A Systematic Mapping on Software Aging and Rejuvenation Prediction Models in Edge, Fog and Cloud Architectures

Paulo do Amaral Costa[1][a], Edward David Moreno Ordonez[2][b] and Jean Carlos Teixeira de Araujo[3][c]

[1]*Coordenadoria de Análise e Desenvolvimento de Sistemas (CADS), Instituto Federal de Sergipe, Aracaju, Brazil*
[2]*Departamento de Computação, Universidade Federal de Sergipe, São Cristovão, Brazil*
[3]*Departamento de Informática, Universidade Federal do Agreste de Pernambuco, Garanhuns, Brazil*

Keywords: Software Aging, Software Rejuvenation, SAR, Predict Model, Cloud Computing, Edge Computing, Fog Computing, Systematic Mapping Literature, SML.

Abstract: This article presents a Systematic Literature Mapping (SLM), related to software aging and rejuvenation prediction models. The study highlights the importance of these models, due to the high cost of software or service downtime in IT datacenter environments. To mitigate this impact and seek greater reliability and availability of applications and services, software aging prediction and proactive rejuvenation are significant research topics in the area of Software Aging and Rejuvenation (SAR). Costs are potentially higher when rejuvenation actions are not scheduled. Various prediction models have been proposed for over twenty-five years, with the aim of helping to find the ideal moment for rejuvenation, in order to optimize the availability of services, reduce downtime and, consequently, the cost. However, the scope of this study was limited to a survey of the last fifteen years of models with a measurement-based prediction strategy. These models involve monitoring and collecting data on resource consumption over time, from a running computer system. The collected data is used to adjust and validate the model, allowing the prediction of the precise moment of the aging phenomenon and the consequent rejuvenation action of the software. In addition to providing a baseline from the compiled prediction models, identifying gaps that could encourage future research, particularly in the areas of machine learning or deep learning, the research also contributed to clarifying that hybrid algorithms based on Long Short-Term Memory (LSTM) are currently situated at the highest level of prediction models for software aging, with recent highlights for two variants: the Gated Recurrent Unit (GRU) and the Bidirectional Long Short Term Memory (BiLSTM). Objectively, in response to the research questions, the article also contributes by presenting, through tables and graphs, trends and consensus among researchers regarding the evolution of prediction models.

1 INTRODUCTION

According to recent research (Yue et al., 2020), the annual cost of system downtime in Information Technology (IT) environments is approximately US$ 26.5 billion. Reducing this costly impact and maximizing the reliability and availability of applications and services justify efforts to predict software aging and proactive and planned rejuvenation action. These topics are highly relevant and of great interest to researchers in the field of Software Aging and Rejuvenation (SAR) (Araujo et al., 2011; Cotroneo et al., 2014; Di Sanzo et al., 2015; Avresky et al., 2017; Umesh et al., 2017; Liu et al., 2019; Wang and Liu, 2020; Tan and Liu, 2021; Oliveira et al., 2021).

This article presents a Systematic Literature Mapping (SLM) conducted to address research questions specifically related to software aging and rejuvenation prediction models in edge, fog and cloud computing environments.

The SLM collected evidence on work on software aging prediction systems in four scientific literature databases, Figure 1, over the last 15 years (period from 2009 to 2023).

The retrieved documents answered several research questions, however the SLM was guided by

[a] https://orcid.org/0009-0004-4252-2963
[b] https://orcid.org/0000-0002-4786-9243
[c] https://orcid.org/0000-0002-1688-4782

Costa, P., Ordonez, E. and Teixeira de Araujo, J.
A Systematic Mapping on Software Aging and Rejuvenation Prediction Models in Edge, Fog and Cloud Architectures.
DOI: 10.5220/0012633900003690
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 26th International Conference on Enterprise Information Systems (ICEIS 2024) - Volume 1*, pages 933-942
ISBN: 978-989-758-692-7; ISSN: 2184-4992
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

the following main question: "What predictive software aging models have been proposed in Edge, Fog and Cloud Computing environments?"

The literature search revealed that algorithms based on decision trees and time series were the most cited in the selected studies. However, in the last years, the Long Short-Term Memory (LSTM) convolutional neural network, the present state of the art in temporal series prediction, demonstrated the most promising results when applied to software aging prediction.

LSTM is a deep learning algorithm often applied to sequential data sets, such as time series, and as software aging is something that occurs over time, its use is appropriate. The research contributed to clarify that the proposed hybrid models based on LSTM (Battisti et al., 2022; Shi et al., 2023; Jia et al., 2023) are at the most evolutionary level of prediction models for software aging, with more recent highlights for two variants: the Gated Recurrent Unit (GRU) and the Bidirectional Long Short - Term Memory (BiLSTM).

The GRU has a simpler structure that provides better performance than the LSTM model and with slightly greater or equivalent accuracy. BiLSTM or Bidirectional LSTM, composed of two LSTMs, has the ability to analyze future and past timestamp inputs, with its backward and forward direction layers, being a more powerful algorithm than the original unidirectional LSTM and, therefore, the prediction of this model tends to be better.

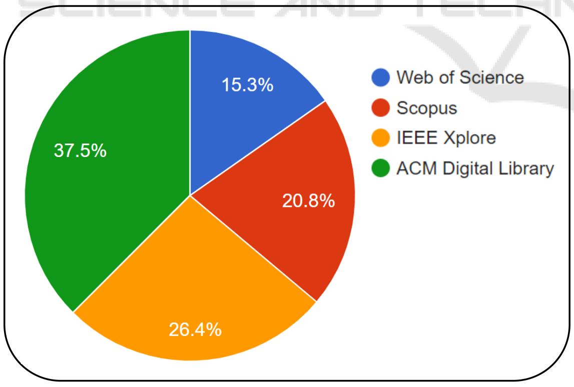

Figure 1: Articles selected per source.

The results of this study, based on the compiled models, also contribute to identifying gaps that can stimulate future research proposals, especially those based on deep learning, whether or not considering the interdependence of aging indicator variables.

Furthermore, the remaining sections of the article are organized as follows. Section 2 discusses the definitions of software aging and rejuvenation, as well as prediction strategies. Section 3 defines the SLM method used in this study. Section 4 summarizes the results obtained from the analysis of selected articles through tables and graphs. Section 5 addresses threats to the validity of the study. Finally, Section 6 presents the conclusions regarding this study.

2 THEORETICAL FOUNDATION

2.1 Software Aging and Rejuvenation

Software aging refers to an increase in system performance degradation that occurs over its operational lifespan. It is a cumulative process caused by software errors that gradually deplete system resources, without immediately resulting in a failure (Grottke et al., 2008; Tang et al., 2020; Oliveira et al., 2021).

The symptoms of software aging are perceived or evidenced through real-time monitoring of system resources, which are commonly referred to as "aging indicators" (Grottke et al., 2008; Tang et al., 2020). The main indicators include primary memory, swap virtual memory, CPU utilization, and secondary memory usage.

Software rejuvenation is a proactive technique that aims to alleviate the effects of software aging (Cotroneo et al., 2014). Its goal is to clean up the internal degradation state of the system, preventing more severe failures such as crashes or malfunctions, and restoring its performance (Sudhakar et al., 2014; Cotroneo et al., 2014; Umesh et al., 2017).

The most common rejuvenation techniques involve restarting a specific software component or rebooting the entire system (Araujo et al., 2011; Araujo et al., 2014; Cotroneo et al., 2014; Battisti et al., 2022). In the latter case, for example, in the case of swap space, it is not cleared after restarting the aging-causing application but only after a complete restart of the operating system (Simeonov and Avresky, 2010).

One of the typical challenges of a rejuvenation approach is the software downtime, as the service or application becomes unavailable during the entire restart process (Araujo et al., 2011; Wang and Liu, 2020; Oliveira et al., 2021; Battisti et al., 2022). Therefore, a software rejuvenation strategy should be carefully planned to avoid more severe failures and optimize downtime (Cotroneo et al., 2014; Tan and Liu, 2021).

The costs of downtime are higher when it is not scheduled (Cotroneo et al., 2014), and predicting the failure time of aging and planning when to perform the rejuvenation action are highly relevant in the SAR field (Cotroneo et al., 2014; Liu et al., 2019; Wang and Liu, 2020). These efforts aim to improve service availability and reduce overall downtime.

2.2 Prediction Strategies

Currently, there are three main categories of proactive rejuvenation prediction strategies aimed at mitigating the effects of software aging (Simeonov and Avresky, 2010; Cotroneo et al., 2014; Liu et al., 2019). These categories are: a) model-based strategy, b) measurement-based strategy, and c) hybrid strategy.

However, the scope of the compilation in this study is limited to the second type of strategy. For the measurement-based strategy, the first step involves monitoring and collecting data on system resource consumption during operation (Di Sanzo et al., 2015). In a subsequent stage, time series analysis (Araujo et al., 2011; Araujo et al., 2014; Umesh et al., 2017), as well as machine learning algorithms (Simeonov and Avresky, 2010; Sudhakar et al., 2014; Di Sanzo et al., 2015; Avresky et al., 2017), including their subfield of deep learning (Yue et al., 2020; Tan and Liu, 2021), or deep learning in hybrid approach (Liu et al., 2019; Battisti et al., 2022; Shi et al., 2023; Jia et al., 2023), are used to process the collected data. The prediction model is then trained to accurately predict the occurrence of software aging as precisely as possible (Tan and Liu, 2021).

When it comes to software aging and rejuvenation for cloud services, prediction methods based on measurement strategies tend to be more promising (Liu et al., 2019).

3 METHODOLOGY

A Systematic Literature Mapping (SLM), among other characteristics, aims to provide an overview of a specific research area, identify utilized techniques, and facilitate the discovery of gaps that can support future investigations through its systematic methodology (Kitchenham et al., 2011; Petersen et al., 2015). This article is the result of an SLM focused on primary studies published on software aging prediction systems.

3.1 Research Questions

The research adopted the systematic PICOC analysis (Population, Intervention, Comparison, Results, Context). It is the starting point for developing the research questions and search string. PICOC is a procedure used to describe the five elements related to the identified problem and to structure the main question of a research.

Following the protocol recommended (Kitchenham et al., 2011; Petersen et al., 2015), when formulating the PICOC framework, it was obtained:

- Population (Group from which evidence is collected). Software aging and rejuvenation prediction systems;
- Intervention (Action applied in the empirical study). Synthetic analysis of prediction models;
- Comparison (Parameters with which the intervention is compared). Regression model, classification model;
- Outcomes (desired outcomes). Prediction techniques or algorithms, machine learning and deep learning algorithms, aging indicators and strategies, environments;
- Context (Segment in which the population is located). Cloud, Fog and Edge Architectures.

It is clear that research questions must take into account the following points of view: Population, Intervention and Results, the objective of which is to answer the components: Comparison and Context. Based on this principle and the research interest defined through PICOC analysis, it became possible to formulate the main research question (MQ) that guided the present study: “What predictive software aging models have been proposed in Edge, Fog and Cloud Computing environments?”

To answer the research interest and provide a broader scope, the MQ was broken down and dissected into the following research questions (RQs):

- RQ1 - Which techniques or algorithms for software aging prediction were used in the experiments?
- RQ2 - Which aging indicators were analyzed in the experiments?
- RQ3 - What was the method used to obtain the aging indicator data in the experiments?
- RQ4 - What aging strategies were adopted in the experiments?
- RQ5 - Which virtualization environments were used in data acquisition or in the aging experiments?
- RQ6 - Which network architecture has the highest incidence of studies on software aging prediction systems?

3.2 Search String

Once the research questions were formulated, the next step was to create the search string, which enabled the retrieval of evidence in the literature. The terms (synonymous descriptors or key words) identified and related to each of the components of the PICO strategy and the use of Boolean operators (AND, OR, NOT) for each of the four components of the strategy, interrelating the sentences (P) AND (I) AND (C) AND (O), provided the construction of the search string, as shown in Figure 2.

```
("software aging" OR "sofware rejuvenation")
AND
("edge computing" OR "fog computing" OR "cloud computing")
AND
(algorithm OR "machine learning" OR "deep learning" OR "neural network"
OR ANN OR CNN OR RNN OR LSTM OR ConvNET OR "time series")
```

Figure 2: Query string.

The search string was submitted directly to four literature databases: ACM Digital Library; IEEE XPlore; Scopus; Web of Science. A total of seventy two articles were retrieved, as shown in Figure 3.

After thorough reading, only seventeen articles were accepted for data extraction. Sixteen articles were duplicates and thirty-nine articles were rejected.

3.3 Inclusion and Exclusion Criteria

Inclusion and exclusion criteria are used to exclude studies that are not relevant to answering the research questions (Petersen et al., 2015). The inclusion criteria (IC) adopted were:

- IC1 - The publication's objective must be related to software aging and rejuvenation prediction.
- IC2 - The publication must describe a model, method, or technique for software aging prediction.

The exclusion criteria (EC) adopted were:

- EC1 - The publication does not have an abstract.
- EC2 - The study is published only as an abstract.
- EC3 - The publication is not written in English.
- EC4 - The publication is an older version of another already considered.
- EC5 - The publication is not a primary study.
- EC6 - The publication does not have full-text availability.
- EC7 - The publication does not pertain to software aging and rejuvenation prediction systems.
- EC8 - Access to the publication was not possible.
- EC9 - The document does not address measurement-based prediction strategy.
- EC10 - The document has not been published in the last 15 years.

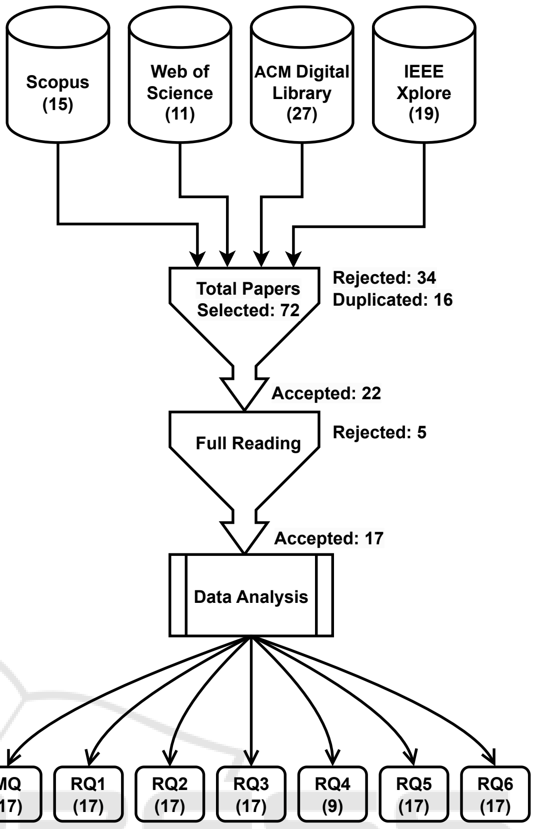

Figure 3: Document selection and data extraction process.

4 RESULTS AND DISCUSSION

4.1 Preliminary Results

In this preamble, additional secondary information is presented, however it is very important for understanding this mapping.

Seventeen articles related to predictive models of software aging and rejuvenation were selected.

Figure 4 presents a comparison over the years between the documents initially selected and the documents accepted after full reading. Both curves are clearly multimodal and point to the years 2014, 2016, 2020 and 2021 with the highest volumes of selected publications and the years 2014, 2017, 2020, 2021 and 2023 with the most articles finally accepted.

The graph in Figure 5 crosses the number of articles selected and articles finally accepted per source. Highlights include the Web of Science, IEEE Xplorer and Scopus databases, which together accounted for 94.1% of the articles finally accepted. The ACM Digital Library source, despite contributing the largest volume of selected articles (37.5%, Figure 1), had only 1 article finally accepted.

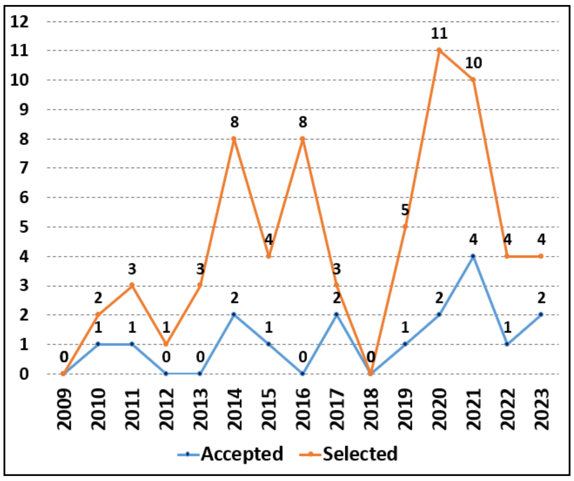

Figure 4: Number of articles published per year according to the search string.

The same publication may involve affiliations of researchers from different countries. Therefore, Asia, Europe, South America and North America constitute the continents of the authors of the 17 accepted publications.

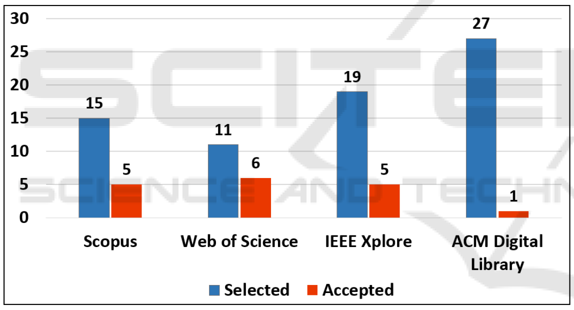

Figure 5: Number of articles selected and articles accepted per source.

Figure 6 highlights the researchers from China, Germany and Brazil, with 8, 4 and 4 participations in accepted publications on software aging prediction models.

4.2 Answers to Research Questions

It is worth noting that no publications with experiments specifically related to the Fog and Edge Computing segments were found using the search string.

Only eight articles did not answer research question number 4. All other RQs were answered in full by all accepted documents (see Figure 3).

The answers to all investigated questions will be presented sequentially through tables, including the main research question.

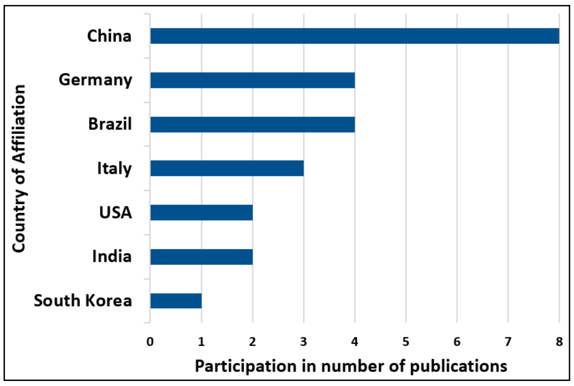

Figure 6: Participation of the country of affiliation in publications.

4.2.1 MQ - What Predictive Software Aging Models Have Been Proposed in Edge, Fog and Cloud Computing Environments?

Table 1 shows that around 88% of the works propose prediction models based on regression algorithms (15 references), in order to estimate Time-to-Failure (TTF) or Remaining Time-To-Failure (RTTF), which is a prognosis of the remaining useful life of the system for performance degradation or sudden downtime, resulting from resource exhaustion and/or accumulations of numerical errors or aging-related bugs, caused by continuous execution over a long period.

It is observed that only 2 works refer to classification algorithms. Simeonov's work (Simeonov and Avresky, 2010) uses a machine learning algorithm based on a decision tree for a binary "yes/no" classification. Machine learning was developed from historical data. If the value is "yes", the corresponding machine needs to be rejuvenated.

Yue (Yue et al., 2020) proposes a deep learning method to predict the phenomenon of microservice aging and a rejuvenation policy. The network architecture is the CNN+LSTM model, where, in the network input layer, each neuron represents a microservice. The output layer activated by the Softmax function is a vector of 10 categories, where each class respectively represents the probability of quality of service (QoS) violations. When the QoS violation probability value is high and CPU, memory, and disk usage exceeds standard limits, the microservice is considered obsolete and is rejuvenated.

4.2.2 RQ1 - Which Techniques or Algorithms for Software Aging Prediction Were Used in the Experiments?

In order to predict aging, data from system resources are monitored over time, generating a typical dataset

Table 1: Types of prediction models used in experiments.

Prediction Model Type	Paper (Reference)
Regression[a]	(Araujo et al., 2011), (Sudhakar et al., 2014), (Araujo et al., 2014), (Di Sanzo et al., 2015), (Avresky et al., 2017), (Umesh et al., 2017), (Liu et al., 2019), (Tang et al., 2020), (Tan and Liu, 2021), (Di Sanzo et al., 2021), (Meng et al., 2021a), (Meng et al., 2021b), (Battisti et al., 2022), (Shi et al., 2023), (Jia et al., 2023)
Classification	(Simeonov and Avresky, 2010), (Yue et al., 2020)

[a] Remaining Time To Failure (RTTF) prediction models regression-based.

as a function of time, that is, a time series. Long Short-Term Memory (LSTM) neural networks and their variants are the current state of the art in time series forecasting.

Models based on the Conv-LSTM network have demonstrated greater accuracy in predicting TTF and are cited in Table 2 in works from 2019 onwards.

The Table 3 consolidates the categories shown in Table 2. There is a quantitative balance between the techniques used. Classic machine learning algorithms appear in 6 references, but the most promising recent state of the art points to hybrid time series analysis models, based on deep learning (6 references).

The Table 3 consolidates the categories shown in Table 2.

4.2.3 RQ2 - What Aging Indicators Were Analyzed in the Experiments?

Most aging is due to computation, network, cache, RAM and disk contention (Yue et al., 2020), therefore, due to their possible exhaustion, these resources are used as indicators of aging. Typically failures due to the aging phenomenon occur when free memory is exhausted, since CPU is a more compressible resource. However, among the various aging indicators pointed out in Table 4, the decline in free RAM memory (14 references) and CPU consumption (14 references) are mentioned more than the others.

Table 2: Algorithms or main techniques that were used in the experiments.

Technique or Algorithm	Paper (Reference)
Time Series Algorithm	(Araujo et al., 2011), (Araujo et al., 2014), (Umesh et al., 2017), (Tang et al., 2020), (Meng et al., 2021a)
Decision Tree	(Simeonov and Avresky, 2010), (Di Sanzo et al., 2015), (Avresky et al., 2017), (Di Sanzo et al., 2021)
CNN+LSTM+Time Series Algorithm	(Liu et al., 2019), (Battisti et al., 2022)
CNN+LSTM	(Yue et al., 2020), (Tan and Liu, 2021)
VMD+ARIMA+ BiLSTM	(Shi et al., 2023)
STL+GRU (DGRU)	(Jia et al., 2023)
Lasso Regression as a Predictor	(Di Sanzo et al., 2021)
Support Vector Machine (SVM)	(Di Sanzo et al., 2021)
ARIMA+ANN (MLP Feedfoward)	(Meng et al., 2021b)
ANN (MLP Feedfoward)	(Sudhakar et al., 2014)

Table 3: Type of main approach used in models.

Approach	Paper (Reference)
Machine Learning	(Simeonov and Avresky, 2010), (Sudhakar et al., 2014), (Di Sanzo et al., 2015), (Avresky et al., 2017), (Di Sanzo et al., 2021), (Meng et al., 2021b)
Deep Learning	(Liu et al., 2019), (Yue et al., 2020), (Tan and Liu, 2021), (Battisti et al., 2022), (Shi et al., 2023), (Jia et al., 2023)
Classic Time Series Analysis	(Araujo et al., 2011), (Araujo et al., 2014), (Umesh et al., 2017), (Tang et al., 2020), (Meng et al., 2021a)

Table 4: Aging indicator resources analyzed in the experiments.

Aging Indicators	Paper (Reference)
RAM	(Simeonov and Avresky, 2010), (Araujo et al., 2011), (Araujo et al., 2014), (Sudhakar et al., 2014), (Di Sanzo et al., 2015), (Avresky et al., 2017), (Umesh et al., 2017), (Yue et al., 2020), (Tang et al., 2020), (Tan and Liu, 2021), (Di Sanzo et al., 2021), (Meng et al., 2021a), (Battisti et al., 2022), (Jia et al., 2023)
CPU	(Simeonov and Avresky, 2010), (Araujo et al., 2014), (Sudhakar et al., 2014), (Di Sanzo et al., 2015), (Avresky et al., 2017), (Umesh et al., 2017), (Liu et al., 2019), (Yue et al., 2020), (Tan and Liu, 2021), (Di Sanzo et al., 2021), (Meng et al., 2021a), (Meng et al., 2021b), (Battisti et al., 2022), (Shi et al., 2023)
Swap	(Simeonov and Avresky, 2010), (Araujo et al., 2011), (Araujo et al., 2014), (Sudhakar et al., 2014), (Di Sanzo et al., 2015), (Avresky et al., 2017), (Di Sanzo et al., 2021), (Battisti et al., 2022)
Response time for requests or services or Throughput or Network transfer rate	(Di Sanzo et al., 2015), (Yue et al., 2020), (Tang et al., 2020), (Tan and Liu, 2021), (Meng et al., 2021b)
Number of active or zombie processes or threads	(Araujo et al., 2011), (Sudhakar et al., 2014), (Di Sanzo et al., 2015)
Disk usage	(Yue et al., 2020), (Battisti et al., 2022)
Number of TCP connections	(Sudhakar et al., 2014)

4.2.4 RQ3 - What was the Method Used to Obtain the Aging Indicator Data in the Experiments?

Table 5 summarizes the methods for obtaining aging indicator data used in the experiments. Battisti (Battisti et al., 2022) reports having used a private dataset obtained from third parties, generated by an experiment that evaluated the effects of software aging on a container-based virtualization platform (Oliveira et al., 2020). Liu (Liu et al., 2019) validated the prediction model with a public dataset obtained from the Google Cluster Workload.

To validate two experiment scenarios of the software aging prediction model, Tan (Tan and Liu, 2021) used public datasets from Google and Alibaba Cloud Cluster. The Google Cluster Workload dataset referred to the resource usage of 1,600 machines over a 1-month interval, while the Alibaba cluster dataset referred to the resource usage of 4,000 machines over 8 days.

Shi (Shi et al., 2023) also used a dataset published by Alibaba in his experiments and Jia (Jia et al., 2023) also used dataset published by Google Cluster Workload. The remaining 12 articles claimed to have used real data sets collected in the experiments.

Table 5: Method of obtaining data.

Method	Paper (Reference)
Monitoring and data collect data over time	(Simeonov and Avresky, 2010), (Araujo et al., 2011), (Araujo et al., 2014), (Sudhakar et al., 2014), (Di Sanzo et al., 2015), (Avresky et al., 2017), (Umesh et al., 2017), (Yue et al., 2020), (Tang et al., 2020), (Di Sanzo et al., 2021), (Meng et al., 2021a), (Meng et al., 2021b)
Public or private dataset	(Liu et al., 2019), (Tan and Liu, 2021), (Battisti et al., 2022), (Shi et al., 2023), (Jia et al., 2023)

4.2.5 RQ4 - What Aging Strategies Were Adopted in the Experiments?

It is proven effective to stress a system or software, aiming to accelerate the manifestation of bugs or resource leaks (Cotroneo et al., 2014). Therefore, it can be attributed that aging is caused by a large number of repeated executions that would lead to the accumulation of system resource leaks (Yue et al., 2020).

Thus, an aging indicator metric may show a clear upward trend over time, suggesting the occurrence of the aging phenomenon after a long period of execution.

Table 6 demonstrates the prevalence of “client requests” to servers over the other types of workload strategies used in the experiments.

Table 6: Aging strategies analyzed in experiments.

Aging Strategies (Stress load or Workload)	Paper (Reference)
Client requests or service requests	(Di Sanzo et al., 2015), (Yue et al., 2020), (Meng et al., 2021a), (Meng et al., 2021b)
Injection of memory leaks and unfinished threads	(Di Sanzo et al., 2015), (Di Sanzo et al., 2021)
Repeated operations of instantiating, restarting and terminating VMs	(Araujo et al., 2011), (Araujo et al., 2014)
Cloud task requests	(Tan and Liu, 2021)
Injection of memory leaks	(Avresky et al., 2017)
Attaching and detaching storage volumes	(Araujo et al., 2014)

4.2.6 RQ5 - Which Virtualization Environments Were Used in Data Acquisition or in the Aging Experiments?

Virtualization technology can divide the physical server into multiple virtual machines (VMs) and cloud-based software services benefit from virtualization technology (Tan and Liu, 2021).

On the other hand, containers, in addition to promoting the process of fair and efficient allocation of physical resources between virtual machines (Yue et al., 2020), are a form of lightweight virtualization also widely used to provide services in the cloud, being subjected to a long cycle of operational life and intense workload (Oliveira et al., 2020).

These environments are subject to the phenomenon of software aging and Table 7 shows the virtualization environments used by software aging prediction models, regardless of whether they reside in the cloud.

The virtualization option using VMs appeared in 15 references, while the use of containers was mentioned in only 2 references.

Table 7: Virtualization environments used in obtaining the data or experiments.

Environment	Paper (Reference)
Virtual Machines (VMs)	(Simeonov and Avresky, 2010), (Araujo et al., 2011), (Araujo et al., 2014), (Sudhakar et al., 2014), (Di Sanzo et al., 2015), (Avresky et al., 2017), (Umesh et al., 2017), (Liu et al., 2019), (Tang et al., 2020), (Tan and Liu, 2021), (Di Sanzo et al., 2021), (Meng et al., 2021a), (Meng et al., 2021b), (Shi et al., 2023), (Jia et al., 2023)
Containers	(Yue et al., 2020), (Battisti et al., 2022)

4.2.7 RQ6 - Which Network Architecture has the Highest Incidence of Studies on Software Aging Prediction Systems?

Table 8: Network architecture used in the aging experiments or in obtaining the dataset.

Architecture	Paper (Reference)
Public Cloud Computing	(Sudhakar et al., 2014), (Di Sanzo et al., 2015), (Avresky et al., 2017), (Liu et al., 2019), (Yue et al., 2020), (Tan and Liu, 2021), (Shi et al., 2023), (Jia et al., 2023)
Private Cloud Computing or LAN[a]	(Simeonov and Avresky, 2010), (Araujo et al., 2011), (Araujo et al., 2014), (Umesh et al., 2017), (Tang et al., 2020)
Cloud Test Environment, Simulator or Framework	(Araujo et al., 2011), (Tan and Liu, 2021), (Battisti et al., 2022), (Meng et al., 2021a), (Meng et al., 2021b)
Hybrid Cloud (Public and Private)	(Avresky et al., 2017), (Di Sanzo et al., 2021)

[a] Local Area Network.

Operating environment with continuous and long-running processing is inherent to cloud computing architecture, which is easily prone to producing software aging (Liu et al., 2019). Table 8 confirms this environment as the target of most data collection or experiments carried out by researchers.

5 THREATS TO VALIDITY

This section presents the main threats to validity regarding the conclusions drawn in this secondary study, namely:

- Although the SLM protocol was initially and impartially defined, it is impossible to rule out the threat to the quality of the articles that were included, as the studies were selected without assigning scores.
- The consultation carried out in only four scientific literature databases limited the scope of the search, excluding possible relevant documents existing in other databases, which can be classified as a threat to the validity of the work.
- Some inclusion and exclusion criteria may have eliminated relevant articles and this certainly also constitutes a threat to the validity of the present study.

However, due to the rigorous adoption of the SLM method, it is perfectly possible that other researchers can replicate the present study and, consequently, come to confirm similar or equivalent results.

6 CONCLUSIONS

Several techniques and algorithms for software aging prediction were used in the tabulated experiments. The specific techniques and algorithms employed varied among the selected articles. The main approaches proposed in the studies were:

- Time Series Analysis. This technique involves analyzing historical data on system resource consumption over time to identify patterns and trends that indicate software aging.
- Machine Learning Algorithms. Various machine learning algorithms such as regression, decision trees, support vector machines (SVM) and neural networks have been used for software aging prediction. These algorithms are trained on historical data to predict the occurrence of software aging based on input features.
- Deep Learning Algorithms. Deep learning techniques, particularly convolutional neural networks and LSTM, have been employed to predict software aging. These algorithms can automatically learn complex patterns and relationships from large volumes of data, including from transfer learning, increasing prediction accuracy.
- Statistical Models. Some studies used statistical models, such as ARIMA (Autoregressive Integrated Moving Average) and others, to analyze time series data and make predictions about software aging. They have been gathered in the categories Time Series Algorithms and Classic Time Series Analysis.
- Hybrid Approaches. Other papers have proposed hybrid approaches that combine two or more techniques or algorithms, with the aim of chaining and leveraging their respective strengths in order to improve forecasting accuracy, given that single aging prediction models typically perform poorly and produce less accurate results (Jia et al., 2023). They were grouped into CNN+LSTM, CNN+LSTM+Time Series Algorithm, VMD+ARIMA+BiLSTM, STL+GRU categories. In some articles the Lasso Regression algorithm was not configured as a linear regressive prediction model. It acted together only in data regularization for the selection of aging indicators and, therefore, not as an integral part of a hybrid prediction model.

It is important to note that the specific techniques and algorithms used varied between the selected studies, and more details can be found in the individual primary articles.

The prediction of software aging is one of the most important issues in the field of Software Aging and Rejuvenation (SAR). This article resulted from an SLM on software aging prediction systems, which may serve as a reference point for future research on this highly relevant topic. It provides insights into the trends and consensus among researchers regarding the current state of the art.

REFERENCES

Araujo, J., Matos, R., Alves, V., Maciel, P., De Souza, F. V., Matias Jr., R., and Trivedi, K. S. (2014). Software aging in the eucalyptus cloud computing infrastructure: Characterization and rejuvenation. *ACM Journal on Emerging Technologies in Computing Systems*, 10(1). Cited by: 49.

Araujo, J., Matos, R., Maciel, P., Vieira, F., Matias, R., and Trivedi, K. S. (2011). Software rejuvenation in eucalyptus cloud computing infrastructure: A method based on time series forecasting and multiple thresholds. In *2011 IEEE Third International Workshop on Software Aging and Rejuvenation*, pages 38–43.

Avresky, D. R., Pellegrini, A., and Di Sanzo, P. (2017). Machine learning-based management of cloud applications in hybrid clouds: A hadoop case study. In *2017 IEEE 16th International Symposium on Network Computing and Applications (NCA)*, pages 1–5.

Battisti, F., Silva, A., Pereira, L., Carvalho, T., Araujo, J., Choi, E., Nguyen, T. A., and Min, D. (2022). hlstm-aging: A hybrid lstm model for software aging forecast. *Applied Sciences*, 12(13).

Cotroneo, D., Natella, R., Pietrantuono, R., and Russo, S. (2014). A survey of software aging and rejuvenation studies. *J. Emerg. Technol. Comput. Syst.*, 10(1).

Di Sanzo, P., Avresky, D. R., and Pellegrini, A. (2021). Autonomic rejuvenation of cloud applications as a countermeasure to software anomalies. *SOFTWARE-PRACTICE & EXPERIENCE*, 51(1, SI):46–71.

Di Sanzo, P., Pellegrini, A., and Avresky, D. R. (2015). Machine learning for achieving self-* properties and seamless execution of applications in the cloud. In *2015 IEEE Fourth Symposium on Network Cloud Computing and Applications (NCCA)*, pages 51–58.

Grottke, M., Matias, R., and Trivedi, K. S. (2008). The fundamentals of software aging. In *2008 IEEE International Conference on Software Reliability Engineering Workshops (ISSRE Wksp)*, pages 1–6.

Jia, K., Yu, X., Zhang, C., Hu, W., Zhao, D., and Xiang, J. (2023). Software aging prediction for cloud services using a gate recurrent unit neural network model based on time series decomposition. *IEEE Transactions on Emerging Topics in Computing*, 11(3):580–593.

Kitchenham, B. A., Budgen, D., and Pearl Brereton, O. (2011). Using mapping studies as the basis for further research – a participant-observer case study. *Information and Software Technology*, 53(6):638–651. Special Section: Best papers from the APSEC.

Liu, J., Tan, X., and Wang, Y. (2019). Cssap: Software aging prediction for cloud services based on arima-lstm hybrid model. In *2019 IEEE International Conference on Web Services (ICWS)*, pages 283–290.

Meng, H., Shi, Y., Qu, Y., Li, J., and Liu, J. (2021a). Arima-based aging prediction method for cloud server system. volume 1043.

Meng, H., Tong, X., Shi, Y., Zhu, L., Feng, K., and Hei, X. (2021b). Cloud server aging prediction method based on hybrid model of auto-regressive integrated moving average and recurrent neural network;. *Tongxin Xuebao/Journal on Communications*, 42(1):163 – 171.

Oliveira, F., Araujo, J., Matos, R., Lins, L., Rodrigues, A., and Maciel, P. (2020). Experimental evaluation of software aging effects in a container-based virtualization platform. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 414–419.

Oliveira, F., Araujo, J., Matos, R., and Maciel, P. (2021). Software aging in container-based virtualization:an experimental analysis on docker platform. In *2021 16th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–7.

Petersen, K., Vakkalanka, S., and Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64:1–18.

Shi, F., Yuan, Z., Wang, M., and Cui, J. (2023). Research on cloud platform software aging prediction method based on vmd-arima-bilstm combined model. *Integrated Ferroelectrics*, 237(1):297 – 309.

Simeonov, D. and Avresky, D. R. (2010). Proactive software rejuvenation based on machine learning techniques. In Avresky, D., Diaz, M., Bode, A., Ciciani, B., and Dekel, E., editors, *CLOUD COMPUTING*, volume 34 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics, and Telecommunications Engineering*, pages 186–200.

Sudhakar, C., Shah, I., and Ramesh, T. (2014). Software rejuvenation in cloud systems using neural networks. In *2014 International Conference on Parallel, Distributed and Grid Computing*, pages 230–233.

Tan, X. and Liu, J. (2021). Aclm:software aging prediction of virtual machine monitor based on attention mechanism of cnn-lstm model. In *2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS)*, pages 759–767.

Tang, M., Zhang, P., Sun, H., and Zhang, L. (2020). A task execution framework based on aging indicator and sarimi. In *2020 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*, pages 18–25.

Umesh, I., Srinivasan, G., and Torquato, M. (2017). Software aging forecasting using time series model. *Indonesian Journal of Electrical Engineering and Computer Science*, 8(3):589 – 596. Cited by: 6; All Open Access, Green Open Access.

Wang, S. and Liu, J. (2020). Harrd: Real-time software rejuvenation decision based on hierarchical analysis under weibull distribution. In *2020 IEEE 20th International Conference on Software Quality, Reliability and Security (QRS)*, pages 83–90.

Yue, J., Wu, X., and Xue, Y. (2020). Microservice aging and rejuvenation. In *2020 World Conference on Computing and Communication Technologies (WCCCT)*, pages 1–5.