分　数：	
评卷人：	

華中科技大學

研 究 生（　　）课 程 论 文（报 告）

题 目：**Improving Service Availability of Cloud Systems by Predicting Disk Error**

学　　号 M201873223

姓　　名 张游

专　　业 计算机技术

课程指导教师 曾令仿，施展

院（系、所） 计算机科学与计算机学院

2018 年　　11 月　　29 日

Improving Service Availability of Cloud Systems by Predicting Disk Error
论文背景综述

1.背景

这篇论文由 Yong Xu, Kaixin Sui, Randolph Yao, Hongyu Zhang, Qingwei Lin, Yingnong Dang, Peng Li, Keceng Jiang, Wenchi Zhang, Jian-Guang Lou,Murali Chintalapati, Dongmei Zhang 发表在 USENIX ATC 会议上。USENIX ATC 是计算机系统领域最悠久的会议之一，也是中国计算机学会推荐的 A 类系统会议，计算机系统领域中 Oak 语言(Java 的前身)、TCL、QEMU、ZooKeeper 等一系列有影响力的研究成果都是在 USENIX ATC 发表或公布的。

服务高可用性对云系统至关重要。一个典型的云系统使用了大量的物理硬盘驱动器。磁盘错误是导致服务不可用的最重要原因之一。磁盘的错误（例如扇区错误和延迟错误）可以被作为灰色故障的一种，其是难以检测的故障，即使在影响应用程序的使用时障时。

在本文中，我们建议在它们对云系统造成更严重的损害之前主动预测磁盘错误。预测故障磁盘的能力可以提前启用现有虚拟机的实时迁移和将新的虚拟机分配到健康的磁盘中，从而提高服务可用性。为了建立准确的在线预测模型，我们利用磁盘级传感器（SMART）数据以及系统级信号。我们开发了一个对成本敏感的基于排列的机器学习模型，该模型可以学习过去故障磁盘的特性，并根据磁盘在不久的将来出现的错误可能性对磁盘进行排名。我们使用从生产云系统收集的真实数据来评估我们的方法。结果证实，所提出的方法是有效的，并且优于相关方法。此外，我们已成功应用所提议的方法来提高 Microsoft Azure 的服务可用性。

2.关于磁盘预测

近年来，软件应用程序越来越多地作为在线服务部署在云计算平台上，例如 Microsoft Azure，Google Cloud 和亚马逊 AWS。由于全球数百万用户可以全天候使用云服务，因此高可用性对基于云的服务至关重要。虽然许多云服务提供商都瞄准了高服务

可用性（例如 99.999％），实际上，服务仍然可能失败并导致用户不满和收入损失。例如，根据一项研究

在美国的 63 个数据中心组织进行的，停机的平均成本已经稳步增加 2010 年为 505,502 美元，2016 年为 740,357 美元（或净变化为 38％）[1]。

云系统中可能出现各种软件，硬件或网络相关问题。我们和 Microsoft Azure 合作的经验显示磁盘问题是硬件问题中最严重的问题。像 Azure 这样的典型云系统使用了数亿个硬盘驱动器。与磁盘相关的问题已成为导致服务停机的最重要因素之一。据报道，Facebook 和谷歌的研究人员也观察到磁盘问题的重要性，他们指出 20-57％的磁盘在 4 到 6 年内收集的数据集中至少遇到一个扇区错误[2,3]。

为了提高服务可用性，许多主动磁盘预测方法已经被提出来[4,5,6,7,8]。这些方法从历史磁盘故障数据中训练预测模型，并使用训练过的模型来预测磁盘是否会在不久的将来发生故障（即磁盘是否可以运行）。然后可以采取一些主动操作，例如更换容易出故障的磁盘。预测模型主要使用 SMART [9]数据构建，SMART [9]数据是嵌入在磁盘驱动器中的固件提供的磁盘级传感器数据。

现有方法集中于预测完整的磁盘故障（即磁盘操作/不可操作）。但是，在云环境中，在完全磁盘故障之前，上层服务可能已经受到影响磁盘错误（例如延迟错误，超时错误和扇区错误）。症状包括文件操作错误，VM 无法响应通信请求等。磁盘错误可以看作是灰色失败的一种形式[10]，这是一个相当微妙的失败，可以快速通过传统的系统故障检测器进行确定的检测，即使应用程序受到它们的影响。 Gunawi 等还指出了仍然可以运行但处于降级模式的故障慢速硬件的影响[11]。

如果不采取任何行动，可能会出现更严重的问题甚至服务中断。因此，我们提倡预测磁盘错误非常重要，以便在对服务系统造成更严重的损害之前采取主动措施。积极的措施包括错误感知的 VM 分配（将 VM 分配给更健康的磁盘），实时 VM 迁移（将 VM 从故障磁盘移动到健康磁盘）等。这样，服务可用性可以通过预测磁盘错误来改进。在本文中，我们开发了一种用于预测磁盘错误的在线预测算法，旨在提高云服务系统的服务可用性。我们发现了磁盘错误通常可以通过系统级信号（如 OS 事件）反映出来。我们的方法称为 CDEF（代表云盘错误预测）SMART 数据和系统级信号。它利用机器学习算法来训练预测模型使用历史数据，然后使用构建的模型来预测故障磁盘。我们设计预测模型具有以下能力：

- 能够根据容易出错的程度对所有磁盘进行排名，以便服务系统可以将 VM 分配给更健康的磁盘。
- 能够在成本和容量的约束下识别一组故障磁盘，指出虚拟机应该从中迁移出来。

3.磁盘预测的挑战

3.1：极其不平衡的数据分布

在现实世界的云服务系统中，极不平衡的数据使预测变得更加困难。 平均而言，每天只有大约 1,000,000 个磁盘中的 300 个可能出现故障。 我们需要识别故障磁盘，并注意不要将健康磁盘预测为故障。 在我们的工作中，我们提出了一个成本敏感的排名模型来应对这一挑战。 我们根据磁盘的错误率对磁盘进行排名，并通过最小化总成本来识别故障磁盘。 使用成本敏感的排名模型，我们只关注识别最重要的排名容易出错的磁盘，而不是分类所有故障磁盘。通过这种方式，我们可以缓解极端不平衡问题。

3.2：无用的特征

某些功能，尤其是系统级信号，是时间敏感（它们的值随着时间的推移而不断变化）或对环境敏感（由于云环境不断变化，它们的数据分布会发生显着变化）。 我们发现使用这些不稳定特征构建的模型可以在交叉验证中产生良好的结果（将数据随机分成训练和测试集），但在实际在线预测中表现不佳（按时间将数据划分为训练和测试集）。为了应对这一挑战，我们进行了系统的特征工程，并提出了一种新的特征选择方法，用于选择稳定和预测的特征。

4.解决方法

4.1: 特征识别

我们收集两类数据，SMART 数据和系统级信号。 SMART（自我监控，分析和报告技术）是一种监控固件，允许磁盘驱动器报告有关其内部活动的数据。在云系统中，还有各种系统级定期收集的事件（通常每小时一次）。许多这些系统级事件（例如 Windows 事件，文件系统操作错误，意外的遥测丢失等）都是磁盘错误的早期信号，例如，FileSystemError 是由磁盘相关错误引起的事件，可以追溯到坏扇区或磁盘完整性损坏。

除了从原始数据中直接识别的功能外，我们还计算了一些统计特征，如下所示：

Diff 通过数据分析，我们发现了随着时间的推移，特征值的变化可能对区分磁盘错误是有用的。

Di f f 给出时间窗口 w，我们在时间戳 t 定义特征 x 的 Di f f，如下所示：

$$Diff(x,t,w) = x(t) - x(t-w)$$

Sigma Sigma 计算一段时间内属性值的方差。 鉴于时间窗 w，Sigma 时间戳 t 处的属性 x 定义为：

$$Sigma(x,t,w) = E[(X-\mu)^2],$$

其中$X = (x_{t-w}, x_{t-w-1}, \ldots, x_t)$ and $\mu = \frac{\Sigma(X)}{w}$.

Bin Bin 计算在窗口 w 中的属性值的总和：

$$Bin(x,t,w) = \sum_{j=t-w+1}^{t} x(j)$$

在本论文中，用了 3,5,7 三种不同的窗口大小

4.2: 特征选择

为了选择稳定和预测特征，我们执行特征选择以修剪在预测中表现不佳的特征。 这个想法是模拟训练集的在线预测。 训练集按时间划分为两部分，一部分用于训练，另一部分用于验证。 如果删除一个功能后验证集上的性能变得更好，则会删除该功能，直到剩余功能的数量减少未超过属性总数的 q%。 算法 1 中描述了详细信息。在我们的实验中，我们默认设置 q = 10%，这意味着如果剩余特征的数量小于 10%，则修剪过程将停止。最后，我们重新调整所有选定属性的范围使用零均值归一化如下：

$$X_{zero-mean} = x - mean(X)$$

Algorithm 1: Prune non-predictive features

Input : Training data TR with feature set F $(f_1, f_2, , , , f_m)$

Output: Reduced feature set F'

```
Split TR by time equivalently into TR1 and TR2
foreach f_i in F do
    // use TR1 to predict TR2, get accuracy result
    r ← train(TR1) and test(TR2)
    // remove data about f_i from TR, then predict
    r_{f_i} ← train(TR1-f_i) and test(TR2-f_i)
    if r_{f_i} > r then
        delete f_i from F
    end
    if number of remaining features <= θ * m
     then
        Break
    end
end
Return F'
```

4.3: Cost-sensitive ranking model

从历史数据中收集了特征，然后我们构建一个预测模型来预测未来几天磁盘的错误率。

在此步骤中，我们将预测问题表示为排名问题而不是分类问题。也就是说，我们不是简单地告诉磁盘是否有故障，而是根据磁盘的容易出错性对磁盘进行排名。排名方法减轻了极端不平衡故障数据的问题，因为它对类不平衡不敏感。为了训练排序模型，我们获得关于磁盘的历史故障数据，并根据磁盘的相对失效时间对磁盘进行排序（即，收集数据和检测到第一个错误之间的天数）。

我们采用 Learning to Rank [12]的概念，它自动从大量数据中学习优化的排名模型，以最大限度地减少损失函数。我们采用了 FastTree 算法[13,14]，它是“多重加法回归树”

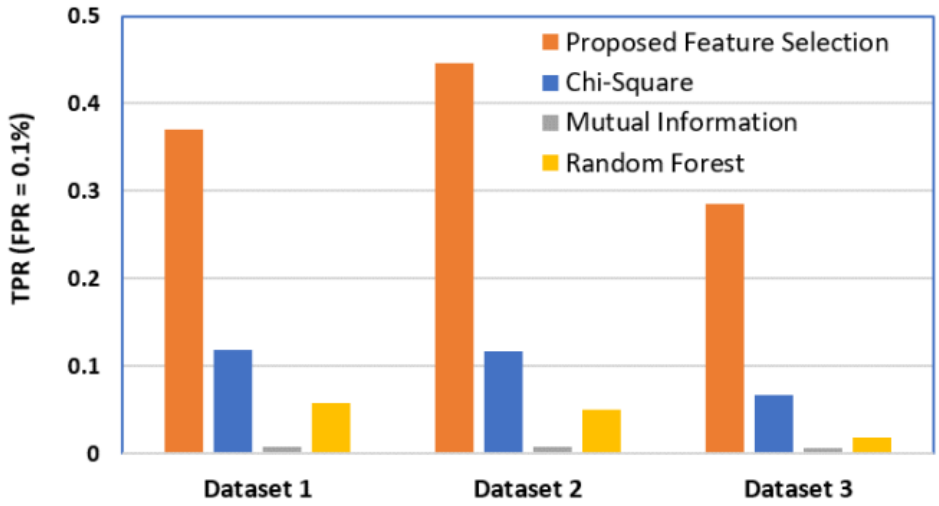
（MART）梯度增强算法的一种形式。它以逐步的方式构建每个回归树（在其假期中具有标量值的决策树）。该算法广泛应用于机器学习和信息检索研究。为了提高服务可用性，我们希望智能地将 VM 分配给更健康的磁盘以便这些 VM 在不久的将来不太可能遭受磁盘错误。为此，我们根据故障的可能性来识别故障和健康的磁盘。由于大多数磁盘都是健康的，只有一小部分磁盘出现故障，我们选择排名模型返回的前 r 个结果为错误。选择最佳的顶级磁盘，以便最大限度地减少错误分类总成本：

$$cost = Cost1 * FPr + Cost2 * FNr;$$

其中 FPr 和 FNr 分别是前 r 个预测结果中的假阳性和假阴性的数量。Cost1 是错误地将健康磁盘识别为故障的成本，这涉及从“故障”磁盘到健康磁盘的不必要的实时迁移的成本。尽管我们拥有非常好的实时迁移技术，但迁移过程仍然会产生不可忽视的成本并降低云系统的容量。 Cost2 是无法识别故障磁盘的成本。 Cost1 和 Cost2 的值由产品团队的专家根据经验确定。在我们目前的实践中，由于担心 VM 迁移成本和云容量，Cost1 远远高于 Cost2（即，我们重视精确度而不是召回）。领域专家根据他们在磁盘错误恢复方面的经验将 Cost1 和 Cost2 之间的比率设置为 3：1。通过从历史数据获得的假阳性和假阴性比率估计假阳性和假阴性的数量。通过最小化总误分类成本来确定最佳 r 值。前 r 个磁盘是预测的故障磁盘，它们是高风险磁盘，并且应该迁移托管在它们上的 VM。

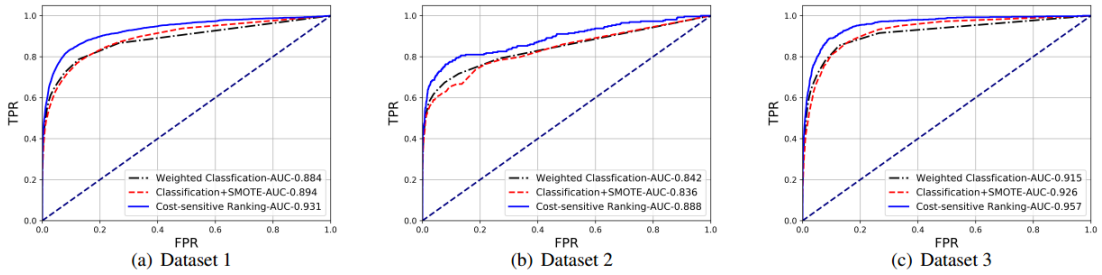
5.结果

5.1：提出的属性选择方法和其他方法对比



从上面结果可以看出，我们提出的特征选择算法在三个数据集中均取得了最好的效果。

5.2：提出的代价敏感排列模型和其他机器学习模型对比



(a) Dataset 1 (b) Dataset 2 (c) Dataset 3

Figure 6: ROC of cost-sensitive ranking and classification

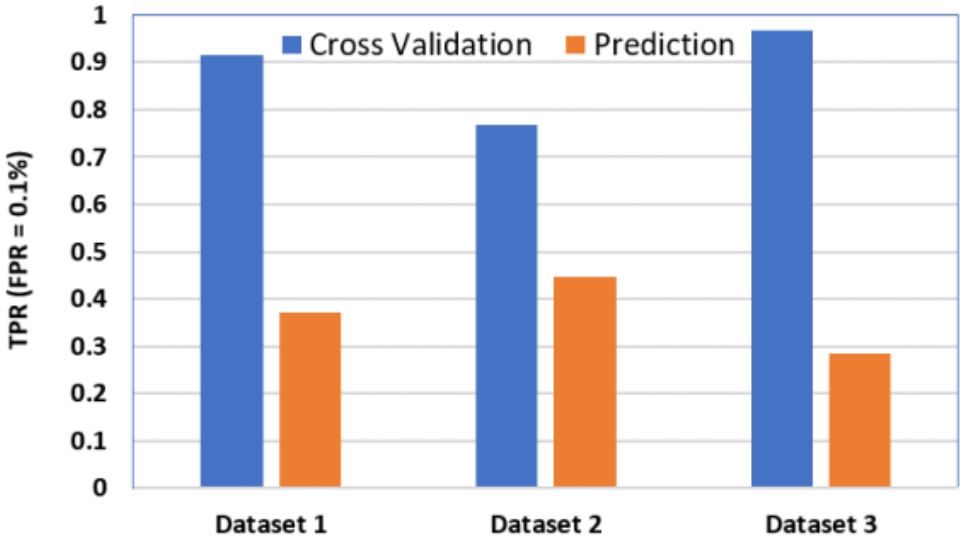


Figure 7: Evaluation results - cross validation vs. online prediction

从上面的结果可以看出，我们提出的基于代价敏感的排列模型比其他模型效果都要好。

参考文献

[1] PONEMONINSTITUTE. Cost of data center outages, 2016.
https://planetaklimata.com.ua/instr/Liebert
Hiross/Cost_of_Data_Center_Outages_2016_Eng.pdf.
[2] MEZA, J., WU, Q., KUMAR, S., AND MUTLU, O. A large-scale study of flash memory failures in the field. In Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems (New York, NY, USA, 2015), SIGMETRICS '15, ACM, pp. 177–190.
[3] SCHROEDER, B., LAGISETTY, R., AND MERCHANT, A. Flash reliability in production: The expected and the unexpected. In 14th USENIX Conference on File and Storage Technologies (FAST 16) (Santa Clara, CA, 2016), USENIX Association,pp. 67–80.
[4] GOLDSZMIDT, M. Finding soon-to-fail disks in a haystack. In Proceedings of the 4th USENIX Conference on Hot Topics in Storage and File Systems (Berkeley, CA, USA, 2012), HotStorage'12, USENIX Association, pp. 8–8.
[5] PINHEIRO, E., WEBER, W.-D., AND BARROSO, L. A. Failure trends in a large disk drive population. In Proceedings of the 5th USENIX Conference on File and Storage Technologies (Berkeley, CA, USA, 2007), FAST '07, USENIX Association, pp. 2–2.
[6] PITAKRAT, T., VAN HOORN, A., AND GRUNSKE, L. A comparison of machine learning algorithms for proactive hard disk drive failure detection. In Proceedings of the 4th International ACM Sigsoft Symposium on Architecting Critical Systems (New York, NY, USA, 2013), ISARCS '13, ACM, pp. 1–10.
[7] ZHU, B., WANG, G., LIU, X., HU, D., LIN, S., AND MA, J. Proactive drive failure prediction for large scale storage systems. In 2013 IEEE 29th Symposium on Mass Storage Systems and Technologies (MSST) (May 2013), pp. 1–5.
[8] WANG, Y., MIAO, Q., MA, E. W. M., TSUI, K. L., AND PECHT, M. G. Online anomaly detection for hard disk drives based on mahalanobis distance. IEEE Transactions on Reliability 62, 1 (March 2013), 136–145.
[9] http://www.distributed-generation.com.
[10] HUANG, P., GUO, C., ZHOU, L., LORCH, J. R., DANG, Y.,CHINTALAPATI, M., AND YAO, R. Gray failure: The achilles' heel of cloud-scale systems. In Proceedings of the 16th Workshop
on Hot Topics in Operating Systems (New York, NY, USA, 2017), HotOS '17, ACM, pp. 150–155.

[11] GUNAWI, H. S., SUMINTO, R. O., SEARS, R., GOLLIHER, C., SUNDARARAMAN, S., LIN, X., EMAMI, T., SHENG, W., BIDOKHTI, N., MCCAFFREY, C., ET AL. Fail-slow at scale:Evidence of hardware performance faults in large production systems.

[12] LIU, T.-Y. Learning to rank for information retrieval. Found.Trends Inf. Retr. 3, 3 (Mar. 2009), 225–331.

[13] MICROSOFT. Machine learning fast tree, https://docs.microsoft.com/en-us/machine-learningserver/python-reference/microsoftml/rx-fast-trees, 2017.

[14] FRIEDMAN, J. H. Greedy function approximation: A gradient boosting machine. Annals of Statistics 29 (2000), 1189–1232.

研　究　生　签　字 _____