doi:10.11887/j.cn.202002007 http://journal.nudt.edu.cn

高铁大风预警模式挖掘*

滕 飞[1],刘鉴竹[1],祝锦烨[1],勾红叶[2]
(1. 西南交通大学 信息科学与技术学院,四川 成都 611756;2. 西南交通大学 土木工程学院,四川 成都 610031)

摘 要:高铁大风预警的传统方法基于风速预测,当瞬时值高于限速阈值时触发报警,存在大量的误报警,不必要的限速控制影响了高铁行车效率。创新地提出了基于序列模式的预警方法,旨在挖掘报警事件前序数据中的频繁模式,找出报警事件的变化规律,通过滤除与非预警序列共有的频繁模式,得到预警序列独有的序列特征,构建了预警模式库。经兰新高铁沿线的监测数据验证,该方法在提高预测准确率的基础上降低了漏报率,同时有效地减少了模式匹配所需的时间,为提前预警预留充分的时间窗口,更加符合实际应用的需求。

关键词:模式挖掘;大风预警;时间序列;频繁序列;Spark
中图分类号:TN181 **文献标志码**:A **文章编号**:1001-2486(2020)02-055-09

Pattern mining of gale warning for high-speed railway

TENG Fei[1], *LIU Jianzhu*[1], *ZHU Jinye*[1], *GOU Hongye*[2]
(1. School of Information Science and Technology, Southwest Jiaotong University, Chengdu 611756, China;
2. School of Civil Engineering, Southwest Jiaotong University, Chengdu 610031, China)

Abstract: The traditional method of alarming high-speed rail traffic in gale is based on an instantaneous threshold. Although it covers all alarm events, there are a lot of unnecessary alarms, which affect the efficiency of high-speed rail traffic. An early warning method based on sequence pattern was proposed. It aimed at mining frequent patterns in the preorder data and finding out the changing rules of alarm events. The unique sequence characteristics of early warning sequences were obtained by filtering out the public frequent patterns of non-early warning sequences, and a database of early warning patterns was constructed. Through the verification of monitoring data along Lanzhou-Urumchi high-speed railway, the method can improve the accuracy of prediction, and reduce the rate of missing reports concurrently. It reduces the time required for pattern matching effectively, and reserves sufficient time windows for early warning, which can accord more with the practical application requirements.

Keywords: pattern mining; gale warning; time series; frequent sequence; Spark

截至2017年6月,我国已建成73条线路的高铁防灾减灾系统,其中风力监测点2612处,形成了能够实时监测铁路沿线大风情况的高铁防灾网[1],为灾害情况下的行车预警提供了数据支持。目前高铁大风预警采用基于风速预测的人工调度处置模式,但由于风力监测系统与指挥调度系统物理隔离,效率较低,与高速铁路遇大风行车的预警需求还存在一定差距[2]。日本的京叶线引入了大风预警系统,主要根据风速计的监测值预测短期内风速的上限值,当任意监测点预测风速的上限值或检测值超出报警阈值,即发布列车运行限速命令[3],德国高铁也采用了相似的风速短时预报系统[4]。近年来,铁路沿线风速预测的研究受到关注[5-10],但基于单值预测的方法时效性较差,预测的时间粒度在分钟量级,不符合由《高速铁路自然灾害及异物侵限监测系统铁路局中心系统暂行技术条件》规定的高铁行车遇大风的报警事件规则[11-12]。根据《铁路自然灾害及异物侵限监测系统工程涉及暂行规定》,风速持续10 s在15~20 m/s时,行车时速不高于300 km/h;风速持续10 s在20~25 m/s时,行车时速不高于200 km/h;风速持续10 s在25~30 m/s时,行车时速不高于120 km/h;风速持续10 s大于30 m/s时,禁止动车组进入风区,而解除报警则需连续600 s低于超限值[13-14]。

总体来说,现施行的高铁大风预警技术,依赖于预测的准确程度和时效程度,难以达到秒级粒度条件下的高精度预测,不符合现有的高铁遇大风事件的报警规则,易发生误报和漏报。故此,本文提出基于序列模式挖掘的预警方法,将单值预

* 收稿日期:2019-09-20
基金项目:四川省科技计划资助项目(2019YJ0214,2018JY0549,2018JY0294)
作者简介:滕飞(1984—),女,山东泰安人,副教授,博士,硕士生导师,E-mail:fteng@swjtu.edu.cn

测问题转化为序列状态的判定问题,对发生在报警事件前的序列集合进行模式挖掘,找出有意义的预警序列模式,用于预测报警事件的发生。频繁序列模式代表了序列集合的特征,是该序列集合中频繁发生的子序列[13]。频繁序列模式曾在很多领域得到应用,如在气象领域对高温、大风、暴雨等报警事件的预警,能够识别出发生在报警事件前的序列特征,进而做出有效的预警提示。值得注意的是,序列模式虽然能够有效地提前识别出报警事件,但由于非预警序列与预警序列存在一定的相似度,易造成误报,给调度人员传递出不必要的预警信息。因此,去除冗余和无效的频繁序列模式成为解决此问题的关键。

此外,监测系统采集的风数据体量巨大且源源不断。以兰新高铁线为例,每个风监测点一年的采样数据超过100亿条,数据量超过1 TB。序列模式挖掘算法复杂度较高,无法在单机完成计算。因此,预警模式挖掘方法需要支持并行化,增加频繁序列模式的长度[14]。本文选用并行计算框架Spark,实现了基于内存存储的预警模式挖掘并行化算法,相较于传统磁盘阵列存储,可以在极大降低I/O操作时延的同时,保持良好的横向扩展性,适合用于处理实际应用中的工业数据[15]。

针对上述问题,本文提出了基于序列模式挖掘的高铁大风预警方法。通过寻找预警频繁序列和非预警频繁序列的最长公共子序列,剔除频繁序列模式中的冗余和无效模式,构建了预警模式库。利用兰新高铁沿线实测数据对预警模式挖掘和预测方法进行了实验,验证了该方法的有效性、区域适用性和执行效率。该方法在提高预测准确率的基础上降低漏报率,同时减少了模式匹配所需的时间,为提前预警预留充分的时间窗口,更加符合实际应用的需求。

1 相关工作

目前,我国高铁行车指挥采用人工调度处置模式,存在方式落后、效率低下等问题[2]。近年来,有学者研究了如何提高大风预警的准确性。Liu等[5-6,10]先后提出了基于递归经验模态分解的自回归积分滑动平均模型(AutoRegressive Integrated Moving Average model, ARIMA)预测方法和基于小波神经网络的预测方法,在预测精度和时效性方面都取得了较好的结果。然而ARIMA模型的参数定阶问题难以自动化得到结果,而基于小波神经网络方法的预测误差会随着时间的推移而增大,不符合实际需求。针对传统神经网络对非平稳风速的预测精度较差的问题,路学海等[7]提出了改进的量子粒子群算法结合小波神经网络的滚动预测方法,优化了小波神经网络的初始参数;王瑞等[8]利用Kalman滤波去除了数据冗余,提出基于径向基函数(Radial Basis Function, RBF)神经网络的滚动预测方法。上述两项研究均是仿真研究,未能利用铁路沿线的实测风速数据进行实验。Li等[9]将风向引入高铁大风预警问题中,研究了基于Kalman滤波的风速、风向预测,进而输出高铁行车的综合风险系数。

上述基于风速预测的预警技术存在以下两个问题。首先,以上研究侧重于提高风速预测的准确性,而非针对实际的报警事件进行预测。事实上,高铁沿线的风速值预测问题与高铁大风预警问题并不能完全等价。其次,现实中高铁大风报警规则精确到秒,而以上研究中风速序列的时间单位为分钟或小时,难以满足秒级粒度的实际需求。故此,本文将高铁大风的单值预测问题转化为更细粒度的序列状态判定,通过挖掘报警事件前序数据中的频繁模式,找出报警事件的变化规律,符合高铁大风报警规则这类持续超限报警的应用背景。

序列模式挖掘被广泛应用于许多领域的事件预测。严兆斌等[16]针对公路隧道交通事故问题,利用PrefixSpan算法生成了隧道交通事件序列模式,反映隧道交通事件的序列特征,并用于建立隧道交通事件规则库。冯钧等[17]针对洪水预报问题,提出构建适用于待预报流域的暴雨洪水模式库的思想,通过对历史水文数据进行符号化模式挖掘处理,完成中小河流暴雨洪水模式库构建。张光兰等[18]针对通信网络的报警预测问题,提出一种基于拓扑约束的序列模式挖掘方法来发现有意义的报警序列模式以实现报警预测,从而保证通信网络的稳定性和可靠性。Niyazmand等[19]针对报警泛滥问题,利用改进的PrefixSpan算法对天然气加工厂的报警数据做序列模式挖掘,利用得到的频繁序列模式定位造成报警泛滥的原因。Yasmin等[20]对基于序列模式挖掘的分类算法进行改进,在提高分类速度、可扩展性的同时,保持了分类的准确性,解决了高精度的数值数据难以挖掘的问题。此外,序列模式挖掘用于预警的研究还涉及库存采购预警[21]、交通堵塞预警[22]、传感器故障预警[23]等领域,但在列车行车报警预警方面还鲜有涉及[24]。

上述基于模式挖掘的预警研究,均是从报警

事件前的序列集合中挖掘出有意义的模式,以作为预测报警事件的依据。然而,预警数据和非预警数据在一定程度上是具有共性的,若只考虑预警状态到报警状态的模式,会在实际应用中造成大量的误报警。对此,本文不仅针对预警数据做序列模式挖掘,还分析了非预警数据的频繁序列模式,通过去除冗余提高了预警的准确率。

2　问题定义

首先对预警模式挖掘方法所涉及的符号及名称给出相关定义:

定义 1　超限报警事件。根据报警规则(超限值 $OverVal$、超限持续时间 T_{per})确定以起点索引 s_i 和终点索引 e_i 的 n 条报警序列($i \in [1,n]$ 且 $i \in \mathbb{Z}$, $n \geqslant 1$),如图 1 的序列 c 所示。

定义 2　预警序列。给定固定窗口大小为 T_{ew},预警序列是截取每条报警序列数据前 T_{ew} 个值组成的序列,即序列索引在$(s_i - T_{ew}) \sim (s_i - 1)$对应的值构成的序列,如图 1 的序列 b 所示,虽然该序列含有超限数据,但其超限持续时间小于相关阈值,并不符合报警事件规则。

定义 3　非预警序列。非预警序列是序列索引在$(s_i - T_{inter}) \sim (s_i - T_{ew})$的值构成的序列,如图 1 的序列 a 所示。为使预警序列数据和非预警序列数据在相同的窗口大小下进行序列模式挖掘,应将每条序列长度大于 T_{ew} 的非预警序列进行数据分段,以每段序列长度为 T_{ew} 的序列集合形式组成非预警序列集,以保证模式挖掘的结果均是 T_{ew} 内发生的频繁序列模式。

序列模式挖掘是指从序列数据库中寻找频繁子序列作为模式的知识发现过程,即输入一个序列数据库和支持度阈值,输出所有不小于支持度阈值序列的过程[25]。序列模式挖掘算法可分为基于 Apriori 和基于模式增长两类。本文涉及序列模式挖掘算法,并以基于模式增长类的 PrefixSpan 算法为例给出相关定义。

定义 4　子序列。对于序列$a = <a_1, a_2, \cdots, a_n>$和序列 $b = <b_1, b_2, \cdots, b_m>$ $(n \leqslant m)$,若存在数字序列 $1 \leqslant j_1 \leqslant j_2 \leqslant \cdots \leqslant j_n \leqslant m$,满足 $a_1 \subseteq b_{j_1}$, $a_2 \subseteq b_{j_2}, \cdots, a_n \subseteq b_{j_n}$,则称 a 是 b 的子序列。

定义 5　支持度。序列 l 在序列数据库 S 中的支持度为序列数据库 S 中包含序列 l 的序列个数,记为 $support(l)$。最小支持度 sup 定义为支持度与序列总数 $count(S)$ 的值,即$sup = \dfrac{support(l)}{count(S)}$。

定义 6　前缀。设每个元素中的所有项目按照字典顺序排列,给定序列 $a = <a_1, a_2, \cdots, a_n>$ 和 $b = <b_1, b_2, \cdots, b_m>$ $(m \leqslant n)$。如果 $b_i = a_i$ $(i \leqslant m-1)$、$b_m \subseteq a_m$,并且$(a_m - b_m)$中的所有项均在 b_m 中的项后面,则称 b 是 a 的前缀。

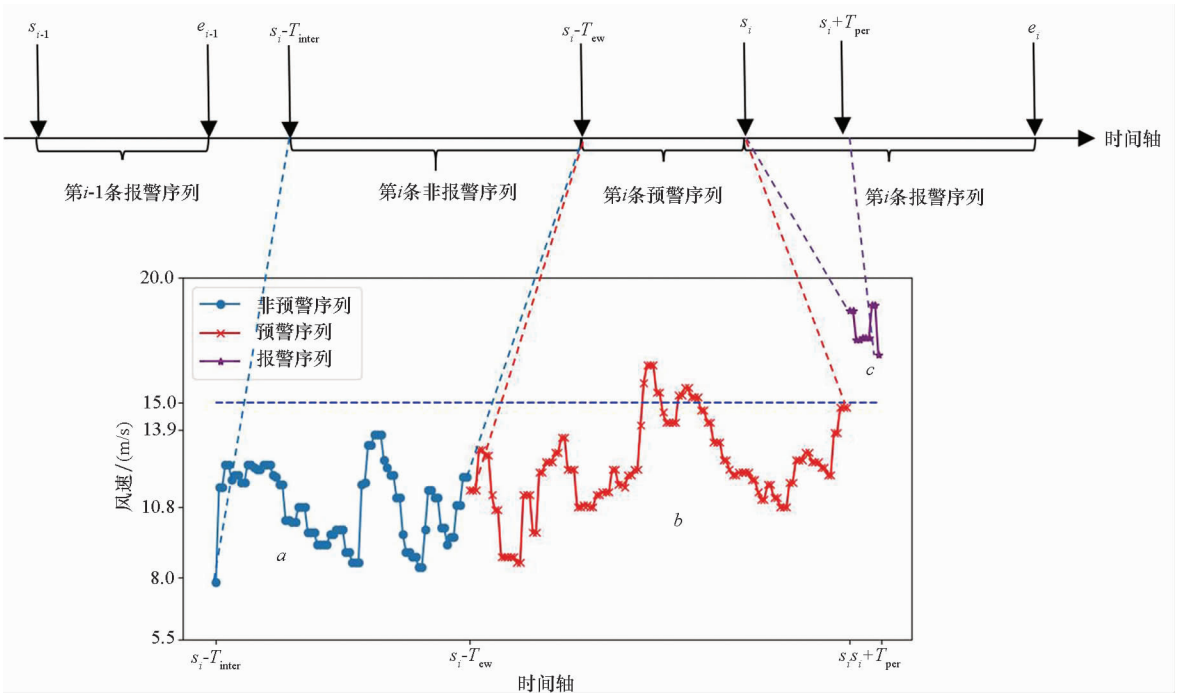


图 1　预警序列及非预警序列

Fig. 1　Alarm sequence and non-alarm sequence

定义 7 投影。给定序列 a 和 b,如果 b 是 a 的子序列,则 a 对应于前缀 b 的投影 a'需满足 b 是 a'的前缀;a'是 a 满足上述条件的最大子序列。

定义 8 后缀。设序列 a 关于子序列 $b = <b_1, b_2, \cdots, b_{m-1}, b_{m'}>$ 的投影为 $a' = <b_1, b_2, \cdots, b_n> (n > m)$,则序列 a'关于子序列 b 的后缀为 $<b_{m''}, b_{m+1}, \cdots, b_n>$,其中 $b_{m''} = (b_m - b_{m'})$。

定义 9 投影数据库。设 a 为序列数据库 S 中的一个序列模式,序列 b 以 a 为前缀,则 a 的投影数据库为 S 中所有以 a 为前缀的序列相对于 a 的后缀,记为 $S|_a$。

例:考虑表 1 中的序列集合,设置支持度阈值 *sup* 为 0.75。

1)对数据库中的所有项进行支持度统计,将支持度低于阈值的项从数据库中删除,得到前缀长度为 1 的频繁序列模式$\{<A>:4, <B>:2\}$;

2)对于每个长度为 1 满足支持度要求的前缀进行递归挖掘,以 $<A>$ 为前缀投影数据库的结果为$\{<(_B), B>, <(AB), B>, <B>, <(_B), A, B>\}$,满足支持度要求的有$\{B:4, _B:3\}$,即以 $<A>$ 为前缀挖掘到的前缀长度为 2 的频繁序列模式有$\{<A, B>, <(AB)>\}$,同理可得以 $<B>$ 为前缀挖掘到的前缀长度为 2 的频繁序列模式有$\{<B, B>\}$,即前缀长度为 2 的频繁序列模式为$\{<A, B>, <(AB)>, <B, B>\}$;

3) 递归执行 2),得到前缀长度为 3 的频繁序列模式有 $<(A, B), B>$。若投影数据库为空或投影数据库中各项支持度计数都低于阈值则递归返回。综上,在该例中由 PrefixSpan 算法得到大于支持度阈值 0.75 的频繁序列模式集为$\{<A>, <B>, <A, B>, <(AB)>, <(A, B), B>\}$。

表 1 序列数据集示例

Tab. 1 An example of sequence dataset

序列编号	序列
1	$<(AB), B, C>$
2	$<(AC), (ABC), B>$
3	$<A, B>$
4	$<(AB), A, B>$

3 预警序列模式挖掘方法

为了除去预警数据和非预警数据的公共频繁序列,降低将非预警序列识别成预警序列的概率,本节提出预警序列模式挖掘方法,其流程如图 2 所示。主要步骤有:①根据报警规则生成预警序列数据和非预警序列数据;②利用序列模式挖掘算法生成预警频繁序列和非预警频繁序列;③采用最长公共子序列算法去除预警数据和非预警数据的公共频繁序列,以生成预警模式库;④在测试或实际应用中,将目标序列与预警模式库中的模式进行模式匹配,若能匹配成功则表示在这段时间后将发生报警事件,应及时通知相关人员发布预警信号。

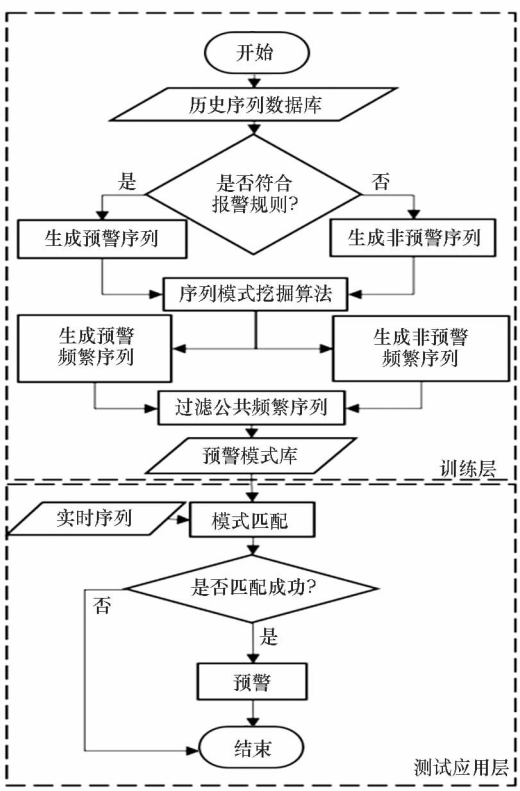


图 2 预警序列模式挖掘方法流程图

Fig. 2 Flow chart of sequential pattern mining for early warning

3.1 预警模式库构建

首先根据第 2 节中对于预警序列和非预警序列的定义,在原始的时间序列数据中筛选出报警事件数据、预警序列数据和非预警序列数据。考虑到时间序列普遍维度很高,不能直接进行模式挖掘,且对于需要对每个前缀做投影操作的序列模式挖掘算法而言,若存在大量的单项,会直接增加算法的时间复杂度和空间复杂度。为解决时间序列的高维特性,通常会进行时间序列符号化操作,该操作能提高存储的效率,加快处理速度。

为了降低将非预警序列识别成预警序列的概率,分别对预警序列和非预警序列做序列模式挖掘,进而得到两者的公共频繁序列模式,即不能明显区分是否预警的序列特征,在预警序列的频繁

模式集中除去包含非预警频繁序列的特征,仅保留能准确辨识预警的分类依据,完成预警模式库的构建。

3.2　预警模式匹配

最长公共子序列(Longest Common Subsequence, LCS)算法旨在求得两个或多个已知序列的最长子序列[26]。此处的子序列是指从序列中得到各个元素属于原始序列(顺序且不一定连续的序列)。如字符串序列“aabcd”和“abcabc”中,两者的公共子序列有顺序且相邻的“abc”,也有顺序且不相邻的“aabc”,所以两者的最长公共子序列为“aabc”。该算法在预警模式挖掘方法流程中,应用于去除预警序列和非预警序列的公共频繁序列模式和验证是否预警,其实现的功能均为在目标序列中检测是否存在非预警频繁序列的特征。

当序列的数量一定时,LCS 求解可以采用动态规划方法,以空间换时间的思想,用数组保存中间状态,方便后续计算。记序列 X 的前 i 个元素为 $X_i=\{x\_1, x\_2,\cdots, x\_i\}$,序列 Y 的前 j 个元素为 $Y_j=\{y\_1, y\_2,\cdots, y\_j\}$;序列 X 和序列 Y 的最长公共子序列为 LCS_{XY};序列 X 的前 i 个元素和序列 Y 的前 j 个元素的最长公共子序列为 $LCS_{X_iY_j}$。设置二维数组 $C[i,j]$ 记录 X 和 Y 子序列的最长公共子序列长度,则可得到式(1)的状态转移方程。根据式(1),可通过递归计算出每个子序列的 LCS,并依据二维数组 C 回溯得到最长公共子序列 LCS_{XY}。通过基于 LCS 的模式匹配方法,求得预警模式和测试集中每条窗口序列的最长公共子序列,若最长公共子序列与该预警模式相等则表示匹配成功。

$$C[i,j]=\begin{cases}0 & i=0\text{ 或 }j=0\\ C[i-1,j-1]+1 & i,j>0\text{ 且 }x_i=y_j\\ \max(C[i,j-1],C[i-1,j]) & i,j>0\text{ 且 }x_i\neq y_j\end{cases}\tag{1}$$

在第 2 节中,从基于 PrefixSpan 算法的频繁序列模式挖掘的示例中可以发现,原始算法的结果集为所有满足支持度阈值的频繁序列模式集,而对于预警模式库来说,长度较短的频繁序列模式不宜作为预警模式,由相同前缀组成的非最长频繁模式会造成结果集的冗余,且模式库中会因此类频繁序列模式的存在而增加模式验证的运行时间[27]。算法 1 给出了预警序列模式挖掘算法,分别以相同支持度阈值挖掘预警序列数据库和非预警序列数据库中大于模式长度阈值且由相同前缀组成的最长频繁序列模式,并采用 LCS 算法在预警频繁序列中除去包含非预警频繁序列模式的特征,完成预警序列模式库的构建。

算法 1　预警序列模式挖掘算法

Alg. 1　Early warning sequence pattern mining algorithm

Input:预警序列数据库 $S+$,非预警序列数据库 $S-$,支持度阈值 sup,频繁模式最大长度 $L_{\max}$,频繁模式最小长度 $L_{\min}$

Output:预警模式库 $pats$

```
pats ←Ø
得到预警频繁序列模式集和非预警频繁序列模式集
P+ ← patternMining(S+, sup, Lmax)
P- ← patternMining(S-, sup, Lmax)
过滤预警频繁模式中长度小于阈值和包含非预警频繁序列特征的无效模式
for each s in P+
    if lenLarger(s, Lmin) and LCS(s) ∉ P- then
    pats ←union(pats,s)
    end if
end for
过滤冗余模式,保留由相同前缀组成的最长频繁序列模式
for each pat in pats
    while slice (pat, length) ≠Ø do
     length ← Lmin;
        if slice(pat, length) ∈ pats then
        pats ← diff(pats, slice(pat, length));
        end if
     length← length + 1
end while
end for
return pats
```

4　实验结果与分析

4.1　实验环境

本文数据存储介质为 Oracle 11 g,在 Spark2.7.0 搭建的完全分布式环境下进行实验。Spark 集群总节点数为 3,每个节点的配置为:Intel(R) Xeon(R)四核 E5-2620 2.10 GHz 处理器;8 G 内存;Ubuntu 16.04LTS 的操作系统。

实验采用兰新高速铁路沿线风基站 2017 年实测数据,其采样频率为 1 Hz。在模式挖掘实验之前对原始数据进行了缺失值、异常值等数据预处理。

4.2 预警模式挖掘实验

选用兰新高铁沿线位于百里风区的5696风基站实测的共15 638 400个风速值构成的时间序列。根据高铁行车限速规定,设置风速超限值 $OverVal = 15$ m/s;超限持续时间 $T_{per} = 10$ s。根据《客运专线防灾安全监控系统总体技术方案》,对风速变化快的强对流短时大风,预警时间不少于2 min;而对风速变化慢的季节性大风,预警时间不少于5 min。因实验数据属于百里风区,故设置预警时间 $T_{ew} = 120$ s,非预警时间 $T_{inter} = 7200$ s。

在上述实验过程的符号化阶段,考虑到大风报警事件规则要求风速超限具有持续性的特点,将某一范围的风速值映射为某一符号,表2为符号化映射规则(风速单位为m/s)。风速小于*OverVal*超限值(15 m/s)按照风力等级对照表划分,大于*OverVal*超限值(15 m/s)按照报警事件的临界风速划分。

表2 符号化映射规则

Tab. 2 Symbolic mapping rules

风速范围	符号表示	风速范围	符号表示
[0.0, 0.2]	'a'	[10.8, 13.8]	'g'
[0.3, 1.5]	'b'	[13.9, 15.0]	'h'
[1.6, 3.3]	'c'	[15.0, 20.0]	'i'
[3.4, 5.4]	'd'	[20.0, 25.0]	'j'
[5.5, 7.9]	'e'	[25.0, 30.0]	'k'
[8.0, 10.7]	'f'	[30.0, $+\infty$]	'l'

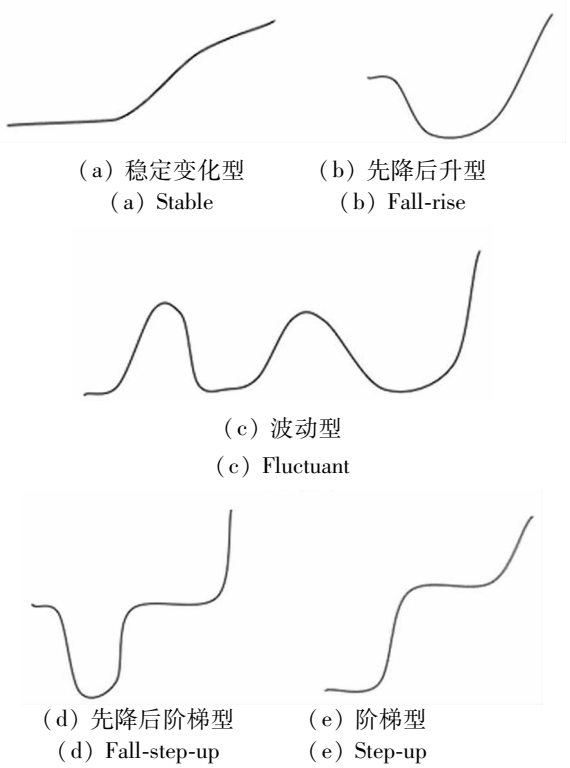
为了检验该预警模式挖掘方法的有效性,据二八定律,将预警序列集按报警时间先后顺序以8∶2的比例分为预警训练集和预警测试集,并在非预警序列集上以相同比例划分为非预警训练集和非预警测试集。为使得序列模式挖掘算法得到的频繁模式均发生在一定的时间内,固定窗口大小为预警时间(本实验中 $T_{ew} = 120$ s),将非预警序列集中大于窗口大小的序列划分长度为 T_{ew} 子序列。本实验利用训练集中的预警序列和非预警序列做预警模式挖掘,在测试集中进行预警模式验证实验。预警模式挖掘算法中的输入参数 *sup*,L_{max},L_{min} 分别设置为0.3、120、60,一共得到15 442条序列长度大于 L_{min} 的频繁模式,经过滤相同前缀组成的非最长频繁模式后,保留8009条由前缀组成的最长频繁模式作为预警模式库。部分具有典型的预警模式见表3(由于模式长度较长,在表3中将预警模式库的结果表示为['符号':持续出现的次数]的字典集合),表中风速单位为m/s。

表3 部分典型预警模式

Tab. 3 Partial typical early warning patterns

符号:持续次数	风速区段:持续次数
['g':87]	[[10.8, 13.8]:87]
['g':26, 'f':34]	[[10.8, 13.8]:26, [8.0, 10.7]:34]
['g':30, 'h':1, 'g':4, 'h':2, 'g':23]	[[10.8, 13.8]:30, [13.9, 15.0]:1, [10.8, 13.8]:2, [13.9, 15.0]:23]
['g':8, 'f':2, 'g':56]	[[10.8, 13.8]:8, [8.0, 10.7]:2, [10.8, 13.8]:56]
['g':84, 'h':2]	[[10.8, 13.8]:84, [13.9, 15.0]:2]

由预警模式库可知,在1级报警事件发生前风速不会发生大幅度的突变,且风速在8~15 m/s范围内变化。图3的五种典型预警模式,其"典型"在风速在不同区段间的趋势变化规律,"非典型"的预警模式与这五种的区别在于风速在不同区段持续的次数不同,其形状规律与以上五种相似,如图3所示。



(a) 稳定变化型 (a) Stable

(b) 先降后升型 (b) Fall-rise

(c) 波动型 (c) Fluctuant

(d) 先降后阶梯型 (d) Fall-step-up

(e) 阶梯型 (e) Step-up

图3 预警模式大致形状总结

Fig. 3 Shapes of early warning patterns

4.3　预警模式验证实验

本实验的验证指标如下:

1)准确率 $=\frac{n_{\text{alarm\_}p}+n_{\text{not\_alarm\_}n}}{n_{\text{test}}}$,其中 n_{test} 为测试序列集中序列的数量,$n_{\text{alarm_}p}$是应该预警且正确预警的事件数量,$n_{\text{not_alarm_}n}$是不应该预警且没有预警的事件数量,该指标的含义为正确预测的比例。

2)漏报率 $=\frac{n_{\text{alarm\_}n}}{n_{\text{alarm\_false}}}$,其中 $n_{\text{alarm\_false}}$ 为测试序列集中未预警的事件数量,$n_{\text{alarm_}n}$为应该预警但没有预警的事件数量。该指标代表的含义为在预警事件识别时遗漏的比例。

本实验分别选用 FPGrowth、PrefixSpan 两种不同的模式挖掘算法进行验证,序列模式挖掘算法 FPGrowth 需要反复循环扫描数据库,复杂度较高。而 PrefixSpan 挖掘算法,无须生成候选序列,数据量较大的情况下只需扫描 2 次数据库进行递归计算,效率更好。选取兰新高铁沿线位于百里风区的 5696 风基站在 2017 年实测的风速序列,该实验旨在验证去除非预警频繁序列对预警准确率的影响。

表 4 中,传统方法只考虑预警序列模式,而本文方法剔除了非预警频繁序列。实验结果表明,基于序列模式挖掘的预警方法能够识别出报警前的序列特征,起到预警作用,对比传统序列模式挖掘方法,其准确率和漏报率都有所改进,原因是传统序列模式挖掘方法不能有效区别预警序列和非预警序列共有的频繁模式,这一点在 FPGrowth 上表现得尤为明显。FPGrowth 算法特殊的数据组织形式,要求一个序列中的元素最多出现一次,使得挖掘得到的序列模式的种类少于 PrefixSpan 算法,但却包含了更多的公共频繁模式。因此,FPGrowth 极易产生假预警,将非预警序列预测为预警序列,导致准确率很低。由此可见,预警模式挖掘方法可对冗余和无效的模式进行有效过滤。

表 4　模型性能

Tab. 4　Model performance

%

挖掘算法	准确率	漏报率
FPGrowth – 传统方法	3.6	3.1
FPGrowth – 本文方法	79.4	0.7
PrefixSpan – 传统方法	64.4	0.8
PrefixSpan – 本文方法	67.4	0.7

为了验证本文算法的区域适用性,选取兰新高铁沿线多个风基站进行该验证实验。表 5 显示了兰新高铁沿线的 8528、8496、8512 三个风基站在 2017 全年数据的预警结果,准确率和漏报率这两个指标均保持较好的一致性,说明本文提出的方法具有较好的区域适用性。

表 5　区域适用性

Tab. 5　Regional applicability

%

风基站	准确率	漏报率
8528 基站	71.0	0.1
8496 基站	75.5	0.2
8512 基站	71.5	0.3

4.4　执行效率实验

由于单机上计算超时,本文采用 3 个节点的 Spark Cluster 进行并行计算。不同长度预警序列的匹配时间对比情况如表 6 所示。基于模式序列的预警方法旨在在较短的时间内相对高精度地预测出未来的报警事件。由于模式匹配算法复杂度较高,需要留出充分的时间窗口才能达到预警效果。期望的预警时间越早,预留给匹配的时间就越长。本文提出的预警方法较传统方法可降低匹配时间。对于 2 min 预警序列,本文方法可将预警窗口从 113.2 s 延长至 116.6 s。对于20 min预警序列,采用传统方法,匹配时间消耗 19.5 min,实际有效预警窗口仅为 0.5 min,而本文方法可将有效预警窗口延长至 9.1 min,这说明本文方法具有较高的工程实用性。

表 6　任务运行时间比较

Tab. 6　Runtime comparison

	20 min 预警序列	2 min 预警序列
传统方法	19.5 min	6.8 s
本文方法	10.9 min	3.4 s

5　结论

本文针对兰新高铁动车组列车遇大风行车限速报警事件进行了预警方法研究。针对单值预测法精度不够且不符合高铁大风事件的报警规则,提出了基于序列模式的预警方法。考虑到非预警序列模式对预警准确率的影响,提出了预警和非预警序列的公共频繁模式过滤方法,从而得到预警序列独有的序列特征,完成了高铁行车遇大风预警模式库构建,并使用兰新高铁沿线的多个风

基站数据验证了该方法的有效性,证明了该方法具有区域适用性。该预警模式挖掘方法从实际应用出发,可在提高预测准确率的基础上降低漏报率,同时有效地减少了模式匹配所需的时间,为提前预警预留更长的时间窗口,符合实际应用的需求。

参考文献(References)

[1] 中国铁路总公司. 我国高速铁路防灾减灾现状分析研究[R]. 北京: 中国铁路总公司, 2017.
China Railway Corporation. Analysis and research on the present situation of disaster prevention and mitigation of high-speed railway in China [R]. Beijing: China Railway Corporation, 2017. (in Chinese)

[2] 朱亮. 高速铁路大风环境行车预警技术研究[J]. 铁道运输与经济, 2018, 40 (12): 76 – 82.
ZHU Liang. A research on an early warning technology of high-speed railway driving in strong wind environment[J]. Railway Transport and Economy, 2018, 40(12): 76 – 82. (in Chinese)

[3] 王瑞, 陈苒, 包云. JR 东日本铁路大风监测技术研究[J]. 中国铁路, 2018(7): 96 – 102.
WANG Rui, CHEN Ran, BAO Yun. The study on JR-east monitoring technology of strong wind[J]. Chinese Railways, 2018(7): 96 – 102. (in Chinese)

[4] Hoppmann U, Koenig S, Tielkes T, et al. A short-term strong wind prediction model for railway application: design and verification [J]. Journal of Wind Engineering and Industrial Aerodynamics, 2002, 90(10): 1127 – 1134.

[5] Liu H, Tian H Q, Li Y F. An EMD-recursive ARIMA method to predict wind speed for railway strong wind warning system [J]. Journal of Wind Engineering and Industrial Aerodynamics, 2015, 141: 27 – 38.

[6] 刘辉, 田红旗, 李燕飞, 等. 铁路风速单步高精度混合预测性能对比研究[J]. 铁道学报, 2016, 38(8): 41 – 49.
LIU Hui, TIAN Hongqi, LI Yanfei, et al. Study on performance comparison of wind speed hybrid high-precision one-step predicting models along railways[J]. Journal of the China Railway Society, 2016, 38(8): 41 – 49. (in Chinese)

[7] 路学海, 潘迪夫, 韩锟, 等. 基于改进的 QPSO-WNN 滚动算法的铁路沿线短期风速预测[J]. 铁道科学与工程学报, 2016, 13(5): 978 – 984.
LU Xuehai, PAN Difu, HAN Kun, et al. Railway short-term wind speed prediction based on improved QPSO-WNN rolling algorithm[J]. Journal of Railway Science and Engineering, 2016, 13(5): 978 – 984. (in Chinese)

[8] 王瑞, 史天运, 王彤. 基于 RBF 神经网络的铁路沿线短时风速预测方法[J]. 中国铁道科学, 2011, 32(5): 132 – 134.
WANG Rui, SHI Tianyun, WANG Tong. Prediction method for short-time wind speed along railway based on RBF neural network[J]. China Railway Science, 2011, 32(5): 132 – 134. (in Chinese)

[9] Li Y T, Huang H, Wang H Y, et al. A strong wind warning method for high speed train using Kalman filter [C]// Proceedings of 2nd International Conference on Remote Sensing, Environment and Transportation Engineering, 2012: 1 – 3.

[10] Liu H, Tian H Q, Li Y F. Short-term forecasting optimization algorithms for wind speed along Qinghai-Tibet railway based on different intelligent modeling theories [J]. Journal of Central South University of Technology (English Edition), 2009, 16(4): 690 – 696.

[11] Hu W K, Chen T W, Shah S. Detection of frequent alarm patterns in industrial alarm floods using itemset mining methods[J]. IEEE Transactions on Industrial Electronics, 2018, 65(9): 7290 – 7300.

[12] Yusof N, Zurita-Milla R, Kraak M, et al. Mining frequent spatio-temporal patterns in wind speed and direction[M]// Connecting a Digital Europe through Location and Place. Springer International Publishing, 2014.

[13] 李海林, 邬先利. 基于频繁模式发现的时间序列异常检测方法[J]. 计算机应用, 2018, 38(11): 3204 – 3210.
LI Hailin, WU Xianli. Time series anomaly detection method based on frequent pattern discovery[J]. Journal of Computer Applications, 2018, 38(11): 3204 – 3210. (in Chinese)

[14] 何清, 庄福振. 基于云计算的大数据挖掘平台[J]. 中兴通讯技术, 2013, 19(4): 32 – 38.
HE Qing, ZHUANG Fuzhen. Big data mining platform based on cloud computing[J]. ZTE Technology Journal, 2013, 19(4): 32 – 38. (in Chinese)

[15] 胡小强, 吴翾, 闻立杰, 等. 基于 Spark 的并行分布式过程挖掘算法[J]. 计算机集成制造系统, 2019, 25(4): 791 – 797.
HU Xiaoqiang, WU Xuan, WEN Lijie, et al. Parallel distributed process mining algorithm based on Spark [J]. Computer Integrated Manufacturing Systems, 2019, 25(4): 791 – 797. (in Chinese)

[16] 严兆斌, 方敏. 基于序列模式挖掘的公路隧道交通事件检测[J]. 计算机应用与软件, 2010, 27(1): 204 – 206, 249.
YAN Zhaobin, FANG Min. Traffic incidents detection for highway tunnel based on sequential pattern mining [J]. Computer Applications and System, 2010, 27(1): 204 – 206, 249. (in Chinese)

[17] 冯钧, 郭涛, 陈志飞. 基于模式挖掘的中小河流暴雨洪水模式库[J]. 计算机与现代化, 2018(12): 32 – 39, 121.
FENG Jun, GUO Tao, CHEN Zhifei. Storm flood pattern library in middle and small rivers based on pattern mining[J]. Computer and Modernization, 2018(12): 32 – 39, 121. (in Chinese)

[18] 张光兰, 杨秋辉, 程雪梅, 等. 序列模式挖掘在通信网络告警预测中的应用[J]. 计算机科学, 2018, 45(z2): 535 – 538, 563.
ZHANG Guanglan, YANG Qiuhui, CHENG Xuemei, et al. Application of sequence pattern mining in communication network alarm prediction [J]. Computer Science, 2018, 45(z2): 535 – 538, 563. (in Chinese)

[19] Niyazmand T, Izadi I. Pattern mining in alarm flood sequences using a modified PrefixSpan algorithm [J]. ISA Transactions, 2019.

[20] Yasmin R Y, Sakya A E, Merdijanto U. A classification of sequential patterns for numerical and time series multiple source data —a preliminary application on extreme weather prediction[C]// Proceedings of International Conference on Data and Software Engineering (ICoDSE), 2017: 1 – 5.

[21] Li L, Wang X J, Huang X, et al. Enterprise lean catering material management information system based on sequence

pattern data mining[C]//IEEE 4th International Conference on Computer and Communications (ICCC), 2018: 1757 - 1761.

[22] Gao J, Sun Y, Liu W H, et al. Predicting traffic congestions with global signatures discovered by frequent pattern mining[C]. IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2016: 554 - 560.

[23] Baek S, Kim D. Fault prediction via symptom pattern extraction using the discretized state vectors of multisensor signals[J]. IEEE Transactions on Industrial Informatics, 2019, 15(2): 922 - 931.

[24] Wang R, Zhao F X, Shi T Y, et al. Reducing negative influence of strong wind on normal train operations based on data mining[C]. International Conference on Intelligent Rail Transportation (ICIRT), 2018: 1 - 5.

[25] Patel R, Chaudhari T. A review on sequential pattern mining using pattern growth approach[C]// Proceedings of IEEE International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), 2016.

[26] 李城, 沙俊淞, 武文. 基于最长公共子序列的微博谣言溯源研究[J]. 计算机与现代化, 2018(1): 107 - 112.
LI Cheng, SHA Junsong, WU Wen. Research on origin of micro-blog rumors based on longest common subsequence[J]. Computer and Modernization, 2018(1): 107 - 112. (in Chinese)

[27] 杨斐, 张万桢, 陆垂伟. 一种无候选项的闭合序列模式挖掘算法[J]. 计算机应用与软件, 2016, 33(3): 279 - 283.
YANG Fei, ZHANG Wanzhen, LU Chuiwei. A closed sequential pattern mining algorithm without candidate terms[J]. Computer Applications and Software, 2016, 33(3): 279 - 283. (in Chinese)