计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2021,30(7):253−258 [doi: 10.15888/j.cnki.csa.007976]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于 LSTM 网络的 Web 软件系统实时剩余寿命预测①

党伟超, 李　涛, 白尚旺

(太原科技大学 计算机科学与技术学院, 太原 030024)
通讯作者: 李　涛, E-mail: s20180580@stu.tyust.edu.cn

摘　要: Web 软件系统剩余使用寿命的预测精度是影响 Web 软件系统抗衰决策的重要方面, 为此, 提出了一种基于长短期记忆网络的 Web 软件系统实时剩余寿命预测方法. 首先搭建加速寿命测试实验平台, 收集反映 Web 软件系统老化趋势的性能指标, 然后根据该指标数据的时序特性, 建立了一种基于长短时记忆网络 (LSTM) 的 Web 软件系统实时剩余寿命预测模型, 并对该模型进行了训练. 实验结果表明, 该预测模型能够有效对 Web 软件系统的剩余寿命进行实时预测, 具有更好的准确性和适用性. 将所提模型应用于 Web 软件系统寿命预测中, 能够有效完成预测, 该方法为优化系统抗衰决策提供了技术支撑.

关键词: Web 软件系统; 抗衰决策; 剩余使用寿命; 长短期记忆网络

引用格式: 党伟超,李涛,白尚旺.基于 LSTM 网络的 Web 软件系统实时剩余寿命预测.计算机系统应用,2021,30(7):253–258. http://www.c-s-a.org.cn/1003-3254/7976.html

Real-Time Residual Life Prediction of Web-Based Software System Based on LSTM

DANG Wei-Chao, LI Tao, BAI Shang-Wang

(School of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan 030024, China)

Abstract: The prediction accuracy of the Remaining Useful Life (RUL) is of vital importance to rejuvenation decision of Web-based software systems, so we propose a real-time prediction method for the remaining useful life of Web-based software systems based on the Long Short-Term Memory (LSTM) network. Firstly, an accelerated life test platform is built to collect the performance indicators of the aging of Web-based software systems. Then, according to the time-series characteristics of indicator data, an LSTM-based real-time prediction model for the remaining useful life of Web-based software systems is constructed and trained. The experimental results show that the model can effectively predict the remaining useful life of Web-based software systems in real time with higher accuracy and stronger applicability. This method provides technical support for optimizing system's rejuvenation decision.

Key words: Web-based software system; rejuvenation decision; Remaining Useful Life (RUL); Long Short-Term Memory (LSTM)

长时间运行的软件系统, 会出现软件错误不断累积、占用资源不断增加、性能持续下降, 最终导致软件失效或崩溃的软件老化现象[1]. 对于发生老化的软件系统, 在适当时刻主动进行抗衰操作可以避免软件发生失效[2]. 因此, 如果能够实时准确预测软件系统的剩余寿命, 通过预测软件剩余使用寿命来推导软件最优

① 基金项目: 山西省应用基础研究项目 (201901D111266); 山西省哲学社会科学规划课题 (2020YY161)
Foundation item: Applied Basic Research Project of Shanxi Province (201901D111266); Philosophic Social Science Planning Project of Shanxi Province (2020YY161)
收稿时间: 2020-10-21; 修改时间: 2020-11-18; 采用时间: 2020-12-02; csa 在线出版时间: 2021-06-30

抗衰时刻, 将会减少抗衰决策判断误差.

软件剩余使用寿命 (Remaining Useful Life, RUL) 指软件以当前的运行条件, 能够实现其正常功能的剩余时间[3]. RUL 常见的预测方法可分为基于模型与数据驱动的方法. 基于模型的方法是指通过构建设备退化的数学或物理模型来进行预测. 马波等人提出了一种基于状态监测信息和滚动轴承退化物理模型的寿命预测方法[4].基于模型的方法取得了不错的效果, 但它依赖与特定的模型, 不利于推广[5].

数据驱动中的深度学习方法由于具有强大的数据驱动能力, 并且无需知道确切的物理模型和领域知识, 以及在非线性映射特征提取方面的优异性能, 成为了剩余寿命预测领域中热门方法[6]. 闫楚良等人[7] 和王佳炜等人[8] 用 BP 神经网络分别对材料疲劳寿命和电磁继电器进行了预测. Felix 等人提出用循环神经网络 (Recurement Netural Network, RNN) 对航空发动机领域剩余寿命进行了预测[9]. RNN 网络在结构设计中引入了时序的概念, 在学习具有内在依赖性的时序数据时能够产生对过去数据的记忆状态, 能够从原始数据获取更多的数据规律性特征. LSTM (Long Short-Term Memory) 作为一种改进后的 RNN 网络, 成为剩余寿命预测的热点技术. 在航天发动机、锂电池、滚动轴承等剩余寿命预测领域都取得了成功[10–12].

基于上述分析, 本文提出了一种基于 LSTM 的 Web 软件系统实时剩余寿命预测模型, 该模型充分考虑了 Web 软件系统资源消耗的时间特性, 将当前系统的剩余寿命动态地与 Web 资源损耗情况相关联.

1 模型原理

1.1 长短期记忆网络

LSTM 是一种改进之后的循环神经网络, 可以解决 RNN 感知能力下降的问题[13]. 与 RNN 相比, LSTM 在其基础上增加了一个细胞状态 (cell state), 内部有 4 个网络层. 一个典型的 LSTM 通过 3 个门来控制细胞状态, 这 3 个门分别为遗忘门、输入门和输出门. 如图 1 所示. 门控循环单元 (Gate Recurrent Unit, GRU) 是 LSTM 网络的一种变体, 它组合了遗忘门和输入门到一个单独的“更新门”中, 合并了细胞状态和隐藏状态, 没有输出门, 增加了重置门, 如图 2 所示.

LSTM 三个门基于 *Sigmoid* 函数来增加或删除细胞状态中的信息, 其中遗忘门和控制门用来控制上一时刻细胞状态c_{t-1}和当前输入新生成$\tilde{c}_t$里有多少信息可以加到当前的细胞状态中c_t去, 通过遗忘门和输入门的输出, 更新细胞状态. 式 (1) 为给定一个老化序列数据 $\{x_1, x_2, \cdots, x_t, \cdots, x_T\}$, 其中$x_t \in R^d$, 其中$x_t \in R^d$, d 为老化指标信息特征个数, T 为时间步, 当输入网络时各门的计算结果.

$$\begin{cases} f_t = Sigmoid(W_{hf}h_{t-1} + W_{xf}x_t + b_f) \\ i_t = Sigmoid(W_{hi}h_{t-1} + W_{xi}x_t + b_i) \\ o_t = Sigmoid(W_{ho}h_{t-1} + W_{xo}x_t + b_o) \\ \tilde{c}_t = \tanh(W_{hc} \cdot h_{t-1} + W_{xc} \cdot x_t + b_c) \\ c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\ h_t = o_t \odot \tanh(c_t) \end{cases} \tag{1}$$

式中,f_t, $\mathrm{i_t}$, $\mathrm{o_t} \in R^m$ 分别表示遗忘门、输入门、输出门的计算结果; $\tilde{c}_t$, c_t, $h_t \in R^m$ 分别表示为新生成的细胞状态、当前细胞状态、隐含层的输出结果. m 为隐含层 LSTM 单元的数量. W_{hf}, W_{hi}, W_{ho}, W_{hc} 和 W_{xf}, W_{xi}, Wxo, $W_{xc} \in R^{m\times m}$是权重参数; b_f, b_i, b_o, $b_c \in R^{m\times d}$为偏差参数. *Sigmoid*(), tanh() 为激活函数, ⊙表示 hadamard 乘积.

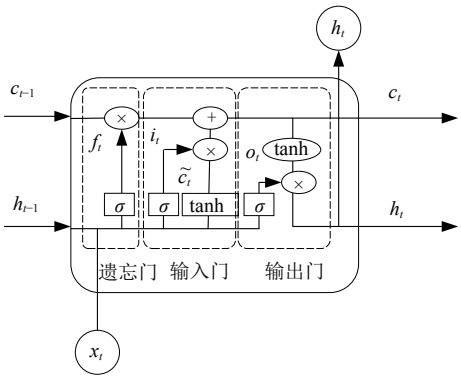


图 1 LSTM 单元结构图

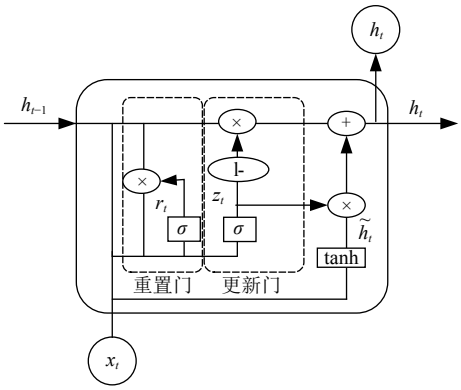


图 2 GRU 单元结构图

1.2 基于 LSTM 网络的 Web 系统寿命预测模型构建过程

基于 LSTM 剩余寿命预测模型由输入层、两层 LSTM 层、池化层、全连接层、输出层构成. 如图 3 所示.

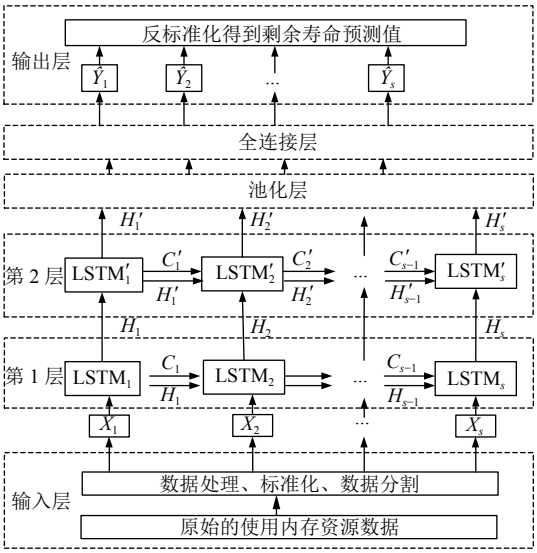


图 3 模型结构图

(1) 收集样本并进行标准化处理

每秒收集一次 Web 软件系统运行过程内存使用量数据, 假设第 n 秒发上了 OOM (Out Of Memory) 故障, 则一共收集了 n 秒的系统运行数据. OOM 指的是当系统因为没有足够的内存为对象分配空间, 就会报出这种故障. 将系统在第 i 秒的内存使用量表示为 mi, 则系统在 i 秒的剩余寿命可表示为$r_i = n - i$, 假设系统运行了 k 秒以后才进入系统老化状态, 则每次运行可得到如下采样数据:

$$(M,R) = \begin{cases} \{(m_i, n-k)\},\ i = 1,2,3,\cdots,k \\ \{(m_i, r_i)\},\ i = k+1,k+2,k+3,\cdots,n \end{cases} \tag{2}$$

得到的样本表示为$(M,R) = \{(m_i, r_i), i = 1,2,3,\cdots,n\}$. 标准化处理后的样本表示为$(M',R') = \{(m_i', r_i'), i = 1,2,3,\cdots,n\}$, 其中$m_i'$和$r_i'$的定义如式 (3) 所示:

$$\begin{cases} m_i' = (m_i - \min\limits_{1\le t\le n}\{m_t\})/(\max\limits_{1\le t\le n}\{m_t\} - \min\limits_{1\le t\le n}\{m_t\}) \\ r_i' = (r_i - \min\limits_{1\le t\le n}\{r_t\})/(\max\limits_{1\le t\le n}\{r_t\} - \min\limits_{1\le t\le n}\{r_t\}) \end{cases} \tag{3}$$

(2) 按照时间步长得到模型的输入

将集合M'按照时间步长w分割为$n-w$个长度为w的时间序列, 用X表示, 对应的实际寿命用Y表示, 如式 (4) 所示:

$$\begin{cases} (X,Y) = \{(X_i, Y_i), i = 1,2,3,\cdots,n-w\} \\ X_i = \{m_i', m_{i+1}', \cdots, m_{i+w}'\}, Y_i = r_{i+w}' \end{cases} \tag{4}$$

(3) 确定 LSTM 的网络结构并初始化网络

确定网络的时间步长、隐含层单元数、每层网络节点的舍弃率以及相应的激活函数、误差的计算方式和权重更新迭代方式. 给定初始权值矩阵, 设置最大迭代次数和最小误差值, 训练网络以更新各项网络参数.

(4) 前向计算

将$X = \{X_1, X_2, \cdots, X_i, \cdots, X_{n-w}\}$输入 LSTM 网络, 根据式 (1) 计算对应的遗忘门、输入门以及输出门的值, 经过 LSTM 层得到的网络输出表示为式 (5):

$$\begin{cases} \hat{Y} = \{\hat{Y}_1, \hat{Y}_2, \cdots, \hat{Y}_i, \cdots, \hat{Y}_{n-w}\} \\ \hat{Y}_i = LSTM_f\{X_i, C_{i-1}, H_{i-w}\} \end{cases} \tag{5}$$

(5) 池化层和全连接层

池化层将经过 LSTM 层处理输出的H_t'平均池化, 得到$\hat{h}_t \in R^m$, 再经过全连接层将$\hat{h}_t$进行特征融合, 得到$\hat{Y}_t \in R$, $\hat{Y}$再经过反标准化处理, 得到最终预测剩余寿命的预测值.

(6) 定义损失函数

损失函数的计算公式如式 (6) 所示, 式中, n 为测试样本总数, $\hat{Y}_i$为剩余寿命的预测值, Y_i为剩余寿命的真实值, w为时间步长的值.

$$loss = \frac{1}{n-w}\sum_{i=1}^{n-w}(\hat{Y}_i - Y_i)^2 \tag{6}$$

(7) 反向误差传播

采用批量梯度下降算法对指标数据进行批次划分, 采用优化算法 (RMSprop、AdaDelta、Adamax、Adam) 对当前的损失函数进行优化, 实时调整网络的偏置和权值, 使网络误差不断减少. 当最小误差和迭代次数满足要求时停止训练模型. 将测试样本数据输入模型得到预测结果.

(8) 评价指标

为评估基于 LSTM 网络的 Web 寿命预测模型的性能, 用平均绝对误差 (Mean Absolute Error, *MAE*) 和均方根误差 (Root Mean Squared Error, *RMSE*) 作为评价指标. 其公式如下:

$$d = \hat{Y} - Y \tag{7}$$

$$MAE = \frac{1}{n}\sum_{d_i \in d}|d_i| \tag{8}$$

$$RMSE=\sqrt{\frac{1}{n}\sum_{d_i\in d}d_i^2} \tag{9}$$

除了 *RMSE* 和 *MAE* 之外, 本文引入另外两个相对评价指标: 指数评价函数 (Scoring Function, *SF*) 和相对准确率 (*Accuracy*). 预测寿命相对剩余寿命偏小和偏大, 分别设置不同的影响因子, 分别记做 a_1 和 a_2, 且 $a_1>a_2$, 这是因为比起 Web 失效后采取抗衰操作, 在失效之前执行抗衰操作带来较小的损失. 本文中影响因子设为 a_1=150, a_2=50.

$$H(d_i)=\begin{cases}e^{-d_i/a_1} & d_i\le 0\\ e^{d_i/a_2} & d_i>0\end{cases} \tag{10}$$

$$SF=\frac{\sum_{d_i\in d}H(d_i)}{n} \tag{11}$$

$$I(d_i)=\begin{cases}1, & -a_1\le d\le a_2\\ 0, & \text{otherwise}\end{cases} \tag{12}$$

$$Accuracy=\frac{\sum_{d_i\in d}I(d_i)}{n}\times 100\% \tag{13}$$

2 实验与结果分析

由于软件老化是一个错误不断累计的过程, Web 软件系统开始出现老化后并不会立刻失效, 需要耗费很长时间才会发现故障. 受 Matias 等人提出的系统化方法的启发[14], 本文仿照工业领域应用的加速理论, 在 Web 软件系统中采用加速内存泄露的方法收集可以反映软件老化情况的指标数据, 基于收集到的时序数据, 构建基于 LSTM 的 Web 软件系统实时剩余寿命预测模型.

2.1 数据收集

为收集因内存泄露导致的软件老化的指标数据, 搭建了一个符合多层 TPC-W 基准测试规范的电子商务网站购书系统. 该软件系统由一个 Web 服务器, 一个数据库服务器和一组模拟的客户端组成. 三者之间的关系如图 4 所示, Web 服务器 Tomcat 关于 JVM 的内存配置参数如表 1 所示.

设定服务端的内存泄漏强度和客户端并发数, 运行该系统直到发生 OOM 故障为一次实验. 当发生 OOM 故障时, 记当前时刻为 *T*, 该时刻系统剩余寿命 *r*=0. 实验过程中每 1 s 收集一次 JVM 的内存使用量, 采集到的样本个数分别为 10 929, 8202. 如图 5 所示, 表示 Web 系统在不同内存泄露强度 (leakage) 和客户端并发数 (concurrency) 下的内存变化趋势图. 表 2 为数据对应的实验参数.

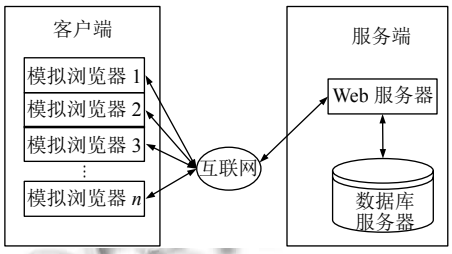
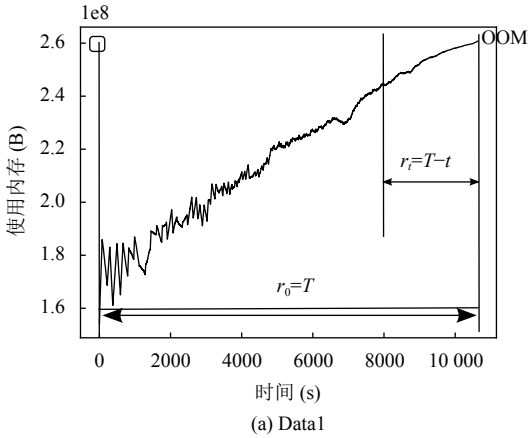


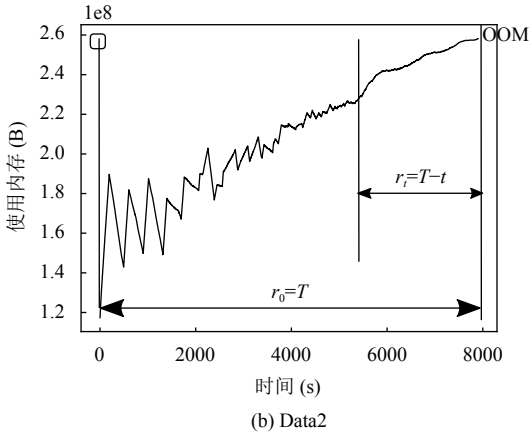
图 4 三者间的工作原理

表 1 Tomcat 关于 JVM 堆内存配置信息

配置	说明
-Xms256M	最小堆内存256 MB
-Xmx256M	最大堆内存256 MB
-Xms60M	年轻代60 MB



(a) Data1



(b) Data2

图 5 JVM 内存使用情况变化趋势图

表 2 实验运行参数

数据	Leakage	Concurrency
Data1	256	200
Data2	476	150

2.2 实验环境

2.3 基于 LSTM 的剩余寿命预测模型

实验环境参数如表 3. 本文使用 Keras 框架搭建并训练 LSTM 网络预测模型, 所使用到的网络主要由循环层 (recurrent) 中的 LSTM 层和全连接层 (dense) 组成. 用 Data1 作为训练集, 其余 3 组作为测试集. 分别比较不同优化算法、隐含层单元数下的模型性能. 最后在相同参数情况下, 与 RNN、GRU 模型结构进行对比.

(1) 寻找合适的隐含层单元数量

用 Data2 做测试集, 用 Adam 算法进行优化, *MAE* 作为评价指标, 尝试不同隐含层单元数量寻找最佳的网络结构. 结果如表 4 所示, 可以看出隐含层单元数量为 120 时, *MAE* 的值最小.

表 3 实验环境参数

项目	参数
CPU	i5-3230M(2.27 GHz)
内存	4 GB
操作系统	Microsoft Windows 7
深度学习框架	Keras
编程语言	Python 2.6

表 4 不同隐含层单元数量的 *MAE*

隐含层单元数量	Data1	Data2
100	88.53	96.69
110	80.37	99.37
120	79.20	85.15
130	91.72	110.85

(2) 优化算法

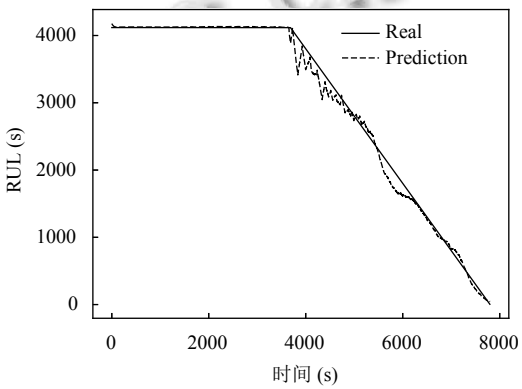
在隐含层单元数为 80 情况下, 用 Data2 做测试集, 采用不同的优化算法训练模型, 用 *MAE* 作为评价指标. 对比不同算法在训练集和测试集上 *MAE* 的大小. 结果如表 5 所示, 可以看出 Adam 的算法的性能优于其它算法.

表 5 不同优化算法的 *MAE* 对比

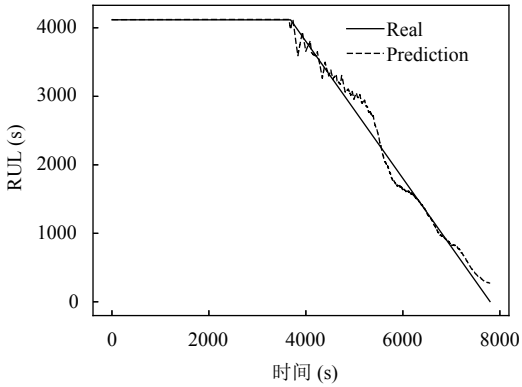
优化算法	Data1	Data2
RMSprop	88.94	94.77
AdaDelta	86.80	91.84
Adamax	90.61	92.45
Adam	75.31	82.33

(3) 与单层 LSTM 网络进行对比

为了验证基于 LSTM 网络的 Web 软件系统剩余寿命预测模型的优势, 用 Adam 优化算法、隐含层单元数量为 120 的参数下, 将双层 LSTM 网络的结构与单层的 LSTM 网络结构做了对比. 从图 6 可以看出, 双层 LSTM 网络结构的拟合效果更好, 用 *MAE* 和 *SF* 作为评价指标, 对比两种网络的性能. 如表 6 所示, 可以看出双层的 LSTM 网络结构相较于单层的 LSTM 网络结构性能有所提升.



(a) 双层 LSTM



(b) 单层 LSTM

图 6 寿命预测结果对比图

表 6 网络性能对比

网络结构	*MAE*	*SF*
双层LSTM	60.10	2.59
LSTM	66.97	2.98

2.4 结果分析

第 2.3 节实验验证了基于神经网络 RNN 以及 LSTM 的 Web 软件系统剩余寿命预测模型的适用性. 而从表 6 可看出, 改进后的基于 LSTM 寿命预测模型对于 Web 软件系统抗衰决策具有良好指导意义. 为了进一步验

证改进后的基于 LSTM 寿命预测模型的预测能力, 表 7 列出了各网络模型在不同指标下的评价指标对比, 表中 *PT* (Prediciton Time) 表示单位样本预测所需要的时间, 单位为 ms, 从表中可以看出, *PT* 值较小, 满足实时预测要求. 从表中可以看出, 改进后 LSTM 寿命预测模型在 3 组实验里的 *MAE*、*RMSE*、*SF* 都是最低, *Accuracy* 最高, 说明该寿命预测相较与 BP 网络和常规循环神经网络而言, 预测精度最高.

表 7 各网络预测精度对比

模型	*MAE*	*RMSE*	*PT*	*SF*	*Accuracy* (%)
双层LSTM	**60.10**	**90.79**	0.20	**2.59**	**78.59**
LSTM	66.97	108.92	0.18	2.98	77.02
GRU	79.96	127.01	0.18	2.98	77.07
RNN	83.68	123.55	0.12	3.47	72.31
BP	84.30	120.71	0.12	7.1	70.27

3 结论

本文通过搭建引入内存泄露的软件老化实验平台收集能够反映软件老化的指标数据, 根据获取到的性能指标数据构建并训练了基于 LSTM 的 Web 软件系统实时剩余寿命预测模型. 通过实验表明该预测模型与 Web 软件系统的寿命趋势一致, 拟合度很高, 能够准确地预测软件的剩余寿命, 为及时采取老化软件的抗衰决策提供保证. 与常规循环神经网络和 BP 网络相比, 满足实时性要求, 预测精度高. 说明 LSTM 网络能够完成 Web 软件系统实时剩余寿命预测的要求.

参考文献

1 Wu XX, Zheng W, Pu MC, *et al*. Invalid bug reports complicate the software aging situation. Software Quality Journal, 2020, 28(1): 195–200. [doi: 10.1007/s11219-019-09481-2]

2 Zheng JJ, Okamura H, Dohi T. A transient interval reliability analysis for software rejuvenation models with phase expansion. Software Quality Journal, 2020, 28(1): 173–194. [doi: 10.1007/s11219-019-09458-1]

3 陈自强. 基于 LSTM 网络的设备健康状况评估与剩余寿命预测方法的研究 [硕士学位论文]. 合肥: 中国科学技术大学, 2019.

4 马波, 翟斌, 彭琦, 等. 基于不同退化阶段状态空间模型及粒子滤波的滚动轴承寿命预测. 北京化工大学学报 (自然科学版), 2017, 44(3): 81–86.

5 高彩霞, 吴彤, 付子义. 线性回归与 EEMD 的滚动轴承剩余寿命预测. 机械科学与技术, 2019, 38(10): 1589–1597.

6 裴洪, 胡昌华, 司小胜, 等. 基于机器学习的设备剩余寿命预测方法综述. 机械工程学报, 2019, 55(8): 1–13.

7 闫楚良, 郝云霄, 刘克格. 基于遗传算法优化的 BP 神经网络的材料疲劳寿命预测. 吉林大学学报 (工学版), 2014, 44(6): 1710–1715.

8 王佳炜, 王召斌, 黄周霖. 果蝇算法优化的 BP 神经网络在电磁继电器贮存寿命预测中的应用. 电器与能效管理技术, 2019, (2): 19–24. [doi: 10.3969/j.issn.1001-5531.2019.02.005]

9 Heimes FO. Recurrent neural networks for remaining useful life estimation. Proceedings of 2008 International Conference on Prognostics and Health Management. Denver, CO, USA. 2008. 1–6. [doi: 10.1109/PHM.2008.4711422]

10 Wu YT, Yuan M, Dong SP, *et al*. Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. Neurocomputing, 2018, 275: 167–179. [doi: 10.1016/j.neucom.2017.05.063]

11 胡天中, 余建波. 基于多尺度分解和深度学习的锂电池寿命预测. 浙江大学学报 (工学版), 2019, 53(10): 1852–1864. [doi: 10.3785/j.issn.1008-973X.2019.10.002]

12 王奉涛, 刘晓飞, 邓刚, 等. 基于长短期记忆网络的滚动轴承寿命预测方法. 振动、测试与诊断, 2020, 40(2): 303–309. [doi: 10.16450/j.cnki.issn.1004-6801.2020.02.013]

13 Sak H, Senior A, Beaufays F. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. arXiv: 1402.1128, 2014.

14 Zhao J, Jin YL, Trivedi KS, *et al*. Software rejuvenation scheduling using accelerated life testing. ACM Journal on Emerging Technologies in Computing Systems, 2014, 10(1): 9. [doi: 10.1145/2539118]