DOI:10.11992/tis.201706027
网络出版地址:http://kns.cnki.net/kcms/detail/23.1538.TP.20170831.1058.016.html

移动社交网络异常签到在线检测算法

赵冠哲,齐建鹏,于彦伟,刘兆伟,宋鹏
(烟台大学 计算机与控制工程学院,山东 烟台 264005)

摘 要:随着智能手机、Pad 等智能移动设备的广泛普及,移动社交网络的应用得到了快速发展。本文针对移动社交网络中用户异常签到位置检测问题,提出了一类基于用户移动行为特征的异常签到在线检测方法。首先,在基于距离的异常模型基础上,提出了基于历史位置 (H-Outlier)和基于好友圈 (F-Outlier)两种异常签到模型;然后,针对 H-Outlier 提出了一种优化的检测算法 H-Opt,利用所提的签到状态模型与优化的邻居搜索机制降低检测时间;针对 F-Outlier 提出了一种基于触发的优化检测算法 F-Opt,将连续的在线异常检测转化成了基于触发的异常检测方式;最后,在真实的移动社交网络用户签到数据集上,验证了所提算法的有效性。实验结果显示,F-Opt 显著降低了 H-Opt 的异常检测错误率;同时,相比于 LUE 算法,F-Opt 和 H-Opt 的效率分别平均提升了 2.34 倍和 2.45 倍。

关键词:移动社交网络;异常检测;签到位置;基于距离的异常;好友圈;签到状态;邻居搜索;时间触发检测
中图分类号:TP391 **文献标志码**:A **文章编号**:1673-4785(2017)05-0752-08

中文引用格式:赵冠哲,齐建鹏,于彦伟,等.移动社交网络异常签到在线检测算法[J]. 智能系统学报, 2017, 12(5): 752-759.
英文引用格式: ZHAO Guanzhe, QI Jianpeng, YU Yanwei, et al. Online check-in outlier detection method in mobile social networks[J]. CAAI transactions on intelligent systems, 2017, 12(5): 752-759.

Online check-in outlier detection method in mobile social networks

ZHAO Guanzhe, QI Jianpeng, YU Yanwei, LIU Zhaowei, SONG Peng
(School of Computer and Control Engineering, Yantai University, Yantai 264005, China)

Abstract: With the increasing popularization of smartphone, Pads and other smart mobile devices, the use of mobile social networks has also developed rapidly. In this paper, we propose an online method for detecting check-in outliers based on user mobility behavior in mobile social networks. First, based on a distance-based outlier model, we propose two check-in outlier models with respect to historical location (H-Outlier) and friend circle (F-Outlier), respectively. Second, for the H-Outlier, we propose an optimized detection algorithm called H-Opt, which utilizes the proposed check-in status model and an optimized neighbor searching mechanism to reduce computation time. For the F-Outlier, we propose a trigger-based optimized detection algorithm called F-Opt, which transforms continuous online outlier detection into trigger-based outlier detection. Lastly, we present our experimental results, based on a real-world check-in dataset, which demonstrate the effectiveness of the proposed algorithm. Our experimental results show that F-Opt significantly reduces the error rate of H-Opt outlier detection. In addition, compared with the LUE algorithm, the F-Opt and H-Opt algorithms improved efficiency by 2.34 and 2.45 times, respectively.

Keywords: location-based social networks; outlier detection; check-in location; distance-based outlier; friend circle; status of check-in; neighbor searching; time-triggered detection

随着 GPS 终端、Pad、智能手机等位置感知设备的广泛普及,各类新型社交网络手机应用不断涌现,促使移动社交网络(mobile social networks)得到了快速发展。移动社交网络也称基于位置的社交网络(location-based social networks, LBSN),本质是提供一个在人群中分享兴趣、爱好、状态和活动等信息的平台[1]。如国内的腾讯 QQ、微信、新浪微博、人人网,国外的 Twitter、Facebook、Gowalla、Foursquare 等,这些 LBSN 应用聚集了大量移动用户。据 We Are Social 公司在 2016 年数字报告[2]中

收稿日期:2017-06-08. **网络出版日期**:2017-08-31.
基金项目:国家自然科学基金项目(61403328, 61572419);山东省重点研发计划项目(2015GSF115009);山东省自然科学基金项目(ZR2014FQ016);烟台大学研究生科技创新基金项目(YDZD1712).
通信作者:于彦伟. E-mail:yuyanwei@ytu.edu.cn.

指出,2016 年全球社交网络用户约 19.7 亿,占全球总人口的 27%。LBSN 中海量带有位置信息的社会媒体数据可被用于各类挖掘应用研究,如挖掘用户兴趣偏好、好友推荐、兴趣点推荐、热点路径推荐等[3]。

移动社交网络正逐渐改变着人们的生活方式和生活习惯,给人类社会带来前所未有的变革以及巨大的经济收益[4]。同时它也吸引了一些不法者盗取用户账号及信息进行各种恶意行为,影响了用户的正常使用,损害了用户利益,甚至给用户带来了巨大的经济损失。因此,面向移动社交网络,追踪用户的历史签到位置数据,从用户移动行为特征视角,对用户状态进行在线异常检测,这对于移动社交网络的安全、用户的隐私保护等具有重要意义。

针对移动社交网络异常账号的检测问题,学术界和工业界都提出了大量检测方案[5],主要包括基于行为特征的检测、基于内容的检测、基于图的检测和无监督学习的异常检测方法等。而针对移动社交网络中签到位置数据异常检测研究还比较少。本文从用户的移动位置特征视角对异常签到位置进行检测,针对移动社交网络用户异常签到检测问题,提出了一类基于签到位置的在线异常检测方法。首先,在基于距离的异常检测基础上,提出了两种异常签到模型,即基于历史位置的异常签到(history location based outlier,H-Outlier)和基于好友圈的异常签到(friendship based outlier,F-Outlier);其次,针对 H-Outlier,提出了一种优化的检测算法 H-Opt,利用优化的检测状态与邻居搜索机制降低检测时间;然后,针对 F-Outlier,基于提出的 3 个优化策略,提出了一种基于触发的优化检测算法 F-Opt,将连续在线异常检测转化成了基于触发的异常检测方式,本文采用滑动窗口技术实现 H-Opt 和 F-Opt;最后,在真实的移动社交网络用户签到数据集上,验证了所提算法的有效性和效率。

在移动社交网络中用户往往会在新地点进行签到或登录,所以 H-Outlier 检测到的异常签到地点很有可能并非真正的异常。为排除这些伪异常状况,本文提出了好友圈的概念和基于好友圈的异常签到模型 F-Outlier,这是因为在移动社交网络中 50%~70%的行为可由周期行为解释,还有 10%~30%行为可根据好友关系行为解释[6]。也就是说,在移动社交网络中用户往往与认识的好友共同出现在不经常签到的地点,如朋友聚餐、商务活动或者公务出差等。针对这些伪异常,我们通过检测好友圈中好友签到位置来判断用户的签到是否为真正的异常签到。

1　相关工作

Knorr 和 Ng[7]最早提出了基于距离的空间异常(DB-Outlier)检测问题。给定一个数据集合,若某数据点的邻域内的数据个数小于给定阈值,则该数据点为基于距离的异常点。之后,Knorr 等[8]将基于距离的异常模型应用到时空轨迹数据异常检测中。Ramaswamy 等[9]提出了一种基于 k 近邻的异常定义,根据数据点到第 k 近邻的距离检测出 n 个 k 近邻距离最大的数据点作为异常点。Breunig 等[10]提出另外一类基于密度的数据异常类型,通过计算本地异常因子 LOF 来检测局部异常点数据,但该算法存在较高计算复杂度问题。

在数据流异常检测方面,Yang 等[11]采用滑动窗口方式在数据流上挖掘基于邻居的模式,主要包括基于密度的聚类和基于距离的异常检测。Angiulli 等[12]通过在数据流上分析数据点是否为“safe inlier”来优化检测方法。Kontaki 等[13]利用“safe inlier”设计了一种事件触发的数据流异常检测优化算法。Cao 等[14]提出了一种数据流异常检测框架,该框架可用于处理基于距离的和基于 k 近邻的两类异常检测模型。

针对移动对象的轨迹数据,Lee 等[15]提出了一种划分-检测的轨迹异常检测框架,将轨迹划分成 t-partition 序列,通过计算 t-partition 间的距离和密度,以发现异常的子 t-partition。Bu 等[16]提出了一种连续监控移动对象实时轨迹数据流的异常检测算法,该方法关注在检测单个移动对象实时轨迹数据中的异常子轨迹段。文献[17]针对海量移动对象系统中异常对象,提出了一种基于邻居的异常移动对象检测方法,可发现不同于其他邻居对象运动轨迹的异常对象。

2　问题定义

本节首先给出一些重要的定义和表示,然后对基于距离的移动社交网络异常签到模型进行描述。

定义 1　基于距离的位置异常。给定位置数据集 D、距离阈值 d、邻居点数量阈值 k,对于位置数据点 $o \in D$,如果 $|\{p \mid \mathrm{dist}(p,o) \leqslant d, p \in D\}| \leqslant k$,则称 o 是一个基于距离的位置异常点。

$U = \{u_1, u_2, \ldots, u_m\}$ 表示移动社交网络的所有用户集,用户 u_i 在 t_j 时间点的签到位置表示为 p_j^i。

本文采用基于签到位置数量的滑动窗口 W 来处理移动社交网络中实时的签到数据,窗口长度标记为 $|W| = w$,也就是说,W 中包含 w 个签到位置。

定义 2　基于历史位置的异常签到。给定距离阈值 d、邻居点数量阈值 k,滑动窗口 W,对于用户 u_i 的签到位置 $p_b^i \in w$,如果 $|\{p_j^i \mid \mathrm{dist}(p_j^i, p_b^i) \leqslant d,$

$p_j^i \in W\}| \leqslant k$，则称签到位置 p_b^i 是一个基于历史位置的异常签到点，记为H-Outlier。

从定义 2 可以看出，H-Outlier 是基于自身历史签到位置的，这也是因为用户日常活动轨迹记录往往具有周期性[18]。

设 $\mathrm{Fr}(u_i)$ 表示用户 u_i 的所有直接好友集合。与 QQ、微信等社交网络的好友圈的定义不同，这里给出的好友圈定义既包括用户 u_i 直接好友又包括与 u_i 有一定数量共同好友的间接好友。

定义 3　好友圈。给定支持阈值 m，用户 u_i 的好友圈包括 u_i 的所有直接好友和与 u_i 存在至少 m 个共同直接好友的间接好友，即 $\mathrm{Fr}(u_i) \cup \{u_j \| \mathrm{Fr}(u_i) \cap \mathrm{Fr}(u_j)| \geqslant m\}$，简记为 $\mathrm{Net}(u_i)$。

定义 4　基于好友圈的异常签到。给定距离阈值 d、好友圈包含的好友数量阈值 k_f、时间 $\triangle t$，对于基于历史位置的异常签到点 p_a^i，如果 $|\{u_s | \exists p_j^s, \mathrm{dist}(p_j^s, p_a^i) \leqslant d, |t_a - t_j| \leqslant \triangle t, u_s \in \mathrm{Net}(u_i)\}| \leqslant k_f$，$p_a^i$ 是一个基于好友圈的异常签到点，记为 F-Outlier。

根据定义 4 可知，F-Outlier 是在 H-Outlier 定义的基础上定义的，因此，F-Outlier 集合是 H-Outlier 集合的子集。

文中涉及的符号名称及描述由表 1 给出。

表 1　符号名称及描述

Table 1　Symbols and its descriptions

名称	解释
p_j^i	用户 u_i 在 t_j 时间的签到位置
W	基于签到位置数量的滑动窗口
d	距离阈值
k	邻居点数量阈值
$\mathrm{Fr}(u_i)$	用户 u_i 的直接好友集合
m	共同直接好友数量阈值
$\mathrm{Net}(u_i)$	用户 u_i 的好友圈
$p.\mathrm{Nei}_{\mathrm{before}}$	签到位置 p 之前的邻居集合
$p.\mathrm{Nei}_{\mathrm{after}}$	签到位置 p 之后的邻居集合
k_f	好友圈中邻居点数量阈值
D	签到位置数据流
$\triangle t$	触发项的有效期

3　基于历史位置的异常签到检测算法

本节将详细介绍 H-Outlier 的在线检测算法。为了提升检测效率，首先介绍检测算法采用的几种优化策略，然后给出详细的检测算法。

3.1　优化策略

若两个签到位置之间距离小于给定距离阈值 d，称它们互为邻居点。

1）签到位置的状态

传统检测方法仅将用户的签到位置分为正常和异常两种状态，而通过在滑动窗口下用户签到位置的检测发现，签到位置的状态可进一步细分，以便减少距离计算。在滑动窗口 W 中签到位置 p 的状态可分为：①确定的正常点。如果 p 在 W 中邻居数量大于等于 k，并且在 p 点之后邻居数量也大于等于 k，这时不管 W 如何滑动，位置 p 在滑出窗口之前，在窗口内邻居数量一定大于等于 k 个，因此，此时 p 的状态可以认定为确定的正常签到点，记为 safe-inlier。②不确定的正常点。如果 p 在 W 中邻居数量大于等于 k，但是在 p 点之后邻居数量小于 k，这时 p 虽然是一个正常状态的签到位置，但是当 W 滑动时，在位置 p 之前的邻居可能会滑出窗口，p 的状态可能会变成异常状态，此时的 p 可认为是一个不确定的正常点，记为 unsafe-inlier。③确定的异常点。设签到位置 p 之前的邻居集合为 $p.\mathrm{Nei}_{\mathrm{before}}$，之后的邻居集合为 $p.\mathrm{Nei}_{\mathrm{after}}$，如果 p 在 W 中邻居数量小于 k，在 p 之前有 $\mathrm{m}_{\mathrm{before}}$ 个位置，如果 $|p.\mathrm{Nei}_{\mathrm{after}}|$ 小于 k-$\mathrm{m}_{\mathrm{before}}$，这时不管 W 如何滑动，位置 p 在滑出窗口之前，都不可能包括 k 个邻居，因此，此时 p 的状态可以认定为确定的异常签到点，记为 safe-outlier。④异常点。剩下的异常签到点都划归为这一类，记为 outlier。

很明显，一旦确定 safe-inlier 和 safe-outlier 状态之后不必再对这些签到位置进行重新检测。而随着窗口滑动，outlier 和 unsafe-inlier 仍需重新检测状态。

2）优化的邻居点搜索机制

根据上述签到位置的状态划分方法，我们将签到位置的邻居点分为在 p 签到之前和在 p 签到之后两个部分，以便快速确定签到位置的状态。若 $|p.\mathrm{Nei}_{\mathrm{after}}| \geqslant k$，$p$ 为 safe-inlier；若 $|p.\mathrm{Nei}_{\mathrm{after}}| < k$，但 $|p.\mathrm{Nei}_{\mathrm{before}}| + |p.\mathrm{Nei}_{\mathrm{after}}| \geqslant k$，$p$ 为 unsafe-inlier；若 $|p.\mathrm{Nei}_{\mathrm{before}}| + |p.\mathrm{Nei}_{\mathrm{after}}| < k$，并且 $|p.\mathrm{Nei}_{\mathrm{after}}| < k$-$\mathrm{m}_{\mathrm{before}}$，$p$ 为 safe-outlier；其他情况下，p 为 outlier。

通过观察发现，随着滑动窗口的滑动，对于签到位置 p 的所有邻居点 $p.\mathrm{Nei}_{\mathrm{before}}$ 和 $p.\mathrm{Nei}_{\mathrm{after}}$，$p.\mathrm{Nei}_{\mathrm{before}}$ 中的位置点不断滑出窗口，而 $p.\mathrm{Nei}_{\mathrm{after}}$ 则一直处在窗口内，直到 p 也滑出窗口才会失效。因此，在搜索 p 的邻居时可优先搜索在其之后的邻居，越靠后的邻居有效期越长；而在其之前的邻居，越靠近 p 存活期越长。基于该观察，在滑动窗口内从最后一个签到位置向前依次搜索邻居，满足优先搜索 $\mathrm{Nei}_{\mathrm{after}}$，又实现了从最近到最远搜索 $\mathrm{Nei}_{\mathrm{before}}$。

3）最少邻居点搜索机制

若查找出签到位置 p 的所有邻居，需扫描一遍窗口。受文献[14]启发，在检测点 p 时，并不需要

搜索出所有邻居,当满足 k 个时,即可判定位置 p 的状态,但是同时为了结合上述的优化策略 1)和 2),我们给出最少邻居点搜索机制。

给定签到位置 p 和新签到位置 p_{new},若 $|p.\text{Nei}_{\text{after}}| \geq k$ 同时 $|p_{\text{new}}.\text{Nei}_{\text{before}}| \geq k$,则无需计算 p 与 p_{new}的距离。此时位置 p 已经确定为 safe-inlier,而 p_{new}也已确定为 unsafe-inlier。否则,将继续计算 p_{new}到所有之前签到位置的距离,以确定 p_{new} 为 outlier。采用该机制,既满足了所有签到位置状态的检测又实现了最少的距离计算。

3.2　H-Outlier 在线检测算法

算法 1 给出了 H-Outlier 的优化检测算法,对于新签到位置 p_{new},对窗口 W 中所有签到位置进行状态重新检测。首先,按照优化的邻居搜索机制,从窗口内最后一个位置,依次向前搜索邻居,如第 1 行所示。然后,采用最少邻居点搜索机制,判定位置 p_{i} 和 p_{new}是否都已确定状态,若已确定,则不用再继续搜索邻居 2)~4)行,否则,继续计算距离,更新邻居集合(见 5)~7))。如果 p_{i}的状态不是 safe-inlier 或 safe-outlier,则根据搜索邻居情况更新其状态,如 8)~17)所示。如果 p_{new}搜索到 k 个在其之前的邻居,状态更新为 unsafe-inlier(见 18)~19)),检测完窗口 W 后,若邻居数量依然少于 k 个,状态标记为 outlier。

算法 1　H-Outlier 优化检测算法

输入　当前窗口 W, k, d, p_{new};

输出　所有 $p \in W$ 的状态。

1)for i 从 W_{end}到 W_{start} do
2)if p_i.sta 是 safe-inlier 或者 p_i.sta 是 safe-outlier
3)　if p_{new}.sta 是 unsafe-inlier then
4)继续
5)　else if p_i与 p_{new}的距离$\leqslant d$ then
6)将 p_i添加到 $p_{\text{new}}.\text{Nei}_{\text{before}}$
7)将 p_{new}添加到 $p_i.\text{Nei}_{\text{after}}$
8)　else
9)if p_{i}与 p_{new}的距离$\leqslant d$ then
10)将 p_i添加到 $p_{\text{new}}.\text{Nei}_{\text{before}}$
11)将 p_{new}添加到 $p_i.\text{Nei}_{\text{after}}$
12)　if $|p_i.\text{Nei}_{\text{after}}| \geqslant k$ then
13)p_i.status 更新为 safe-inlier
14)else if $|p_i.\text{Nei}_{\text{before}}| + |p_i.\text{Nei}_{\text{after}}| \geqslant k$ then
15)p_i.status 更新为 unsafe-inlier
16)else if $|p_i.\text{Nei}_{\text{before}}| < k\text{-m}_{\text{before}}$ then
17)p_i.status 更新为 safe-inlier
18)if $|p_{\text{new}}.\text{Nei}_{\text{before}}| \geqslant k$ then
19)p_{new}.status 更新为 unsafe-inlier
20)　if $|p_{\text{new}}.\text{Nei}_{\text{before}}| < k$ then
21)p_{new}.status 更新为 outlier

4　基于好友圈的异常检测算法

本节将详细介绍基于好友圈的异常签到 F-Outlier的检测算法。F-Outlier 定义在 H-Outlier 之上,对于当前窗口 W 上 H-Outlier 需进一步验证是否为 F-Outlier。H-Outlier 仅在用户自身近期的历史签到位置上查找邻居点(滑动窗口 W 内),而 F-Outlier 则在其好友圈用户的最近签到位置中搜索邻居点($\triangle t$ 时间差内)。

为了进一步降低 F-Outlier 检测的时间消耗,本节同样给出了 3 个优化策略。

1)最少好友搜索机制

与 H-Outlier 检测的最少邻居点搜索机制相似,检测用户 u_i签到位置 p_a^i 的状态时,并非需要完整搜索一遍 $\text{Net}(u_i)$,若已存在 k_f个好友在 t 内在同一地点(小于给定距离 d)签到过,则可以停止搜索,此时可判定 p_a^i 并非 F-Outlier。

2)历史邻近好友优先原则

在社会学领域研究中发现,人们在某一段时间内往往与相同一群人存在较密集的交互[4]。根据上述社会学的发现,我们对每个用户维护一个邻近好友的排序列表 L_f,用于记录历史邻近签到状况,最近一次的邻近签到好友排在首位,每次搜索邻近好友时同时更新列表次序可以较大概率快速搜索邻近的签到好友,以排除 H-Outlier,而不用随机遍历 $\text{Net}(u_i)$。

3)基于时间触发的检测机制

当签到位置 p_a^i 在当前时刻检测为 F-Outlier 后,在后续的 Δt 内需要不断重新检测它的状态是否有变化,但是当好友圈用户没有签到数据更新时,重新检测会导致冗余计算。因此,提出了一种基于时间触发的检测机制。

每个用户 u 维护一个触发列表 trigger,触发列表中每项表示为$\langle p_a^i \rangle$, 当用户 u 更新签到数据时,触发对 p_a^i 的 F-Outlier 状态重检测。可以看出,每个触发项都存在一个有效期,即$|t_c - t_a| \leqslant \Delta t$,当前时间 t_c 与 Δt_a 的时间差不大于 Δt,超过该时间差,触发项自动失效。

基于时间触发的检测机制将连续异常检测转化成了基于触发的检测,当有新签到数据时,才对未过期的签到位置进行重新检测,大大减少了额外的周期性重新检测成本。

算法 2 描述了采用优化策略的 F-Outlier 的检测过程,针对用户 u_i的新签到位置 p_c^i,首先执行 H-

Outlier 检测，如 1）所示，如果 p_c^i 是一个 H-Outlier，则继续检测 p_c^i 是否为 F-Outlier，如 2）~12）所示。根据历史邻近好友优先原则，从 L_f 中由前向后搜索好友，若存在邻近签到的好友，则插入好友邻居集合 Nei_{fri}，并将该好友放置于 L_f 首位（见 4）~7）），然后根据最少好友搜索机制，满足 k_f 个好友，停止搜索（见 8）~9））。搜索完一遍 u_i 好友圈之后，认为满足 k_f 条件，则认定为 F-Outlier，并将放入好友圈 Net（u_i）中每个用户的 trigger 列表中，如 10）~12）所示。

算法 2 F-Outlier 检测算法

输入 新的签到 p_c^i、k_f、d；

输出 所有的 F-Outlier。

1）对于状态为 H-Outlier 的 p_c^i

2）if p_c^i.sta 为 outlier then

3）for j 取值 1，2，…，|Net(u_i)| do

4） L_f 获取 j 后给 u_j

5）if p_c^i 与 $p_{c\pm\Delta t}^i$ 的距离≤d then

6） 将 $p_{c\pm\Delta t}^i$ 添加到 p_c^i.Nei_{fri}

7）将 u_j 插入到 L_f 首位

8）if $|p_c^i.\text{Nei}_{\text{fri}}| \geqslant k_f$ then

9） p_c^i.sta 更新为 F-Inlier、break

10）if $|p_c^i.\text{Nei}_{\text{fri}}| < k_f$ then

11） p_c^i.sta 更新为 F-Outlier

12）将 p_c^i 添加到 Nei(u_i).trigger

13） u_i.trigger 更新到 Trigger

14）for 每个 $p_a^i \in$ Trigger do

15）if $|t_a - t_c| \leqslant \Delta t$ 并且 p_c^i 与 p_a^j 的距离≤d then

16）将 p_c^i 添加到 p_a^i.Nei_{fri}

17）if| p_a^i.Neifri| ≥k_f then

18） p_a^i.sta 更新为 F-Inlier

19）将 p_a^i 从 u_j.trigger 中删除

接下来，就要触发 u_i 的 trigger 列表，如 13）~19）所示。每个触发项 p_a^j，若仍在有效期内，且与 p_c^i 为邻居，则判断它邻居点的数量是否满足 k_f，若满足，则更新 p_a^j 的状态，并将其移出 u_i 的 trigger 列表（见 17）~19））。

5 实验评估

实验平台配置为 4.0 GHz i7-6700k 处理器，8 GB内存，Windows7 操作系统，所有算法由 Java 实现。

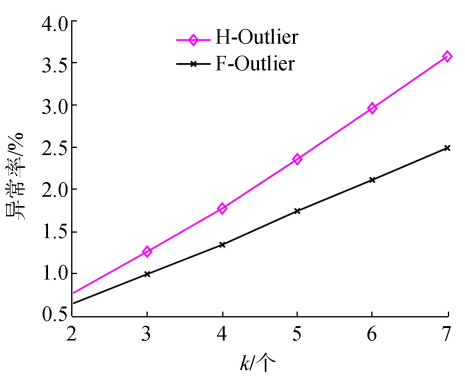
数据集。实验数据集采用的是基于位置的移动社交网络 Gowalla 真实数据集[18]。数据集中包括了 195 591个用户，95 万条好友关系，收集了 2009 年 2 月~2010 年 10 月之间的 644 万个签到位置数据。

对比方法。算法 1 描述的基于历史位置的异常检测算法记为 H-Opt 算法，算法 2 描述的基于好友圈的异常检测算法记为 F-Opt 算法，对比方法记为 LUE（lazy with update events）算法[13]。

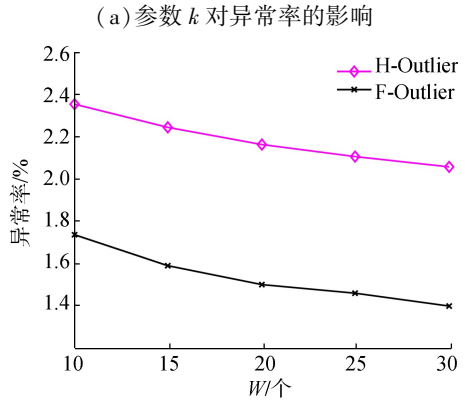
评估方法。采用所有用户在单个窗口内异常率来评估所提方法的有效性。实验结果取所有滑动窗口下的平均值。对于效率评估，通过变化各重要参数，采用单个窗口平均消耗的 CPU 时间和内存占用来评估算法的性能。每次窗口滑动一个新签到位置。

5.1 有效性评估

首先，对 H-Opt 和 F-Opt 算法的有效性进行了评估与分析。默认参数设置：d = 300 m，w = 20，k = 4，k_f = 3，Δt = 3 h，m = 4。H-Opt 与 F-Opt 的异常检测结果如图 1 所示。图 1（a）描述的是变化参数 k 对不同算法的有效性影响。可以发现，随着 k 的增加，H-Opt 和 F-Opt 检测出的异常率都呈线性增加，这是因为增加邻居点数量阈值 k 会使较多的签到点被认定为 H-Outlier。由于 F-Outlier 基于 H-Outlier，随着 H-Outlier 数量的增加，F-Outlier 也会有所上升，这与它们定义相符。同时还可以发现，随着 k 的增加，F-Opt 算法与 H-Opt 算法在异常率检测上的差异不断增大，即 F-Opt 的异常误判率不断降低。当 k = 7时，降低的异常率已达到 1.09%，也就是说，F-Outlier 有效排除了 30.27%的伪异常。



（a）参数 k 对异常率的影响



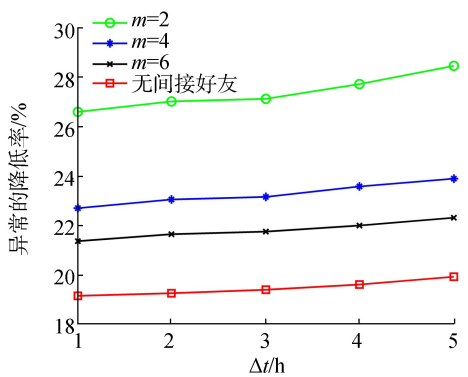
（b）参数 W 对异常率的影响

图 1 不同参数下的算法异常率

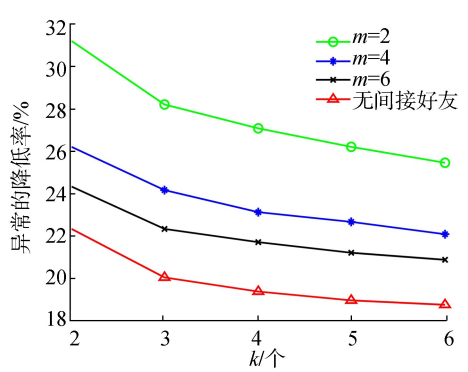
Fig.1 Outlier rate under different parameters

图 1(b)显示的是变化参数 W 对不同算法的有效性影响。由图 1 可以看出，随着 W 的增大，异常率不断下降。这是因为当增大 W 时，邻居点的历史签到时间范围也随之增加，从而签到的邻居点数量也会相应增加，使 H-Outlier 和 F-Outlier 的异常率随之减少，这也与它们定义相符。同时可以看出，F-Opt 的异常率明显低于 H-Opt 算法，这也证明了 F-Outlier 检测的有效性。

为了更好地考察 F-Opt 算法的有效性，进一步对 F-Opt 算法进行了评估。图 2 是 k_f、Δt 和 m 在不同组合时，F-Outlier 相比 H-Outlier 降低的异常率情况。如图 2(a)所示，当 k_f 值从 2 变化到 6 时，考虑间接好友的 F-Outlier 进一步降低了 H-Outlier 的异常率。共同好友的要求越低(m 值越小)，异常的降低率越高，即伪异常越少，当 $m=2$ 时，相对仅考虑直接好友，F-Outlier 异常的降低率平均提高了 8%。图 2(b)展示的是当 $k_f=4$ 时，Δt 从 1～5 变化，对算法异常降低率的影响。可以发现，随着 Δt 的增加，异常的降低率在不断增大，这是因为搜索的时间区间增加，在时间区间内邻近签到好友的数量也在增加。同样地，考虑好友圈的 F-Outlier 减少了更多伪异常。在 $\Delta t \leqslant 4$ 时，F-Outlier 减少的异常率变化不大，这也说明大多邻近好友的签到是在较短时间差内完成。



(a)参数 k_f、m 对异常降低率的影响



(b)参数 Δt、m 对异常降低率的影响

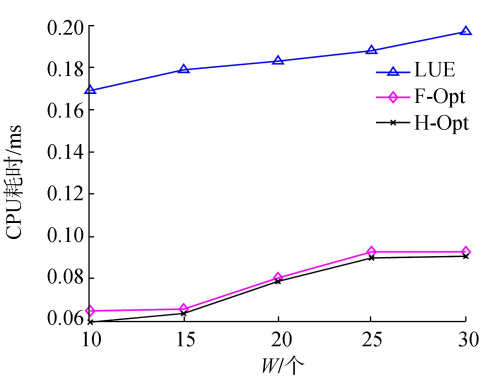
图 2　不同参数对 F-Opt 异常降低率的影响

Fig. 2　F-Opt outlier decreasing rate w.r.t. different parameters

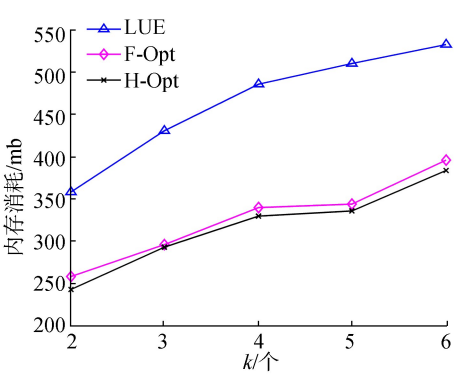
5.2　效率评估

本节评估了滑动窗口 W 和邻居点数量 k 对 H-Opt 和 F-Opt 算法的消耗时间和内存的影响，并与 LUE 算法进行了对比分析。默认参数：$d=300$，$k_f=3$，$\Delta t=3$ h，$m=4$。

在图 3(a)中，固定 $k=4$，W 从 10 变化到 30，随着 W 的增加，3 个算法消耗的时间都有所增加，这是因为窗口增加，都需要增加对异常点邻居搜索范围，而 LUE 需要计算当前签到点与窗口内所有签到点的距离，所以导致了耗时较长。同时，随着窗口增长，H-Outlier 和 F-Outlier 数量反而越少，虽然搜索邻居的范围增加导致了总消耗时间增加，但是由于采用了优化的邻居搜索机制和最少邻居搜索机制，仅有异常点增加了邻居搜索，因此，消耗的 CPU 时间增长越来越缓慢，而 F-Opt 算法需要再次检测的异常点变少也使得消耗的时间越来越少。相比于 LUE，F-Opt 和 H-Opt 分别平均提升了 2.34、2.45 倍效率。在图 3(b)中固定 $W=20$，k 从 2 变化到 6。随着 k 的增加，3 个算法消耗的 CPU 时间也逐渐增加。虽然 F-Opt 算法需要重新检测更多 H-Outlier 异常，但 F-Opt 算法并没有快速增加时间消耗，仅平均增加了 0.002 ms。这也体现了我们提出的基于触发的优化检测策略的作用。相比于 LUE，F-Opt 和 H-Opt 分别平均提升了 2.31、2.36 倍效率。



(a)参数 W 对 CPU 消耗时间的影响



(b)参数 k 对 CPU 消耗时间的影响

图 3　不同参数下的算法消耗的时间

Fig.3　CPU time w.r.t. different parameters

在内存方面,如图 4 所示,随着 W 的增加,3 个算法消耗的内存也逐渐增加。这是因为随着滑动窗口的增长,窗口内存储的签到点也随之增多。LUE 算法需要储存窗口存储所有签到点的邻居点,消耗的内存较多。H-Opt 和 F-Opt 消耗内存较少,这是因为采用了优化的邻居搜索机制和最少邻居搜索机制。由于增加了 F-Outlier 好友圈的邻近签到存储,F-Opt 略高于 H-Opt。

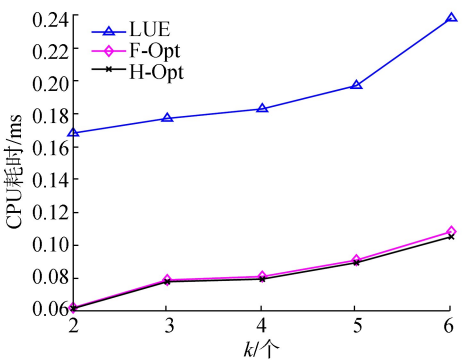
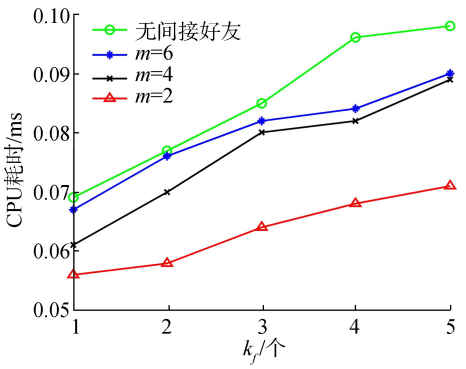


图 4 参数 W 和 k 变化下内存的消耗情况

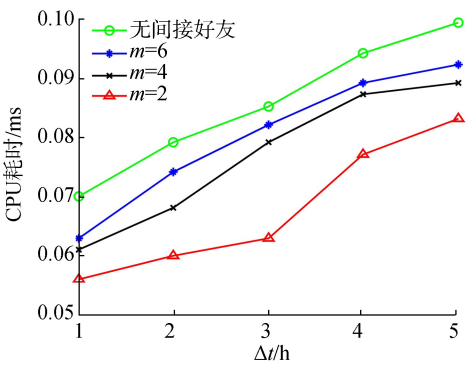
Fig.4 Memory w.r.t. parameter W and k

随着 k 的增加,3 个算法消耗的内存也在增加。这是因为所有算法都需要寻找更多的邻居。H-Opt 和 F-Opt 消耗内存增长缓慢,这是因为 H-Opt 算法采用最少邻居点搜索机制,随着 k 值的增加,在 H-Opt 算法中需要存储的自身签到点也随之增加。相比于 LUE,F-Opt 和 H-Opt 分别平均减少了 30% 和 32% 的内存消耗。

接下来,进一步评估了参数 m、k_f 和 Δt 对 F-Opt 算法效率的影响。我们测试了变化 k_f 和 Δt 对多个 m 值下 F-Opt 算法效率的影响,如图 5 所示。固定 $W=20$,$k=4$,$d=300$,在图 5(a)中固定 Δt = 3 h,在图 5(b)中固定 $k_f=3$,可以看出,随着 k_f 和 Δt 增加,F-Opt 消耗的 CPU 时间都逐渐增加。这是因为 k_f 增加,使得邻近签到好友搜索的个数增加;Δt 增加使得时间区间增加。因此消耗的 CPU 时间也要相应增加。同时也发现,不考虑间接好友时的 F-Opt 算法消耗最多时间,考虑间接好友时共同好友数量 m 值越少,使用的检测时间越少。这是因为 m 值越小,Net(u)集合越大,虽然搜索范围增加了,伪异常也增多了,由于我们采用了历史邻近好友优先原则和基于触发的检测机制,可快速发现伪异常,有效提升了算法检测效率。结合图 2 的实验结果可以说明,在移动社交网络异常签到检测中考虑用户间接好友的必要性和优势。



(a)参数 k_f 和 m



(b)参数 Δt 和 m

图 5 不同参数变化的算法 CPU 时间

Fig.5 CPU time w.r.t. different parameters

6 结束语

本文提出了一种针对移动社交网络异常签到位置的在线检测方法。基于距离的异常检测,定义了基于历史位置和基于好友圈的两种异常签到模型。然后,针对两种异常签到模型,分别提出了优化的检测算法,从签到位置的状态模型、优化的邻居搜索机制和基于时间触发的检测机制方面有效降低了检测时间。最后,在真实的移动社交网络用户签到数据集上,验证了所提模型与算法的有效性和效率。下一步将结合用户的操作行为进一步研究有效的移动社交网络异常检测方法。

参考文献:

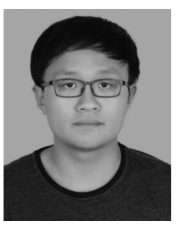
[1]於志文, 周兴社, 郭斌. 移动社交网络中的感知计算模型、平台与实践[J]. 中国计算机学会通讯,2012,8(5):15-21.

[2] WE ARE SOCIAL LTD. DIGITAL IN 2016[EB/OL]. [2017-03-10]. http://wearesocial.com/uk/special-reports/digital-in-2016.

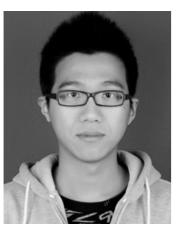
[3] ZHENG Yu, XIE X. Location-based social networks: locations[J]. Computing with spatial trajectories, 2011: 277-308.

[4]萧世埝. 基于位置服务与人类活动的关系和影响[J]. 中国计算机学会通讯, 2010,6(6): 30-35.
[5]张玉清, 吕少卿, 范丹. 在线社交网络中异常帐号检测方法研究[J]. 计算机学报, 2015, 38(10): 2011-2027. ZHANG Yuqing,LV Shaoqing,FAN Dan. Anomaly detection in online social networks[J]. Chinese journal of computers, 2015, 38(10): 2011-2027.
[6] CHO E, MYERS S A, LESKOVEC J. Friendship and mobility:user movement in location-based social networks [C]//ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Diego, USA, 2011: 1082-1090.
[7] KNORR E M, NG R T. Algorithms for mining distance-based outliers in large datasets [C]//International Conference on Very Large Data Bases. New York, USA, 1998: 392-403.
[8] KNORR E M, NG R T, TUCAKOV V. Distance-based outliers: algorithms and applications [J]. Vldb journal, 2000, 8(3/4): 237-253.
[9] RAMASWAMY S, RASTOGI R, SHIM K. Efficient algorithms for mining outliers from large data sets [C]// ACM SIGMOD International Conference on Management of Data. Dallas, USA, 2000: 427-438.
[10] BREUNIG MARKUS M. LOF: identifying density-based local outliers [J]. ACM sigmod record, 2000, 29(2): 93-104.
[11] YANG D, RUNDENSTEINER E A, Ward M O. Neighbor-based pattern detection for windows over streaming data [C]//International Conference on Extending Database Technology: Advances in Database Technology. Saint Petersburg, Russia, 2009: 529-540.
[12] ANGIULLI F, FASSETTI F. Detecting distance-based outliers in streams of data [C]//Sixteenth ACM Conference on Conference on Information and Knowledge Management. Lisboa, Portugal, 2007:811-820.
[13] KONTAKI M, GOUNARIS A, PAPADOPOULOS A N, et al. Continuous monitoring of distance-based outliers over data streams [C]// IEEE, International Conference on Data Engineering. Hannover, Germany, 2011:135-146.
[14] CAO L, YANG D, WANG Q, et al. Scalable distance-based outlier detection over high-volume data streams [C]//International Conference on Data Engineering. Chicago,USA, 2014: 76-87.
[15] LEE J G, HAN J, LI X. Trajectory outlier detection: a partition-and-detect framework[C]// International Conference on Data Engineering. Cancun, Mexico, 2008: 140-149.
[16] BU Yingyi, CHEN L, FU W C, et al. Efficient anomaly monitoring over moving object trajectory streams [C]// ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Paris, France, 2009: 159-168.
[17] YU Yanwei, CAO Lei, RUNDENSTEINER E A, et al. Detecting moving object outliers in massive-scale trajectory streams [C]//ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, USA, 2014: 422-431.
[18] LI Zhenhui, WANG J, Han J. Mining event periodicity from incomplete observations [C]// ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Beijing, China, 2012: 444-452.

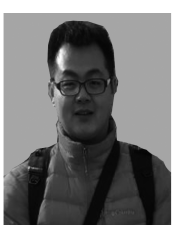
作者简介:



赵冠哲,男,1992 年生,硕士研究生,主要研究方向为数据挖掘。



齐建鹏,男,1992 年生,硕士研究生,主要研究方向为数据挖掘。



于彦伟,男,1986 年生,讲师,博士,主要研究方向为时空数据挖掘、流式数据处理、分布式计算。主持国家自然科学基金青年基金 1 项,参与国家自然科学基金面上项目 1 项,山东省重点研发计划项目 1 项。发表学术论文 30 余篇。