDM 2016. 2016. 377−386. [doi: 10.1145/2835776.2835781]

[18] Zhong E, Shi Y, Liu N, Rajan S. Scaling factorization machines with parameter server. In: Proc. of the CIKM 2016. 2016. 1583−1592. [doi: 10.1145/2983323.2983364]

[19] Welling M, The YW. Bayesian learning via stochastic gradient langevin dynamics. In: Proc. of the ICML 2011. 2011. 681−688.

[20] He Q, Xin J. Hybrid deterministic-stochastic gradient langevin dynamics for bayesian learning. Communications in Information and Systems, 2012,12(3):221−232. [doi: 10.4310/CIS.2012.v12.n3.a3]

[21] Ahn S, Korattikara A, Liu N, Rajan S, Welling M. Large-Scale distributed bayesian matrix factorization using stochastic gradient MCMC. In: Proc. of the SIGKDD 2015. 2015. 9−18. [doi: 10.1145/2783258.2783373]

[22] Koren Y. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In: Proc. of the SIGKDD 2008. 2008. 426−434. [doi: 10.1145/1401890.1401944]

[23] Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems. Computer, 2009,42(8):30−37. [doi: 10.1109/MC.2009.263]

[24] Rendle S, Schmidt-Thieme L. Pairwise interaction tensor factorization for personalized tag recommendation. In: Proc. of the WSDM 2010. 2010. 81−90. [doi: 10.1145/1718487.1718498]

[25] Rendle S, Freudenthaler C, Schmidt-Thieme L. Factorizing personalized Markov chains for next-basket recommendation. In: Proc. of the WWW 2010. 2010. 811−820. [doi: 10.1145/1772690.1772773]

[26] Ahn S, Shahbaba B, Welling M. Distributed stochastic gradient mcmc. In: Proc. of the ICML 2014. 2014. 1044−1052.

[27] Sun H, Wang W, Shi Z. Parallel factorization machine recommended algorithm based on mapreduce. In: Proc. of the SKG 2014. 2014. 120−123. [doi: 10.1109/SKG.2014.26]

[28] Li M, Andersen DG, Smola A, Yu K. Communication efficient distributed machine learning with the parameter server. In: Proc. of the NIPS 2014. 2014. 19−27.

[29] Hewitt C, Bishop P, Steiger R. A universal modular actor formalism for artificial intelligence. In: Proc. of the IJCAI'73. 1973. 235−245.

[30] Hong L, Doumith AS, Davison BD. Co-Factorization machines: Modeling user interests and predicting individual decisions in Twitter. In: Proc. of the WSDM 2013. 2013. 557−566. [doi: 10.1145/2433396.2433467]

[31] Rendle S. Social network and click-through prediction with factorization machines. In: Proc. of the KDD 2012. 2012.

[32] Rendle S, Gantner Z, Freudenthaler C, Schmidt-Thieme L. Fast context-aware recommendations with factorization machines. In: Proc. of the SIGIR 2011. 2011. 635−644. [doi: 10.1145/2009916.2010002]

[33] Rendle S. Scaling factorization machines to relational data. Proc. of the VLDB Endowment, 2013,6(5):337−348. [doi: 10.14778/2535573.2488340]

[34] Zaharia M, Chowdhury M, Franklin MJ, Shenker S, Stoica I. Spark: Cluster computing with working sets. In: Proc. of the HotCloud 2010. 2010. 10−10.

[35] Low Y, Bickson D, Gonzalez J, Guestrin C, Kyrola A, Hellerstein J. Distributed graphLab: A framework for machine learning and data mining in the cloud. Proc. of the VLDB Endowment, 2012. 716−727. [doi: 10.14778/2212351.2212354]

[36] Malewicz G, Austern MH, Bik AJ, Dehnert JC, Horn I, Leiser N, Czajkowski G. Pregel: A system for large-scale graph processing. In: Proc. of the SIGMOD 2010. 2010. 135−146. [doi: 10.1145/1582716.1582723]

[37] Xing EP, Ho Q, Dai W, Kim JK, Wei J, Lee S, Zheng X, Xie P, Kumar A, Yu Y. Petuum: A new platform for distributed machine learning on big data. In: Proc. of the SIGKDD 2015. 2015. 1335−1344. [doi: 10.1109/TBDATA.2015.2472014]

[38] Dean J, Corrado GS, Monga R, Chen K, Devin M, Le QV, Mao MZ, Ranzato M, Senior A, Tucker P. Large scale distributed deep networks. In: Proc. of the NIPS 2012. 2012. 1223−1231.

[39] Jiang J, Yu L, Jiang J, Liu Y, Cui B. Angel: A new large-scale machine learning system. National Science Review, 2017. [doi: 10.1093/nsr/nwx018]

[40] Li M, Andersen DG, Park JW, Smola AJ, Ahmed A, Josifovski V, Long J, Shekita EJ, Su BY. Scaling distributed machine learning with the parameter server. In: Proc. of the OSDI 2014. 2014. 583−598. [doi: 10.1145/2640087.2644155]

[41] Lan MW, Li CP, Wang SQ, Zhao KK, Lin ZX, Zou BY, Chen H. Survey of sign prediction algorithm in signed social networks. Journal of Computer Research and Development, 2015,52(2):410−422 (in Chinese with English abstract). [doi: 10.7544/issn1000-1239.2015.20140210]

附中文参考文献:

[41] 蓝梦微,李翠平,王绍卿,赵衍衍,林志侠,邹本友,陈红.符号社会网络中正负关系预测算法研究综述.计算机研究与发展,2015,52(2):410−422. [doi: 10.7544/ issn1000-1239.2015.20140210]

赵衍衍(1991－),男,陕西渭南人,博士生,主要研究领域为推荐系统,大数据分析.

李翠平(1971－),女,博士,教授,博士生导师,CCF 杰出会员,主要研究领域为社交网络分析,社会推荐,大数据分析及挖掘.

张静(1984－),女,博士,讲师,CCF 专业会员,主要研究领域为数据挖掘,社会网络挖掘.

陈红(1965－),女,博士,教授,博士生导师,CCF 杰出会员,主要研究领域为数据库技术,新硬件平台下的高性能计算.

张良富(1991－),男,博士生,主要研究领域为机器学习,数据挖掘.