J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr

A machine learning framework for performance anomaly detection☆

Muhammad Hasnain[1] Muhammad Fermi Pasha[2] Imran Ghani[3] Seung Ryul Jeong[4*] Aitizaz Ali[5]

ABSTRACT

Web services show a rapid evolution and integration to meet the increased users' requirements. Thus, web services undergo updates and may have performance degradation due to undetected faults in the updated versions. Due to these faults, many performances and regression anomalies in web services may occur in real-world scenarios. This paper proposed applying the deep learning model and innovative explainable framework to detect performance and regression anomalies in web services. This study indicated that upper bound and lower bound values in performance metrics provide us with the simple means to detect the performance and regression anomalies in updated versions of web services. The explainable deep learning method enabled us to decide the precise use of deep learning to detect performance and anomalies in web services. The evaluation results of the proposed approach showed us the detection of unusual behavior of web service. The proposed approach is efficient and straightforward in detecting regression anomalies in web services compared with the existing approaches.

☞ keyword : Web services update, undetected regression anomalies, performance metrics, services integrate.

1. Introduction

A web service integrated into a composite service requires regression testing to comply with the integrators' assumptions and provider-integrator contract. A test can be triggered either by periodic invocations or by sending a notification of a service update [1]. Frequent web service releases or updates make conducting a thorough or detailed performance evaluation of the updated applications challenging.

Performance regression happens when software systems evolve or update and significantly degrade the software performance [2]. Due to the high overhead of this testing activity, it is performed infrequently. Since final testing, several commits merge, and software testers require to spend a lot of time and effort to detect problems. Existing approaches show efforts to improve performance regression testing efficiency by test case reduction or test case prioritization [3].

Transaction profiling (TP) based regression testing [4] uncovers the performance regression anomalies caused by the software updates. A report generates to isolate real anomalies from those one produced from workloads. This research incorporates the explainable methods with the software domain to detect performance anomalies in web services.

The contribution of this paper is as follows:

- First, we propose the detection of performance and regression anomalies from Causal analysis.
- We proposed an explainable framework to enhance the interpretation of root causes behind performance and regression anomalies.
- Second, we propose the evaluation of the proposed approach by using the real-world dataset of web services.

The rest of the paper includes a literature review in section 2, the proposed approach in section 3, results and discussion in section 4, and conclusions and research implications in section 5.

2. Literature Review

Web services have several vulnerabilities and attacks. To overcome these attacks, experimental analysis has been performed. This analysis provides an understanding of

1, 2, 5 School of Information Technology, Monash University, Selangor, 47500, Malaysia.
3 Virginia Military Institute, Lexington, VA, 24450, USA.
4 Graduate School of Business IT, Kookmin University, Seoul, 136, Korea.
* Corresponding author: srjeong@kookmin.ac.kr
[Received 10 October 2021, Reviewed 14 October 2021(12 November 2021), Accepted 7 December 2021]
☆ A prelimimary version of this paper was presented at APIC-IST 2021.

vulnerabilities and attacks on the exchanged information over the web services and resources. This study proposed an approach based on countermeasures to mitigate or prevent such attacks. This study suggests developing an anomaly-based intrusion detection system using the latest data mining techniques [5]. A recently published study [6] reveals that agents are used to detecting violations of normal profiling from data mining techniques, such as clustering, sequential association, and association rules. Violations or anomalies are further analyzed to differentiate between genuine and normal attacks. As a result, genuine attacks' prevention reduces the false alarm rate.

The detection of performance degradation is still a hot research area, and many research efforts have been made to detect mal-functional resources as earlier as possible. Every change in the software code, hardware conditions, and workloads results in the state of the applications from normal to abnormal and causes anomalies or changes in software applications [7].

Individual classes of Java programs have been evaluated for their performance validation. Unnecessary synchronization makes it difficult because diverse workloads access the classes. A performance regression technique, namely SpeedGun [8], was proposed for the thread-safe classes. The main idea behind the proposed approach was to notify the developers when a change in thread class occurs and results in influencing the performance of a class. This proposal of SpeedGun is convincing for programs with access to their codes. Still, it shows limitations when it becomes difficult to access integrated web services without prior obtaining the consent of services providers.

(Table 1) Summary of studies

Study	Technique	Advantages and Disadvantages
Martin et al. [9]	Machine learning-based	Early warning about security risks. Storage of use cases is a difficult task.
ElSayed et al. [10]	Convolutional neural network (CNN) based intrusion detection system	Overcome the overfitting problem and detect the unseen intrusion events. This approach has not been evaluated in a real-world environment.
El-shamsy et al. [11]	Support vector machine	Precise and accurate failure detection. There is no comparison between the proposed approach and the latest deep learning models.
Nedelkoski et al. [12]	Long short term memory (LSTM)	Anomaly detection in the execution of system components. This approach shows limitations to anomaly detection in large-scale systems.
Ahmed et al. [13]	Application performance management tools	APM tools detect and diagnose injected performance regression. APM based approach has limitations of extending
Ocriza Jr. and Zhao [14]	Automated comparison and execution of timelines	Detection of performance regression causes with 100% path precision. The proposed approach is highly time-consuming and can be optimized.
Ahmed et al. [15]	An approximate solution to execute a large number of loops	Increased usage of resources. The proposed approach performs on the cost of high utilization of resource
Lin et al. [16]	Microscopic approach	Good diagnostic results were presented. It shows a limitation to detecting anomalies from the source code of web services.

Table 1 is the illustration of literature regarding the research topic. Several studies have been performed to handle the performance and regression issues in various web systems.

Machine learning and statistical methods are being employed to detect abnormalities in trends or patterns used as a standard for entities and users. These methods are helpful in cybersecurity because administrators and managers can know about potential cybersecurity risks. Fingerprints as a source of identification detect users' behavior from the stored information. Although the proposed approach is validated and evaluated, we still need broad storage of use cases for identity management Martin et al. [9].

The applications of machine learning-based approaches are highly penetrating in the other research areas. Anomalous power consumption is a highly focused area of research to understand the cause of each anomaly in the power sector. Machine learning approaches are gaining value in anomalies detection in building energy consumption [17].

Recurrent neural networks (RNNs) are widely used to analyze time-series information and exhibit temporal dynamic behavior. One of the essential applications is to determine the pattern deviation from emerging energy usage. Though RNNs canforecast energy usage and detect anomalies, they suffer in dealing with the unbalanced features of datasets for anomaly detection. Therefore, the proposal of "generative adversarial networks" can model complex as well as high dimensional data such as cybersecurity, time series, and images [18].

Many organizations have migrated their systems to the cloud as they expected high reliability, performance, and quality services. A variety of applications are being deployed to manage their sources effectively. Software-defined networks (SDN) based approaches are used to detect performance anomalies in data centers at clouds. Support vector machine (SVM) technique is trained by the data collected from various connected devices. Although static threshold performance values are used, dynamic features are missed to forecast the correct performance behavior of applications.

Several machine learning applications running on Google, Facebook and Amazon, and many other large-scale systems must ensure high responsiveness and availability. The main issue with these systems is the monitoring of their performance bottlenecks [18]. The error operation and avoidance of violations at services level agreement are the main focused areas. Therefore, post-deployment performance diagnostic (PPD) methods provide performance analysis and result in the selection of better approaches.

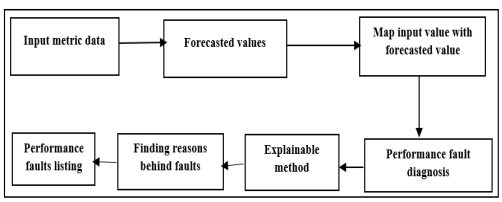
Web services' stable operations via monitoring by staff guarantee that the operations meet the requirements of a regular web service. Shi et al. [19] used machine learning and statistical methods in detecting anomalies. Key performance indicators' time-series information was used to detect the anomalous points. Next, the forecasted value is compared with the actual KPI values to determine whether the current point was an anomalous point or not.

In recent research, Delgado-Perez [20] investigated performance degradation from the execution of test cases. Similar to earlier stated studies, artificial faults were injected to detect performance bugs. The feasibility of applying the performance mutation testing (PMT) was tested against the mutation operators in C++ programs. Compared with the traditional mutation operator-based approaches, the proposed operators induced the design of particular test inputs. A new set of challenges regarding the PMT can be carried out in future works.

- First, we propose the detection of performance and regression anomalies from Causal analysis.
- We proposed an explainable framework to enhance the interpretation of root causes behind performance and regression anomalies.
- Second, we propose the evaluation of the proposed approach by using the real-world dataset of web services.

3. Proposed Approach

The proposed approach is based on the fundamental processes to build the deep learning predictive and explainable methods to detect anomalies and performance regression in this research. In the following Figure, we show the process flow of the proposed approach. Next, we explain each step of the proposed approach.



(Figure 1) Workflow of the proposed approach

3.1 Input metric data

The first process of the proposed approach is about the performance metric data of web services. Response time and throughput metrics are focused on in this research study. Throughput is mainly used as a quality web service metric. Throughput is referred to the number of invocations handled by a running web service in unit time [21]. Service providers declare the original throughput values in the service level agreement. In this study, we employ throughput as a dynamic performance metric. The motivation behind modeling the performance metrics is a practical one. It is naturally adapted to user requests.

3.2 Forecasted values

Performance metrics generated in a controlled environment (JMeter) are further used and fed in deep learning models. Therefore, we have chosen the popular prediction model such as long-short term memory (LSTM) to efficiently predict web services performance.

3.3 Map input and forecasted values

The third process in the proposed approach involves mapping the original performance metric values and forecasted values. If a forecasted value is lower than the actual values, an alert is activated for the web service providers.

3.4 Performance fault diagnosis

The purpose of this step is to analyze the performance issues in the running web services. Both original and predicted performance metrics patterns are plotted to find out the outliers in the performance metric values. Performance metric value above and below the defined metric value is taken as a performance regression in web services. Outlier detection from the neighboring point has been discussed in the literature [22]. However, the reason behind the exception behavior of data is not explored.

Therefore, the present sheds light on the causes behind the outlying points. One of these reasons is the workload applied to a web service analyzed to see its impact on the performance of web services. To explore the reasons behind performance anomalies and enhance the fault diagnosis, this approach proposes to use explainable methods in this study.

3.5 Explainable method

Understanding the deep learning-baseddecisions and explainable methods could enhance the capabilities of models to understand them beyond their performance metrics. These metrics include precision and accuracy. Explainable methods can add valuable insights from the deep learning fault-detection. The proposed approach is beneficial in the decision-making of detection models and mimicking human intelligence. The primary rationale behind using the explainable method is to avoid inequality and bias in decision-making.

3.6 Finding reasons behind faults

Since several security approaches operate in a block-box model and only a trace detection is not sufficient to mark a performance point anomalous. Instead, we need details on the anomaly, its severity, and causes [22]. Therefore, we met the challenge of determining the root causes behind the performance anomaly. A visualization method can reveal sufficient information to know the root causes. A fishbone or Ishikwa diagram can be used to identify the root causes of performance anomalies in web services.

3.7 Performance faults listing

The final process of the proposed approach is focused on listing the faults identified from the previous processes. Priority handling each identified performance fault is stated to avoid further performance degradation in web service performance. The priority of each fault is gauged by its severity and impacts.

3.8 Causal inference machine learning

As a powerful modeling tool, causal inference enables machine learning to become explainable. The critical step in this path is to develop an explainable artificial intelligence technique. Several themes in this area have been proposed in the literature: (1) "Estimating average treatment effect" [23] and (2) "causal potential theory" [24]. None of the proposed themes is focused on anomaly detection from causal inference regarding the performance of web services. Wu et al. [25] proposed an

approach to measure theanomalous propagation from the CPU hog issue. The proposed approach missed the key performance metrics (response time and throughput) to use and exploit the causal inference in detecting performance anomalies. A web service graph from these performance metrics can be constructed and infer the root causes of performance issues. Response time or throughput fluctuations can be detected by an efficient performance testing approach [26]. A fundamental notion in science is causality that plays a key role in prediction, explanation, and decision making.

Since most web applications are migrating to the cloud environment, the instrumentation of web services at the code level is difficult. In addition, reliability and cost are other important metrics:

To release anupdated version of the software, reliability and cost are two important criteria determining the release policy of software systems [27]. Reliability determines how a software system behaves after a new release or an updated version. Let Cr denotes the defects identified in an updated version, and Ft indicates the total faults of the system. To determine the percentage of corrected faults, we propose the following equation:

$$Re\frac{C_r}{F_t} \tag{1}$$

If the percentage value is higher, it denotes the efficiency of a proposed approach.

Causality analysis of faults or anomalies can be performed by Transfer entropy between linear and non-linear relationships [28]. Transfer entropy finds the causal map amongst time series between linear and non-linear relationships. The proposed approach uses the concept of causal inference to investigate the regression anomaly among updated software versions.

The successful fault detection and potential anomalies in the updated version and their possible root causes are proposed to use the causality analysis. According to causality, the data is beyond the upper bound, and the lower bound of a data value should highlight the regression anomalies. Each data node has either two states (1 or 0)indicating the current regression anomalies and non-occurrence of regression anomalies, respectively (see Table 2).

(Table 2) Performance anomaly detection cases

Case	Performance Anomaly detected	Explanation
1	Yes	Value is beyond the upper and lower bounds
0	No	The value lies between upper and lower bounds

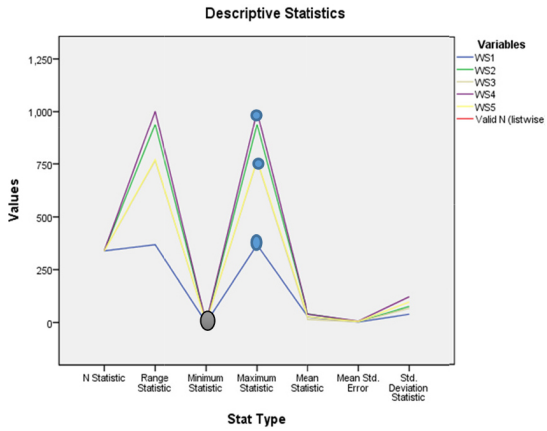
A node failing in its functional results shows a regression anomaly. Updated web services can be caused by code faults or component failures resulting from external factors not meeting the requirements.

First, we need to use quality of service (QoS) values predefined in service level agreement (SLA) documents to determine the values of the upper bound and lower bounds. The SLA documents mention the peak values of each factor, which is provided with every release of a software system. Software undergoing modifications or updates needs to meet the performance requirements. Therefore, unchanged portions of the software systems should show similar working and performance before change or updates.

4. Results and Discussions

To generalize the proposed approach and make it reproducible, we evaluate the proposed approach on a real-world dataset [29] of web services. This dataset has a collection of throughput and response time values of 5285 web services from 339 users. First, we obtain simple information of chosen dataset as shown in the following Figure.

Figure 2 is the illustration of statistics of chosen web services from the dataset. A total of five web services were randomly selected in this study. The concurrent 339 users accessed each web service. As shown in Figure 2, the minimum and maximum throughput values of web services are expressed by black and blue circles, respectively. These values indicate the lower bounds and upper bounds, respectively. In other words, we detect anomalous points that go beyond the upper and lower thresholds. Metrics values mentioned in the SLA are taken as reference values. Among these web services datasets, WS4 is showing us a similar lower bound value to other web services but a higher upper bound value than other web services. It indicates an



(Figure 2) Upper and lower bound performance values

unusual behavior of WS4 than other web services in this study.

For instance, we assume that the SLA document of WS1 has 400 and 20 kbs\sec upper bound and lower bound values, respectively. Although the mean value is 26.41, most of the users' transactions show less than 20 kbs\sec value for most web service users (WS1). This shows the performance regression in WS1.Similarly, we noticed the performance.

The explainable deep learning detection model may have influenced the performance improvement plan of web services. Web services providers may understand why the detection model was used for fault-detection in web services. These findings help researchers to find ways of improving the performance of web services.

The proposed approach is simple in its implementation as compared to the existing approaches. Compared to a recently published work [30], our proposed approach considers performance and regression anomalies rather than handling test case minimization and prioritization in web services.

5. Conclusions

In this paper, we present innovative means of detecting performance and regression anomalies in web services. We propose applying an explainable method and applying the causal analysis to reveal the anomalies' detection in updated versions of web services.

To clarify this study's results and findings, we may apply the proposed approach to a simulated dataset of web services. The current study evaluated the proposed approach on real-world web services performance data. However, invoked web services show constant users' workloads. To simulate the varying workloads, a simulation environment can be used to analyze the performance behavior of web services.

References

[1] G. Canfora and M. Di Penta, "Testing services and service-centric systems: Challenges and opportunities," *IT Professional,* vol. 8, no. 2, pp. 10-17, 2006. https://doi.org/10.1109/MITP.2006.51

[2] P. Huang, X. Ma, D. Shen, and Y. Zhou, "Performance regression testing target prioritization via performance risk analysis," in *Proceedings of the 36th International Conference on Software Engineering*, pp. 60-71, 2014. https://doi.org/10.1145/2568225.2568232

[3] S. Elbaum, A. G. Malishevsky, and G. Rothermel, "Test case prioritization: A family of empirical studies," *IEEE transactions on software engineering,* vol. 28, no. 2, pp. 159-182, 2002. https://doi.org/10.1109/32.988497

[4] S. Ghaith, M. Wang, P. Perry, Z. M. Jiang, P. O'Sullivan, and J. Murphy, "Anomaly detection in performance regression testing by transaction profile estimation," *Software Testing, Verification and Reliability,* vol. 26, no. 1, pp. 4-39, 2016. http://dx.doi.org/10.1002/stvr.1573

[5] A. Ghourabi, T. Abbes, and A. Bouhoula, "Experimental analysis of attacks against web services and countermeasures," in *Proceedings of the 12th International Conference on Information Integration and Web-based Applications & Services*, pp. 195-201, 2010. http://dx.doi.org/10.1145/1967486.1967519

[6] C. I. Pinzón, J. Bajo, J. F. De Paz, and J. M. Corchado, "S-MAS: An adaptive hierarchical distributed multi-agent architecture for blocking malicious SOAP messages within Web Services environments," *Expert Systems with Applications,* vol. 38, no. 5, pp. 5486-5499, 2011. http://dx.doi.org/10.1016/j.eswa.2010.10.088

[7] S. Kardani Moghaddam, R. Buyya, and K. Ramamohanarao, "Performance anomaly detection using isolation trees in heterogeneous workloads of web applications in computing clouds," *Concurrency and Computation: Practice and Experience,* vol. 31, no. 20, p. e5306, 2019.

http://dx.doi.org/10.1002/cpe.5306

[8] M. Pradel, M. Huggler, and T. R. Gross, "Performance regression testing of concurrent classes," in *Proceedings of the 2014 International Symposium on Software Testing and Analysis*, pp. 13-25, 2014. https://doi.org/10.1145/2610384.2610393

[9] Martín, A.G., et al., *An approach to detect user behaviour anomalies within identity federations.* Computers & Security, p. 102356, 2021. https://doi.org/10.1016/j.cose.2021.102356

[10] ElSayed, M.S., et al., *A novel hybrid model for intrusion detection systems in SDNs based on CNN and a new regularization technique.* Journal of Network and Computer Applications, 191, p. 103160, 2021. https://doi.org/10.1016/j.jnca.2021.103160

[11] El-Shamy, A.M., et al., *Anomaly detection and bottleneck identification of the distributed application in cloud data center using software‐defined networking.* Egyptian Informatics Journal, 2021. https://doi.org/10.1016/j.eij.2021.01.001

[12] Nedelkoski, S., J. Cardoso, and O. Kao. *Anomaly detection from system tracing data using multimodal deep learning*. in *2019 IEEE 12th International Conference on Cloud Computing (CLOUD)*, IEEE, 2019. https://doi.org/10.1109/CLOUD.2019.00038

[13] Ahmed, T.M., et al. *Studying the effectiveness of application performance management (apm) tools for detecting performance regressions for web applications: an experience report*. in *2016 IEEE/ACM13th Working Conference on Mining Software Repositories (MSR)*. IEEE, 2016. https://doi.org/10.1145/2901739.2901774

[14] Ocariza Jr, F.S. and B. Zhao, *Localizing software performance regressions in web applications by comparing execution timelines.*Software Testing, Verification and Reliability, 31(5): p. e1750, 2021. https://doi.org/10.1002/stvr.1750

[15] Ahmad, T., D. Truscan, and I. Porres, *Identifying worst-case user scenarios for performance testing of web applications using Markov-chain workload models.* Future Generation Computer Systems, 87, p. 910-920, 2018. http://dx.doi.org/10.1016/j.future.2018.01.042

[16] Lin, J., P. Chen, and Z. Zheng. *Microscope: Pinpoint performance issues with causal graphs in micro-service environments*. in *International Conference on Service-Oriented Computing*, Springer, 2018. http://dx.doi.org/10.1007/978-3-030-03596-9_1

[17] Huang, S. and K. Lei, *IGAN-IDS: An imbalanced generative adversarial network towards intrusion detection system in ad-hoc networks.* Ad Hoc Networks, 105, p. 102177, 2020. https://doi.org/10.1016/j.adhoc.2020.102177

[18] Malik, H. and E.M. Shakshuki, *Classification of post-deployment performance diagnostic techniques for large-scale software systems.* Procedia Computer Science, 37, p. 244-251, 2014. http://dx.doi.org/10.1016/j.procs.2014.08.036

[19] Shi, J., G. He, and X. Liu. *Anomaly detection for key performance indicators through machine learning*. in *2018 International Conference on Network Infrastructure and Digital Content (IC-NIDC)*, IEEE, 2018. https://doi.org/10.1109/ICNIDC.2018.8525714

[20] Delgado Pérez, P., et al., *Performance mutation testing.* Software Testing, Verification and Reliability, 31(5), p. e1728, 2021. https://doi.org/10.1002/stvr.1728

[21] Oloieri, A. and P. Diac, *Throughput-based Service Composition.* Procedia Computer Science, 192, p. 1092-1101, 2021. https://doi.org/10.1016/j.procs.2021.08.112

[22] Böhmer, K. and S. Rinderle-Ma. *Multi instance anomaly detection in business process executions*. in *International Conference on Business Process Management*. 2017. Springer. http://dx.doi.org/10.1007/978-3-319-65000-5_5

[23] Kuang, K., et al., *Causal inference.* Engineering, 6(3): p. 253-263, 2020. https://doi.org/10.1016/j.eng.2019.08.016

[24] Xu, L. *Deep bidirectional intelligence: AlphaZero, deep IA-search, deep IA-infer, and TPC causal learning*. in *Applied Informatics*, Springer, 2018. https://doi.org/10.1186/s40535-018-0052-y

[25] Wu, L., et al. *Causal Inference Techniques for Microservice Performance Diagnosis: Evaluation and Guiding Recommendations*. in *ACSOS 2021-2nd IEEE International Conference on Autonomic Computing and Self-Organizing Systems*, 2021. https://hal.archives-ouvertes.fr/hal-03323055

[26] Hasnain, M., et al. *An efficient performance testing of web services*. in *2019 22nd International Multitopic*

Conference (INMIC), IEEE, 2019. http://10.1109/INMIC48123.2019.9022763

[27] R. Peng, Y.-F. Li, J.-G. Zhang, and X. Li, "A risk-reduction approach for optimal software release time determination with the delay incurred cost," *International Journal of Systems Science,* vol. 46, no. 9, pp. 1628-1637, 2015. https://doi.org/10.1080/00207721.2013.827261

[28] B. Rashidi and Q. Zhao, "Autonomous root-cause fault diagnosis using symbolic dynamic based causality analysis," *Neurocomputing,* vol. 401, pp. 10-27, 2020. https://doi.org/10.1016/j.neucom.2020.03.007

[29] M. Tang, Y. Jiang, J. Liu, and X. Liu, "Location-aware collaborative filtering for QoS-based service recommendation," in *2012 IEEE 19th international conference on web services*, IEEE, pp. 202-209, 2012. https://doi.org/10.1109/ICWS.2012.61

[30] U. Sivaji and P. S. Rao, "Test case minimization for regression testing by analyzing software performance using the novel method," *Materials Today: Proceedings,* 2021. https://doi.org/10.1016/j.matpr.2021.01.882

◐ Authors ◑

Muhammad Hasnain
2007 MCS, Arid agriculture university, Rawalpindi Pakistan
2016 Master in Computer Science (MSCS), Abasyn University, Islamabad Pakistan
2021 PhD in Information Technology, Monash University, Malaysia
February 2022~Present: Assistant Professor in Information Technology, Lahore Leads University Pakistan
Research Interests: Software Testing, Performance Testing, Software Scalability and Performance and Anomaly Detection
Email: drhasnain.it@leads.edu.pk

Muhammad Fermi Pasha
2010 PhD, Universiti Sains Malaysia, Malaysia
2016~Present: Senior Lecturer Monash University Malaysia
Research interests: Computational neuroimaging, intelligent network security traffic analysis, healthcare and radiology IT with emphasis on big data.
Email: muhammad.fermipasha@monash.edu

Imran Ghani
2002 MIT in CS, BIIT. Arid Agriculture University, Rawalpindi, Pakistan.
2007 M.S. in CS, Universiti Teknologi Malaysia
2010 Ph.D. in BIT, Kookmin University, South Korea
2021~Present: Associate Professor, Dept of Computer and Information Sciences, VMI, U.S.A
Research Interests: Software Engineering, Agile Software Development Methods, Secure Software Engineering, Software Testing
E-mail: ghanii@vmi.edu

◐ Authors ◑

Seung Ryul Jeong
1985 B.A. in Economics, Sogang Univ., Seoul, Korea
1989 M.S. in MIS, Univ. of Wisconsin - Milwaukee, WI, U.S.A.
1995 Ph.D. in MIS, Univ. of South Carolina, SC, U.S.A.
1997~Present: Professor, Graduate School of Business IT, Kookmin Univ., Korea
Research Interests: Machine Learning, Systems Implementation, Process Reengineering, Information Resource Management etc.
E-mail: srjeong@kookmin.ac.kr

Aitizaz Ali
2013 MS in Computer Science from Ghulam Ishaq Khan Institute of Engineering Sciences and Technology
2019~Present PhD scholar at School of Information Technology, Monash University Malaysia
Research interest: Networking, cyber security, and trust management
Email: Aitizaz.Ali@monash.edu