**基于MapReduce和Spark的大数据主动学习比较研究**

翟俊海 齐家兴 沈矗 宋丹丹 王谟瀚 田石

河北省机器学习与计算智能重点实验室 河北大学数学与信息科学学院 保定 071002

摘 要 在我们以前的工作中，提出了基于MapReduce的大数据主动学习算法。在本文中，我们将这一算法移植到Spark环境，提出了基于Spark的大数据主动学习算法，并对基于MapReduce和Spark的两种大数据主动学习算法从运行时间、文件数目、同步数目和耗费内存四个方面进行了比较研究。得出了一些有价值的结论，这些结论对对从事相关研究的人员将提供很好的帮助。

关键词 大数据；机器学习；主动学习；样例选择；开源框架

中图法分类号：TP181 文献标识码：A

#### **Comparative Study of Big Data Active Learning Based on MapReduce and Spark**

Zhai Junhai Qi Jiaxing Shen Chu Song Dandan Wang Mohan Tian Shi

*Hebei Key Laboratory of Machine Learning and Computational Intelligence, College of Mathematics and Information Science, Hebei University, Baoding 071002, China*

**Abstract** In our previous work, a MapReduce based algorithm for big data active learning was proposed. The algorithm was transplanted to Spark environment, a Spark based algorithm for big data active learning was presented in this paper. Furthermore, the two algorithms were experimentally compared on four aspects: running times, number of files, number of synchronization and memory requirements. Some valuable conclusions were obtained, and the conclusions can be very helpful to researchers in related fields.

**Key words** Big data;machine learning; active learning; instance selection; Open source framework

# 1引言

主动学习的思想[1]最早可追溯到1988年，它是一种迭代学习的过程，每一次迭代从大量无类别标签的样例中选择重要的样例，交给领域专家标注类别。主动学习的目的是减少标注的代价，目标是用尽可能少的样例，即标注了类别的样例，训练一个泛化能力好的学习系统。自主动学习被提出以来，研究人员提出了许多主动学习算法。根据无类标样例的呈现方式，主动学习算法可分为基于池的主动学习和基于流的主动学习两种。在基于池的主动学习中，无类标样例全部呈现给主动学习算法；在基于流的主动学习算法中，无类标样例源源不断地呈现给主动学习算法。也可以根据学习任务将主动学习算法大致划分为两类：面向分类的主动学习和面向回归的主动学习。相对而言，面向分类的主动学习研究较多，是主动学习研究领域的主流。

在面向分类的主动学习算法中，最常用的样例选择策略是不确定性，包括信息熵、投票熵、分类错误率等。Lewis和Gale[2]提出了一种基于不确定性选择样例的方法，该方法认为最难以判断其类别的样例最重要，应选择这样的样例交给领域专家进行标注。Seung等人[3]提出了一种基于分类错误率的主动学习算法，该算法认为分类器分类错误的样例是重要的样例，他们提出的算法可以与任何判别模型一起使用，并提供预测错误分类的概率估计。Schohn和Cohn[4]提出一种以支持向量机（SVM, Support Vector Machine）为分类器的主动学习算法，该算法选择为SVM提供最多信息的样例，这样的样例位于分类超平面附近，分类不确定性最大。Tong等人[5]提出一种基于变形空间的主动学习算法，该算法利用SVM选择那些使变形空间减小的样例进行标注。针对逻辑回归分类器模型的主动学习，Yang和Loog做了大量的研究工作，分别提出了基于最大化方差的主动学习算法[6]和基于伪标注者的单次迭代主动学习方法[7]。此外，他们还在逻辑回归框架下，对主动学习算法进行了比较研究[8]。Hsu等人[9]也研究了针对逻辑回归分类器模型的主动学习，提出了一种基于贪心策略的主动学习方法，该方法能同时进行样例选择和特征选择。Kee等人[10]提出了基于样例多样性和样例密度的QBC（Query-by-Committee）批量主动学习算法。Vandoni等人[11]也研究了基于QBC的主动学习，并成功应用于高密度人群的行人检测。Wu等人[12]将分类框架下的单标记主动学习推广到多标记主动学习，提出了基于次模函数的多标记主动学习方法。与此同时，Reyes等人[13]也研究了多标记主动学习，基于类标记向量不一致性和类标记得分排序的思想，提出了一种有效的多标记样例选择方法。Altalhi和Ventura[14]利用非参数统计测试方法，在多个数据集上对多种主动学习策略进行了比较研究，得出了一些有价值的结论。Pereira-Santos等人[15]也做了类似的比较研究。

与针对分类任务的主动学习研究相比，针对回归任务的主动学习研究相对较少。从已有的相关文献来看，在这两类主动学习中，选择样例的原则是一致的。例如，Burbidge等人[16]将分类框架下基于QBC的主动学习，推广到了回归场景，提出了针对回归问题的QBC主动学习方法。Wu等人[17]提出了两种针对回归问题的基于贪心采样的主动学习方法。针对支持向量回归模型，Demir等人[18]提出了一种多判据主动学习方法。Li和Jian[19]利用相关匹配和标签误差抑制机制，提出了针对回归问题的主动学习方法，取得了良好的效果。基于期望模型变化最大化的思想，Cai等人[20]提出一种针对回归问题的批处理模式主动学习方法，实验结果显示这种方法非常行之有效。

针对大数据环境的主动学习研究还非常少，翟等人[20]研究了针对分类问题的主动学习算法在大数据环境中的可扩展性，提出了基于MapReduce的大数据主动学习算法。在本文中，我们将这一算法移植到Spark环境，提出了基于Spark的大数据主动学习算法，并对基于MapReduce和Spark的两种大数据主动学习算法从运行时间、文件数目、同步数目和耗费内存四个方面进行了比较研究。得出了一些有价值的结论，这些结论对对从事相关研究的人员将提供很好的帮助。

# 2 基础知识

# 2.1 主动学习

在分类框架下，一般认为，已标注类别的数据越多，标注越精准，基于这些数据训练得到的分类器泛化能力也越好。然而，在许多实际任务中，容易获得大量无类别标签的数据，请领域专家标注这些数据需要付出巨大的代价。特别是在大数据时代，这种情况更加突出。例如，在图像大数据分类任务中，绝大部分用户上传的图像缺乏准确的语义标签。主动学习[1,14,15]是解决这一问题的有效学习方式，它以迭代方式从无类别标签的数据中选择重要的样例，然后交给专家标注。主动学习的目标是用尽可能少的样例。训练一个高泛化性能的分类器。

主动学习可以用一个四元组刻画：AL=(C, L, U, O)。其中，C表示分类器，L表示有类别标签的样例集合，U表示无类别标签的样例集合，O表示领域专家。主动学习是一个迭代学习的过程，如图1所示。开始时，L包含少量有类别标签的样例。首先，用某种训练算法从L中训练一个分类器C，并用某种预定义的度量指标评估U中样例的重要性，选择若干个重要的样例交给领域专家O进行标注；然后，将标注的样例添加到L中。接下来，重复这一过程，直到训练出的分类器C的泛化性能达到指定的要求，主动学习过程结束。在主动学习中，不确定性是常用的样例选择准则。不确定性是指用分类器C对U中样例进行分类时，分类的不确定性。如果一个样例分类到每一类的概率相同，那么这个样例的不确定性最大。

Fig. 1 The process of active learning

图1 主动学习的过程

# 2.2 MapReduce

MapReduce[22]是开源大数据处理平台Hadoop的重要组件之一，是针对大数据处理的一种并行编程框架，MapRecuce处理大数据的过程如图2所示。从MapRecuce的名称和图2可以看出，MapRecuce由两个阶段构成：Map和Reduce。用户的各种应用逻辑，都是通过这两个函数来实现的。换句话说，用户只需完成这两个函数的设计，即可完成相应的大数据处理任务。

Map函数的输入是一系列键值对，用<*K*\_1, *V*\_1>表示。其输出也是是一系列键对，把每一个键以及与之关联的值组成一个列表，这一过程称为数据重排（Shuffle），重排的结果用<*K*\_2, List(*V*\_2)>表示。Reduce函数的输入是<*K*\_2, List(*V*\_2)>，其输出是另一系列键值对，用<*K*\_3, *V*\_3>表示。

# 2.3 Spark

Spark[23]是另一个常用的开源大数据处理平台，它和Hadoop处理大数据的流程非常相似。Spark是在克服Hadoop不足的基础上提出的，它的首要设计目标是避免运算时出现过多的网络和磁盘I/O开销，为此它将核心数据结构设计为弹性分布式数据集RDD（Resident Distributed Dataset）。Spark使用RDD实现基于内存的计算框架，在计算过程中它会优先考虑将数据缓存在内存中，如果内存容量不足的话，Spark才会考虑将数据缓存到磁盘上，或者部分数据缓存到磁盘上进行处理。Spark为RDD提供了一系列算子，以对RDD进行有效的操作。此外，为了避免Hadoop启动和调度作业消耗过大的问题，Spark采用基于有向无环图DAG（Directed Acyclic Graph）的任务调度机制进行优化，这样可以将多个阶段的任务并行或串行执行，无需将每一个阶段的中间结果存储到HDFS（Hadoop Distributed File System）上。



Fig. 2 Big data processes with MapReduce

图2 MapReduce处理大数据的过程

# 3大数据主动学习

# 3.1 基于MapReduce的大数据主动学习

在大数据主动学习中，大数据是指无类别标签的数据集U是大数据，而有类别标签的数据集L是中小型数据集。大数据主动学习的基本思想依然是分而治之。具体地，将大数据集U划分为若干个子集，并部署到不同的云计算结点上，这些结点并行地选择重要的样例交给专家O进行标注。因为有类别标签的数据集L是中小型数据集，所以可以将L部署到每一个云计算结点，并在本地训练分类器C。在大数据主动学习的MapReduce编程实现中，分类器C用的是极限学习机（ELM, Extreme Learning Machine），样例重要性的度量用的是信息熵。大数据主动学习的基本思想如图3所示，在我们之前的工作中[21]，实现了基于MapReduce的大数据主动学习。



Fig. 3 The basic idea of big data active learning

图3 大数据主动学习的基本思想

# 3.2基于Spark的大数据主动学习

Spark处理大数据的逻辑是通过RDD实现的，基于Spark的大数据主动学习包括以下步骤：

第一步，初始化RDD，将有类别信息的数据转化为labeledRDD，无类别信息的数据转化为unlabeledRDD。

第二步，将有类别的数据（labeledRDD）广播至各个云计算节点。

第三步，对无类别信息的数据（unlabeledRDD）执行mapPartation操作，在每个分区中又执行如下操作：（1）用有类别的数据集训练一个分类器，在本文中用ELM作为分类器。（2）使用训练好分类器ELM计算无类别数据的信息熵。

第四步，根据信息熵值对无类别的数据按由大到小排序，选择K个熵值最大的样例作为本次迭代选择的样例，输出到HDFS。

第五步，对第四步中选择的样例，交给领域专家进行标注，并转化为RDD。

第六步，将有类别数据（labeledRDD）与第五步得到的RDD进行合并（union操作），将合并后的RDD作为更新后有类别数据（labeledRDD）。

迭代执行第一步至第六步，输出最后一次迭代得到的有类别数据（labeledRDD）至HDFS。

# 4基于两种开源平台的大数据主动学习的比较

对基于MapReduce和Spark的两种大数据主动学习算法从运行时间、文件数目、同步数目和耗费内存四个方面进行了实验比较。实验集群环境由5台计算机组成，其中1台为主节点，另外4台为从节点。操作系统是RedHat Linux 9.0，表1列出了5台计算机的主机名、机器模式、IP地址、CPU和内存信息。5台计算机都在同一局域网内，并通过端口速率为100Mbps的H3C S5100交换机连接。实验使用Hadoop和Spark版本分别是2.7.1和2.1.1。

实验比较所用的数据集包括4个UCI数据集和3个人工数据集，数据集的基本信息列于表2中。选择的数据集都是有类别信息的，为了模拟主动学习算法，我们选择一部分数据作为有类别信息的数据，剩余部分作为无类别信息的数据，也就是将类别信息隐藏。在Poker和Covtype中，有些类别的样例数很少，为了消除类别不平衡对训练分类器ELM的影响，删除了这些样例。另外，因为训练ELM分类器的需要，对数据进行了标准化处理。

Table 1 The basic information of experimental cluster environment

表1 实验集群环境的基本信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 主机名 | 计算机型号 | IP | CPU | 内存 |
| 主节点 | Dell PowerEdge R820 | 10.187.86.241 | Intel E5 2.20GHz | 2G |
| 从节点 | Dell PowerEdge R820 | 10.187.86.242-10.187.86.245 | Intel E5 2.20GHz | 2G |

Table 2 The basic information of experimental data sets

表2 实验所用数据集的基本信息

|  |  |  |  |
| --- | --- | --- | --- |
| 数据集 | 样例数 | 属性数 | 类别数 |
| Artificial 1 | 1000000 | 2 | 2 |
| Artificial 2 | 1200000 | 2 | 3 |
| Artificial 3 | 1000000 | 3 | 4 |
| HT-Sensor | 928991 | 11 | 3 |
| Covtype | 495141 | 54 | 2 |
| Poker | 923707 | 10 | 2 |
| SUSY | 5000000 | 18 | 2 |

第一个人工数据集（Artificial 1）是一个2类包含1000000个点的数据集，每类500000个点，每类都服从高斯分布，参数列于表3中。

Table 3 The parameters of the Gaussian distributions of the first artificial data set

表3 第一个人工数据集服从的高斯分布的参数

|  |  |  |
| --- | --- | --- |
| *i* |  |  |
| 1 |  |  |
| 2 |  |  |

第二个人工数据集（Artificial 2）是一个3类包含1200000个点的2维数据集，每类400000个点，3类服从的概率分布如下：

****

****

****

其中，**，。**

第三个人工数据集（Artificial 3）是一个3类包含1000000个点的3维数据集，每类250000个点，每类都服从高斯分布，参数列于表4中。

Table 4 The parameters of the Gaussian distributions of the third artificial data set

表4 第三个人工数据集服从的高斯分布的参数

|  |  |  |
| --- | --- | --- |
| *i* |  |  |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |

所用ELM分类器的超参数主要有输入权重、偏置和隐藏层节点个数。对于输入和偏置的初始化，选择标准正太分布进行随机初始化；对于隐藏层节点个数的设置，设置为输入层节点个数的2-15倍。此外，实验中还需设置初始训练分类器样本数量，对每类样本选择250个，考虑到每一个数据集的样例数和属性数不同，迭代次数和每次选择样例数设置也不同，超参数设置列于表5中。

因为基于两种开源大数据处理平台的主动学习算法思路一样，所以选择的样例数和选择的样例发布情况基本相同，实验结果也证明了这一点，用基于Hadoop和Spark的主动学习算法从数据集Artificial 1一次迭代选择的部分样例的分布如图4和图5所示。

Table 5 The configuration of the hyper parameters in ELM

表5 ELM分类器中超参数的设置

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据集 | 隐藏层节点数 | 初始训练集 | 迭代次数 | 每次选择样例数 |
| Artificial1 | 20 | 500 | 10 | 200 |
| Artificial2 | 25 | 500 | 10 | 200 |
| Artificial3 | 25 | 1000 | 10 | 200 |
| HT-Sensor | 25 | 750 | 15 | 150 |
| Covtype | 110 | 500 | 20 | 100 |
| Poker | 20 | 500 | 20 | 100 |
| SUSY | 40 | 500 | 20 | 100 |

Fig. 4 The distribution of some instances selected from the data set artificial 1 in the first iteration with the Hadoop based active learning algorithm

图4 用基于Hadoop的主动学习算法从数据集Artificial 1一次迭代选择的部分样例的分布

Fig. 5 The distribution of some instances selected from the data set artificial 1 in the first iteration with the Spark based active learning algorithm

图5 用基于Spark的主动学习算法从数据集Artificial 1一次迭代选择的部分样例的分布

图6和图7显示的是基于两种开源平台的大数据主动学习算法在人工数据集和UCI数据集上运行时间的比较，纵坐标代表主动学习算法中一次迭代过程的平均时间。图8和图9显示的是基于两种开源平台的大数据主动学习算法在人工数据集和UCI数据集上内存占用的比较，纵坐标代表主动学习算法中一次迭代过程的内存平均占用。

从实验结果可以看出，在运行时间上，基于Spark的大数据主动学习算法比基于Hadoop的大数据主动学习算法快3-7倍。对于人工数据集，数据规模大小相近，基于Spark的大数据主动学习算法一般比基于Hadoop的大数据主动学习算法快4倍左右；对于UCI数据集，数据规模大小相差较大，所以在运行时间上相差较大。在内存使用上，基于Hadoop的大数据主动学习算法内存使用大小由数据集规模决定，数据集规模越大，内存使用越多，例如数据集HT-Senor仅使用340 MB内存，而数据集SUSY使用了1332 MB内存；基于Spark的大数据主动学习算法内存使用情况与之不同，虽然相对于不同数据集，内存使用情况不同，却相差不大，把内存充分利用起来（即达到集群中每个从节点的内存限制）。同时，我们在记录内存使用情况时，发现Hadoop平台内存使用量会随着迭代次数的增加而周期性地增加和减少；对于Spark平台，内存使用量仅随着迭代次数的增加呈现周期性逐渐增加，但是每个循环都没有峰值。

Fig. 6 The comparison of running time on 3 artificial data sets

图6 在3个人工数据集上运行时间的比较

Fig. 7 The comparison of running time on 4 UCI data sets

图7 在4个CUI数据集上运行时间的比较

Fig. 8 The comparison of memory requirements on 3 artificial data sets

图8 在3个人工数据集上内存占用的比较

Fig. 9 The comparison of memory requirements on 4 UCI data sets

图9 在4个UCI数据集上内存占用的比较

对于实验结果，受MapReduce和Spark比较工作[23]的启发，我们通过一个包含6个样例的例子，从原理上对两种算法进行对比分析。在这个例子中，MapReduce和Spark的处理过程如图10和图11所示。MapReduce作业的执行过程主要分为Map阶段、中间结果排序与传递阶段和Reduce阶段。MapReduce作业执行过程受读取输入文件时间Tread、中间数据排序时间Tsort、中间数据传递时间Ttrans和写输出文件到HDFS时间Twrite影响。因为两种算法的输入/输出数据是相同的，且主要比较的是MapReduce与Spark运行机制以及调度策略不同所导致的运行时间大小，所以不考虑网络传输速度以及文件读写速度的因素，分析过程中，默认Tread和Twrite在两种平台的值相同，主要关注Tsort和Ttrans的比较。



Fig. 10 The illustration of big data active based on MapReduce by an example

图10 用一个例子说明基于MapReduce的大数据主动学习



Fig. 10 The illustration of big data active based on Spark by an example

图10 用一个例子说明基于Spark的大数据主动学习

关于两种算法的中间数据排序时间Tsort与中间数据传输时间Ttrans。MapReduce规定每次Shuffle必须对中间结果进行排序，主要是为了将中间结果进行初步的归并操作，使得需要传输的数据减少，降低网络传输压力；并且可以保证每个Map任务只输出一个有序的中间数据文件，减少文件数目。在MapReduce中，在Map阶段对每一分区的数据进行排序（本例只有一个分区，见图10），在Reduce阶段对不同Map任务的输出结果进行归并。假设共有*n*个Map任务，平均每个Map任务有*N*条数据，平均每个Reduce任务有*R*条数据。可以得到Tsort-MR=*N*\*log*N*+*R*=O(*N*log*N*)。在Spark中，主要是对每一分区的数据进行排序（本例有两个分区，见图11），如果把一个分区看做MapReduce中的一个Map，并且假设条件相同，可以得到Tsort-Spark=O(*N*log*N*)。

中间数据的传输是指由Map任务的执行节点发送到Reduce任务的执行节点的数据，所以Ttrans由Map任务输出的中间数据的大小|D|和网络文件传输速度Ct决定。在不考虑网络传输速度带来的性能差异，默认在MapReduce和Spark中Ct大小相等，则Tsort与|D|成正比。在该例子中，MapReduce和Spark中间数据都有6条数据，所以两者中间数据传输时间也相同，但这是在计算熵值没有相同情况下，如果熵值有相同的，MapReduce中间数据会小于Spark，则相应Spark的Ttrans大于MapReduce。

对于中间结果需要缓存的文件数目，在分布式系统中，中间数据是以文件的形式进行存储的。文件数目过多，会严重占用内存并影响磁盘的I/O性能。对于MapReduce，每一个Map只会产生一个中间数据文件，不同分区的数据都会存在一个文件中，之所以可以做到这样，是因为MapReduce的排序操作使得分区内数据有序，不同的分区数据只需要通过增加一个偏移量便可以区分。所以在MapReduce中，文件数目等于Map任务数量*m*；而在Spark中，因为没有对数据进行预排序，所以只能将不同分区的数据放在不同的文件中，则每一个Map任务都会生成*r*个文件，其中*m*为Reduce任务数量。则Spark总文件数目等于*m*×*r*。在本例中，MapReduce和Spark的文件数目都为3。如果增加Reduce任务数量，Spark的中间文件数目会远远大于MapReduce。

关于同步次数，同步模型要求所有节点完成当前阶段后才可以进行下一阶段，这严重限制了计算性能。在MapReduce中，所有的步骤都严格遵守同步模型。即，Reduce操作要在所有的Map操作结束后进行。在算法执行过程中，同步次数越小、所占比例越小，越有利于算法的局部性能。在每次迭代过程中，MapReduce与Spark的同步次数皆为1。

综合以上分析可知，Spark在运行时间上优于MapReduce，主要是因为Spark在内存足够的情况下，允许将常用的数据缓存到内存中，加快了系统的运行速度。由图10和图11可知，MapReduce每次迭代将中间结果写入磁盘，如阴影部分所示，而Spark在第一次迭代读取数据后，不再将中间结果写入磁盘，存储在内存中，内存使用一直增加直至迭代任务结束。这也是Spark使用内存远远大于MapReduce的原因。

# 5 结束语

本文将我们之前提出的基于MapReduce的大数据主动学习算法移植到Spark环境，并对基于两种开源平台的大数据主动学习算法进行了比较研究。在3个人工数据集和4个UCI数据集进行了实验，通过记录比较两者的运行时间和内存使用，并从原理上进行了详细分析，发现Spark通常比Hadoop更快，但是消耗更多的内存。对于迭代任务，应先考虑使用Spark，但是要保证足够内存，究竟有多少内存就足够了，取决于特定的迭代算法和处理的数据集的规模。因为迭代过程中的中间结果以RDD形式存储在内存中，由于RDD的只读性质，将在每次迭代中创建新的RDD，所以很难确定在Spark上运行的迭代任务的确定内存量。

参 考 文 献

1. Angluin D. Queries and concept learning[J]. Machine Learning, 1988, 2(4):319-342.
2. Lewis D D , Gale W A . A sequential algorithm for training text classifiers[C]// Proceedings of the 17th ACM International Conference on Research and Development in Information Retrieval, Berlin: Springer, New York, 1994, 3-12.
3. Seung H. S, Opper, et al. Query by committee[C]// Proceedings of the fifth annual workshop on Computational Learning Theory, 1992, 287-294.
4. Schohn G , Cohn D . Less is More: Active Learning with Support Vector Machines[C]// Proceedings 17th International Conference on Machine Learning, Morgan Kaufmann, San Francisco, CA, 2000, 839-846.
5. Tong S , Koller D . Support vector machine active learning with applications to text classification[J]. Journal of Machine Learning Research, 2002, 2(1):999-1006.
6. Yang Y Z, Loog M. A benchmark and comparison of active learning for logistic regression[J]. Pattern Recognition, 2018, 83:401-415.
7. Yang Y Z, Loog M. A variance maximization criterion for active learning[J]. Pattern Recognition, 2018, 78:358-370.
8. Yang Y Z, Loog M. Single shot active learning using pseudo annotators[J]. Pattern Recognition, 2019, 89:22-31.
9. Hsu, H L, Chang, Y C I, Chen R B. Greedy active learning algorithm for logistic regression models[J]. Computational Statistics & Data Analysis, 2019, 129:119-134.
10. Kee S, del Castillo E, Runger G. Query-by-committee improvement with diversity and density in batch active learning[J]. Information Sciences, 2018,454-455:401-418.
11. Vandoni J, Aldea E, Hégarat-Mascle S L. Evidential query-by-committee active learning for pedestrian detection in high-density crowds[J]. International Journal of Approximate Reasoning, 2019, 104:166-184.
12. Wu, K L, Cai D, He X F. Multi-label active learning based on submodular functions[J]. Neurocomputing, 2018, 313:436-442.
13. Reyes, O, Morell C, Ventura S. Effective active learning strategy for multi-label learning[J]. Neurocomputing, 2018, 273:494-508.
14. Altalhi A H, Ventura S. Statistical comparisons of active learning strategies over multiple datasets[J]. Knowledge-Based Systems, 2018, 145:274-288.
15. Pereira-Santos, D, Prudêncio, R B C, de Carvalho A C P L F. Empirical investigation of active learning strategies[J]. Neurocomputing, 2019, 326-327:15-27.
16. Burbidge R, Rowland J J, King R D. Active Learning for Regression Based on Query by Committee [C]// International Conference on Intelligent Data Engineering and Automated Learning, 2007, 209-218.
17. Wu, D R, Lin C T, Huang J. Active learning for regression using greedy sampling[J]. Information Sciences, 2019, 474:90-105.
18. Demir, Begüm, Bruzzone L . A multiple criteria active learning method for support vector regression[J]. Pattern Recognition, 2014, 47(7):2558-2567.
19. Li X H, Jian Z. Active Learning for Regression With Correlation Matching and Labeling Error Suppression[J]. IEEE Signal Processing Letters, 2016, 23(8):1081-1085.
20. Cai W B, Zhang M H, Zhang Y. Batch Mode Active Learning for Regression With Expected Model Change[J]. IEEE Transactions on Neural Networks and Learning Systems, 2017, 28(7):1668-1681.
21. 翟俊海, 张素芳, 王聪, 等. 基于MapReduce的大数据主动学习. 计算机应用, 2018, 38(10):2759-2763.

(Zhai Junhai, Zhang Sufang, Wang Cong, et al. Big Data Active Learning Based on MapReduce[J]. Journal of Computer Applications, 2018, 38(10):2759-2763.)

1. Dean J, Ghemawat S. MapReduce: Simplified Data Processing on Large Clusters. Communications of the ACM, 2008,51(1),107-113.
2. 吴信东, 嵇圣硙. MapReduce与Spark用于大数据分析之比较[J]. 软件学报, 2018, 29(06):260-281.

(Wu Xindong, Ji Shengwei. Comparative study on MapReduce and Spark for big data analytics. Journal of Software, 2018, 29(6):1770-1791.)