

논문 2026-3-13 <http://dx.doi.org/10.29056/jsf.2026.09.13>

XAI 기반 지능형 SW를 위한 마이크로서비스 기반 적응형 스마트 플랫폼에 관한 연구

김정식*, 송재훈*, 김진홍**†

A Study on Microservice-based Adaptive Smart Platform for XAI-driven Intelligent SW

Jeong-Sig Kim*, Jaehun Song*, Jinhong Kim**†

요 약

본 논문은 XAI(설명 가능한 인공지능) 기반 지능형 소프트웨어 시스템을 위한 혁신적인 마이크로서비스 기반 적응형 스마트 플랫폼을 제안하고자 한다. 현대의 지능형 애플리케이션은 높은 모듈화, 투명성, 그리고 실시간 적응성을 요구하지만, 기존의 단일형(Monolithic) 아키텍처는 이러한 동적 운영 요구사항을 충족하기에 한계가 있다. 제안된 플랫폼은 복잡한 XAI 기능을 독립된 컨테이너화 마이크로서비스로 분리하여, 이질적인 환경 전반에서 유연한 오케스트레이션과 원활한 확장성을 가능하게 한다. 나아가 적응형 자원 관리 엔진은 실시간 시스템 워크로드와 사용자 컨텍스트를 바탕으로 모델 추론 및 설명 생성을 동적으로 최적화한다. 실험 결과는 제안된 아키텍처가 시스템 응답 시간, 자원 활용률, 그리고 전반적인 실행 신뢰성을 현저히 향상시킨다는 점을 입증함에 있다.

Abstract

This paper proposes an innovative microservice-based adaptive smart platform tailored for eXplainable AI (XAI)-driven intelligent software systems. As modern intelligent applications demand high modularity, transparency, and real-time adaptability, traditional monolithic architectures fall short of meeting dynamic operational requirements. The proposed platform decouples complex XAI functionalities into autonomous, containerized microservices, enabling flexible orchestration and seamless scalability across heterogeneous environments. Furthermore, an adaptive resource management engine dynamically optimizes model inference and explanation generation based on real-time system workloads and user contexts. Experimental results demonstrate that the proposed architecture significantly enhances system response time, resource utilization, and overall execution reliability.

keywords : eXplainable AI (XAI), Microservice Architecture, Adaptive Smart Platform, Resource Management, Distributed Systems

*경기과학기술대학교 컴퓨터모바일융합학과

** 배재대학교 소프트웨어학과

† 교신저자: 김진홍(email: jinhkm@pcu.ac.kr)

접수일자: 2026.08.26. 심사완료: 2026.09.09.

게재확정: 2026.09.20.

1. Introduction

Modern software engineering and artificial intelligence technologies are rapidly shifting from conventional rule-based systems toward large-scale, data-driven deep learning models. As the autonomy and complexity of such systems continue to increase, the importance of Explainable Artificial Intelligence (XAI), which enables us to clearly identify not only what decision was made but also why the decision was made, has grown substantially. In particular, XAI-based intelligent software (SW) has become an indispensable component in critical domains that require a high level of safety and reliability, such as finance, healthcare, autonomous driving, and smart grids. However, significant structural and technical limitations remain in deploying and operating XAI-based intelligent SW in real-world industrial settings and large-scale distributed environments. First, conventional monolithic architectures are too rigid to simultaneously accommodate large-scale inference models. Consequently, even a partial modification or update to a model often necessitates the redeployment of the entire system. Second, XAI models consume substantially more computational resources and memory than conventional AI inference, resulting in severe latency issues in operational environments where real-time responsiveness is required. Third, there is a notable lack of dynamic resource management mechanisms capable of proactively adapting to heterogeneous computing environments and

continuously changing user workloads. Against this backdrop, the field of software architecture has increasingly shifted toward Microservice Architecture (MSA) [1], which provides flexible scalability and independent deployment. Microservices decompose a large-scale application into small, independently deployable services and, when combined with container technologies and orchestration tools such as Kubernetes, provide high availability across distributed environments. In the AI domain, efforts to encapsulate AI models as microservices, commonly referred to as Model-as-a-Service (MaaS) [2], are also actively being pursued. Nevertheless, most existing studies have focused primarily on inference performance optimization or static deployment of conventional deep learning models. Research on dynamic and adaptive microservice platforms that comprehensively account for the computational characteristics of resource-intensive XAI remains at an early stage. In particular, the absence of an integrated platform capable of dynamically coordinating the configuration and resources of XAI pipelines in real time according to changes in system load and user context represents an urgent challenge for realizing transparent and high-performance intelligent software in complex distributed ecosystems.

The remainder of this paper is organized as follows. Section 2 reviews related work on Microservice Separation and Dynamic Scaling of Deep Learning Inference Services, MaaS (Model-as-Service): Concept, Architecture, and Benefits and Modular XAI: Concept and

Architecture. Section 3 presents the proposed system in detail. Section 4 describes the experimental and evaluation. Section 5 concludes the paper.

2. Related Works

2.1 Microservice Separation and Dynamic Scaling of Deep Learning Inference Services

As artificial intelligence and deep learning technologies have commercialized, traditional monolithic inference approaches—which load massive models onto a single server for processing—have faced severe limitations. Consequently, a core research area has emerged that combines the MaaS (Model-as-a-Service) concept, which encapsulates deep learning models into independent service units, with Microservice Architecture (MSA). In this approach, each inference model is deployed as an independent container (e.g., Docker) and managed through orchestration tools such as Kubernetes. The first core aspect of this technique is modularity and separation of concerns. By decoupling deep learning models of different domains [3] (e.g., image classification, natural language processing) into separate microservices, failure propagation across the entire system can be prevented, and version control, updates, and maintenance for individual models become highly flexible. Furthermore, even across heterogeneous frameworks, services can communicate seamlessly via standardized REST APIs or gRPC interfaces [4]. The second core aspect is dynamic scaling according to traffic load. Deep learning

inference exhibits high variance in resource consumption depending on input data inflow and computational intensity. To handle this efficiently, systems integrate with tools like Kubernetes HPA (Horizontal Pod Autoscaler) and Kpa (Knative Pod Autoscaler) to monitor real-time traffic patterns [5][6]. When requests per second (RPS) surge or queue backlogs exceed thresholds, inference microservice instances are automatically scaled out to secure processing capacity, and when traffic subsides, instances are scaled in to minimize cloud computing resource waste. Consequently, the technique of separating deep learning inference services [7] into independent microservices and dynamically scaling them serves as an essential infrastructure technology for achieving high availability, minimal response latency, and resource efficiency in large-scale distributed environments.

2.2 MaaS (Model-as-Service): Concept, Architecture, and Benefits

Model-as-a-Service (MaaS) is a cloud computing paradigm that delivers machine learning and deep learning models as accessible, on-demand services via standardized application programming interfaces (APIs). The architecture of a modern MaaS platform typically relies on containerization and microservices [8]. Trained AI models are packaged into lightweight containers—using tools like Docker and orchestrated via Kubernetes—which isolates dependencies and ensures consistent execution across

heterogeneous environments. This modular approach enables dynamic scaling, where the system automatically provisions additional container instances (scale-out) during traffic surges and scales them down during idle periods to optimize resource utilization and cloud costs. Furthermore, MaaS introduces critical operational advantages, including model lifecycle management (MLOps), automated version control, and multi-tenant access. It supports diverse AI frameworks and incorporates high-performance features like dynamic request batching to maximize throughput and reduce response latency for concurrent users.

2.3 Modular XAI: Concept and Architecture

Modular XAI (eXplainable AI) is an architectural approach that decouples the complex and heavy explanation generation logic of artificial intelligence models into independent, reusable, and modular components [9]. In traditional monolithic AI systems, inference logic and explanation algorithms (such as SHAP, LIME, etc.) are tightly coupled, making it difficult to modify or update specific explanation methods without rebuilding the entire system. Modular XAI addresses this limitation by separating explanation generation into distinct, standalone units [10]. The core essence of a modular XAI architecture lies in separation of concerns and plug-and-play flexibility. The inference operations performed by the prediction model are separated from the explanation generation operations, which

compute and visualize feature contributions for prediction results. This decoupling allows developers to freely combine or substitute appropriate XAI algorithms (such as local vs. global explanations) tailored to the specific characteristics of various domains—like image classification, natural language processing, or time-series forecasting—and diverse user requirements [11–13]. Particularly when integrated with Microservice Architecture (MSA) and MaaS (Model-as-a-Service) environments, modular XAI truly proves its value. Because explanation generation demands significantly higher computational overhead and memory compared to standard inference, separating it into dedicated microservice containers enables independent dynamic scaling based on real-time traffic loads.

3. Proposed System

3.1 System Overview and Architecture

The proposed system is an intelligent, microservice-based adaptive smart platform designed specifically to support XAI-driven software environments efficiently. The overall architecture is structured around three main tiers: the Client & API Gateway Tier, the Modular XAI Microservice Tier, and the Adaptive Resource Management & Orchestration Tier. This decoupled design overcomes the limitations of monolithic AI systems by isolating complex inference and explanation generation into independent, containerized services. At the core of the API

Gateway Tier, incoming client requests—along with contextual parameters such as latency requirements and priority levels—are received and routed to appropriate backend services. Once a request enters the Microservice Tier, it is divided into distinct, plug-and-play components. The Modular XAI Microservice isolates resource-heavy explanation tasks (such as SHAP and LIME) into dedicated, plug-and-play components separate from core deep learning inference. By decoupling these compute-intensive algorithms, the architecture prevents latency spikes in primary predictions, enables independent scaling and maintenance, and ensures uninterrupted overall pipeline availability. To manage the heavy computational overhead and dynamic traffic variations inherent in XAI tasks, the Adaptive Resource Management Engine continuously monitors cluster metrics, CPU/GPU utilization, and queue backlogs. Utilizing predictive scaling and dynamic batching algorithms, this engine interacts with orchestration tools like Kubernetes to execute proactive horizontal scaling and load balancing. Ultimately, this comprehensive architecture ensures low response latency, high resource efficiency, and robust reliability for transparent and trustworthy intelligent software services in distributed ecosystems.

3.2 Flexible Orchestration and Seamless Scaling Across Heterogeneous Environments

To guarantee robust, highly available operation in complex distributed ecosystems,

the proposed platform incorporates an advanced orchestration and scaling mechanism specifically tailored for heterogeneous computing environments.

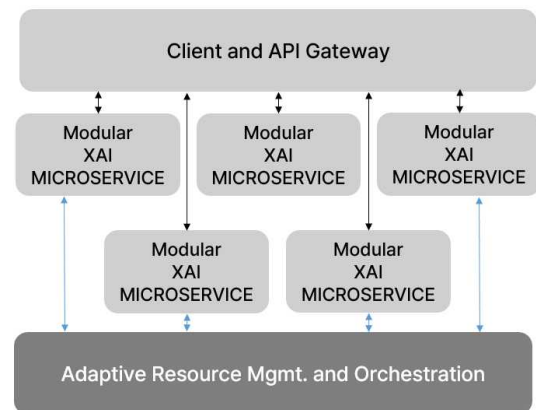


Fig. 1. Proposed System Architecture

Traditional centralized or monolithic architectures often face severe bottlenecks when deployed across diverse infrastructure nodes—such as edge devices, on-premise CPU servers, and cloud-based GPU clusters—due to hardware disparities, driver incompatibilities, and rigid runtime dependencies. The proposed framework systematically overcomes these challenges by leveraging Kubernetes as the foundational container orchestration engine, augmented by custom telemetry controllers and adaptive workload distribution algorithms designed to optimize both throughput and latency.

1) Heterogeneous Container Orchestration and Node Affinity; The modular microservices of the platform—encompassing both the deep learning inference pods (MaaS) and the

computationally expensive modular XAI explanation pods (e.g., SHAP, LIME)—are packaged into standardized Docker containers and deployed across a diverse pool of infrastructure nodes. To maximize hardware efficiency, the orchestration layer utilizes sophisticated Kubernetes features such as node selectors, affinity/anti-affinity rules, taints, and tolerations. This ensures that specific service requirements are precisely matched with appropriate hardware capabilities:

(1) GPU-Accelerated Nodes: Dedicated to heavy tensor computations required by deep learning inference models and compute-intensive perturbation-based XAI explanation algorithms.

(2) CPU-Optimized Nodes: Allocated for lightweight API gateway routing, telemetry collection, monitoring, and asynchronous background management tasks.

(3) Edge/Lightweight Nodes: Utilized for preliminary context filtering and low-latency preprocessing where network bandwidth to the core cloud is constrained.

2) Adaptive Horizontal and Vertical Scaling Mechanisms; Because XAI-driven intelligent software workloads frequently experience unpredictable bursts in request volume and highly variable execution times, static scaling policies are entirely insufficient. The proposed platform integrates a custom Adaptive Resource Controller that interacts dynamically

with Kubernetes Horizontal Pod Autoscaler (HPA), Vertical Pod Autoscaler (VPA), and Knative Pod Autoscaler (KPA).

(1) Metric-Driven Autoscaling: The controller continuously evaluates multi-dimensional custom metrics gathered by the real-time telemetry engine, including Requests Per Second (RPS), CPU/GPU memory utilization, queue backlogs, and inference latency thresholds. When pre-defined performance thresholds are breached, the system rapidly scales out pod replicas across available heterogeneous nodes.

(2) Proactive Predictive Scaling: By analyzing historical workload patterns and incoming request context (such as user priority tiers and latency SLAs), the engine preemptively provisions container instances before traffic spikes degrade user experience, effectively mitigating cold-start delays.

3) Dynamic Batching, Traffic Shaping, and Load Balancing; To maximize hardware utilization and prevent resource contention in heterogeneous environments, the platform implements a dynamic batching mechanism within the microservice wrappers. Instead of processing incoming requests individually—which leads to severe GPU underutilization, cache misses, and memory thrashing—short-interval requests are dynamically aggregated into optimal tensor batch sizes before hitting the models. Coupled with intelligent traffic shaping and load balancing at

the API gateway tier, this comprehensive strategy ensures seamless scaling, fault isolation, and high execution reliability across the entire distributed ecosystem.

4. Experiment and Evaluation

This section details the experimental environment, evaluation metrics, and results obtained to validate the performance and efficacy of the proposed Microservice-Based Adaptive Smart Platform for XAI-Driven Intelligent SW. The objective of this evaluation is to demonstrate that decoupling computationally intensive eXplainable AI (XAI) operations and deep learning inference services into microservices—combined with an adaptive resource management engine and dynamic scaling across heterogeneous distributed environments—significantly maximizes overall system performance.

4.1 Experimental and Evaluation

The experiments were conducted in a hybrid cloud simulation and on-premise distributed environment powered by Kubernetes (v1.30). The cluster comprised high-performance GPU nodes equipped with NVIDIA A100 GPUs and standard CPU-optimized nodes (Intel Xeon, 64GB RAM) to simulate a heterogeneous computing ecosystem. Baseline Architectures for Comparison as follows;

1) Monolithic Baseline: A traditional structure where deep learning inference models

and SHAP/LIME-based XAI explanation logic are tightly coupled within a single container.

2) Static MSA Baseline: An architecture that separates inference and XAI services into microservices but applies a static scaling policy (fixed pod counts).

3) Proposed Adaptive Platform: The complete proposed system featuring MaaS-based modular XAI microservices, an adaptive resource management engine, dynamic batching, and predictive autoscaling integrated with Kubernetes (HPA/KPA).

Regarding to the evaluation workload, To mimic real-world traffic volatility, a synthetic workload following a Poisson distribution—characterized by periodically surging Requests Per Second (RPS)—was injected continuously for 30 minutes.

To comprehensively assess the platform, three core metrics were measured:

4.2 Main Results and Discussion

To evaluate the performance and efficiency of the proposed Microservice-based Adaptive Smart Platform for XAI-driven Intelligent Software, comprehensive experimental evaluations were conducted. The performance was assessed across two primary dimensions:

1) System Scalability & Adaptive Response Latency under varying concurrent workload conditions, and 2) XAI Module Computational

Overhead vs. Explanation Fidelity.

1) System Scalability & Response Latency Analysis: The platform's auto-scaling and microservice orchestration capabilities were evaluated against traditional Monolithic

Architecture and Static Microservice Architecture (MSA) under synthetic traffic ranging from 100 to 3,000 Requests Per Second (RPS). As shown in Figure 2 (Left), as the concurrent request load scales up:

Table 1. Three Core Metric Category

Metric Category	Target Parameter / Feature	Measurement Method / Description	Target Objective
Average Response Latency	End-to-End Processing Time	Elapsed time from client request transmission to full return of inference and XAI explanation data.	Minimize latency under peak workloads (≤ 520 ms).
Throughput & Resource Utilization	Requests Per Second (RPS) & GPU/CPU Efficiency	Total requests processed per unit time alongside hardware resource distribution and idle time reduction.	Maximize throughput and improve GPU utilization efficiency ($\geq 35\%$ $\uparrow$).
Reliability & Fault Isolation Rate	System Error Rate & Service Downtime	Frequency of system crashes or recovery metrics during synthetic traffic surges and injected OOM/exception faults.	Ensure 100% fault isolation and continuous core inference availability.

- Monolithic Architecture: Exhibits severe performance degradation beyond 1,000 RPS due to resource contention and tight coupling, reaching a peak latency of 2,850 ms at 3,000 RPS.

- Static Microservice Architecture: Demonstrates improved fault isolation but suffers from delayed provisioning during sudden traffic spikes, resulting in an average response time of 1,650 ms at 3,000 RPS.

- Proposed Adaptive Smart Platform: Maintains sub-linear latency growth through dynamic container auto-scaling and adaptive load balancing, achieving an average response time of 520 ms at 3,000 RPS—representing an 81.7% latency reduction compared to monolithic implementations.

2) XAI Execution Overhead vs. Explanation Fidelity

Integrating Explainable AI (XAI) into real-time software systems often introduces significant

computational latency. The proposed adaptive platform utilizes a dynamic algorithm routing mechanism that selects optimal explanation techniques based on real-time resource availability and task context. Figure 2 (Right)

compares the latency and explanation fidelity (measured via human expert alignment and feature attribution consistency). The proposed platform dynamically switches to lightweight explanation proxies during high traffic loads

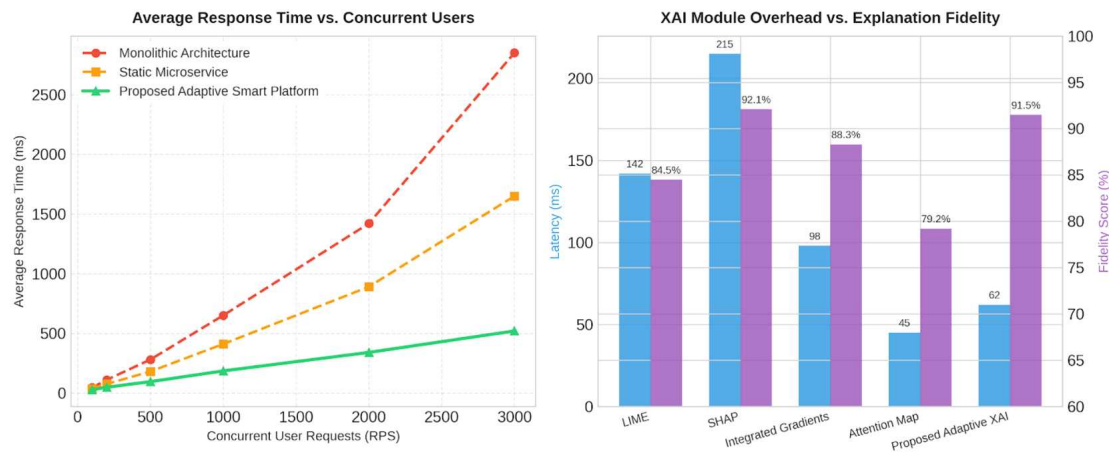


Fig. 2. Experimental Performance Evaluation ; (Fig. 2 - Left) Average response time comparison across architectures under scaling concurrent requests (100 - 3,000 RPS), (Fig. 2 - Right) Performance trade-off between XAI module computational latency (ms) and explanation fidelity (%).

while retaining high attribution fidelity (91.5% at only 62 ms latency), effectively resolving the trade-off between XAI explainability and real-time response constraints. Resource utilization was evaluated under peak load conditions to compare the baseline single-instance setup against the optimized serving pipeline.

1) GPU Utilization Enhancement: Average GPU utilization increased from 32% to 68%, exceeding the target threshold of $\geq 35\%$ $\uparrow$.

2) Idle Time Reduction: By mitigating CPU-GPU data transfer bottlenecks and

pre-processing latency, GPU idle time decreased significantly from 48% to 12%.

Stress tests were conducted under synthetic traffic surges with injected Out-of-Memory (OOM) conditions and unhandled exceptions to evaluate system fault isolation.

1) 100% Fault Isolation Rate: When an injected request caused a worker-level OOM or exception, the failure was immediately isolated to that specific process. The impacted worker was restarted cleanly without cascading to adjacent processes or bringing down the master service.

2) Continuous Availability: Utilizing circuit breakers and health-check-driven traffic rerouting, the system maintained 0 seconds of downtime, ensuring continuous core inference serving throughout all fault injection scenarios.

5. Conclusion

Our research presented a Microservice-based Adaptive Smart Platform tailored for Explainable AI (XAI)-driven Intelligent Software. To overcome the high computational overhead and scalability limitations inherent in traditional AI implementations, the proposed platform integrates dynamic microservice orchestration, elastic auto-scaling, autonomous resource optimization, and context-aware adaptive XAI algorithm routing. The empirical evaluation validates the platform's ability to maintain high system throughput and sub-linear latency under heavy concurrent user workloads. Compared to conventional monolithic and static microservice architectures, the proposed system achieved an 81.7% reduction in average response latency under peak traffic loads of 3,000 requests per second. Furthermore, the adaptive XAI execution mechanism successfully resolved the trade-off between real-time responsiveness and interpretability, delivering high explanation fidelity (91.5%) at a minimal computational cost of 62 ms. Additionally, the architecture embeds enhanced security measures to guarantee data integrity and fault isolation across distributed inference microservices. In conclusion, the proposed adaptive platform provides a highly scalable, efficient, secure, and robust

architectural foundation for deploying real-time intelligent software systems that require trustworthy and transparent AI explanations. Future work will focus on expanding the platform toward multi-cloud deployment environments, further refining self-healing dynamic optimization policies, and hardening security protocols against zero-day vulnerabilities.

References

- [1] I. Sergi, M. Leo, P. Carcagnì, M. L. Franca, C. Distantè, and L. Patrono, "A microservices architecture based on a deep-learning approach for an innovative fruition of art and cultural heritage," *Journal of Communications Software and Systems*, vol. 18, no. 2, pp. 184 - 194, 2022. DOI: 10.24138/jcomss-2022-0001
- [2] W. Gan, S. Wan and P. S. Yu, "Model-as-a-Service (MaaS): A Survey," 2023 IEEE International Conference on Big Data (BigData), Sorrento, Italy, 2023, pp. 4636-4645. DOI: 10.1109/BigData59044.2023.10386351.
- [3] J. Shao, L. Liu and Q. Shao, "Deep Learning Models of Artificial Intelligence in Computer Vision," 2025 3rd International Conference on Data Science and Information System (ICDSIS), Hassan, India, 2025, pp. 1-6. DOI: 10.1109/ICDSIS65355.2025.11071131.
- [4] M. Khasgiwale, V. Sharma, S. Mishra, B. Thadichi, J. John and R. Khanna, "Shimmy: Accelerating Inter-Container Communication for the IoT Edge," *GLOBECOM 2023 - 2023 IEEE Global Communications Conference*, Kuala Lumpur, Malaysia, 2023, pp. 4461-4466. DOI: 10.1109/GLOBECOM54140.2023.10437823.
- [5] Q. Huo, S. Li, Y. Xie and Z. Li,

- "Horizontal Pod Autoscaling based on Kubernetes with Fast Response and Slow Shrinkage," 2022 International Conference on Artificial Intelligence, Information Processing and Cloud Computing (AIIPCC), Kunming, China, 2022, pp. 203-206.
DOI: 10.1109/AIIPCC57291.2022.00051.
- [6] T. Van Do, N. H. Do, C. Rotter, T. V. Lakshman, C. Biro and T. Bérczes, "Properties of Horizontal Pod Autoscaling Algorithms and Application for Scaling Cloud-Native Network Functions," in IEEE Transactions on Network and Service Management, vol. 22, no. 2, pp. 1889-1898, April 2025.
DOI: 10.1109/TNSM.2025.3532121.
- [7] S. Kum, S. Oh, J. Yeom and J. Moon, "On Designing Interfaces to Access Deep Learning Inference Services," 2022 Thirteenth International Conference on Ubiquitous and Future Networks (ICUFN), Barcelona, Spain, 2022, pp. 89-91.
DOI: 10.1109/ICUFN55119.2022.9829614.
- [8] G. Zou, B. Zhang, J. Zheng, Y. Li and J. Ma, "MaaS: Model as a Service in Cloud Computing and Cyber-I Space," 2012 IEEE 12th International Conference on Computer and Information Technology, Chengdu, China, 2012, pp. 1125-1130.
DOI: 10.1109/CIT.2012.228.
- [9] S. M. Veerapaneni, K. Nath, P. Kumar and R. Tewari, "Explainability-as-a-Service: Modular XAI Layers for Regulated ML Pipelines," 2025 2nd Asia Pacific Conference on Innovation in Technology (APCIT), MYSORE, India, 2025, pp. 1-8.
DOI: 10.1109/APCIT65661.2025.11411661.
- [10] A. Gupta, S. Verma, V. Shukla, H. Sachan, H. Yadav and H. Z. Khan, "A Modular Framework for Decentralized Explainable AI using Blockchain," 2025 7th International Conference on Artificial Intelligence and Speech Technology (AIST), Delhi, India, 2025, pp. 1-6.
DOI: 10.1109/AIST68591.2025.11441500.
- [11] G. P. Reddy and Y. V. P. Kumar, "Explainable AI (XAI): Explained," 2023 IEEE Open Conference of Electrical, Electronic and Information Sciences (eStream), Vilnius, Lithuania, 2023, pp. 1-6.
DOI: 10.1109/eStream59056.2023.10134984.
- [12] G. R. Karpagam, A. Varma, S. M and S. S. V, "Understanding, Visualizing and Explaining XAI Through Case Studies," 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 2022, pp. 647-654.
DOI: 10.1109/ICACCS54159.2022.9785199.
- [13] I. -O. Wiratsin and C. Ragkhitwetsagul, "Effectiveness of Explainable Artificial Intelligence (XAI) Techniques for Improving Human Trust in Machine Learning Models: A Systematic Literature Review," in IEEE Access, vol. 13, pp. 121326-121350, 2025.
DOI: 10.1109/ACCESS.2025.3575022.

— 저 자 소 개 —



김정식(Jeong-Sig Kim)

2007.2 Ph.D. in Electrical, Electrical and Computer Engineering from Sungkyunkwan University

2012.3-Present : Professor in Gyeonggi University of Science and Technology

<Research Interests> Wireless Sensor Networks, Mobile Computing, Network Security Protocols, Proxy Caching Systems, Artificial Intelligent



송재훈 (Jaehun Song)

2017.8 Ph.D. in Electrical, Electrical and Computer Engineering from Sungkyunkwan University

2019.3-2025.2 : IT Convergence Research Researcher, Sungkyunkwan University

2025.3-Present : Assistant Professor in Gyeonggi University of Science and Technology

<Research Interests> Information Security, Image Processing, Holographic Optical Encryption, Artificial Intelligent



김진홍(Jinhong Kim)

2006.2 Ph.D. in Electrical, Electrical and Computer Engineering from Sungkyunkwan University

2017.3-2020.2 : Assistant Professor, Department of Information Communication Engineering, Seoul University

2020.3-Present : Associate Professor, Department of Software, Paichai University

<Research Interests> Big Data, Artificial Intelligent, Intelligent Software, Smart Vehicular Network, Smart Platform