基于MapReduce检测僵尸网络的贝叶斯算法的实现

邵秀丽 耿梅洁 蒋鸿玲
(南开大学信息技术科学学院 天津 300071)

摘 要 利用贝叶斯算法检测僵尸网络具有较高的准确性,但僵尸网络具有流量大的特征,同时贝叶斯分类训练阶段需要对大量的网络数据集进行训练,用单一结点来检测僵尸网络将会遇到计算时间和计算资源瓶颈。为此设计了基于MapReduce检测僵尸网络的贝叶斯算法,把贝叶斯算法训练阶段的先验概率、条件概率和检测阶段的后验概率的计算并行化处理。通过大量运行在Hadoop平台上的实验表明,该方法提高了检测僵尸网络的效率。

关键词 僵尸网络,贝叶斯,MapReduce,流量,Hadoop
中图法分类号 TP393 **文献标识码** A

Realization of Bayesian Algorithm for Detecting Botnets Based on MapReduce

SHAO Xiu-li GENG Mei-jie JIANG Hong-ling
(College of Information Technical Science, Nankai University, Tianjin 300071, China)

Abstract Although botnets are detected in a more accurate way by using Bayesian algorithm, it has the character of large flow and the training of Bayesian classification needs to train a large number of network datasets. Therefore, it will lead to meet a bottleneck of calculation of the time and resources by using a single node to detect the botnets. To this end, this paper designed a Bayesian algorithm based on the MapReduce to parallely process the calculation of the prior probability and the conditional probability in the training phase, and the posterior probability in the detection phase of Bayesian algorithm. A large number of experiments running on Hadoop platform show that this method improves the efficiency of bonnets detecting.

Keywords Botnets, Bayesian, MapReduce, Flow, Hadoop

1 引言

僵尸网络是攻击者通过命令与控制信息,控制其已经攻占的计算机。目前主流的僵尸网络检测方法是通过分析网络流量来进行检测的,已有的僵尸网络检测技术有通过昵称检测IRC类型[1]的僵尸网络;通过网络通信图来识别P2P网络[2],但需要利用已知的信息区分合法P2P网络与P2P僵尸网络;通过PageRank算法计算主机级别,再根据外部系统提供的僵尸网络信息进行检测;通过识别僵尸网络的恶意行为检测僵尸网络。这些僵尸网络检测方法针对Http僵尸网络检测技术相对较少[3];或是已有的一些检测方法依赖僵尸主机的恶意行为,在僵尸主机处于"发呆"状态下不能检测出僵尸网络;或是已有的一些检测算法需要先验知识,不能满足大规模网络的检测需要[4]。

僵尸网络数据与正常网络相比网络属性会发生变化,可通过分析网络异常采用分类的方法检测僵尸网络。将网络数据分为两类:一个为正常网络,另一个为僵尸网络。为提高正确率、降低检测时间,常利用神经网络、聚类等分类算法检测僵尸网络[5]。然而比较各种分类算法,贝叶斯算法是基于统计学习的分类技术,可以通过训练样本得到先验知识,虽然需要对阈值做假定,但从大量实验得出贝叶斯算法不仅具有较强的分类能力、正确率较高,而且既可以检测IRC类型的僵尸网络又可以检测Http僵尸网络[6]。

但由于僵尸程序不断向僵尸主机发送命令,僵尸主机做出响应,造成额外的网络流量开销,使得僵尸网络具有流量大的特征,因此,贝叶斯算法训练阶段需要对大量的网络数据集进行训练,需要较大的系统开销[7]。尽管利用现有的贝叶斯算法检测僵尸网络具有较高的准确率,但会遇到计算时间和计算资源的瓶颈。近年来出现的MapReduce编程框架主要针对大规模群组中的海量数据处理[8],因此可以运用MapReduce框架解决基于贝叶斯算法检测僵尸网络的问题[9]。本文设计的基于MapReduce检测僵尸网络的贝叶斯算法能有效地提高检测效率并且避免遇到计算时间与资源的瓶颈。

2 数据预处理

为了得到检测性能好的贝叶斯分类器并让数据适用于MapReduce框架,首先要准备数据并进行预处理,使训练数据能体现出僵尸网络流量特征,区分出正常流量和僵尸流量。

到稿日期:2013-03-27 返修日期:2013-07-22 本文受天津市重点资助项目(11jczdjc28100),国家科技支撑计划资助项目(2012BAF12B00)资助。

邵秀丽(1963-),女,教授,博士生导师,主要研究方向为云计算与软件工程等,E-mail:shaoxl@nankai.edu.cn;**耿梅洁**(1988-),女,硕士生,主要研究方向为云计算;**蒋鸿玲**(1986-),女,博士生,主要研究方向为网络安全与云计算。

本文实验使用的网络访问流量信息是从某高校校园网核心交换机上采集的以 3 次握手开始、4 次握手结束的数据包。分两步对采集的流量信息进行预处理，首先将捕获的网络流量信息数据包文件处理为由若干行构成的文本格式存储在文件 file 中，使得每行表示一条网络访问信息，以便后续程序的分析处理。然后利用云环境的 Hadoop 完成数据分块工作，以便基于 MapReduce 的并行化贝叶斯算法的计算处理工作。

前者形成的文本文件中的每一行信息都有共同的特征，例如，如果分析 TCP 协议的流量，则数据包中包括 TCP 协议的标志位等如下信息：

“序号|数据包到达时间|源 IP 地址|目的 IP 地址|TCP 数据流|时间间隔平均值|时间间隔变化|数据包字节数|数据包个数平均值|持续时间平均值|类标签”。

第二步的工作是首先对信息进行初筛，即把 file 中每行的信息分类为正常网络和僵尸网络数据，依据 file 中的每行信息是否是可疑数据，在每行后增加一列信息，即类标签值为 1 或者 0 来标明该条网络数据是否属于僵尸网络，本文设类标签值为 0 的网络为正常网络，否则为僵尸网络。

然后把 file 分为两部分，即从 file 中随机选择类标签值为 0 的正常网络信息行的 2/3，再随机选择类标签值为 1 的僵尸网络信息行的 2/3，这些行信息合成文件作为训练数据文件 file1，剩余数据行作为检测数据文件 file2。file1 被用来通过贝叶斯算法训练形成知识库，file2 被用来检测是否为僵尸网络。

3 检测僵尸网络行为的贝叶斯算法的设计

首先采用大量的训练数据训练贝叶斯分类器，然后用训练好的贝叶斯分类器检测僵尸网络。图 1 为基于贝叶斯的僵尸网络检测模型。

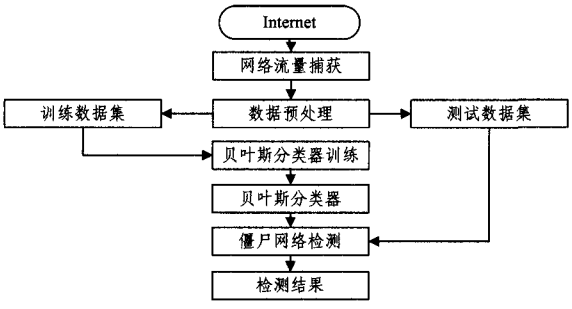

图 1 基于贝叶斯检测僵尸网络模型

本文基于贝叶斯的僵尸网络检测过程如下：

1)捕获网络流量。主要获取被测网络内部主机与外部主机通信的数据包。2)数据预处理。将捕获的网络信息处理成文本文件。3)形成训练数据集和测试数据集。把预处理后的文本文件分为两部分，一部分为训练数据集，另一部分为测试数据集。4)贝叶斯分类器训练。利用具有类标签的训练数据集对贝叶斯分类器进行训练，得到训练好的贝叶斯分类器。5)僵尸网络检测。利用训练好的贝叶斯分类器检测测试数据集中的僵尸网络。

数据预处理后得到的 file 文件中一行数据 d 可解析为用一个六维特征向量 $d=(w_1,w_2,\cdots,w_6)$ 表示，其中，6 个属性分别表示数据的 6 个特征，w_1 为 TCP 数据流、w_2 为时间间隔平均值、w_3 为时间间隔变化、w_4 为数据包字节数、w_5 为数据包个数平均值、w_6 为持续时间平均值。

图 2 给出了对这些处理的网络访问流量信息，贝叶斯算法检测僵尸网络的流程。假定有两类网络访问信息流 C_0，C_1，其中，C_0 为正常网络，C_1 为僵尸网络。给定一行数 d，当 $P(C_0|d)>P(C_1|d)$ 时，d 属于正常网络，否则，属于僵尸网络。

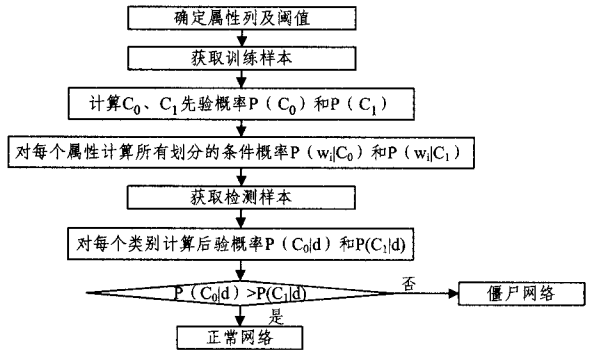

图 2 检测僵尸网络贝叶斯算法流程图

根据贝叶斯定理，计算正常网络、僵尸网络的 $P(C_k|d)$ 采用相同公式：$P(C_k|d)=\dfrac{P(C_k)*P(d|C_k)}{P(d)}$；$k=0,1$，而 $P(d)$ 对于一条数据是固定的。因此，要计算 $P(C_k|d)$ 只需算出 $P(d|C_k)*P(C_k)$ 即可[10]。其中的先验概率 $P(C_k)$，可通过训练样本得到，即 $P(C_0)=\dfrac{S0}{S}$，$P(C_1)=\dfrac{S1}{S}$。S_0 是正常网络数据个数，S_1 是僵尸网络数据个数，S 是训练数据总数。各属性是否在各阈值内且属于僵尸网络的条件概率也可通过训练样本得到，即 $P(w_i|C_0)=\dfrac{S_{i0}}{S_0}$，$P(w_i|C_1)=\dfrac{S_{i1}}{S_1}(1\leqslant i\leqslant 6)$，$S_{i0}$ 是属性 w_i 在阈值内且属于正常网络的数据个数，S_{i1} 是属性 w_i 在阈值内且属于僵尸网络的数据个数。

朴素贝叶斯假定属性之间是相互独立的，因此，在检测阶段，计算检测样本在正常网络及僵尸网络条件下的概率方法，$P(d|C_k)=P(w_1|C_k)*P(w_2|C_k)*\cdots*P(w_6|C_k)$。检测阶段由计算出的先验概率 $P(C_k)$ 及条件概率 $P(d|C_k)$，来求检测数据的后验概率 $P(C_k|d)$。比较正常网络和僵尸网络后验概率大小，判断是否为僵尸网络。

4 基于 MapReduce 的贝叶斯算法的设计

贝叶斯算法耗用大量的计算资源，用 MapReduce 把贝叶斯算法训练阶段求先验概率、条件概率和检测阶段计算后验概率分别进行并行化处理，以缩短训练和检测过程，从而提高检测僵尸网络的性能。

4.1 基于 MapReduce 检测僵尸网络的贝叶斯算法设计

基于 MapReduce 实现僵尸网络行为检测的贝叶斯算法需要先作数据处理工作，即 Hadoop 接收前面数据预处理所形成的 file1、file2 文件，由 Hadoop 自动地将其处理成 64M 大小的若干数据分块 split，分别以文件 $spliti\{i=1,\cdots,n\}$、$split\_i\{i=1,\cdots,n\}$ 形式存储，并将每个文件按行分割形成〈key,value〉对，其中 key 为该行的首字母相对于文本文件的首地址的偏移量，value 值存储的是文本文件中的一行信息。

图 3 是基于 MapReduce 检测僵尸网络的贝叶斯算法流程图。把贝叶斯的训练阶段和检测阶段分别由不同的 MapReduce 处理。在贝叶斯训练阶段通过统计 file1 文件中最后标签列的值来计算贝叶斯算法中的先验概率和条件概率值，

后者涉及阈值内、外的判断，因此，为了方便计算，设计了两个 MapReduce 来协同完成贝叶斯算法的训练阶段的工作任务 MapReduce1 和 MapReduce2。其中，设计 MapReduce1 用于计算正常网络和僵尸网络的先验概率，设计 MapReduce2 用于计算 value 所含的 6 个特征项的条件概率。MapReduce1 的输出为 2 个先验概率，由于 MapReduce2 要对训练数据的 6 个属性列进行训练，每个属性既要判断是否为僵尸网络又要判断是否在阈值内，因此每个属性有 4 个判断条件。所以 MapReduce2 的输出共有 24 个条件概率，这样一来总共 26 个条件概率就构成了知识库，供检测使用。设计了 MapReduce3 来实现贝叶斯检测阶段的工作任务，以完成僵尸网络的检测。

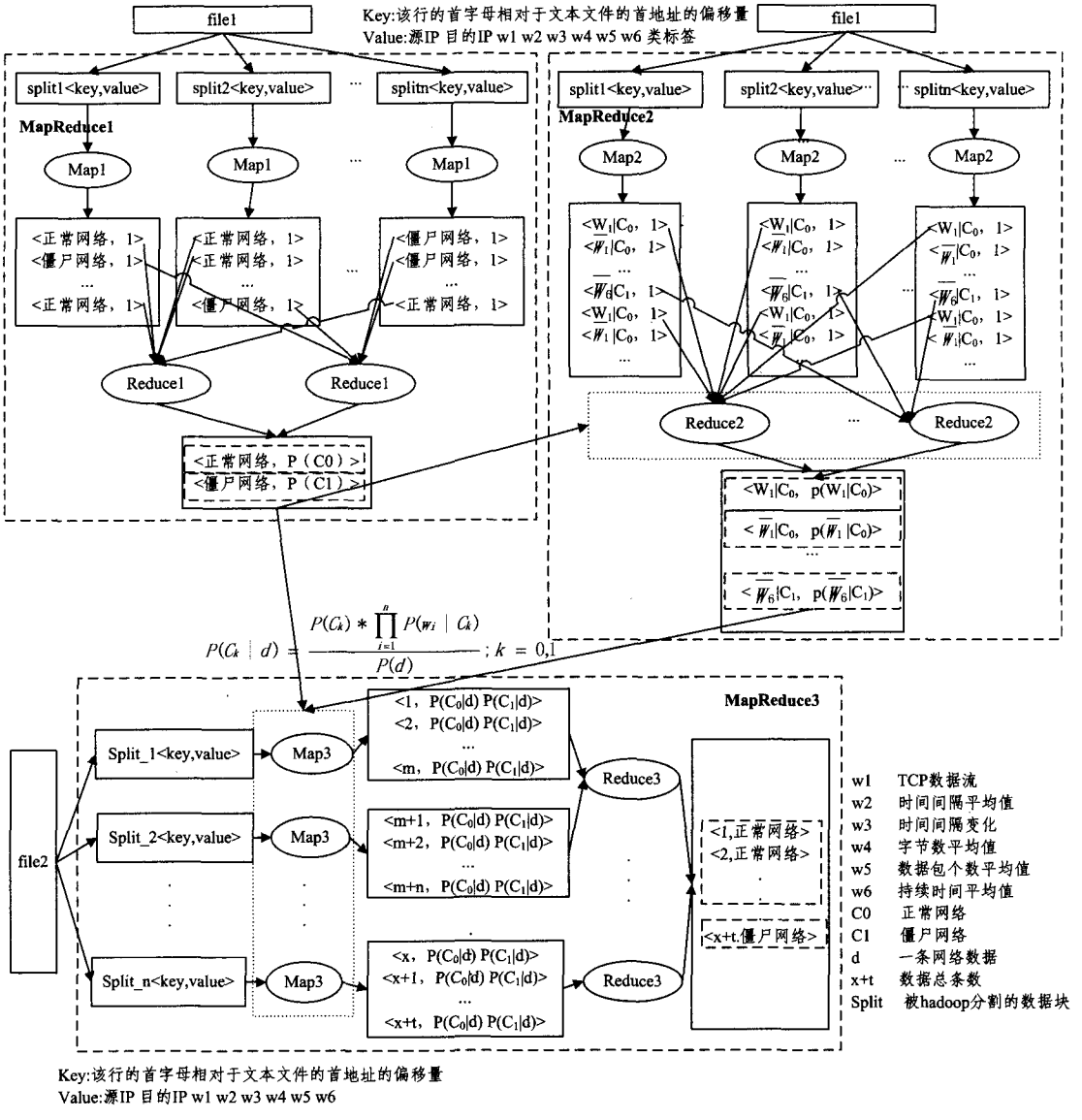

图 3　基于 MapReduce 检测僵尸网络的贝叶斯算法流程

4.2　计算先验概率的 MapReduce 算法设计

MapReduce1 计算 Hadoop 自动处理后的文件 $spliti\{i=1,\cdots,n\}$ 中信息的正常网络和僵尸网络的先验概率。其中 Map1 接收到 Hadoop 自动形成的形式为〈key，value〉的信息，将每行 value 按空格分隔成字符串数组，数组最后一项为类标签值。重新计算 key，value，即〈key1，value1〉，作为 Map1 的输出键值对。重新计算 key 的方法是：若类标签值为 0，则把 key1 值设为“正常网络”；若类标签值为 1，则把 key1 值设为“僵尸网络”。重新计算 value 的方法是：将 value1 值设为“1”，即出现一次。

经过 Map1 把 *spliti* 的每行信息都处理成以〈正常网络，1〉或〈僵尸网络，1〉形式的等待整体处理的中间文件 A 输出，然后 MapReduce 框架将每个 Map1 输出的中间文件 A 中的中间结果〈正常网络，1〉或〈僵尸网络，1〉按照 key 值（正常网络、僵尸网络）进行分组形成新的 key，value，即〈key2，value2〉，其中 key2 为类标签值（正常网络，僵尸网络），value2 是对应的类标签值的计数值所组成的列表。将相同类标签的中间结果数据分配给同一个 Reduce1。

其中，Reduce1 的数量是由 MapReduce 框架程序设定的，Reduce1 个数可以与分组个数相同，即每个分组对应一个 Reduce1。若程序设置的 Reduce1 个数不够给每个分组分配一个，则有可能多个分组共用一个 Reduce1。

Reduce1 的输入为〈key2，value2〉，形式为（正常网络，[1，1，…，1]）或（僵尸网络，[1，1，…，1]）。Reduce1 按照 value2 的值汇总求和，即可得出正常网络和僵尸网络各自总次数，分别用于求得正常网络先验概率 $P(C_0)$ 和僵尸网络先验概率 $P(C_1)$。图 4 给出了执行过程。

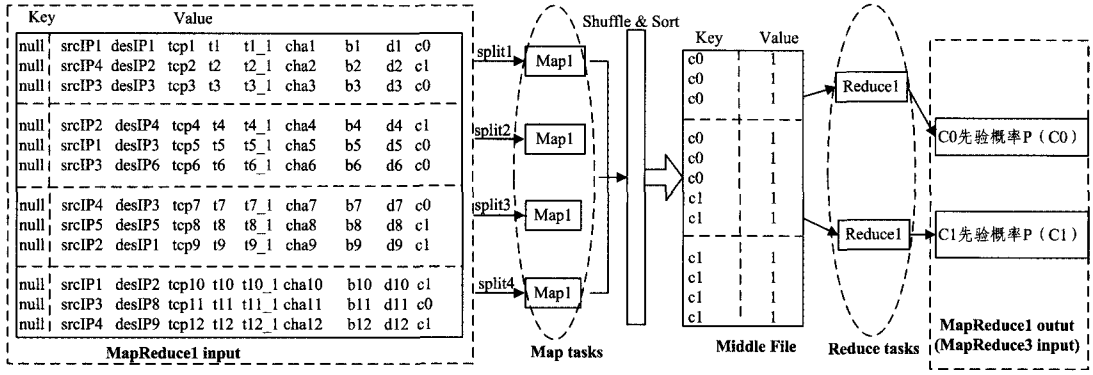

图 4　MapReduce1 程序运行示例

经 MapReduce1 处理后，形成两个概率：正常网络先验概率和僵尸网络先验概率。它们构成知识库的一部分，供检测阶段使用。

4.3　计算条件概率的 MapReduce 算法设计

MapReduce2 计算 Hadoop 自动处理后的每个 $spliti\{i=1,\cdots,n\}$ 的条件概率。MapReduce2 处理过程与 MapReduce1 类似，即 Map2 接收到形式为〈key，value〉的信息。将每行 value 按空格分隔成字符串数组，数组由 7 项组成，分别为：W_1（TCP 数据流）、W_2（时间间隔平均值）、W_3（时间间隔变化）、W_4（字节数平均值）、W_5（数据包个数平均值）、W_6（持续时间平均值）、类标签值。

重新计算 key，value，即〈key3，value3〉，作为 Map2 的输出键值对。重新计算 key 的方法是：判断属性列的值是否在各自的阈值内并且属于正常网络。若类标签值为 0 且属性列在各阈值内，把 key3 值表示为“$W_i|C_0(1\leqslant i\leqslant 6)$”；若类标签值为 0 且属性列在阈值外，把 key3 值表示为“$\overline{W_i}|C_0(1\leqslant i\leqslant 6)$”；若类标签值为 1 且属性列在阈值内，把 key3 值表示为“$W_i|C_1(1\leqslant i\leqslant 6)$”；若类标签值为 1 且属性列在阈值外，把 key3 值表示为“$\overline{W_i}|C_1(1\leqslant i\leqslant 6)$”。重新计算 value 的方法是：将 value3 值设为“1”，即出现一次。

经过 Map2 把 spliti 的每行信息都处理成以〈$W_1|C_0$，1〉、〈$\overline{W_1}|C_0$，1〉、〈$W_1|C_1$，1〉、〈$\overline{W_1}|C_1$，1〉…〈$\overline{W_6}|C_1$，1〉形式的等待整体处理的中间文件 A 输出，然后 MapReduce 框架将每个 Map2 输出的中间文件 A 中的中间结果〈$W_1|C_0$，1〉、〈$\overline{W_1}|C_0$，1〉、〈$W_1|C_1$，1〉、〈$\overline{W_1}|C_1$，1〉…〈$\overline{W_6}|C_1$，1〉按照 key 值（$W_1|C_0$、$\overline{W_1}|C_0$、$W_1|C_1$、$\overline{W_1}|C_1\cdots\overline{W_6}|C_1$）进行分组形成新的 key，value，即〈key4，value4〉，其中 key4 为条件字符串（$W_1|C_0$、$\overline{W_1}|C_0$、$W_1|C_1$、$\overline{W_1}|C_1\cdots\overline{W_6}|C_1$），value4 是对应的类标签值的计数值所组成的列表。将相同类标签的中间结果数据分配给同一个 Reduce2。

其中，Reduce2 的数量是由 MapReduce 框架程序设定的，Reduce2 个数可以与分组个数相同，即每个分组对应一个 Reduce2。若程序设置的 Reduce2 个数不够给每个分组分配一个，则有可能多个分组共用一个 Reduce2。

Reduce2 的输入为〈key4，value4〉，形式为（$W_1|C_0$，[1，1，…，1]）、（$\overline{W_1}|C_0$，[1，1，…，1]）、…（$\overline{W_6}|C_1$，[1，1，…，1]）。Reduce2 按照 value4 的值汇总求和，即可得出条件字符串各自出现的总次数，将其分别用于求得各条件概率 $P(W_1|C_0)$、$P(\overline{W_1}|C_0)$、$P(W_1|C_1)$、$P(\overline{W_1}|C_1)$、…、$P(\overline{W_6}|C_1)$。具体执行过程如图 5 所示。

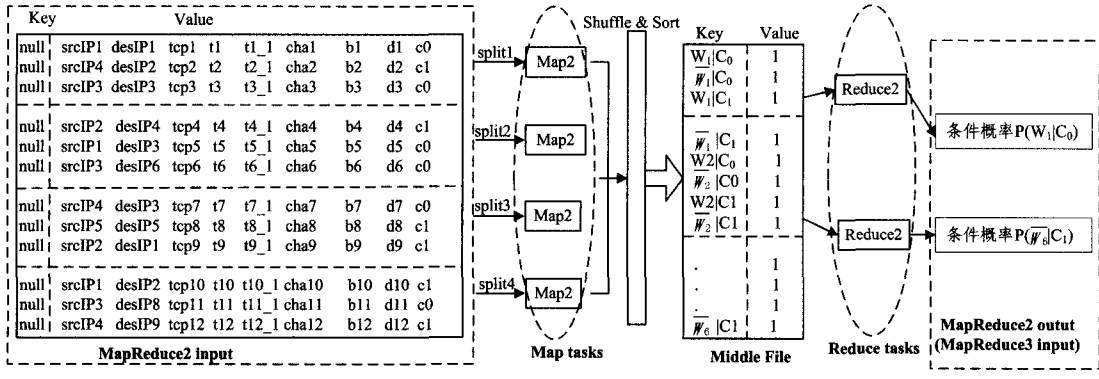

图 5　MapReduce2 程序运行示例

由于 MapReduce2 要对训练数据的 6 个属性列进行训练，每个属性既要判断是否为僵尸网络又要判断是否在阈值内，因此每个属性有 4 个判断条件。所以，经过 MapReduce2 处理后，形成 24 个条件概率，构成知识库的一部分。至此，知识库已经完全形成，可用于检测僵尸网络。

4.4　检测僵尸网络的 MapReduce 算法设计

设计 MapReduce3 实现贝叶斯算法的检测阶段的任务，MapReduce3 基于训练阶段形成的由 26 个条件概率构成的知识库，根据 Hadoop 自动处理后的每个 $split\_i\{i=1,\cdots,n\}$ 信息计算后验概率，以检测是否为僵尸网络。其中 Map3 接收到 Hadoop 自动形成的形式为〈key，value〉的信息，将每行 value 按空格分隔成字符串数组，数组由 7 项组成，分别为：序号、W_1（TCP 数据流）、W_2（时间间隔平均值）、W_3（时间间隔变化）、W_4（字节数平均值）、W_5（数据包个数平均值）、W_6（持续时间平均值）。

重新计算 key，value，即〈key5，value5〉，作为 Map3 的输出键值对。重新计算 key 的方法是：将 key5 值设为序号值。重新计算 value 的方法是：value5 是由正常网络后验概率和僵

尸网络后验概率两个概率以空格连接形成的字符串。要计算正常网络后验概率 $P(C_0|d)$ 和僵尸网络后验概率 $P(C_1|d)$（d 为一行数据）的大小，就要计算后验概率大小，即计算 $P(C_0)\prod_{i=1}^{n}P(W_i|C_0)$ 和 $P(C_1)\prod_{i=1}^{n}P(W_i|C_1)$ 的大小。而 $P(C_0)$ 和 $P(C_1)$ 在知识库已经存在，现在只需求 $\prod_{i=1}^{n}P(W_i|C_0)$ 和 $\prod_{i=1}^{n}P(W_i|C_1)$ 的值。判断属性列的值是否在各自的阈值内，若在阈值内，则使用知识库的条件概率 $P(W_i|C_0)$、$P(W_i|C_1)(i=1,2,\cdots,6)$；若在阈值外，使用 $P(\overline{W_i}|C_0)$、$P(\overline{W_i}|C_1)(i=1,2,\cdots,6)$。$P(C_0)$、$P(C_1)$、$\prod_{i=1}^{n}P(W_i|C_0)$ 和 $\prod_{i=1}^{n}P(W_i|C_1)$ 都已经求得，因此根据贝叶斯公式可以计算 $P(C_0|d)$ 和 $P(C_1|d)$。

经过 Map3 把 *split_i* 的每行信息都处理成以〈序号，$P(C_0|d)P(C_1|d)$〉形式的等待整体处理的中间文件 A 输出，然后 MapReduce 框架将每个 Map3 输出的中间文件 A 中的中间结果〈序号，$P(C_0|d)P(C_1|d)$〉按照 key 值（序号）进行分组后发送给 Reduce3。

其中，Reduce3 的数量是由 MapReduce 框架程序设定的，Reduce3 个数可以与分组个数相同，即每个分组对应一个 Reduce3。若程序设置的 Reduce3 个数不够给每个分组分配一个，则有可能多个分组共用一个 Reduce3。

Reduce3 的输入为〈key5，value5〉，形式为〈序号，$P(C_0|d)P(C_1|d)$〉。Reduce3 比较 $P(C_0|d)$ 和 $P(C_1|d)$ 大小，若 $P(C_0|d)>P(C_1|d)$，则 d 属于 C_0 类，即正常网络；否则，d 属于 C_1 类，即僵尸网络。具体执行过程如图 6 所示。

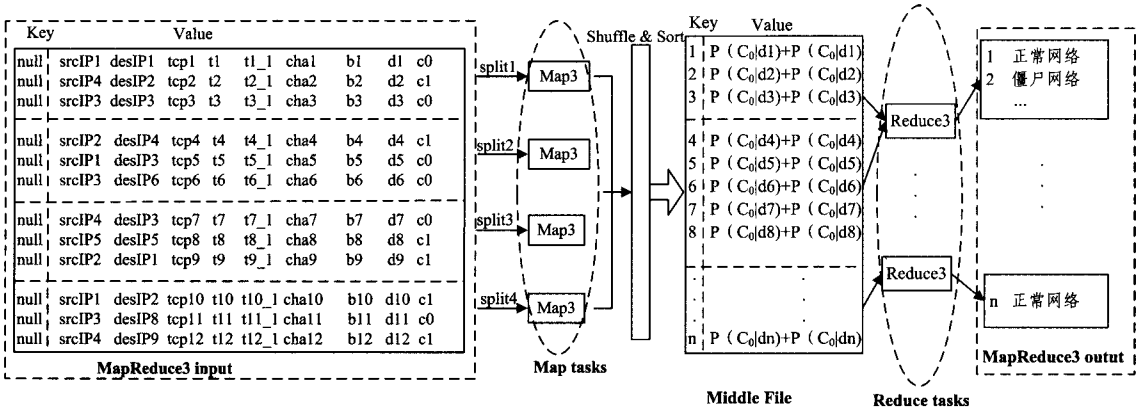

图 6 MapReduce3 程序运行示例

经过 MapReduce3 处理后，可以判断数据是否属于僵尸网络，进而完成检测。

5 实验分析

5.1 检测效率随数据量变化的情况

为测试不同数据量对效率的影响，将训练数据和检测数据再分别划分。数据集 1 选取训练数据前 1/3 行作为新的训练数据，检测数据前 1/3 行作为新的检测数据；数据集 2 选取训练数据前 2/3 行作为新的训练数据，检测数据前 2/3 行作为新的检测数据；数据集 3 是原训练数据和原检测数据。实验 A 是在普通的 PC 单机上采用贝叶斯分类检测僵尸网络，实验 B 是在 Hadoop 集群上采用 1 个节点基于 MapReduce 的贝叶斯分类来检测僵尸网络。比较不同数据量在实验 A、B 中检测所需的时间。实验结果分别以表 1 和图 7 折现图表示。

表 1 不同数据集检测时间比较（单位：毫秒）

	数据集 1	数据集 2	数据集 3
实验 A	15657.0	27504.0	52909.0
实验 B	11967.0	18716.0	26829.0

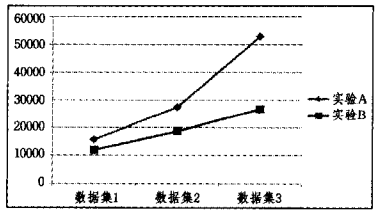

图 7 不同数据集检测时间比较

图 7 中的横坐标代表不同的数据集，数据集 1 中所包含的数据行数最少，数据集 3 中所包含的数据行数最多。纵坐标代表运行时间。

由表 1 和图 7 可见，与传统贝叶斯检测僵尸网络相比，基于 MapReduce 的检测不仅降低了训练成本，也提高了系统执行效率。而且，随着数据量增大，基于 MapReduce 的贝叶斯检测僵尸网络的优势越发明显。

5.2 正确率随检测属性列个数变化

利用相同数据集检测 1－6 列属性个数时正确率的变化，分别以表 2 和图 8 折现图表示。

表 2 不同属性列个数的检测正确率比较

1	2	3	4	5	6
96.95%	97.27%	97.62%	98.41%	99.29%	99.95%

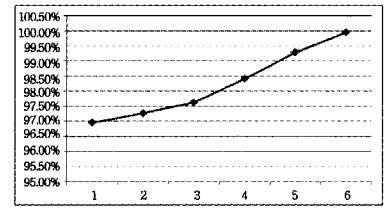

图 8 不同属性列个数的检测正确率比较

图 8 中，横坐标代表的是检测的属性列个数。纵坐标代表检测正确率。

由表 2 和图 8 所示的实验结果，得到如下结论：

1. 基于 MapReduce 的贝叶斯分类对于检测僵尸网络是有效的：对属性列个数选取从 1－6 的平均正确率均在 90% 以上。

2. 选取不同的属性个数直接影响基于 MapReduce 的贝叶斯分类器的准确率：从图 8 看出对属性个数选取从 1－6 的正确率逐渐上升。说明一定程度上，检测属性个数越多正确率越高。但是，由于硬件条件和实现难度的局限，借鉴国外重点实验室数据分析的结果和国内入侵检测研究的论文，初步确定训练数据选取 6—8 个属性值进行检测。

5.3 检测效率随 Hadoop 集群节点数变化

为测试基于 MapReduce 检测僵尸网络的贝叶斯算法随节点数变化的性能，本文将原数据集放大，实验 C 仍采用普通 PC 单机，实验 D 的 Hadoop 集群分别采用 1 台、2 台、3 台、4 台同配置的节点。比较同样数据量在不同节点数时检测所需的时间。该实验结果分别以表 3 和图 9 折现图表示。

表 3 不同节点数检测时间比较(单位：毫秒)

实验组别	机器数	检测时间
实验 C	1	99489.0
实验 D	1	80867.0
	2	80838.0
	3	76762.0
	4	74643.0
	5	71689.0
	6	71656.0

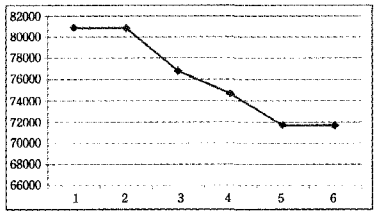

图 9 不同节点数检测时间比较

图 9 中横坐标代表的是 Hadoop 集群上不同的节点数，纵坐标代表运行时间。

由表 3 和图 9 可见，Hadoop 集群节点数越多，检测所需时间越少。由于网络传输延迟、任务分配量变大等原因，随着节点数增多，运行时间缩小程度不明显。

结束语 利用贝叶斯分类算法能有效地检测僵尸网络，但僵尸网络流量大及贝叶斯分类器前期需要对大量的网络数据集进行训练和学习，从而占用较多的系统资源和网络资源。为此，本文提出基于 MapReduce 僵尸网络的贝叶斯检测算法，把贝叶斯算法训练阶段求先验概率、条件概率和检测阶段后验概率的计算分别并行化处理，解决了大量数据在贝叶斯算法中耗用资源的问题。实验表明，该算法具有很好的执行效率。

参 考 文 献

[1] OikarinenJ, Reed D. Internet relay chat protocol[R]. Request for Comment s (RFC) 1459, IETF, May 1993

[2] Jiang H, Shao X. Detecting P2P botnets by discovering flow dependency in C&C traffic[J]. Peer-to-Peer Networking and Applications, 2012, 5: 1-12

[3] 李晓桢，程佳，胡军. 基于聚类分析的僵尸网络识别系统[J]. 计算机系统应用，2009，8：130-135

[4] 王威，方滨兴，崔翔. 基于终端行为特征的 IRC 僵尸网络检测[J]. 计算机学报，2009，32(10)：1980-1988

[5] 蒋鸿玲，邵秀丽. 基于神经网络的僵尸网络检测方法[J]. 智能系统学报，2013，8(2)：113-118

[6] Goebel J, Holz T. Rishi: identify bot contaminated hosts by irc nickname evaluation[C]// Proceedings of USENIX First Workshop on Hot Topics in Understanding Botnets. Cambridge, USA, 2007: 1—12

[7] 杜跃进，崔翔. 僵尸网络及其启发[J]. 中国数据通信，2005，7(5)：9-13

[8] Dean J, Ghemawat S. MapReduce: Simplified dtat process-ing on large cluster[J]. Communications of the ACM, 2005, 51(1): 107-113

[9] 陶永才，薛正元，石磊. 基于 MapReduce 的贝叶斯垃圾邮件过滤机制[J]. 计算机应用，2011，31(9)：2412-2416

[10] 张鹏，唐世渭. 朴素贝叶斯分类中的隐私保护方法研究[J]. 计算机学报，2007，30(8)：1267-1276

(上接第 131 页)

参 考 文 献

[1] Fikret S, Bülent Y. Time synchronization in sensor networks: a survey[J]. IEEE Network, 2004, 18(4): 45-50

[2] 肖竹，谭光华，李仁发，等. 无线传感器网络中基于超宽带的 TOA/AOA 联合定位研究[J]. 计算机研究与发展，2013，50(3)：453-460

[3] Saurabh G, Ram K, Mani S. Timing-sync protocol for sensor networks[C]// Proceedings of the 1st international conference on embedded networked sensor systems, 2003. Los Angeles, CA, USA: ACM Press, 2003: 138-149

[4] Sichitiu M L, Chanchai V. Simple, accurate time synchronization for wireless sensor networks[J]. Wireless Communications and Networking, 2009, 2: 1266-1273

[5] Greunen J V, Jan R. Lightweight time synchronization for sensor networks[C]// Proceedings of the 2nd ACM international conference on wireless sensor networks and applications, 2003. San Diego, CA, USA: ACM, 2003: 11-19

[6] Li Li, Liu Yong-pan, Yang Hua-zhong, et al. A Precision Adaptive Average Time Synchronization Protocol in Wireless Sensor Networks[C]// Proceedings of the 2008 IEEE International Conference on Information and Automation, 2008. Zhangjiajie, China, 2008: 20-23

[7] 王义君，钱志鸿，王桂琴，等. 无线传感器网络能量有效时间同步算法研究[J]. 电子与信息学报，2012，34(9)：2174-2179

[8] Miklós M, Branislav K, Gyula S, et al. The Flooding Time Synchronization Protocol[C]// Proceedings of the 2nd international conference on Embedded networked sensor systems, 2004. Los Angeles, CA, USA: ACM Press, 2004: 39-49

[9] 周鸣争，汪军，严楠，等. 无线传感器网络中一种基于行为可信的访问控制机制[J]. 计算机科学，2012，39(B06)：72-76

[10] 刘政，狄佳. 一种自适应 Huffman 算法在无线传感器网络数据压缩中的应用[J]. 重庆理工大学学报：自然科学版，2013，27(2)：84-88