Chameleon 聚类算法的 Weka 实现①

刘文凤[1]　卿晓霞[2] (1. 重庆大学 计算机学院 重庆 400044; 2. 重庆大学 城市建设与环境工程学院 重庆 400045)

摘　要： 为了克服 Weka 系统在聚类算法方面的不足，在 Weka 的开源环境下进行了二次开发，扩充了聚类算法。介绍了 Chameleon 算法的基本原理和构建步骤，给出算法的具体描述，并将 Chameleon 算法嵌入 Weka 平台，充分利用了其中的类和可视化功能。对实现的系统进行了实验和测试，并对嵌入的算法和原有聚类算法 k-means 进行了对比分析。实验结果表明，Chameleon 算法可获得更好的聚类效果。

关键词： Chameleon；聚类分析；互连度；近似度；Weka

Study of Chameleon Clustering Algorithm and Implementation in Weka

LIU Wen-Feng[1], QING Xiao-Xia[2] (1. College of Computer Science of Chongqing University, Chongqing 400044, China; 2. Faculty of Urban Construction &Environmental Engineering, Chongqing University, Chongqing 400045, China)

Abstract: To overcome the Weka's weakness in clustering algorithms, the clustering algorithm was developed based on the Weka open source environment. An introduction is made on the basic principle and construction steps of the Chameleon algorithm which was described concretely and embedded into Weka platform whose inner class and visualization function were exploited adequately. The newly built Weka system was tested by comparing the embedded and the already existed K-means algorithm. The result shows that Chameleon algorithm acquires better clustering effect.

Keywords: chameleon; cluster analysis; interconnectivity; closeness; weka

聚类是指将物理或抽象的集合分组成为由类似的对象组成的多个类的过程[1]。目前文献中存在大量的聚类算法，通常可以将它们分为基于划分的、基于层次的、基于密度的、基于网格的和基于模型的 5 大类。k-means 算法是基于划分的典型算法[2,3]，比较简单，容易实现，复杂度与数据集的大小成线性关系。k-means 算法在数据划分上不考虑类的分层结构问题，对于聚类中心相距较远的数据样本具有很好的聚类效果，而对于具有同心圆特征的数据样本很难得到好的聚类效果。Chameleon 算法是基于层次的聚类方法[4]，在合并两类时既考虑了互连度，又考虑了近似度，特别是簇内部的特征，因而能够自动地适应被合并簇的内部特征，具有发现任意形状簇的能力，并且可以降低噪声和离群点的影响，提高了计算的有效性。

Weka(Waikato environment for knowledge analysis)是新西兰怀卡托大学(The University of Waikato)研究的基于 Java 语言的开源数据挖掘平台，集成了大量数据挖掘任务的机器学习算法，包括数据预处理、分类、回归、聚类、关联规则以及可视化等

① 基金项目:国家科技重大专项(2008ZX07315-001)
收稿时间:2010-03-28;收到修改稿时间:2010-04-26

诸多工具。由于其源码的开放性，Weka不仅可以用于完成常规的数据挖掘任务，也可以用于数据挖掘的二次开发。但它在聚类方面较为薄弱，只集成了少数几种传统的算法，并没有反映出新兴的研究成果。本文在开放源码的 Weka 平台上，探索了将 Chameleon 算法嵌入到 Weka 中的方法，扩充了系统功能，并与原有聚类算法进行了性能对比。

1 Chameleon算法

1.1 Chameleon 算法的基本原理

Chameleon 算法是一个在层次聚类中采用动态模型的聚类算法。其主要思想如图 1 所示：首先由数据集构造一个 k 最近邻图，再通过图划分算法将 k 最近邻图划分为大量相对较小的子簇，然后使用凝聚层次聚类算法，基于子簇的相似度反复地合并子簇形成最终的簇。

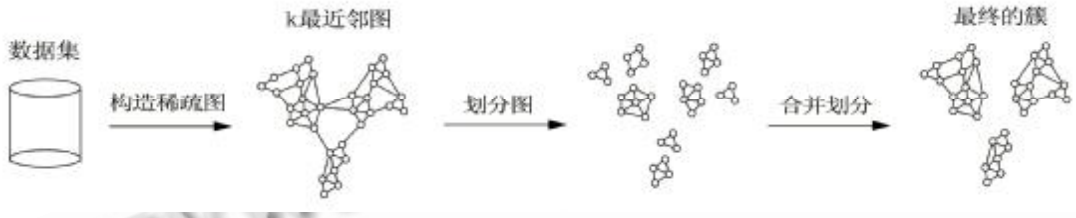

图 1 基于 k 最近邻和动态建模的层次聚

1.2 Chameleon 算法的构建

Chameleon 算法具体可分为三个阶段。

① 构造 k 最近邻图。

Chameleon 先构造一个 k 最邻近图，图中每一个点代表数据集中的一个数据点，若数据点 a 到另一个数据点 b 的距离值是所有数据点到数据点 b 的距离值中 k 个最小值之一，则称数据点 a 是数据点 b 的 k 最近邻对象之一。若一个数据点是另一个数据点的 k 最近邻对象之一，则在这两个点之间加一条带权边，其权重用两个对象间的相似度表示。

② 划分图。

用图形划分算法(hMETIS 算法[7])划分该 k 最邻近图，使得割边(edge cut)最小化，即是使簇 C 划分为两个子簇 C_i 和 C_j 时需要截断的边的加权和最小(这个过程称为最小截断)。然后把每一个子图看成一个初始图，重复这个算法直至达到一定的标准。

③ 合并子簇形成最终簇。

定义 1. 绝对互连度(Absolute Interconnectivity) $EC(C_i,C_j)$：簇 C 划分为两个子簇 C_i 和 C_j 割边的权重和。

定义 2. 内部互连度(Internal Interconnectivity) $EC(C_i)$ (或 $EC(C_j)$)：将 C_i (或 C_j)划分成大致相等的两部分的割边的权重和。

定义 3. 相对互连度(Relative Interconnectivity) $RI(C_i,C_j)$：簇 C_i 和 C_j 之间的绝对互连度关于两个簇 C_i 和 C_j 的内部互连度的规范化，即是：

$$RI(C_i,C_j)=\frac{\left|EC(C_i,C_j)\right|}{\frac{1}{2}\left(\left|EC(C_i)\right|+\left|EC(C_j)\right|\right)}$$

定义 4. 绝对近似度(Absolute Closeness) $\overline{SEC}(C_i,C_j)$：簇划分为两个子簇 C_i 和 C_j 割边的平均权重。

定义 5. 内部近似度(Internal Closeness) $\overline{SEC}(C_i)$ (或 $\overline{SEC}(C_j)$)：将 C_i (或 C_j)划分成大致相等的两部分的割边的平均权重。

定义 6. 相对近似度(Relative Closeness) $RC(C_i,C_j)$：簇 C_i 和 C_j 之间的绝对近似度关于两个簇 C_i 和 C_j 的内部近似度规范化，即是：

$$RC(C_i,C_j)=\frac{\overline{SEC}(C_i,C_j)}{\frac{|C_i|}{|C_i|+|C_j|}\overline{SEC}(C_i)+\frac{|C_j|}{|C_i|+|C_j|}\overline{SEC}(C_j)}$$

其中，$|C_i|$ 和 $|C_j|$ 表示簇 C_i 和 C_j 各自的数据点个数。

Chameleon 采用自底向上的凝聚的层次聚类算法，基于子簇的相似度反复合并子簇。Chameleon

使用两种方法来合并相邻的簇。

相对互联度 $RI(C_i,C_j)$ 和相对近似度 $RC(C_i,C_j)$ 必须满足用户指定的阀值 T_{RI} 和 T_{RC}。

计算每个簇 C_i 和其邻近簇 C_j 的相对互联度 $RI(C_i,C_j)$ 和相对近似度 $RC(C_i,C_j)$，看是否满足下列条件：

$$RI(C_i,C_j)\geq T_{RI} \quad \text{AND } RC(C_i,C_j)\geq T_{RC}$$

若满足该条件则将其合并，若满足条件的邻近簇大于一个，选择与 C_i 具有最高绝对互连度的簇合并。重复这个过程直到没有满足条件的簇为止。

① 定义度量函数。若簇 C_i 和它的邻近簇 C_j 的函数值最大则合并，函数定义为：

$RI(C_i,C_j)\times RC(C_i,C_j)^a$　a 是用户指定的参数 $a>1$，则表示相对近似度更重要；$a<1$，则表示相对互连度更重要。重复这个过程直至没有满足条件的簇为止。

2 Weka数据挖掘平台

Weka 是一个运行于 Java 平台的开源软件，它实现并提供了适用于任意数据集的数据预处理分类、回归、聚类、关联规则以及可视化等诸多工具，还可以用于开发新的机器学习方案，具有很强的扩展性和兼容性。2005 年 8 月，在第 11 届 ACM SIGKDD 国际会议上，Waikato 大学的 Weka 小组荣获了数据挖掘和知识探索领域的最高服务奖，Weka 系统得到了广泛的认可，被誉为数据挖掘和机器学习历史上的里程碑，是现今最完备的数据挖掘工具之一。

Weka 包括 4 个图形用户界面：①Explorer(探索者)，通过这个用户界面，所有 Weka 的功能都可以由菜单选择及表单填写的方式完成。②Experimenter(实验者)，这是专门设计来帮助用户解答实际应用中所遇到的基本问题，使用户能够将不同的学习技术进行比较。③Knowledge Flow(知识流)，使用户能够自己设置如何处理流动中的数据。④Simple CLI(简单命令行)，以原始形式通过命令行界面运行上述三个界面所实现的功能。

但是，Weka 还存在不足，首先它所集成的大部分算法对于大数据集的挖掘效率低，甚至对运行环境的配置要求比较高，离海量的数据挖掘需求仍有一定的距离。其次 Weka 能直接访问的数据源种类比较少，不支持对非结构化的文本文件、图像、视频、Web 等数据的挖掘；同时它所集成的算法并没有反映出新兴的研究成果，特别是在聚类方面较为薄弱，只集成了少数几种传统的算法。针对此不足，本文选择 Weka 系统中算法较少的聚类模块进行算法的二次开发，嵌入原 Weka 系统中没有的 Chameleon 算法。

3 Chameleon算法在Weka中的实现

了解 Weka 聚类器的相关接口，是在 Weka 中实现 Chameleon 算法的基础。分析 Weka 的源代码可以得到，weka.core 包是 Weka 的核心，该包中又有 3 个类是重中之重，实现首先就从这里入手。这 3 个类是 Instances(实例集)类、Attribute(属性)类和 Instance(实例)类，Weka 里的每个算法都会用到这 3 个类。聚类器类结构是实现 Chameleon 算法的另一个关键。Weak.clusterers 包里是各种聚类算法以及聚类器接口和评估器接口，其中有 K-means、EM、Cobweb、FarthestFirst 等算法。在了解 Weka 聚类器的接口之后，就可以在 Weka 中设计与实现 Chameleon 算法。为充分利用 Weka 开放源码所提供的原有类，所开发的 Chameleon 算法的主类 Chameleon 类调用了相应的 Weka 中的包和 Java 的包。

设计 Chameleon 算法组件时，Chameleon 类派生于聚类器类，其方法概述如下：

① 构造函数：Chameleon() //默认构造器，初始化种子(seed)为 10。

② 成员方法概述(部分)：

void buildClusterer(Instances data) //产生一个聚类器

int clusterInstance(Instance instance) //分类一个给定的实例

int clusterProcessedInstance(Instance instance , boolean updateErrors) //聚类器处理实例

double distance(Instance first, Instance second) //计算两个实例之间的距离

double[ ][ ] solveW(Instances instances) //构造相似矩阵

void merge(Instances instances) //合并最近的两个簇

int getNumClusters() //取簇的个数

int numberOfClusters() //返回聚类器的个数

void setNumClusters(int n) //设置簇个数的大小

同时，本文对 Weka 系统进行了部分汉化工作，图 2 为 Weka 运行 Chameleon 算法的界面。

图 2　Weka 中运行 Chameleon 算法

4　实验与测试

为验证嵌入的 Chameleon 算法性能，在嵌入算法后的 Weka 上针对不同数据集进行 Chameleon 算法和 k-means 算法的性能测试。测试数据集 Wine、Glass、Iris(鸢尾花)均来自 UCI 机器学习数据集[8]。Wine 数据集包含 178 个样本、13 个数值型属性，分成数量不等的 3 个类。Glass 数据集包含 214 个样本，9 个数值型属性，分成数量不等的 6 个类。Iris 数据集包含 150 个样本，4 个数值型属性，分成数量相等的 3 个类。表 1 为 Chameleon 算法和 k-means 算法的聚类错误率对比。

表 1　Chameleon、k-means 算法的聚类错误率(%)对比

数据集	Chameleon	K-means
Wine	26.157 0	32.425 1
Glass	52.562 7	56.830 6
Iris	35.259 0	43.336 1

从表 1 可以看出 Chameleon 算法的聚类平均错误率比 k-means 算法平均错误率低。在实验中发现，当数据集比较大时，Chameleon 算法容易产生内存溢出现象。

为检测聚类的稳定性，就两种算法在取不同簇数目的情况下对目标函数值进行了试验。图 3 中的 3 个折线图分别为数据集 Wine、Glass、Iris 取不同簇数目进行测试时目标函数的取值情况。可看出这两种算法的目标函数值都随着 k 值的增大而减小，并逐渐趋于平稳，而 Chameleon 算法的折线相对于 k-means 算法的折线要平稳一些，因此 Chameleon 算法相对比较稳定。

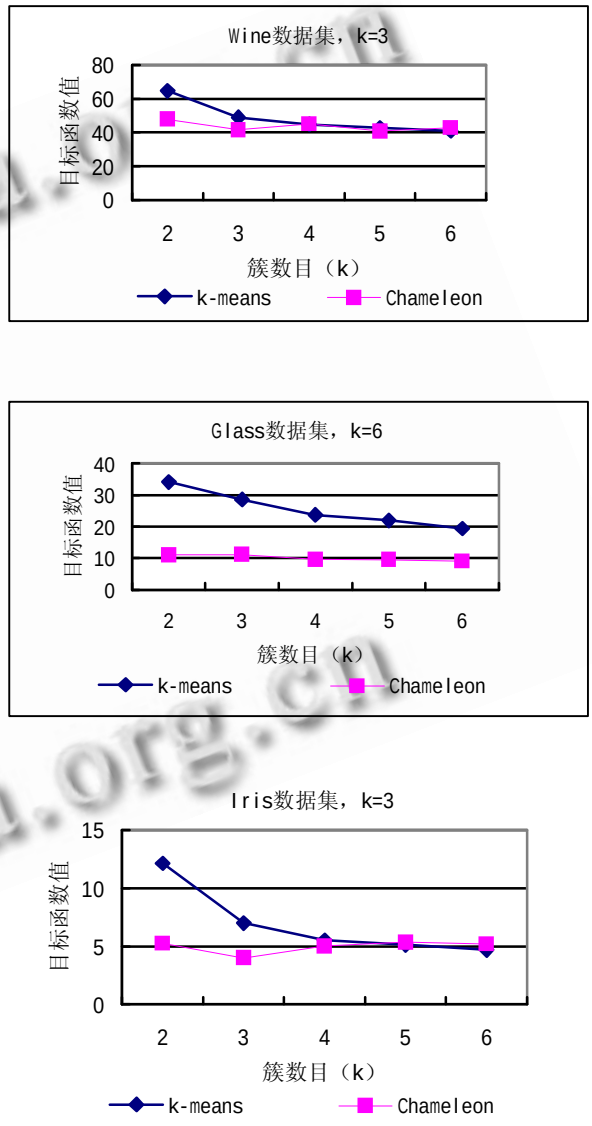

图 3　Chameleon、k-means 算法稳定性比较

为进一步说明 Chameleon 算法具有较强发现任意形状和任意大小簇的能力，在由二维数据点组成的数据集 DS 上进行了实验，数据集 DS 包含三种几何形

状。从图 4 可以看出，Chameleon 算法对这三种几何形状的聚类效果较 k-means 算法更优。并且 k-means 算法的聚类错误率为 40.54%，而 Chameleon 算法为 27.02%，明显低于 k-means 算法。

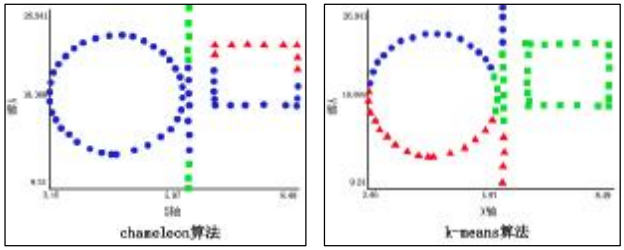

图 4　Chameleon、k-means 算法发现形状能力比较

5　结束语

本文介绍了 Chameleon 算法的基本原理，具体描述了 Chameleon 算法的构建过程，并介绍了其在 Weka 中的实现，使该方法能更方便的应用。Chameleon 算法是具有较强发现任意形状和任意大小簇能力的聚类算法，因此用 Chameleon 算法对数据集进行聚类分析可得到较好的聚类质量。但 Chameleon 算法的时间复杂度为 O(n2)，对于大型数据集 Chameleon 算法的时间代价较大。

参考文献

1 Jiawei Han, Micheline Kamber. 数据挖掘:概念与技术.范明,孟小峰译. 北京: 机械工业出版社, 2007: 14－18,51－305.

2 Hartigan J, Wong M. A K-means clustering algorithm. Applied Statistics, 1997(28):100－108.

3 Krishna K, Murty MN. Genetic K-Means algorithm. IEEE Trans SystMan Cybern: Part B, 1999, 29(3):433－439.

4 Karypis G, Han Eui-Hong, Kumar V. Chameleon: Hierarchical Clustering Using Dynamic Modeling. Computer, 1999,32(8):68－75.

5 Witten I H, Frank E.数据挖掘实用机器学习技术.董琳, 邱泉,于晓峰等译.北京:机械工业出版社, 2006.

6 Weka 3 – Data Mining with Open Source Machine Learning Software in Java. 2009-11-10. http://www.cs.waikato.ac.nz/ml/weka/.

7 Karypis G, Kumar V, hMETIS 1.5: A Hypergraph Partitioning Package. Tech. Report, Dept. of Computer Science, Univ. of Minnesota. 2009-12-15. http://glaros.dtc.umn.edu/gkhome/metis/hmetis/changes.

8 UCI machine learning repository: Data Sets. 2010-01-20.http://archive.ics.uci.edu/ml/datasets.html.