基于微簇的K-Dmeans聚类算法改进

张　昊*

摘　要:在现有的分布式聚类算法中普遍都存在着聚类效率较低且不能有效的保护数据隐私问题。本文针对K-Dmeans算法中原始数据聚类效果不佳的情况,对此算法进行改进。即提出了中心-边缘的系统框架结构,同时在数据集边缘节点进行数据聚类时提出了微簇概念,即边缘节点的数据按照规律进行二次细化聚类,再将聚类结果直接传输给中心节点,由中心节点对再次聚类的数据进行整合。简化了数据在各站点传输时的网络开销,同时保证了各站点数据的独立性,借此提高了数据隐私保护能力。

关键词:K-Dmeans算法、聚类、分布式聚类、微簇

一、概述

伴随信息化的不断发展,数据挖掘领域越来越被人们重视。所谓数据挖掘[1-2]就是从大量的、不完全的、有噪声的、模糊的、随机的数据中,提取潜在有用信息知识的过程。而如今数据挖掘技术面对海量的数据信息如何提高数据处理效率成为越来越多研究者面临的课题。利用聚类将大量数据分割可以大大简化数据挖掘的过程。聚类[3]是将大量待处理数据按照规律分为若干个簇,使每个簇内的数据具备高度相似性,而簇间数据则存在较大相异过程。较常用的聚类算法有层次聚类算法、分割聚类算法、基于密度算法、基于网格算法。

本文主要针对分割聚类算法中的K-Dmeans算法进行改进。文献[6]中提出传统的K-Dmeans算法,各站点间需要传递大量数据,既破坏了各站点的独立性,使网络开销增加,同时不同数据簇之间的界限在数据传输过程中变的模糊,导致数据聚类的效率降低。鉴于此,在K-Dmeans算法的基础上,本文提出了中心-边缘化数据节点的框架结构,引入微簇[3]的概念,即边缘节点的数据按照规律进行二次细化聚类,再将聚类结果传输给中心节点,由中心节点对再次聚类的数据进行整合。借此提高系统处理数据的能力。

二、相关概念

定义1　k-means算法[4-7]

K-means算法是将数据集分为k组,k组数据代表数据集分为k类。每个类随机抽取一个中心点,通过数据反复迭代,重新整合改变原有的分组情况,使得在原有的数据分组不断优化,最终中心点不再发生变化。聚类标准函数E收敛。

准则函数E定义为

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} \| p - M_i \| \qquad (1)$$

其中:p为类C_i的空间点;M_i为类C_i数据对象的平均值。

定义2　k-Dmeans算法[8]

K-Dmeans算法其实就是分布式K-means算法,其指导思想是:任意选择数据集中一点作为主站点,利用k-means算法将其换分为k个簇,分割后各个簇中心点被主站点广播给其余k-1个子站点,通过数据的迭代将计算后的各样本点并入距离最近的中心点,并将不属于自身的样本对象传递给其他簇中心点,直到全局函数E收敛。

定义3　微簇(Cclass_id)

微簇就是对同一个集合中的多维数据进行一种整体表示方式。它的数据结构是:

$(\overrightarrow{CF1^x}, n, class\_id)$

* 张昊,男,铜陵学院数学与计算机学院讲师。

$\overline{CF1^x}$ 表示为该微簇的中心，所有数据在不同维度上的平均值均包含在其中；n 为该微簇中的数据数量；class_id 为该微簇的 ID。

三、基于微簇的 K-Dmeans 聚类算法的改进

K-Dmeans 算法较 K-means 算法有所改进，但是任然存在各站点之间传递数据占用大量网络资源，数据簇间界限不明，同时数据传递过程中还会泄露内部数据。鉴于此，在 K-Dmeans 的基础上引入微簇的概念，不仅可以大大提高数据聚类效率，也可减少隐私数据泄露的可能。

（一）系统框架结构

本文从另一个角度诠释 K-Dmeans 算法的系统框架，将主站点看做中心点，其他被划分好的 k 个簇的中心点被看作边缘点，这样系统框架看做为中心——边缘结构。

该系统框架中，每个边缘节点只处理该节点附近的局部数据并对处理好的数据结果进行分析，再将分析结果直接提交给中心点，在中心点进行二次处理和分析，最终得到数据聚类的结果，系统框架如图 1 所示。因为在该系统结构中边缘节点之间没有数据交互，每个边缘节点只与中心点进行交流，所以整个系统中就不存在元数据的传输。这样就降低了原数据在传输时的巨大消耗，同时也防止了原数据在传输过程中的泄露。大大提高数据聚类的效率。

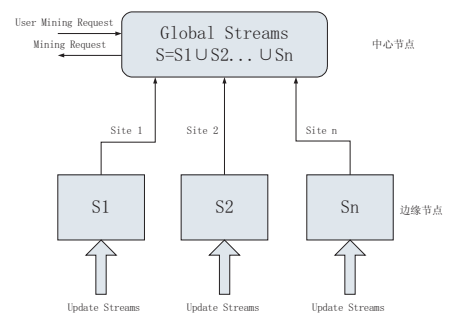

图　基于 K-Dmeans 聚类算法改进后的系统框架

从图 2 看出原始数据集 Di 通过微聚类结果传输到边缘节点 Pi 得到微簇集 Ci，所有微粗集通过聚类算法得到综合聚类结果 C，并最终输出。

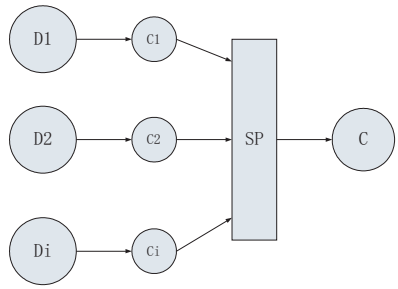

图 2　基于微簇的算法过程图

（二）原始数据在边缘节点上的聚类过程

在分布式聚类环境中，考虑到各节点的差异性，一般在原始数据聚类上会存在时间差异，一般采用 K-Dmeans 算法进行数据聚类，然而 K-Dmeans 算法中选取的节点数目越少，会造成聚类的结果不稳定，而且在每个边缘节点上都存在这聚类不稳定的累加效果，导致最终传输到中心节点的数据不准确。此时在边缘节点引入微簇，对原始数据进行聚类，可以避免这种情况的发生。

定理 1　聚类产生的微簇是类的子集

利用反证法，假设一个数据集中存在 n 个类：$P_1, P_2, \cdots P_n$，该类与 C 类相邻。则 $P_1, P_2, \cdots P_n$ 中所有原始数据点都距 C 类较远。而 $P_1, P_2, \cdots P_n$ 中的原始数据进行聚类形成簇的过程符合微簇的定义，所以这些微簇也离 C 类的质心较远，从而微簇的二次聚类过程不会纳入到 C 类。同理其他类也可得到证明，定理得证。

以边缘节点 Sn 为例，假设在该节点上的原始数据集为 N。

Step1　在 N 个数据中选择 k 个随机数据作为初始中心点，以建立的中心点为基础，自然形成 k 个微簇，每个微簇是以个 d+3 维的向量，其格式为：($\overline{CF1^x}$,n,class_id)。

Step2　计算所有数据点到 k 个中心点的距离，选择距离最近的簇加入，形成微簇。

Step3　所有数据点加入簇完毕后，根据数据集的变化情况，重新调整微簇的中心 $\overline{CF1^x}$ ，和包含的节点个数 n。

Step4　当 $\overline{CF1^x}$ 和 n 不在发生变化时，输出所有微簇，否则返回 Step2。

（三）基于微簇的中心节点数据聚类

在边缘节点形成的微簇，最终都将传输到中

心节点进行融合,由于微簇之间具有不同的权值,即微簇包含的节点数目越多权重越大,其成为类中心的可能性也越大,且微簇可能存在重合性,所以每个微簇并不平等,故不能按照一般的聚类算法进行运算。因此考虑在中心点进行的数据聚类采用基于权值 K-means 聚类算法,将微簇的质心作为中心节点的数据对象,并根据每个微簇的 n 值分配不同的权重。算法步骤如下:

输入:来自 m 个节点的微簇集 $\{C_1, C_2, \cdots, C_m\}$,其中每个微簇集 C_j 包含该节点进行聚类后的 k' 个微簇 $\{c_{j1}, c_{j2}, \cdots, c_{jm}\}$。

输出:整个数据集的聚类结果。

Step**1**　对 m * k' 个微簇进行处理,选择中心相等的微簇进行合并,调整 n 的值;

Step**2**　选择 k 个相互之间距离较大且权重大的簇作为初始聚类中心;

由于聚类的数据可能分布不均匀,若存在节点 C_1、C_2 的质心 A、B 不重合但非常接近的情况,则此时 C_1、C_2 权重相当,如果按照权重排序的方法选取初始质心,会导致以 A、B 为质心进行聚类,聚类结果不准确。因此,本文采用了在进行权重排序的前提下设置门槛值的方法确保选择合适的质心。

首先,对 m * k' 个微簇进行权重排序;

其次,计算任意两个微簇间距离的平均值

$$\delta = \bar{d} = \sum d_{ij} / C_{mk}^2$$

再次,按照权重大小进行排序,将排名第一的微簇作为第一个初始质心,再将计算好的新微簇与设定的门槛值进行比较,只有当选定的微簇距离大于门槛值,才可确定该微簇为初始质心:

$$d_{ij} > \delta$$

最终,选择 k 个相互之间距离较大且权重大的簇作为初始聚类中心;

Step**3**　按照距离将微簇分配到最近的类中,更新类中心及类中的原始数据数目;

由于微簇本身就是一个小规模的数据集合,与类中原先包含的数据源不相同,所以将微簇作为"原始数据"进行聚类,应该计算其几何平均值,而不能单纯计算数据点的平均值来确定类的中心。记录微簇中心点乘以微簇中所包含的数据量 $\overline{CF1^x}$ * n,得到结果求平均,作为类更新后的中心。有公式:

$$\overline{CF1} = \sum_j \frac{n_j}{\sum n_i} \overline{CF1_j} \qquad (2)$$

Step**4**　最终类中心不再发生变化,执行 Step5,否则返回 Step3;

Step**5**　输出聚类结果。

三、实验结果与性能分析

实验数据针对本文提出的基于微簇的 K-Dmeans 算法与 K-Dmeans 算法进行比较,编程环境为 Micsoft Visual C++,操作系统为 Windows XP,实验用的 4 台计算机配置为 3.5GHz Pentium Ⅳ CPU 以及 2G 内存和一台 24 口的百兆以太网交换机。

实验采用 Iris[ ]数据集进行测试,如表 1 所示:

表 1　Iris 测试数据集

数据集	纬度	数据样本个数
Iris	4	160
	8	220
	12	1000
	19	2000

实验过程将 4 台计算机组成局域网,利用 Iris 数据集分别对 K-Dmeans 算法和基于微簇的 K-Dmeas 算法进行测试,总共进行 10 次实验。对所选取的 4 组数据集分别进行测试,实验第一部分测试两种算法的效率比较,实验中 K-Dmeans 分布式聚类直接在局域网内进行数据的聚合;涉及到利用微簇进行聚类时每次随机选取 6 个边缘节点,每个边缘节点数据量为 1000,总共聚合为 30 个微簇,微簇结果传递给中心节点聚合为 6 个类。最终将 10 次实验结果取平均值导出,将测试结果的数据聚类效率与精度进行对比。

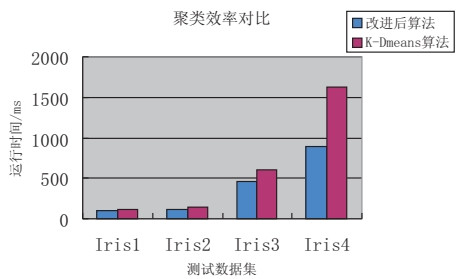

图 3　改进后算法与 K-Dmeans 算法效率比较

从图3可以明显看出在数据集规模较小时，改进后的K-Dmeans算法与K-Dmeans算法用时相差不大，但随着数据集的规模不断变大，两种算法的时间差异性就比较明显了。这是因为改进后的算法采用中心-边缘架构且数据集在边缘节点进行微簇聚类，然后将聚类结果直接传递给中心节点进行处理，减少了类与类之间的数据传递，所以在处理数据时节省时间。

实验第二部分主要证明改进后的算法的精度问题，实验采用Micro-precision[ ]，标准，其定义为：

$$\text{Micro - precision} = \frac{1}{n}\sum_{i=1}^{k}\alpha_i \qquad (3)$$

公式中n代表采集数据样本的总数，k表示聚类的个数，α_i表示数据聚类后被正确归类的样本个数。从该公式中反映出当Micro-precision值越大，最终聚类的精度越高。对数据样本集Iris进行10次实验，其聚类精度对比如图：

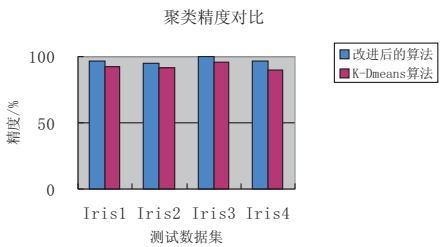

图4　改进后算法与K-Dmeans算法精度比较

从图4可以看出，改进后的算法由于在在边缘节点进行了微簇聚类，等于对数据集的二次细化，所以在聚类效果上精度略高于K-Dmeans算法。

四、结束语

本文针对K-Dmeans算法在聚类中的数据传递消耗大、效率低的缺点进行改进，提出了中心-边缘框架结构，引入了微簇概念，在边缘节点进行微簇聚类，并将聚类结果直接传递给中心节点进行处理，有效的减少了原算法在数据传递中的网络开销，且提高了数据聚类效果，降低了局部站点的数据迭代次数，实验结果证明，该算法的改进是一种有效可行的分布式聚类算法。但算法中对微簇参数的进一步分析，更有效的找出异类数据对象，还需要更加深入的研究。

参考文献：

[1] Han Jiawei, KAMBER M. Data Mining Concepts and Technique, Second Edition[M]. Beijing: China Machine Press, 2007: 312-402.

[2] Lee G, Chang CY, Chen ALP. Hiding sensitive Patterns in association rules mining. In: Wong E, Kanoun K, eds. Proc. of the 28th Int'l Computer Software and Applications Conf. (COMPSAC 2004). Piseataway: IEEE Computer Society, 2004: 424-29.

[3] Charu C Aggarw, Jiawei Han, Jianyong Wang. A Framwork for On-Dem and Classification of Evolving Data Strings [J]. IEEE Transaction on Knowledges and Data Engineering 2006, 18(5): 322-326.

[4] Wang ET, Lee G, Lin YT. A novel method for protecting sensitive knowledge in association rules mining. In: Chen IR, Ibbett R, Mei H, eds. Proc. of the 29th Annual Int'l Computer Software and Applications Conf. (COMPSAC 2005). Edinburgh: IEEE Computer Society, 2005: 511-516.

[5] Oliveira SRM, Zafanc OR, Saygin Y. Secure association rule sharing. In: Dai H, Srikant R, Zhang C, eds. Proc. of the 8th Pacific-Asia Conf. on knowledge Discovery and Data Mining (PAKD 2004). Berlin: Springer-Verlag, 2004. 74-85.

[6] 刘力雄，郭云飞，康晶. 分布式数据流聚类算法[J]. 计算机工程与设计，2011，32(8)：2708-2711.

[7] 陈文. 基于FP树的加权频繁模式挖掘算法[J]. 计算机工程，2011，38(6)：63-65.

[8] Oliveira SRM, Zaitane OR. Algorithms for balancing Privacy and knowledge discovery in association rule mining. In: Desai BC, Ng W, eds. Proc. of the 7th Int'l Database Engineering and Applications symp. Hong Kong: IEEE Computer society, 2003: 54-63.