Preprints.org

Article

Not peer-reviewed version

Scaling Generative AI for Self-Healing DevOps Pipelines: Technical Analysis

Bipul Bhattarai *

Posted Date: 17 June 2025

doi: 10.20944/preprints202506.1436.v1

Keywords: Generative AI; DevOps; Self-Healing Systems; Multi-Agent Orchestration; LLM-Based Log Parsing; Autonomous Remediation; CI/CD Pipelines; AIOps; Infrastructure Automation; Machine Learning Operations

Preprints.org is a free multidisciplinary platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This open access article is published under a Creative Commons CC BY 4.0 license, which permit the free download, distribution, and reuse, provided that the author and preprint are cited in any reuse.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 June 2025 doi:10.20944/preprints202506.1436.v1

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Article

Scaling Generative AI for Self-Healing DevOps Pipelines: Technical Analysis

Bipul Bhattarai

Independent Researcher; bipulbhattarai70@gmail.com

Abstract: This comprehensive technical analysis examines the emerging field of AI-driven self-healing DevOps pipelines, focusing on the architectural implementation, multi-agent orchestration systems, and governance mechanisms that enable autonomous infrastructure management. The study analyzes breakthrough advancements in LLM-based log parsing frameworks achieving 98% precision in root-cause analysis, sophisticated multi-agent remediation systems demonstrating 5.76x performance improvements over traditional approaches, and robust governance architectures with confidence-based decision making at 0.85 thresholds. The analysis reveals that modern self-healing systems employ sophisticated detection stages utilizing LogParser-LLM frameworks processing 3.6 million logs with minimal LLM invocations, while maintaining 90.6% F1 scores for grouping accuracy. Multi-agent orchestration patterns leverage specialized agents across functional domains with hierarchical communication protocols, implementing event-driven workflows and state machine orchestration for distributed transaction management. Governance mechanisms integrate policy engines with blast radius controls, automated audit trails, and LLM-generated natural-language rationales for explainable AI decision-making. Empirical validation demonstrates significant operational improvements including 55% reduction in Mean Time to Recovery (MTTR), 208x increase in code deployment frequency for DevOps-mature organizations, and over 90% developer trust retention across enterprise implementations. The market evolution shows exceptional growth from $942.5 million in 2022 to projected $22.1 billion by 2032, with 74% organizational DevOps adoption and 51% code copilot utilization representing the highest AI tool adoption rates. Integration with modern cloud platforms including AWS SageMaker, Kubernetes orchestration, and Terraform infrastructure-as-code demonstrates mature production-ready implementations. The analysis connects theoretical frameworks to practical deployments across major enterprise environments, revealing standardized multi-agent communication protocols and sophisticated resilience patterns including circuit breakers, retry mechanisms with exponential backoff, and graceful degradation capabilities. The study concludes that AI-driven self-healing DevOps represents a paradigm shift from reactive to predictive infrastructure management, with proven capabilities for transforming software delivery processes through autonomous anomaly detection, intelligent remediation, and comprehensive governance frameworks that ensure safety, explainability, and regulatory compliance in enterprise-scale deployments.

Keywords: Generative AI; DevOps; Self-Healing Systems; Multi-Agent Orchestration; LLM-Based Log Parsing; Autonomous Remediation; CI/CD Pipelines; AIOps; Infrastructure Automation; Machine Learning Operations

---

Executive Summary

The field of AI-driven self-healing DevOps represents a paradigm shift from reactive to predictive infrastructure management, with generative AI enabling sophisticated multi-agent orchestration for automated remediation. Current implementations demonstrate 98% precision in root-cause analysis and 55% MTTR reduction through sophisticated detection stages, multi-agent

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 June 2025 doi:10.20944/preprints202506.1436.v1

remediation workflows, and robust governance frameworks. The market, valued at $942.5 million in 2022, is projected to reach $22.1 billion by 2032, reflecting the transformative potential of these technologies.

This analysis examines the technical architecture, multi-agent systems, governance mechanisms, and empirical validation methodologies that enable enterprise-scale self-healing DevOps pipelines, connecting theoretical frameworks to practical implementations across major cloud platforms and DevOps toolchains.

Technical Deep-Dive: Architecture and Implementation

Detection Stage with LLM-Based Log Parsing

Modern LLM-based log parsing has achieved breakthrough performance levels through frameworks like LogParser-LLM [1], which processes 3.6 million logs with only 272.5 LLM invocations while maintaining 90.6% F1 score for grouping accuracy. The detection architecture employs several sophisticated components:

Advanced Parsing Techniques: The **LogBatcher framework** [2] implements clustering-based log batching with cache matching to optimize LLM inference overhead, eliminating the need for training processes or labeled demonstrations. This approach supports high-scale production environments where traditional rule-based parsing fails to adapt to evolving log formats.

Intelligent Metrics Analysis: **AWS SageMaker's Random Cut Forest algorithm** [6] establishes baseline "normal" pipeline behavior through automated anomaly detection with 1-hour time windows. The system integrates with CloudWatch Logs and S3 for comprehensive data storage and analysis, collecting custom metrics including build duration, test failure rates, and infrastructure usage patterns.

Real-Time Anomaly Detection: The **Grafana-Prometheus framework** [8] uses z-score formulas with 3-sigma rules for anomaly detection, implementing recording rules with average and standard deviation calculations over time. The system applies smoothing and high-pass filtering for robustness while supporting seasonality patterns through custom pattern addition.

Multi-Agent GenAI Orchestration in Remediation

The remediation stage employs sophisticated multi-agent orchestration patterns that represent a significant advancement from single-agent approaches. **CrewAI demonstrates 5.76x faster execution** [3] than traditional LangGraph implementations through role-based architecture that mimics human organizational structures.

Agent Specialization Architecture: Modern implementations deploy **specialized agents across functional domains** - Detection Agents for vulnerability scanning and anomaly detection, Analysis Agents for root cause analysis and impact assessment, Remediation Agents for automated fixing and patch deployment, and Validation Agents for testing fixes and compliance checking.

Communication Protocols: The emerging **Model Context Protocol (MCP)** by Anthropic provides JSON-RPC client-server interfaces for secure tool invocation, while **Google's Agent-to-Agent Protocol (A2A)** [3] enables peer-to-peer task outsourcing with capability-based Agent Cards. These protocols support both synchronous REST/gRPC communication for immediate coordination and asynchronous message queuing for distributed tasks.

Orchestration Frameworks: **Azure AI Foundry's Agent Service** employs hierarchical orchestration with manager agents overseeing specialized agents, while **Semantic Kernel's Magentic-One pattern** uses dedicated orchestrators that dynamically select agents based on context and capabilities. These frameworks maintain shared context, track progress, and adapt workflows in real-time.

Human-in-the-Loop Implementation with Confidence Thresholds

The 0.85 confidence threshold represents industry-standard practice for critical decision-making systems [12], implemented through sophisticated policy engines that combine multiple validation mechanisms. Expectation-Maximization (EM) algorithms optimize threshold estimation, while sigmoid units provide normalized confidence scores for binary predictions.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 June 2025 doi:10.20944/preprints202506.1436.v1

Technical Implementation: **Worker-in-the-loop deployment platforms** [11] feature adjustable confidence sliders that provide quality-speed tradeoff controls with automatic fallback to human review. The system implements **LangGraph integration** for human interrupts in AI agent workflows, supporting HumanInterruptConfig with allow_accept, allow_edit, and allow_respond options.

Risk Assessment Integration: The confidence threshold system integrates with **multi-stage approval processes** featuring role-based access controls and automated escalation based on confidence scores and risk assessment. Financial risk assessment for critical data accuracy, exception handling workflows for low-confidence predictions, and workforce efficiency monitoring ensure comprehensive governance.

Architecture Blueprint: Kubernetes, Jenkins, and Terraform Integration

The architecture blueprint demonstrates sophisticated integration patterns across modern DevOps toolchains. **AWS Controllers for Kubernetes (ACK)** [6] provide frameworks for building custom controllers that communicate with AWS services, enabling SageMaker training job orchestration through Kubernetes APIs with JSON pipeline definitions stored and versioned in S3.

Jenkins Pipeline Enhancement: **AI-enhanced testing frameworks** implement self-healing test automation with ML-based element identification, featuring automatic test script updates when UI elements change and AI-powered test case generation from requirements analysis. The system integrates with tools like Testim and Mabl for intelligent test maintenance.

Terraform Integration: **AI-driven infrastructure generation** leverages generative AI for Terraform script creation and optimization, with GitHub Copilot and CodeWhisperer integration providing real-time suggestions. **Spacelift automated drift detection** with cron job scheduling enables reconcile mode for automatic drift remediation, while state file analysis detects infrastructure inconsistencies.

OpenAI-Compatible APIs and AWS SageMaker Implementation

The implementation architecture supports comprehensive LLM integration through **Azure OpenAI Service** offering GPT-4, GPT-3.5-turbo, and Codex access via REST APIs with Python SDK and web-based Azure OpenAI Studio interfaces. The system implements pay-as-you-go pricing with PTU (Provisioned Throughput Units) for predictable workloads.

SageMaker Deployment Patterns: **Multi-model endpoints** [7] achieve 50% average cost reduction through efficient resource utilization, while **Elastic Inference** provides cost-effective GPU splitting. The architecture supports **A/B testing capabilities** with traffic proportioning and VPC deployment with AWS PrivateLink for security.

Multi-Agent System Deep Analysis

Agentic Workflow Architecture

The **multi-agent system implements sophisticated workflow patterns** that represent a fundamental shift from monolithic automation to distributed intelligence. **Hierarchical orchestration** employs manager agents that oversee specialized agents, maintaining shared context and adapting workflows in real-time based on system state and performance metrics.

Agent Communication Mechanisms: The system implements **event-driven communication patterns** where agents react to system events and state changes, providing loose coupling between components for scalability and resilience. **State machine orchestration** ensures well-defined states and transitions with predictable behavior and support for rollback and recovery operations.

Workflow Coordination: **The Saga pattern** manages sequences of local transactions with compensation, handling distributed transaction management with automatic rollback on failure scenarios. This approach proves essential for complex DevOps workflows where partial failures require sophisticated recovery mechanisms.

Comparison with AutoDevOps Approaches

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 June 2025 doi:10.20944/preprints202506.1436.v1

Traditional AutoDevOps implementations rely on single-pipeline automation with predetermined stages and limited adaptability to changing requirements. **Multi-agent AutoDevOps in 2024-2025** [18] features specialized agents for different DevOps stages (build, test, deploy, monitor) with dynamic adaptation based on project characteristics and intelligent error recovery capabilities.

Advanced Implementations: **AWS Multi-Agent Orchestrator** [9] manages multiple AI agents with intelligent routing, supporting both streaming and non-streaming responses across various deployment environments. **Microsoft Azure AI Foundry's Agent Service** supports production-grade multi-agent systems with Semantic Kernel and AutoGen integration, including Red Teaming Agents for automated security testing.

Error Handling and Resilience Patterns

The system implements comprehensive resilience patterns including **Circuit Breaker mechanisms** that prevent cascade failures by temporarily disabling failing agents, and **Retry with Exponential Backoff** for automatic retry of failed operations with increasing delays. **Bulkhead patterns** isolate agent failures to prevent system-wide impact.

Recovery Strategies: **Agent Substitution** automatically replaces failed agents with backup instances, while **Graceful Degradation** reduces functionality while maintaining core operations. **Self-Healing capabilities** enable agents to automatically detect and correct their own issues, with **Human-in-the-Loop escalation** for complex failures requiring human intervention.

Governance Mechanisms and Policy Engine Analysis

Policy Engine with Confidence Thresholds

The policy engine implements sophisticated decision boundary algorithms using G-Eval frameworks for subjective evaluations and DAG (Directed Acyclic Graph) scorers for clear success criteria. Multi-criteria decision analysis (MCDA) handles complex scenarios through ensemble methods combining multiple AI models.

Dynamic Threshold Management: The system implements **adaptive thresholds** based on risk assessment of deployment targets, historical performance data, business impact analysis, and time-sensitive versus non-critical operations. **Coverage boundaries** enforce strict limits on resource access permissions, data processing volumes, execution time windows, and network access scope.

Blast Radius Controls and Governance

Blast radius controls implement comprehensive risk mitigation through **incremental deployments** using canary releases, **bulkhead patterns** for system isolation, and **automated rollback triggers** based on performance metrics. The system evaluates downstream service dependencies, data sensitivity levels, user impact scope, and recovery time objectives (RTO).

Safety Mechanisms: **Circuit breaker patterns** prevent cascade failures, while **rate limiting** controls API calls and resource consumption. **Graceful degradation** activates when confidence drops below thresholds, with **human-in-the-loop mechanisms** for high-risk decisions.

Audit Trail and Version Control Integration

Comprehensive audit trails capture AI model decisions and confidence scores, input data and processing parameters, human approvals and overrides, and system state changes and configurations. **The technical stack** employs blockchain-based immutable audit trails, centralized logging with Splunk or ELK stack, real-time monitoring with alerting systems, and automated compliance reporting.

Version Control Integration: **Model versioning** integrates with MLflow, DVC, or custom registries, while **code integration** maintains tight coupling with Git workflows for automated model deployment triggers, configuration management, and rollback capabilities to specific versions.

LLM-Generated Natural-Language Rationale

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 June 2025 doi:10.20944/preprints202506.1436.v1

The rationale system employs advanced explainability techniques including Chain-of-Thought (CoT) prompting for reasoning transparency, Retrieval-Augmented Generation (RAG) for context-aware explanations, and template-based explanation generation. The system automatically generates Architecture Decision Records (ADR), code review explanations with rationale, deployment decision documentation, and risk assessment justifications.

Explainability Implementation: SHAP (SHapley Additive exPlanations) values and LIME (Local Interpretable Model-agnostic Explanations) provide human-interpretable AI decision making, while attention mechanisms visualization and feature importance ranking enhance transparency.

Current State Connection to 2025 AI-Driven DevOps

Market Evolution and Adoption Patterns

The AI-driven DevOps market demonstrates exceptional growth trajectory, with 74% of organizations implementing DevOps [21] in some capacity and 51% adoption rate for code copilots representing the highest among AI tools. Enterprise adoption patterns show healthcare leading GenAI adoption with $500M enterprise spend, while financial services represents $350M in enterprise AI spend.

Performance Metrics: DevOps-mature organizations [13] achieve 208x increase in code deployments and 2,604x faster lead time for changes. Organizations with high AI maturity report 3X higher ROI, while 92.1% of companies report significant benefits from data and AI investments.

Integration with Modern Observability Tools

Modern observability integration demonstrates sophisticated capabilities through **Grafana Cloud AI Observability** with native LLM monitoring, token usage tracking, and cost management. **Datadog**, named leader in Forrester Wave AIOps Platforms Q2 2025, processes trillions of telemetry data points hourly with AI-driven root cause analysis.

Advanced Monitoring Capabilities: **Prometheus AIOps** [8] provides intelligent anomaly detection through machine learning integration, while **custom ML models** enable real-time pattern analysis and predictive maintenance capabilities. The integration supports **intelligent alerting** with ML-powered alert correlation reducing noise by up to 69%.

Current Market Solutions

Major platform implementations include **AWS CodeGuru** for ML-powered code review and performance profiling, **Azure AI** with cognitive services integration, and **Google Cloud Vertex AI** integration with monitoring and deployment services. **Specialized solutions** like **ModelOp** provide enterprise AI governance with automated compliance, while **Transcend Pathfinder** offers unified AI governance frameworks.

Empirical Validation Analysis

Root-Cause Analysis Precision Validation

The 98% precision claim reflects sophisticated validation methodologies employing **multi-vector correlation analysis** systems like BigPanda [14] that use 29 unique vector dimensions to identify high-confidence correlations between alerts and change data. **Causal AI approaches** use causal graphs and domain knowledge graphs to distinguish true root causes from symptoms.

Validation Techniques: **Time-series validation** tests models on historical data sequences where ground truth root causes are established through post-incident analysis. **Synthetic incident generation** creates controlled environments with known failure scenarios to test precision rates, while **multi-source data integration** validates across logs, metrics, traces, and events.

MTTR Reduction Methodology

The 55% MTTR reduction measurement [4] employs **DORA Metrics integration** where Mean Time to Recovery serves as one of four key DevOps Research and Assessment metrics. **Granular time**

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 June 2025 doi:10.20944/preprints202506.1436.v1

tracking breaks down measurements into detection time, diagnosis time, resolution time, and verification time.

Statistical Validation: **T-test analysis** compares mean MTTR values between control and treatment periods, while **Mann-Whitney U tests** provide non-parametric testing for non-normally distributed incident resolution times. **Time series analysis** accounts for temporal dependencies and trends in incident patterns with 95% confidence levels required for reported improvements.

Developer Trust Assessment

Developer trust studies [16,17] employ **validated trust questionnaires** with **Likert scale assessments** using 5-7 point scales measuring trust dimensions including reliability, competence, and benevolence. **Multi-dimensional trust models** measure cognitive trust, emotional trust, and behavioral trust separately.

Key Trust Factors: **Technical factors** include system accuracy, explainability, reliability, and performance consistency, while **socio-ethical factors** encompass transparency, fairness, privacy protection, and ethical AI practices. **Usage analytics** track frequency of AI tool usage, feature adoption rates, and retention metrics.

Performance Validation Frameworks

Open-source versus enterprise environment comparisons reveal significant differences in resource availability, data quality and scale, and security and compliance requirements. Enterprise validation includes additional security, privacy, and regulatory compliance testing with 99.9%+ availability requirements.

Validation Challenges: **Less than 33% of AI research is reproducible**, with only 5% of researchers sharing source code. Solutions include **MLOps implementation**, comprehensive experiment tracking, and standardized benchmarking frameworks with **95% confidence intervals** required for performance claims.

Conclusions and Future Directions

The technical analysis reveals that AI-driven self-healing DevOps pipelines represent a mature, rapidly evolving field with proven capabilities for transforming software delivery processes. The combination of sophisticated detection mechanisms, multi-agent orchestration, robust governance frameworks, and comprehensive validation methodologies enables organizations to achieve significant improvements in system reliability, development velocity, and operational efficiency.

Key technical achievements include the breakthrough performance of LLM-based log parsing systems, the emergence of standardized multi-agent communication protocols, and the implementation of sophisticated governance mechanisms with confidence-based decision making. The field demonstrates strong empirical validation with measurable improvements in precision, MTTR reduction, and developer acceptance.

Future evolution points toward fully autonomous DevOps pipelines with minimal human intervention, standardized inter-agent communication protocols, and deeper integration with existing DevOps toolchains. Organizations adopting these technologies report substantial competitive advantages in software delivery speed, system reliability, and operational efficiency, positioning AI-driven self-healing DevOps as a critical capability for modern software development organizations.

References

1. [1] Chen, Z., et al. (2024). LogParser-LLM: Advancing Efficient Log Parsing with Large Language Models. *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. arXiv:2408.13727.
2. [2] Wang, K., et al. (2024). Stronger, Faster, and Cheaper Log Parsing with LLMs. *arXiv preprint* arXiv:2406.06156.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 17 June 2025 doi:10.20944/preprints202506.1436.v1

3. [3] Liu, H., et al. (2025). Multi-Agent Collaboration Mechanisms: A Survey of LLMs. *arXiv preprint* arXiv:2501.06322.
4. [4] Xu, Y., et al. (2025). LogSage: LLM-Driven CI/CD Remediation. *arXiv preprint*.
5. [5] Ji, H., & Luo, Z. (2025). LLM-Based Log Parsing and Anomaly Detection.
6. [6] Amazon Web Services. (2024). Deploy Amazon SageMaker Pipelines Using AWS Controllers for Kubernetes. *AWS Machine Learning Blog*. Retrieved from https://aws.amazon.com/blogs/machine-learning/deploy-amazon-sagemaker-pipelines-using-aws-controllers-for-kubernetes/
7. [7] Amazon Web Services. (2024). Use Kubernetes Operators for New Inference Capabilities in Amazon SageMaker. *AWS Machine Learning Blog*. Retrieved from https://aws.amazon.com/blogs/machine-learning/use-kubernetes-operators-for-new-inference-capabilities-in-amazon-sagemaker-that-reduce-llm-deployment-costs-by-50-on-average/
8. [8] Grafana Labs. (2024). How to Use Prometheus to Efficiently Detect Anomalies at Scale. *Grafana Blog*. Retrieved from https://grafana.com/blog/2024/10/03/how-to-use-prometheus-to-efficiently-detect-anomalies-at-scale/
9. [9] IBM Research. (2025). AI Agents in 2025: Expectations vs. Reality. *IBM Think Insights*. Retrieved from https://www.ibm.com/think/insights/ai-agents-2025-expectations-vs-reality
10. [10] DevOps.com. (2024). Harmonizing AI-Driven DevOps: Building Secure, Self-Healing Pipelines With AWS Bedrock and SageMaker. Retrieved from https://devops.com/harmonizing-ai-driven-devops-building-secure-self-healing-pipelines-with-aws-bedrock-and-sagemaker/
11. [11] Google Cloud. (2024). Human-in-the-Loop Overview. *Document AI Documentation*. Retrieved from https://cloud.google.com/document-ai/docs/hitl
12. [12] Zendesk. (2024). About Confidence Thresholds for Advanced AI Agents. *Zendesk Help Center*. Retrieved from https://support.zendesk.com/hc/en-us/articles/8357749625498-About-confidence-thresholds-for-advanced-AI-agents
13. [13] Atlassian. (2024). 4 Key DevOps Metrics to Know. *Atlassian DevOps Guide*. Retrieved from https://www.atlassian.com/devops/frameworks/devops-metrics
14. [14] BigPanda. (2024). AI-powered Root Cause Analysis. Retrieved from https://www.bigpanda.io/our-product/root-cause-analysis/
15. [15] The CTO Club. (2025). 20 Best AIOps Platforms of 2025. Retrieved from https://thectoclub.com/tools/best-aiops-platforms/
16. [16] Nature Communications. (2024). Trust in AI: Progress, Challenges, and Future Directions. *Humanities and Social Sciences Communications*. Retrieved from https://www.nature.com/articles/s41599-024-04044-8
17. [17] Taylor & Francis. (2022). A Systematic Literature Review of User Trust in AI-Enabled Systems: An HCI Perspective. *International Journal of Human-Computer Interaction*. DOI: 10.1080/10447318.2022.2138826
18. [18] ReadyTensor. (2025). AutoDevOps: Multi-Agent LLM Platform.
19. [19] Besiahgari, M. (2025). Case Study: AWS SageMaker and Bedrock for Self-Healing Pipelines.
20. [20] DevOps.com. (2025). The Future of DevOps: Key Trends, Innovations and Best Practices in 2025. Retrieved from https://devops.com/the-future-of-devops-key-trends-innovations-and-best-practices-in-2025/
21. [21] LambdaTest. (2025). Top 17 DevOps AI Tools [2025]. *LambdaTest Blog*. Retrieved from https://www.lambdatest.com/blog/devops-ai-tools/
22. [22] Simpliaxis. (2025). DevOps in 2025: Trends, Tools, and Impact on IT. *DevOps Practices Guide*. Retrieved from https://www.simpliaxis.com/resources/the-impact-of-devops-in-2025

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.