Time series data cleaning论文翻译1-3页

摘要

错误在时间序列数据中很普遍，例如 GPS 轨迹或传感器读数。现有方法更多地关注异常检测，而不是修复检测到的异常。通过简单地通过异常检测过滤掉脏数据，应用程序在不完整的时间序列上仍然可能不可靠。我们建议（迭代地）修复时间序列数据中的异常，而不是简单地丢弃异常，通过创造性地将异常检测中时间性质的美与数据修复中广泛认为的最小变化原则结合起来。我们的主要贡献包括：（1）时间序列数据上的迭代最小修复（IMR）的新框架，（2）对所提出的迭代最小修复收敛性的显式分析，以及（3）每次迭代中参数的有效估计。值得注意的是，通过增量计算，我们将参数估计的复杂度从 O(n) 降低到 O(1)。与最先进的方法相比，在真实数据集上的实验证明了我们的提议的优越性。特别是，我们表明（建议的）修复确实改善了时间序列分类应用。

引言

时间序列数据通常带有脏值或不精确值，例如 GPS 轨迹、传感器读取序列 [15]，甚至股票价格 [16]。例如，SAL VEPAR (SY) 的价格被误用为 SYBASE (SY) 的价格，在某些来源中两者共享相同的符号 (SY)。它与现实生活中实际发生的有趣异常不同，例如，当冷空气突然涌入时，温度会在一天内从 20°C 突然变为 10°C。为了区分这种情况，我们建议使用一些标记肮脏观察的真相。 （请参阅示例 1 中有关脏数据及其标记真相的更详细示例。）

* 1. 异常修复的动机 基于脏时间序列数据的应用程序，例如模式挖掘 [17] 或分类 [26] 显然是不可靠的。时间序列上的异常检测通常用于过滤掉脏数据（有关异常检测技术的全面和结构化概述，请参见 [11]）。也就是说，检测到的异常数据点被简单地作为无用的噪声丢弃。不幸的是，由于大量连续数据点被消除，应用程序几乎无法在相当不完整的时间序列上执行。最近的研究 [21] 表明，修复脏值可以改善空间数据的聚类。对于时间序列数据，我们认为修复异常也可以改进时间序列分类等应用[26]。接近真相的修复极大地帮助了应用程序。

1.2

修复的潜在方法一个简单的想法是直接解释异常检测中的预测值，例如，通过 AR [3, 14, 27] 或 ARX [3, 19]，作为修复（参见第 2 节中的详细信息）。如果数据点的（真实）预测与（嘈杂的）观察显着不同，则数据点被视为异常。不幸的是，在实践中，嘈杂/错误的数据通常接近事实，直觉是人类或系统总是试图尽量减少他们的错误，例如拼写错误（John Smith vs. Jhon Smith），错别字（555-8145 vs. 555- 8195），如 [1] 所示，四舍五入（76,821,000 vs. 76M）或单位误差（76M vs. 76B）如 [16] 所示。由于这种分歧，直接应用异常检测技术的修复性能很差，如下面的示例 1 和第 6 节中的实验所示。另一方面，基于约束的修复 SCREEN [22]，严格遵循最小变化原则在数据修复 [1] 中，很大程度上依赖于对值变化速度的适当约束。修复是基于两个连续点进行的，即只考虑一个历史点，因此不会感知错误的时间性。如下例 1 所示，基于速度约束的 SCREEN 在修复尖峰错误方面是有效的，但很难处理一系列连续的脏点。

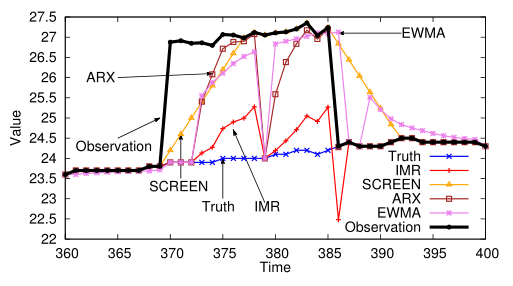


图 1：传感器读数的示例片段简而言之，异常检测方法并不期望在实践中将错误最小化，而基于约束的修复在解决错误的时间性质方面无效。

1.3

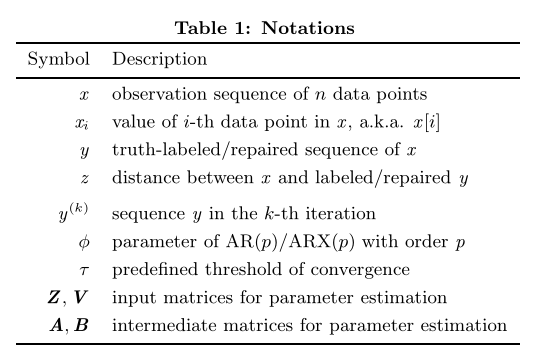
我们建议的直觉 由于完全自动的数据修复可能无法很好地修复时间序列数据（例如我们在图 9 中的实验中观察到的 SCREEN [22]），这受到利用主数据（高质量数据的单一存储库）的想法的启发) 在数据修复 [9] 中，我们使用一些脏观察的标记真相来推进修复。可以通过手动标记或通过更可靠的来源自动获得真相。例如，通过用户登记活动在地图中手动标记准确位置（并用于修复不精确的 GPS 读数）。此外，在某些情况下可能会发生周期性的自动标记，例如，精确的设备在相对较长的感应周期内报告准确的空气质量数据（作为标记值），而人群和参与式感应会以恒定的方式产生不可靠的观察结果 [28]。意识到异常检测中的错误性质和数据修复中的最小变化原则，我们提出了迭代最小修复（IMR）。所提出的迭代最小修复背后的理念是，前迭代中的高置信度修复可以帮助后修复。具体而言，IMR 每次最小更改一个点，只为获得最自信的修复，指的是人类或系统总是试图将错误最小化的数据修复中的最小更改原则。高置信度修复与错误点的标记真相一起用于学习和增强异常检测中错误的时间性质，从而在后面的迭代中生成更准确的修复候选。

示例 1。图 1 显示了传感器读数的示例片段，用黑线表示。假设传感器错误发生在从时间点 370 到 385 的时间段内，在此期间，由于粒度不匹配或单位错误，观察结果偏离了事实。为了修复错误，标记了几个观察的真实性，包括时间点 {370, 371, 372, 379, 387}。现有的基于速度约束的清洗 (SCREEN) [22] 无法有效修复一段时间内的此类连续错误（这在 [22] 的示例 1 中也确实存在）。原因是限速，限制量f 值相对于时间差变化，可以检测到急剧的偏差，例如从时间点 369 到 370，但不能检测连续误差，例如，在 383 和 384 中。指数加权移动平均（EWMA）[13] 算法也很难找到清洁痕迹的正确方法。这两种方法具有相似的修复痕迹。通过将异常检测中的预测值视为修复（参见第 2 节中的更多详细信息），还报告了基于 ARX 的修复的结果。 ARX 考虑到真值和观测值之间的误差，显示出比 EWMA 和 SCREEN 方法更好的修复结果。最后，我们提出的 IMR 方法，同时考虑了错误预测和最小变化，获得了最接近真相的修复。迭代最小修复带来了新的挑战：（1）修复过程是否收敛； (2) 如何在每次迭代后在修复后的数据上有效/增量地更新时间模型的参数。异常检测研究中没有考虑异常修复中的这两个问题。

贡献。我们在本文中的主要贡献总结如下。 (1) 我们在第 2 节中将异常修复问题形式化，给定一个带有一些标记为真值的时间序列。引入了现有异常检测技术（例如 AR 和 ARX）的适应以进行异常修复。 (2) 我们在第 3 节中设计了一种迭代最小变化感知修复算法 IMR。值得注意的是，我们说明了基于 ARX 的方法（在第 2 节中）确实是具有静态参数的 IMR 的特例（命题 2）。 (3) 我们在第 4 节中研究了 IMR 在各种场景下的收敛性。特别是针对 p = 1 阶的 IMR(1) 的特殊情况明确分析了收敛性，这足以在实践中实现高修复精度（如第 6 节中的实验所示）。我们证明了在一定的输入下，可以直接计算收敛修复结果，无需迭代计算（命题 8）。 (4) 我们在第 5 节中为每次修复迭代中的参数估计设计了有效的剪枝和增量计算。不是对所有 n 点执行参数估计，而是可以通过简单地删除值为 0 的行来剪枝用于参数估计的矩阵（命题 9 ）。同样值得注意的是，不同修复迭代之间的增量计算（命题 10）可以进一步将参数估计的复杂度从 O(n) 降低到 O(1)。 (5) 在第 6 节中对具有真实错误和合成错误的真实数据集进行的实验表明，我们的 IMR 方法显示出比最先进的方法显着更好的修复性能，包括上述异常检测和基于约束的修复。表 1 列出了本文中常用的符号。所有的证明都可以在完整的技术报告中找到（http://ise.thss.tsinghua.edu.cn/sxsong/doc/anomaly.pdf）。

1. 预备知识

本节首先介绍异常修复的问题。然后，我们将现有的异常检测模型用于异常修复，即不考虑标记数据的 AR 和支持标记数据的 ARX。



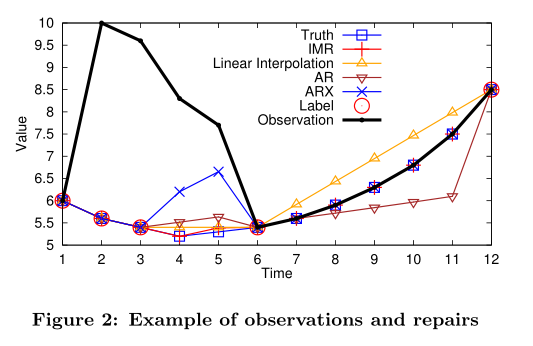
这种简单适应的主要问题是：（1）将具有显着差异的预测应用于观察，因为修复与数据修复中的最小变化原则相矛盾[1]，如引言中所述。 (2) 需要预设一个静态参数（以下等式 1 和 3 中的 φ），例如，在初始化期间根据脏数据估计。

2.1

问题陈述 考虑n个观察的时间序列，x = x[1], 。 . . , x[n]，其中每个 x[i] 是第 i 个数据点的值。为简洁起见，我们将 x[i] 写为 xi。让 y 表示 x 的标记/修复序列。每个 yi 要么是标记的真值，要么是 xi 的修复值。给定时间序列 x 和 x 的部分标记子集 y，修复问题是确定未在 y 中标记的 xi 的修复 yi。

示例 2

（观察 x，部分标记为 y，完全修复 y）。考虑 x = {6, 10, 9.6, 8.3, 7.7, 5.4, 5.6, 5.9, 6.3, 6.8, 7.5, 8.5}，图 2 中有 12 个观测数据点，其中移动（向上）误差发生在四个点 x2 上， . . . , x5。假设五个点被标记为真值，即部分标记为 y。通过修复（使用下面介绍的方法），我们建议获得完全修复的 y，例如，y = {6, 5.6, 5.4, 5.2, 5.4, 5.4, 5.6, 5.9, 6.3, 6.8, 7.5, 8.5}，如图所示在图 2 中。在修复后的 y 中，x4 和 x5 分别从 8.3 和 7.7 变为 5.2 和 5.39。标注的 y2 和 y3 在修复结果中不会被修改。图 2 还通过将点与标记值连接的方法（即线性插值 [25]）呈现了另一种修复 y'。如图 2 所示，也在 [22] 中指出，这种（平滑）方法的主要问题是几乎所有（未标记）数据点的严重损坏，例如 y′7 。 . . y′11，原来是正确的，不应该修改。相比之下，我们提出的方法在保留 y7 的同时修复 y4 和 y5。 . . y11 不变。



2.2 AR 模型

直观地说，异常检测技术可以适用于异常修复。例如，我们考虑 AR（自回归）模型 [14, 27] 如下： x ′t = c + pX i=1 φixt−i + ǫt (1) 其中 x ′t 是 xt 的预测，p 是阶，φi 是模型的参数，c 是由 c = µ(1-pP i=1 φi)，μ是过程的平均值，εt是白噪声（通常是高斯白噪声[3]，根据高斯分布生成的正态随机变量，均值μ=0，方差σ2；换句话说, c = 0)。如果 x ′t 与原始观测值 xt 显着不同，则有 |x ′t - xt| > τ 其中 τ 是预定义的阈值，此预测被接受 xt = x 't，a.k.a. 修复。背后的直觉是，距离越远表明成为异常值的概率越高。阈值 τ 可以通过使用预测区间 [14, 12] 观察 x 't 和 xt 之间的距离的统计分布来确定。因此，基于 AR 的修复过程是：（1）如果 xt 被标记，则用 yt 替换 xt，（2）从 x 学习 AR(p) 的参数 φ，以及 (3) 用 AR(p) 在 x 上填充所有未标记的 yt ，如果 yt 未标记，则 yt = ( x ′t 并且 |x ′t − xt| > τ xt 否则 (2)

示例 3

（示例 2 继续）。再次考虑 x = {6, 10, 9.6, 8.3, 7.7, 5.4,​​ 5.6, 5.9, 6.3, 6.8, 7.5, 8.5} 在图 2 中。为简单起见，我们使用阶数 p = 1 且 c = 0 的 AR(1)，即 x ′t = φ1xt−1。通过普通的最小square [20]，我们从 x 估计参数 φ，φ1 = 1.022。令 τ = 0.1。y1 被标记为真值。参考等式 2，它输出不变的 y1 = 6。对于 y2 和 y3 类似。我们有 x ′4 = φ1x1 = 1.022 ∗ 5.4 = 5.52。由于 |5.52 − 8.3| = 2.78 > 0.1，x ′4 被接受为新的 x4。类似地，我们有 x ′5 = φ1x4 = 1.022 ∗ 5.52 = 5.64。参考 | 5.64 − 7.7| = 2.06 > 0.1，预测被接受，以此类推，得到最终修复结果 y = {6, 5.6, 5.4, 5.52, 5.64, 5.4, 5.6, 5.72, 5.84, 5.97, 6.10 , 8.5}。其 RMS 误差为 0.51（参见第 6 节了解 RMS 定义）。

2.3 ARX 模型

为了利用标记的 y，我们考虑 ARX 模型（具有外生输入的自回归模型）[19] y ′t = xt + pX i=1 φi(yt−i − xt−i) + ǫt (3 ) 其中 y ′t 为 xt 的可能修复，其他与上述 AR 模型相同。如等式 3 所示，不仅前面的观测 xt-i 会影响 y't 的确定，还会影响之前标记/修复的 yt-i。因此，基于 ARX 的修复过程是： (1) 从 x 和部分标记的 y 中学习 ARX(p) 的参数 φ，以及(2) 用 ARX(p) 填充所有未标记的 yt，类似于公式 2，将 x 't 替换为 y 't。

示例 4

（示例 2 继续）。再次考虑 x = {6, 10, 9.6, 8.3, 7.7, 5.4, 5.6, 5.9, 6.3, 6.8, 7.5, 8.5} 和部分标记的 y 超过五个点，如图 2 所示。为简单起见，我们还使用 ARX( 1) 阶数 p = 1，即 y ′t = xt + φ1(yt-1 - xt-1)。与示例 3 中的 AR 类似，我们通过普通最小二乘法 [20] 估计参数 φ1 = 0.5。令 τ = 0.1。同样，标记的 y3 = 5.4 没有被修改。对于第四点，我们有 y ′4 = 8.3 + 0.5 ∗ (5.4 − 9.6) = 6.2。从 |6.2 − 8.3| = 2.1 > 0.1，我们指定 y4 = y ′4 = 6.2。最后，ARX 的修复结果为 y = {6, 5.6, 5.4, 6.20, 6.65, 5.4, 5.6, 5.9, 6.3, 6.8, 7.5, 8.5}，RMS 误差为 0.49，低于例 3 中的 AR。考虑 ARX 模型，因为它可以捕获观察到的错误和标记的事实之间的差异，而 AR 和 SCREEN 等其他方法忽略（不能利用）这种差异。通过对错误和真相之间的这种差异进行建模，而不是对原始值进行建模，ARX 和我们提出的 IMR 可能无法像 SCREEN [22] 那样处理尖峰错误（即，图 9 中的 # 个连续错误 = 1）。然而，通过迭代修复，IMR 在解决大量连续错误时总是显示出明显更好的结果（也参见图 9）。 ARX 也无法很好地解决此类连续错误，因为它仅适用于与引言中第 1.2 节中讨论的数据修复中的最小更改原则相矛盾的重大更改。