# 基于伪梯度提升决策树的内网防御算法

厉柏伸 李领治 孙 涌 朱艳琴
(苏州大学计算机科学与技术学院 江苏 苏州 215006)

**摘 要** 结合TF-IDF算法思想,提出了特征频率、森林频率以及伪梯度提升决策树,解决了梯度提升决策树随着迭代次数的增加,错误数据被边缘化的问题。在伪梯度提升决策树中,所有决策树分别在原始数据集的Bootstrapping后的数据集上产生,无须针对每次迭代来对数据集采样。在分布式集群上进行内网防御的实验,结果表明在一定规模的训练集上,伪梯度提升决策树具有更好的预测准确度。

**关键词** 伪梯度提升决策树,分布式集群,内网防御
**中图法分类号** TP393 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.04.026

## Intranet Defense Algorithm Based on Pseudo Boosting Decision Tree

LI Bai-shen LI Ling-zhi SUN Yong ZHU Yan-qin
(School of Computer Science and Technology, Soochow University, Suzhou, Jiangsu 215006, China)

**Abstract** Combining with the idea of TF-IDF algorithm, the frequency of characteristics (Eigen Frequency), the frequency of forest(Forest Frequency) and the pseudo boosting decision tree(PBDT) were put forward, solving the marginalized problem of wrong data with the increasing number of iterations for gradient boosting decision tree(GBDT). In PBDT, all the decision trees produce respectively in data sets after the original data set of the Bootstrapping, without aiming at each iteration to sample data sets. Then intranet defense experiment was conducted on distributed cluster. The experimental results show that on the training set with a certain scale, PBDT has better prediction accuracy.

**Keywords** Pseudo boosting decision tree, Distributed cluster, Intranet defense

## 1 前言

入侵检测一直是网络安全中的热门话题。例如,在外网中,主要防御和检测的对象是数据流和信息流,以及是否有恶意的远程登录、是否有恶意的网络间密文监听等。除已有的访问控制、防火墙、通信加密技术等手段的应用外,每年还会产生新技术。然而,局域网内部的安全却经常被忽视,导致内网成为攻击事件的潜在爆发区域。因为内网中的数据流都是默认安全的,所以防御的中心侧重于用户的行为或习惯,使得内网防御相较于外网防御尚显薄弱。因此,进一步理解和研究内网防御变得愈发重要。

在对内网用户行为进行分析的研究中,各类方法层出不穷,以机器学习领域的方法最为公众所认同。决策树应用于该研究中时,分裂特征采用信息增益率替代信息增益,得到了更好的效果。决策树是一种弱分类器,在至少拥有 $k$ 种威胁的内网中,其分类正确率是否大于 $1/k$ 具有很大的随机性,分类效果的偶然性成分偏大。但通过集成学习模型对若干弱分类器进行集成,得到的最终分类器往往可以克服前者容易出现过拟合现象的不足。

随机森林RF是由美国科学家Leo Breiman将Ho于1998年提出的随机子空间方法与其在1996年提出的Bagging理论[7]相结合而衍生出的一种机器学习算法。随机森林是以决策树为基本分类器的一个集成学习模型,它包含多个由Bagging技术训练得到的决策树。当输入待分类的样本时,最终的分类结果由多个决策树的输出结果投票决定。随机森林克服了决策树容易过拟合的缺点,对噪声和异常值具有较好的容忍性,对高维数据分类问题具有良好的可扩展性和并行性。形式上对多个弱分类器进行随机糅合,并不能使其得到理论上的保证。SCHAPIRE等人提出的AdaBoosting思想[9]在RF的基础上更精细化,并且拥有了理论上的证明,但只适用于二元分类问题。AdaBoosting.M1算法[10]把AdaBoosting算法推广到多分类问题中,但要求单个分类器的准确度至少达到1/2。AdaBoosting.MH[10]把多元分类的样本转为 $k$ 个1和0的标签,间接地把 $k$ 类问题转化为对数量为 $k$ 的二元分类的算法进行求解。当 $k$ 值较大时,计算过程会变得冗余复杂。在内网防御这种需要实时调整预测模型的应用环境中,其反应灵敏度尚待改进。

本文提出了伪梯度提升决策树Pseudo Boosting Decision Tree(PBDT),并从理论上证明了其分类正确率要高于RF。通过对AdaBoost-DTree的研究,提出了具有类似思想的算法,旨在综合运用并考虑所有决策树的特长和优势,降低算法的最终错误率。

到稿日期:2017-01-09 返修日期:2017-03-11 本文受国家自然科学基金(61373164,11375221)资助。
**厉柏伸**(1991-),男,硕士生,主要研究方向为并行与分布式计算、数据挖掘,E-mail:zgzjlbs@163.com;**李领治**(1977-),男,博士,副教授,主要研究方向为网络安全、无线网络,E-mail:sdlilingzhi@163.com(通信作者)。

## 2 AdaBoost-DTree 和梯度提升

### 2.1 AdaBoost-DTree 算法

机器学习中的 RF 算法虽然使用了集成方法，但其实质是属于 Uniform Aggragation，即一种通过平均若干决策树的结果的平均集成，只是从概率意义上降低了误差。AdaBoost 方法在集成学习中应用广泛，属于 Non-Uniform Aggragation，该算法在使用过程中会生成对应各个决策树的权重 $\alpha$，最终得到子树权重集成的结果。

以下是 AdaBoost-DTree 算法的一般流程。

**算法 1** AdaBoost-DTree(D,T)

输入：训练数据集 D，其大小为 n

输出：G

1. t 的初始值为 1，$u^1$ 的初始值是 n 维向量(1/n,…,1/n)，$D_t^1$ 每个子树对应的训练数据。

2. WHILE：t <= T

2.1 THEN：

通过 DTree($D_t'$,$u^t$)得到使得训练误差 $E_{in}^{u^t}(h)=\frac{1}{n}\sum_{i=1}^{n}u_i^t(-y_i h(x_i))$ 达到最小值的 h，此时令 $g_t=h$。

2.2 计算 $\varepsilon_t=\frac{\sum_{i}^{n}u_i^t[y_i\neq g_t(x_i)]}{\sum_{i=1}^{n}u_i^t}$，$\varphi_t=\sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}}$，$\alpha_t=\ln\varphi_t$。

2.3 更新 $u^t$ 为 $u^{t+1}$：

$[y_i\neq g_t(x_i)]$：$u_i^{t+1}=u_i^t*\varphi_t$

$[y_i=g_t(x_i)]$：$u_i^{t+1}=u_i^t/\varphi_t$

3. 返回 $G(x)=\mathrm{sign}(\sum_{t=1}^{T}\alpha_t g_t(x))$。

其中，$u^t$ 表示 $D_t'$ 中 $n$ 个数据的权重。每一轮迭代计算新决策树时，为了方便，令 $E_m^{u^t}(h)=\varepsilon_t$，表示当前迭代计算的决策树的误差值。$\varphi_t$ 表示当前迭代计算对错误数据的偏向值，使得当前决策树产生错误数据的权重增大，正确数据的权重减小。$\alpha_t$ 表示当前迭代的决策树的分类正确度量，其值越大，对应的子树在最终分类器中占得的权重也就越大。

AdaBoost 关注错误数据的修正，初次训练时，没有任何可修正偏差，因此抽样时对所有的数据点分配同样的选中概率 $1/n$。

AdaBoost 的本质是梯度提升在二元分类问题上的一个特例。梯度提升(Gradient Boosing)实质上是一个框架，可以在其中套入很多不同的算法，此处正是将决策树算法嵌入该框架中，从而使得算法整体的误差率降低。

### 2.2 AdaBoost-DTree 分析

对 Boosting 方法进行研究后发现，其关键步骤的迭代具有数据依赖性。为了最后的集成，AdaBoost-DTree 算法需要不断根据上次的迭代错误来修正此次的数据采样比例。例如，进行 $t+1$ 次迭代时，需要根据 $u^{t+1}=u^t*\varphi_t^{y*g_t(x)}$[9] 对原数据集进行抽样。在总共 $T$ 轮的迭代计算中，每一轮都依赖于上一轮的关键信息，在训练问题单一的情况下，会导致大量的计算资源被闲置，分布式计算的优势也得不到应有的发挥。

数据抽样大小是需要考虑的另外一个因素。在 AdaBoost-DTree 方法中，随着迭代的进行，越大的抽样数据集可以使得最终分类器的效果越好。

每一轮迭代中，抽样的训练数据 $D^t$ 都是在 $D$ 中按 $u^t$ 的值来进行抽取的。例如对于 $u^1$，若其对于所有数据实例的权重都相同，则所有实例被选中的概率相同，因为初始时并不知道哪些实例是 AdaBoost-DTree 会判断出错的。后面的迭代训练会着重增大前次迭代出错的实例的比重，从而使得 AdaBoost-DTree 整体的误差越来越小。

第 1 次迭代时，在大小为 $n$ 的原数据集 $D$ 上进行抽样，得到大小为 $m$ 的训练数据集 $D^1$。假设 $D$ 中有数量为 $a$ 的正确数据和数量为 $b$ 的错误数据，则 $n=a+b$。$u^1$ 对于每个数据的权重都相同，即为$[\frac{1}{n},\frac{1}{n},\cdots,\frac{1}{n}]$，则在 $D$ 中，错误的训练数据有$(1-\frac{mb}{n^2})^m$ 的概率未被纳入训练范围，令$\frac{b}{n}=\beta(1)$。

在进行可放回数据抽样时，先从大小为 $n$ 的数据集中选出 $m$ 个数据的概率是 $m/n$，之后因为是第一次抽样，所有的数据具有相同的权重，所以在这 $m$ 个数据中，一个数据在之后决策树成为错误数据的概率是 $b/n$，从而第一次抽样的错误数据不被纳入训练的概率是$(1-\frac{m}{n}*\frac{b}{n})^m$。

第 2 次迭代时，在大小为 $n$ 的原数据集 $D$ 上进行抽样得到大小为 $m$ 的训练数据集 $D^2$。同样地，$D$ 中有数量为 $a$ 的正确数据和数量为 $b$ 的错误数据。对于 $D^2$ 的错误数据，$u^2$ 中增大了相应维度的值。根据 $u$ 的更新公式可知，在此次训练时，$D$ 中错误数据依旧不被抽中的概率为：

$$(1-\frac{m}{n}*\frac{nb+nm-2bm}{n^2+nm-2bm})^m \tag{1}$$

在原数据集 $D$ 上进行第二次抽样时，其中数据量大小为 $a$ 的数据可由未被选入 $D^2$(数量为 $m$)的数据 $d_1$(数量为$(n-m)a/n$)和选入 $D^2$ 的数据 $d_2$(数量为 $ma/n$)构成。同样地，数据量大小为 $b$ 的数据可由未被选入 $D^2$ 的数据 $d_4$(数量为$(n-m)b/n$)和选入 $D^2$ 中的数据 $d_3$(数量为 $mb/n$)构成。根据 Adaboost 算法，减小分类的正确数据的比例，将未参与训练的 $d_1$ 和 $d_4$ 也视为正确数据。因此在这 $m$ 个数据中，一个数据在之后决策树成为错误数据的概率是 $\frac{\frac{(n-m)b}{n}*b+\frac{mb}{n}*a}{\frac{(n-m)a}{n}*b+\frac{ma}{n}*b+\frac{(n-m)b}{n}*b+\frac{mb}{n}*a}=\frac{nb+nm-2bm}{n^2+nm-2bm}=\beta(2)$。

令$\frac{nb+nm-2bm}{n^2+nm-2bm}=\beta(2)$，可以得到 $\beta(2)>\beta(1)$。继续迭代，并由数学归纳法可得：

$$1>\beta(t+1)>\beta(t)>0 \tag{2}$$

同时，可以列出在第 $t$ 轮迭代时错误数据不被抽中的概率的递推关系式：

$$z=(1-\frac{m}{n}\beta)^{\frac{m}{n}*n} \tag{3}$$

令 $z=y^n$，此时 $n$ 为常量。对于两种不同 $m$ 取值下的训练，将 $\beta$ 当作常量，其中令$\frac{m}{n}=x$，$x\in(0,1)$。求 $y$ 对于 $x$ 的一阶导函数，可得：

$$y'=e^{\beta x\ln(1-\beta x)}[\beta\ln(1-\beta x)-\frac{\beta^2}{1-\beta x}] \tag{4}$$

可以发现，式(4)的值小于 0。这意味着，在某一次迭代

中，随着 $m$ 的增加，错误数据不被抽中的概率下降，最终得到高效的分类器的可能性增大。

当需要训练的决策树数量 $T$ 和 $m$ 均取较大值时，最终分类器的误差率才可能达到期望要求。同时，$T$ 和 $m$ 又是分布式环境中计算速度的主要因素，但是在实时性要求较高的内网防御中，迭代次数和抽样数据量对时间成本都会有直接作用，这个矛盾在分布式环境下增加了 AdaBoost-DTree 结果的不稳定性。

## 3　Pseudo Boosting Decision Tree(PBDT)

### 3.1　伪梯度提升决策树算法

与 RF 算法一样，PBDT 算法要求弱分类器 $g_t(x)$ 的性能比随机猜测略优，其中信息熵阈值 $\delta$ 可以按照不同的数据规模和类型来设置，也可以先找到一个可以接受的训练时间和误差率的折中值。

本算法中的迭代次数 $T$ 是一个需要手动设置的参数。理论上，在一个特定的数据集上产生 PBDT 算法结果的最终误差率首先会随着 $T$ 值的增加而增加。而此时迭代次数的增加，将使得最终分类器的规模更接近目标问题的真实规模。而当 $T$ 值超过一个最优值后便会产生过拟合现象，即过分注重少数几个特别事务的对错，使得分离器在新的数据集上的分类效率不佳。

在 PBDT 中，所有决策树分别在原始数据集的 Bootstrapping 后的数据集上产生，无须针对每次迭代对数据集进行采样。这样做有两点好处：1)无须串行迭代，在数据预处理时，可以为不同 DT 提前分配好相应的训练数据，加快总体执行速度，在内网防御中，可更快地形成防御模型，从而可以更快地甄别威胁操作；2)当有新类型的威胁指令进入时，可以立即重新训练，所花费的代价也要比串行迭代小得多，这使得内网防御中的抗变异性得到强化。

在 AdaBoost-DTree 算法中，最终的 $G(x)$ 是对所有迭代产生的子分类器按照权重 $\alpha_t=\ln\sqrt{\frac{1-\varepsilon_t}{\varepsilon_t}}$ 进行求和（$\varepsilon_t=\frac{\sum_{i=1}^{n}u_i^t[y_i\neq g_t(x_i)]}{\sum_{i=1}^{n}u_i^t}$），这些权重的计算和有针对性地训练数据抽样制约了分布式计算的效率。本文提出了类似于 TF-IDF 中词频的特征频率 EF(Eigen Frequency)以及森林频率 FF(Forest Frequency)，采用每棵树自身产生的最优分裂特征来计算该树的 $EF$ 和 $FF$ 权值，并将这些权值当作该棵树的权重 $\alpha$。在对所有 DT 完成基于 Bootstrapping 抽样的数据集的最优分裂特征时，位于不同深度的分裂特征所提供的信息熵增益随着分裂特征所处层数的不同而不同(位于上层的特征划分的数据集较大)。可以给出如下定义：

$$EF=\frac{1}{level} \tag{5}$$

其中，$level$ 值表示该分裂特征在决策树中的深度。

确定完并行计算中的每个 $DT$ 的最优特征之后，所有具有分裂价值的特征集 $S_A$ 已经生成。为了得到某个具体特征 $a$ 在 S 中的重要性，给出如下 $FF$ 的表达式：

$$FF=\frac{|DT_a|}{T} \tag{6}$$

其中，$DT_a$ 表示包含特征 $a$ 的 $DT$ 的集合。对于 $DT$ 的分裂特征，$|DT_a|$ 越大，表明这个特征的划分能力越强。综上，可以给出 $DT$ 的权重：

$$\alpha_t=\sum_{j=1}^{p}EF_j*FF_j \tag{7}$$

其中，$p$ 表示最优分裂特征集 $S_A$ 的大小。式(7)可表示为第 $t$ 棵决策树所的分裂特征在整个森林中的贡献。这些权重的计算是全局独立的，有利于实现分布式环境中的效率提升。

PBDT 具有的优化效果，基于原来的随机森林，综合考虑了多个 DT 的多次采样，而对于 $\alpha$ 的选择，又综合考虑了 DT 中的特征的分类能力。不同于 AdaBoost-DTree 算法，PBDT 计算权重时是以当前已知错误数据为基础来进行训练的。在 PBDT 中，$DT_a$ 和 $S$ 不具有冲突性，给分布式计算的实现带来了便利，更加有利于快速形成对威胁指令的甄别模型。

设训练数据集为 $D^t$，$|D^t|$ 为样本容量，共有 $K$ 个类别，$|C_k|$ 是属于类 $C_k$ 的样本数，则 $\sum_{k=1}^{K}|C_k|=|D^t|$。设特征 $A$ 有 $n$ 个不同的取值，根据 $A$ 的取值将 $D^t$ 划分为 $n$ 个子集 $D_1^t$，$D_2^t$，…，$D_n^t$，$|D_i^t|$ 为 $D_i^t$ 的样本个数，$\sum_{i=1}^{n}|D_i^t|=|D^t|$。记子集 $D_i^t$ 中属于类 $C_k$ 的样本集合为 $D_{ik}^t$。

本算法的步骤如算法 2 所示。

**算法 2**　伪梯度提升决策树算法(PBDT)

输入：迭代次数 T，原始训练数据集内网用户的行为日志 D（1≤t≤T），信息熵阈值 δ，特征集 S

输出：伪梯度提升决策树 G

1. WHILE：t<＝T，对 D 进行 Bootstrapping 得到 $D^t$
2. THEN：
   - 2.1 若 $D^t$ 中的实例属于同一类 $C_k$，则 $g_t$ 为单节点树，并将类 $C_k$ 作为该节点的类标记，返回 $g_t$。
   - 2.2 若 A＝∅，则 $g_t$ 为单节点树，将 $D^t$ 中实例数最大的类作为该节点的类标记，返回 $g_t$。
   - 2.3 否则，根据 $\Delta=H(D^t|A)-H(D^t)$，计算 S 中各个特征对 $D^t$ 的信息增益 Δ，得到具有最大信息增益的特征 $A_g$。
   - 2.4 如果 $A_g$ 的信息增益 Δ 小于阈值 δ，则置 $g_t$ 为单节点树，并将 $D^t$ 中实例数最大的类 $C_k$ 作为该节点的类标记，返回 $g_t$。
   - 2.5 否则，针对 $A_g$ 的每一个可能值，将 $D^t$ 分割为若干非空子集 $D_i^t$，对于第 i(1≤i≤n)个子节点，以 $D_i^t$ 为训练集 $D^t$，以{S－$A_g$}为特征集 S，递归执行步骤 2。
3. 对以上数量为 T 的决策树的计算在逻辑上是并行执行的，因此在得到 T 个 $g_t$ 之后，可以计算得到 $g_t$ 所具有的 $S_t$。对数量为 T 的 $g_t$，分别计算：

$$\alpha_t=\sum_{j=1}^{p}EF_j*FF_j$$

4. 返回 $G(x)=\mathrm{sign}(\sum_{t=1}^{T}\alpha_t g_t(x))$。

### 3.2　相关证明

下面给出 PBDT 的分类能力强于 RF 的证明。

**推论 1**　PBDT 的分类能力要强于 RF。

证明：由排序不等式可知，设有两组数 $a_1$，$a_2$，…，$a_n$，$b_1$，$b_2$，…，$b_n$，满足 $a_1\leqslant a_2\leqslant\cdots\leqslant a_n$，$b_1\leqslant b_2\leqslant\cdots\leqslant b_n$，则有 $a_1\ b_n+a_2b_{n-1}+\cdots+a_n\ b_1$ ①$\leqslant a_1\ b_{t1}+a_2b_{t2}+\cdots+a_nb_{tn}$ ②$\leqslant a_1b_1+a_2\ b_2+\cdots+a_n\ b_n$ ③。其中，$t_1$，$t_2$，…，$t_n$ 是 1，2，…，$n$ 的任意一个排列，当且仅当 $a_1=a_2=\cdots=a_n$ 或 $b_1=b_2=\cdots=b_n$ 时等号成立。

随机森林的生成和决策的本质是找到向量空间中各个维度上区分程度的最大化界限，再在其上判断新的输入向量的类别。因此，在区间的具体维度上拥有较大权重的决策树，就对该维度具有较大的决策权，即可以将随机森林看成是一棵抽象的决策树 FR，它的每一个分裂特征都是由一系列决策树决定的。例如，对于分裂特征 $c_i$，可能有 $k$ 棵子决策树同时将其作为分裂特征。

在 PBDT 中，对 DT 赋予权重，是对拥有的分裂特征的决策重要程度的强化，加大这些 DT 所拥有的各个分裂特征的值，从而提高这个分裂特征的决策权，使得 FR 在这个分裂维度上能最大限度地按这些 DT 所划分的范围决策。在同一份训练数据中，越上层的分裂特征所划分的数据量越大，从而使得结果更加趋向于最终分类器。因此，考查一个具体的 DT 时，可以分析其中的分裂特征之间的关系。

1）考虑一个深度值较小的分裂特征 $s$，其在 FR 中出现的概率较大[13]，即 $EF$ 的取值与整个森林的最终分类能力在概率模型下具有正相关性。

2）对于 $DT_j$ 的第 $i$ 个分裂特征 $s_{ij}$：

①若假设所有的 DT 拥有这个 $s_{ij}$（即 $FF$ 相等），并且所有层级都拥有相同的 $EF$，这种特殊情况就是随机森林采取的策略，未经排序的 $FF$ 和 $EF$ 是理论上的乱序值，而最终的 $\alpha$ 的贡献本质上是排序不等式的乱序和[2]。

②若考虑 $EF$ 的大小，即深度值较小的 $s$ 具有较大的 $EF$，同时考虑 $FF$ 的情况，即 $FF$ 随着 $s$ 在 $FR$ 中的重要性增加而增加，这样 $FF$ 和 $EF$ 都是理论上的顺序值，对最终 $\alpha$ 的贡献可当作排序不等式的顺序和[3]。

3）同理，考虑一个具有较强分类能力的 DT，其所产生的权重 $\alpha$ 也与其分类能力具有概率模型下的正相关性。对于 PBDT 和 RF 而言，$G_{\mathrm{RF}}$ 算法没有计算 DT 之间的差异，即所有 $\alpha$ 的取值相等，其分类能力属于乱序和：

$$G_{\mathrm{RF}}(x)=\mathrm{sign}\left(\sum_{t=1}^{T}g_t(x)\right)$$

而 $G_{\mathrm{PBDT}}$ 的分类能力是顺序和：

$$G_{\mathrm{PBDT}}(x)=\mathrm{sign}\left(\sum_{t=1}^{T}\alpha_t g_t(x)\right)$$

4）因此，$E_{RF}\geqslant E_{PBDT}$。

**推论 2** PBDT 的执行速度要快于 GBDT。

证明：由文献[13]可知，对于拥有 $n$ 个数据项和 $m$ 个属性的数据集，不考虑剪枝消耗的时间，结合分布式环境的并行处理，在有 $L$ 个执行单位的情况下，具有 $T$ 个 DT 的 RF 的时间复杂度为 $O(Tmn\log n/L)$。相较于 RF，PBDT 需要额外包含 $EF$ 和 $FF$ 的计算，$EF$ 需要 $O(1)$ 的复杂度，构造 DT 时利用一个数组存储各个层级的属性；而 $FF$ 需要 $O(Tm)$ 的复杂度。此外，$EF$ 和 $FF$ 在权重分配上的优化可以带来 $O(W)$ 的增速效果。综上，PBDT 的时间复杂度为 $O(Tm(n\log n+1)/LW)$。而在 GBDT 中，由于本次迭代中的 $u$ 依赖于上一轮，假设不考虑 $u$ 的更新消耗和每次 DT 训练的数据集的生成耗时，理论上 $T$ 次训练是串行执行的，则需要的时间复杂度为 $O(Tmn\log n)$。证毕。

## 4 实验结果及分析

### 4.1 实验环境

在实际环境的安装和配置中，设置流程较为繁琐，此处以图表的形式给出。表 1 列出了具体的相关参数。

表 1 实验环境的参数设置

Table 1 Parameter setting of experiment environment

| 参数 | 数值 |
|---|---|
| 操作系统版本 | Ubuntu 16.04 |
| 计算机节点数 | 1 * Master+3 * Worker |
| Hadoop 版本 | 2.6.0 |
| Spark 版本 | 1.6.0 |
| 可用最大内存 | 3GB |
| Executor 可用核数 | 2 |

### 4.2 实验设计

为了防止出现歧义，先对内网的一些固有设置进行说明。由于内网集群上运行着许多样式的服务，为了保证整个系统的安全性，在机器上不提供一些特殊命令，这就大大减少了一些比较生僻的且具有高威胁性动作的产生。在 Linux 这种开放型的系统上，多种命令的重复或交错能产生数量庞大的操作组合，这对内网安全是十分不利的。

对于所有的用户操作指令数据，可以找到一个适合且囊括以上所有用户指令的核心向量表示。最终，包含本文归纳的多种内网威胁指令以及一些可用于附加判断的信息在内的核心向量的格式如下：

[Label，hostIP，doubleIP，cmd1 _ Vec，cmd2 _ Vec，…，cmd15_Vec]

其中，各元素分别代表了类别标签、用户登录 IP、是否通过一个 IP 登录了 2 个及以上用户，以及 15 种对应的命令向量（标示该内网用户执行过的系统命令）。其中，Label 代表单位时间内某 IP 用户所产生的核心向量的威胁标识。doubleIP 的作用是记录是否有用户在同一设备上登录了 2 个不同的账号，在内网中，理论上所有的用户都只能拥有一个账户，即该帐户是初次联网创建的；而且，在一般性人员离开之后，所有账号都会被删除。因此，如果此时同个 IP 短时间内登录了两个账号，则需要对其进行甄别。

对于 cmdx_Vec，举例分析 cmd10_Vec。其指令为：

rm [选项] [文件]

则用户操作转换为 cmd10_Vec 的格式如下：

uID，pID，comdDir，callDir，cmd10 _ name，cmd10 _ arg1，cmd10_name1，cmd10_multiple

其中，uID 和 pID 分别表示用户 ID 和进程 ID。而 comdDir，callDir 这 2 个信息同样也很重要，前者表示用户所执行指令的文件夹路径，后者表示用户当前所在文件夹路径。以“cmd10”开头的分别对应指令的每一项，而最后一项的作用是标识该系统命令是否出现多次。

### 4.3 实验分析

本文在进行性能测试时构建的输入向量中目前仅考虑了用户操作；涉及到的相同的文件，以期降低预测模型的空间复杂度；今后可以考虑加入所涉及文件之间的关系，以增强对威胁的预测能力。而在构建预测模型的过程中，考虑到 Linux 命令的用法多变，不同的参数、不同版本的指令不尽相同，为了消除不必要的误差，利用数据清洗过程过滤掉无关的重复指令和失败指令，侧重于找到威胁指令序列的相关特征。

RF 和 PBDT 都可以采用 Bootstrapping 进行抽样，可以形成用于训练的数据集和用于验证的袋外数据集，并分别在不同数据规模上进行多次测试。实验比较了 RF 和 PBDT 在

不同迭代次数下 10 次测试的平均值，并将其作为评价指标。图 1—图 4 给出了迭代次数不同的情况下二者误差率的比较：在机器学习领域，除算法训练所消耗的时间外，算法之间的时间复杂度往往可以用相同误差率条件下所消耗的训练数据量来替换和比较。

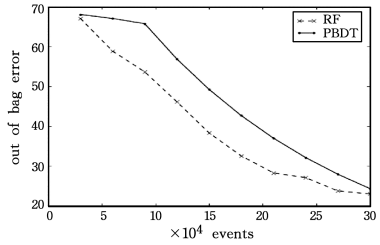


图 1　迭代数为 $T_1$ 时的误差率走势

Fig. 1　Error rate trend when iteration is $T_1$

图 1 中，两种算法设置的迭代次数 $T_1=50$。初始训练数据不足时，误差率都接近 70%；随着每次增加 3 万条训练数据，RF 和 PBDT 的误差率都呈现下降的趋势，为凸显效果，选择的迭代数目使得二者具有趋于一致的下降速度。因此，在迭代次数为 $T_1$ 的情况下，PBDT 与 RF 的时间复杂度比值约为 30∶30。

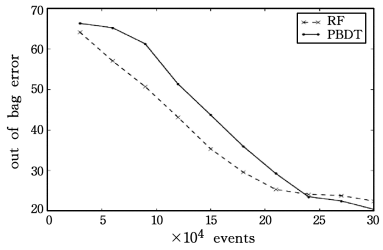


图 2　迭代数为 $T_2$ 时的误差率走势

Fig. 2　Error rate trend when iteration is $T_2$

图 2 中，两种算法设置的迭代次数 $T_2=100$。RF 算法在初期事务量较小时，理论上没有形成弱分类器之间的干扰，较 PBDT 有一定优势，误差率快速减小；但随着数据量的增多，O($W$)的值也逐渐增大。由图 2 可知，PBDT 在 24 个数据单位附近趋于平稳，RF 却在 25 个数据单位左右趋于平稳。因此 PDBT 与 RF 的时间复杂度比值约为 24∶26。

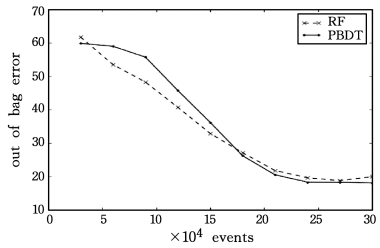


图 3　迭代数为 $T_3$ 时的误差率走势

Fig. 3　Error rate trend when iteration is $T_3$

图 3 中，在迭代次数达到一定值，即 $T_3=150$ 后，PBDT 算法的误差下降的速度更快，此时 O($W$)可以更快达到最大值，所有决策树之间的信息共享也更加迅速，且从 20 个数据单位附近开始，其始终拥有较低的误差率。而 RF 直到 23 个数据单位才达到与 PBDT 一样的效果。因此 PBDT 与 RF 的时间复杂度比值约为 20∶23。

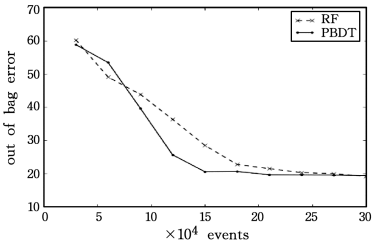


图 4　迭代数为 $T_4$ 时的误差率走势

Fig. 4　Error rate trend when iteration is $T_4$

图 4 是拥有充足的迭代次数即 $T_4=200$ 后的情形，当二者的误差率趋于稳定时，RF 需要的训练数据为 25 个单位，而 PBDT 则需要 15 个单位左右。因此，PBDT 与 RF 的时间复杂度比值约为 15∶25。

由此可以看出，在理想状况下，PBDT 算法有着比 RF 更低的时间复杂度。此外，当迭代次数充足时，PBDT 算法充分分享了不同弱决策树之间的经验($EF$ 和 $FF$)，从而达到加速学习的效果，误差下降的速度明显快于 RF。而 RF 算法达到同等效果需要的数据量却不能明显地呈现减少的趋势，即迭代次数的增加没有为算法得到最终分类器带来额外的加速。

表 2 列出了二者在 10 次测试过程中计算的误差率的方差。

表 2　RF 和 PBDT 误差率的方差

Table 2　Variance of error rate in RF and PBDT

| 事务量/万个 | RF | PBDT |
|---|---|---|
| 3 | 0.291 | 0.352 |
| 6 | 0.314 | 0.343 |
| 9 | 0.256 | 0.312 |
| 12 | 0.239 | 0.272 |
| 15 | 0.303 | 0.253 |
| 18 | 0.288 | 0.220 |
| 21 | 0.297 | 0.181 |
| 24 | 0.273 | 0.157 |
| 27 | 0.254 | 0.133 |
| 30 | 0.262 | 0.129 |

可以看到，对于 PBDT，事务量较小，袋外误差的波动范围也较大，随着数据量规模的增大，误差率均值不断减小，方差也趋于减小。由于没有决策树之间的信息共享，训练数据量的增大对 RF 误差率的稳定性没有影响；而对于 PBDT，随着事务量的增加，在保证拥有足够迭代次数的情况下，其间接地减少了目标函数的个数，从而使得误差率更加集中。PBDT 的误差均值的下降速率比 RF 快，且在相同规模数据集上的误差方差值也更小。

综上，实验初步验证了推论 1，即 PBDT 在相同规模的数据集上具有比 RF 更低的误差率，在数据规模较小的情况下要略差于 RF，这很可能是 PBDT 算法在该情况下对决策树信息误共享产生过拟合导致的结果。

**结束语**　本文针对 RF 在模型收敛速度上的不足和 GBDT 中迭代次数的依赖性问题，结合了 TF-IDF 的思想和当前大数据下分布式计算的背景，提出了伪梯度提升决策树算法。所提算法减少了训练数据集在 Spark 平台上的计算时间，有效地避免了迭代计算的依赖性，且在内网防御的用户行为威胁程度的判断上取得了不错的效果。

本实验虽然进行了实际测试，但是算法中核心的 $EF$ 和 $FF$ 的取值本质上还具有一定的偶然性，这种形式上的定义

有时不能很准确地描述分类器的目标函数。例如,在新类型的威胁指令刚出现时,其对应的分裂特征不大可能出现在决策树的上层,因为具有这种分裂特征的实例数量还比较少,所以其 $EF$ 值也会较小。同理,对于新出现的分裂特征,其在其他决策树中出现的概率会偏小,导致 $FF$ 值也较小。造成这种现象的原因是,在训练初期,权值容易受到分类器之间的相互干扰。因此,本文提出的算法在应对新攻击时还有问题需要解决,后续工作可以在这方面进行尝试和努力。

## 参 考 文 献


[1] PRADHAN B. A comparative study on the predictive ability of the decision tree, support vector machine and neuro-fuzzy models in landslide susceptibility mapping using GIS[J]. Computers & Geosciences, 2013, 51(2): 350-365.

[2] SHOTTON J. Real-time human pose recognition in parts from single depth images[J]. Communications of the ACM, 2013, 56(1): 116-124.

[3] FREUND Y, SCHAPIRE R E. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting[J]. Journal of Computer and System Sciences, 1997, 55(1): 119-139.

[4] GLIGOROV V V, WILLIAMS M. Efficient, reliable and fast high-level triggering using a bonsai boosted decision tree[J]. Journal of Instrumentation, 2012, 8(2): 6.

[5] RUTKOWSKI L. Trees for Mining Data Streams Based on the McDiarmid's Bound[J]. IEEE Transactions on Knowledge and Data Engineering, 2013, 25(6): 1272-1279.

[6] SCHAPIRE R E. The strength of weak learn ability[J]. Machine Learning, 1990, 5(2): 197-227.

[7] XIE J B. Research and Implementation of Intranet Security Situation Awareness Technology[D]. Guangzhou: Guangdong University of Technology, 2015. (in Chinese)
谢锦彪. 内网安全态势感知技术的研究与实现[D]. 广州: 广东工业大学, 2015.

[8] BREIMAN L. Bagging predictors [J]. Machine Learning, 1996, 24(2): 123-140.

[9] SCHAPIRE R E. A brief introduction to boosting[C]// International Joint Conference on Artificial Intelligence. Sweden, 1999: 1401-1406.

[10] SCHAPIRE R E, SINGER Y. Improved boosting algorithms using confidence-rated predictions[J]. Machine Learning, 1999, 37(3): 297-336.

[11] WITTEN I H, FRANK E, HALL M A. Data Mining: Practical Machine Learning Tools and Techniques(Second Edition)[M]. San Francisco: Morgan Kaufmann publications, 2005.

[12] PAIK J H. A novel TF-IDF weighting scheme for effective ranking[C]// 36th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2013: 343-352.

[13] WU H C, LUK R W P, WONG K F, et al. Interpreting TF-IDF term weights as making relevance decisions[J]. ACM Transactions on Information Systems, 2008, 26(3): 55-59.

[14] ESCALANTE H J. Term-weighting learning via genetic Programming for text classification[J]. Knowledge-Based Systems, 2014, 83(1): 176-189.

[15] KUNCORO B A, ISWANTO B H. TF-IDF method in ranking keywords of Instagram users' image captions[C]// International Conference on Information Technology Systems & Innovation. 2015: 1-5.


(上接第 151 页)


[2] ASOKAN N, EKBERG J E, KOSTIAINEN K, et al. Mobile Trusted Computing[J]. Proceedings of the IEEE, 2014, 102(8): 1189-1206.

[3] FUGINI M G, BREVEGLIERI L, PELOSI G, et al. Trusted Computing for Embedded Systems[OL]. http://rd.spring.com/content/pdf/bfm%3A978-3-319-09420-5%2F1.pdf.

[4] MU Y. Zhong Guan Cun Trusted Computing Industry Alliance was Established[J]. Information Security and Communications Privacy, 2014(5): 16. (in Chinese)
木易. 中关村可信计算产业联盟成立[J]. 信息安全与通信保密, 2014(5): 16.

[5] SONG X L, ZHANG L H, CHEN D Y. Preventing Hypervisor-based Rootkit with Trusted Execution Technology[J]. Information Security & Communications Privacy, 2009, 7: 76-81.

[6] YU A, ZHAO S. Enhancing Flexibility of TCG's TNC through Layered Property Attestation[C]// IEEE International Conference on Trust, Security and Privacy in Computing and Communications. IEEE Computer Society, 2011: 751-756.

[7] ARTHUR W, CHALLENER D, GOLDMAN K. Platform Configuration Registers[M]// A Practical Guide to TPM 2.0. Apress, , Berkeley, CA, 2015.

[8] SAILER R, ZHANG X, JAEGER T, et al. Design and implementation of a TCG-based integrity measurement architecture [C]// Usenix Security Symposium. San Diego, CA, USA, 2004: 16-16.

[9] JAEGER T, SAILER R, SHANKAR U. PRIMA: policy-reduced integrity measurement architecture[C]// SACMAT 2006, ACM Symposium on Access Control MODELS and Technologies. Lake Tahoe, California, USA, 2006: 19-28.

[10] CAMENISCH J, CHEN L, DrRJVERS M, et al. One TPM to Bind Them All: Fixing TPM 2.0 for Provably Secure Anonymous Attestation[C]// Security and Privacy. IEEE, 2017: 901-920.

[11] XU Z Y, HE Y P, DENG L L. Efficient Remote Attestation Mechanism with Privacy Protection[J]. Journal of Software, 2011, 22(2): 339-352. (in Chinese)
徐梓耀, 贺也平, 邓灵莉. 一种保护隐私的高效远程验证机制[J]. 软件学报, 2011, 22(2): 339-352.

[12] ZHU Y, LI Q B, ZHONG C L, et al. Non-balanced Binary Hash-tree Model for Fine-grained Integrity Measurement[J]. Journal of Chinese Computer Systems, 2014, 35(7): 1604-1609. (in Chinese)
朱毅, 李清宝, 钟春丽, 等. 用于细粒度完整性度量的非平衡二叉哈希树模型[J]. 小型微型计算机系统, 2014, 35(7): 1604-1609.

[13] FU D, PENG X, YANG Y. Unbalanced tree-formed verification data for trusted platforms[J]. Security & Communication Networks, 2016, 9(7): 622-633.

[14] DENNING P J. The Locality Principle[J]. Communications of the Acm, 2005, 48(7): 19-24.