基于 RUL 和 SVs-GFF 的云服务器老化预测方法

孟海宁 [1, 2] 童新宇 [1] 谢 国 [1] 张贝贝 [1] 黑新宏 [1]

摘 要 针对云服务器中存在软件老化现象, 将造成系统性能衰退与可靠性下降问题, 借鉴剩余使用寿命 (Remaining useful life, RUL) 概念, 提出基于支持向量和高斯函数拟合 (Support vectors and Gaussian function fitting, SVs-GFF) 的老化预测方法. 首先, 提取云服务器老化数据的统计特征指标, 并采用支持向量回归 (Support vector regression, SVR) 对统计特征指标进行数据稀疏化处理, 得到支持向量 (Support vectors, SVs) 序列数据; 然后, 建立基于密度聚类的高斯函数拟合 (Gaussian function fitting, GFF) 模型, 对不同核函数下的支持向量序列数据进行老化曲线拟合, 并采用 Fréchet 距离优化算法选取最优老化曲线; 最后, 基于最优老化曲线, 评估系统到达老化阈值前的 RUL, 以预测系统何时发生老化. 在 OpenStack 云服务器 4 个老化数据集上的实验结果表明, 基于 RUL 和 SVs-GFF 的云服务器老化预测方法与传统预测方法相比, 具有更高的预测精度和更快的收敛速度.

关键词 云服务器, 软件老化, 支持向量回归, 高斯函数拟合, 剩余使用寿命

引用格式 孟海宁, 童新宇, 谢国, 张贝贝, 黑新宏. 基于 RUL 和 SVs-GFF 的云服务器老化预测方法. 自动化学报, 2024, **50**(10): 2036−2048

DOI 10.16383/j.aas.c211112 **CSTR** 32138.14.j.aas.c211112

Cloud Server Aging Prediction Method Based on RUL and SVs-GFF

MENG Hai-Ning[1, 2] TONG Xin-Yu[1] XIE Guo[1] ZHANG Bei-Bei[1] HEI Xin-Hong[1]

Abstract Aiming at the problem that software aging in cloud servers will cause system performance degradation and reliability descending, a software aging prediction method based on support vectors (SVs) and Gaussian function fitting (SVs-GFF) with the use of the concept of remaining useful life (RUL) is proposed. Firstly, the statistical characteristic indexes of aging data on a cloud server are extracted, and then support vector regression (SVR) is used to sparse the data of statistical characteristic indexes into support vector sequences. Then, the Gaussian function fitting (GFF) model based on density clustering is established to fit the aging curves of support vector sequence data under different kernel functions, and the Fréchet distance optimization algorithm is used to select the optimal aging curve. Finally, based on the optimal aging curve, the remaining useful life before the system reaches the aging threshold is evaluated to predict when software aging occurs. The experiment results on four aging data sets of an OpenStack cloud server show that, the proposed cloud server aging prediction method based on remaining useful life and SVs-GFF has higher accuracy and faster convergence speed compared with traditional prediction methods.

Key words Cloud server, software aging, support vector regression (SVR), Gaussian function fitting (GFF), remaining useful life (RUL)

Citation Meng Hai-Ning, Tong Xin-Yu, Xie Guo, Zhang Bei-Bei, Hei Xin-Hong. Cloud server aging prediction method based on RUL and SVs-GFF. *Acta Automatica Sinica*, 2024, **50**(10): 2036−2048

长时间运行的软件系统出现性能衰退、异常和错误增加甚至系统崩溃现象, 称为软件老化[1]. 软件老化是由于软件系统可分配资源匮乏、碎片化严重以及内部错误 (如内存泄漏和未正确释放的线程) 的长期积累造成的, 而这些错误在软件开发和测试阶段难以检测和消除[2]. 软件老化存在于安全关键软件系统中, 如航天器系统[3]、Android 系统[4] 和 Web 服务器[5] 等. 云服务器系统的体系结构复杂, 系统内计算资源的申请和释放较频繁, 且云系统长期运行为用户提供服务, 因此云系统更易出现软件老化现象, 影响系统可用性并带来经济损失[6].

如图 1 所示, 云服务器中存在的软件老化现象主要表现为: 在系统负载整体保持不变的情况下, 云服务器内部资源消耗逐渐增加, 系统性能逐步下

收稿日期 2021-11-24 录用日期 2022-04-28
Manuscript received November 24, 2021; accepted April 28, 2022
国家自然科学基金 (61602375, 61773313), 陕西省自然科学基础研究计划基金 (2019JQ-749) 资助
Supported by National Natural Science Foundation of China (61602375, 61773313) and Natural Science Basic Research Plan of Shaanxi Province (2019JQ-749)
本文责任编委 王雪松
Recommended by Associate Editor WANG Xue-Song
1. 西安理工大学计算机科学与工程学院 西安 710048 2. 陕西省网络计算与安全技术重点实验室 西安 710048
1. School of Computer Science and Engineering, Xi′an University of Technology, Xi′an 710048 2. Shaanxi Key Laboratory Network Computer and Security Technology, Xi′an 710048

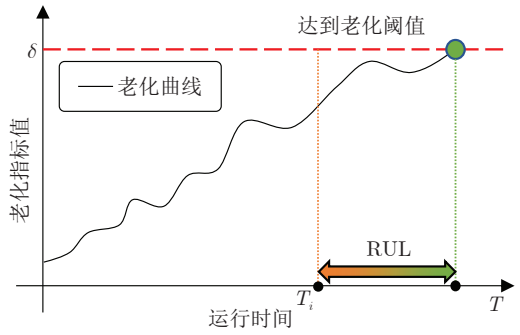


图 1　云服务器老化现象

Fig. 1　Software aging phenomenon in a cloud server

降. 当云服务器的资源消耗或性能衰退至系统不可接受状态, 即到达系统老化临界态, 设置老化阈值 δ. 当云服务器系统的内部资源消耗超过阈值 δ 时, 系统随时可能发生故障, 需提前预测系统老化趋势, 并评估系统从任意时刻 T_i 到达老化阈值前的剩余使用寿命 (Remaining useful life, RUL), 从而可以及时采取系统预维护策略[7].

为避免软件老化, Huang 等[8] 提出了软件再生. 软件再生是一种主动预防性的容错技术, 通过清理系统运行环境, 重置系统内部状态, 以防止系统未来发生重大故障. 然而, 频繁的软件再生操作将会带来额外的系统开销, 因此需选择最佳时机进行软件再生.

为减少软件再生给系统带来的时间和资源消耗成本, 需要在系统到达老化阈值或发生重大故障前, 对软件老化趋势进行有效预测, 然后及时进行软件再生, 以降低运行风险与成本, 提高系统的可靠性和可用性. 现有的软件老化预测方法可分为基于状态模型的方法和基于数据度量的方法 2 类. 基于状态模型方法通常依据软件系统的运行状态, 采用 Petri 网[9]、马尔科夫 (Markov) 模型[10−14]、随机过程[15] 和概率论[16] 等数学工具建立系统状态转换模型, 然后计算最优的软件再生时间. 然而, 对实际软件系统很难建立精确的状态模型, 并且随着软件系统规模的增加, 该方法构建的状态空间模型亦呈指数级增长, 将导致状态空间爆炸[17].

目前, 基于数据度量的老化预测方法首先选取系统性能历史数据作为老化指标 (如 CPU 使用率、空闲内存与交换区内存等), 然后设定系统老化阈值, 采取时间序列分析、机器学习等方法预测老化指标历史数据变化趋势, 在预测值到达老化阈值前采取软件再生. 时间序列分析法包括移动平均自回归模型 (Auto-regressive integrated moving average, ARIMA)、滑动模型等预测方法, 如 Yan 等[18] 提出通过整合移动平均自回归模型, 对 Web 服务器资源进行老化预测; Pereira 等[19] 提出指数平滑模型的软件老化预测方法. Cotroneo 等[20] 针对 Linux 系统, 提出基于线性回归和主成分分析的老化预测方法. Matos 等[21] 针对 Eucalyptus 云系统老化现象, 提出基于多阈值的时间序列预测方法, 该方法求解得到最优再生时机, 可减少停机时间. 郑鹏飞等[22] 采用多元时间序列模型, 预测 Helix-Server VOD 服务器系统老化. 但该类方法对数据量大且波动较大的数据, 难以提取数据特征, 因此预测精度较低. 机器学习方法包括神经网络、支持向量 (Support vectors, SVs) 机等, 如 Islam 等[23] 针对亚马逊 EC2 云系统, 采用神经网络和线性回归的方法, 预测系统资源的使用情况. Yan 等[24] 采用支持向量机对 IIS 服务器系统进行老化预测. Qiao 等[25] 针对安卓系统, 提出基于长短期记忆 (Long short-term memory, LSTM) 网络的老化预测方法. 然而该类算法在预测时间序列数据时, 需要经验给定模型中的超参数且计算复杂度高. 一些学者提出运用智能算法优化神经网络超参数来提高预测精度, 如 Padhy 等[26] 在研究软件老化时, 采用进化算法优化系统性能参数阈值. Yan[27] 基于萤火虫种群算法优化的神经网络模型进行老化预测. 但智能优化算法在网络训练过程中易陷入局部最优, 且收敛速度慢. 此外, Liu 等[28] 提出 ARIMA 与 LSTM 组合模型, 实现对 Web 服务器老化预测. 孟海宁等[29] 提出 ARIMA 和循环神经网络的组合模型, 对 OpenStack 云服务器进行老化预测, 解决了单一老化预测模型存在泛化能力弱、预测精度低的问题.

综上, 上述老化预测方法存在数据特征提取难、模型超参数选择难、计算复杂度高及收敛速度慢的问题. 本文面向云服务器系统, 提出基于支持向量和高斯函数拟合 (Support vectors and Gaussian function fitting, SVs-GFF) 的老化预测方法, 预测系统老化趋势, 评估系统的 RUL, 以方便系统预运维人员在系统可用状态下计划再生时间, 从而优化系统运行效率, 并避免计划外重大故障的发生.

本文主要工作包括以下 4 点:

1) 通过特征筛选与融合算法提取原始老化数据的统计特征指标, 捕获云服务器老化数据时间序列变化规律;

2) 基于支持向量回归 (Support vector regression, SVR) 方法的稀疏性, 对云服务器老化特征指标进行数据拟合, 得到稀疏化后的 SVs 序列数据作为预测输入数据, 减少预测模型的计算复杂度;

3) 考虑到稀疏化后 SVs 序列数据存在空间分布不均衡问题, 先对 SVs 序列数据进行基于密度的聚类, 依据聚类结果对 SVs 赋予不同权重信息, 再进

行老化预测, 避免预测过程陷入局部最优, 提高预测收敛速度;

4) 采用基于密度聚类的 GFF 方法拟合基于不同核函数的 SVs 序列, 得到对应多条老化曲线, 基于 Fréchet 距离优化算法选取最优老化曲线, 无需选择模型超参数, 避免人工干预.

1 支持向量回归

支持向量回归是一种基于支持向量机的回归预测方法[30]. 设数据集 $D=\{x_i, y_i\}_{i=1}^{N}$, $x_i \in \mathbf{R}^m$, $y_i \in \mathbf{R}$, 其中 m 为输入维度, N 为样本个数. 本文所研究的云服务器老化时间序列数据集, 对应地, x_i 为样本序号, y_i 表示老化数据值, 且 $m=1$. 采用 SVR 对数据集 D 中序列数据进行回归预测, 对应输入 x_i 的预测值 $f(x_i)$ 可表示为:

$$f(x_i)=\langle \boldsymbol{W}, x_i\rangle + b \tag{1}$$

式中, $\langle\cdot,\cdot\rangle$ 为内积函数, $\boldsymbol{W}$ 和 b 为模型参数. $\boldsymbol{W}$ 值可通过求解以下目标函数得到:

$$\begin{aligned}&\min \frac{1}{2}\|\boldsymbol{W}\|^2 + C\sum_{i=1}^{N}(\xi_i+\hat{\xi}_i)\\ &\text{s.t.}\\ &y_i - f(x_i) \le \varepsilon' + \xi_i\\ &f(x_i) - y_i \le \varepsilon' + \hat{\xi}_i\\ &\xi_i, \hat{\xi}_i \ge 0\end{aligned} \tag{2}$$

式中, $\|\boldsymbol{W}\|^2=\langle \boldsymbol{W}, \boldsymbol{W}\rangle$, C 为正则化常数, ξ_i 和 $\hat{\xi}_i$ 分别为上、下边界松弛变量, ε' 为间隔边界, $i=1, 2, \cdots, N$.

2 高斯函数拟合

高斯函数拟合 (Gaussian function fitting, GFF) 是对序列数据进行函数逼近的非线性拟合方法, 其中高斯函数参数物理意义明确、计算复杂度低[31]. 对于数据集 $D=\{x_i, y_i\}_{i=1}^{N}$, 输入 x_i 和预测值 $f(x_i)$ 间的关系为:

$$f(x_i)=\alpha \times \exp\left\{-\frac{(x_i-\beta)^2}{2\sigma^2}\right\}+\gamma \tag{3}$$

式中, α、β、σ 和 γ 分别为高斯函数的峰值、峰值位置、半宽度信息和偏置.

对云服务器老化曲线进行拟合即需求最优的参数 α^*、β^*、σ^* 和 γ^*, 使函数 $f(x)$ 与老化曲线匹配.

3 SVs-GFF 老化预测方法

本文提出的基于 SVs-GFF 的云服务器老化预测方法框图见图 2. 首先, 提取云服务器老化数据的统计特征指标, 并对特征指标进行筛选并融合; 其次, 采用具有不同核函数的 SVR 对老化特征指标进行数据稀疏化, 获得多组支持向量 SVs 序列; 然后, 采用密度聚类的 GFF 方法对 SVs 进行拟合, 得到不同核函数下云服务器的老化曲线, 并基于 Fréchet 距离优化算法选取最优老化曲线; 最后, 通过老化阈值评估云服务器的 RUL, 以判定云服务器何时老化.

3.1 数据特征处理

本文以云服务器系统性能参数时间序列作为原始老化数据集 D, 为获取云服务器老化数据的变化

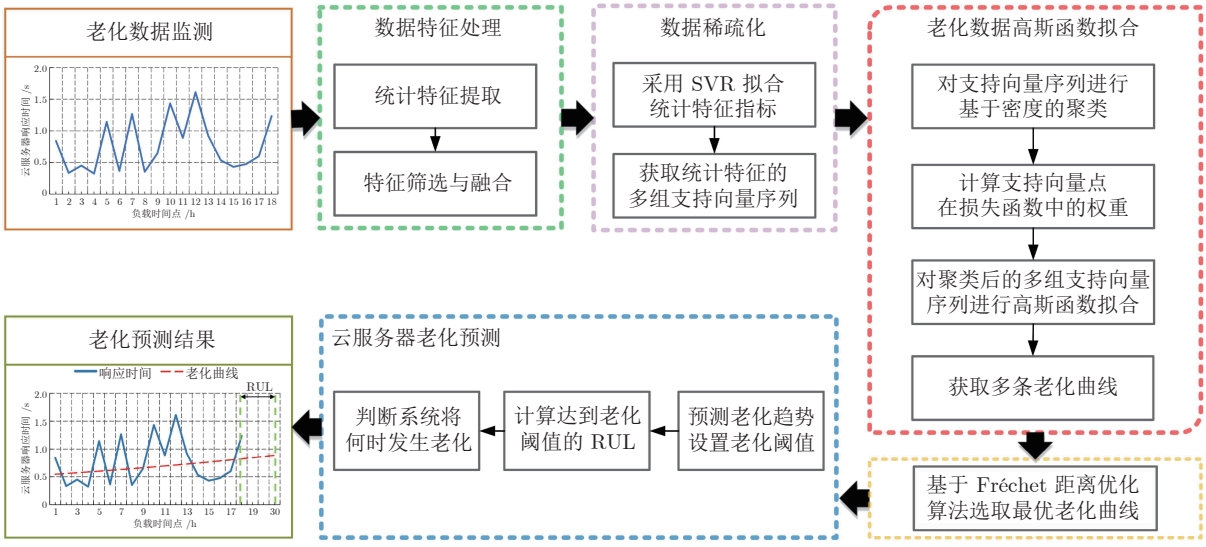


图 2 基于 SVs-GFF 的云服务器老化预测方法框图

Fig. 2 Block diagram of cloud server aging prediction method based on SVs-GFF

特征, 首先提取 D 中样本的统计特征指标 (包括均方根、峰度、偏斜度和峰值因子等)[32], 记为 $\{\boldsymbol{f}^i\}_{i=1}^M$, 其中 $\boldsymbol{f}^i=\{f_1^i, f_2^i, \cdots, f_n^i\}$ 且 $n=N/l$, M 为统计特征指标类别数, N 为 D 中样本个数, l 为提取统计特征指标时滑动窗口长度.

根据文献 [33] 相关研究, 当软件系统出现老化时, 系统性能参数数据序列呈显著单调性趋势 (例如可用内存具有单调下降趋势). 本文首先基于单调性算法, 筛选与云服务器系统老化关联关系强 (即单调性强) 的统计特征指标, 剔除单调性较弱的统计特征指标; 然后, 对筛选后的指标进行融合. 具体过程见算法 1.

算法 1. 基于统计特征指标筛选与融合算法

输入. 统计特征指标数据集 $\{\boldsymbol{f}^i\}$.

输出. 筛选与融合后的统计特征指标集 $\overline{f}$.

1) 给出云服务器统计特征指标数据集 $\{\boldsymbol{f}^i\}_{i=1}^M$ 的差分序列数据集 $\{diff^i\}_{i=1}^M$;

2) 计算云服务器统计特征指标 $\boldsymbol{f}^i$ 的单调度: $\Delta_i=|NumL^i-NumS^i|\times N/l$, 其中 $NumL^i$ 为差分序列 $diff^i$ 中大于零元素的个数, $NumS^i$ 为 $diff^i$ 中小于零元素的个数; 并计算单调度阈值: $\lambda=\sum_{i=1}^M\Delta_i/M+\sigma'$, 其中 σ' 为单调度集合 $\{\Delta_1, \Delta_2, \cdots, \Delta_M\}$的标准差;

3) 筛选统计特征指标, 若统计特征指标 $\boldsymbol{f}^i$ 的单调度 $\Delta_i<\lambda$, 则删除统计特征指标 $\boldsymbol{f}^i$, 剩余的统计特征指标为筛选后的统计特征指标集 $\{\boldsymbol{f}^i\}_{i=1}^L$, 其中 $L\le M$;

4) 融合统计特征指标, 先进行去数据中心化处理: $\boldsymbol{f}^i=\boldsymbol{f}^i-\sum_{j=1}^n f_j^i/n$, 其中 n 为 $\boldsymbol{f}^i$ 的长度;

5) 计算统计特征指标集 $\{\boldsymbol{f}^i\}_{i=1}^L$ 的协方差矩阵 $\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}/L$, 并进行特征值分解, 其中 $\boldsymbol{X}=\{\boldsymbol{f}^i\}_{i=1}^L$;

6) 选取协方差矩阵中特征值最大的特征向量所对应的单位向量 $\boldsymbol{\omega}$;

7) 输出融合后的统计特征指标集: $\overline{\boldsymbol{f}}=\boldsymbol{\omega}\times\boldsymbol{X}$.

3.2　基于 SVR 的数据稀疏化处理

对于第 3.1 节筛选并融合后的统计特征指标序列 $\overline{f}$, 采用 SVR 进行拟合, 确定统计特征指标各数据点的重要性, 保留满足重要性要求的数据点, 得到支持向量 SVs 序列, 即统计特征指标数据的稀疏化处理结果.

具体方法是, 对 SVR 的目标函数式 (2) 优化时进行判断, 若特征指标 x_i 满足 $y_i-f(x_i)-\xi_i\ge\varepsilon'$ 或 $f(x_i)-y_i-\hat{\xi}_i\ge\varepsilon'$, 则 x_i 为支持向量. 将得到的支持向量序列记作 $\mathrm{SVS}=\{(x_i, y_i)\}_{i=1}^h$, 其中 $h\ll n$. 显然, SVS 为统计特征指标数据集的子集, 且通过 SVR 对统计特征指标数据进行稀疏化处理, 得到的 SVs 为保留老化趋势特征的统计特征指标, 可表示老化趋势的主要变化特征. 因此, 本文后续通过支持向量序列预测数据变化趋势, 根据数据变化规律评估系统状态跃迁老化阈值的时间间隔.

3.3　基于密度聚类的 GFF 老化曲线拟合

第 3.2 节稀疏化后的统计特征指标数据序列即支持向量 SVs 序列, 其样本点在空间分布不均衡, 以云服务器系统响应时间数据序列为例, 如图 3 所示, 阴影区域的支持向量分布较为密集, 带黑色箭头支持向量位置较远, 可看作离群点.

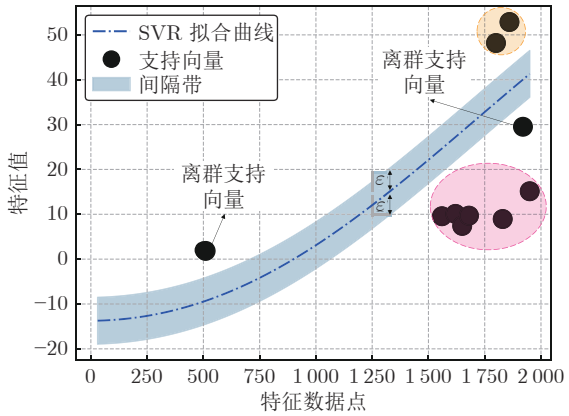


图 3　支持向量的空间分布

Fig. 3　Spatial distribution of support vectors

鉴于 SVs 序列样本空间中存在离群支持向量点, 若直接对 SVs 序列进行预测, 则算法可能会陷入局部最优. 考虑到 K-均值等聚类算法对离群点噪声敏感[34], 本文首先用 DBSCAN (Density-based spatial clustering of applications with noise) 算法 (一种基于密度的空间聚类算法) 对支持向量进行聚类. 根据 SVs 序列数据的分布密度情况, 对不同类中支持向量赋予相应权重, 支持向量越密集, 赋予权值越大; 反之, 支持向量越稀疏, 赋予权值越小. 其中离群支持向量赋予最小权值.

然后采用 GFF 方法对聚类后 SVs 序列进行拟合, 其误差函数使用 Huber 函数如下:

$$J_\delta(y_i, f(x_i))=\begin{cases}\dfrac{1}{2}p_i(a_i)^2, & |a_i|\le v\\ p_i v|a_i|-\dfrac{1}{2}v^2, & |a_i|>v\end{cases}\tag{4}$$

式中, y_i 和 $f(x_i)$ 分别为支持向量的真实值和预测值, a_i 为预测误差值且 $a_i=y_i-f(x_i)$, v 为误差阈值. p_i 为 SVs 权重, 依据聚类结果, p_i 计算方法为:

$$p_i=\frac{|C||C_i|}{\sum\limits_{j=1}^{|C|}|C_j|^2}\tag{5}$$

式中, $|C|$ 为类别数, $|C_i|$ 为第 i 类中样本的个数.

基于密度聚类的 GFF 算法进行老化曲线拟合的具体过程见算法 2.

算法 2. 基于密度聚类的 GFF 老化曲线拟合算法

输入. 支持向量序列 SVS $= \{(x_i, y_i)\}_{i=1}^{h}$.

输出. 老化曲线 $\hat{s}$.

1) 从支持向量序列 SVS 中任意选取一个支持向量样本 f_i.

2) 统计支持向量样本 f_i 的 ε- 邻域 $N_\varepsilon(f_i) = \{f_j \in D | d(f_i, f_j) \le \varepsilon\}$, $d(\cdot, \cdot)$ 为欧氏距离函数, ε 为距离阈值.

3) 判断样本 f_i 的 ε- 邻域包含的支持向量个数是否满足 $|N_\varepsilon(f_i)| \ge MinPts$. 若满足条件, 则 f_i 为核心样本. 其中 $MinPts$ 为样本 f_i 的 ε- 邻域最小样本数.

4) 建立以 f_i 为核心样本的类 C_i.

5) 遍历 $N_\varepsilon(f_i)$ 中的支持向量, 若 f_j 属于核心样本, 则令 $C_i = C_i \cup N_\varepsilon(f_j)$.

6) 对所有的支持向量, 重复步骤 1) ~ 5), 得到聚类结果 $C = \{C_1, C_2, \cdots, C_k\}$.

7) 基于 C 和式 (5), 计算支持向量 f_i 的误差权重 p_i.

8) 基于步骤 7) 和式 (4), 计算误差 J 函数值.

9) 得到式 (3) 中最优参数 α^*、β^*、σ^* 和 γ^*, 即得到基于密度聚类的 GFF 算法拟合的老化曲线.

3.4 基于 Fréchet 距离优化算法的老化曲线选取

基于不同核函数对统计特征指标序列进行数据稀疏化, 采用第 3.3 节基于密度聚类的 GFF 方法对稀疏化后数据进行拟合, 得到多条老化曲线, 本文基于 Fréchet 距离优化算法选取其中最优老化曲线.

Fréchet 距离优化算法是一种路径空间相似性描述方法, 相较欧氏距离, 其考虑曲线点的位置和方向, 可度量曲线间相似度[35]. 计算任意两条数据曲线 S 和 $\hat{S}$ 的 Fréchet 距离如下:

$$\delta(S, \hat{S}) = \inf_{\zeta, \psi} \max_{x \in [0, 1]} \{d(S(\zeta(x)), \hat{S}(\Psi(x)))\} \tag{6}$$

式中, $\zeta(\cdot)$ 和 $\Psi(\cdot)$ 为非减实数函数, 且 $\zeta(0) = \Psi(0) = 0$, $\zeta(1) = \Psi(1) = 1$.

若两条数据曲线的 Fréchet 距离越小, 则表明曲线越相似. 因此, 本文选取 Fréchet 距离最小的老化曲线作为最优老化曲线, 用于预测云服务器的老化趋势.

3.5 云服务器老化预测方法及流程

设置云服务器老化阈值为 δ, 当系统性能指标数据预测值超过阈值 δ 时, 则云服务器进入老化状态. 基于第 3.4 节得到的最优老化曲线, 计算云服务器到达老化阈值 δ 前的 RUL, 即得到从当前时刻 T_k 到云服务器发生老化的时间, 从而决定何时进行软件再生.

云服务器在时刻 T_k 的 RUL 的计算方法为[36]:

$$L(T_k) = \inf\{r : f(r + T_k) \ge \delta\} \tag{7}$$

式中, $\inf\{\cdot\}$ 表示变量的下确界, 云服务器在时间点 T_k 的剩余寿命 RUL 为 r, 即在时间点 T_k 预估云服务器系统将在时间点 $r + T_k$ 时达到老化阈值 δ. $f(r + T_k)$ 是最优老化曲线上对应时间点 $r + T_k$ 的指标数据预测值.

基于 RUL 的计算公式, 算法 3 给出云服务器老化预测流程.

算法 3. 云服务器老化预测算法

输入. 云服务器原始老化数据集 D.

输出. 云服务器的 RUL.

1) 设置滑动窗口长度为 l, 计算老化数据集 D 的特征统计特征集 $\{\boldsymbol{f}^i\}_{i=1}^{M}$, 其中 M 为统计特征指标的数量;

2) 基于算法 1 对 $\{\boldsymbol{f}^i\}_{i=1}^{M}$ 进行特征筛选与融合, 得到统计特征指标集 $\overline{f}$;

3) 采用 p 种不同核函数 SVR 对统计特征指标集 $\overline{f}$ 进行数据稀疏化处理, 得到 p 组特征向量 SVS $= \{SVs_i\}_{i=1}^{p}$;

4) 通过算法 2 拟合 SVS, 得到 p 条老化曲线 $\hat{S} = \{\hat{S}_i\}_{i=1}^{p}$;

5) 基于式 (6) 计算 $\hat{S}$ 中与原始老化数据 Fréchet 距离最小的曲线, 即为最优老化曲线 $\hat{S}_{opt}$;

6) 基于 $\hat{S}_{opt}$ 和式 (7) 计算云服务器的 RUL, 即为云服务器从当前时刻到发生老化的时间间隔.

4 实验结果与分析

为探究基于 OpenStack 云服务器的软件老化现象, 本文基于如图 4 所示的实验平台, 制定了加速老化实验[37], 设计负载发生方案模拟云服务器的实际负载, 并收集云服务器的资源和性能数据, 作为研究软件老化的实验数据. 通过对云服务器的资源和性能数据进行老化预测, 估算剩余寿命, 为云服务器软件再生部署提供理论依据.

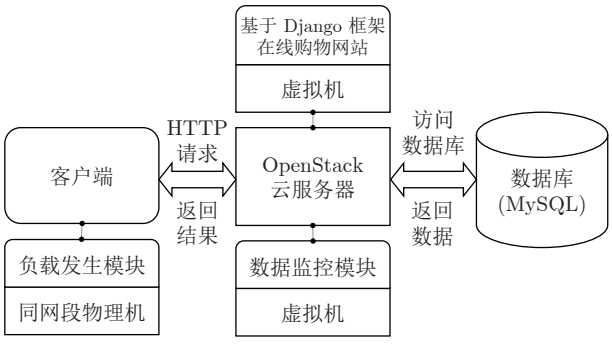


图 4 实验平台

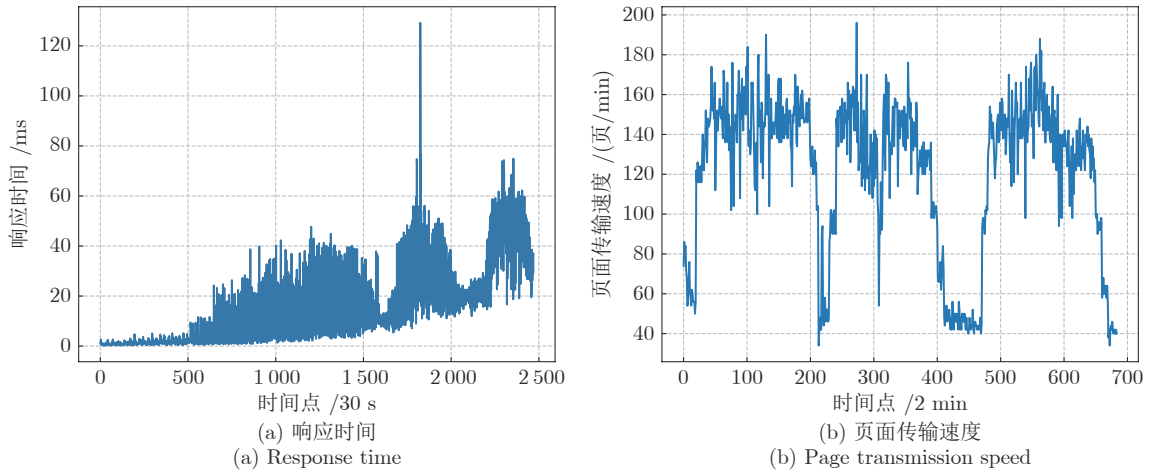
Fig. 4 Test bed

本文实验基于 OpenStack 框架和 QEMU/KVM 管理程序, 实验系统由云服务器、MySQL 数据库服务器和 1 组模拟客户端组成, 如图 4 所示. 云服务器运行环境为 Intel Core i5-4590 CPU, 4 核 8 线程 3.3 GHz, 内存 8 GB, 操作系统为 18.04.1-Ubuntu X86_64. 云服务器上部署基于 Django 框架在线购物网站, 包括用户登录、商品详情、商品查询、添加购物车和账单结算等功能. 客户端编写负载脚本, 通过多线程模拟多用户的购物网站访问场景, 同时监控并收集云服务器的性能参数和服务参数.

首先, 将云服务器的响应时间和页面传输速度作为系统老化数据, 如图 5 所示. 响应时间是用户向服务器发送请求至接收到系统响应的时间, 实验样本数据为 2 471 个, 采样时间间隔为 30 s. 页面传输速度指云服务器平均每分钟发出的页面数量, 实验样本数据为 684 个, 采样时间间隔为 2 min. 由图 5 可知, 随着云服务器系统的持续运行, 云服务器的响应时间逐步增长, 页面传输速度逐步下降即系统服务性能逐渐下降, 则云服务器出现老化现象. 其次, 实验中基于 RUL 的老化预测方法, 评估云服务器何时发生老化.

4.1　特征处理结果

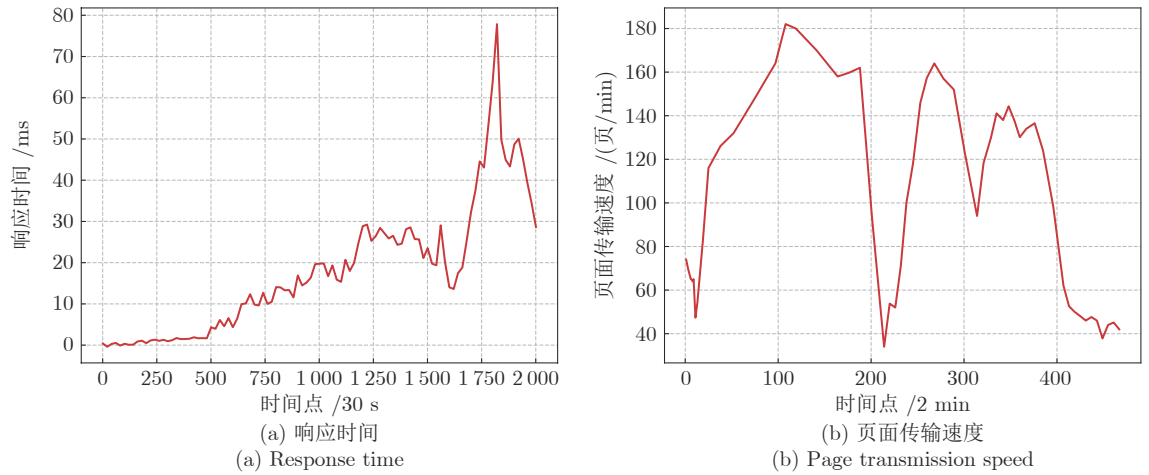
首先, 提取云服务器原始数据的统计特征指标 $\{\boldsymbol{f}^i\}_{i=1}^{10}$, 包括均值、标准差、偏斜度、峰度、双峰值、均方根、峰值因子、波形因子、脉冲因子和裕度因子共 10 项, 其中滑动窗口长度 l 凭经验设置为 30; 其次, 通过算法 1 对提取的统计特征指标进行筛选和融合, 得到云服务器响应时间和页面传输速度的统计特征指标, 如图 6 所示.



(a) 响应时间　(a) Response time　(b) 页面传输速度　(b) Page transmission speed

图 5　OpenStack 云服务器原始数据

Fig. 5　Original data of OpenStack cloud server



(a) 响应时间　(a) Response time　(b) 页面传输速度　(b) Page transmission speed

图 6　原始数据的统计特征指标

Fig. 6　Statistical characteristic index of origin data

4.2 老化曲线拟合结果

首先, 基于线性、多项式和 Sigmoid 核函数的 SVR 对图 6 中的响应时间和页面传输速度统计特征指标序列数据进行稀疏化; 然后, 采用基于密度聚类的 GFF 算法, 对稀疏化后的统计特征指标进行老化曲线拟合, 并基于 Fréchet 距离优化算法选取老化曲线, 设置老化阈值 δ 分别为 60 ms 和 38 (页/min), 可得到如图 7 的最优老化曲线.

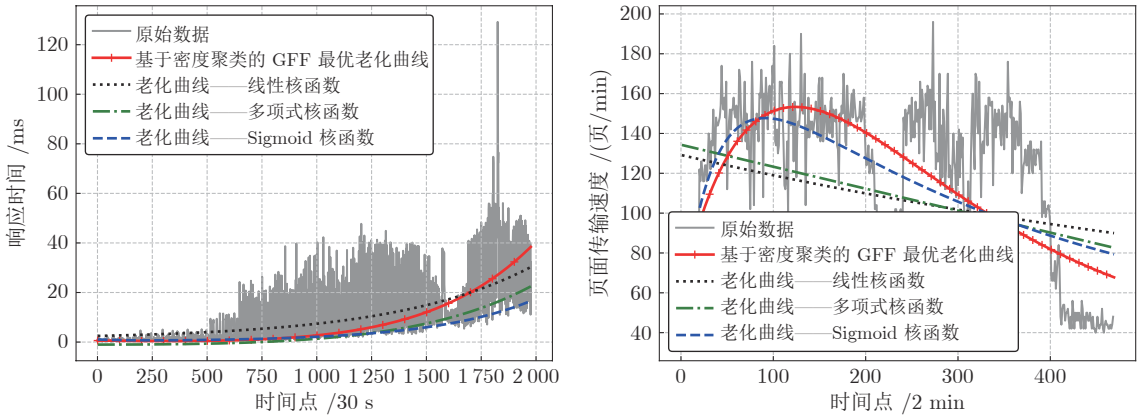
同时, 本文将基于密度聚类的 GFF 最优老化曲线与 SVR 拟合曲线进行对比, 如图 8 所示. 在图 8(a) 中, 本文方法将响应时间统计特征指标序列中的 SVs 点聚为 C_1、C_2 和 C_3 三类. 在图 8(b) 中, 本文方法将页面传输速度统计特征指标序列中的 SVs 点聚为 C_1、C_2、C_3 和 C_4 四类. 由图 8 可知, 本文基于密度聚类的 GFF 最优老化曲线相较于 SVR 拟合曲线, 更接近于原始数据的统计特征指标曲线, 能更好地拟合统计特征指标序列数据的变化趋势.

此外, 本文采用均方根误差 (Root mean square error, RMSE)[38] 对比 2 种老化曲线拟合方法, 计算公式如下:

$$\mathrm{RMSE}=\sqrt{\frac{1}{h}\sum_{i=1}^{h}(y_{pre}(i)-y_{true}(i))^2} \tag{8}$$

式中, h 为样本数量, y_{pre} 为拟合曲线序列数据, y_{true} 为统计特征指标序列数据.

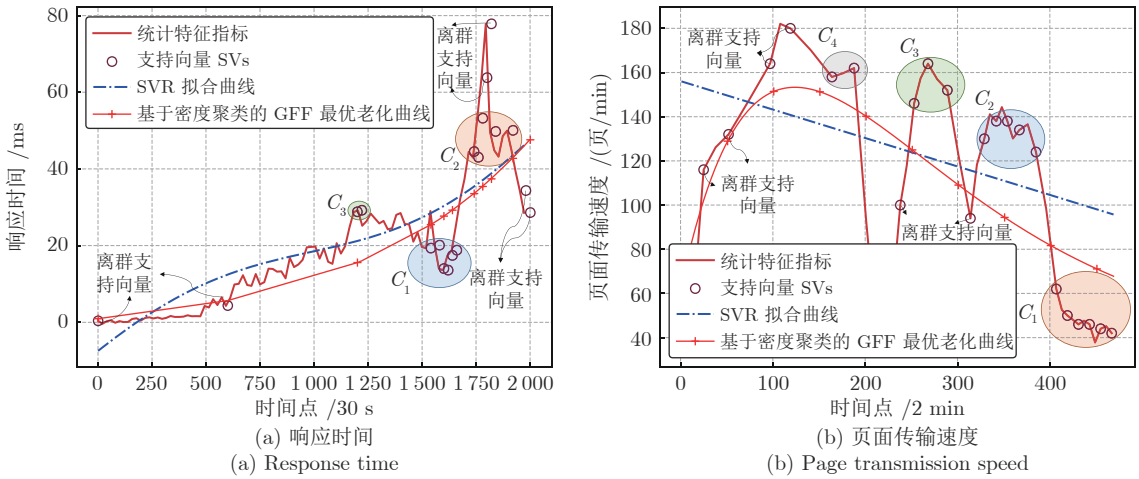
表 1 给出了老化曲线拟合误差对比结果. 由表 1 可知, 本文基于密度聚类的 GFF 最优老化曲线与原数据序列误差更小, 其误差值相较于 SVR 拟合曲线, 在响应时间指标上降低了 62.33%, 在页面传



(a) 响应时间 (a) Response time (b) 页面传输速度 (b) Page transmission speed

图 7 基于 Fréchet 距离选取最优老化曲线

Fig. 7 Select the optimal aging curve via the Fréchet distance



(a) 响应时间 (a) Response time (b) 页面传输速度 (b) Page transmission speed

图 8 老化曲线拟合对比

Fig. 8 Comparison of aging curve fitting

表 1　老化曲线拟合 RMSE 对比 (%)
Table 1　Comparison of aging curve fitting RMSE (%)

拟合方法	响应时间集	页面传输速度集
基于密度聚类的 GFF	21.598	47.129
SVR	57.334	114.239

输速度指标上降低了 58.75%. 可见本文方法对原始统计特征指标数据的拟合更加精确. 同时, 本文采用支持向量回归对统计特征指标进行稀疏化处理后, 基于稀疏化后的支持向量 SVs 序列数据及逆行老化曲线拟合, 能够保留原有数据序列的重要特征, 反映原有统计特征指标数据的变化规律.

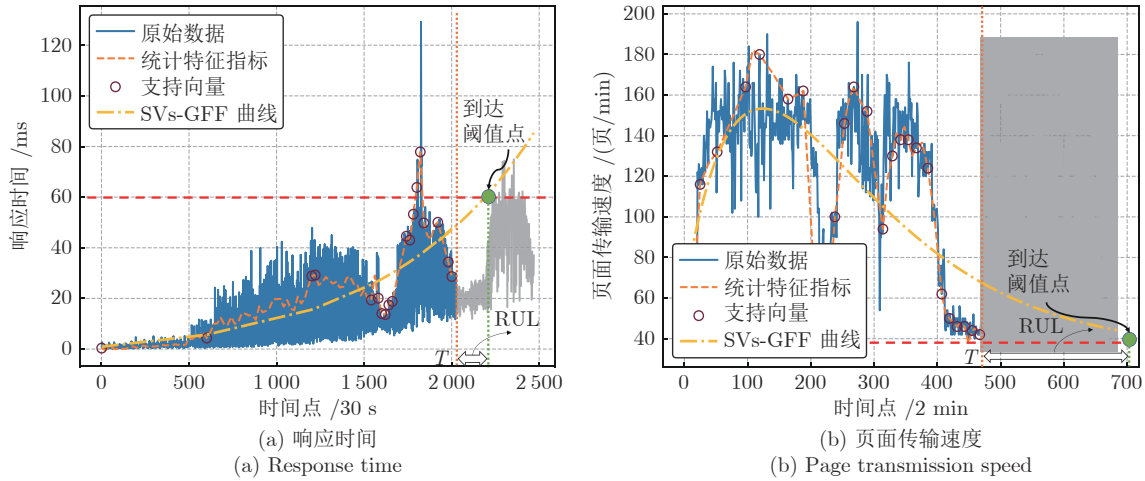
4.3　老化预测结果

基于第 4.2 节的云服务器响应时间和页面传输速度的最优老化曲线, 依据本文提出的 SVs-GFF 预测方法, 可获得云服务器老化数据序列变化趋势, 得到图 9 的云服务器老化预测结果, 并能计算在时间点 T 时云服务器的 RUL. 例如, 图 9(a) 中基于响应时间的老化曲线, 在时间点 $T = 60\,990$ s 时, 云服务器 RUL 为 5 400 s. 图 9(b) 中页面传输速度的老化曲线, 在时间点 $T = 920$ min 时, 云服务器 RUL 为 7 680 s.

4.4　预测方法性能评价

首先, 本文以云服务器的 RUL 预测结果作为依据, 并与 SVR[39]、GFF[40]、粒子滤波 (Particle filtering, PF) 模型[41]、人工神经网络 (Artificial neural network, ANN) 模型[42]、LSTM[43] 和 Markov 模型[44] 进行比较, 评估本文提出的 SVs-GFF 预测方法性能. 各预测方法的参数设置如表 2 所示.

图 10 给出了云服务器响应时间 (预测时间间隔 500 ~ 1 250 (时间点/30 s)) 和页面传输速度 (预测时间间隔 523 ~ 673 (时间点/2 min)), 当老化阈值 δ 分别为 60 ms 和 38 (页/min) 时, 采用各预测方法对云服务器剩余寿命 RUL 的评估结果. 图 11 给出了各预测方法对 RUL 预测绝对误差 (预测值与真实值之差的绝对值) 对比结果. 由图 10 可知, 各预测方法在初始阶段对 RUL 预测值偏离实际值较大, 但随着时间点后移, 对 RUL 的预测偏差逐步



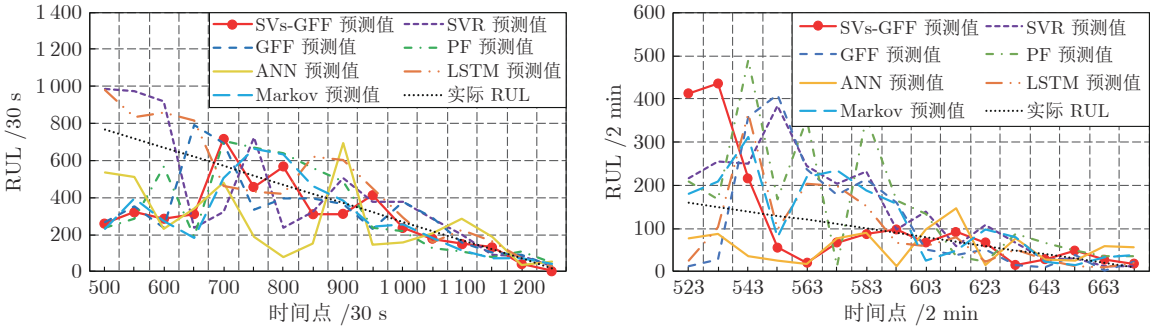
(a) 响应时间
(a) Response time

(b) 页面传输速度
(b) Page transmission speed

图 9　云服务器老化预测结果
Fig. 9　Cloud server aging prediction results

表 2　不同预测方法的参数设置
Table 2　Parameter setting of different prediction methods

预测方法	参数设置
SVs-GFF	高斯函数中 α、β、σ 和 γ 的初始值: 0, 寻优方法: 最小二乘法, SVR 中正则化参数: 1.0, 距离阈值 ε : 0.5
SVR	正则化参数: 1, 核函数: RBF
GFF	高斯函数中 α、β、σ 和 γ 的初始值: 0, 寻优方法: 最小二乘法
PF	基于 PF 模型更新指数模型参数, 老化特征值 α : 1.979, 指数模型参数 b : 0.00271, c : −0.1697, 白噪声标准差 d : −0.06942
ANN	神经元数: [输入层: 30, 隐藏层 1: 64, 隐藏层 2: 64, 隐藏层 3: 32, 输出层: 1], 激活函数: ReLU, 迭代次数: 100
LSTM	神经元数: [输入层: 10, 隐藏层 1: 32, 隐藏层 2: 32, 隐藏层 3: 16, 输出层: 1], 激活函数: ReLU, 迭代次数: 100
Markov	基于 Markov 模型更新指数模型参数, 指数模型参数的初始值: 0

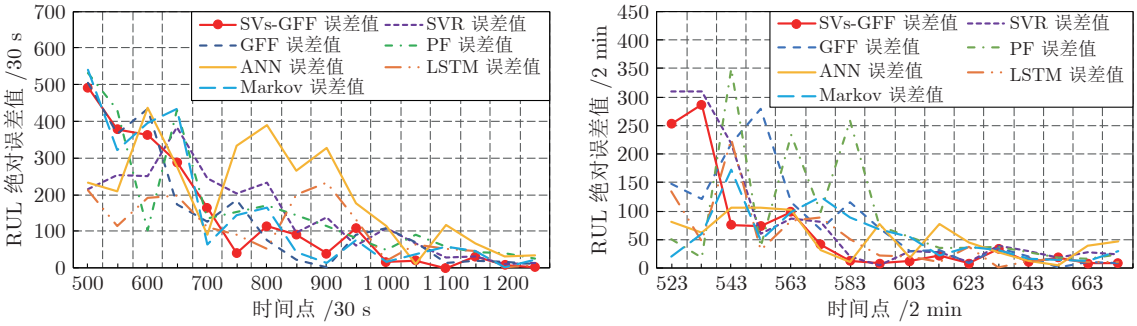


(a) 响应时间 (b) 页面传输速度

(a) Response time (b) Page transmission speed

图 10 云服务器 RUL 预测结果

Fig. 10 Cloud server RUL prediction results



(a) 响应时间 (b) 页面传输速度

(a) Response time (b) Page transmission speed

图 11 云服务器 RUL 预测绝对误差

Fig. 11 Absolute error of cloud server RUL prediction results

缩小, 最终收敛至实际 RUL. 相比较而言, 本文方法的预测曲线更接近真实 RUL 曲线, 并且由图 11 可知, 本文方法预测绝对误差相对较小, 表明本文方法预测效果更好.

其次, 本文以累计相对精度 (Cumulative relative accuracy, CRA)[45] 和收敛速度 C_{PE}[46] 作为评价指标, 评估预测方法的性能.

CRA 评估预测方法的精度, 计算方法如下:

$$\mathrm{CRA} = \sum_{k=1}^{H} \omega_k RA(T_k) \tag{9}$$

$$RA(T_k) = 1 - \frac{|L_{true}(T_k) - L(T_k)|}{L_{true}(T_k)} \tag{10}$$

式中, $\omega_k = k/\sum_{i=1}^{H} i$ 为归一化权重, H 为预测 RUL 的实验次数 (本文设置 H =16). $RA(T_k)$ 为 T_k 时刻对 RUL 的相对预测精度, $L_{true}(T_k)$ 为 T_k 时刻 RUL 的真实值, $L(T_k)$ 为 T_k 时刻对 RUL 的预估值. CRA 值越大, 表明方法的预测精度越高.

C_{PE} 评估预测方法的收敛速度, 用预测误差曲线下方区域的原点和质心之间的欧氏距离来度量, 计算方法如下:

$$C_{PE} = \sqrt{(C_x - T_1)^2 + C_y^2} \tag{11}$$

式中, T_1 为第 1 次预估 RUL 的时刻点, (C_x, C_y) 为预测误差曲线下方区域的质心, $(T_1, 0)$ 为预测误差曲线下方区域的原点. 预测云服务器的 RUL 时, C_{PE} 值越小, 表明预测值逼近真实值的速度越快, 预测方法对 RUL 评估过程的收敛速度越快.

表 3 给出了 CRA 和 C_{PE} 两种评价指标在云服务器响应时间和页面传输速度数据集上预测性能对比结果. 由表 3 可知, 在两个数据集上, 相较 GFF、SVR、PF、ANN、LSTM 和 Markov 六种预测方法, 本文 SVs-GFF 方法 CRA 值最大, 且 C_{PE} 值最小. 在响应时间数据集上, 相较于其他六种预测方法, 本文 SVs-GFF 方法的 CRA 值分别提高了 13.66%、1.51%、7.04%、18.54%、0.38% 和 5.36%, C_{PE} 值分别降低了 4.90%、5.72%、1.64%、17.63%、21.93% 和 1.65%. 在页面传输速度数据集上, 本文 SVs-GFF 方法的 CRA 值分别提高了 20.6%、2.07%、9.74%、21.91%、10.86% 和 8.12%, C_{PE} 值分别降低了 2.70%、2.94%、9.00%、15.65%、19.53% 和 7.80%. 这表明本文 SVs-GFF 方法的预测精度最

表 3　预测性能比较

Table 3　Comparison of prediction performances

数据集名称	评价指标	SVs-GFF	GFF	SVR	PF	ANN	LSTM	Markov
响应时间数据集	CRA	0.904	0.769	0.891	0.841	0.737	0.901	0.858
	C_{PE}	15.117	15.896	16.035	15.369	18.353	19.360	15.371
页面传输速度数据集	CRA	0.879	0.698	0.861	0.801	0.721	0.792	0.813
	C_{PE}	16.487	16.945	16.987	17.897	19.547	20.487	17.881

高, 且收敛速度最快.

4.5　预测方法的普适性

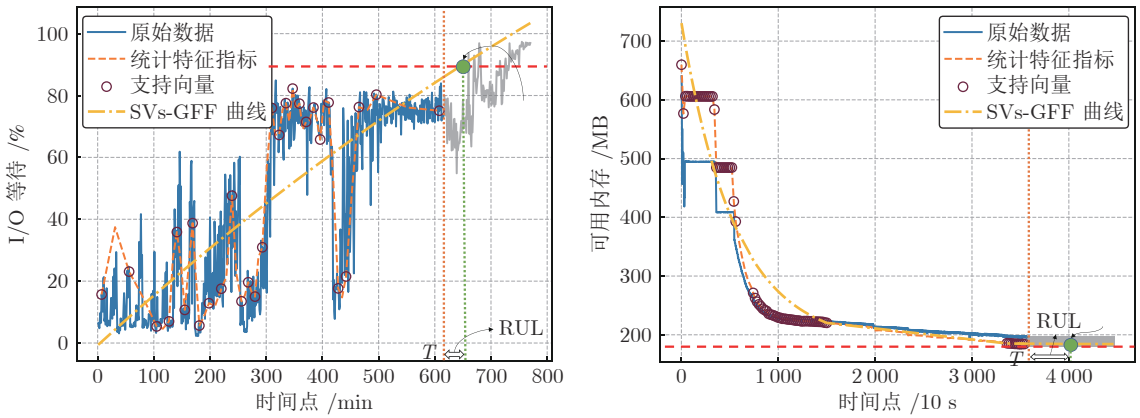
为验证预测方法的普适性, 基于相同实验平台做了两组老化实验, 并提取了云服务器的性能参数数据. 选取 I/O 等待和可用内存作为老化指标, 其中 I/O 等待为 CPU 空闲且存在未完成 I/O 操作的等待时间占总时间的比例, 该值受到系统响应时间和 CPU 使用率的影响, 数值过高说明系统响应时间长, 整个系统的处理速度降低. I/O 等待样本数据为 770 个, 采样时间间隔为 1 min; 可用内存样本数据为 4 460 个, 采样时间间隔为 10 s. 老化阈值 δ 分别为 90% 和 180 MB.

图 12 给出了采取本文 SVs-GFF 方法得到的云服务器老化预测结果, 可获得老化数据序列变化趋势, 并能计算在任意时间点 T 时云服务器的 RUL. 图 12(a) 给出基于 I/O 等待的老化预测曲线, 在时间点 $T = 621$ min 时, 云服务器的 RUL 为 32 min. 图 12(b) 给出基于可用内存的老化预测曲线, 在时间点 $T = 35\,700$ s 时, 云服务器的 RUL 为 4 420 s. 图 13 和图 14 分别给出了基于 I/O 等待和可用内存的云服务器的 RUL 预测结果及预测绝对误差对比结果, 可知本文的 SVs-GFF 老化预测方法, 相较 GFF、SVR、PF、ANN、LSTM 和 Markov 六种预测方法, 仍保持较高的预测精度和收敛速度.

综上所述, 本文筛选并融合老化数据的多个统计特征指标, 可捕获云服务器在复杂运行环境下老化数据变化规律, 提高预测精度. SVR 将统计特征数据稀疏化, 降低老化预测方法计算量. 本文方法考虑稀疏化后的数据的空间分布不均衡特性, 基于聚类方法对数据分配不同权重信息, 避免预测过程陷入局部最优解. 另外, 高斯函数拟合过程计算复杂度低, 用其预测稀疏化后数据, 无需大规模训练过程, 可提高预测收敛速度. 本文提出的基于 RUL 的老化预测方法, 为运维人员提供软件再生时机, 保障系统可靠性[47−48].

5　结束语

软件老化预测可以保障软件系统可靠运行, 降低系统风险与损失, 因此该研究问题一直以来受到学术界和工业界的关注. 针对收集到的 OpenStack 云服务器性能数据, 本文提出基于 RUL 和 SVs-GFF 的老化预测方法, 首先提取老化数据的统计特征指标, 采用具有不同核参数的 SVR 对统计特征指标数据进行稀疏化处理, 获取多组支持向量 SVs. 然后使用基于密度聚类的 GFF 拟合各组支持向量

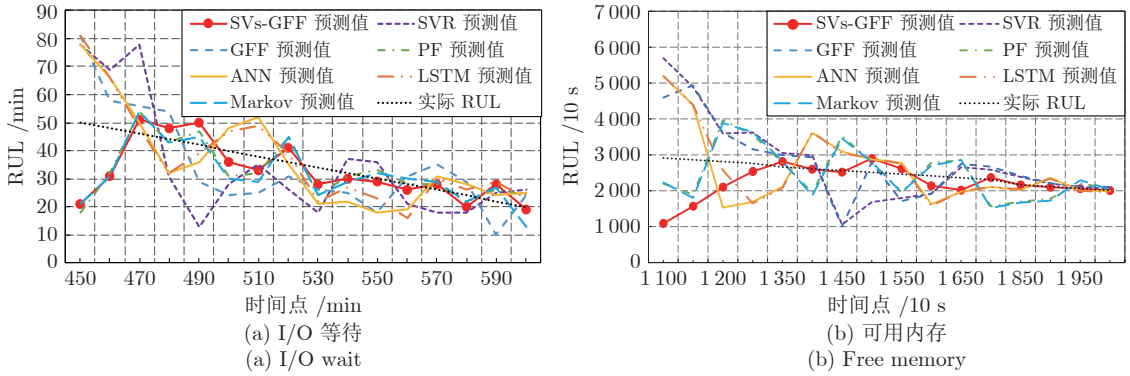


(a) I/O 等待　(b) 可用内存

(a) I/O wait　(b) Free memory

图 12　云服务器老化预测结果

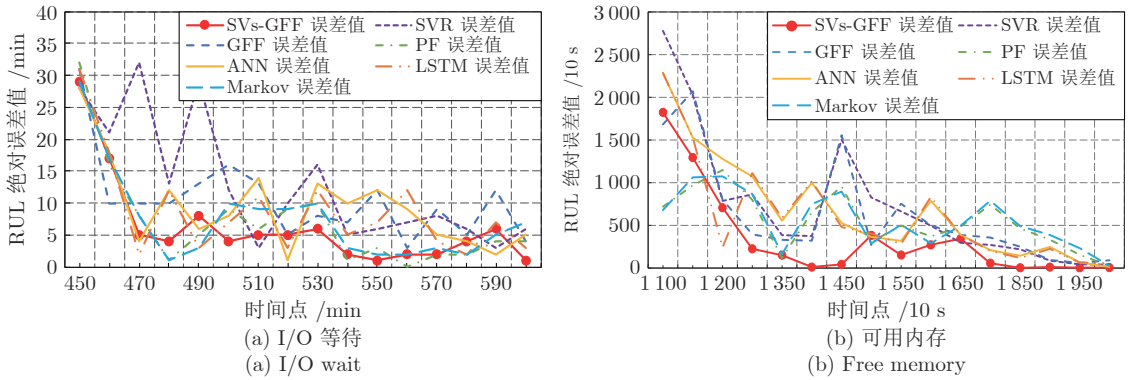
Fig. 12　Cloud server aging prediction results



(a) I/O 等待 (a) I/O wait (b) 可用内存 (b) Free memory

图 13 云服务器 RUL 预测结果

Fig. 13 Cloud server RUL prediction results



(a) I/O 等待 (a) I/O wait (b) 可用内存 (b) Free memory

图 14 云服务器 RUL 预测绝对误差

Fig. 14 Absolute error of cloud server RUL prediction results

SVs, 得到多条表征云服务器老化状态的老化曲线. 最后通过 Fréchet 距离优化算法选取最优老化曲线, 评估云服务器老化前的剩余寿命 RUL. 实验结果表明, 本文所提老化预测方法, 不依赖于人工经验选取超参数, 提高了预测精度和收敛速度. 本文首次通过剩余使用寿命解决云服务器老化预测问题, 相较于现有基于数据度量的软件老化预测方法, 本文提出的基于 RUL 的老化预测方法, 可为软件运行维护人员提供软件再生时机, 从而保障系统可靠性.

下一步将研究预测方法对于数据变化情况下的敏感性问题. 使当数据出现急剧变化时, 仍能获得较为精确的预测结果.

References

1 Grottke M, Matias R, Trivedi K S. The fundament of software aging. In: Proceedings of the IEEE International Conference on Software Reliability Engineering Workshops. Seattle, USA: IEEE, 2008. 1−6

2 Vizarreta P, Sieber C, Blenk A, Bemten A V, Trivedi K. Ares: A framework for management of aging and rejuvenation in softwarized networks. *IEEE Transactions on Network and Service Management*, 2021, **18**(2): 1389−1400

3 Tai A T, Chau S N, Alkalaj L. On-board preventive maintenance: Analysis of effectiveness and optimal duty period. In: Proceedings of the 3rd International Workshop on Object Oriented Real-time Dependable Systems. Newport Beach, USA: IEEE, 1997. 40−47

4 Xiang J, Weng C, Zhao D, Andrzejak A. Software aging and rejuvenation in android: New models and metrics. *Software Quality Journal*, 2020, **28**(3): 85−106

5 Grottke M, Li L, Vaidyanathan K, Trivedi K S. Analysis of software aging in a web server. *IEEE Transactions on Reliability*, 2006, **55**(3): 411−420

6 Pietrantuono R, Russo S. A survey on software aging and rejuvenation in the cloud. *Software Quality Control*, 2020, **28**(1): 7−38

7 Shi Quan, Hu Chang-Hua, Si Xiao-Sheng, Hu Xiao-Xiang, Zhang Zheng-Xin. Remaining useful lifetime prediction method of controlled systems considering performance degradation of actuator. *Acta Automatica Sinica*, 2019, **45**(5): 941−952
(施权, 胡昌华, 司小胜, 扈晓翔, 张正新. 考虑执行器性能退化的控制系统剩余寿命预测方法. 自动化学报, 2019, **45**(5): 941−952)

8 Huang Y, Kintala C, Kolettis N, Fulton N D. Software rejuvenation: Analysis, module and applications. In: Proceedings of the 25th International Symposium on Fault-tolerant Computing. Pasadena, USA: IEEE, 1995. 381−390

9 Andrade E, Machida F. Analysis of software aging impacts on plant anomaly detection with edge computing. In: Proceedings of the IEEE International Symposium on Software Reliability Engineering Workshops. Berlin, Germany: IEEE, 2019. 204−210

10 Machida F, Maciel P R M. Markov chains and Petri nets for

software rejuvenation systems. In: Proceedings of the IEEE International Symposium on Software Reliability Engineering Workshops. Coimbra, Portugal: IEEE, 2020. 325−326

11 Dohi T, Goseva-Popstojanova K, Trivedi K S. Estimating software rejuvenation schedule in high assurance systems. *The Computer Journal*, 2001, **47**: 473−485

12 Machida F, Xiang J, Tadano K, Maeno Y. Lifetime extension of software execution subject to aging. *IEEE Transactions on Reliability*, 2017, **66**(1): 123−134

13 Ning G, Zhao J, Lou Y. Optimization of two-granularity software rejuvenation policy based on the Markov regenerative process. *IEEE Transactions on Reliability*, 2016, **65**(4): 1630−1646

14 Okamura H, Dohi T. Performance-aware software rejuvenation strategies in a queuing system. In: Proceedings of the IEEE Second International Workshop on Software Aging and Rejuvenation. San Jose, USA: IEEE, 2010. 1−6

15 Levitin G, Xing L, Xiang Y. Cost minimization of real-time mission for software systems with rejuvenation. *Reliability Engineering & System Safety*, 2020, **193**: Article No. 106593

16 Zheng Z, Trivedi K S. Guest editorial: Special issue on modeling and mitigation techniques for software aging. *Software Quality Journal*, 2020, **28**(1): 3−5

17 Vaidyanathan K, Trivedi K S. A comprehensive model for software rejuvenation. *IEEE Transactions on Dependable and Secure Computing*, 2005, **2**(2): 124−137

18 Yan Y, Guo P. A practice guide of software aging prediction in a web server based on machine learning. *China Communications*, 2016, **13**(6): 225−235

19 Pereira P, Araujo J, Matos R. Software rejuvenation in computer systems: An automatic forecasting approach based on time series. In: Proceedings of the 37th IEEE International Performance Computing and Communications Conference. Orlando, USA: IEEE, 2018. 1−8

20 Cotroneo D, Natella R, Pietrantuono R. Software aging analysis of the Linux operating system. In: Proceedings of the IEEE 21st International Symposium on Software Reliability Engineering. San Jose, USA: IEEE, 2010. 71−80

21 Matos R, Araujo J, Maciel P. Software rejuvenation in eucalyptus cloud computing infrastructure: A method based on time series forecasting and multiple thresholds. In: Proceedings of the IEEE 3rd International Workshop on Software Aging and Rejuvenation. Hiroshima, Japan: IEEE, 2011. 38−43

22 Zheng Peng-Fei, Qi Yong, Chen Peng-Fei. Multivariate time series analysis of software aging. *Journal of Frontiers of Computer Science & Technology*, 2012, **6**(2): 33−41
(郑鹏飞, 齐勇, 陈鹏飞. 软件老化的多元时间序列分析方法. 计算机科学与探索, 2012, **6**(2): 33−41)

23 Islam S, Keung J, Lee K. Empirical prediction models for adaptive resource provisioning in the cloud. *Future Generation Computer Systems*, 2012, **28**(1): 155−162

24 Yan Y, Guo P, Liu L. A practice of forecasting software aging in an IIS web server using SVM. In: Proceedings of the IEEE International Symposium on Software Reliability Engineering Workshops. Naples, Italy: IEEE, 2014. 443−448

25 Qiao Y, Zheng Z, Fang Y. An empirical study on software aging indicators prediction in android mobile. In: Proceedings of the IEEE International Symposium on Software Reliability Engineering Workshops. Memphis, USA: IEEE, 2018. 271−277

26 Padhy N, Panigrahi R, Neeraja K. Threshold estimation from software metrics by using evolutionary techniques and its proposed algorithms, models. *Evolutionary Intelligence*, 2021, **14**(2): 315−329

27 Yan Y. Software aging prediction using neural network with ridge. *IET Software*, 2020, **14**(6): 517−524

28 Liu J, Tan X, Wang Y. CSSAP: Software aging prediction for cloud services based on ARIMA-LSTM hybrid model. In: Proceedings of the IEEE International Conference on Web Services. Beijing, China: IEEE, 2019. 283−290

29 Meng Hai-Ning, Tong Xin-Yu, Shi Yue-Kai, Zhu Lei, Feng Kai, Hei Xin-Hong. Cloud server aging prediction method based on hybrid model of auto-regressive integrated moving average and recurrent neural network. *Journal on Communications*, 2021, **42**(1): 163−171
(孟海宁, 童新宇, 石月开, 朱磊, 冯锴, 黑新宏. 基于 ARIMA-RNN 组合模型的云服务器老化预测方法. 通信学报, 2021, **42**(1): 163−171)

30 Nguyen H, Choi Y, Bui X N. Predicting blast-induced ground vibration in open-pit mines using vibration sensors and support vector regression-based optimization algorithms. *Sensors*, 2020, **20**(1): 1−21

31 Kwak M, Lkhagvasuren B, Sun X. An efficient method for fitting Gaussian functions. *Iranian Journal of Science and Technology, Transactions A: Science*, 2021, **45**(3): 1043−1056

32 Guo L, Li N, Jia F. A recurrent neural network-based health indicator for remaining useful life prediction of bearings. *Neurocomputing*, 2017, **240**(3): 98−109

33 Cotroneo D, Natella R, Pietrantuono R, Russo S. A survey of software aging and rejuvenation studies. *ACM Journal on Emerging Technologies in Computing Systems*, 2014, **10**(1): 1−34

34 Ahmed F. *K* and starting means for *K*-means algorithm. *Journal of Computational Science*, 2021, **55**(1): Article No. 101445

35 Gudmundsson J, Van Renssen A, Saeidi Z. Translation invariant Fréchet distance queries. *Algorithmica*, 2021, **83**: 1−19

36 Si X S, Wang W, Hu C H. A wiener-process-based degradation model with a recursive filter algorithm for remaining useful life estimation. *Mechanical Systems and Signal Processing*, 2013, **35**(1): 219−237

37 Matias R, Barbetta P A, Trivedi K S. Accelerated degradation tests applied to software aging experiments. *IEEE Transactions on Reliability*, 2010, **59**(1): 102−114

38 Engel Y, Mannor S, Meir R. The kernel recursive least-squares algorithm. *IEEE Transaction on Signal Process*, 2004, **52**(8): 2275−2285

39 Qiu G, Gu Y, Chen J. Selective health indicator for bearings ensemble remaining useful life prediction with genetic algorithm and Weibull proportional hazards model. *Measurement*, 2020, **150**: Article No. 107097

40 Aye S A, Heyns P S. An integrated Gaussian process regression for prediction of remaining useful life of slow speed bearings based on acoustic emission. *Mechanical Systems and Signal Processing*, 2017, **84**: 485−498

41 Guo R, Sui J. Remaining useful life prognostics for the electro-hydraulic servo actuator using Hellinger distance-based particle filter. *IEEE Transactions on Instrumentation and Measurement*, 2020, **69**(4): 1148−1158

42 Fauzi M F A M, Aziz I A, Amiruddin A. The prediction of remaining useful life (RUL) in oil and gas industry using artificial neural network (ANN) algorithm. In: Proceedings of the IEEE Conference on Big Data and Analytics. Milan, Italy: IEEE, 2019. 7−11

43 Xia J, Feng Y W, Lu C. LSTM-based multi-layer self-attention method for remaining useful life estimation of mechanical systems. *Engineering Failure Analysis*, 2021, **125**(12): Article No. 105385

44 Wang T, Liu Z, Mrad N A. Probabilistic framework for remaining useful life prediction of bearings. *IEEE Transactions on Instrumentation and Measurement*, 2021, **70**: 1−12

45 Saxena A, Celaya J, Saha B A. Metrics for offline evaluation of prognostic performance. *International Journal of Prognostics & Health Management*, 2010, **1**(1): 2153−2648

46 Wang B, Lei Y, Li N. A hybrid prognostics approach for estimating remaining useful life of rolling element bearings. *IEEE Transactions on Reliability*, 2018, **86**: 1452−1463

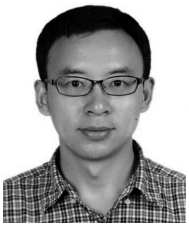
47 Zhang Zheng-Xin, Hu Chang-Hua, Si Xiao-Sheng, Zhang Wei. Degradation modeling and remaining useful life prediction with bivariate time scale. *Acta Automatica Sinica*, 2017, **43**(10): 1789−1798

(张正新, 胡昌华, 司小胜, 张伟. 双时间尺度下的设备随机退化建模与剩余寿命预测方法. 自动化学报, 2017, **43**(10): 1789−1798)

48 Pei Hong, Hu Chang-Hua, Si Xiao-Sheng, Zhang Zheng-Xin, Du Dang-Bo. Remaining life prediction information-based maintenance decision model for equipment under imperfect maintenance. *Acta Automatica Sinica*, 2018, **44**(4): 719−729
(裴洪, 胡昌华, 司小胜, 张正新, 杜党波. 不完美维护下基于剩余寿命预测信息的设备维护决策模型. 自动化学报, 2018, **44**(4): 719−729)

孟海宁 西安理工大学计算机科学与工程学院副教授. 主要研究方向为机器学习, 故障诊断与预测. 本文通信作者. E-mail: hnmeng@xaut.edu.cn
(**MENG Hai-Ning** Associate professor at the School of Computer Science and Engineering, Xi′an University of Technology. Her research interest covers machine learning and fault prognosis & prediction. Corresponding author of this paper.)

童新宇 西安理工大学计算机科学与工程学院博士研究生. 主要研究方向为机器学习, 时间序列预测.
E-mail: tongxinyu@stu.xaut.edu.cn
(**TONG Xin-Yu** Ph.D. candidate at the School of Computer Science and Engineering, Xi′an University of Technology. His research interest covers machine learning and time series prediction.)



谢 国 西安理工大学教授. 主要研究方向为数据分析, 故障诊断.
E-mail: guoxie@xaut.edu.cn
(**XIE Guo** Professor at Xi′an University of Technology. His research interest covers data analysis and fault diagnosis.)

张贝贝 西安理工大学计算机科学与工程学院讲师. 主要研究方向为数据挖掘, 大数据技术.
E-mail: bbzhang115@hotmail.com
(**ZHANG Bei-Bei** Lecturer at the School of Computer Science and Engineering, Xi′an University of Technology. His research interest covers data mining and big data technology.)

黑新宏 西安理工大学计算机科学与工程学院教授. 主要研究方向为机器学习, 系统安全.
E-mail: heixinhong@xaut.edu.cn
(**HEI Xin-Hong** Professor at the School of Computer Science and Engineering, Xi′an University of Technology. His research interest covers machine learning and system security.)