Software Aging in Image Classification Systems on Cloud and Edge

Ermeson Andrade*, Fumio Machida†, Roberto Pietrantuono‡, Domenico Cotroneo‡
*Department of Computing, Federal Rural University of Pernambuco, Recife, Brazil, ermeson.andrade@ufrpe.br
†Department of Computer Science, University of Tsukuba, Tsukuba, Japan, machida@cs.tsukuba.ac.jp
‡University of Naples Federico II, Naples, Italy, {roberto.pietrantuono, cotroneo}@unina.it

***Abstract*—Image classification systems using machine learning are rapidly adopted in many software application systems. Machine learning models built for image classification tasks are usually deployed on either cloud computing or edge computers close to data sources depending on the performance and resource requirements. However, software reliability aspects during the operation of these systems have not been properly explored. In this paper, we experimentally investigate the software aging phenomena in image classification systems that are continuously running on cloud or edge computing environments. By performing statistical analysis on the measurement data, we detected a suspicious phenomenon of software aging induced by image classification workloads in the memory usages for cloud and edge computing systems. Contrary to the expectation, our experimental results show that the edge system is less impacted from software aging than the cloud system that have four times larger allocated memory resources. We also disclose our software aging data set on our project web site for further exploration of software aging and rejuvenation research.**

***Index Terms*—Cloud computing, Edge computing, Image classifiers, Machine learning, Software aging**

I. Introduction

Image classifiers using machine learning models are now pervasively deployed in consumer and industrial software systems. Recent advances in deep learning have enable highly accurate image classification in versatile application domains such as face recognition, surveillance systems, and autonomous vehicles [1]. To build an image classifier, a deep learning model is trained from image data sets on rich computing resources. The copies of the trained models can be used in software programs that receive image samples and output the classification results for the applications.

Depending on the requirements and constraints from the applications, image classifiers are deployed in either a cloud computing or an edge computer that is located near the data sources. The edge computer can be a stand-alone server, a small computer, or a smart device that is directly connected to an image sensor (e.g., camera). Compared to the cloud computing that offers scalable computing resources on demand, the edge computers have limited resources that may not be sufficient for processing image processing [2]. On the other hand, network connection and bandwidth tend to be the bottleneck of cloud computing, especially for latency-critical applications such as connected car [3]. Existing studies for edge computing discussed when and how to allocate computing tasks between cloud and edge considering the performance and resource requirements [4]. However, little has been explored on the software reliability aspects during the operation, in particular for continuously running software programs dealing with image classification tasks.

In this paper, we experimentally evaluate the software aging phenomena in image classification systems continuously running on cloud or edge computing environments. We examine how software aging phenomenon appears differently between these two environments. In the experiments, we used the MNIST data set [5] that contains hundreds of handwritten images of digits from 0 to 9. The image classifier is trained from the training data set offline. The software program employing the trained classifier is deployed on either a cloud or an edge system on which classification tasks are executed continuously in response to the image samples sent from a client. For different architecture options (i.e, either the cloud or the edge) and different workload intensities, we collected system performance metrics for 72 hours. Using Mann-Kendall test [6] with Sen's slope estimate [7], we confirmed the increasing trends in the memory consumption both in the edge and the cloud systems. In high workload case for the cloud system, we observed a system failure due to the depletion of memory. Contrarily to our expectation, the experimental results indicate that the cloud system has more significant impact on software aging. Although the results cannot be generalized easily to other applications, our observation implies that we cannot always assume that cloud is more robust than edge computing against software aging. Since the manifestation of software aging highly depends on software stacks under the executed program, the amount of available resource does not guarantee the safe execution of long running software program. Our data analysis provides only a preliminary result and further analysis are required. To open further studies on software aging and rejuvenation in cloud and edge computing systems, we provide our data set online [8] that can be used for research purpose.

The rest of the paper is organized as follows. Section II describes the related work for software aging analysis. Section III explains our experimental plan in detail. Section IV shows the experimental results with some statistical analysis. Section V presents our conclusion.

II. Related Work

Studying software aging in cloud-based systems is becoming increasingly important because of the negative impact that a problem can have both on user-perceived quality of services (e.g., availability, reliability and performance) and, consequently, on the huge market around cloud-based systems and services. There is an increasing interest from the Software Aging and Rejuvenation (SAR) community in analyzing the phenomenon in cloud-based systems. A recent survey reports almost one hundred papers in the last ten years with a considerable number of research groups around the world addressing this topic [9].

A considerable number of works have been proposed to investigate SAR in the cloud, or more generally in virtualized systems, by exploiting stochastic models such as stochastic Petri nets (SPN) and stochastic reward nets (SRN) [10]–[13], continuous-time Markov chains (CTMC) [14], [15], semi-markov processes (SMP) [16], [17], as well as combinatorial models such as reliability block diagrams (RBD) [18] and dynamic fault trees (DFT) [19]. In these works, the cloud architectures – including virtual machines (VMs), virtual machine monitor (VMM), and physical host(s) - and the associated rejuvenation strategies are modeled with the aim of computing the optimal time for rejuvenation and of fine-tuning the adopted rejuvenation techniques.

Many other researchers considered a measurement-based approach to the aging problem in cloud-based systems or cloud applications, which is the same approach we adopt in this work. Statistical techniques for time series analysis of indicators of interest (such as response time and memory/resource consumption) is the most common approach. For instance, the work by Araujo *et al.* characterize the aging phenomenon on the Eucalyptus cloud computing framework [20]. That work adopts several regression models including linear, quadratic, exponential growth, and Pearl-Reed logistic models to predict memory consumption trends and schedule software rejuvenation properly. Sukhwani *et al.* analyze the aging of IBM cloud controller systems with a similar strategy [21]. Umesh *et al.* also exploit time series models to forecast software aging patterns of Windows active directory service for virtualized environments [22].

Mohan and Reddy study the effect of aging on an uncommon, but very important indicator for cloud computing, namely energy consumption [23]. The authors exploit linear regression to estimate the trend. Energy consumption is also considered by Villalobos *et al.* [24], where an IDS-based self-protection mechanism at the virtual machine level inspired by software rejuvenation concepts is presented. A correlation between IDS accuracy, attack rate, cloud system workload, energy consumption, and response time is identified – in fact, security-related aging problems are also of increasing interest. The work presented in [25] performs a workload-dependent analysis of performance degradation and memory indicators in Apache Storm, an event stream processing (ESP) application, deploying tasks over a cloud architecture, by means of workload-dependent time series analysis. In addition to time-series analysis, machine learning strategies have also been used in cloud-based systems or cloud applications to detect/predict the possible aging system state (e.g., [26] [27] [28]), as well as the simpler threshold-based approach on specific aging indicators [29] [30].

In this work, we focus not only on aging at cloud level, but also explore aging in an edge-computing scenario. Deploying physical resources and distributing computational efforts following a different architectural style, such as in the edge computing paradigm, can have effect on the overall performance degradation perceived by the end user. Moreover, to the best of our knowledge, the task we consider as application, namely machine-learning-based image classification, is also unexplored from the SAR perspective. We hereafter show whether this task is able to expose aging phenomena in the underlying cloud-based and edge-based architectures.

III. Experimental Plan

A. Research questions

The objective of our study is to analyze the potential software aging issues of image classification systems running on cloud or edge. If software aging appears, it is interesting to know the difference of aging impacts between cloud and edge systems. Therefore, we set the following research questions for our experimental study.

- RQ1: Does an image classification system executing on a cloud encounter any software aging issue? and, if so, how significant it is and what is the cause?
- RQ2: Does an image classification system executing on an edge computer encounter any software aging issue? and, if so, how significant it is and what is the cause?
- RQ3: Does the image classification system executing on an edge computer have a higher impact from software aging than a the same system executing on a cloud?

To answer these questions, we setup the following experiments.

B. Setup

For an application of image classification system, we used the MNIST data set that contains hundreds of handwritten images of digits from 0 to 9. This data set is a well known benchmark for image classification. The challenge with this data set is to correctly classify a handwritten digit based on a 28-by-28 black and white image. Therefore, we implemented in Python a neural network system that recognizes handwritten digits based on the MNIST data set. In this system, images are generated and sent from a client's device over the network to a server for image classification on a continuous basis. Note that the input of the image classification system is a 784 size vector, obtained from converting the image, originally a 28x28 size matrix, to a 784 position vector.

Two distinct testbeds were adopted for the experiments. One for the cloud architecture and the other for the edge architecture. Our testbed for the former consisted of a client device and a virtual machine executing the image classification system (hosted on Google's infrastructure). The latter consisted of a client device and an edge also executing the same image classification system. Note that the client device is the workload generator, which is a program written in Python 3 to create constant workload. The settings for the devices are the following:

- Client device: Apple MacBook Air 11-in, Intel Core i5 1.60GHz, 4 GB, 64 GB, Mac OS X Lion 10.7.
- Virtual machine: n1-standard-1 (1 vCPUj, 3.75 GB of memory), Debian GNU/Linux 10 located in us-central1-a.
- Edge: Raspberry Pi 3, 900MHz quad-core ARM CPU, 1GB RAM, running the default Raspbian Linux image.

C. Experiments, metrics, and analysis method

To address the research questions, two sets of experiments was carried out. The first set aimed at investigating the possible presence of software aging in the image classification system executing on the edge computer. For that, we considered the following workload settings: no workload, low workload (1 image every 1 second), medium workload (1 image every 0.5 seconds) and high workload (1 image every 0.1 seconds). For each workload setting, we executed the experiments for 72 hours. The second set investigated the possible presence of software aging in the image classification system executing on the cloud. The workload settings were the same as the ones presented for the edge. The difference is that the experiments were carried out in the cloud.

For each experiment, we gathered data and analyzed aging indicators. These indicators refer to the system variables that can be directly measured and can be related to the software aging phenomena [31]. The analysis of aging indicators can be performed both at system level and process level. System level analysis investigates system resources which can age. On the other hand, process level analysis aims to identify those processes more responsible for resource consumption and user-perceived performance degradation, if any. In this work, the main aging indicators we considered were regard both the user-perceived performance and the resource depletion in terms of real memory consumption. As stated in [32], these are the typical aspects considered in software aging studies. Regarding the user-perceived performance, we adopted the mean response time, which is the mean time from sending the image to the end of image processing.

In term of aging detection, we adopted the conventional Mann–Kendall test (MKT) to estimate the trends of aging indicators, and the Sen's slope estimate to calculate the magnitude of the trends. The Mann-Kendall analysis checks the null hypothesis (H0) that there is no trend in the data during the time, while the alternative hypothesis (H1) indicates an upward or a monotonic downward trend in the data. If the p-value of the test is lower than the significance level (α =0.05), then there is statistically significant evidence that a trend is present in the time series data. As software aging is a cumulative process, the MKT can be used to reveal patterns of software internal state degradation. Although MKT suffers from high rates of false positives, it remains the most widely adopted test to detect aging [33]. The statistic to measure the magnitude of the trend is the Sen's slope. It is computed as the median between each pair of data points, so that a positive Sen's slope implies a positive trend, while a negative Sen's slope means a negative trend.

IV. Result Analysis

In this section, we present the experimental results with some statistical analysis to address our research questions.

A. Results from the cloud environment: RQ1

For the image classification system running on the cloud, we observed software aging issues both in the response time and in the memory usages. Figure 1 plots the measured response times for the image classification system by different workload settings. The horizontal and vertical axes represent the experimental time and the observed response times, respectively. The results clearly show the response times are affected by the workloads of image classification tasks. In particular, we observed an increase in response time for the case of high workload around 10 hours in which the virtual machine crashed. For the middle workload case, on the other hand, we observed the characteristic behavior of the response time that had two peaks in the observation period. After reaching the first peak around 18 hours, the response time decreases slowly, but it jumps up again around 56 hours. In total, the response time is getting worse. For the low workload case, the response time is relatively stable, although it can be observed a little increasing trend.

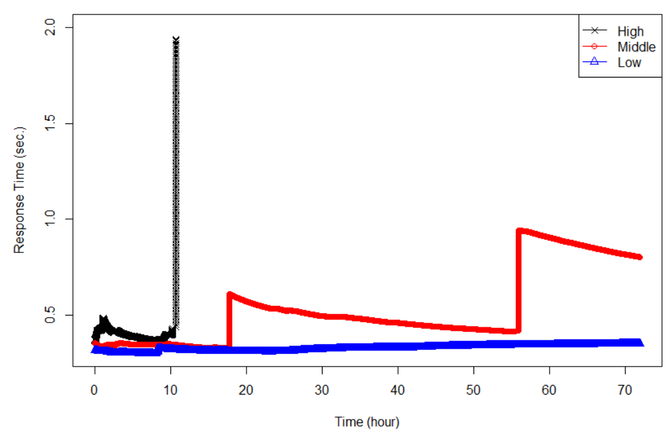


Fig. 1: Response time for the image classification system on the Cloud.

In Figure 2, we show the traces of memory usages in the cloud VM under the different workloads: (a) None, (b) Low, (c) Middle, and (d) High. For all the cases including no workload case, we observe the increasing trends in the memory usage. For the high workload case, the memory usage reached the maximum capacity of the VM around 10 hours at which the VM is restarted by the cloud. Similar behavior was detected in the middle workload case, but the memory was not released because it did not reached the maximum memory capacity.

As a result of Mann-Kendall test, the p-values are less than 0.05 for all the cases, leading to the conclusion that the null hypothesis is rejected. In order to compare the significance of trends, we applied Sen's slope estimator. Table I presents Sen's slope estimators for cloud memory usages under different workloads with the Lower Confidence Interval (LCI) and Upper Confidence Interval (UCI) at 95% confidence level. For high and middle workloads, we only considered the data until the first peak points. The results showed that the slope becomes more steep as the intensity of the workload increases.

TABLE I: Sen's slope for estimates for the cloud memory usage data in different workload settings at 95% confidence level with confidence intervals.

Workload	Slope estimate (MB/hour)	LCI	UCI
None	5.871991e-04	5.439642e-04	6.309148e-04
Low	5.345212e-04	5.034965e-04	5.658627e-04
Middle	1.5831818	1.53600	1.63016
High	5.7149268	5.689639	5.760805

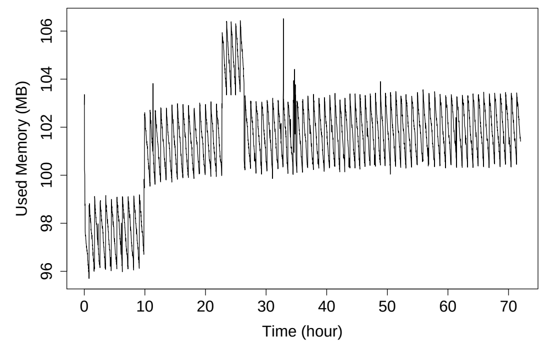
In order to understand the the underlying causes of the aging phenomenon, we performed a process analysis for the VM on the Cloud. For this purpose, we created a program in Python 3 to gather information about the processes running on the system every hour. This program uses the ps command[1] to retrieve such information, including the process identification numbers. Our analysis indicated that two main process were responsible for such increase in the memory consumption: "tmux" and "systemd-journald". Tmux is a terminal multiplexer, while systemd-journald is a system service that collects and stores logging data. In the experiments, tmux was used to keep the system running regardless of the cloud SSH connection. Although tmux is not the main part of image classification system (rather it is the part of experimental configuration), the memory consumption was affected by the workload intensity of image classification tasks. If this causes a failure like the one observed in the high workload case, it is not a negligible issue. To locate the root-cause, we need more deep investigation on the dependencies of related software components.

B. Results from the edge environment: RQ2

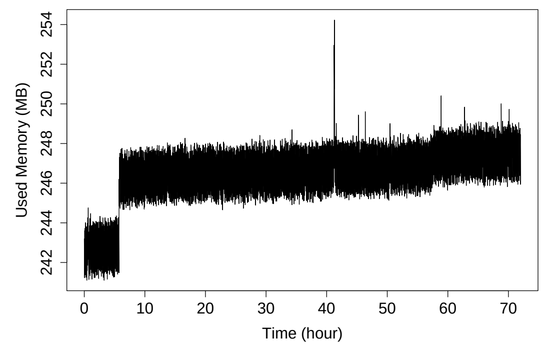
Likewise the cloud environment, we deployed the image classification system on our edge system and performed the same workload tests. Figure 3 shows the observed response times of the system by different workload settings. In contrast to the results from the cloud system, we do not see clear increasing trends of the response time regardless of the workload intensity. For the high and middle workload cases, the difference is almost negligible especially after 20 hours (the difference is less than 0.01 second). The response time for middle workloads, however, is consistently worse than the others (about 0.02 second longer than the other cases after 35 hours). We suspect this small difference was caused by temporally congestion of local area network traffic.

In Figure 4, we show the traces of memory usages in the edge computer under the different workloads: (a) None,

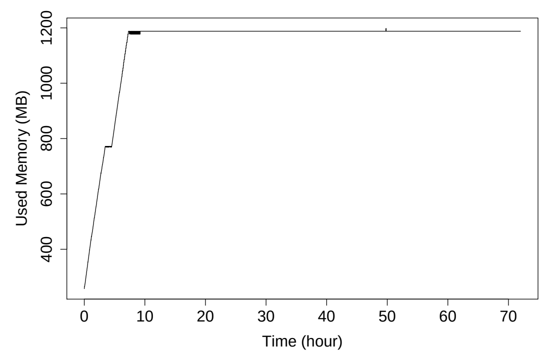
[1]The ps command is a traditional Linux command to lists running processes.



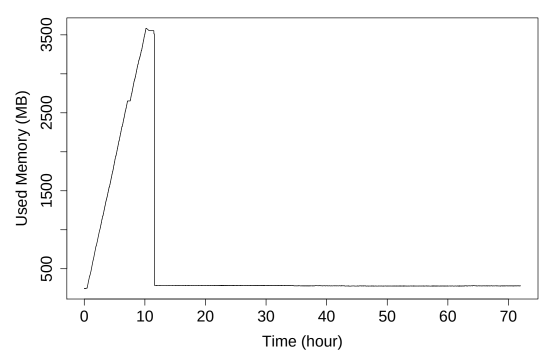
(a) None workload



(b) Low workload



(c) Middle workload



(d) High workload

Fig. 2: Cloud memory usages considering the following workloads: (a) None, (b) Low, (c) Middle and (d) High.

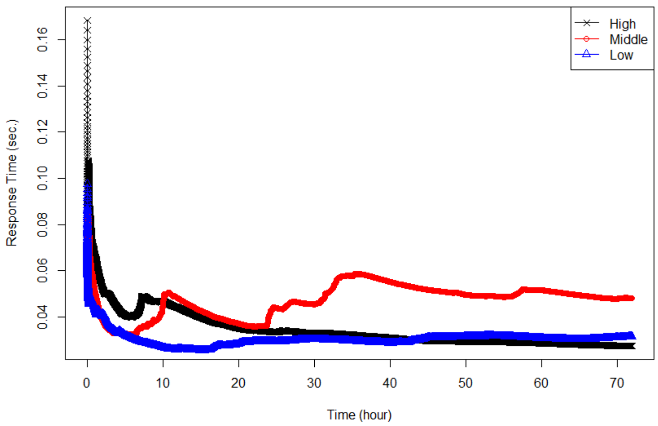


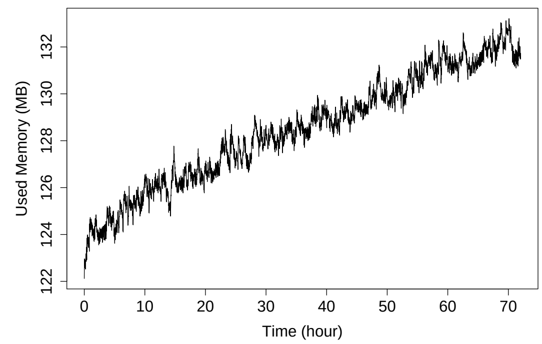
Fig. 3: Response time for the image classification system on the Edge.

(b) Low, (c) Middle, and (d) High. As can be seen, the results highlight the trends on memory consumption for all workloads, even in the case without workload. The results of Mann-Kendall test for these memory usage data showed the p-values are less than 0.05. Thus, the null hypothesis is rejected implying that there are trends in the data. We also applied the Sen's slope estimator to compare the significance of trends among different workloads. Table II shows Sen's slope estimators for edge memory usages, including the lower and upper confidence intervals. They reveal that higher workloads cause the steepest slopes of memory usage trends. The increasing trends in memory consumption maybe caused by benign process which does not have any aging-related bugs. However, there could be a factor that accelerates the memory usage in response to the workloads of image classification tasks.

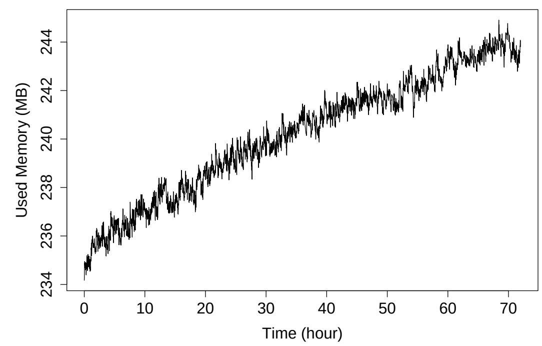
TABLE II: Sen's slope for estimates for the edge memory usage data in different workload settings at 95% confidence level with confidence intervals.

Workload	Slope estimate (MB/hour)	LCI	UCI
None	1.883117e-03	1.871147e-03	1.895147e-03
Low	1.962545e-03	1.950317e-03	1.974779e-03
Middle	1.984127e-03	1.971902e-03	1.996338e-03
High	2.078947e-03	2.060263e-03	2.097693e-03

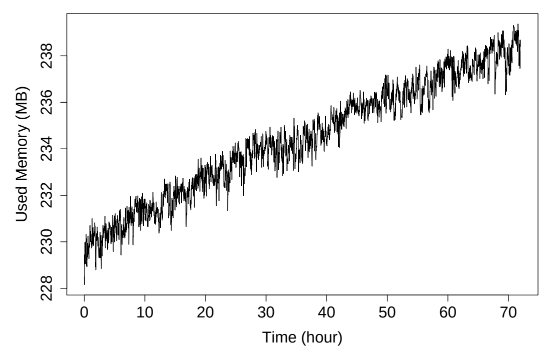
In order to understand the underlying causes of the increasing memory trends, we performed the process analysis to identify the processes that are stripping the edge's memory. Our analysis indicated that two main process were responsible for the increase in the memory consumption: "hwrng" and "systemd-timesyncd". The former is a library for random number generation, while the latter is a daemon used for synchronizing the system clock across the network. Since these processes are not the part of image classification tasks, we need more investigation for analyzing the potential software aging.



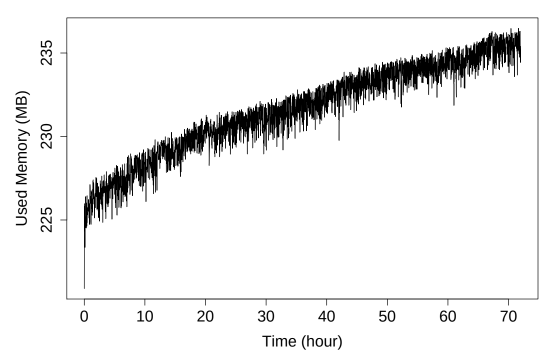
(a) None workload



(b) Low workload



(c) Middle workload



(d) High workload

Fig. 4: Edge memory usages considering the following workloads: (a) None, (b) Low, (c) Middle and (d) High.

C. Comparison between cloud and edge: RQ3

From the obtained results, we cannot state that an edge computer is more impacted from software aging than a cloud environment which has more computer system resources. In fact, our experimental results showed a counter-intuitive fact that the edge system provides more robust execution environment for the image classification system than the cloud system. For our experimental settings, the allocated memory for the cloud was about four times higher than the memory available on the edge (3.75GB vs. 1GB). With numerous interdependent and tightly coupled components, cloud-based environments can lead to aging gaps and make it difficult to find the underlying causes of this phenomenon. Additionally, as expected, the edge had a better performance than the cloud. For instance, if we consider high workload, the edge had, on average, a response time 17 times faster than the Cloud.

V. Conclusion

In conclusion, we observed that the manifestation of software aging phenomenon in image classification tasks had quite different characteristics between cloud and edge computing environments. In our test application program, software aging has a significant impact on the cloud environment rather than edge computing systems with less amount of resources. This could be just an instance of complex environment-dependent software aging problems among cloud, fog and edge computing architectures. Software aging experiments are essential to understand the actual software aging impacts on different software stack and to determine the right place to deploy the classifier in terms of software system reliability.

Acknowledgment

We would like to thank Dr. Bruno Nogueira for his help in setting up the experiments. Additionally, this research was partially funded by CNPq - Brazil, grant 406263/2018-3.

References

[1] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017.

[2] F. Machida, M. Fujiwaka, S. Koizumi, and D. Kimura, "Optimizing resiliency of distributed video surveillance system for safer city," in *2015 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, 2015, pp. 17–20.

[3] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected vehicles: Solutions and challenges," *IEEE internet of things journal*, vol. 1, no. 4, pp. 289–299, 2014.

[4] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7944–7956, 2019.

[5] L. Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[6] H. B. Mann, "Nonparametric tests against trend," *Econometrica: Journal of the econometric society*, pp. 245–259, 1945.

[7] P. K. Sen, "Estimates of the regression coefficient based on kendall's tau," *Journal of the American statistical association*, vol. 63, no. 324, pp. 1379–1389, 1968.

[8] "Software aging data in image classification systems," 2020, [Online]. https://www.dependability.cs.tsukuba.ac.jp/dataset/.

[9] R. Pietrantuono and S. Russo, "A survey on software aging and rejuvenation in the cloud." *Software Quality Journal*, vol. 28, pp. 7–38, 2020.

[10] J. Xu, X. Li, Y. Zhong, and H. Zhang, "Availability Modeling and Analysis of a Single-Server Virtualized System with Rejuvenation," *Journal of Software*, vol. 9, no. 1, pp. 129–139, 2014.

[11] A. Rezaei and M. Sharifi, "Rejuvenating high available virtualized systems," in *5th International Conference on Availability, Reliability, and Security (ARES)*. IEEE, 2010, pp. 289–294.

[12] F. Machida, D. S. Kim, and K. S. Trivedi, "Modeling and analysis of software rejuvenation in a server virtualized system," in *Second International Workshop on Software Aging and Rejuvenation (WoSAR)*. IEEE, 2010.

[13] ——, "Modeling and analysis of software rejuvenation in a server virtualized system with live VM migration," *Performance Evaluation*, vol. 70, no. 3, pp. 212–230, 2013.

[14] M. Myint and T. Thein, "Availability Improvement in Virtualized Multiple Servers with Software Rejuvenation and Virtualization," in *Fourth International Conference on Secure Software Integration and Reliability Improvement (SSIRI)*. IEEE, 2010, pp. 156–162.

[15] T. Thein and J. S. Park, "Availability Analysis of Application Servers Using Software Rejuvenation and Virtualization," *Journal of Computer Science and Technology*, vol. 24, no. 2, pp. 339–346, 2009.

[16] F. Machida, V. F. Nicola, and K. S. Trivedi, "Job completion time on a virtualized server subject to software aging and rejuvenation," in *Third International Workshop on Software Aging and Rejuvenation (WoSAR)*. IEEE, 2011, pp. 44–49.

[17] ——, "Job Completion Time on a Virtualized Server with Software Rejuvenation," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 10, no. 1, pp. 10:1–10:26, 2014.

[18] M. Melo, P. Maciel, J. Araujo, R. Matos, and C. Araujo, "Availability Study on Cloud Computing Environments: Live Migration As a Rejuvenation Mechanism," in *43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2013.

[19] J. Rahme and H. Xu, "A Software Reliability Model for Cloud-Based Software Rejuvenation Using Dynamic Fault Trees," *International Journal of Software Engineering and Knowledge Engineering*, vol. 25, no. 09n10, pp. 1491–1513, 2015.

[20] J. Araujo, R. Matos, V. Alves, P. Maciel, F. Vieira de Souza, R. Matias Jr., and K. S. Trivedi, "Software Aging in the Eucalyptus Cloud Computing Infrastructure: Characterization and Rejuvenation," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 10, no. 1, pp. 11:1–11:22, 2014.

[21] H. Sukhwani, R. Matias, K. S. Trivedi, and A. Rindos, "Monitoring and Mitigating Software Aging on IBM Cloud Controller System," in *28th International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, 2017, pp. 266–272.

[22] I. M. Umesh and G. N. Srinivasan, *Dynamic Software Aging Detection-Based Fault Tolerant Software Rejuvenation Model for Virtualized Environment*, ser. Advances in Intelligent Systems and Computing. Singapore: Springer, 2017, vol. 469, pp. 779–787.

[23] B. R. Mohan and G. R. M. Reddy, "The effect of software aging on power usage," in *9th International Conference on Intelligent Systems and Control (ISCO)*. IEEE, 2015.

[24] J. J. Villalobos, I. Rodero, and M. Parashar, "Energy-Aware Autonomic Framework for Cloud Protection and Self-Healing," in *International Conference on Cloud and Autonomic Computing (ICCAC)*. IEEE, 2014, pp. 3–4.

[25] M. Ficco, R. Pietrantuono, and S. Russo, "Aging-related performance anomalies in the Apache Storm stream processing system," *Future Generation Computer Systems*, vol. 86, pp. 975–994, 2018.

[26] C. Sudhakar, I. Shah, and T. Ramesh, "Software Rejuvenation in Cloud Systems Using Neural Networks," in *International Conference on Parallel, Distributed and Grid Computing (PDGC)*. IEEE, 2014, pp. 230–233.

[27] D. R. Avresky, P. D. Sanzo, A. Pellegrini, B. Ciciani, and L. Forte, "Proactive Scalability and Management of Resources in Hybrid Clouds via Machine Learning," in *14th International Symposium on Network Computing and Applications (NCA)*. IEEE, 2015, pp. 114–119.

[28] I. M. Umesh and G. N. Srinivasan, "Optimum software aging prediction and rejuvenation model for virtualized environment," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 3, no. 3, pp. 572–578, 2016.

[29] L. Silva, J. Alonso, and J. Torres, "Using Virtualization to Improve Software Rejuvenation," *IEEE Transactions on Computers*, vol. 58, no. 11, pp. 1525–1538, 2009.

[30] J. Araujo, R. Matos, P. Maciel, F. Vieira, R. Matias, and K. Trivedi, "Software Rejuvenation in Eucalyptus Cloud Computing Infrastructure: A Method Based on Time Series Forecasting and Multiple Thresholds," in *Third International Workshop on Software Aging and Rejuvenation (WoSAR)*. IEEE, 2011, pp. 38–43.

[31] K. S. Trivedi, M. Grottke, and E. Andrade, "Software fault mitigation and availability assurance techniques," *International Journal of System Assurance Engineering and Management*, vol. 1, no. 4, pp. 340–350, 2010.

[32] D. Cotroneo, R. Natella, R. Pietrantuono, and S. Russo, "A survey of software aging and rejuvenation studies," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 10, no. 1, pp. 1–34, 2014.

[33] F. Machida, A. Andrzejak, R. Matias, and E. Vicente, "On the effectiveness of mann-kendall test for detection of software aging," in *2013 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, 2013, pp. 269–274.