基于 Hadoop 的局部支持向量机*

崔文斌[1], 温孚江[2+], 牟少敏[1,2], 浩庆波[1]

[1](山东农业大学 信息科学与工程学院,山东泰安 271018)

[2](山东农业大学 农业大数据研究中心,山东泰安 271018)

Local Support Vector Machine Based on Hadoop*

CUI Wen-Bin[1], WEN Fu-Jiang[2+], MU Shao-Min[1,2], HAO Qing-Bo[1]

[1](School of Information Science and Engineering,Shandong Agricultural University,Taian 271018)

[2](Agricultural Big-Data Research Center,Shandong Agricultural University,Taian 271018)

+ Corresponding author: Phn: +86-15005486826, E-mail: msm@sdau.edu.cn

Abstract: With the continuous development of Internet of things, cloud computing technology, The generated data is growing at an explosive rate, How to mining and analysis has become a hot research in the present academic circles in large data. Hadoop distributed computing platform has become the main technology of data analysis. Support Vector Machine is widely used in Data mining , and Local Support Vector Machine was a new classification algorithm that is based on support vector machine. But local support vector machine construct classifier for each test samples, In large data carries on the classification of high time complexity, the classification efficiency is low. In view of the above problems, combined with the Hadoop parallel computing platform, we proposed a local support vector machine algorithm based on Hadoop. This paper makes two improvements on the local support vector machine: the first is that the calculation of k-nearest neighbor for the test sample is parallel, second is the training of model is parallel. We did some tests, and the results show that the local support vector machine based on Hadoop can effectively reduce the classification time, and the classification accuracy of this algorithm is consistent with the classification accuracy in local support vector machine.

Key words: Hadoop; Big Data Analytics; Local Support Vector Machine; Big Data

摘要：随着物联网、云计算等技术的不断发展，产生的数据也以爆炸式的速度不断增长，如何在大数据中进行挖掘和分析成为了当前学术界研究的热点. Hadoop 分布式计算也因此逐渐成为了大数据挖掘和分析的主要技术.支持向量机则是一种应用比较广泛的数据挖掘方法，局部支持向量机是在支持向量机的基础上引入局部学习算法的一种有效的分类算法.但是，局部支持向量机需要为每个测试样本分别构造分类器，在大数据上进行分类的时间复杂度较高，分类效率比较低.针对上述问题，本文结合 Hadoop 并行计算平台，提出了基于 Hadoop 的局部支持向量机算法.本文对局部支持向量机进行了两方面的改进：第一是将计算测试样本的 k 近邻并行化，第二是将训练模型并行化.通过测试实验，结果表明：基于 Hadoop 的局部支持向量机能够有效降低分类时间，且在分类精度上与局部支持向量机基本保持一致.

关键词：Hadoop；大数据分析 ；局部支持向量机；大数据

中图法分类号: TP391 **文献标识码**: A

0 引言

支持向量机(Support Vector Machine, SVM) 是由 Vapnik[1,2]于 1995 年根据统计学理论提出来的一种分类算法.支持向量机在解决小样本、非线性和高维空间分类的情况下具有一定优势，其原理是借助于核函数将样本数据映射到高维空间中并使其具有线性可分性，然后利用决策函数来判定测试样本的类别.但支持向量机也存在其局限性，即不能够充分利用样本的局部信息.Brailovsky 等人[3]在支持向量机核函数的基础上引入两个乘子使之具有局部性，提出了局部支持向量机(Local SVM,LSVM)，其模型定义如下：

$$h(|\mathbf{x}_i-\mathbf{w}_r|)=\begin{cases}1 & \text{如果}|\mathbf{x}_i-\mathbf{w}_r|\le\theta_r\\ 0 & \text{否则}\end{cases} \tag{1}$$

* Supported by the Shandong Provincial Natural Science Foundation, China (No. ZR2012FM024), the 2013 Shandong province agriculture major application of technology innovation project.

作者简介: **崔文斌**(1992－),男,山东省临沂市莒南县人,研究生,主要研究领域为机器学习,图像处理,大数据.

定义新的核函数如公式(2)：

$$K^*(\mathbf{x}_i,\mathbf{x}_j)=\sum_{r=1}^{t}K_{wr}(\mathbf{x}_i,\mathbf{x}_j) \tag{2}$$

其中

$$K_{wr}(\mathbf{x}_i,\mathbf{x}_j)=K(x_i,x_j)h(|x_i-w_r|)h(|x_j-w_r|) \tag{3}$$

$w_1,w_2,w_3,...w_r$ 等 r 个元素能够使 $|\mathbf{x}_i-\mathbf{w}_r|$ 覆盖训练集的所有样本.通过设定 θ_r 的取值来定义局部核函数：

$$K^*(\mathbf{x}_i,\mathbf{x}_j)=\begin{cases}K(\mathbf{x}_i,\mathbf{x}_j) & \text{如果}\mathbf{x}_i\text{在}\mathbf{x}_j\text{的}k\text{近邻中}\\ 0 & \text{否则}\end{cases} \tag{4}$$

局部支持向量机与标准的支持向量机相比能够充分利用样本的局部信息[4,5]，并且分类的精度比支持向量机高[6].但局部支持向量机需要为每个测试样本分别构造一个分类模型，因此算法的时间复杂度较高.

近年来随着物联网、云计算等技术的不断发展，产生的数据也以爆炸式的速度不断增长。这一现象引发了学术界和产业界的高度关注，在 2008 年《Nature》推出了大数据专刊对其展开探讨，从互联网技术、超级计算等多个方面介绍了大数据所带来的挑站[7].一般意义上，大数据是指无法在一定时间内用常规机器和软硬件工具对其进行感知、获取、管理、处理和服务的数据集合[8].

Hadoop 是一个可以实现大数据计算的分布式平台，由 Apache 公司于 2005 推出，由于其具有较好的容错机制，并且对设备的性能要求较低，能够在普通 PC 机上进行集群搭建，因此 Hadoop 已成为当前大数据领域内使用频率最高的技术.Hadoop 主要包括并行化编程模型(MapReduce)和分布式文件存储系统(HDFS)两部分.MapReduce 负责对大规模数据进行并行化计算，HDFS 则是管理大规模数据的分布式存储.本文提出的算法是在 Hadoop 平台下将局部支持向量机进行并行化，本文将主要对局部支持向量机算法进行了两方面的并行改进，首先是将局部支持向量机中的 k 近邻并行，然后在测试样本的 k 近邻的基础上，将支持向量机训练模型的过程进行并行化.最终将分类的结果进行汇总计算出基于 Hadoop 的局部支持向量机的分类精度.

本文主要分为五部分，第二部分介绍了 MapReduce 并行化编程模型，第三部分主要介绍本文提出的基于 Hadoop 的局部支持向量机，第四部分对本文提出的算法进行测试并对测试结果进行分析.最后对本文进行总结.

1 MapReduce 并行化编程模型

MapReduce 是由 Google 实验室提出的一种并行的编程模型，该模型对数据的处理分为 Map 和 Reduce 两个过程[9].其中 Map 接收<key1,value1>键值对类型的数据，根据 Map 的功能执行相应的操作，并将执行结果以<key2,list<value2>>键值对的形式输出.Reduce 接收<key2,list<value2>>键值对的数据，然后根据 Reduce 的功能进行对应的处理，最后以<key3,value3>键值对的形式输出结果.Hadoop 平台则是根据 MapReduce 和 HDFS 系统产生的开源大数据处理平台.

1.1 MapReduce

在 Map 阶段之前需要对数据集进行分片，将输入的数据集分为多个 Split 片，Map 读取 Split 分片时，将分片中的数据转换成 Map 阶段需要的<key1,value1>形式的键值对.Map 类中主要有 setup()、map()、cleanup()和 run()四个函数.run 函数是 Map 类的启动接口，每个 Map 阶段的开始默认都会调用 run()函数.setup()函数是在读取数据之前调用的函数.map()函数主要进行数据的读取操作.cleanup()函数则是在数据读取完以后调用,默认的 cleanup()函数中没有任务操作.我们可以通过重写 setup()、cleanup()和 map()三个函数来定义需要的 Map 类.

Reduce 阶段主要通过调用 Reduce 类中的 reduce()方法来完成.Reduce 阶段接收 Map 阶段传来的数据后会将接收到的数据根据 key 值进行汇总和排序工作.然后在 Reduce 类中进行数据处理，最后将处理结果写入到 HDFS 文件中.Reduce 类中主要有四个方法，分别是 setup(),reduce(),cleanup(),run()，我们可以通过重写 setup()、reduce()和 cleanup()三个方法定义需要的 Reduce 类.

1.2 MapReduce数据流图

MapReduce 程序通过操作键/值对来处理数据，一般形式如下：

Map: (K1,V1)⟶list(K2,V2)

Reduce: (K2,list(V2))⟶list(K3,V3)

其数据流如图 1 所示

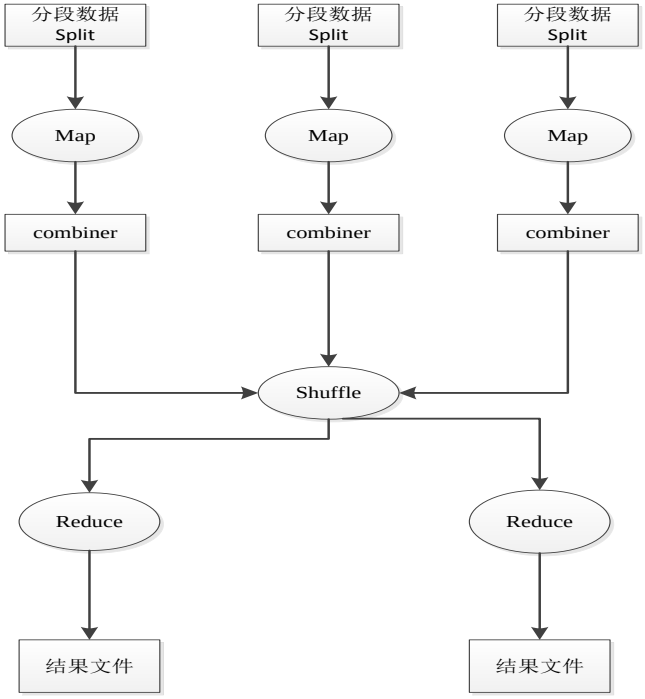

图 1 MapReduce 数据流图

为提高算法效率，在 Map 后可以加入一个 Combiner 阶段，Combiner 的本质是一个本地的 Reduce.Combiner 是将 Map 的结果在本地进行一个 Reduce 操作，这样可以减少各个节点之间的数据传递，降低算法的时间复杂度.但不是所有算法都适合 Combiner 操作，需要用户根据自己的情况决定是否使用 Combiner 操作.

2 基于 Hadoop 的局部支持向量机

局部支持向量机的分类过程可以分为两个阶段，第一阶段是在训练集中计算每个测试样本的 k 近邻，第二阶段则是根据测试样本的 k 近邻为每个测试样本建立分类的模型，确定测试样本的类别.一个算法能够被并行化，其数据集需要具备以下特点：算法的数据集允许被分割成多个数据集，且被分割后的数据集可以完全并行的处理[10].局部支持向量机训练分类模型用的是训练集的部分信息，训练集中的样本数据之间没有关联性，因此数据集能够被分割成为多个小的数据集.根据 MapReduce 编程模式的需要，我们将数据集进行分片，并且保证分片之间的数据无关[11].由于局部支持向量机的数据包括测试集和训练集两种数据集，并且需要保证本文算法中计算的 k 近邻与一般的局部支持向量机计算的 k 近邻保持一致，所以只需将训练集进行分片处理，测试集不做任何处理.

本文提出的基于 Hadoop 的局部支持向量机共分为两个阶段.第一个阶段是在分片的训练集中计算测试样本的 k 近邻，将计算的测试样本的 k 近邻进行汇总.最终计算出测试样本在全部训练集中的 k 近邻.第二个阶段主要训练分类模型，在测试样本 k 近邻的基础上训练分类的模型确定测试样本的类别，并统计局部支持向量机的分类精度.

2.1 算法步骤

1)利用 map 函数为每个测试样本在当前节点的测试集中计算的 k 近邻，称之为局部 k 近邻.

2)利用 reduce 函数对每个测试样本的局部 k 近邻进行汇总，然后在汇总的局部 k 近邻中计算测试样本的 k 近邻，本文称之为全局 k 近邻.每个 Reduce 类都会将计算的 k 近邻，按一定规则写到 HDFS 文件系统中[12].

3)在第二个 Job 的 Map 函数中进行支持向量机的分类工作，最后将分类结果与测试集的类别标号传到 Reduce 类中.

4)在 Reduce 类中进行分类的统计工作，计算分类的精度.

2.2 算法的流程图

基于 Hadoop 的局部支持向量机算法流程图如图 2 所示：

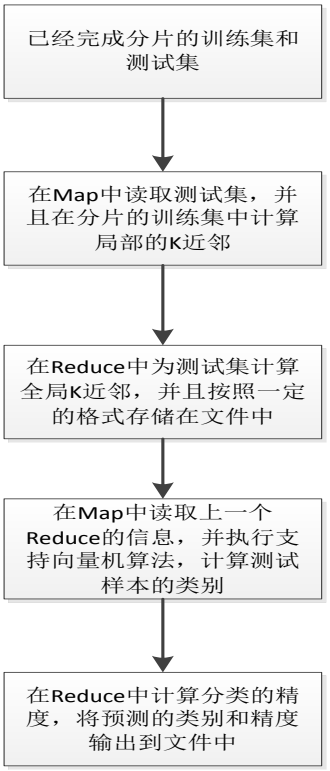

图 2 基于 Hadoop 的算法流程图

局部支持向量机首先计算测试样本的 k 近邻，然后再利用每个测试样本的 k 个近邻构造分类模型.通过算法的步骤可以看出本文提出的算法是将计算 k 近邻和训练模型并行，不会影响 k 近邻的最终计算结果和支持向量的选择，所以当分类的参数选择相同时，本文提出算法的分类精度与一般的局部支持向量机基本保持一致.局部支持向量机中计算量最大的是计算测试样本与训练样本之间的欧式距离和训练分类的模型两部分，本文提出的算法主要是对上述部分进行并行化处理，因此分类的时间效率比一般的局部支持向量机高.

2.3 Map和Reduce函数的设计

Map 主要是从 HDFS 文件系统中按行读取数据，并且在 map 函数中对测试数据进行处理，然后计算测试样本在当前节点的训练集中的局部 k 近邻.Map 类的伪代码描述如下：

第一个 Job 的 Map：

```
public void map(Object Key, Text value, Context context){
        读取 HDFS 文件系统中的文件.
            将读取到的文件内容进行解析，放在 List 集合中.
}
public void cleanup(Context context){
    while（读取 HDFS 中的测试文件）{
        解析读取到的字符串.
        进行 kNN 分类算法，将得到的 k 近邻及测试样本与训练样本的距离信息传到 Reduce 中.
        }
}
```

第一个 Job 的 Reduce：

```
public void reduce(IntWritable Key, Text value, Context context){
                                读取第一个 Job 中的 Map 阶段传递的数据.
                                根据训练样本与测试样本的距离.
                                计算出全局的 k 近邻.
}
```

第二个 Job 的 Map:

```
public void map(Object Key, Text value, Context context){
                读取上一个 Job 的 Reduce 输出文件.
                将数据按组进行封装.
}
public void cleanup(Context context){
                读取封装的数据.
                    解析得到的数据.
                执行支持向量机.
                将预测的结果与真实值传递到 Reudce 中.
}
```

第二个 Job 的 Reduce：

```
public void reduce(IntWritable Key, Text value, Context context){
                解析第二个 Job 中的 Map 阶段传递的数据.
                计算分类的精度.
}
```

2.4 算法时间复杂度分析

设有 N 个测试样本，每个测试样本有 s 个特征属性，则 K 近邻分类(kNN)算法的时间复杂度为 O(Ns).标准的支持向量机分类的时间复杂度为 O(n3)，n 为训练集的数目.局部支持向量机是在测试样本的 k 近邻的基础上利用支持向量机训练分类模型，所以局部支持向量机的时间复杂度为 O(Ns)+O(Nk3).

在 MapReduce 编程模式中会将数据集自动进行分片处理，在 Map 阶段，一个数据块就会对应一个 Map 任务来处理，整个数据集会被分配到不同的节点.假设每个节点可以完成 m 个任务，集群中共有 n 个节点参与并行计算，基于 Hadoop 的局部支持向量机的时间复杂度为：(O(Ns)+O(Nk3))/(m*n)，因此，本文提出的算法在分类时间效率上有所提高.

3 实验及结果分析

3.1 实验环境

实验集群由 6 台计算机搭建完成，配置如下：CPU 型号为 Intel(R) Core(TM)i5-3470 CPU @3.20GHz；内存为 4G；硬盘容量为 1T，Hadoop 的版本为 1.2.0；集群的配置是参考 Hadoop 官方提供的方法进行搭建配置.

3.2 单机与集群处理对比

为了验证算法的有效性，本文在 UCI 数据集中选取了 4 组数据集进行了测试工作，在 4 组数据集中提取部分数据作为测试集，剩余的数据作为测试集，数据集的具体情况如表 1 所示

表 1 数据集

数据集	测试集记录	训练集记录	测试集大小(KB)	训练集大小(KB)
cod_rna	70000	201617	5064	14584
letter.	5000	10500	857	1799
shuttle	10000	32602	962	3138
segment	1000	1310	148	196

我们将本文提出基于 Hadoop 的局部支持向量机(HkNNSVM)与一般的局部支持向量机(kNNSVM)在相同的参数条件下进行了测试对比，核函数为 RBF 核函数，具体实验参数的选择如表 2 所示：

表 2 测试参数表

数据集	惩罚因子	-g
cod_rna	100	5*10-6
letter	100	5*10-6
shuttle	100	5*10-6
segment	100	5*10-6

测试结果如表 3、4 所示,T1 表示一般局部支持向量机所用的时间，T2 为基于 Hadoop 的局部支持向量机所用的时间.

表 3 K=100 各个数据集测试结果

数据集	kNNSVM(%)	T1(ms)	HkNNSVM(%)	T2(ms)
cod_rna	90.898	625216	90.898	242307
letter	79.38	44102	79.2	22937
shuttle	99.3	41207	99.3	28964
segment	92.76	6634	92.76	15914

表 4 K=150 各个数据集测试结果

数据集	kNNSVM(%)	T1(ms)	HkNNSVM(%)	T2(ms)
cod_rna	91.15	687474	91.15	319425
letter	74.58	68713	74.7	26941
Shuttle	99.22	59017	99.22	47977
segment	93.17	10075	93.17	15950

图 3 为 k=100 时，单机与集群的分类精度折线图，图 4 为 k=100 时，单机与集群的分类时间折线图，图 5 为 k=150 时，单机与集群的分类精度折线图，图 6 为 k=150 时，单机与集群的分类时间折线图：

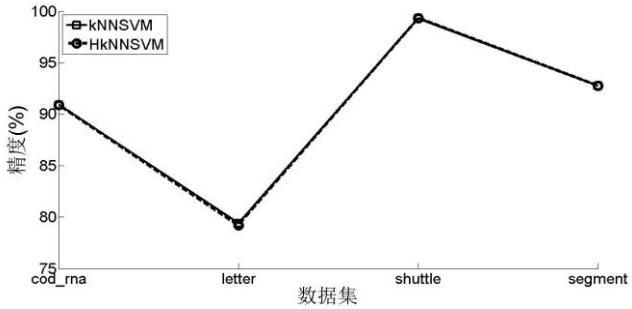

图 3 K=100 时不同数据集的分类精度折线图

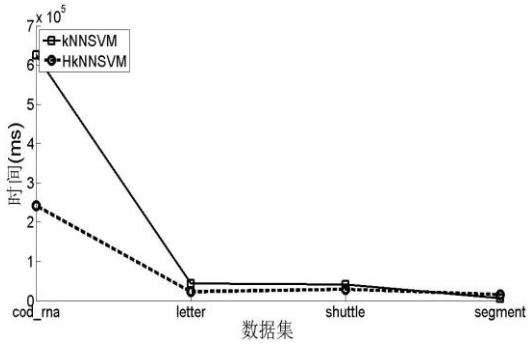

图 4 K=100 时不同数据集分类时间折线图

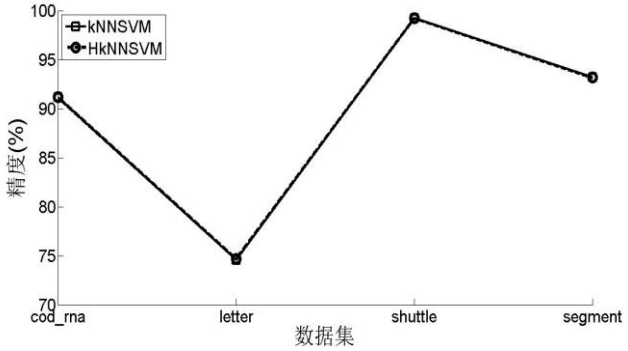

图 5 K=150 不同数据集的分类精度折现图

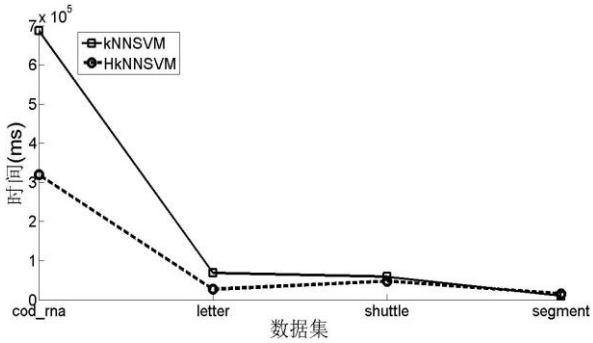

图 6 K=150 不同数据集的分类时间折线图

通过图 4 和图 5 可以看出，当数据集比较小时，基于 Hadoop 的局部支持向量机比一般的局部支持向量机时间复杂度要高.其原因为当数据集较小时，实际计算所占的时间比例比较小，数据在各个节点之间的传递和 Hadoop 每个节点的服务启动所占用的时间比例较大，所以算法的时间消耗比一般的局部支持向量机高.随着数据集的增大，实际计算所占的时间比例也会增大，这样就能够降低服务的启动与数据的传递对总体时间的影响，所以本文提出的算法在大数据量下的分类比一般的局部支持向量机更有时间优势，随着数据量的变大，本文算法优势越明显.通过测试的结果可以得出相同的结论：当数据集较小时，一般的局部支持向量机的分类效率要高于基于 Hadoop 的局部支持向量机，没有体现集群计算的优势.当数据量增大到一定的程度，基于 Hadoop 的局部支持向量机的分类效率就会超过一般的局部支持向量机.通过图 3 和图 6 可以看出基于 Hadoop 的局部支持向量机与一般的局部支持向量机在分类精度上基本上保持一致.因此我们可以得出基于 Hadoop 的局部支持向量机在处理大规模数据分类时比一般的局部支持向量机更有优势.

4 结论

局部支持向量机需要为每个测试样本构造分类模型，当样本数量庞大时，其时间复杂度很高，本文提出了一种基于 Hadoop 的局部支持向量机算法，并在 Hadoop 平台上进行了设计与实现.该算法实现了选取 K 近邻的并行化以及构造分类模型的并行化，可以提高大数据量下的分类速度.实验结果表明，基于 Hadoop 的局部支持向量机在大数据量下的分类效率比一般的局部支持向量机高.本文主要探讨了局部支持向量机分类算法的并行化实现方案，并在 MapReduce 编程将局部支持向量机算法在 Hadoop 平台上进行了设计与实现.通过相关的测试表明，基于 Hadoop 的局部支持向量机在大数据量下的分类效率比一般的局部支持向量机高.

参考文献:

[1] Vapink V. The Nature of Statistical Learning Theory [M]. New York: Springer Verlag, 1995.

[2] Cortes C, Vapink V. Support Vector Networks[J]. Machine Learning, 1995, 20: 273-297.

[3] Brailovsky V L, Barzilay O, Shahave R. On global, local, mixed and neighborhood kernels for support vector machines[J]. Pattern Recognition Letters, 1999, 20(11-130): 1183-1190.

[4]Cheng H et al. Efficient Algorithm for Localized Support Vector Machine[J]. Knowledge and Data Engineering, 2010, 20(4): 537-549.

[5]Segata N et al. Fast and Scalable Local Kernel Machines[J]. Journal of Machine Learning Research, 2010: 1883-1926.

[6] 尹传环, 牟少敏, 田盛丰, 黄厚宽, 朱莹莹. 局部支持向量机的研究进展[J]. 计算机科学, 2012, 01: 170-174.

[7] Big data. Nature, 2008, 455(7209):1-136.
[8]李国杰，程学旗．大数据研究：未来科技及经济社会发展的重大战略领域———大数据的研究现状与科学思考．中国科学院院刊，2012, 27 (6): 647-657.
[9] Lammel R. Google’s MapReduce Programming Model Revised[J].Science of Computer Programming, 2007, 68(3) 208-237l.
[10] Srirama S N, Jakovits P, Vainikko E, Adapting scientific computing problems to clods using MapReduce[J]. Future Generations Computer Systems,2012, 28(1):184-192.
[11]李建江,崔健,王聃,严林,黄义双. MapReduce 并行编程模型研究综述[J]. 电子学报,2011,11:2635-2642.
[12]闫永刚,马廷淮,王建. KNN 分类算法的 MapReduce 并行化实现[J]. 南京航空航天大学学报,2013,04:550-555.