一种分布式隐私保护的密度聚类算法

吉根林,姚 瑶
(南京师范大学 数学与计算机科学学院,江苏 南京 210097)

摘 要:对基于密度的分布式聚类算法 DBDC进行改进,提出了一种基于密度的分布式隐私保护聚类算法 DBPPDC. 在由局部模型确定全局模型时,通过相关安全协议有效地保护了局部模型,同时不影响全局聚类. 在利用全局模型更新局部模型时,通过改进算法、应用安全协议保护隐私信息,最终使各站点分布的数据能够安全聚类. 理论分析和实验结果表明,DBPPDC算法是有效的.

关键词:隐私保护;分布式聚类; DBDC; DBPPDC
中图分类号: TP311. 1 **文献标识码**: A **文章编号**: 1673-4785(2009)02-0137-05

Density-based privacy preserving distributed clustering algorithm

JI Gen-lin, YAO Yao
(School of Mathematics and Computer Science, Nanjing Normal University, Nanjing 210097, China)

Abstract: A density-based privacy preserving distributed clustering algorithm (DBPPDC) was proposed following the improvements to the density-based distributed clustering DBDC algorithm. When a global model is determined from a local model, (DBPPDC) effectively protects the local model without obstructing global clustering. On the contrary, when the local model is updated with the global model, DBPPDC makes all the data in local sites cluster safely by improving the previous algorithm and appling a secure protocol. Experimental results showed that DBPPDC is effective and efficient.

Keywords: privacy preserving; distributed clustering; DBDC; DBPPDC

分布式聚类算法[1-4]在聚类过程中将本站点有关真实数据传送给其他站点,从而导致信息泄露. 在实际分布式聚类应用中,有时候需要保护本站点的真实信息不被传送给其他站点,即需要进行隐私保护,为此,需要研究基于隐私保护的分布式聚类算法. 聚类过程中的隐私保护方法可大致分为数据扰乱和安全多方计算 2种. 基于数据扰乱的隐私保护聚类思想是通过转换数据使得真实的敏感数据不为人知,然后再进行聚类分析. 而基于安全多方计算的隐私保护聚类主要通过构造安全多方协议,使得一组站点在仅仅拥有自己私有信息的情况下能最终获知全局聚类信息. 后者主要应用于分布式聚类分析.

针对水平划分的分布式数据库,文献[5-6]提出基于隐私保护的分布式聚类算法,本文同样在水平划分的分布式数据库环境下,对基于密度的分布式聚类算法[7] (density based distributed clustering, DBDC)进行改进,提出了一种基于密度的分布式隐私保护聚类算法 (density based privacy preserving distributed clustering, DBPPDC). 在由局部模型确定全局模型时,通过相关安全协议有效地保护局部模型,同时不影响全局聚类. 在利用全局模型更新局部模型时,通过改进算法、应用安全协议保护隐私信息,最终使各站点分布的数据能够安全聚类.

1 问题描述

1.1 相关定义

定义 1 全局数据集. 分布式系统中有 m 个站点,各站点相应的 d 维局部数据集分别为 $\{DB_1, DB_2, \cdots, DB_m\}$,各局部数据集的大小分别为 $N_1, N_2, \cdots, N_m$, $DB = \bigcup_{i=1}^{m} DB$,称为全局数据集.

收稿日期: 2008-12-16.
基金项目:国家自然科学基金资助项目 (40771163).
通信作者:姚 瑶. E-mail: cindy_yaoyao@hotmail.com.

定义 2 核心点. 给定邻域半径 Eps和最小密度 MinPts,若对象 q的 Eps邻域 $N_{Eps}(q)$包含的对象个数 $|N_{Eps}(q)| \geq$MinPts,则称 q为核心点. 在上述条件下对全局数据集执行 DBSCAN聚类,可划分为 s个聚簇 $w_1, \cdots, w_s$,各类的核心点集分别为 CorePointSet$_1$, CorePointSet$_2$, …, CorePointSet$_s$.

定义 3 特殊核心点集. 类 i的特殊核心点集 Special_CorePointSet$_i$是该类核心点集 CorePointSet$_i$的一个子集 $(1 \leq i \leq s)$,满足以下条件:

1) pecial_CorePointSet$_i \subseteq$ CorePointSet$_i$;

2) $\forall d_h, d_j \in$Special_CorePointSet$_i$,若 $d_h \neq d_j$,则 $d_h \notin N_{Eps}(d_j)$;

3) $d_B \in$CorePointSet$_i$, $\exists d_A \in$Special_CorePointSet$_i$且 $d_B \in N_{Eps}(d_A)$.

1.2 分布式聚类算法 DBDC

分布式聚类算法 DBDC是基于密度的聚类算法(DBSCAN)在分布式环境下的扩展. 该算法设分布式系统中有 m个站点,从中任意选定一个站点 P_m为主站点,其余 $m-1$个站点为从站点. 它按 2步进行聚类:局部聚类和全局聚类. 首先,各站点在给定邻域半径 Eps和最小密度 MinPts的前提下分别执行 DBSCAN进行局部聚类,得到局部核心点集;然后从中选择能够反映数据分布特征的特殊核心点集作为局部代表对象,同时,将这些特殊核心点及其邻域半径发送到主站点;主站点对所有的特殊核心点再次执行 DBSCAN聚类得到全局聚类结果,并将其广播到各从站点,各站点根据全局信息重新标识局部数据集. 算法描述如算法 1所示.

算法 1 分布式聚类算法 DBDC

输入:局部数据集 $\{DB_1, DB_2, \cdots, DB_m\}$, Eps, MinPts

输出:聚类结果.

步骤:

slave site P_i: $(1 \leq i \leq m-1)$

{ CorePointSet$_{i1}$, …, CorePointSet$_{is}$ } = DBSCAN(DB$_i$, Eps, MinPts); /* 执行 DBSCAN得到 s个核心点集 */

for j=1 to s do

for each $d_A \in$CorePointSet$_{ij}$ do

if({ d_B | $d_B \in$CorePointSet$_{ij}$, $d_B \neq d_A$, $d_B \in N_{Eps}(d_A)$ }! =Null) /* 若某核心点集中的某一核心对象 d_B处于另一核心对象 d_A的 Eps领域内 */

{Add({ d_A, Eps+max{ dist(d_A, d_B) }})} to Special_CorePointSet$_{ij}$;

/* 添加核心对象 d_A到该类的特殊核心点集中 */

Delete(d_A, { d_B }) from CorePointSet$_{ij}$; /* 删除核心点 d_A,及其 Eps领域内的核心点{ d_B } */

}

Special_CorePointSet$_i$ = merge(Special_CorePointSet$_{i1}$, …, Special_CorePointSet$_{is}$); /* 合并特殊核心点集 */

send(Special_CorePointSet$_i$) to master site P_m; /* 向主站点传送局部代表信息 */

receive(Global_CorePointSet); /* 接收全局模型 */

relabel(DB$_i$, Global_CorePointSet); /* 根据全局特殊核心点重新标识所有对象 */

master site P_m:

{ CorePointSet$_{m1}$, …, CorePointSet$_{ms}$ } = DBSCAN(DB$_m$, Eps, MinPts); /* 执行 DBSCAN得到 s个核心点集 */

for j=1 to s do

for each $d_A \in$CorePointSet$_{mj}$ do

if({ d_B | $d_B \in$ CorePointSet$_{mj}$, $d_B \neq d_A$, $d_B \in N_{Eps}(d_A)$ }! =Null)

/* 若某核心点集中的某一核心对象 d_B处于另一核心对象 d_A的 Eps领域内 */

{Add({ d_A, Eps+max{ dist(d_A, d_B) }})} to Special_CorePointSet$_{mj}$; /* 添加核心对象 d_A到特殊核心点集中 */

Delete(d_A, { d_B }) from CorePointSet$_{mj}$; /* 删除核心点 d_A,及其 Eps领域内的核心点{ d_B } */

}

receive(Special_CorePointSet$_i$); /* 主站点接收从站点 i的代表信息,得到全局代表信息 */

Special_CorePointSet = merge(Special_CorePointSet$_1$, …, Special_CorePointSet$_p$);

/* 合并各站点的特殊核心点集 */

Global_CorePointSet = DBSCAN(Special_CorePointSet, 2Eps, MinPts);

/* 主站点对所有特殊核心点再次执行 DBSCAN得到全局模型 */

broadcast(Global_CorePointSet); /* 向从站点广播全局模型 */

relabel(DB$_m$, Global_CorePointSet); /* 根据全局模型重新标识所有对象 */

2　分布式隐私保护聚类算法 DBPPDC

2.1　相关协议

协议:A站点有私有输入$(X_{A1}, X_{A2}, \cdots, X_{An})$,$B$站点有私有输入$(Y_{B1}, Y_{B2}, \cdots, Y_{Bn})$,要求判断$\{(X_{A1}-Y_{B1})^2+(X_{A2}-Y_{B2})^2+\cdots+(X_{An}-Y_{Bn})^2\}\leq eps^2$是否成立.但是,$A$站点不能知道$(Y_{B1}, Y_{B2}, \cdots, Y_{Bn})$的值,$B$站点也不能知道$(X_{A1}, X_{A2}, \cdots, X_{An})$的值.为了能够进行安全计算,需要 1个第三方站点 TTP.具体步骤如下:

1)A站点的工作

①产生一个加密向量 rand = $(rand_1, rand_2, \cdots, rand_n)$,发送 rand给站点 B.

②计算 $(T_{A1}, T_{A2}, \cdots, T_{An}) = \{(X_{A1}+rand_1), (X_{A2}+rand_2), \cdots, (X_{An}+rand_n)\}$,发送 $(T_{A1}, T_{A2}, \cdots, T_{An})$给 TTP.

2)B站点的工作

①接收站点 A发送来的 rand

②计算 $(T_{B1}, T_{B2}, \cdots, T_{Bn}) = \{(Y_{B1}+rand_1), (Y_{B2}+rand_2), \cdots, (Y_{Bn}+rand_n)\}$,发送 $(T_{B1}, T_{B2}, \cdots, T_{Bn})$给 TTP.

3) TTP的工作

①计算 res = $\{(T_{A1}-T_{B1})^2+(T_{A2}-T_{B2})^2+\cdots+(T_{An}-T_{Bn})^2\}$.

②判断 res$\leq eps^2$是否成立,广播给 A、B.

上述协议是基于两方安全计算的,在多方情况下,只需要将 A的加密向量广播给从站点,从站点做 B方工作后,发送给 TTP就实现了多方安全计算.

2.2　算法思想与描述

DBDC分为 2步:局部聚类和全局聚类.局部聚类可在各站点独立完成,而全局聚类需要各站点中的特殊核心点聚集到一起进行聚类,这样会泄露各站点特殊核心点的私有信息.通过安全协议,可使各站点在不泄露特殊核心点私有信息的前提下,在第三方 TTP完成全局聚类;得到全局模型. DBDC在得到全局模型后,广播给各站点,使各站点根据全局模型进行聚类.但全局模型包含各站点的相关信息,广播后每个站点都将或多或少知道其余各站点的信息.本文考虑的方法是将全局模型按站点进行分解,分解后的部分全局模型发送给相应站点,各站点根据部分全局模型进行聚类,但得到的聚类结果是不完整的,聚类后的一些噪音和未被分类的对象很可能在完整的全局模型中能归于某一类.本文的解决方法是将这些对象和其他站点的部分全局模型再次根据安全协议在第三方 TTP上进行安全聚类,完成后将这些对象返回到相应站点,重新标识.此时各站点所有对象聚类完毕.这就对 DBDC进行改进达到隐私保护目的的 DBPPDC算法思想.算法描述如算法 2所示.

算法 2　分布式隐私保护聚类算法 DBPPDC

输入:局部数据集 $\{DB_1, DB_2, \cdots, DB_m\}$, Eps, MinPts

输出:聚类结果.

步骤:

```
master site P_m:
    rand = randVector();
/*随机产生加密向量*/
    broadcast (rand) to slave site P_i; (1≤i≤p-1) /*向从站点广播加密向量*/
slave site P_i: (1≤i≤m-1)
    rand = rceive(rand);
    /*接收加密向量*/
    site P_i: (1≤i≤m)
        {CorePointSet_i1, ⋯, CorePointSet_is} = DBSCAN(DB_i, Eps, MinPts); /*执行 DBSCAN得到 s个核心点集*/
    for j=1 to s do
    for each d_A ∈CorePointSet_ij do
    if({d_B | d_B ∈CorePointSet_ij, d_B ≠d_A, d_A ∈N_Eps(d_A)}!=Null)
        /*若某核心点集中的某一核心对象 d_B处于另一核心对象 d_A的 Eps领域内*/
    {Add({d_A, Eps+max{dist(d_A, d_B)}}) to Special_CorePointSet_ij;
        /*添加核心对象 dA到该类的特殊核心点集中*/
    Delete(d_A, {d_B}) from CorePointSet_ij}; /*删除核心点 d_A,及其 Eps领域内的核心点{d_B}*/
    Special_CorePointSet_i = merge(Special_CorePointSet_i1, ⋯, Special_CorePointSet_is); /*合并特殊核心点集*/
    Special_CorePointSet'_i = encrypt(Special_CorePointSeti, rand); /*加密特殊核心点*/
    sent(Special_CorePointSet'_i) to TTP; /*发送伪装后的特殊核心点集给 TTP方*/
    receive(Special_CorePointSet'_i); /*接收 TTP方更新后的特殊核心点集*/
    Special_CorePointSet_i = decryption(Special_CorePointSet'_i, rand); /*对特殊核心点集解密*/
```

relable (DB$_i$, Special_CorePointSet$_i$); /*根据特殊核心点集重新对 DB$_i$聚类 */

noise$_i$ = getnoise(DB$_i$); /*找出聚类后 DB$_i$中的噪音和未分类的对象 noise$_i$ */

noise$_i'$ = encrypt(noise$_i$, rand);

/*加密 noise$_i$ */

sent(noise$_i'$) to TTP;

/*发送 noise给 TTP */

receive(noise$_i'$);

decryption (noise$_i'$, rand);

site TTP:

rceive(Special_CorePointSet$_i'$);

/*接收各站点加密后的特殊核心点集 */

Special_CorePointSet′ = merge (Special_CorePointSet$_1'$, …, Special_CorePointSet$_p'$); /*合并特殊核心点集 */

Global_CorePointSet = DBSCAN (Special_CorePointSet′, 2Eps, MinPts);

/*主站点对所有特殊核心点再次执行 DBSCAN得到全局模型 */

update(Special_CorePointSet$_i'$);

/*根据全局模型更新各站点的特殊核心点集信息 */

sent(Special_CorePointSet$_i'$) to site P_i;

/*发送更新后的特殊核心点集信息给相应站点 */

rceive(noise$_i'$);

/*接收各站点加密后的噪音和未分类的对象 */

relabel (noise$_i'$, Global_CorePointSet); /*根据全局特殊核心点集重新对 noise$_i'$聚类 */

sent ((noise$_i'$) to site P_i; /*发送更新后的 noise'$_i'$给相应站点 P_i */

3 实验结果与性能分析

为了研究算法 DBPPDC的性能,使用 3台微机构成 100 MB的局域网,其中一台作为第三方 TTP.微机配置为 Intel Pentium Ⅳ 2.93 GHz/512 MB,开发环境为 JBuilder 2006 Enterprise,利用 Java实现了 DBPPDC、DBDC和 DBSCAN.实验数据源如表 1所示,其中, Iris[8]是植物样本数据库, KDD-CUP-99 800和 KDD-CUP-99 8000是从 KDD-CUP-99[9]中分别随机抽取 800个记录和 8 000个记录构成的数据库,实验中选择了其中的 34个连续值属性维.集中式聚类算法 DBSCAN运行在一台 PC机上,采用表 1所列测试数据集.分布式聚类算法 DBPPDC、DBDC运行在处于同一局域网的 3台 PC机上,表 1所示的每个数据集被随机分成 2个不相交的子数据集,存放于非 TTP的站点.

表 1 实验数据集

Table 1 Experiment data sets

数据集	对象个数	属性维数
Iris	150	4
KDD-Cup-99 800	800	34
KDD-Cup-99 8000	8 000	34

3.1 聚类精度

Modha和 Spangler[10]利用了数据的分类信息来评价聚类结果的好坏,即当数据有分类信息时,可认为该分类信息在一定程度上表达了数据的一些内部分布特性.如果该分类信息没被聚类算法利用,则可以用它来评价聚类性能,其度量标准 Micro-precision定义如下:

$$\text{Micro-precision} = \frac{1}{n}\sum_{i=1}^{k}\alpha_i.$$

式中: n为数据集样本总数, k为聚类的类数, α_i为聚类的类 i与已知数据集类别对应后,类 i中被正确归为相应类别的样本个数. Micro-precision的值越大,表示在该数据集上聚类效果越好.这种度量方法适合聚类时产生固定类数的算法,如 K-means等,因此实验中采用该标准对各算法的精度进行比较,实验结果取 10次实验的平均值,聚类精度如图 1所示.实验结果表明 DBPPDC与 DBDC在聚类精度上是相当的,经过隐私保护进行的分布式聚类并没有改变聚类的结果.由于算法 DBPPDC与 DBDC在全局聚类时采用 2×Eps进行邻域查询,会导致了一些类间的错误合并,所以精度比 DBSCAN稍低.

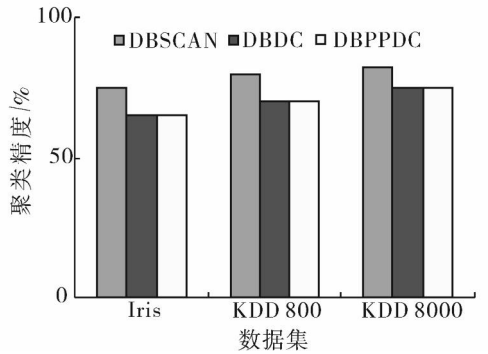

图 1 聚类算法精度比较

Fig 1 Comparison of the accuracy of clusters

3.2 聚类效率

算法的执行时间如图 2所示.其结果取 10次实验平均值.分布式隐私保护聚类算法 DBPPDC和分布式聚类算法 DBDC时间代价分为 2部分: 1)站点间通信代价, 2)站点内计算代价.从图 2可以看出,当测试数据集较小时 (如: Iris),分布式算法 DBPP-

DC与 DBDC的执行时间多于集中式 DBSCAN聚类计算时间,这是由于此时站点间的通信代价大于计算代价.因此,当数据源较小时不宜使用分布式聚类算法.但随着数据集规模不断变大(如: KDD-Cup-99子集 1,子集 2),DBPPDC与 DBDC的时间代价增长速度明显低于 DBSCAN.同时,由于在 DBPPDC算法过程中运用协议,增加了第三方的通信,相比较 DBDC通信代价增加,执行时间稍长.

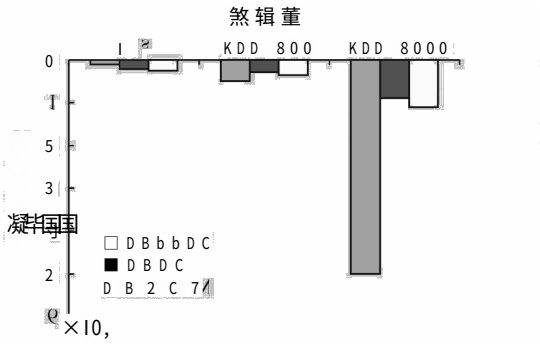

图 2　聚类算法效率比较

Fig 2　Comparison of the efficiency of algorithms

3.3　安全性分析

DBDC算法分为 2步:局部聚类和全局聚类.在局部聚类过程中,各站点进行聚类确定局部模型,此时各站点不进行相互通信,所以不涉及到隐私暴露问题.在全局聚类过程中又可分为 2步:1)根据所有局部模型确定全局模型;2)根据全局模型更新局部模型.这 2步都需要进行站点间的通信,所以需要保护各站点的私有信息.在确定全局模型时,通过相关安全协议有效地保护局部模型,同时不影响全局聚类.在更新局部模型时,是将全局模型按站点进行分解,分解后的部分全局模型发送给相应站点,各站点根据部分全局模型进行聚类,也没有泄露其余站点在全局模型的信息.聚类后的一些噪音和未被分类的对象与其他站点的部分全局模型再次根据协议在第三方 TTP上进行安全聚类,完成后将这些对象返回到相应站点.这样同时保护了本站点的对象和其余站点的部分全局模型.通过对 DBDC算法各个环节的保护,保证了整个算法的安全性.

4　结束语

本文对分布式聚类算法 DBDC改进,应用相关安全协议,提出了一种基于密度的分布式隐私保护聚类算法 DBPPDC,有效保护了算法过程中的私有信息.同时,与 DBDC相比,算法 DBPPDC没有降低算法聚类精度.在数据挖掘过程中增加隐私保护技术,使其在发现知识的同时,又保护了数据安全,这是一项非常重要的研究工作.

参考文献:

[1] KANTABUTRA S, COUCH A L. Parallel k-means clustering algorithm on Nows[J]. NECTEC Technical Journal, 2000, 1(6): 243-247.

[2] PRODD H, LAWRENCE H. Scalable clustering: a distributed approach[C]//Proc IEEE Int'l Conf Fuzzy Systems. Budapest, Hungary: ETATS-UNIS, 2004: 143-148.

[3] JANUZAJ E, KRIEGEL H P, PFEIFLE M. Scalable density based distributed clustering[C]//Proc the 8th Eur Conf Principles and Practice of Knowledge Discovery in Databases. Paris, 2004: 231-244.

[4] 李锁花,孙志挥,周晓云.基于特征向量的分布式聚类算法[J].计算机应用, 2006, 26(2): 379-382.
LI Suohua, SUN Zhihui, ZHOU Xiaoyun. Distributed clustering algorithm based on feature vector[J]. Journal of Computer Applications, 2006, 26(2): 379-382.

[5] INAN A, SAYGIN Y. Privacy preserving clustering on horizontally partitioned data[C]//Proc the 22nd International Conference on Data Engineering. Atlanta, GA: IEEE Press, 2006: 95-103.

[6] JAGANNATHAN G, PILLAIPAKKAMNATT K, WRIGHT R N. A new privacy-preserving distributed k-clustering algorithm[C]//Proc Sixth SIAM Intel Conf Data Mining. Bethesda, MD, USA, 2006: 492-496.

[7] JANUZAJ E, KRIEGEL H P, PFEIFLE M. DBDC: density based distributed clustering[C]//Proc the 9th Int'l Conf Extending Database Technology. Heraklion, Greece, 2004: 88-105.

[8] DATASET[EB/OL]. [1999-10-28]. http://www.ics.uci.edu/~mlearn/databases/iris/.

[9] The third international knowledge discovery and data mining tools competition dataset[EB/OL]. [1999-10-28]. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[10] MODHA D S, SPANGLER W S. Feature weighting in k-means clustering[J]. Machine Learning, 2003, 52(3): 217-223.

作者简介:

吉根林,男,1964年生,教授,博士生导师,博士,主要研究方向为数据库、数据挖掘、XML技术、入侵检测技术.主持或参加多项国家自然科学基金项目、江苏省自然科学基金项目和江苏省高校自然科学基金项目,主持研制的"分布式数据挖掘原型系统 DDMINER"通过了江苏省科技成果鉴定.发表学术论文 80余篇,其中被 EI、ISTP收录 16篇.

姚　瑶,女,1984年生,硕士研究生,主要研究方向为分布式聚类.