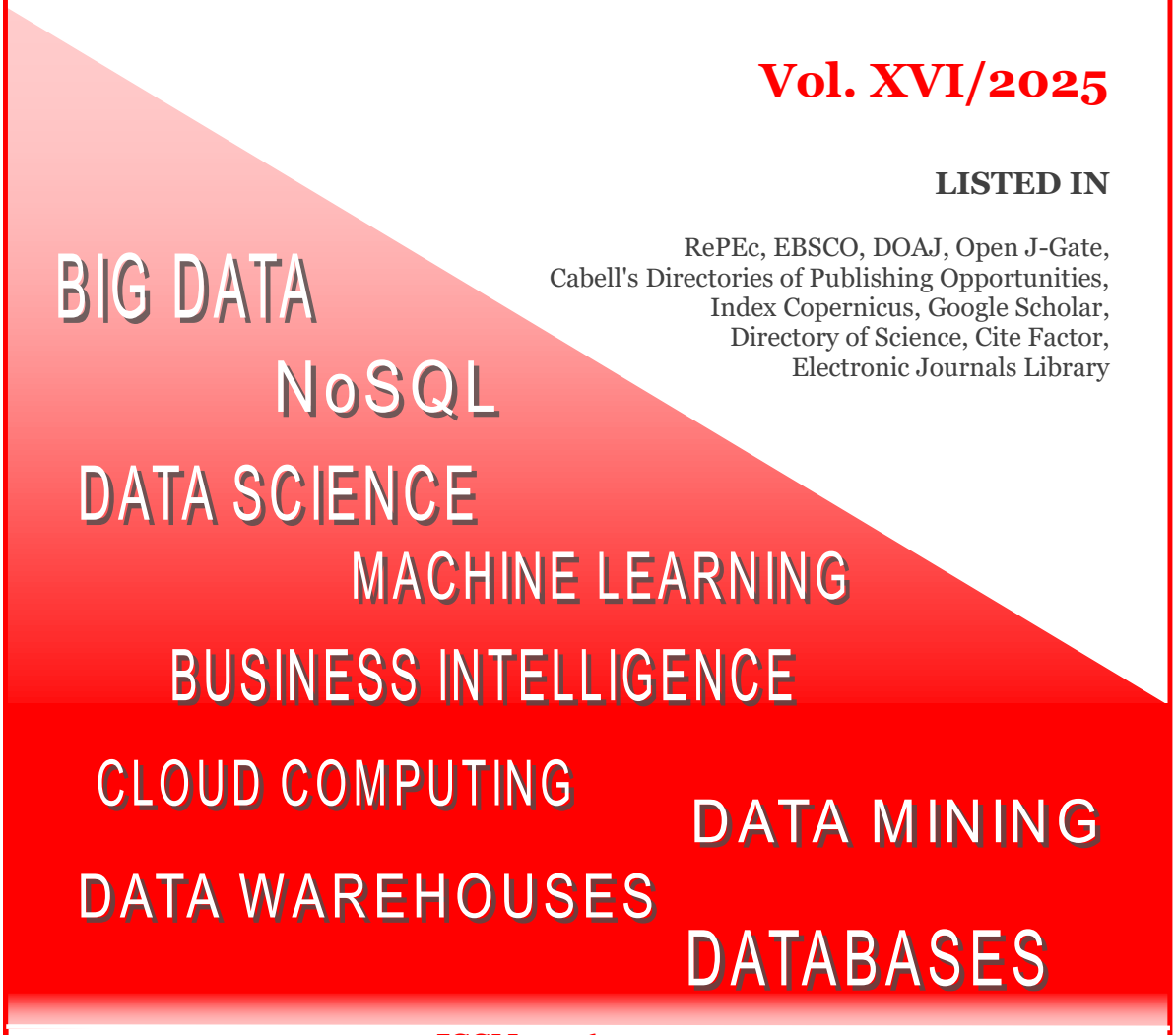
THE BUCHAREST UNIVERSITY OF ECONOMIC STUDIES

DATABASE SYSTEMS JOURNAL

Vol. XVI/2025

LISTED IN

RePEc, EBSCO, DOAJ, Open J-Gate,
Cabell's Directories of Publishing Opportunities,
Index Copernicus, Google Scholar,
Directory of Science, Cite Factor,
Electronic Journals Library



ISSN: 2069 – 3230
dbjournal.ro

Database Systems Journal BOARD

Director
Prof. Ion Lungu, PhD, University of Economic Studies, Bucharest, Romania

Editors-in-Chief
Prof. Adela Bara, PhD, University of Economic Studies, Bucharest, Romania
Conf. Iuliana Botha, PhD, University of Economic Studies, Bucharest, Romania

Editors
Conf. Anca Andreescu, PhD, University of Economic Studies, Bucharest, Romania
Conf. Anda Belciu, PhD, University of Economic Studies, Bucharest, Romania
Prof. Ramona Bologa, PhD, University of Economic Studies, Bucharest, Romania
Conf. Vlad Diaconița, PhD, University of Economic Studies, Bucharest, Romania
Lect. Alexandra Florea, PhD, University of Economic Studies, Bucharest, Romania
Prof. Adina Uța, PhD, University of Economic Studies, Bucharest, Romania

Editorial Board
Prof. Ioan Andone, PhD, A.I.Cuza University, Iasi, Romania
Prof. Emil Burtescu, PhD, University of Pitesti, Pitesti, Romania
Joshua Cooper, PhD, Hildebrand Technology Ltd., UK
Prof. Marian Dardala, PhD, University of Economic Studies, Bucharest, Romania
Prof. Dorel Dusmanescu, PhD, Petrol and Gas University, Ploiesti, Romania
Prof. Marin Fotache, PhD, A.I.Cuza University, Iasi, Romania
Dan Garlasu, PhD, Oracle Romania
Prof. Marius Guran, PhD, University Politehnica of Bucharest, Bucharest, Romania
Lect. Ticiano Costa Jordão, PhD-C, University of Pardubice, Pardubice, Czech Republic
Prof. Brijender Kahanwal, PhD, Galaxy Global Imperial Technical Campus, Ambala, India
Prof. Dimitri Konstantas, PhD, University of Geneva, Geneva, Switzerland
Prof. Hitesh Kumar Sharma, PhD, University of Petroleum and Energy Studies, India
Prof. Marinela Mircea, PhD, University of Economic Studies, Bucharest, Romania
Prof. Mihaela I.Muntean, PhD, West University, Timisoara, Romania
Prof. Stefan Nitchi, PhD, Babes-Bolyai University, Cluj-Napoca, Romania
Prof. Corina Paraschiv, PhD, University of Paris Descartes, Paris, France
Davian Popescu, PhD, Milan, Italy
Prof. Gheorghe Sabau, PhD, University of Economic Studies, Bucharest, Romania
Prof. Nazaraf Shah, PhD, Coventry University, Coventry, UK
Prof. Ion Smeureanu, PhD, University of Economic Studies, Bucharest, Romania
Prof. Traian Surcel, PhD, University of Economic Studies, Bucharest, Romania
Prof. Ilie Tamas, PhD, University of Economic Studies, Bucharest, Romania
Silviu Teodoru, PhD, Oracle Romania
Prof. Dumitru Todoroi, PhD, Academy of Economic Studies, Chisinau, Republic of Moldova
Prof. Manole Velicanu, PhD, University of Economic Studies, Bucharest, Romania
Prof. Robert Wrembel, PhD, University of Technology, Poznan, Poland

Contact
CaleaDorobanţilor, no. 15-17, room 2017, Bucharest, Romania
Web: https://dbjournal.ro/
E-mail: editordbjournal@gmail.com

CONTENTS

The Use of Communication Platforms in Military Operations: Enhancing Strategic and Tactical Effectiveness 1
Rexhep MUSTAFOVSKI

Query Completion for Small-Scale Distributed Databases in PostgreSQL and MongoDB 11
Marin FOTACHE, Cătălina BADEA, Marius-Iulian CLUCI, Codrin-Ștefan EȘANU

An Overview of Big Data and NoSQL in the Video Game Industry 28
Cristiana COSTAN

Evaluating Deep Learning and Machine Learning Models in Federated Learning for Credit Card Fraud Detection. A comparative study 37
Șener ALI

The Advantage of NoSQL Databases over SQL Databases 45
Mario-Tudor CHIRIAC

House Market Prediction Using Machine Learning 55
Nicușor-Andrei ANDREI

Detection of Fake News Using Deep Learning and Machine Learning 65
Gabriela CHIRIAC, Ada Maria CATINA

The Use of Communication Platforms in Military Operations: Enhancing Strategic and Tactical Effectiveness

Rexhep MUSTAFOVSKI
Ss. Cyril and Methodius University
Faculty of Electrical Engineering and Information Technologies
Skopje, Republic of North Macedonia
rexhepmustafovski@gmail.com

Abstract: *The way communication platforms are used in military operations has changed a lot over the years. They're now essential for mission success, quick decision-making, and maintaining strategic advantages. This paper dives into the modern communication tools that are making waves in military settings, with a spotlight on networked communication, signal support, cybersecurity strategies, and how different forces work together in joint and multinational missions. It also tackles some of the major hurdles, like congestion in the electromagnetic spectrum, cyber threats, and the need for secure data transmission in challenging environments. By pulling insights from FM 6-02: Signal Support to Operations and the latest scientific research, this paper highlights recent advancements in military communication tech and how they boost operational effectiveness. Additionally, the research looks ahead at future possibilities, such as AI-driven communication platforms, quantum encryption, and cutting-edge satellite networks for defense purposes. This paper's primary contribution is the development of a structured, AI-enabled communication workflow that integrates quantum-safe encryption, blockchain authentication, and satellite-based coordination to improve decision-making and resilience in multi-domain operations.*
Keywords: *Military communication, secure networks, battlefield connectivity, radio systems, cybersecurity, signal support, tactical communications, AI in defense*

1 Introduction

The effectiveness of military operations depends heavily on reliable, secure, and adaptable communication platforms. As modern warfare becomes increasingly network-centric, the ability to transmit real-time information across multiple operational domains—land, air, sea, space, and cyberspace—has become a strategic necessity. Military communication systems have evolved to integrate advanced radio networks, satellite-based communication, and encrypted data transmission, ensuring seamless coordination between deployed forces and command centers [1], [3], [5]. The increasing complexity of multinational operations, joint task forces, and asymmetric warfare has further emphasized the need for highly secure and interoperable communication infrastructures that can operate even in contested environments [6]–[8].

A key aspect of modern military communication is signal support, which provides robust network connectivity and ensures continuous information flow despite cyber threats, electronic warfare (EW) attacks, and environmental disruptions. Military doctrines emphasize the importance of agility, redundancy, and survivability in communication networks, ensuring that forces remain operational even when primary channels are compromised [9], [11]. The U.S. Army's Field Manual 6-02 (FM 6-02) outlines fundamental principles of signal support, including interoperability, network resilience, and cybersecurity strategies essential for mission success [17]. These principles are crucial in large-scale combat operations, where the ability to maintain command and control (C2) can determine operational success or failure [12], [14].

The transition from analog to digital military communication platforms has introduced a range of advantages, such as higher data

transfer rates, AI-assisted network management, and automated encryption protocols [15]–[17]. The introduction of software-defined radios (SDRs), wideband satellite networks, and AI-driven cyber defense mechanisms has further enhanced battlefield communication, allowing for real-time situational awareness and faster decision-making [4], [10]. However, despite these advancements, challenges remain, particularly regarding electromagnetic spectrum congestion, jamming threats, and cyberattacks from adversarial forces [7], [9].

Cybersecurity remains one of the most critical concerns in military communication platforms. Cyber warfare, espionage, and digital sabotage pose a significant risk to military networks, requiring advanced encryption techniques, quantum-safe cryptographic methods, and AI-based intrusion detection systems to mitigate vulnerabilities [6], [8]. The Department of Defense Information Network (DODIN) plays a crucial role in securing data transmission, integrating multi-layer encryption protocols, blockchain-based authentication, and adaptive firewall mechanisms [13], [16]. The role of artificial intelligence and machine learning in automating cybersecurity responses and threat detection is expected to further revolutionize military communication strategies [14]–[17].

This paper explores the current state of military communication platforms, their integration with modern defense technologies, and the challenges posed by electronic warfare and cybersecurity threats. The study also discusses future trends in military communication, including quantum encryption, AI-assisted battlefield networking, and low-Earth orbit (LEO) satellite communication systems [2], [5], [12]. By analyzing these technological advancements and their impact on military operations, this research aims to provide valuable insights into the future of secure and resilient communication platforms in defense applications [1], [4], [8].

The main scientific contribution of this study lies in its proposal of an adaptive workflow for next-generation military communication platforms. By integrating AI-driven automation, blockchain security, and low-latency satellite connectivity, the paper introduces a novel architecture that significantly enhances tactical coordination, cyber resilience, and real-time situational awareness in hostile environments.

2 Literature Review

The study of military communication platforms has evolved significantly in recent years, with various technological advancements, operational challenges, and security concerns shaping the field. Several scholarly works and military doctrines have explored the integration of digital communication networks, cybersecurity measures, and tactical communication strategies in modern warfare [1]–[3]. The transition from analog to digital systems, the emergence of AI-driven networking, and the adoption of secure satellite communications have played a crucial role in improving the efficiency of military communication platforms [4], [5].

One of the key components of military communication is interoperability, which ensures seamless information exchange between different branches of the armed forces and multinational coalitions [6], [7]. The U.S. Army's Field Manual FM 6-02 emphasizes the importance of joint operations, where different communication networks must operate efficiently under a unified system [17]. Research highlights the role of secure radio networks, software-defined radios (SDRs), and advanced satellite communication systems in achieving this goal [8], [9]. Table 1 provides an overview of traditional vs. modern military communication platforms, showing their differences in terms of data transmission, security, and adaptability.

Table 1. Comparison of Traditional and Modern Military Communication Platforms

Feature	Traditional Communication	Modern Digital Communication
Transmission Medium	Analog radio signals	Encrypted digital networks
Data Rate	Low	High-speed broadband
Security	Vulnerable to interception	End-to-end encryption & AI-based security
Interoperability	Limited	Fully integrated with joint/multinational operations
Adaptability	Fixed-frequency systems	Software-defined, frequency-agile systems
Reliability in Combat	Prone to jamming	Resistant to EW & cyber threats

While prior studies have discussed individual elements such as AI, SDRs, or satellite systems, this paper uniquely proposes an integrated framework combining these technologies with blockchain and quantum encryption. The proposed workflow advances beyond existing architecture by offering real-time threat adaptation, predictive decision support, and unified cross-domain operations, which are not concurrently addressed in existing literature.

2.1 Advancements in Military Communication Platforms

The integration of AI and machine learning (ML) in military communication networks has allowed automated threat detection, real-time data processing, and network self-healing capabilities [10], [11]. AI-based systems can predict cyber threats, manage bandwidth distribution, and optimize data transmission in congested environments, significantly enhancing operational efficiency [12], [13].

Satellite communication (SATCOM) plays a pivotal role in long-range military operations, providing beyond-line-of-sight (BLOS) connectivity, global reconnaissance, and secure battlefield networking [14], [15]. Low-Earth Orbit (LEO) satellites have gained popularity due to their low latency, high-speed data transmission, and enhanced resilience against jamming [16], [17]. These systems complement traditional geostationary satellites (GEO), ensuring continuous coverage in remote and contested environments. Table 2 highlights the differences between various satellite communication technologies used in military operations.

Table 2. Comparison of Military Satellite Communication Technologies

Feature	Geostationary Satellites (GEO)	Medium Earth Orbit (MEO)	Low Earth Orbit (LEO)
Altitude	~35,786 km	~5,000–20,000 km	~500–1,500 km
Latency	High (~500ms)	Moderate (~200ms)	Low (~50ms)
Coverage	Global but with delay	Regional	High-speed global coverage
Jamming Resistance	Moderate	High	Very High
Data Transfer Speed	Limited	Moderate	High-speed broadband
Deployment Cost	High	Moderate	Lower than GEO/MEO

2.2 Cybersecurity Challenges in Military Communication

As digital transformation continues in military operations, cybersecurity remains a critical challenge. Hostile actors, state-sponsored cyber units, and non-state entities are continuously attempting to exploit vulnerabilities in military networks [5], [6]. Recent cyber-attacks on defense infrastructures highlight the necessity for multi-layered encryption, real-time intrusion detection, and AI-driven anomaly detection systems [7]–[9].

Cyber threats can disrupt battlefield communications, intercept classified intelligence, or manipulate strategic data, leading to severe consequences on national security. The FM 6-02 doctrine outlines cyber protection measures, including encryption protocols, threat monitoring, and secure data transmission techniques to mitigate these risks [17].

To address these cybersecurity concerns, military forces are investing in post-quantum encryption, blockchain authentication, and AI-driven cybersecurity tools [10], [11]. Automated network defense mechanisms, combined with zero-trust security models, are expected to play a crucial role in securing military communication networks [12], [13].

2.3 Future Trends in Military Communication Platforms

The future of military communication is being shaped by AI-driven automation, quantum-safe encryption, and next-generation tactical communication platforms. Some of the emerging trends include:

- AI-Powered Autonomous Networks: AI-driven systems will automate network management, optimize signal transmission, and predict potential threats [14].
- Quantum Communication: The adoption of quantum key distribution (QKD) and quantum-safe encryption will enhance data security in military operations [15], [16].
- Integrated IoT-based Defense Systems: The Internet of Battlefield Things (IoBT) will enable real-time sensory integration, predictive analytics, and automated reconnaissance [17].
- Augmented Reality (AR) Communication: The use of AR and holographic battlefield interfaces will improve situational awareness and mission coordination [9], [10].

3 A New Workflow for the Next Generation of Military Communication Systems

The proposed workflow consists of six key stages, ensuring secure, adaptive, and mission-driven communication that enhances interoperability among military branches, joint task forces, and multinational coalitions.

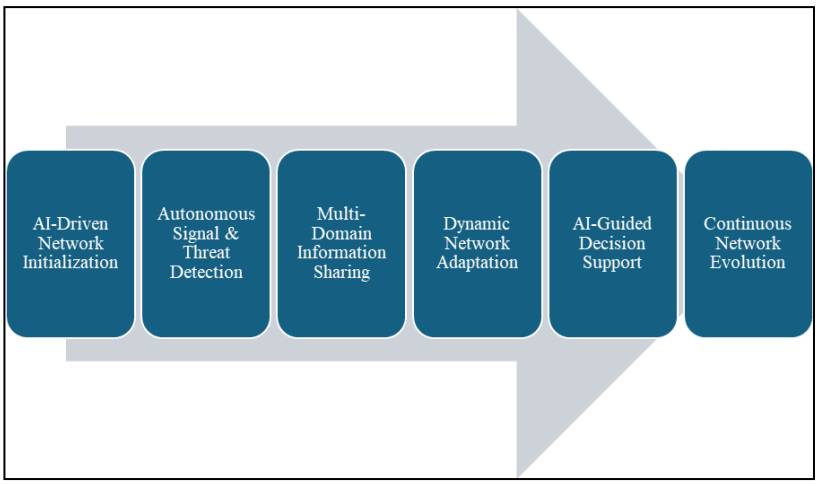


Fig. 1. Workflow for the Next Generation of Military Communication Systems

- **Workflow Steps and Explanation:**

1. **AI-Driven Network Initialization & Configuration**
   - The network automatically establishes a secure and encrypted connection across all deployed units.
   - AI-based network optimization identifies the best communication pathways to minimize latency and bandwidth congestion.
   - Quantum encryption secures data transmission from the start to prevent cyberattacks.

2. **Autonomous Tactical Signal Management & Threat Detection**
   - AI-powered systems continuously monitor radio frequency (RF) environments for jamming attempts or cyber threats.
   - Smart spectrum management automatically adjusts frequency usage, ensuring optimal connectivity in contested environments.
   - Blockchain-based authentication ensures that only authorized personnel can access sensitive communications.

3. **Real-Time Multi-Domain Information Sharing**
   - Edge computing and IoT sensors collect, analyze, and distribute battlefield intelligence instantly.
   - AI-based speech recognition converts voice commands into encrypted digital messages for secure, instant dissemination.
   - Unified Command Network (UCN) ensures seamless communication between land, air, sea, space, and cyber forces.

4. **Dynamic Network Adaptation in Combat Environments**
   - Low-Earth Orbit (LEO) satellite networks provide real-time battlefield coverage, reducing reliance on vulnerable ground infrastructure.
   - SDR-based radio networks dynamically switch frequencies to avoid electronic warfare (EW) interference.
   - Cloud-based situational awareness platforms allow commanders to access real-time intelligence anywhere.

5. **AI-Guided Decision Support & Predictive Analytics**
   - AI-powered battlefield assistants analyze incoming data and recommend the best tactical responses based on real-time intelligence.
   - Predictive cybersecurity models anticipate cyber threats and apply preemptive countermeasures.
   - Automated logistical coordination ensures supply chains remain intact and uninterrupted.

6. **Continuous Network Evolution & Learning**
   - Self-healing networks use machine learning (ML) to identify weak points and automatically reinforce connectivity.
   - AI-based after-action reviews (AARs) analyze communication effectiveness and optimize future missions.
   - Secure post-mission data storage ensures historical operational records remain classified and protected.

Table 3. Workflow for Next-Generation Military Communication

Step	Process	Technology Used	Impact on Military Operations
1. AI-Driven Network Initialization	Automatic secure connections & encrypted channels	AI-driven routing, quantum encryption	Faster & safer mission readiness
2. Autonomous Signal & Threat Detection	Real-time RF scanning & adaptive spectrum use	AI-assisted EW defense, blockchain security	Enhanced jamming resistance & secure access control
3. Multi-Domain Information Sharing	Seamless cross-platform communication	IoT-based intelligence, AI speech recognition	Faster battlefield awareness
4. Dynamic Network Adaptation	Resilient satellite & SDR-based networks	LEO satellites, cloud-based ops centers	Reliable comms in contested environments
5. AI-Guided Decision Support	AI-driven battlefield analysis & predictive logistics	AI-powered analytics threat modeling	Improved decision-making speed
6. Continuous Network Evolution	Self-healing networks & adaptive AI	ML-based security, after-action data analysis	Long-term resilience & future optimization

The proposed next-generation military communication workflow will revolutionize military operations by providing:

1. **Uninterrupted Communication in High-Risk Environments**
   - AI-powered adaptive frequency shifting and SDR ensure continuous connectivity even under electronic warfare conditions.
   - Decentralized blockchain authentication prevents data spoofing or interference.
   - LEO satellites ensure resilient BLOS (Beyond-Line-of-Sight) communication in remote areas.

2. **Improved Decision-Making & Tactical Coordination**
   - AI-driven command assistants analyze real-time battlefield data and offer rapid, informed decision recommendations.
   - Enhanced situational awareness allows commanders to adapt quickly based on automated intelligence reports.
   - Interoperable networks enable seamless collaboration among land, air, sea, and cyber units.

3. **Faster Deployment & Readiness**
   - AI-driven network automation reduces deployment times by instantly configuring communication systems.
   - Autonomous security systems ensure data protection from cyberattacks before missions begin.
   - Predictive analytics assist in early detection of potential threats, ensuring forces are prepared.

4. **Advanced Cybersecurity Protection Against Digital Threats**
   - AI-driven cybersecurity models predict malware, phishing, and cyber-intrusion attempts before they happen.
   - Quantum-safe encryption ensures that classified military data remains unbreakable.
   - Dynamic firewall systems prevent adversaries from exploiting communication vulnerabilities.

5. **Greater Interoperability Between Allied Forces**
   - The Unified Command Network (UCN) ensures that all allied forces, regardless of branch or nation, can

communicate without compatibility issues.
- Standardized AI-assisted voice translation systems allow seamless communication between different military forces in multinational operations.
- The integration of cloud-based command centers enables real-time data access from global military installations.

Table 4. Impact of Next-Generation Military Communication on Operational Success

Key Advantage	Traditional Communication	Next-Generation Communication	Impact on Operations
Speed of Deployment	Manual setup & configuration	AI-automated network setup	Reduced setup time & faster mission readiness
Reliability in Combat	Prone to jamming	Adaptive SDRs & LEO satellites	Continuous connection in high-risk areas
Cybersecurity	Encryption only	AI-driven security & quantum-safe encryption	Stronger data protection & cyber resilience
Interoperability	Limited to branch-specific networks	Unified Command Network (UCN)	Seamless allied coordination
Threat Response	Manual security checks	AI-assisted real-time anomaly detection	Preemptive cyber defense measures

4 Challenges in Implementing the Proposed Next-Generation Military Communication Platform

While the proposed AI-driven, quantum-encrypted, and multi-domain military communication platform offers significant advantages in security, interoperability, and efficiency, its implementation presents several challenges. Military environments demand high resilience, rapid adaptability, and uncompromising security, making the integration of advanced technologies a complex and resource-intensive endeavour. Below are the key challenges that must be addressed for successful deployment.

a. Cybersecurity Threats and Quantum-Resistant Encryption

One of the most significant challenges in implementing this next-generation communication platform is ensuring robust cybersecurity. Military networks are prime targets for cyber warfare, espionage, and digital sabotage by state-sponsored actors and sophisticated adversaries. The adoption of quantum encryption enhances security, but the transition from traditional encryption methods poses integration challenges, high computational costs, and compatibility issues with legacy systems. Additionally, AI-powered cybersecurity tools must be rigorously tested to avoid vulnerabilities that adversaries could exploit.

b. Electromagnetic Spectrum Management and Jamming Resistance

Modern warfare relies heavily on the electromagnetic spectrum for secure radio, satellite, and networked communication. However, spectrum congestion, frequency interference, and electronic warfare (EW) attacks pose severe threats. Adaptive frequency hopping, AI-driven spectrum allocation, and SDR (Software-Defined Radios) help mitigate these risks, but their deployment requires extensive testing in real-world combat scenarios. Additionally, adversaries constantly develop advanced jamming and spoofing techniques, making continuous adaptation and countermeasure development essential.

c. Interoperability with Legacy and Allied Systems

Military forces operate with a mix of old and new communication technologies, making interoperability a major concern. Many allied forces and joint coalitions use different encryption standards, radio frequencies, and data-sharing protocols, requiring a universal framework to ensure seamless communication. The proposed Unified Command Network (UCN) can enhance cross-force connectivity, but integrating diverse national defense systems into a single secure framework is a technically and politically complex challenge.

d. AI and Automation Risks in Battlefield Decision-Making

The integration of AI-driven decision support systems and automated battlefield analytics enhances efficiency but also introduces risks related to machine bias, decision latency, and potential adversarial manipulation. AI must interpret real-time battlefield data with high accuracy, but errors in object classification, speech recognition, or enemy movement prediction can lead to tactical failures. Additionally, AI's reliance on massive data sets raises concerns about data security, ethical considerations, and the risk of enemy AI-based countermeasures.

e. Infrastructure Deployment in Remote and Hostile Environments

Deploying a resilient, battlefield-ready communication infrastructure requires secure, mobile, and easily maintainable networking equipment. Remote operations—such as desert, jungle, or arctic warfare—face logistical challenges in establishing LEO satellite links, maintaining power sources, and protecting hardware from environmental stressors. Military-grade energy-efficient, solar-powered, and autonomous relay nodes can help mitigate these issues, but ensuring long-term network uptime remains a challenge.

f. High Costs and Budget Constraints

The development, testing, and large-scale implementation of next-generation military communication networks require substantial investment. Quantum-safe encryption, AI-driven cybersecurity, and satellite-based networking involve expensive research, hardware procurement, and system training costs. Many military organizations face budget constraints, long procurement cycles, and policy debates that could delay adoption and full-scale deployment.

g. Resistance to Change and Human Adaptation

The shift to AI-enhanced, real-time, and automated communication platforms may face resistance from military personnel accustomed to traditional systems. Soldiers and commanders require extensive training to operate new encrypted networks, AI-driven interfaces, and cloud-based data-sharing platforms. Without proper user adaptation, trust, and operational readiness, even the most advanced system could fail under real-world combat pressure.

5 Conclusion

The rapid advancement of military communication platforms has redefined modern warfare, enabling faster decision-making, improved battlefield coordination, and enhanced cybersecurity. As conflicts become more technologically driven, military forces must adopt AI-driven automation, quantum-safe encryption, and real-time satellite-based connectivity to maintain operational superiority. The transition from traditional analog and single-frequency radio systems to adaptive, software-defined networks and AI-enhanced communication infrastructures ensures that military units remain connected even in contested, high-risk environments. Interoperability remains a cornerstone of future military communication, enabling seamless integration among different

branches, joint forces, and multinational coalitions. The Unified Command Network (UCN), enhanced by AI-assisted real-time speech translation and blockchain authentication, will allow allied forces to share intelligence, coordinate logistics, and execute joint operations without compatibility issues. Additionally, the deployment of LEO satellite networks will significantly reduce reliance on vulnerable ground-based infrastructure, ensuring uninterrupted connectivity and beyond-line-of-sight (BLOS) communication in remote battlefields. Cybersecurity is another crucial factor in military communication, as digital threats continue to evolve. AI-powered threat detection, self-healing networks, and quantum-resistant encryption will provide unparalleled protection against cyber espionage, electronic warfare, and digital sabotage. The integration of predictive analytics and automated cyber-defense mechanisms ensures preemptive threat mitigation before hostile actors can exploit vulnerabilities.

The next generation of military communication will be more adaptive, intelligent, and secure, offering greater operational flexibility and resilience. By integrating AI, autonomous network management, and multi-domain interoperability, military forces can achieve superior situational awareness, faster response times, and enhanced combat effectiveness. These advancements will not only strengthen battlefield capabilities but also ensure long-term strategic dominance in modern and future conflicts. Military success will increasingly depend on who can communicate faster, more securely, and more efficiently in the evolving digital battlespace.

References

[1] Abdelzaher, Tarek / Wigness, Mike / Russell, Sean / Swami, Ananthram: *Internet of Battlefield Things: Challenges, Opportunities, and Emerging Directions.* IoT for Defense and National Security, 2023, pp. 5–22.

[2] Akbar, Ridho S / Kholid, Fadhil / Kasiyanto, Kharis / Widiatmoko, Donny / Achmad, Agus: *Design of Fuel Monitoring Application for Reservoir Tanks in Army Fuel Supply Point on Military Logistics Corps Based on Internet of Things.* International Journal of Engineering and Computer Science Applications (IJECSA), 2024, pp. 19–32.

[3] Alkanjr, Bilal / Mahgoub, Ibrahim: *A Novel Deception-Based Scheme to Secure the Location Information for IoBT Entities.* IEEE Access, 2023, pp. 15–554.

[4] Azar, Jad / Makhoul, Amal / Barhamgi, Mounira / Couturier, Raphael: *An Energy Efficient IoT Data Compression Approach for Edge Machine Learning.* Future Generation Computer Systems, 2023, pp. 168–175.

[5] Butun, Ismail / Mahgoub, Ibrahim: *Expandable Mix-Zones as a Deception Technique for Providing Location Privacy on Internet-of-Battlefield Things (IoBT) Deployments.* IEEE Access, 2024.

[6] Doku, Reginald / Rawat, Danda B / Garuba, Muritala / Njilla, Laurent: *Fusion of Named Data Networking and Blockchain for Resilient Internet-of-Battlefield-Things.* IEEE 17th Annual Consumer Communications & Networking Conference (CCNC), 2023, pp. 1–6.

[7] Farooq, M. Junaid / Zhu, Quanyan: *Secure and Reconfigurable Multi-Layer Network Design for Critical Information Dissemination in the Internet of Battlefield Things (IoBT).* IEEE Transactions on Wireless Communications, 2023, pp. 2618–2632.

[8] Feng, Yujie / Li, Ming / Zeng, Chen / Liu, Hu: *Robustness of Internet of Battlefield Things (IoBT): A Directed Network Perspective.* Entropy, 2023, pp. 1166.

[9] Heidari, Alireza / Jamali, Jabraeil: *Internet of Things Intrusion Detection Systems: A Comprehensive Review and Future Directions.* Cluster Computing, 2023, pp. 3753–3780.

[10] Joshi, Suraj / Thakar, Anshuman / Patel, Chirag: *Applications of Machine Learning and Deep Learning in Securing Internet of Battlefield Things: A Futuristic Perspective.* 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom), IEEE, 2023, pp. 333–338.

[11] Karim, Hossain / Rawat, Danda B: *Evaluating Machine Learning Classifiers for Data Sharing in Internet of Battlefield Things.* IEEE Symposium Series on Computational Intelligence (SSCI), 2023, pp. 1–7.

[12] Kufakunesu, Rachel / Myburgh, Herman / De Freitas, Allan: *The Internet of Battle Things: A Survey on Communication Challenges and Recent Solutions.* Discover Internet of Things, 2024, pp. 3–19.

[13] Kunatsa, Tendai / Myburgh, Herman C / De Freitas, Allan: *Optimal Power Flow Management for a Solar PV-Powered Soldier-Level Pico-Grid.* Energies, 2024, pp. 459.

[14] Liu, Dandan / Abdelzaher, Tarek / Wang, Tianyu / Hu, Yibo / Li, Jian / Liu, Shiyang / Caesar, Matthew / Kalasapura, Deepak / Bhattacharyya, Jay / Srour, Nabil: *IoBT-OS: Optimizing the Sensing-to-Decision Loop for the Internet of Battlefield Things.* IEEE International Conference on Computer Communications and Networks (ICCCN), 2023, pp. 1–10.

[15] Masuduzzaman, Mohammad / Rahim, Tariqul / Islam, Asif / Shin, Sun Yoon: *UAV-Employed Intelligent Approach to Identify Injured Soldier on Blockchain-Integrated Internet of Battlefield Things.* IEEE Transactions on Network and Service Management, 2024.

[16] Rutravigneshwaran, Prabu / Anitha, Gunasekaran / Prathapchandran, Krishnakumar: *Trust-Based Support Vector Regressive (TSVR) Security Mechanism to Identify Malicious Nodes in the Internet of Battlefield Things (IoBT).* International Journal of System Assurance Engineering and Management, 2024, pp. 287–299.

[17] Singh, Rakesh Kumar / Mishra, Satyendra: *TinyML Meets IoBT Against Sensor Hacking.* The Network and Distributed System Security (NDSS) Symposium, Workshop on Security and Privacy in Standardized IoT (SDIoTSec), 2024, pp. 1–9.

Rexhep Mustafovski, MSc, is a Teaching and Research Assistant at the Military Academy "General Mihailo Apostolski" - Skopje, Department for Cybersecurity and Digital Forensics. He holds a Master's degree from the Faculty of Electrical Engineering and Information Technologies, University "Ss. Cyril and Methodius" - Skopje. His research interests include radar systems, IoT security, object detection technologies, and integrated control and monitoring systems. He has authored and co-authored more than 20 scientific and professional papers published in international conferences and journals, including several indexed in high-impact factor journals. His work contributes to advancements in next-generation radar systems, cybersecurity, and military applications of digital technologies.

Query Completion for Small-Scale Distributed Databases in PostgreSQL and MongoDB

Marin FOTACHE[1], Cătălina BADEA[2], Marius-Iulian CLUCI[3], Codrin-Ștefan EȘANU[4]
[1]Dept.of Accounting, Information Systems and Statistics
Alexandru Ioan Cuza University of Iaşi, Faculty of Economics and Business Administration
[2]Alexandru Ioan Cuza University of Iaşi
[3] Alexandru Ioan Cuza University of Iaşi, S.C. HVM SRL
[4] Alexandru Ioan Cuza University of Iaşi, Cegeka Romania
Iaşi, Romania
fotache@uaic.ro, catalina.badea@student.uaic.ro, marius.cluci@uaic.ro, codrin.stefan@feaa.uaic.ro

Relational/SQL and document/JSON data stores are competing but also complementary technologies in the OLAP (On-Line Analytical Processing) systems. Whereas the traditional approaches for performance comparison use the duration of queries performing similar tasks, in this paper we compare the performance of two distributed setups deployed on PostgreSQL/Citus and MongoDB by focusing only on the query's successful completion within a 10-minute timeout. The TPC-H benchmark database was converted into a denormalized JSON schema in MongoDB. An initial set of 296 SQL queries was devised for execution in PostgreSQL/Citus and then mapped for execution in MongoDB using Aggregation Framework (AF). Query execution success within a 10-minute timeout was collected for both PostgreSQL and MongoDB in six scenarios defined by two small-scale data factors (0.01 and 0.1 GB) and three different node counts (3, 6, and 9) for data distribution and processing. The relationships between the query completion and the query parameters were assessed with statistical tests and a series of machine learning techniques.

Keywords: *PostgreSQL, Citus, MongoDB, SQL, Aggregation Framework, OLAP performance comparison.*

1 Introduction

Current data architectures incorporate various technologies for data storage and processing, such as relational/SQL and NoSQL data servers [1][2][3]. In this paper, we analyse and compare the OLAP [4] performance of PostgreSQL and MongoDB on six distributed configurations deployed on OpenStack [5], by varying the database size and the number of data distribution nodes.

For this purpose, we used the TPC-H Benchmark [6] database and tools, to populate the TPC-H database tables with a specified volume of data. We started with the original TPC-H schema, devising a module to populate a corresponding MongoDB schema by transforming (mainly by nesting) the data structure. This transformation allows a MongoDB collection to integrate data originally scattered across two or more tables.

We developed a set of 296 SQL queries (for the original TPC-H schema) that vary in terms of number of joins, filter clauses, or group clauses. Subsequently, this query set was translated into MongoDB's Aggregation Framework (MAF) and adapted to the new document database structure. In this new structure, some of the joins between collections become unnecessary, as the data is denormalized and stored as nested arrays inside documents.

The queries were run on both PostgreSQL and MongoDB setups across six scenarios. These scenarios were defined by the combination of two small-scale factors (of 0.01GB and 0.1GB) and three distribution architectures with 3, 6, and 9 nodes. A 10-

minute timeout was imposed for the execution of each query. Queries that did not complete within 600 seconds were canceled.

The preliminary results on the relationship between query completion (for both database servers) and the database size and the distribution setup were presented in [6]. In this paper, we examine the relationships between the query completion and some parameters describing the query complexity; we also developed, tuned, and interpreted Machine Learning (ML) classification models for predicting query completion on various predictors about database size, distribution setup and query complexity.

The remaining of this paper is organized as follows: section 2 examines the results of some previous studies approaching the performance comparison of SQL (PostgreSQL) and document (MongoDB) data stores. Section 3 provides a brief overview of the distributed architectures of Citus (a distributed PostgreSQL solution) and MongoDB. Section 4 describes the experimental setup, data analysis methods and tools used for obtaining the results. Results are analyzed in section 5. The paper concludes with the main findings, pointing out the study's main limitations, and some future directions for research.

2 Previous studies on OLAP performance comparison of SQL and NoSQL database servers

Both PostgreSQL and MongoDB are highly popular on the database market [7] and a considerable body of literature targeted their OLTP (On-Line Transaction Processing) and/or OLAP performance comparison. Results are far from convergent, as seen below.

In [8] PostgreSQL outperformed MongoDB. Güney and Ceylan [9] found that MongoDB had certain advantages in specific cases, particularly in data sorting operations, while PostgreSQL performed better in bulk data extraction and more complex query operations. Villalobos et al. [10] found that PostgreSQL performs more efficiently with complex queries involving intersection or combination functions, while MongoDB is better suited for simpler queries involving only filtering of descriptive, non-geographic data. In [11] MongoDB excelled in managing national election data, with consistent execution times regardless of operation size. PostgreSQL also performed well, with memory usage increasing linearly with the size of the operation. Also, in [12] MongoDB significantly outperformed SQL Server in processing e-government data.

Yedilkhan et al. [13] showed that both MongoDB and PostgreSQL are suitable for particular use cases and scenarios. The comparison was based on performance evaluations conducted across multiple clusters, including both cloud and on-premises environments. In [14] MongoDB outperformed PostgreSQL in terms of latency. Makris et al. [15] concluded that PostgreSQL performed better than MongoDB when managing spatio–temporal data. In [16] results showed that relational databases are more efficient than non-relational databases when executing the basic types of database operations (CRUD – Create, Read, Update, Delete).

Tracz and Plechawska-Wójcik [17] analyzed the performance of relational and non-relational databases using MS SQL Server, MongoDB, and CouchDB. A total of 30 test series were conducted for each database server across the scenarios of interest, using the following record counts: 500, 1000, 2000, 5000, and 10000; results revealed that SQL Server ranks first, followed by MongoDB in the second place, and CouchDB as the least efficient of the three. In [18] results indicated that NoSQL databases are a better alternative to their relational counterparts. Setyawan et al. [19] found that MongoDB is well-suited for handling large volumes of data, while MySQL recorded optimal performance for queries executed on a smaller scale.

Figueiredo et al. [20] showed that PostgreSQL manages better data that is well structured, due to its robust support for complex queries. In their study, a 10 GB dataset was processed by each database server, with three types of queries: 1st, 5th and 6th (from the TPC-H benchmark query-set) selected for performance evaluation. Two testing scenarios were employed, one involving approximately 750 MB of data inserted into PostgreSQL and MongoDB, and another with a 10 MB file inserted into HarperDB. On the response time, PostgreSQL recorded the best database performance.

Sals et al. (2023) [21] argued that MongoDB outperforms MySQL in data storage and processing capabilities. Moreover, Antas et al. [22] concluded that MongoDB has better results for almost all large data volume tests; similarly, in [23] MongoDB performed better for CRUD operations. Abukabar et al. [24] designed an evaluation benchmark and identified that MongoDB recorded superior performance in handling CRUD operations, while MySQL excels in executing stored procedures. Matallah et al. [25] found that MongoDB recorded smaller execution time than MySQL for all types of query operations. In [26] the results showed that MongoDB had significantly higher throughput than both MySQL and PostgreSQL, particularly for small transaction sizes, indicating that MongoDB is a better choice for applications which involve heavy writing of indexed spatial data.

Reichardt et al. [27] compared the performance of NoSQL data stores, relative to their relational/SQL counterparts. They found that MongoDB is more suitable than MySQL for a wide range of dataset sizes. Naufal et al. [28] found MongoDB outperforms in the Update and Delete operations based on runtime differences.

3 Distributed architectures for PostgreSQL/Citus and MongoDB. Deployment on OpenStack

Citus is an open-source and easy-to-deploy PostgreSQL extension which enhances PostgreSQL by distributing the database using horizontal scaling and parallel query processing [29]. It utilizes multiple PostgreSQL instances to distribute both data and queries across the nodes (in this paper we deployed distributed setups with 3, 6 and 9 nodes). Thus, it provides efficient management of large datasets and high query volumes. In a Citus cluster (see Fig. 1), data is divided into shards and distributed across worker nodes. The master node handles the metadata for the distributed tables, coordinates query execution by delegating tasks to worker nodes, and aggregates the results. Worker nodes store the data shards and execute the queries [30].

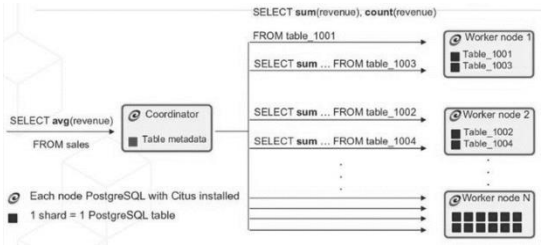


Fig. 1. Citus architecture ([29])

This architecture enables Citus to uphold the robustness and flexibility of PostgreSQL. At the same time, it delivers substantial performance improvements for OLAP workloads [31].

MongoDB is the most popular NoSQL product [7] which proved to be a reliable solution in providing database scalability, flexibility, and high performance. The database uses JSON-like documents with flexible schemas, which makes it easier to manage various data types and speeds up the development process.

Moreover, it employs a sharding architecture (Fig.2) to spread the processing load and storage across multiple servers. By horizontal scalability, MongoDB can handle large volumes of data with good overall performance in data processing. Data is split

into smaller, more manageable units known as shards, which can be hosted on separate servers or containers for better and easier management.

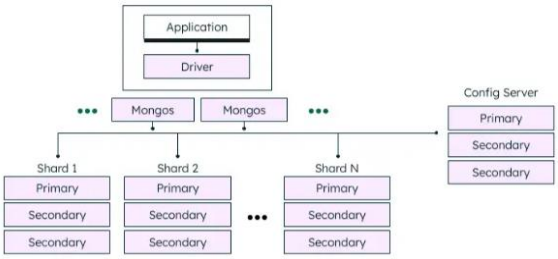


Fig. 2. MongoDB architecture ([32])

MongoDB distributes queries across these shards, allowing it to efficiently handle large data volumes and high traffic loads [32]. The data chunks are duplicated among shards to ensure high availability. Mongos instances route queries and manage data distribution, while config servers handle metadata and configuration settings.

MongoDB supports replication through replica sets, which further enhances data availability and fault tolerance in critical systems. Each replica set includes a primary node for writes and multiple secondary nodes that maintain copies of the data, these are in pairs of 3 machines in most cases. If failure occurs, one of the secondary nodes is automatically promoted to primary, ensuring minimal downtime and data consistency.

More details on Citus and MongoDB distributed setups are provided in section 4.3.

4 The experimental design. Research method and tools.

We created an initial set of 296 SQL queries (for the TPC-H benchmark database) in PostgreSQL and translate them into MongoDB Aggregation Framework queries on a JSON schema with nested arrays in collections (as described in section 4.1). The queries were executed multiple times on both servers, with a 600-second timeout, varying the database size (scale factor) and the data distribution (the number of nodes).

Whereas most of the previous studies presented in section 2 assessed the database performance with the query response times metric, our research focuses on the successful completion of queries within a timeout period of 10 minutes. In [6] the successful query completion was examined in relation to the database server and the nodes of data distribution using inferential statistics. In this paper we approached a series of research questions and sub-questions, also concerning the query successful completion, as follows:

• RQ1: Is the query completion/success associated with the parameters describing the query complexity?

  o RQ1a: Is query completion associated with the number of joins/lookups in the query?

  o RQ1b – Is query completion associated with the number of filters (predicates for record selection)?

  o RQ1c – Is query completion associated with the record grouping?

  o RQ1d – Is query completion associated with limiting the number of records in the result?

• RQ2: Can the query successful execution be predicted based on database size, physical setup and query complexity parameters?

  o RQ2a: Do Machine Learning classification models reliable predict the success of query execution based on the predictors related to the data volume, data distribution and query complexity?

  o RQ2b: Which are the most important variables for the prediction of query completion? How their distribution influences the query success?

4.1 Database structure in PostgreSQL and MongoDB

Since our study is focused on the OLAP performance [33] of MongoDB and PostgreSQL, we relied on the TPC-H benchmark database. This benchmark evaluates database systems performance by

using a series of business-oriented queries (which we did not use in this paper).

In PostgreSQL the database followed the original TPC-H schema consisting of eight tables, as represented in Fig. 3. For details on the table attributes and constraints, see [34] and [6].

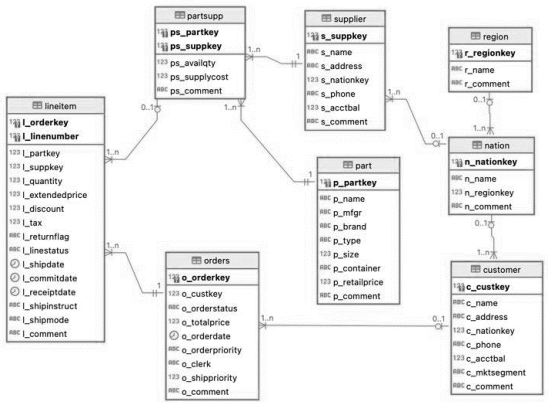


Fig. 3. TPC-H database schema ([34] [6])

When devising the corresponding MongoDB collections, we did not simply map the PostgreSQL tables into "flat" collections. Instead, we combined some tables through nesting and thus we incorporate some redundancy into the database. The resulting collections were:

- *customer_nation*
- *lineitem*
- *lineitem_orders*
- *lineitem_partsupp*
- *part*
- *partsupp*
- *region*
- *supplier_nation*.

Collections with simple names mirror the PostgreSQL tables directly, containing a straightforward JSON representation with the same attributes. The compounded collections, identified by names with underscores, merge data from two PostgreSQL tables. These collections are designed to reflect more complex relationships by nesting documents and arrays. As illustrated in Fig. 4, each document in the *lineitem_orders* collection represents an order from the *orders* table which includes (lines 7 to 29 in Fig. 4) an array of line items associated with that order, integrating data from both the *orders* and *lineitem* tables.

```
1  {   // this is the first document (describing the first order)
2      "o_orderkey": 1, "o_custkey": 370, "o_orderstatus": "O",
3      "o_totalprice": 172799.49,
4      "o_orderdate": {"$date": "1996-01-02T00:00:00.000-0700"},
5      "o_orderpriority": "5-LOW         ", "o_clerk": "Clerk#000000951",
6      "o_shippriority": 0, "o_comment": "nstructions sleep furiously among ",
7      "lineitem": [
8          { // this is the first line of the current order
9              "l_orderkey": 1, "l_partkey": 1552, "l_suppkey": 93,
10             "l_linenumber": 1, "l_quantity": 17.00, "l_extendedprice": 24710.35,
11             "l_discount": 0.04, "l_tax": 0.02, "l_returnflag": "N",
12             "l_linestatus": "O", "l_shipdate": {"$date": "1996-03-13T00:00:00.000-0700"},
13             "l_commitdate": {"$date": "1996-02-12T00:00:00.000-0700"},
14             "l_receiptdate": {"$date": "1996-03-22T00:00:00.000-0700"},
15             "l_shipinstruct": "DELIVER IN PERSON        ",
16             "l_shipmode": "TRUCK     ", "l_comment": "egular courts above the"
17         }
18         ,{  // this is the second line  of the current order
19             "l_orderkey": 1, "l_partkey": 674, "l_suppkey": 75,
20             "l_linenumber": 2, "l_quantity": 36.00, "l_extendedprice": 56688.12,
21             "l_discount": 0.09, "l_tax": 0.06, "l_returnflag": "N",
22             "l_linestatus": "O", "l_shipdate": {"$date": "1996-04-12T00:00:00.000-0700"},
23             "l_commitdate": {"$date": "1996-02-28T00:00:00.000-0700"},
24             "l_receiptdate": {"$date": "1996-04-20T00:00:00.000-0700"},
25             "l_shipinstruct": "TAKE BACK RETURN         ",
26             "l_shipmode": "MAIL      ", "l_comment": "ly final dependencies: slyly bold "
27         }
28         // ... the remaining lines of tge current order
29     ]
30 }
31 //, ... the remaining orders
```

Fig. 4. Structure of a document in collection *lineitem_orders* ([6])

In PostgreSQL the join is required when extracting data from both *lineitem* and *orders* tables; but in MongoDB's Aggregation Framework, this join (*$lookup* function) is not needed because all the relevant data is contained within a single collection.

In MongoDB the names of the collections which map two PostgreSQL tables are formatted with the child table's name on the left side of the underscore and the parent table's name on the right side. In collection *lineitem_orders*, *lineitem* is the child table and table *orders* is the parent. Each document in this collection corresponds to a record from the parent table (*orders*), and within each document, there is an array that stored all related records from the child table (*lineitem*).

4.2 SQL and Aggregation Framework Queries

The initial 296-query set (written in the PostgreSQL dialect of SQL) was devised to ensure some variability on the parameters describing the query complexity, as contained in the SELECT, FROM, WHERE, GROUP BY, ORDER BY, LIMIT/FETCH and OFFSET/SKIP clauses. The query set

was converted into the syntax of the main declarative query language in MongoDB – the Aggregation Framework, considering the new (nested) JSON structure.

Table 1 illustrates two scenarios of mapping SQL queries to the Aggregation Framework language, considering the MongoDB's "nested" structure (using arrays). In SQL, the 187th query involves a join between *customer*, *orders*, *lineitem* and *partsupp* tables. For the same query in MongoDB, Aggregation Framework uses the *$lookup* method to achieve a similar effect by joining data from different collections. By contrast, the 277th query requires join only in SQL, because in Aggregation Framework there is no need for join since all query data are stored in a single collection, *customer_nation*.

Table 1. Two examples of query mapping

Query/ Syntax	Query content
Q187 in SQL	select customer.c_name, orders.o_orderdate, lineitem.l_quantity,partsupp.ps_availqty from customer join orders on customer.c_custkey = orders.o_custkey join lineitem on lineitem.l_orderkey = orders.o_orderkey join partsupp on lineitem.l_partkey = partsupp.ps_partkey where l_quantity < 5 and l_tax > 0.05 and l_discount > 0.08 limit 800;
Q187 in AF	db.customer_nation.aggregate([{$unwind: ""$customer""}, {$lookup: {from:""lineitem_orders"", localField:""customer.c_custkey"", foreignField: ""o_custkey"", as: ""orders_1join""}}, {$unwind: ""$orders_1join""}, {$lookup: {from: ""lineitem"",localField: ""orders_1join.o_orderkey"", foreignField: ""l_orderkey"", as: ""lineitem_2join""}}, {$match: {""lineitem_2join.l_quantity"": {$lt: 5}, ""lineitem_2join.l_tax"": {$gt: 0.05}, ""lineitem_2join.l_discount"": {$gt: 0.08}}}, {$lookup: {from: ""partsupp"",localField: ""lineitem_2join.ps_partkey"", foreignField: ""l_partkey"",as: ""partsupp_3join""}}, {$project: { _id: 0, c_name:""$customer.c_name"", o_orderdate:""$orders_1join.o_orderdate"", l_quantity: ""$lineitem_2join.l_quantity"", ps_availqty: ""$partsupp_3join.ps_availqty""}}, {$limit: 800}]);
Q277 in SQL	select count(c_custkey), n_name from customer join nation on customer.c_nationkey = nation.n_nationkey where c_acctbal > 0 and c_mktsegment = 'FURNITURE' and n_regionkey = 1 group by n_name;
Q277 in AF	db.customer_nation.aggregate([{$unwind: "$customer"}, {$match: {"customer.c_acctbal": { $gt: 0 }}}, {$match: {"customer.c_mktsegment": "FURNITURE "}}, {$match: {"n_regionkey": 1}},{$group: {_id:"$n_name",NumberOfCustomer: { $sum: 1 }}}, {$project: {_id: 0, n_name: "$_id",NumberOfCustomer:1}}]);

When translating the SQL queries into MongoDB Aggregation Framework queries for yielding the equivalent results, we strived to maintain simplicity in the syntax for both languages.

4.3 Physical Setup

The experimental setup was deployed on the RaaS-IS platform, which operates as a private cloud managed by OpenStack, as described in [6]. The RaaS-IS datacentre

architecture includes 20 servers: 3 controller nodes to ensure high availability of OpenStack services, 16 compute nodes for Virtual Machine and container provisioning, and a management server responsible for MAAS, Juju, LDAP, and network management. The storage infrastructure uses an HPE 3PAR 8440 SAN with a total raw capacity of 760 TB, of which 550 TB is usable, and includes an 8 TB SSD cache for enhanced performance. The SAN's NL-SAS HDDs are set up in RAID 5, while the SSD cache uses RAID 6 to deliver higher performance. The compute nodes have dual Intel Xeon Gold 6240 CPUs (18 cores, 2.6 GHz each), 128 GB of RAM@2933MHz, and 300 GB SAS drives in RAID 1. Together, these provide 1152 virtual cores and 2 TB RAM for the RaaS-IS project and infrastructure.

4.3.1 PostgreSQL Citus Cluster Setup

The Citus cluster was configured and partially deployed using the OpenStack Heat orchestration module, which allows for Infrastructure as Code (IaC) using built-in OpenStack tools. The cluster was set up on virtual machines within the cluster nodes running a custom-configured Ubuntu Linux 18.04.6 LTS (kernel version 4.15.0-213-generic), along with TPC-H v3.0 and Citus v11.3.
To streamline cluster management, custom bash scripts were developed to enhance monitoring, result retrieval, and automated deployment.
The Citus cluster consisted of 10 machines, designed to provide distributed parallel query execution and scalability for analytical workloads. The cluster setup was as follows:

- Coordinator Node (1 machine): This node served as the primary entry point for queries, distributing them across worker nodes while maintaining metadata and query routing.
- Worker Nodes (9 machines): These nodes handled data storage and parallel execution of queries, improving performance for analytical workloads

4.3.2 MongoDB Sharded Cluster setup

The MongoDB cluster was configured and partially configured using the OpenStack Heat orchestration module. This module uses built-in OpenStack tools to allow Infrastructure as Code (IaC). The MongoDB cluster was set up on virtual machines within the compute nodes, using a custom-configured Ubuntu Linux 18.04.6 LTS (kernel version 4.15.0-213-generic), TPC-H v3.0, and MongoDB v6.0.14. A set of bash scripts was developed to improve monitoring and configuration for retrieving results and automating the cluster setup.
The MongoDB cluster comprised 13 machines configured to ensure optimal performance and reliability, mirroring a typical production environment.
The servers were organized as follows:

- Router Node (1 machine): This server directed client requests to the correct shard.
- Config Nodes (3 machines) – these nodes managed the cluster's metadata and configuration, ensuring smooth operation.
- Shards (9 machines): The data was spread across three shards, each with three replica sets. Each set included a primary node for write operations and two secondary nodes for redundancy and high availability. The number of shards could be adjusted depending on the test scenario.

Each machine was set up with 4 cores, 4GB of RAM, and a 200GB storage volume on the SAN.

4.4 Variables (outcomes and predictors)

Table 2 shows the variables used in the analysis. The first three variables define the physical setup of each scenario in the experiment. Variable *success* refers to the outcome of the query execution (whether it was completed within the 10-minute timeout).

Table 2. Variable description

Variable	Description
db_server	name of the DBMS
scale_factor	size of the DBMS
n_of_nodes	number of nodes used for data distribution
success	binary variable whose value is TRUE when the query was successfully completed within the 10-minute timeout, and FALSE otherwise
join	number of JOIN clauses (PostgreSQL), equivalent to $lookup (MongoDB)
where	number of WHERE clauses (PostgreSQL), equivalent to $match (MongoDB)
count	number of COUNT clauses (PostgreSQL), equivalent to count: (MongoDB)
group_by	number of GROUP BY clauses (PostgreSQL), equivalent to $group (MongoDB)
having	number of HAVING clauses (PostgreSQL), equivalent to $match (MongoDB)
limit	number of LIMIT clauses (PostgreSQL), equivalent to $limit (MongoDB)

The variable names related to the query complexity are inspired from the SQL syntax, and the second column points to the Aggregation Framework equivalent feature.

4.5 Method and tools for data analysis

For RQ1a-RQ1c the analysis relied on classical inferential statisticss. For RQ2a and RQ2b we built, tuned and interpreted a series of ML classification models based on two of the most popular classification algorithms, Random Forest and Extreme Gradient Boosting (XGBoost) algorithms.
After the query results and parameters were collected and integrated, the dataset was examined using Exploratory Data Analysis. The Chi-square test of independence was applied to assess the association between nominal variables. We employed Random Forest and XGBoost to identify the most important predictors for the success of query execution.

The dataset was processed and analyzed using R programming language [35], employing tools from the *tidyverse* package collection [36] along with *ggstatsplot* [37]. The *tidyverse* suite is a set of R packages designed for data manipulation and visualization, while *ggstatsplot* and *rstatix* provide additional functionalities for statistical analysis and visualization of results. All models were built and tuned with the *tidymodels* ecosystem [38].
All the Interpretable ML techniques (Variable Importance, Partial Dependency Profiles, Individual Conditional Expectation, and Accumulated Local Effects Profiles) were deployed using the *DALEX* ecosystem ([39]).

5 Results and discussion

All 296 queries were run on both database servers across six different scenarios. The chart in Fig. 5 illustrates the distribution of query parameters, providing a glimpse into the queries complexity.

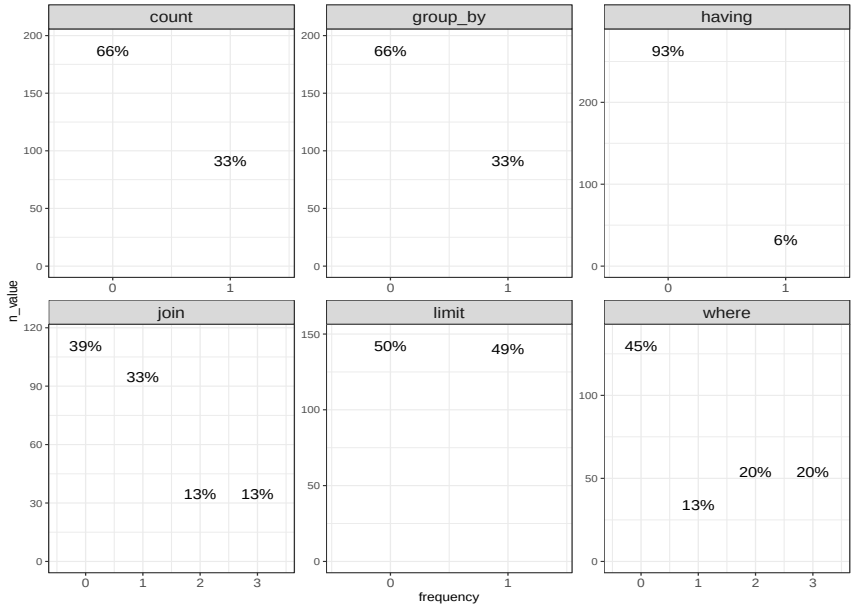


Fig. 5. Distribution of queryparameters

The chart reveals that 33% of the queries involved grouping records and performing aggregation by counting the records within each group, 45% had no filters applied, and roughly half of the queries imposed a limit on the result size.

As presented in [6], there is a correlation between query completion (within the 10-minute timeout) and the database server. Of the 3552 queries executed across both servers in all six scenarios, 315 failed to complete. Of these, 309 failed on MongoDB, while only six failed on PostgreSQL/Citus. Of the 3237 successful queries, 1770 (55%) were executed on Citus, with the remaining 1467 (45%) completed on MongoDB.

Also, [6] found that the relationship between query completion (within the 10-minute timeout) and the data distribution setup differs between database servers. In Citus, the proportion of successful queries remains consistent across the three data distribution scenarios. However, MongoDB shows more variability: the 6-node setup had the highest failure rate, while the 9-node setup performed the best.

The subsequent series of statistical tests concerned the associations between the successful query completion and some variables describing the query complexity. To start with *RQ1a: (Is query completion associated with the number of joins/lookups in the query?)*, Fig. 6 shows the results of the Chi-Square test of independence between variables *success* and *joins* for PostgreSQL/Citus (left) and MongoDB (right).

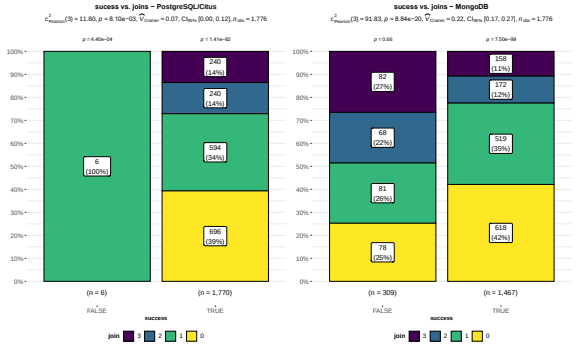


Fig. 6. Association query success vs. number of joins

In Citus, successful and unsuccessful queries recorded a similar average number of joins (1). By contrast, in MongoDB, the joins/lookups appear more costly since for the completed queries (1467), the average number of joins was 0.91, whereas, for the canceled queries (309), the average number of joins was 1.50.

For Citus, the Chi-square test of independence indicated a p-value below 0.05 and an effect size (*Cramer's V*) of 0.07,

providing evidence to state that the number of joins is significantly linked to the successful completion of queries within the 10-minute timeout. Meanwhile, for MongoDB, the Chi-square test computed a p-value less than 0.05 and a *Cramer's V* effect size of 0.22, which offer support to state that the number of joins is significantly related with the successful completion of queries within the 10-minute timeout.

Regarding *RQ1b (Is query completion associated with the number of filters /predicates for record selection?),* Fig.7 shows the results of the Chi-Square test of independence between variables *success* and *filters* for Citus (left) and MongoDB (right).

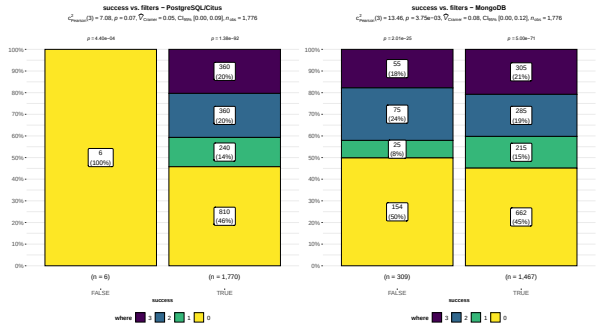


Fig. 7. Association success vs. number of filters

In Citus, for the successful queries (1770), the average number of filters was 1.14 and for the unsuccessful queries (6) the average number of filters was 0. In MongoDB, successful and unsuccessful queries recorded a similar average number of filters, ~ 1.

For Citus, the Chi-square test recorded a p-value greater than 0.05 and an effect size (*Cramer's V*) of 0.05, suggesting that the number of filters is not associated with the successful completion of queries. Similarly, for MongoDB, the Chi-square computed a p-value smaller than 0.05 and an effect size (Cramér's V) of 0.08, indicating that the number of predicates is not significantly associated with the successful completion of queries, just as observed in Citus.

Addressing the next research question, *RQ1c - Is query completion associated with grouping (aggregation) the records?*, Fig. 8 shows the findings of the Chi-Square test of independence between the variables *success* and *group_by* for both Citus (left) and MongoDB (right).

In Citus, the unsuccessful queries (6) included those that were executed without any record grouping. In the case of successful queries (1770), 66% were executed without record grouping, whereas 34% included it. In contrast, MongoDB experienced 309 unsuccessful queries, the majority of which did not feature record grouping, while out of 1467 successful queries, only 37% included record grouping.

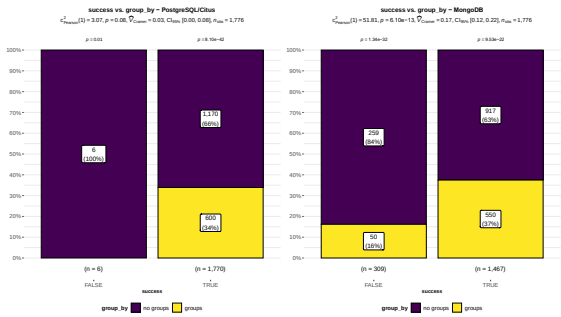


Fig. 8. Association success vs. record grouping

For Citus, the Chi-square test computed a p-value greater than 0.05 and an effect size (*Cramer's V)* of 0.03, which fail to support a significant association between record grouping and query completion. Conversely, for MongoDB, the Chi-square test of independence showed a p-value less than 0.05 and a higher effect size (0.17), which points to a strong association between record grouping and query completion.

Regarding the subsequent research question, *RQ1d – Is query completion associated with fixing a limit of records in the result?*, Fig. 9 presents the outcomes of the Chi-Square test of independence between variables *success* and *limit* for Citus (left) and MongoDB (right).

In Citus, the unsuccessful queries (6) were

those executed without limiting the result size, whereas for the successful queries (1770), more than a half of queries were executed without result size limitation. In contrast, MongoDB recorded 309 unsuccessful queries, the majority of which did not impose a limit on the result size. Among the 1467 successful queries, less than a half of the total applied a size limitation.

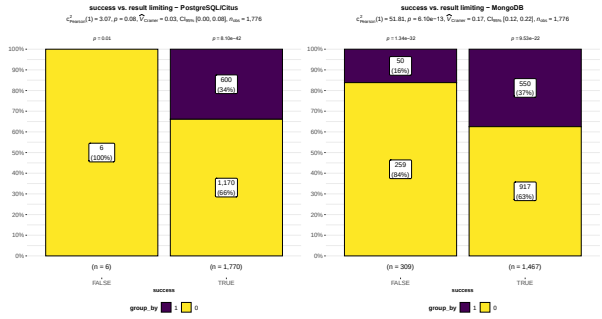


Fig. 9. Association of success and limiting the result size

For Citus, the Chi-square test of independence revealed a p-value greater than 0.05 and an effect size (*Cramer's V)* of 0.03, so limiting the result size seems not to be associated with the successful completion of queries. Alternatively, for MongoDB, the p-value of the test was below 0.05 and the effect size (0.17) was higher; for this database server, limiting the result size seems associated with the query completion.

Next, we examined whether the predictors in Table 2 can reliably predict the query completion. We built and tuned a series of ML classification models, with *query_completion* (variable that recorded the query success) as the outcome (target) and the remaining variables in Table 2 as the predictors. As described in Section 4.5, only two classification algorithms were employed for this paper, Random Forest (RF) and XGBoost. The metric performance for selecting the best model was *ROC_AUC*. Best RF performance was recorded for the hyper-parameter combination of *mtry* = 9 and *min_n* = 13, whereas for the best hyper-parameter combination for XGB was *mtry*=7, *min_n*=5, *tree_depth*=9, *learn_rate*=0.0618, *loss_reduction*=0.0000000189, and *sample_size* =0.509.

Fig.10 displays the performance of both algorithms on the new data (the test test). RF slightly overperformed XGB, with a *ROC_AUC* of 0.968 (relative to 0.965 for XGB). Recorded accuracy was higher for XGB - 0.950 (relative to 0.945 for RF). Consequently, for further estimation of predictor's importance, the RF best model was preferred.

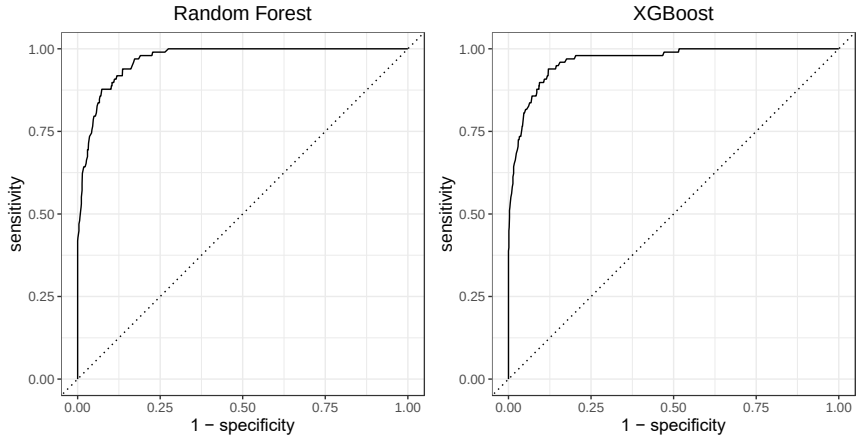


Fig. 10. Performance of the tuned classification models on the test set

Fig. 11 displays the permutation-based importance of variables as estimated by the RF algorithm. The graphical representation shows that the variable *db_server_pg.citus* has the highest importance, as permuting this variable results in the most significant performance loss, followed by the variables *join*,

scale_factor and *n_of_nodes*. Variables *group_by*, *having*, and *count* have the lowest importance with a minimal impact on the model's performance.

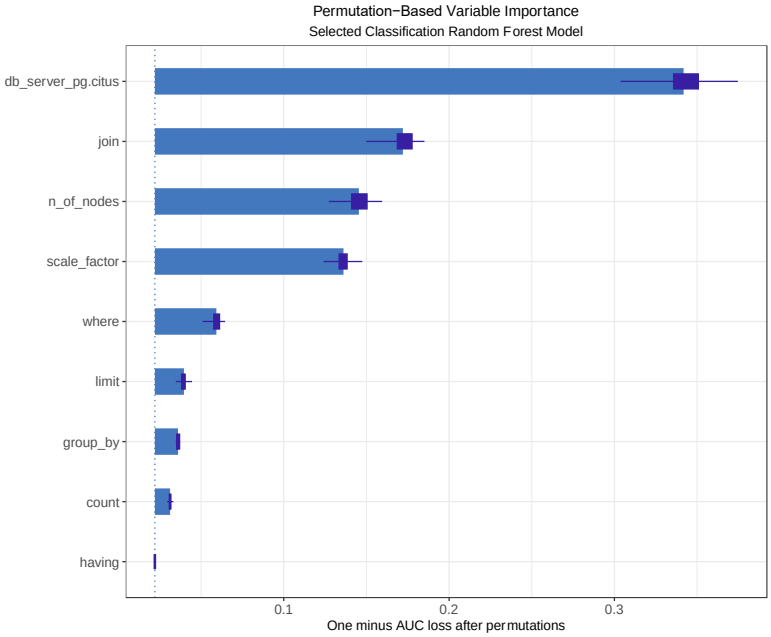


Fig. 11. Permutation-based variable importance on RF classification model

The six most influential variables in Fig.11 were selected for computing and displaying their effects on the average prediction of the outcome probability (probability of query to be successfully completed), using some techniques of interpretable ML. Fig. 12 shows three metrics describing the predictor's effect, Partial Dependency Profiles (PDP), Conditional Dependency Profiles (CDP) and Accumulated Local Effects Profiles (ALE).

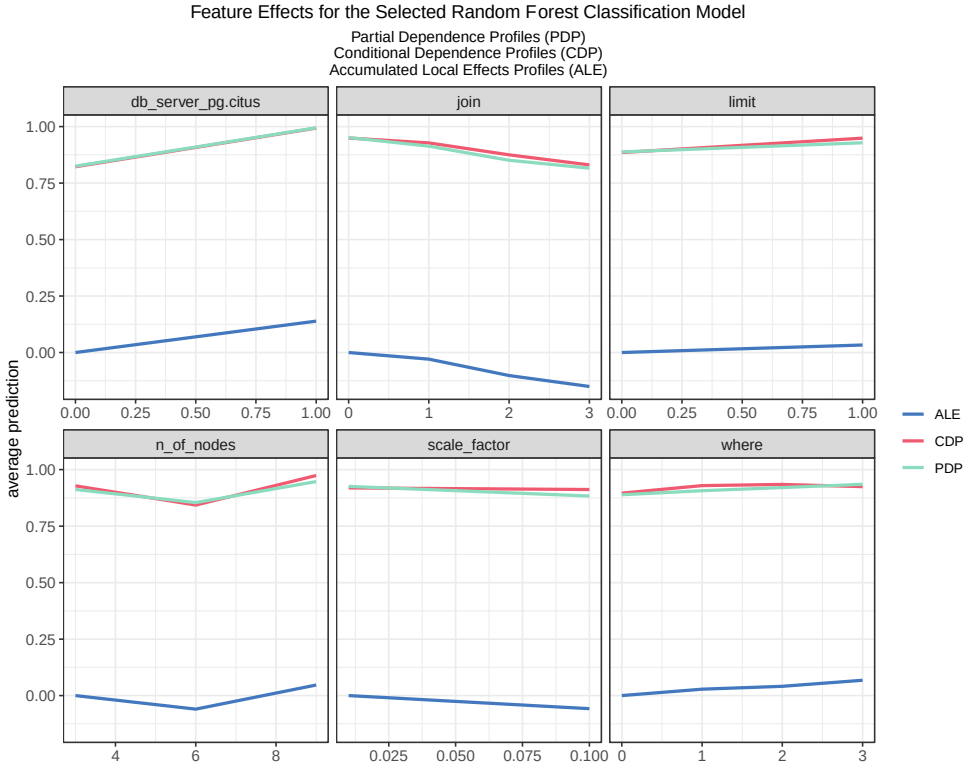


Fig. 12. Model explanations for the selected RF classification model.

Regarding the *db_server_pg.citus* variable, the average prediction of query completion probability appears to be positively associated with the use of the Citus database server, as indicated by the upward trend of the curve. For the *join* variable, the average prediction of query completion probability decreases as more joins are added in constructing the query. Regarding variable *limit*, setting a limited size of records contributes to achieving a higher average prediction. As for the *n_of_nodes* variable, employing either 3 or 9 nodes increase the odds of success for query completion. Unexpectedly, the influence of the database size (*scale_factor*) does not seem to differ between the two levels relatively, which can be explained by the small scales of 0.01 and 0.1 GB used. As for variable *where*, the average prediction of query completion increases when additional where clauses are added into the query.

6 Conclusion

Consistent with some of the earlier research, our findings indicate that distributed PostgreSQL/Citus outperforms MongoDB regarding query performance for smaller databases, even when both systems are subjects of the same requirements and physical resources. Also, we identified that the most critical factor influencing query execution is the type of database server, followed by the complexity of joins used in query construction and the data distribution setup.

The primary limitation of the experimental setup is the small database size, which, at 0.01 GB and 0.1 GB, is modest by OLAP standards. Additional limitations include the relatively small query set (296 queries), the complexity of the queries, and the brief 10-minute timeout imposed for query execution.

Future experiments on the setup should involve processing larger datasets and employing different data nesting structures in MongoDB collections. Moreover, increasing the query set size, expanding the variability of query parameters, and exploring various data distribution scenarios could provide more comprehensive insights.

7 Acknowledgment

Data processing and analysis for this study were supported by Competitiveness Operational Program Romania under project SMIS 124759 - RaaS-IS (Research-as-a-Service Iasi).

References

[1] C. Madera and A. Laurent, “The next information architecture evolution: the data lake wave,” in *Proc. 8th Int. Conf. on Management of Digital EcoSystems (MEDES)*, 2016, pp. 174-180. doi: 10.1145/3012071.3012077.

[2] F. Naregsian, E. Zhu, R. Miller, K. Pu, and P. Arocena, “Data Lake Management: Challenges and Opportunities,” *Proc. VLDB Endow.*, vol. 12, no. 12, pp. 1986-1989, 2019. doi: 10.14778/3352063.3352116.

[3] P. Pedreira, O. Erling, K. Karanasos, S. Schneider, and W. McKinney, “The Composable Data Management System Manifesto,” *Proc. VLDB Endow.*, vol. 16, no. 10, pp. 2679-2685, 2023. doi: 10.14778/3603581.3603604.

[4] D. Martinez-Mosquera, “Integrating OLAP with NoSQL Databases in Big Data Environments: Systematic Mapping,” *Big Data Cogn. Comput.*, vol. 8, no. 6, 2024.

[5] K. Prabhakar, J. Kurunandan, A. Amjad, P. Prabu, and B. Rajkumar, “OpenStackDP: A scalable network security framework for SDN-based OpenStack cloud infrastructure,” *J. Cloud Comput.: Adv. Syst. Appl.*, vol. 12, no. 1, 2023. doi: 10.1186/s13677-023-00406-w.

[6] M. Fotache, C. Badea, M. I. Cluci, C. Pinzaru, C. S. Esanu, and O. Rusu, “OLAP Performance of Distributed PostgreSQL and MongoDB on

OpenStack. Preliminary Results on Smaller Scale Factors," in *Proc. 23rd RoEduNet Conf.: Networking in Education and Research*,2024, doi: 10. 1109/RoEduNet64292.2024.10722556

[7] DB Engines Ranking [Online]. https://db-engines.com/en/ranking.

[8] A. Makris, K. Tserpes, G. Siliopoulos, D. Zissis, and D. Anagnostopoulos, "MongoDB vs PostgreSQL: A comparative study on performance aspects," *Geoinformatica*, vol. 25, pp. 243–268, 2021. doi: 10.1007/s10707-020-00407-w.

[9] E. Güney and N. Ceylan, "Response Times Comparison of MongoDB and PostgreSQL Databases in Specific Test Scenarios," in *ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, 2022, pp. 178–188.

[10] M. T. Villalobos, L. V. Acuna, and R. Q. Oviedo, "Comparison of the Response Times of MongoDB and PostgreSQL According to Type of Query in Geographical Databases," *Computación y Sistemas*, vol. 24, no. 4, pp. 1461–1469, 2020.

[11] L. G. Wiseso, M. Imrona, and A. Alamsyah, "Performance Analysis of Neo4j, MongoDB, and PostgreSQL on 2019 National Election Big Data Management Database," in *International Conference on Science in Information Technology*, Palu, Indonesia, 2020.

[12] A. Flores, S. Ramírez, R. Toasa, J. Vargas, R. U. Barrionuevo, and J. M. Lavin, "Performance Evaluation of NoSQL and SQL Queries in Response Time for the E-government," in *2018 International Conference on eDemocracy & eGovernment (ICEDEG)*, 2018, pp. 257–262. doi: 10.1109/ICEDEG.2018.8372362.

[13] Y. D. Mukasheva, A. Bissengaliyeva, D. Suynullayev, "Performance Analysis of Scaling NoSQL vs SQL: A Comparative Study of MongoDB, Cassandra, and PostgreSQL," in *2023 IEEE International Conference on Smart Information Systems and Technologies (SIST)*, Astana, Kazakhstan, 2023, pp. 479–483. doi: 10.1109/SIST58284.2023.10223568.

[14] M. M. Eyada, W. Saber, M. M. El Genidy, and F. Amer, "Performance Evaluation of IoT Data Management Using MongoDB Versus MySQL Databases in Different Cloud Environments," *IEEE Access*, vol. 8, pp. 110656–110668, 2020. doi: 10.1109/ACCESS.2020.3002164.

[16] W. Ali, M. U. Shafique, M. A. Majeed, and A. Raza, "Comparison between SQL and NoSQL databases and their relationship with Big Data analytics," *Asian Journal of Research in Computer Science*, vol. 4, no. 2, pp. 1–10, 2019. Article no. AJRCOS.51946, ISSN: 2581-8260.

[15] A. Makris, K. Tserpes, G. Spiliopoulos, and D. Anagnostopoulos, "Performance evaluation of MongoDB and PostgreSQL for spatio-temporal data," in *EDBT: 22nd International Conference on Extending Database Technology*, Lisbon, Portugal, 2019, pp. 1–9.

[16] R. Čerešňák and M. Kvet, "Comparison of query performance in relational and non-relational databases," in *Proc. of the 13th Scientific Conference on Sustainable, Modern and Safe Transport (TRANSCOM 2019)*, Novy Smokovec, vol. 40, pp. 170–177, May 29–31, 2019. doi: 10.1016/j.trpro.2019.07.027.

[17] P. M. Tracz and M. Plechawska-Wójcik, "Comparative analysis of the performance of selected database management system," *Journal of Computer Sciences Institute*, vol. 31, pp. 89–96, 2024. doi: 10.35784/jcsi.5927.

[18] A. M. Kausar and M. Nasar, "SQL Versus NoSQL Databases to Assess Their Appropriateness for Big Data

Application," *Recent Advances in Computer Science and Communications*, vol. 14, no. 4, 2021, doi: 10.2174/2213275912666191028111632

[19] A. B. Setyawan, I. A. Kautsar, and N. L. Azizah, "Query Response Time Comparison SQL and No SQL for Contact Tracing Application," in *Proceedings of the 4th Seminar Nasional Sains. Procedia of Engineering and Life Science*, vol. 2, no. 2, 2022, doi: https://doi.org/10.21070/pels.v2i2.1296.

[20] D. Figueiredo, G. Saraiva, J. Rebelo, R. Rodrigues, F. Cardoso, C. Wanzeller, P. Martins, and M. Abbasi, "Performance Evaluation Between HarperDB, Mongo DB and PostgreSQL," in *Marketing and Smart Technologies. ICMarkTech 2022. Smart Innovation, Systems and Technologies*, vol. 344. Springer, 2024. doi: https://doi.org/10.1007/978-981-99-0333-7_7.

[21] M. Sals, A. Sghir, N. Rafalia, and J. Abouchabaka, "Analysis and comparison of NoSQL databases with relational databases: MongoDB and HBase versus MySQL," *International Journal on Technical and Physical Problems of Engineering (IJTPE)*, vol. 55, no. 15, pp. 155–161, June 2023, ISSN 2077-3528.

[22] J. Antas, R. R. Silva, and J. Bernardino, "Assessment of SQL and NoSQL Systems to Store and Mine COVID-19 Data," *Computers*, vol. 11, no. 2, pp. 29, 2022. doi: https://doi.org/10.3390/computers11020029.

[23] B. Alyasiri, B. Sahi, and N. AL-Khafaji, "NoSQL: Will it be an alternative to a relational database? MySQL vs MongoDB comparison," in *Proceedings of 2nd International Multi-Disciplinary Conference Theme: Integrated Sciences and Technologies, IMDC-IST 2021*, Sakarya, EAI, 7-9 September 2021. doi: 10.4108/eai.7-9-2021.2314925.

[24] M. Abukabar, S. Abukabar, and U. M. Bello, "Analyzing and Designing an Evaluation Benchmark for SQL and NoSQL Database Systems for Some Selected Higher Institution in Zamfara State," *International Journal of Science for Global Sustainability (IJSGS)*, vol. 10, no. 2, p. 63, 2024, doi: 10.57233/ijsgs.v10i2.644.

[25] H. Matallah, G. Belalem, and K. Bouamrane, "Comparative study between the MySQL relational database and the MongoDB NoSQL database," *International Journal of Software Science and Computational Intelligence (IJSSCI)*, vol. 13, no. 3, pp. 38–63, 2021. doi: 10.4018/IJSSCI.2021070104.

[26] C. Axell, E. Schøien, I. L. Thon, L. O. Vågene, and L. Tveiten, "Insertion speed of indexed spatial data: comparing MySQL, PostgreSQL and MongoDB," 2022. *[Online]*: https://folk.idi.ntnu.no/baf/eremcis/2022/Group02.pdf.

[27] M. Reichardt, M. Gundall, and H. D. Schotten, "Benchmarking the operation times of NoSQL and MySQL databases for Python clients," in *IECON 2021 – 47th Annual Conference of the IEEE Industrial Electronics Society*, Toronto, Canada, 2021, pp. 1-8, doi: 10.1109/IECON48115.2021.9589382.

[28] N. Naufal, S. Nurkhodijah, G. B. Anugrah, A. Pratama, M. I. Rabbani, F. A. Dilla, T. N. Anggraeni, and T. Firmansyah, "Comparisonal analysis of MySQL and MongoDB response time query performance," *Jurnal Informatika Dan Tekonologi Komputer (JITEK)*, vol. 2, no. 2, pp. 158-166, 2022. https://doi.org/10.55606/jitek.v2i2.245

[29] Citus Data Documentation, *[Online]*: https://docs.citusdata.com

[30] L. F. Silva and J. V. F. Lima, "An evaluation of relational and NoSQL distributed databases on a low-power

cluster," *J. Supercomput.*, vol. 79, pp. 13402–13420, 2023. https://doi.org/10.1007/s11227-023-05166-7

[31] U. Cubukcu, O. Erdogan, S. Pathak, S. Sannakkayala, and M. Slot, "Citus: Distributed PostgreSQL for data-intensive applications," in *Proc. of the 2021 International Conference on Management of Data*, ACM, 2021, doi: 10.1145/3448016.3457551.

[32] MongoDB Documentation, *[Online]*: https://www.mongodb.com/docs/manual/

[33] T. Taipalus, "Database management system performance comparisons: A systematic literature review," *J. Syst. Softw.*, vol. 208, no. 111872, 2024, doi: 10.1016/j.jss.2023.111872.

[34] Transaction Processing Council Benchmark H (Decision Support) Standard Specification, *[Online]*: https://www.tpc.org/TPC_Documents_Current_Versions/pdf/TPC-H_v3.0.1.pdf.

[35] R, "R: A Language and Environment for Statistical Computing," R. C. Team, Producer, & R Foundation for Statistical Computing, R version 4.4.0, 2024. Available: https://www.R-project.org.

[36] H. Wickham et al., "Welcome to the Tidyverse," *J. Open-Source Softw.*, vol. 4, no. 43, pp. 1–6, 2019, doi: 10.21105/joss.01686.

[37] I. Patil, "Visualizations with statistical details: The 'ggstatsplot' approach," *J. Open Source Softw.*, vol. 6, no. 61, 2021, doi: 10.21105/joss.03167.

[38] M. Kuhn and J. Silge, *Tidy Modeling with R*, Sebastopol, California, USA: O'Reilly, 2022.

[39] P. Biecek, "Dalex: Explainers for complex predictive models in R," *J. Mach. Learn. Res.*, vol. 19, no. 84, pp. 1–5, 2018.

Marin FOTACHE graduated from the Faculty of Economics at Alexandru Ioan Cuza University of Iasi, Romania in 1989. He holds a PhD diploma in Business Information Systems (Business Informatics) from 2000 and he had gone through all didactic positions since 1990 when he joined the staff of Al. I. Cuza University, from teaching assistant in 1990, to full professor in 2002. Currently he is professor within the Department of Accounting, Business Informatics and Statistics in the Faculty of Economics and Business Administration at Alexandru Ioan Cuza University. He is the (co)author of books and journal/conference articles in areas such as SQL, database design, NoSQL, Big Data, Data Engineering and Machine Learning.

Cătălina BADEA completed her Master's degree in Data Mining at the Faculty of Economics and Business Administration, Alexandru Ioan Cuza University of Iași. Since 2024, she has been enrolled as a PhD student at the Doctoral School of Economics and Business Administration, focusing on Business Informatics. Her doctoral research approaches performance and architectural problems in Big Data platforms, exploring the transition from Data Lake to Data Lakehouse. Her conference participation includes RoEduNet: Networking in Education and Research 2024, Globalization and Higher Education in Economics and Business Administration 2024, and International Conference on Informatics in Economy (IE 2025). She serves as an associate teaching staff

member at Alexandru Ioan Cuza University of Iași, where she conducts laboratory sessions in database systems. Her research interests focus on Big Data system architectures and the performance of data processing in business applications.

Marius-Iulian CLUCI graduated from the Faculty of Economics and Business Administration at Alexandru Ioan Cuza University of Iași, Romania. He holds a master's degree in Software Development and Business Information Systems and currently he is a Ph.D. candidate at the Doctoral School of Economics and Business Administration. His research focuses on Big Data systems, the integration of Machine Learning in Apache Spark, and performance benchmarking of modern data architectures. He has taught on topics related to Big Data, Machine Learning, OpenStack, and distributed computing. He works as a Cloud Engineer on Microsoft Azure, designing scalable data integration and processing solutions. His academic contributions include papers on TPC-H benchmarking, Spark optimizations, and schema evolution. His areas of expertise include cloud computing, Apache Spark, Machine Learning, and data integration.

Codrin-Stefan Esanu graduated from the Faculty of Economics and Business Administration Iasi, specializing in Business Informatics, and obtained his Master's degree in Software Development and Business Information Systems. Currently, he is pursuing a Ph.D. at the Doctoral School of Economics and Business Administration. Started his career in the IT Service Management industry at Capgemini, he advanced quickly towards management roles. Leveraging the solid foundation in managing critical IT services, he successfully transitioned to a technical DevOps role at Cegeka Romania. Additionally, he serves as a university associate lecturer at Alexandru Ioan Cuza University in Iași, teaching courses in databases, Big Data and distributed computing. His professional expertise includes Linux administration, Scripting, Infrastructure as Code (Puppet, Ansible), Containers orchestration (OpenShift), GitOps (ArgoCD) and database systems like PostgreSQL, Citus, and Neo4j.

An Overview of Big Data and NoSQL in the Video Game Industry

Cristiana COSTAN
Bucharest University of Economic Studies
Faculty of Cybernetics, Statistics and Economic Informatics
Bucharest, Romania
costancristiana21@stud.ase.ro

The connection between relational and non-relational databases and game development is covered in this article. The concept of big data and its implications in the gaming industry is also stated, the capability to store user data becoming essential for marketing and game improvements. The ideas in this article are reinforced by creating a web application to see how NoSQL databases can be implemented in game development and how they can help us extract relevant data about users. Overall, the findings of this paper are intended to guide people who want to be part of or are part of the gaming industry in the choices related to which database to utilize in future games and to provide insights on how they might obtain information about potential consumers to improve their products or services.

Keywords: *Big Data, NoSQL, Video Games, Gaming Industry*

1 Introduction

The video game market is shifting so quickly that in order to have an impact, game developers need to design products with a strong emphasis on user needs. Companies are investing more money than ever in creating games designed to achieve popularity and generate profits. As video games have become a popular form of entertainment and relaxation, more and more individuals are choosing to spend their leisure time playing them. Game producers can offer potential consumers the experiences they desire due to the evolution of data and its integration into various industries. A lot of games have to be able to store information on player accounts, characters, or even in-game items. Depending on the situation, relational databases or non-relational databases may be used to store such information. For instance, non-relational databases are preferred when the stored data changes frequently, as their scalability is crucial in managing multiple modifications made in short periods of time. In contrast, relational databases are preferred when the game information is static and does not change often. Additionally, in some cases, both types of databases can be utilized for the same product, allowing the game to include simultaneously static and dynamic components that can be stored in different formats, corresponding to each database used.

The volume of data generated by the large number of users who play video games is massive, therefore NoSQL is mostly utilized for storing such data because it is more scalable, faster, cheaper, and flexible, addressing challenges in real-time applications like online games that relational databases are unable to solve.

Big data is often used for analytics in the gaming industry. Analysis of player information assists in the development of marketing guidelines for the gaming industry and the improvement of games, whether they are still in testing or are already available on the market. Thus, gaming companies can effectively promote their game and retain as many of the users they attract.

2 Overview of Big Data

Big data refers to massive data sets with a large, complex, and varied structure [1]. These data are produced by online transactions, logs, posts, search queries,

mobile and computer applications. Their massive growth led to difficulties in storing, managing, analyzing, and visualizing data through standard database software tools.

There are three main characteristics that define big data: variety, velocity, and volume. Through variety, big data generally has three types, depending on the distinct sources it might come from structured, semi-structured, and unstructured. Structured data is defined as data that is already labeled and easily sorted, whereas unstructured data is unpredictable and challenging to analyze. Semi-structured data is in the middle, not conforming to specified fields but having tags to distinguish data items. The volume of big data is larger than terabytes and petabytes (10^{15} bytes), and its size still increases. Velocity, which is crucial for organizations when they acquire data, is the rate at which data may be created and processed.

According to Dr. Aditya Vailaya, an expert in computer science, machine learning, and statistical pattern recognition, large amounts of user information can be utilized to improve product recommendations or deliver targeted information [2]. Big data analytics has applications in merchandising, pricing, marketing, and advertising. Understanding users' interests based on previous purchases or preferences and understanding the characteristics of best-selling products are only two of the analytical uses of big data.

3 SQL vs NoSQL

Structured Query Language (SQL) is a non-procedural data access language developed in the 1970s at the IBM research laboratories [3]. Nowadays, most database management systems implement a dialect of SQL, setting SQL as the predominant language for enterprise databases in our surroundings.

The term "NoSQL" was first utilized in 1998 to refer to a relational database that did not employ SQL [4]. The main reason for the NoSQL movement was stated by Eric Evans, a blogger who contributed to popularizing the term. He declared that this alternative appeared due to the need to solve problems that relational databases could not. Nowadays, as NoSQL might include SQL, it is usually considered as "not only SQL" by developers.

NoSQL databases are preferred over SQL databases in certain situations when scalability, high availability, and low latency are needed [5]. The scale-out strategy on which NoSQL databases are often based enables a cheaper process of scaling to large data volumes and provides the ability to upgrade or change the structure of a database with no downtime [6]. NoSQL is mostly utilized to manage big data and real-time web applications. Four different forms of data storage are included in NoSQL databases: document, key-value, graph, and column. The performance of NoSQL databases varies according to their data model.

Relational databases use the ACID (Atomicity, Consistency, Isolation, Durability) model for transaction processing due to its high reliability and consistency [7]. Atomicity refers to transaction failure and commits. If it were for a transaction to fail, the database would be left in the state before the transaction started. Only when all the transactions are successful the data would be updated. Consistency represents the stable state in which the database needs to be both before and after a transaction occurs. Isolation refers to a transaction being able to start only when another transaction finishes. Durability means that a completed transaction has permanent effects in case of later system failures. Because of their different priorities, NoSQL databases use the BASE (Basic Availability, Soft state, Eventual consistency) model, derived from the CAP (Consistency, Availability, and Partition-tolerance) theorem. Maintaining a single copy of current data, having the data available for updates, and being able to

continue operating even in the presence of network partitions represent the elements of the CAP theorem. “ACID properties are to RDBMS as BASE is to NoSQL systems.” [8]. Basic availability refers to the extent to which a user can modify the data. During periods of disconnection or synchronization delays, the user can still make changes and his actions will synchronize with the database later, once connectivity is restored. Soft state represents changing the state without any input, which ensures consistency. Eventual consistency means that if an updated database element is not further updated for a significant period of time, all users will see the same updated item value.

Addressing relational database management systems (RDBMS) vs NoSQL in detail, both have their advantages and disadvantages. NoSQL is easily scalable, the database does not necessarily need administrators, is faster, cheaper, and more flexible, some of the suppliers can manage hardware failure, and is utilized for big data applications. RDBMS is easy to design, execute, maintain, and use, and information is kept in one place, being secured. When looking at drawbacks, each disadvantage of NoSQL can be solved by SQL and vice-versa. For example, SQL presents issues in high availability, which can be managed by using NoSQL.

Modern big data-driven applications require the performance and scalability offered by NoSQL. Cisco replaced Oracle RAC with Neo4J, while Netflix switched from Oracle RDBMS to Apache Cassandra. These changes improved latency, shortened query times, and decreased expenses. These are only a few of the benefits brought by the usage of NoSQL databases. NoSQL databases are considered a better fit for handling large datasets with a lot of data streaming quickly from various sources [9].

4 The Video Game Industry

Video gaming is considered to belong to the field of culture and involves recreational activities [10], allowing individuals to distance themselves from their everyday problems by redirecting their attention to new goals from within imaginary worlds.

“The development of this industry characterizes our generation: fast paced, technologically oriented, and targeted toward the young and young at heart.” [11]

The first computer game, called “Spacewar”, was developed more than 50 years ago by Steve Russell, a Massachusetts Institute of Technology (MIT) student, as a demonstration of expertise and not for profit. In a short period of time, this game gained popularity within the United States of America (USA) and entrepreneurs sought the opportunity to make a profit off of this, game development becoming a business. The starting point of this industry began from the enjoyment of Nolan Bushnell playing Spacewar during his time at university, which led to the development of “Atari” in 1972.

Nowadays, the video game industry is growing rapidly, with over 10 video game genres, including sandbox, real-time strategy (RTS), shooters, and multiplayer online battle arena (MOBA) [12]. The strong competition determines companies to constantly improve game graphics and features in order to enhance user experience. User feedback represents the key factor in game development. In 2018 the production costs of some video games ranged from $100 million to $500 million. These gigantic investments represent a risk taken by the companies, the profits depending solely on the users.

5 SQL and NoSQL in the Gaming Industry

Choosing RDBMS or NoSQL when creating a game is up to the developers nowadays. If the game is based on static elements one can utilize relational databases and whether a faster save of the data is needed NoSQL can be employed. The benefits of NoSQL can be

seen, for instance, in scoreboards and player statistics, which are dynamic components. Table 1 highlights the main advantages of utilizing NoSQL databases in the development of video games.

Table 1. Advantages of NoSQL in the Video Game Industry

Use Case	Advantages of Utilizing NoSQL
Storing User Profiles	NoSQL's flexibility and scalability enable the rapid creation and continuous updates of user profiles, allowing many users to manage their accounts efficiently.
Storing User Items	In video games where inventories contain items with attributes that change based on user modifications, NoSQL can efficiently manage a wide variety of items with diverse attributes, handling large amounts of data.
Storing Game States Which Adapt to Player Choices	NoSQL efficiently stores and manages dynamic data from player decisions that impact game progression and outcomes.
Storing Leaderboard	NoSQL is ideal for managing leaderboard data, enabling fast updates and optimized querying.
Storing Player Statistics	Player statistics are frequently generated during gameplay, and NoSQL is crucial for handling the volumes of data produced.

In 2018 Fortnite announced hitting a new peak of 3.4 million concurrent players and shared within their article the "challenges of rapidly scaling a game and its online services" [13]. Fortnite has a service named MCP which is comprised of 9 MongoDB shards, utilized for retrieving data regarding game profiles, statistics, items, matchmaking info, and more.

In the same year, Riot Games wrote in their blog about deploying League of Legends to more than 12 disparate game shards, divided by regions [14]. MySQL was employed as the database backend for storing player account information, Riot putting a high priority on ACID. Both Riot and their parent company, Tencent, widely utilize MySQL and thus have a significant number of engineers who are familiar with how it operates. Because of this and the type of data that needs to be stored, MySQL is employed to protect user accounts.

In 2022 Amazon Web Services (AWS) posted in their AWS for Games Blog the architecture employed in "New World", a massively multiplayer online role-playing game [15]. Every second, millions of states change, thus appearing their need of utilizing Amazon DynamoDB, a NoSQL database which provides high performance and reliability and is suitable for game data, which frequently lacks organization and coherence. Moreover, employing Amazon DynamoDB allowed developers to successfully expand the game from a singular test server during development to over 500 in production without making any changes to the code.

Blizzard Entertainment, a video game developer and publisher, has listed on its careers page many jobs, including in Engineering & Technology. For Overwatch, a game which in 2024 surpassed 100 million players [16], experience with SQL databases was a plus, whilst for World of Warcraft (WoW), another popular game released by Blizzard, knowledge of relational or non-relational databases represented a plus. It is possible to conclude from these optional requirements that WoW uses both SQL and NoSQL, while Overwatch uses SQL.

Thus, the decision between RDBMS and NoSQL depends on the needs of the game developers. Developers prefer RDBMS

when dealing with static data, such as game characters, fixed-attribute items, game rules, and dialogues that don't change based on player actions. In contrast, NoSQL is more popular when dealing with player statistics, multiplayer leaderboards, match scores, and game states that adapt to player choices.

6 Big Data's Impact on Gaming

The gaming industry typically utilizes big data for analytics. The term "big data analytics" refers to the processing and analysis of massive amounts of data in order to gain useful information [17]. With the aim to assist analysts in making data-informed decisions, big data analytics makes it possible to find trends, patterns, and correlations in large amounts of raw data.

Big data analytics offers those in the gaming sector the ability to analyze large data sets, consumer preferences, market trends, and other crucial business data [18]. This information aids gaming companies in improving player experiences and developing customized products and services. Furthermore, a new era of gaming has emerged as a result of the application and review of game analytics, in which data analysis is now crucial to the success of game development businesses rather than just advantageous.

Nowadays, various methods are employed in big data analytics to enhance the gaming experience and increase user engagement. Player behavior analysis, which examines in-game movements, purchase history, and playtime patterns can lead to improved engagement and higher customer loyalty. Predictive analytics utilize historical data to predict future trends and player actions, aiding in improving player satisfaction and reducing game dropouts. Personalization algorithms leverage machine learning to create game recommendations, in-game content customization, and adapt advertising messages based on player information. Real-time analytics enable immediate correction of any problems, improving the overall gaming experience by tracking and examining server performance, live player behavior, and in-game transactions. Heat maps and spatial data analysis focus on player activities from within the game environment in order to improve level design, balance game difficulty, and ensure an engaging player experience. A/B testing allows game developers to compare two versions of a game or a feature among various player segments to determine which performs better and is essential for making data-driven decisions about game design and updates, ensuring that modifications reflect user preferences. Finally, the economic analysis examines player spending patterns, the balance of the virtual economy, and pricing strategies to optimize in-game economies, ensuring they remain engaging, fair, and profitable.

In the rapidly evolving gaming industry, these analytical methods provide gaming companies valuable insights on player behavior and market trends, enabling them to provide their customers with quality products and services that suit their preferences.

Ubisoft, SEGA Europe, and Kolibri Games represent only some examples from the gaming industry in which big data analytics are employed to ensure player engagement through personalized content, targeted marketing strategies, and gameplay adjustments. SEGA, for instance, gathers more than 25,000 data events each second, such as player behavior and in-game interactions.

Another interesting topic is related to how big data analytics can be improved with the utilization of machine learning (ML). Ubisoft leverages machine learning, which evolved from years of data gathering, to analyze player behaviors and enhance game features [19]. Since the early 2010s, advancements in tools and methods have allowed them to move beyond basic analysis to applying machine learning for actionable

insights and innovation.

The involvement of machine learning in behavioral analysis is also suggested in an article that discusses the use of neural networks to observe patterns and behaviors of players in the game World of Warcraft (WoW), a popular Massively Multiplayer Online Role-Playing Game (MMORPG) [20]. The characteristics utilized were analyzed using big data and data visualization tools. This experiment was conducted with the aim of showing how player analysis in an online gaming environment works on a small scale when using neural networks. Additionally, other analysis methods from those previously mentioned in the paper have been used for various studies regarding the behavior of WoW players. These demonstrations show us that by using big data and neural networks, companies can streamline their analyses and improve their games more easily.

One type of artificial intelligence (AI) program that can detect and produce text, among other things, is called a large language model (LLM) [21]. Massive amounts of data are utilized to train LLMs, which are based on machine learning. Although there is still room for improvement in the understanding of data by the LLMs, accurate outputs can be achieved when a more detailed and precise user input request is made [22]. This study also demonstrates the possibilities offered by the use of neural networks in the gaming industry, this time analyzing data related to the behavior of players in a gaming community, the data being extracted from an online forum for online board games. It has been discovered that exploratory data analysis (EDA) can be used effectively in this context, being an analysis that provides important insights from data.

These findings underscore the importance of big data in the gaming industry and highlight how game developers can benefit greatly from the analysis that can be performed with this data. Additionally, the chapter illustrates how machine learning has improved this sector by making it easier to understand players through big data analysis.

7 Building a Game

For a deeper understanding of NoSQL in the gaming industry I created a browser game with states that adapt to player decisions, as illustrated in Figure 1. It contains choices from three distinct categories: health, sanity, and supplies. On every in-game day, each user answers three questions.

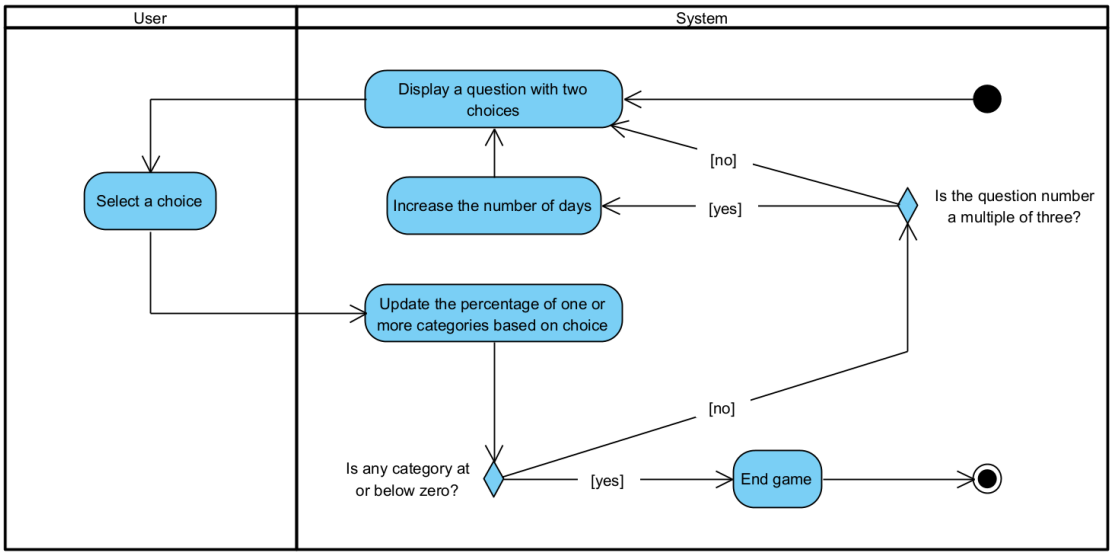


Fig. 1. Activity Diagram for Game Flow

Each category's percentage is affected by the choices made, and the game is over when one of the categories hits zero. A visual representation of the game's core features is provided in Figure 2.

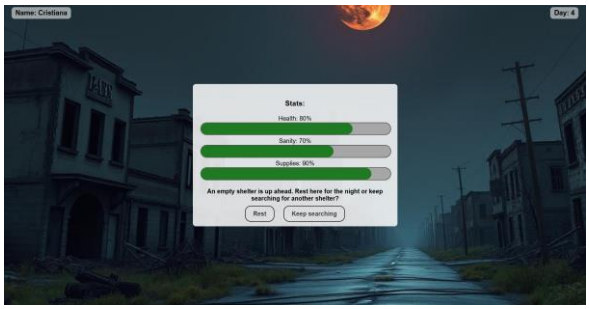


Fig. 2. Gameplay Overview

Since the game is still in its early stages, it merely keeps track of user data to generate the online leaderboard, which shows details about the players and how many days each of them survived. As depicted in Figure 3, every time the player completes the game, a POST request is made to the MongoDB database to insert the leaderboard data.

```
data = request.get_json()
days = int(data.get('days_survived'))
user = users_collection.find_one({"username": username})
high_score = int(user['high_score'])
if days < high_score:
    result = users_collection.update_one(
        {"username": username},
        {"$set": {
            "last_score": days
        }}
    )
else:
    result = users_collection.update_one(
        {"username": username},
        {"$set": {
            "last_score": days,
            "high_score": days
        }}
    )
```

Fig. 3. Python code for Updating User Score in MongoDB

The number of days survived is checked in the Python backend, entered into the database as the last score, or updated as both the last and high scores, and displayed on the final screen, as shown in Figure 4.

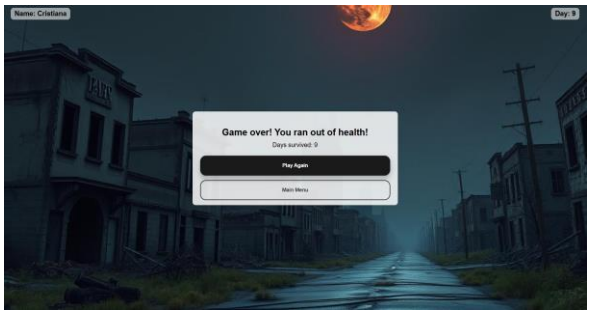


Fig. 4. Game Over Screen

All users with a score greater than zero days of surviving have their data extracted from the database and sorted in descending order, with a limit of ten users. This data is utilized to create the leaderboard. The code utilized for this process is displayed in Figure 5.

```
@app.route('/leaderboard')
def leaderboard():
    if 'username' not in session:
        return redirect(url_for('login'))
    username = session['username']
    leaderboard_info = users_collection.find({"high_score": {"$gt": 0}}).sort({"high_score": -1}).limit(10)
    return render_template('leaderboard.html', username=username, leaderboard_info=leaderboard_info)
```

Fig. 5. Python Code for Retrieving Leaderboard Data

If we were to think about the future of the game and the possibility of it becoming used by a multitude of people, the leaderboard data, as depicted in the leaderboard from Figure 6, would be extracted from a huge dataset, requiring NoSQL for its management. This will ensure continuous availability, low response times, and high-speed writes, ensuring that the database will intake data rapidly.

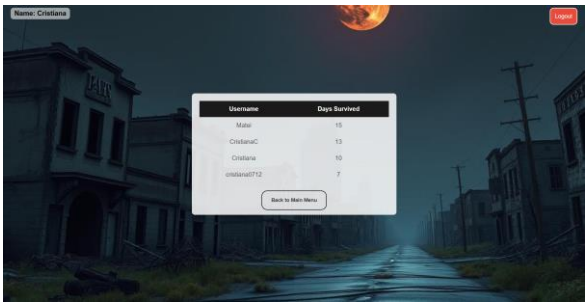


Fig. 6. Scoring Leaderboard

Taking all this into account, relational databases are recommended for developing a game that contains static features, including items with fixed basic attributes. When a game has dynamic elements, such as character traits, inventory items, states that change based on player choices, or player statistics, NoSQL databases are preferred over relational databases.

8 Conclusions and Future Work

The existence of big data has changed the gaming industry, and the emergence of NoSQL has made it easier to store this type

of data. Both relational and NoSQL databases can provide various benefits for game developers, and the choice of a database type is made depending on the needs of the respective game. NoSQL databases are preferred over relational databases in cases where games will have a huge number of players, each statistic of each player having to be stored in the database. Additionally, NoSQL makes it easier to store large amounts of data from which information can be collected, processed using pre-existing algorithms, and used to easily analyze player behavior.

In terms of data storage, relational databases and NoSQL databases are both significant for the gaming sector. While NoSQL databases are more appropriate for handling dynamic and massive volumes of player data, relational databases are better suited when utilizing static elements.

Regarding the future development of the application, potential analyses can be conducted to improve the game based on the data provided by the players, such as time played and maximum scores achieved. As a concrete example, if a player takes a long time to reach an average score, the game can be altered and made simpler, or difficulty levels can be added that users can choose at the beginning of the game.

References

[1] S. Sagiroglu and D. Sinanc, "Big data: A review," *2013 International Conference on Collaboration Technologies and Systems (CTS)*, San Diego, CA, USA, 2013, pp. 42-47.

[2] N. Salim, "What's All the Buzz Around "Big Data?": Meet Dr. Aditya Vailaya [The Good, the Bad, and the Ugly: Engineering FACTS]," *IEEE Women in Engineering Magazine*, vol. 6, no. 2, Dec 2012, pp. 24-31.

[3] D. C. Kreines, "Oracle SQL: the essential reference," *"O'Reilly Media, Inc."*, 2000.

[4] C. Strauch, U. L. Sites and W. Kriha, "NoSQL databases," *Lecture Notes, Stuttgart Media University*, vol. 20, no. 24, Feb 2011.

[5] B. Feltrin, "SQL versus NOSQL performance in Big Data analytics," *Master's thesis*, University of Zagreb, Faculty of Economics, Zagreb, 2022.

[6] MongoDB, "When to Use a NoSQL Database," mongodb.com. https://www.mongodb.com/resources/basics/databases/nosql-explained/when-to-use-nosql (accessed Dec 2024).

[7] G. C. Deepak, "A critical comparison of NOSQL databases in the context of ACID and BASE," *Culminating Projects in Information Assurance*, St Cloud State University, 2016.

[8] V. N. Gudivada, D. Rao and V. V. Raghavan, "NoSQL Systems for Big Data Management," *2014 IEEE World Congress on Services*, Anchorage, AK, USA, 2014, pp. 190-197.

[9] J. Bhogal and I. Choksi, "Handling Big Data Using NoSQL," *2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops*, Gwangju, Korea (South), 2015, pp. 393-398.

[10] R. Baltezarevic, B. Baltezarevic and V. Baltezarevic, "The video gaming industry: From play to revenue," *International Review*, no. 3-4, 2018, pp. 71-76.

[11] P. Zackariasson and T. L. Wilson, "Paradigm shifts in the video game industry," *Competitiveness Review*, vol. 20, no. 2, Dec 2012, pp. 139-151.

[12] D. Pavlovic, "Video Game Genres: Everything You Need to Know," hp.com. https://www.hp.com/us-en/shop/tech-takes/video-game-genres (accessed Dec 2024).

[13] The Epic Team, "Postmortem of Service Outage at 3.4M CCU," fortnite.com. https://www.fortnite.com/news/postmortem-of-service-outage-at-3-4m-ccu (accessed Dec 2024).

[14] Riot Games, "Globalizing Player Accounts," technology.riotgames.com. https://technology.riotgames.com/news/

globalizing-player-accounts (accessed Dec 2024).
[15] AWS for Games Content Team and N. Walsh, "The Unique Architecture behind Amazon Games' Seamless MMO New World," aws.amazon.com. https://aws.amazon.com/blogs/gametech/the-unique-architecture-behind-amazon-games-seamless-mmo-new-world/ (accessed Dec 2024).
[16] Blizzard Entertainment, "Relive Your Overwatch Glory!," overwatch.blizzard.com. https://overwatch.blizzard.com/en-us/news/24104603/relive-your-overwatch-glory/ (accessed Dec 2024).
[17] T. Mucci and C. Stryker, "What is big data analytics?," ibm.com. https://www.ibm.com/think/topics/big-data-analytics (accessed Dec 2024).
[18] F. Hagverdiyev, "Data Governance in Gaming Industry," *Problems of Information Technology*, vol. 15, no. 1, 2024, pp. 44-51.
[19] C. Watters, "Shaping the Future of Games: How Machine Learning and Data Science Will Help Transform the Industry," news.ubisoft.com. https://news.ubisoft.com/en-us/article/5eZ3g9FHSlIkSIkp12lyVL/shaping-the-future-of-games-how-machine-learning-and-data-science-will-help-transform-the-industry (accessed Dec 2024).
[20] V. P. Barros and P. Notargiacomo, "Big data analytics in cloud gaming: Players' patterns recognition using artificial neural networks," *2016 IEEE International Conference on Big Data (Big Data)*, Washington, DC, USA, 2016, pp. 1680-1689.
[21] Cloudflare, "What is a large language model (LLM)?," cloudflare.com. https://www.cloudflare.com/learning/ai/what-is-large-language-model/ (accessed Dec 2024).
[22] V. Sharma and M. Dave, "SQL and NoSQL Databases," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 2, no. 8, August 2012.

Cristiana COSTAN graduated from the Faculty of Cybernetics, Statistics and Economic Informatics of the Academy of Economic Studies in 2024, obtaining a Bachelor's Degree in Economic Informatics. She is currently pursuing a master's degree in Databases - Business Support and will earn her degree in 2026. Her main areas of interest include data analysis, database management, and web development.

Evaluating Deep Learning and Machine Learning Models in Federated Learning for Credit Card Fraud Detection. A comparative study

Șener ALI
Bucharest University of Economic Studies
Faculty of Cybernetics, Statistics and Economic Informatics
Bucharest, Romania
alisener20@stud.ase.ro

This study aims to evaluate the effectiveness of both machine learning and deep learning algorithms for credit card fraud detection in the context of a federated learning framework. The fast evolution of digital banking created better experiences for customers and facilitated their access to financial services, but it has simultaneously opened new pathways for cybercriminals, making real-time fraud detection essential. Both models used in this study, XGBoost and a neural network, were trained on a publicly available dataset containing highly imbalanced data, reflecting realistic fraud scenarios. Results demonstrate that both models achieved high accuracy, yet the neural network consistently outperformed XGBoost across critical metrics such as precision, recall, and F1 score. This indicates a superior ability of deep learning models to detect fraudulent transactions in federated learning environments, highlighting their potential to improve financial security through collaborative yet privacy-preserving approaches.

Keywords: *Federated Learning, Credit Card Fraud Detection, Privacy Protection, Machine Learning, Deep Learning*

1 Introduction

In the last years, digital banking has evolved significantly, but this progress has also given rise to more advanced fraud techniques. Cybercriminals can use advanced methods, such as stealing card information used in online transactions, to commit financial crimes. Besides the traditional ways, such as card theft, criminals have at this moment online methods to commit fraud [1].

Based on the central bank's reports, billions are lost every year due to fraudulent activities. Besides the financial losses of the clients, these frauds can have a negative impact on the trust of people in financial institutions. Undermining the credibility of the banking system can create crashes on financial markets and threaten the global economy [2].

To keep a high level of trust in financial institutions, we can employ two mechanisms: fraud prevention and fraud detection. In this study, we will focus on fraud detection, the mechanism that tries to detect fraudulent transactions in real time, through the use of federated learning (FL).

Introduced by Google in 2017 to improve their keyboard text prediction, federated learning became an approach used for training AI models without the need of sharing the data across institutions [3]. Unlike the traditional approaches where aggregating the data in a centralized dataset was necessary, the federated learning offers a decentralized approach. Each institution trains a model locally on its dataset and sends it to the server, where it is aggregated to the global model. This approach enables us to obtain a model trained on all the available data while the data remains confidential [4].

The key point in developing efficient AI models is to have diverse and high-quality data. The obligation to keep the data confidential presents a significant challenge for financial institutions because they can not share their data to create a centralized dataset and train an AI model on it [5]. Moreover, based on the dataset of

an institution, the AI model is trained to detect only the frauds that the bank has already experienced, but it can remain susceptible to the attacks that other banks encountered. This makes the entire financial system more vulnerable, allowing cybercriminals to repeatedly exploit the same type of vulnerability across multiple financial institutions [6].

To address the challenges of the centralized approach, FL solves these limitations by offering the financial institutions the possibility to train a global model without sharing their data. Each bank trains a local model on its own dataset. After training, the model updates are sent to a central server and aggregated to the global model [7]. Adopting this approach, the financial institutions solve both problems, they train a model on all their available data while keeping them confidential. By leveraging knowledge from multiple institutions, FL strengthens the entire financial system by protecting the companies even from fraud cases they have not previously experienced [8].

The application of FL in fraud detection has gained significant attention due to the growing sophistication of fraudulent activities. Traditional machine learning (ML) and deep learning (DL) models have been widely used in fraud detection, but their performance in a federated setting remains an area of active research. While deep learning models, such as convolutional neural networks and recurrent neural networks, can capture complex fraud patterns, traditional ML models, such as decision trees and support vector machines, may offer advantages in interpretability and computational efficiency [9].

Given the increasing adoption of FL and the diverse range of ML and DL models, it is crucial to evaluate their effectiveness in a federated setting. This study aims to investigate the comparative performance of these models in detecting credit card fraud. Specifically, the research seeks to answer to the following question:

RQ. Which performs better in a Federated Learning framework for credit card fraud detection: a machine learning model or a deep learning model?

By addressing this research question, this study aims to provide a comprehensive evaluation of ML and DL models in FL for fraud detection, offering insights into their real-world applicability and potential for improving financial security.

The rest of the paper is organized as follows. In Section 2, related work is discussed. Section 3 provides an analysis of the dataset. Section 4 gives the details of the federated learning fraud detection implementation. The results of the study and their interpretation are presented in section 5. Conclusions are discussed in section 6.

2 Literature review

The importance of credit card fraud detection is represented by the number of public available works. There are numerous articles that talk about this topic or researches that are trying to find the best method to stop fraudsters.

Given the new context of the rise of online financial transactions and the improvements of AI in the last years, the machine learning algorithms have gotten more attention to become a relevant solution for fraud detection [10]. One of the algorithms used for handling the complexity of credit card problems is XGBoost [11]. Being one of the gradient boosting frameworks, the algorithm has a focus on regularization and adopts an iterative approach, improving its accuracy over time.

Comparing XGBoost with other machine learning algorithms when it comes to detecting credit card fraud detection shows us that the XGBoost has better performance, obtaining better results for indicators such as accuracy, precision and recall [12].

Credit card fraud detection has two major challenges: the very limited time span in

which a transaction has to be accepted or rejected and the huge amount of transactions. Only VISA has millions of transactions every day. To address these challenges, the researchers thought that neural networks would be a great solution [13]. As shown in [14], using neural networks can be a great approach to detect credit card fraud.

Besides the two previous challenges we discussed, there is another challenge when it comes to fraud detection. The financial institutions can not share the data, it would be against the data privacy and safety policies [15]. As a result, each financial institution can train models that detect only the patterns of the attacks they experienced, not the patterns of the attacks experienced by other institutions. In this way, the fraudsters can attack each financial institution with the same pattern. What if we could prevent the other financial institutions from experiencing the same type of attack that other banks already experienced? To address this problem, we can implement federated learning.

Federated learning offers the possibility to train models on local devices and share just the model's updates. The main idea of federated learning is to train a global model on all the available data without sharing it. The local trained models are sent to the central server that aggregates the weights. After update, the global model is sent back to the local devices [16]. In this way, the global model's training takes advantage of all available data while keeping it confidential [17].

Given the three problems: the necessity for fast decisions, the need to handle large amount of transactions and the requirement to keep data confidential and the good results obtained in the other studies by both XGBoost and neural networks models, this study aims to compare the performance of these two algorithms in a federated learning architecture.

3 Data

For this study, we used a publicly available dataset that consists of 284.807 transactions [18]. Given the sensitivity of financial data, each transaction has 30 numerical features that have been transformed using Principal Component Analysis (PCA) to protect customers confidentiality.

Additionally, the transactions have two explicit features: Time, representing the seconds elapsed since the first transaction and Amount, denoting the transaction's value in monetary units. The dataset also contains a Class column that identifies whether a transaction is fraudulent (1) or legitimate (0).

As all financial frauds datasets, the dataset we used in this study is highly unbalanced. The fraudulent transactions make up only 0.1727% of the total data, with 492 fraud cases out of 284,807 transactions. To illustrate this imbalance, we created a 2D scatter plot where each point represents a transaction, colored by its class label. Legitimate transactions (blue) form the majority of the data points, while fraudulent transactions (red) are less frequent, reflecting the small proportion of fraud cases.

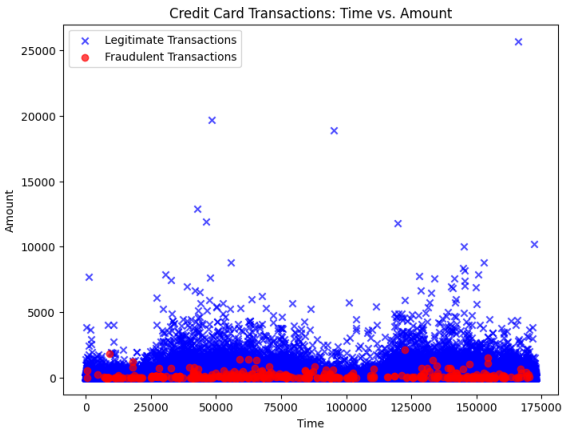


Fig. 1. Dataset visualization

For conducting this study, we needed to split the original dataset into five different smaller datasets. These subsets did not originate from actual banks, but were created to simulate local datasets. For

training each local model, 80% of the corresponding dataset was used as a training set, while the remaining 20% was reserved for testing. The initial dataset was used to test the global model.

4 Methodology

In this section, we explain the methodology employed in this research. We describe the technical approach, the algorithms implemented to address the research problem. Specifically, the methodology revolves around the implementation of XGBoost and neural networks models within the Federated Learning framework.

A. Federated Learning architecture

Fig. 2 illustrates the FL architecture used in this research. Each client node operates independently with its own dataset, representing a financial institution (e.g., a bank). Once a local model is trained on a bank's dataset, the learned parameters (weights) are transmitted to a central server.

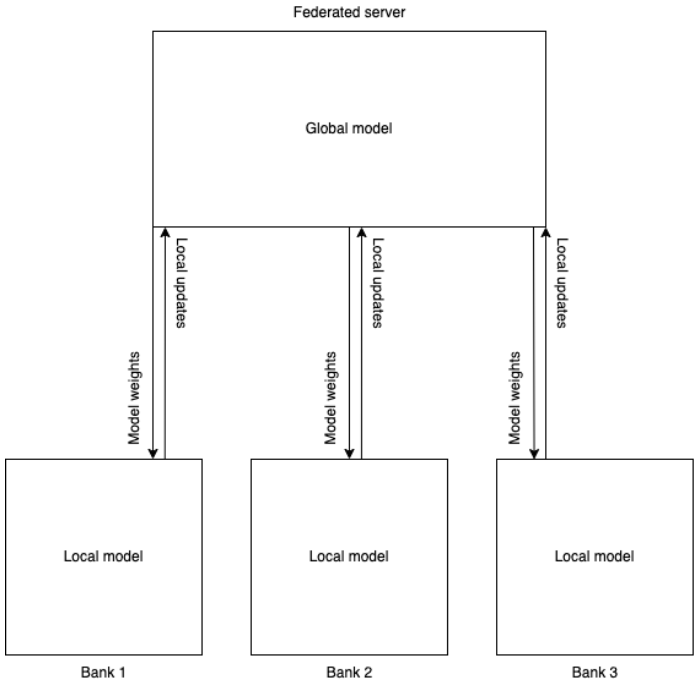


Fig. 2. Federated Learning Architecture

At the central server, the local updates are aggregated to improve the global model. The aggregation process involves computing an average of the local weights, where each bank's contribution is scaled by the number of samples it used during training. The aggregation is expressed by the formula [5]:

$$\omega_t = \sum_{k=1}^{K} \frac{n_k}{n} \omega_{t+1}^k \quad (1)$$

This approach ensures that the global model benefits from diverse data sources while preserving the privacy of each client's data. Before any updates are received, the server initializes the global model either with random parameters or, as in our approach, by pretraining on an available dataset. Once the initial global model is distributed to the clients, each institution trains locally and sends back only its updated model parameters.

On the client side, each institution is responsible to train a local model on its own dataset and send the model weights to the server. This process is important because only the weights are shared, the data remains private.

B. XGBoost Model

The machine learning model chosen for this study was XGBoost. Each financial institution trains locally a model using the

XGBoost classifier, a gradient boosting framework effective with tabular data and robust in handling class imbalance. To ensure the optimal selection of hyperparameters, we integrated GridSearchCV. The grid included variations in tree depth, learning rates and number of estimators. In this way, different combinations of these parameters were evaluated based on their performance, measured by F1-score. This exhaustive search ensured that the selected hyperparameters were the best suited for handling the class imbalance challenge common in financial datasets.

C. Neural Network Model

Our research employs a neural network model, a model well-suited for handling the imbalanced nature of financial frauds datasets.

The network consists of multiple layers:

a. **Input Layer**: The input layer has two major responsibilities: to receive the data and to properly format its features, ensuring that the next layers receive standardized inputs. The job of the input layer is critical because it builds the foundation for accurate pattern recognition throughout the network.

b. **Hidden Layers**: The hidden layers are the core of the network. The learning happens here. Our neural network implements three hidden layers, each with a progressively reduced number of neurons, from 128 neurons in the first layer down to 32 neurons in the final layer. Each hidden layer is followed by batch normalization, which improves the network's training stability, and ReLU, for improving the catch of complex patterns in the data. Moreover, we apply dropout regularization at a rate of 50% after each hidden layer to avoid the risk of overfitting.

c. **Output Layer**: The output layer is responsible for preparing the data for getting out of the network. The output layer's job is to synthetize the information learned by the hidden layers into a final decision. The final input produces a logit, the raw output value, corresponding to the log-odds of the positive class. To get a probability score ranging between 0 and 1, the logit is passed through a sigmoid function. The probability score decides if the transaction is classified as fraudulent or legitimate.

To handle one of the biggest challenges of financial fraud detection, the highly imbalanced nature of datasets, we integrated a focal loss function. By doing this, the model can concentrate more on the minority class, the fraudulent cases, rather than being overly influenced by the majority class. In this way, the overall performance of the model is improved by learning more effectively from the minority class. Mathematically, the focal loss function is expressed as [19]:

$$FL(p_t) = -\propto_t (1 - p_t)^{\gamma} \log(p_t) \quad (2)$$

5 Results and discussions

This section presents the results of our study. The discussion about each model's performance revolves around five key metrics: accuracy, precision, recall, F1 score and AUC-PR [20]. Moreover, for a better evaluation, we interpreted the confusion matrices.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (3)$$

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

$$F1\ score = \frac{2*Precision*Recall}{Precision+Recall} \quad (6)$$

Method	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	AUC-PR (%)
XGBoost	99.91	73.88	73.58	73.73	71.88
Neural Networks	99.94	85.13	80.28	82.64	76.08

Fig. 3. Results

Fig. 3 presents the results obtained by each model in the federated learning framework. The results and its interpretations for the XGBoost and neural network models are discussed individually below.

A. XGBoost model

The high accuracy of 0.9991 achieved by the XGBoost model indicates that almost all predictions were correct. The percent of positive predictions is around 74% indicated by the precision of 0.7388. This also shows us that the percent of false positives is pretty high, around 26%. The recall value of 0.7358 obtained by this study shows us that 74% of the true positive instances were detected, while 26% of them went undetected. The F1 score of 0.7373 shows us the trade-off between precision and recall, highlighting both the model's strength in overall prediction accuracy and its limitations in precisely capturing all relevant cases. The model obtained an AUC-PR score of 0.7188. This suggests that it has a moderate balance between precision and recall across different thresholds.

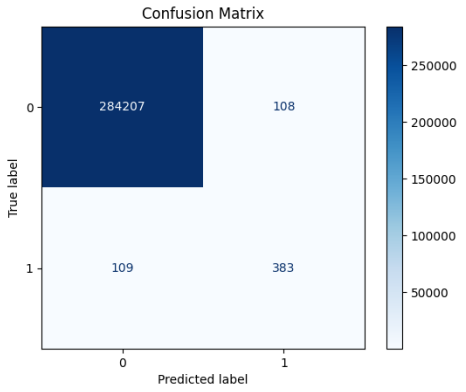


Fig. 4. XGBoost - confusion matrix

The confusion matrix presented in Fig. 4 illustrates the number of instances classified by the XGBoost model as true negatives, false negatives, true positives and false positives. The model performed well in detecting non fraudulent transactions, with 284.207 instances classified as true negatives and only 108 false negatives. For positive cases, the model demonstrated worse performance, with 383 true positives, but 109 false positives. This indicates that around 22% of fraudulent transactions were not correctly identified by the model.

B. Neural Network Model

As shown in Fig. 3, the Neural Network model's accuracy has a value of 0.9994, meaning that nearly all predictions were correct. The precision of 0.8513 indicates that approximately 85% of positive predictions were correct, while the the rest of 15% were false positives. Similarly, the recall of 0.8028 means that the model correctly identified about 80% of the true positive instances, indicating that a portion of actual positives went undetected. The F1 score of 0.8264, which balances precision and recall, illustrates this trade-off, underscoring the model's effectiveness in overall prediction while also highlighting its limitations in accurately identifying all relevant instances. The AUC-PR of 0.7608 highlights that the Neural Network model achieves a stronger level of precision across various recall thresholds than the XGBoost model.

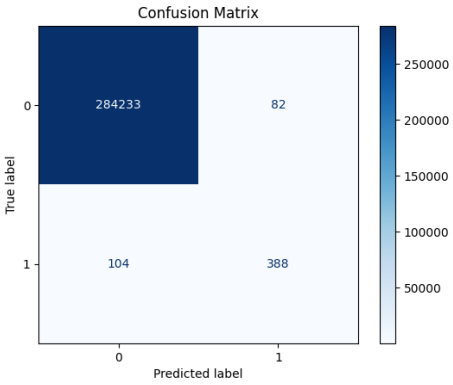


Fig. 5. Neural Network - confusion matrix

The confusion matrix represented by Fig. 5 shows the results obtained by the neural

network model. This matrix shows 284.233 correctly classified negative cases and 388 correctly identified positive cases. Compared to the XGBoost model, the false positives have decreased to 82, suggesting the model is more precise in avoiding incorrect positive classifications. Similarly, false negatives have reduced slightly to 104, enhancing the model's sensitivity by detecting a higher proportion of actual positives.

The results of this comparative study can serve as a starting point for developing a real-world fraud detection system based on the federated learning framework. Given its better performance, the neural network can be the primary candidate model for building such a system. The main limitation encountered during this study was the lack of sufficient data, as the confidentiality requirements restrict the availability of public financial datasets. Having multiple distinct datasets, rather than dividing one into several subsets, would likely have led to improved results. Future research can focus on extending the federated learning approach to other types of financial fraud, including insurance fraud, loan fraud or money laundering. Such studies would validate the applicability and effectiveness of federated learning framework across a broader spectrum of financial crime detection scenarios.

6. Conclusions

In this study we evaluated the performance of both XGBoost and neural network models within a federated learning model. In terms of accuracy, both models performed exceptionally. In key performance metrics such as precision, accuracy, F1 score, and AUC-PR score the neural network model outperformed the XGBoost model. While in previous studies, XGBoost had the best performance when compared to other machine learning models, the deep learning model demonstrated a better ability in detecting fraudulent transactions, making it a promising choice for real-world cases. Overall, this study highlights the potential of federated learning to improve collaboration among financial institutions to develop better fraud detection systems.

References

[1] K. S. Amit, "An Overview of Digital Payment Frauds: Causes, Consequences, and Countermeasures," *Journal of Informatics Education and Research*, vol. 5, no. 1, pp. 2297-2311, 2025.

[2] Yukun, "The Impact of Financial Fraud on Economic Stability: An Extensive Economic Analysis," *Journal of Internet Banking and Commerce*, vol. 28, no. 4, 2023.

[3] Y. Wensi, Z. Yuhang, Y. Kejiang, L. Li and X. Cheng-Zhong, "FFD: A Federated Learning Based Method for Credit Card Fraud Detection," *Big Data – BigData 2019*, vol. 11514, pp. 21-31, 2019.

[4] M. M. Priyanka, "Federated Learning: Opportunities and Challenges," in *Proceedings of ACM Conference (Conference'17)*, New York, 2021.

[5] M. Brendan, M. Eider, R. Daniel, H. Seth and A. y. A. Balise, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, Florida, 2017.

[6] M. Prakash, "Federated Learning for Cross-Bank Fraud Defense," *Intenational Journal of Computer Engineering & Technology*, vol. 16, no. 2, pp. 221-241, 2025.

[7] G. Poonam, H. Jayesh and K. Prakash, "Federated Learning in Banking: A Secure and Intelligent Approach to Financial Fraud Detection," *Journal of Emerging Technologies and Innovative Research*, vol. 12, no. 6, pp. 629-636, 2025.

[8] T. Nguyen, S. Kai, W. Siyao, G. Florian and G. Yike, "Privacy

preservation in federated learning: An insightful survey from the GDPR perspective," *Computers & Security,* vol. 110, pp. 1-23, 2021.

[9] Z. Han, "Federated Learning-Based Credit Card Fraud Detection: A Comparative Analysis of Advanced Machine Learning Models," *ITM Web of Conferences,* vol. 70, pp. 1-6, 2025.

[10] A.Abdulalem, A. R. Shukor, H. O. Siti, A. E. E. Taiseer, A.-D. Arafat, N. Maged, E. Tusneem, E. Hashim and S. Abdu, "Financial Fraud Detection Based on Machine Learning: A Systematic Literature Review," *Machine Learning for Cybersecurity Threats, Challenges, and Opportunities II* , pp. 1-24, 26 September 2022.

[11] R. N. Teuku, M. I. Ghalieb, M. Aga, H. Irsan, S. R. Edi and I. Rinaldi, "Credit Card Fraud Detection for Contemporary Financial Management Using XGBoost-Driven Machine Learning and Data Augmentation Techniques," *Indatu Journal of Management,* vol. 1, no. 1, pp. 29-35, 2023.

[12] K. M. Krishna, Z. K. Mohammad and I. Ajay, "Credit Card Fraud Prediction Using XGBoost: An Ensemble Learning Approach," *International Journal of Information Retrieval Research (IJIRR),* vol. 12, no. 2, pp. 1-17, 2022.

[13] P. Raghavendra and S. Lokesh, "Credit Card Fraud Detection Using Neural Network," *International Journal of Students' Research in Technology & Management,* vol. 2, no. 2, pp. 84-88, 2015.

[14] R. Asha and K. K. Suresh, "Credit card fraud detection using artificial neural network," *Global Transitions Proceedings,* vol. 2, no. 1, pp. 35-41, 2021.

[15] K. A. Saif, J. A. Saif, D. Hussain and A. K. Muhammad, "Secure and Transparent Banking: Explainable AI-Driven Federated Learning Model for Financial Fraud Detection," *Journal of Risk and Financial Management,* vol. 18, no. 4, pp. 1-26, 2025.

[16] Y. Qiang, L. Yang, C. Tianjian and T. Yongxin, "Federated Machine Learning: Concept and Applications," *ACM Transactions on Intelligent Systems and Technology* , vol. 10, no. 2, pp. 1-19, 2019.

[17] Tomisin, M. S. Raj and P. Bernardi, "Transparency and Privacy: The Role of Explainable AI and Federated Learning in Financial Fraud Detection," *IEEE Access,* vol. 12, pp. 64551-64560, 2024.

[18] M. L. G. -. ULB, "Credit Card Fraud Detection Dataset," Kaggle, 2018. [Online]. Available: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud. [Accessed 4 February 2025].

[19] L. Tsung-Yi, G. Priya, G. Ross, H. Kaiming and D. Piotr, "Focal Loss for Dense Object Detection," *IEEE International Conference on Computer Vision (ICCV)* , pp. 2999-3007, 2017.

[20] N. M.Z and A. Amir, "Insights into Performance Fitness and Error Metrics for Machine Learning," *Architecture, Structures and Construction,* pp. 1-19, 2020.

[21] M. Samaneh, B. Ali, S. Sima and F. Francesco, "Balancing privacy and performance in federated learning: A systematic literature review on methods and metrics," *Journal of Parallel and Distributed Computing,* vol. 192, 2024.

Șener Ali is a master's student at the Academy of Economic Studies in Bucharest, where he is expected to graduate in 2025. He currently works as a web developer, and his main areas of interest include web development, artificial intelligence, and blockchain.

The Advantage of NoSQL Databases over SQL Databases

Mario-Tudor CHIRIAC
Bucharest University of Economic Studies
Faculty of Cybernetics, Statistics and Economic Informatics
Bucharest, Romania
chiriacmario21@stud.ase.ro

The aim of this article is to show the advantages of No-SQL databases compared to SQL (relational) databases. Traditionally, the concept of database is implicitly associated with relational databases, most often ignoring the huge potential that non-relational databases have in the fields such as BigData, Data Analysis, Artificial Intelligence. Besides, non-relational databases are also remarkable for their structure flexibility, their speed and for the distribution of their data on multiple servers simultaneously, for their fast writing and reading speed, which is crucial when analysing very large volumes of data.

***Keywords**: Relational databases, non-relational databases, No-SQL databases, SQL databases, SQL, No-SQL, redundancy, efficiency, scalability, portability, large volume of data, big data, management of large volumes of data, big data flow.*

1 Introduction

Before the invention of computers and electronic devices, information would be written on paper, stone, or tree bark. These methods are quite inefficient compared to the electronic storage, as they involve a high human presence in the process of writing and managing the information. Besides, the risks associated with these methods are high: damage to materials (paper, stone, wood) leading to information losses. It is almost impossible to recover the information unless there are backup copies. Duplicating these sources involves human resources and the process is very time-consuming.

An important step in the field of information storage was made with the invention of the magnetic tape in the 1960s. Such a magnetic tape would be read sequentially, which meant that in order to find the desired information, the entire magnetic tape had to be read. Compared to today's standards, this is outdated and inefficient.

The emergence of the Oracle corporation in 1977 facilitated the development of the Oracle Database relational database, which is this corporation's most well-known and highly appreciated product. At the end of the ninth decade of the last century (more precisely, in 1998), the notion of a "No-SQL" database was also postulated, a term coined by Carlo Strozzi [1]. Initially, this term designated those relational databases that would not use SQL as a data manipulation language. Later, with the emergence of the Cassandra and Voldemort databases in the early 2010s, this term became an umbrella name for all non-relational databases.

The invention of the personal electronic computer – Altair 8800 in the 1970s, would mark yet another important stage in the field of technology development and information management, while also being an important moment in the history of technology. This was an essential step, because it facilitated the emergence of several computer models in the following years, such as the Apple II and the IBM PC (Model 5150). The year 1995 marked the launch of the famous Windows 95 operating system, which became a successful brand of the Microsoft corporation. The Windows operating system has become one of the most important operating systems in the whole world, being used everywhere, from

personal computers to servers within production processes.

The exponential evolution of technology, its availability at the highest level and the fast digitalization have enhanced the population's perception of how information is used and stored, something that can be seen today.

Below, I will analyse some fundamental concepts that can be found in the above-mentioned databases, listing some of the studies that made a comparative analysis of these aspects.

The practical part gives the opportunity to understand some key concepts, useful in the presentation of the notions to be discussed, such as: scalability, performance, diversity of query languages.

2. Literature review

According to the specialised literature, we could say that No-SQL databases have several advantages over classic relational databases. No-SQL databases are designed with high scalability, which is essential nowadays, considering the high Internet access rate, as it is necessary for fast data processing. Instead of needing new servers, NoSQL databases can be scaled using the existing hardware. Thus, they can handle traffic growth and respond to demand without activity interruptions. Through this scaling, NoSQL databases become more robust and more performant, thus becoming a preferred choice for expanding data volumes. [2], [15]

In terms of performance, No-SQL databases work optimally even on devices equipped with a common hardware configuration, while relational databases require complex configurations, being specific to devices that are not readily available to the general public (Oracle Exadata) [3]. Thus, it could be said that non-relational databases are very useful for software developers and smaller businesses that do not have complex hardware resources, as they can fit on modestly-configured devices.

The data manipulation language within relational databases is quite orderly and centralised, which applies to most relational databases (Oracle Database, MySQL, MariaDB, SQLite). The differences in syntax/structure are minimal for general queries, and they occur only in advanced functionalities such as multimedia, geographical facilities, etc. No-SQL databases provide a very high flexibility. There is no well-defined standard for the language syntax, as it varies from one model to another.

One of the most well-known languages is the MongoDB Query Language (MQL) used for MongoDB document-oriented databases [4]. This decentralised approach to query languages provides several advantages for both developers and programmers. One advantage is a better cooperation between programmers and developers, thus, a continuous flow of feedback being ensured, on functionalities, bugs, and potential improvements. Besides No-SQL databases are specifically designed to be able to manage huge amounts of data efficiently, unlike the classic relational models, where their rigorous structure and integrity restrictions make such processing impossible. More detailed information about No-SQL databases, and their use in the field of big data, shall be developed in the future chapters.

According to the article [3], in the US, an SQL developer earns $84,328 per year, in average, while a No-SQL developer earns $72,174 per year. Economically speaking, the SQL developer earns more, but the No-SQL developer has the advantage of flexibility provided by the vast scope of his knowledge. A No-SQL database expert can also work as a data analyst, where the average wages are much higher than those of a simple SQL or No-SQL developer [5].

3 Theoretical concepts related to SQL and No-SQL databases. Introduction to BigData

With the invention of the computer and the Internet, the world has experienced an irreversible and revolutionary transformation of the framework in which the development of our personal lives, economic processes and activities take place. The symbiosis between the Internet and the computer has triggered the acceleration of economic and cultural globalisation. The need for rapid and instant access to information and data is the basis of today's notions and fields of activity, such as: data science, data analysis, data mining, etc.

All these sciences form the basis of the so-called BigData. It is obvious that data has a physical support on which it is stored and, via computer networks, it is transmitted between a computer and several other computers, thus forming a network.

Therefore, a database is formation of one or more interdependent data collections, accompanied by a file describing, globally, the data and the links between them. [6],[16]. Nowadays, the most common and frequently used types of databases are: SQL (Standard Query Language) and No-SQL (Not Only Standard Query Language).

There are several well-established manufacturers, on the relational (SQL) and No-SQL databases, providing database management software services. The role of database management systems is of particular importance, because, such a tool allows us to take care of the administrative sides of a database (data backup and recovery, access rights management, database performance analysis) and its security (data access authorisation and control, data encryption, protection against external attacks). According to the ranking made by the db-engines.com[7] website, the first five positions are held by the following database systems: Oracle, MySQL, Microsoft SQL Server, PostgreSQL and MongoDB.

The difference between the two types of data consists of several criteria, such as: *structural criteria*: redundancy, internal data organisation structure, query language, etc. The second criterion used to differentiate the two types of databases is related to *characteristics*: efficiency, allocated memory, speed of data processing, etc.

Based on these criteria, we can make a comparison between the two types of databases: The SQL model has a rigorous and strict structure, where redundancy is minimal and controlled. The form of organisation within an SQL-type database is represented by tables, where the following relationships can exist: *1:1 – one to one, 1:m – many to many and m:m – many to many*.

All these relationships within a relational database create the database's logical diagram. In other words, the logical diagram represents the theoretical structure of the way in which data is organised and linked together.

Besides, the logical diagram also shows the complexity of a relational database, but there are also other important concepts in the field of relational databases, such as: primary key, foreign key, integrity constraints or form of normalisation.

Unlike the SQL model, the No-SQL model is more permissive from a structural and functional point of view. No-SQL databases are divided into 4 categories: document-oriented databases (MongoDB), column-oriented databases (Apache Cassandra), graph-oriented databases (Neo4j) and key-value pair-oriented databases (Redis). No-SQL databases have a wide range of uses and applications: from the e-commerce field (electronic commerce) to the BigData field, which will be addressed in this chapter. Other domains using these databases include social media, mobile apps, gaming, and streaming services. Examples of corporations using such databases include

Netflix, Uber, and Airbnb.[8]

Netflix uses No-SQL databases to store the huge amounts of customer-generated data (viewing history, customer personal data to generate customer preferences). Uber also uses a non-relational database to store customer and driver data, trip history, trip details, and real-time location. Airbnb also uses No-SQL databases to store and manage data about properties available online, users, and a customer's accommodation history. This helps process very large amounts of data for a more complex and accurate analysis, thus helping improve the services offered on the site.

In order to illustrate more theoretical notions related to a No-SQL database, I will use MongoDB, as it is the most popular database (First place in the No-SQL database category and fifth place globally according to that ranking)[7].

Document-oriented databases have the *document* as their basic form. The document types supported by this database are: JSON (Java Script Object Notation), BSON (Binary Java Script Object Notation), XML (eXtenseble Markup Language), YAML (Yet Another Markup Language). The structure of the documents is flexible, thus allowing a greater customization of the document content. Essentially, this format is an improved version of the No-SQL key-value database model, thanks to the fact that the document has a well-defined form. Redundancy in this case is allowed and even encouraged, since, in a No-SQL database, data is stored on multiple servers simultaneously. One advantage is the fact that it provides greater flexibility in searching and retrieving data, if a server in the network cannot be accessed, due to internal or external issues. This means less disruption to data access, which is essential and necessary for today's businesses. This principle is called *scalability*.

Another important aspect, nowadays, is the speed of data writing and reading. The everyday person wants to have fast access, preferably instant, to as much data as possible simultaneously, for various fields of activity (analysis, processing). This is provided by the lack of a database diagram, the lack of normalization forms, which allows the fast development of applications with No-SQL databases. Within document databases (MongoDB), the concept on of junction does not exist. An equivalent of the junction concept is nesting an action through which a new document is inserted within the reference document. This significantly helps the processing of the document. This principle is called *efficiency*.

The variety of existing No-SQL database types has not achieved a standardisation of languages. In fact, this is an advantage, as it allows programmer communities and software development companies to compete with each other, to generate creative and unique ideas, ideas that can translate into new and innovative solutions, thus bringing progress and innovation in the field of No-SQL databases. Although the field of non-relational databases is quite new, compared to that of relational databases (which started in the late 1960s, more precisely in 1969[9]), this domain has known an impressive development. Today, many companies use a non-relational database as support, as we have seen above. This feature of non-relational databases is called *open-source*.

Non-relational databases are suitable for those who want to discover the universe of database knowledge and implicitly that of data analysis. The theoretical concepts related to No-SQL databases are fewer and much easier to understand. No-SQL database terminologies are easier to grasp, thus allowing for faster understanding and in-depth learning.

Table 1. Comparison of concepts between relational and non-relational databases

	Non-relational (No-SQL)	**Relational (SQL)**
Data storage	In tables populated with data - built based on the relational model	Storage using non-relational formats
Junctions	Yes	No
Data storage	Predefined types are respected	There are no defined forms, so it is semi-structured or unstructured.
Primary key	Yes	Yes, to which ObjectID or DocumentID can be added (MongoDB)
Diagram	Dynamic, adaptable depending on the reference model	Static
Transaction management	ACID Model Atomicity Consistency Isolation Durability	CAP Model Consistency Availability Partition Tolerance
Entity name	Table	Collection

BigData is a very important field, developed over the past years. Nowadays, this field is hugely useful, because the analyses and results provided by processing huge volumes of data help improve business processes, optimize decision-making processes, help with economic and financial forecasts, etc. Thus, the BigData field designates extremely large and complex data sets, which cannot be managed or analysed efficiently using traditional data processing tools, such as spreadsheets.[10]

The increasing access to the Internet, economic growth and the increase of living standards worldwide, have irreversibly shaped the course of humanity. Through all his interactions (social, professional, economic, etc.) the Man becomes a data provider. Between 2012 and 2013, it was estimated, according to the IBM corporation, that 2.6 exabytes of data[11] were being produced daily, which means (2.17×260 bytes). Given this huge amount of data, in order to be able to process it, there are some steps that any analyst must take in order to reach conclusive and relevant results: data collection, data processing, data systematisation and then their analysis.

Data collection involves searching for data in different sources of interest. The resulting data can be structured or unstructured. Data processing is the second stage in the set of operations that the collected data goes through. This operation involves the transformation and processing the data into a standardised and easy-to-process format. The next stage is data systematization. This process involves the removal of duplicates from the data set, the removal of outliers, which, after processing, are lost, they become corrupted, inaccessible, etc. This is an essential stage for maintaining the unity and consistency of the data, so that the result provided is as relevant and appropriate as possible. The last stage, data analysis, involves the application of one or more algorithms. Some of the algorithms used for analysis are: linear regression, logistic regression, k-clustering, k-nearest neighbours.[12] In order see how these algorithms are applied and used, the implementations will be shown place in the next chapter.

Besides, the field of BigData is the basis of AI (Artificial Intelligence). Artificial intelligence is the science aiming to make machines capable of performing tasks that would require intelligence, if they were performed by humans.[13],[17]. This

definition was provided by Marvin Minsky in 1956, during a scientific conference at Dartmouth College. BigData provides the information support on which this concept is developed. This is a field that has developed over the recent years. Four models of artificial intelligence have been discovered so far: Machine Learning (ML) which is based on the principle of autonomous learning. This is done based on the introduced data set, regardless of the field (mathematical, computer science, psychological, economic, etc.). Generative AI is the most advanced model known to date. Its characteristics are: generating images, videos, sounds or code written in different programming languages. These functionalities come as a result of a more focused and deeper learning, but also as a result of integrating more advanced languages (LLM). The other two models are: neural networks and deep learning (DL). Neural networks are reconstructions of the human brain at the algorithm level. Through this concept, an attempt is made to detail and deepen the functions of the human brain. Therefore, the starting point of these notions is represented by theoretical biology concepts. The human brain is made up of multiple neurons. A neuron is a cell specialised in receiving and transmitting information [14], [18]. Structurally, a neuron comprises: dendrites, a nucleus, and an axon. Synapses are the points of contact between two neurons, where the transmission of nerve impulses occurs. The following figure shows the structure of a neuron within the human brain:

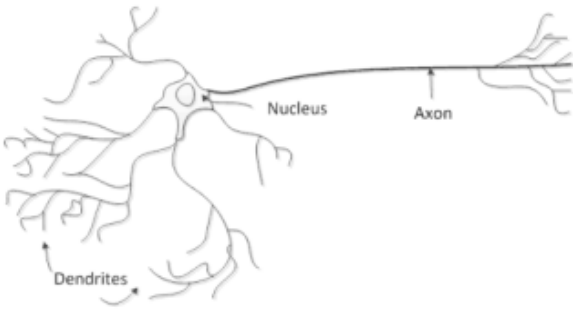


Fig. 1. The structure of a neuron in the human brain

Starting from these concepts, the first abstraction of a neural network has been created: the synapses became *inputs*, the axon became *an output*, and the nucleus became a *summator*.

Thus, the BigData field has been in a permanent and continuous evolution. Most likely, in the future we will witness even more theoretical notions and concepts that are unknown today.

4 NoSQL databases in the fields of Big Data and AI

In this chapter, we will present a case study: building a web application supported by a No-SQL database, provided by MongoDB, which will retrieve data about some customers from an XLSX file, and, data will then be recorded in the database. Based on the analyses made, the web programme will determine the customer's behaviour, thus suggesting offers for future purchases. This application can be useful for the marketing department of an e-commerce store. In order create the user interface, we will use Vue.js, and, in order to create the server connecting the database to the interface, we will use Node.js based on the JavaScript language. The advantage of web applications lays in their flexibility, as they can be run on any type of device (tablet, mobile phone, laptop, desktop, etc.), but also because most of these devices can be connected to the internet. Besides, another advantage of web applications resides in the variety of programming languages or frameworks available on the market to satisfy the needs of each programmer. Among the most famous frameworks we can mention: Vue.js, React.js, ASP.NET, Angular.js, Express.js, etc. The Vue.js framework will allow us to build a modern and user-friendly web interface. Vue.js is based on the HTML, CSS and JavaScript languages, for building objects (buttons, text writing areas, etc.) and for defining their behaviours (performing the operation shown by the button, etc.). CSS has the role of styling the web page with a modern,

pleasant and user-friendly design.

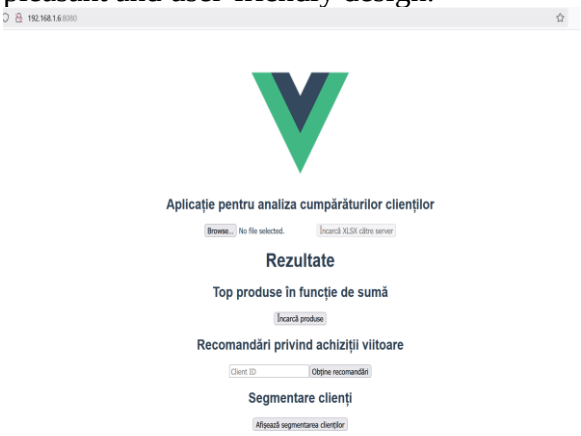


Fig. 2. Example of a user interface made with Vue.js

All of the application's operations are performed by the server (reading the XLSX file, inserting data into the MongoDB collection and the analytical data processing). In order to exemplify one of the analytical operations applied to the data set, such as ranking the most frequently purchased products, we will use, as a basis the following code fragment made in Node.js:

```
app.get("/analytics/top-products", async (req, res) => {
  let client;
  try {
    client = await connectToDb();
    const db = client.db(dbName);
    const collection = db.collection(collectionName);

    const topProducts = await collection
      .aggregate([
        { $group: { _id: "$ProductID", totalSales: { $sum: "$PurchaseAmount" } } },
        { $sort: { totalSales: -1 } },
        { $limit: 5 },
      ])
      .toArray();

    res.json({ topProducts });
  } catch (error) {
    console.error("Eroare la preluarea produselor:", error);
    res.status(500).send({ error: "Eroare de server" });
  } finally {
    if (client) await client.close();
  }
});
```

Fig. 3. Route making a connection to MongoDB, then applying processing operations, using MongoDB aggregations.

This route connects to the database, then to the desired collection. An aggregation is then applied using the .aggregate([…]) operator. Then, using the $group operator, the records are grouped according to the $ProductID field. After that, the amount spent by the buyer during the analysed period is calculated. The $sort operator sorts the data set in descending order, according to totalSales. Subsequently, the data is sent to the frontend, in JSON format, where it will be displayed. The frontend implementation will have two components: a JavaScript part represented by a function that will retrieve the data and a HTML part that will manage the display on the web page. The function retrieving the data from the server is the following:

```
async fetchTopProducts() {
      try {
        const response = await
axios.get("http://localhost:3000/analy
tics/top-products");
        this.topProducts =
response.data.topProducts;
      } catch (error) {
        console.error("Error fetching
top products:", error);
      }
    }
```

The specific HTML code for providing the final result is:

```
  <section>
        <h2>Top produse în funcție de
sumă</h2>
        <button
@click="fetchTopProducts">Încarcă
produse</button>
        <ul v-if="topProducts.length">
          <li v-for="product in
topProducts" :key="product._id">
            ID: {{ product._id }} -
Total vânzări: {{ product.totalSales
}}
          </li>
        </ul>
    </section>
```

In the result retrieval part, we used the Axios dependency. This dependency is useful when we want to write, delete, overwrite or obtain a data set. In this case, we want to retrieve the data using the axios.get() method. Then, we handle the possible errors that may occur during data retrieval. In the area where we display the result, by clicking on the *Incarca produse (Load products)* button, we will be able to see the results, because the button is marked with the @click="fetchTopProducts" event which

will refer to the JavaScript function. After that, a list of the elements obtained from the data retrieval, will be drawn up. The result will have the following form

Fig. 4. Example of result within the web application

The *k-nearest-neighbours* algorithm is one of the most well-known algorithms in the field of BigData processing and in the field of artificial intelligence. This algorithm is used in classification and regression problems. It is considered one of the easiest algorithms in the field of *machine learning*. The implementation stages of this algorithm are: choosing the number of k neighbours closest to a reference point. Then, the distance between any two points in the plane is calculated until the entire range of points is covered. The most commonly used formula to calculate the distance is the Euclidean distance. There are other types of distances as well: Manhattan distance, Minkowski distance, Hamming distance: which is used for categorical data. After calculating the distance, the shortest Ks to the reference point are chosen.

Thus, we will implement this algorithm to highlight the following scenario: generating the potential preferences of a customer in the store based on their previous purchases. Let's assume we choose k=5, that is, the number of suggestions for the customer. Therefore, we will build the function calculating the distance, with the formula for the Euclidean distance.

```
const calculateDistance = (vectorA,
vectorB) => {
  return
Math.sqrt(vectorA.reduce((sum, val, i)
=> sum + Math.pow(val - vectorB[i],
2), 0));
};
```

Now, we implement the route within the server. Due to the fact that the code is very large, I will present it in several different figures, as follows:

```
app.get("/analytics/knn-recommendations/:clientId", async (req, res) => {
  const clientId = parseInt(req.params.clientId);
  const k = parseInt(req.query.k) || 5; // Setăm implicit numărul de sugestii implicit
la 5
  let client;

  try {
    client = await connectToDb();
    const db = client.db(dbName);
    const collection = db.collection(collectionName);

    // Preluăm toți clienții cu cumpărăturile lor

    const clients = await collection.find().toArray();
    const targetClient = clients.find((c) => c.ClientID === clientId);

    if (!targetClient) {
      return res.status(404).send({ message: "Clientul nu s-a găsit." });
    }

    // Construim vectorul caracteristic unde vom salva sugestiile

    const featureVectors = clients.map((c) => ({
      clientId: c.ClientID,
      vector: [
        c.PurchaseAmount || 0,        ...(c.ProductCategory ?
[c.ProductCategory.length] : [0]),      ],
    }));

    const targetVector = [
      targetClient.PurchaseAmount || 0,
      ...(targetClient.ProductCategory ? [targetClient.ProductCategory.length] : [0]),
    ];
```

Fig. 5. Part I – connecting to the database and building the characteristic vector.

```
    // Calculăm distanțele folosindu-ne de funcția specifică a calculului distanței

    const distances = featureVectors
      .filter((fv) => fv.clientId !== clientId)
      .map((fv) => ({
        clientId: fv.clientId,
        distance: calculateDistance(targetVector, fv.vector),
      }));

    // Găsim cei mai apropiați K vecini

    const nearestNeighbors = distances
      .sort((a, b) => a.distance - b.distance)
      .slice(0, k);

    // Preluăm datele celorlați clienți

    const recommendations = [];
    for (const neighbor of nearestNeighbors) {
      const neighborPurchases = await collection.find({ ClientID: neighbor.clientId
}).toArray();
      recommendations.push(...neighborPurchases.map((p) => p.ProductID));
    }

    res.json({ recommendations: [...new Set(recommendations)].slice(0, 5) }); //
Returnăm noile sugestii

  } catch (error) {
    console.error("Eroare:", error);
    res.status(500).send({ error: "Eroare de server" });
  } finally {
    if (client) await client.close();
  }
});
```

Fig. 6. Calculating the distance and selecting the neighbours.

Now, we will see the result for a customer,

having an ID with the value of 1001.

```
{
    "recommendations": [
        "P2001",
        "P5003",
        "P9003",
        "P5007",
        "P10010"
    ]
}
```


Fig. 7. Example of result within the web application using the k nearest neighbours algorithm

Of course, these algorithms can also be used in other fields of study (medicine, education, networking, etc.), or, in our case, for other scenarios: optimising results based on preferences/Internet searches/budget allocated by the consumer. This way, the online advertising provided by merchants will be more relevant to the consumer. In order to have the expected, desired results, we must have as much data as possible, which also has to be relevant. The premises that we want to analyse must be logical and they must have a well-defined purpose, so that the decision-maker can always make the best decision.

5 Conclusions

In conclusion, No-SQL databases have greatly contributed to deepening and and consolidating the BigData field. With their flexible and scalable way of managing data, non-relational databases prove that they can be an alternative to traditional relational databases, which have a more rigid structure and which are not so flexible, when it comes to processing large volumes of data. MongoDB databases also stand out in BigData processing, as they provide a new perspective on the process of storing and analysing data. The use of data collections and nesting within documents allows for easy and fast data management, compared to the traditional relational models, which do not benefit from so much flexibility and contain more theoretical concepts, as seen in the previous chapters. Besides, the use of operators in data processing is an advantage, as they are intuitive and easy to use, being of great help to new programmers in this field. In addition, the study of No-SQL databases is an important step for any programmer, entering a new path of knowledge. Besides, the syntax of the MongoDB operators, is also easy and intuitive. Machine-learning algorithms along with the processing of large volumes of data make up a symbiosis that will develop the study and research in any scientific field, providing as many solutions as possible, to questions that have not yet been answered.

References

[1] https://www.telacad.ro/cursuri/baze-de-date-nosql-redis-mongodb-neo4j/

[2] https://www.oracle.com/ro/database/nosql/what-is-nosql/, accessed on 21.03.2025

[3] https://www.guru99.com/ro/sql-vs-nosql.html, accessed on 21.03.2025

[4] https://ro.simeononsecurity.com/articles/sql-vs-nosql-choosing-the-right-database-management-system/, accessed on 21.03.2025

[5] https://www.unite.ai/ro/ce-este-un-inginer-de-date-foaie-de-parcurs-responsabilit%C4%83%C8%9Bi-salariale/, accessed on 21.03.2025

[6] coord. I. Lungu, Adela Bâră, Constanța Bodea, Iuliana Botha, Vlad Diaconița, Alexandra Florea și Anda Velicanu, *Tratat de baze de date vol. I: Baze de date, organizare, proiectare, implementare*, Editura ASE, București, 2011, p. 77

[7] https://db-engines.com/en/ranking, accessed on 12.12.2024

[8] https://www.datastax.com/guides/nosql-use-cases, accessed on 12.12.2024

[9] coord. I. Lungu, Adela Bâră, Constanța Bodea, Iuliana Botha, Vlad Diaconița, Alexandra Florea și Anda Velicanu, *Tratat de baze de date vol. I: Baze de date, organizare, proiectare,*

implementare, Editura ASE, București, 2011, p. 77
[10] https://www.oracle.com/ro/big-data/what-is-big-data/, accessed on 15.12.2024
[11] https://web.archive.org/web/20130824213031/http://www.ibm.com/big-data/us/en/, accessed on 16.12.2024
[12] https://tdwi.org/articles/2018/07/02/adv-all-5-algorithms-for-big-data.aspx, accessed on 17.12.2024
[13] https://www.sap.com/romania/products/artificial-intelligence/what-is-artificial-intelligence.html, accessed on 17.12.2024
[14] https://code-it.ro/introducere-in-retele-neuronale-teorie-si-aplicatii/, accessed on 18.12.2024
[15] https://codegym.cc/ro/quests/lectures/ro.questhibernate.level19.lecture00, accessed on 21.03.2025
[16] https://support.microsoft.com/ro-ro/topic/no%C8%9Biuni-elementare-despre-bazele-de-date-a849ac16-07c7-4a31-9948-3c8c94a7c204, accessed on 21.03.2025
[17] https://www.europarl.europa.eu/topics/ro/article/20200827STO85804/ce-este-inteligenta-artificiala-si-cum-este-utilizata, accessed on 21.03.2025
[18] https://askabiologist.asu.edu/neuronii-%C5%9Fi-nervii, accessed on 21.03.2025

CHIRIAC Mario-Tudor First year student, in the Master's Programme named: "Databases - business support", within the Faculty of Cybernetics, Statistics and Economic Informatics - the Bucharest University of Economic Studies. In 2024, I graduated from the Bachelor's programme - Economic Informatics in Romanian, within the Faculty of Cybernetics, Statistics and Economic Informatics at the Bucharest University of Economic Studies. Areas of interest: study of SQL and No-SQL databases, programming web applications using Vue.js, React.js.

House Market Prediction Using Machine Learning

Nicușor-Andrei ANDREI
Bucharest University of Economic Studies
Faculty of Cybernetics, Statistics and Economic Informatics
Bucharest, Romania
andreinicusor20@stud.ase.ro

This study explores and compares the performance of tree-based machine learning algorithms' predictions for Bucharest real estate market prices. The dataset was obtained from a local platform in March 2025 and contains residential apartments for sale in Bucharest. The comprehensive data preprocessing step, including imputation of missing values, encoding of categorical variables, and the engineering of new key features such as distance to public transport, played a key role in the models' performance. The models were optimized using a grid search algorithm with 5-fold cross-validation and evaluated with key performance indicators including root mean squared error, mean absolute error, and coefficient of determination. The results indicate that XGBoost outperforms both Random Forest and a single Decision Tree, reducing all the key performance indicators used in analysis.
Keywords: *House market, Machine learning, Price prediction, Tree-based machine learning, XGBoost*

1 Introduction

The housing real estate market in Bucharest underwent a drastic transformation over time due to rapid urbanization, demographic changes, and shifting economic conditions. Homebuyers and investors require an effective price forecast based on newer techniques to provide a robust house market analysis. The performance of tree-based machine learning techniques, decision tree, random forest, and XGBoost, on this particular economic segment of Bucharest is presented in this paper. One Decision Tree provides simple interpretability but will probably overfit; Random Forests increase stability by averaging a large number of trees; and XGBoost continues to improve predictions using gradient boosting that adjusts residual errors iteratively. Despite the optimistic results presented by these approaches in different global markets, their comparative performance in the specific context of Bucharest is yet to be explored. The current research fills this gap by benchmarking the performance of Decision Tree, Random Forest, and XGBoost regressors against an elaborate dataset of real estate transactions in Bucharest, collected in March 2025 from a regional online marketplace. After an extensive preprocessing step including imputation of the missing values, outliers handling, and feature engineering, the algorithms are fine-tuned using the grid search method. The grid search is carried out using a 5-fold cross-validation mechanism with the best model determined as the minimizer of the mean squared error metric. The optimised versions of the three algorithms are next benchmarked against the key indicators relevant to regression analysis: root mean squared error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), and the coefficient of determination (R2). Apart from precision, the approach also considers the drivers of the prices of properties, with model interpretability highlighted using the feature importance score.

The structure of the paper is outlined as follows: Section 2 gives an overview of the literature relevant to the topic under investigation; Section 3 gives a comprehensive description of the

methodology used during this research; Section 4 presents the findings together with a related discussion; and Section 5 recapitulates the major conclusions of this research and makes recommendations for future research. This framework ensures an empirically supported comparison of tree-based models in the context of Bucharest's vibrant real estate market.

2 Literature Review

The rapid evolution of machine learning has opened new opportunities for modeling complex systems, including the real estate market, where price dynamics are difficult to capture. In the context of Bucharest's urban housing landscape, machine-learning-based price prediction promises to help buyers, investors, and policymakers navigate a market shaped by diverse offers and fluctuating demand. In the next paragraphs, studies related to this field are synthesized to highlight methodologies, key findings, and their implications for a Bucharest-focused investigation.

2.1 Hedonic Foundations

Before the beginning of modern machine learning, hedonic-price models provided a framework for decomposing sale prices into intrinsic and external attributes [1]. Early applications in Beijing and Paris confirmed the influence of building quality and neighborhood heterogeneity on prices [2, 3], and more recent work has extended hedonic indices to capture the effects of transit accessibility [4]. Although most hedonic studies focus on explanatory power rather than forecasting, they offer valuable guidance on feature selection and baseline model structure, insights that remain relevant when constructing machine-learning pipelines for Bucharest.

2.2 Ensemble and Tree-Based Methods

A recurring theme across multiple markets is the strong performance of tree-based ensemble models. For example, the authors of [5, 6, 7] show that Random Forest Regression achieves high accuracy when forecasting house prices in their case studies. Similarly, in Hong Kong, a Random Forest model outperformed both Support Vector Machines and Gradient Boosting Machines on a dataset of 40,000 transactions spanning 18 years [8]. Extreme Gradient Boosting algorithm (XGBoost) also stands out: a Vilnius study that scraped nearly 19,000 listings found XGBoost to be the most predictive among fifteen tested models [9]. In Bengaluru, XGBoost again outperformed Linear, Ridge, Lasso and SVM baselines [10], and hybrid approaches (for example stacking with CatBoost) further reduced error on 1.9 million transactions in Hong Kong [11]. Taken together, these ensemble successes suggest that machine-learning models for Bucharest will likely benefit from tree-based learners, provided that sufficient feature engineering and hyperparameter tuning are applied.

2.3 Linear and Kernel Methods

Although less dominant than ensemble models, linear and kernel-based regressors remain valuable for their interpretability and computational efficiency. In the Bengaluru comparison, Ridge and Lasso regressions, as well as a support-vector machine, delivered reasonable accuracy but were ultimately outperformed by XGBoost [10]. In Melbourne, an SVM achieved the lowest mean squared error at the expense of the longest training time, underscoring a trade-off between predictive power and computational cost [12]. A Multiple linear regression with the right data split ratio is validated as being more efficient than simple linear regression for house market predictions [13]. In Beijing, after a rigorous data preprocessing step, which included outlier elimination, feature engineering, and one-hot encoding algorithm, Hybrid and Stacked Generalization Regression delivered promising results on the training set and test set [14].

For Bucharest, where data volumes may vary significantly across districts, it may be

worthwhile to benchmark a well-tuned SVM alongside faster linear models as baseline approaches.

2.4 Classification Formulations

Some researchers have reframed house-price forecasting as a binary classification task, predicting whether a property's selling price will exceed its listing price. In Fairfax County, Virginia, the RIPPER rule-learning algorithm yielded fewer classification errors than C4.5, Bayesian classifiers, and AdaBoost [15]. A related study applied SVM, Random Forest, and a neural network, finding that Random Forest delivered the highest accuracy, precision, sensitivity, and specificity [16]. While classification methods do not provide information on the magnitude of price changes, they may prove useful in Bucharest for stakeholders interested primarily in upside versus downside risk.

2.5 Deep and Hybrid Architectures

Deep-learning models, especially when enriched with a good data engineering step, have great potential but also require large datasets and careful regularization to avoid overfitting. One study combined street-view imagery and socio-economic indicators with a gradient boosting machine to forecast neighborhood-scale appreciation rates with high accuracy [17]. Comparative work found that both deep networks and classical regression models tended to overestimate control-sample prices, indicating a need for larger datasets and ensemble hybrids [18]. The most advanced approaches leverage Transformer architectures optimized via Bayesian hyperparameter search, achieving substantial RMSE reductions on large Hong Kong datasets [11]. For Bucharest, integrating aerial or street-view data, if such data are available, could yield similar improvements, although attention must be paid to model complexity and generalization.

2.6 Research Gaps and Outlook

Despite the extensive literature, few studies have applied these advanced machine-learning techniques explicitly to the Bucharest housing market. Key knowledge gaps include the integration of geographic data with house attributes and the adaptation of machine learning models and deep-learning networks to conform to the data structure of Romania. By overcoming these limitations and integrating best practices across the different stages of preprocessing of the data, model development, and model evaluation, future studies can provide accurate and interpretable price forecasts customized to the specific urban context of Bucharest.

3 Methodology

The following chapter, summarized in Figure 1, presents the methodology steps in depth, starting with data acquisition and preprocessing, continuing with the architectures of the proposed models, and closing with the metrics used for comparison.

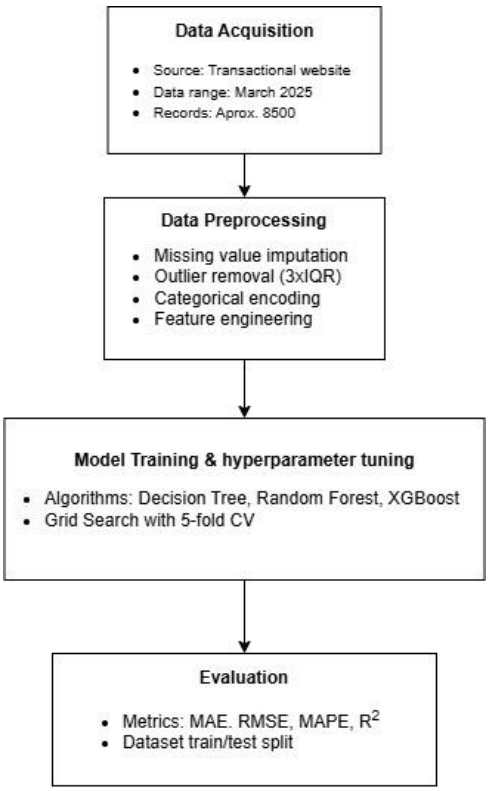


Fig. 1 Methodology applied

3.1 Data Acquisition and Preprocessing

Transactional data for the Romanian residential real estate market was obtained by web scraping a local listings website in March 2025. Initial preprocessing involved the removal of variables presenting over 5% missing values. Discrete features with missing entries were imputed according to domain-specific rules: absent bathroom counts were assigned a value of 1, and missing balcony counts were set to 0. The year built field, with a low percentage of missing values, was median-imputed to maintain discreteness, while missing "built area" values were replaced by the median built area stratified by room count.

Feature engineering produced several binary and continuous variables to enhance model performance. The categorical attributes housing_type, comfort, and compartmentalization_type were encoded as the binary indicators is_penthouse, is_luxury, and is_detached. Vertical position within the building was captured by the flags is_first_floor and is_last_floor, together with floor_ratio, defined as the apartment's floor number divided by the total building floors.

Spatial features were derived from latitude and longitude coordinates by computing distances (in meters) to the city center and the nearest metro station using the Haversine formula [19]:

$$\varphi_i^* = \varphi_i \cdot \frac{\pi}{180},\ \lambda_i^* = \lambda_i \cdot \frac{\pi}{180},\ i = 1,2 \quad (1)$$

$$\Delta\varphi = \varphi_2^* - \varphi_1^*, \qquad \Delta\lambda = \lambda_2^* - \lambda_1^* \quad (2)$$

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos(\varphi_1^*)\cos(\varphi_2^*)\sin^2\left(\frac{\Delta\lambda}{2}\right) \quad (3)$$

$$d = 2R\ \arcsin\left(\sqrt{a}\right), \quad (4)$$

where φ represents latitude, λ represent longitude and R represents the radius of the Earth, which is 6371 km. [20]

Interquartile filtering was applied to mitigate the influence of extreme outliers. For each variable of interest—sale price, usable area, built area, bathroom count, and floor count—the first (Q1) and third (Q3) quartiles were calculated, and the interquartile range (IQR = Q3 − Q1) was determined. Observations falling outside the interval [Q1 − 3·IQR, Q3 + 3·IQR] were removed, resulting in a homogeneous and robust dataset suitable for subsequent inferential and predictive modeling.

3.2 Model Architectures

This study presents a comparative analysis between the Decision Tree, Random Forest, and XGBoost regression algorithms. To minimize the mean squared validation error, the exhaustive grid search approach was employed to select the optimal hyperparameters for each model, with the aid of 5-fold cross-validation. The exhaustive grid search provided the best parameter settings that achieved the lowest average validation MSE for each of the respective algorithms, thus attaining a good trade-off between bias and variance [21].

The Decision Tree algorithm partitions the feature space by using splits that increase variance reduction according to the squared-error criterion. To avoid the over-splitting and overfitting, three complexity parameters were tuned as explained below:

- **max_depth**: (3, 5, 7, 10, 12, 15, 20, 25) - Limits the depth of the tree to balance representation ability with overfitting risk.
- **min_samples_split**: (2, 5, 10, 20) - Specifies the minimum number of samples to be used when making a split for a node.
- min_samples_leaf: (1, 2, 4, 10) - Guarantees terminal nodes to have a reasonable quantity of observations, smoothening predictions in regions of sparsity.

The Random Forest algorithm builds a series of decision trees by bootstrap aggregation (bagging) and random feature selection, thus reducing variance by averaging [22]. Four hyperparameters were explored:

- **n_estimators**: (10, 50, 100, 200, 400, 600, 800) - Controls the size of the forest of trees; larger forests

reduce variance more effectively at increased computational cost.

- **max_depth**: (3, 5, 7, 10, 12) - Max out the depth of each tree to control individual complexity.
- **max_features**: ("sqrt", "log2") - Controls the number of predictors considered at each split, introducing randomness which decorrelates individual trees further.
- **bootstrap**: (True, False) - Switches sampling with or without replacement, evaluating the effect of bootstrap aggregation on ensemble stability.

This extensive search guaranteed the choice of a forest configuration that minimized validation error while ensuring computational efficiency. XGBoost uses gradient tree boosting to train trees sequentially on the residuals of earlier iterations, with a squared-error loss combined with regularization to avoid overfitting. [23] Three of the most important parameters were optimized:

- **learning_rate**: (0.01, 0.1, 0.3) - Scales the contribution of each new tree, with lower values promoting slow learning and improved generalization.
- **max_depth**: (3, 5, 7, 10, 12) - Cuts off the depth of each boosted tree, balancing the ability to model complex patterns against overfitting.
- **n_estimators**: (10, 50, 100, 200, 400, 600, 800) – The total number of trees to be trained, more iterations allowing for more accurate residual correction, but also using more training time.

By applying the grid search technique over these grids, the optimal balance among learning rate, tree complexity, and ensemble size was determined.

3.3 Evaluation Measures

Models' performance was compared on an unseen test set of data against four complementary metrics chosen for their individual interpretive strengths in housing-market forecasting contexts: Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), and the coefficient of determination (R^2).

Root Mean Squared Error was selected since it punishes huge deviations more severely than small ones, aligning with the fact that huge pricing errors (e.g., underpricing a home by tens of thousands euros) are particularly unperformant for real-estate applications. By squaring residuals before averaging, RMSE rewards models that avoid occasional but big mispredictions, so it is a sensitive metric of worst-case performance.

Mean Absolute Error provides a straightforward, linear measure of average prediction error in the same currency units as the target, which is euros. Unlike RMSE, MAE treats all errors equally and is thus more robust to outliers and more interpretable for stakeholders who require an intuitive sense of actual deviation from the actual sale price.

To measure errors concerning scale, the Mean Absolute Percentage Error (MAPE) was applied. MAPE is a measure of the average deviation from the true price, thus allowing effective comparisons between segments with different levels of prices (e.g., comparing central business district penthouses with suburban apartments). The ratio measure is useful in providing decision-makers with the ability to measure the relative accuracy of the models.

Finally, the coefficient of determination quantifies the degree to which observed price variation is accounted for by the model. A value of R^2 close to 1 signifies strong explanatory power for market movements, while values close to zero imply that the predictor set fails to adequately capture systematic pricing patterns. Together, these four measures offer a complete and balanced evaluation of accuracy, robustness, relative error, and

goodness-of-fit, and thus allow for a strict comparison of the Decision Tree, Random Forest, and XGBoost models.

4 Results and Discussions

This chapter presents an in-depth view of both dataset features and exploratory analysis, along with each individual model's performance, and concludes with a comparative view of the trained models.

4.1 Dataset Presentation

The final dataset used in the analysis has 8500 entities, each with the following independent variables based on type:

- Continuous variables:
  - Built_area
  - Usable_surface
  - Lat
  - Lon
  - Metro_distance_in_meters
  - Distance_to_center
  - Floor_ratio
- Integer Variables:
  - Year_built
  - Floor_number
  - Number_of_floors
  - Bedroom_count
  - Bathroom_count
- Dummy variables:
  - Is_penthouse
  - Is_luxury
  - Is_first_floor
  - Is_last_floor
  - Is_detached

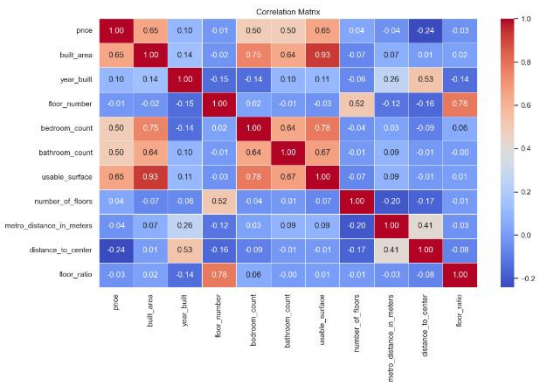


Fig. 2. Correlation Matrix

The correlation table in Figure 2 gives a general overview of the association between predictors and between predictors and the target variable. There is a moderate positive relationship between price and bathroom_count, bedroom_count, and usable_surface, which reflects the major role of building size and layout in influencing the price. In contrast, there is a weak negative relationship between distance_to_center and price, which shows that an increase in distance from the city center has the general tendency to lower the value of the properties. Furthermore, distance_to_center and metro_distance_in_meters have moderate positive relationships with year_built, which shows that newer constructions are most likely to be located far from the city center and metro stations. Finally, the strong relationship between usable_surface, built_area, bedroom_count, and bathroom_count supports the fact that larger living spaces usually have more bedrooms and bathroom counts. These findings reinforce the significant part played by structural and location variables in establishing the value of Bucharest residential properties.

Fig. 3. Price per m2 based on the location of the apartment

The hexbin map in Figure 3 illustrates price per square meter variability across the geography of the city. The higher values cluster together in the north-central suburbs, where there is high access speed to central amenities and a well-connected transport network, increasing demand. Prices fall off gradually from the center,

then fall lowest in the southern, eastern, and western peripheries—suburbs dominated by newly developed areas that haven't yet had time to experience the benefits of a fully settled urban infrastructure.

4.2 Models Overview

This chapter presents the final resulted models by the grid search algorithm along with their parameters and feature importance.

For the Decision Tree Regressor, the optimal model based on the MSE value resulting from the grid search algorithm has the following parameters:

- Max_depth: 10
- Min_samples_leaf: 10
- Min_samples_split: 2

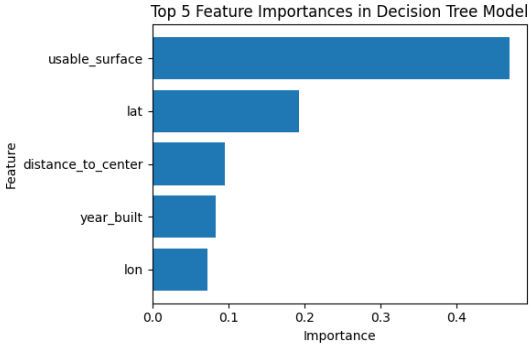


Fig. 4. Top 5 feature importances in Decision Tree Regressor

The barplot presented in Figure 4 presents the top 5 characteristics by importance, which are used by the decision tree model to predict the final price of the residential building. The most important variable used in forecasting is the usable surface, but the geospatial data represented by latitude and longitude, along with the distance from the city center, also play a key role in predictions.

Based on the same search algorithm, the optimised Random Forest algorithm has the following parameters:

- Max_depth: 12
- Max_features: ‘sqrt’
- N_estimators: 200
- Bootstrap: False

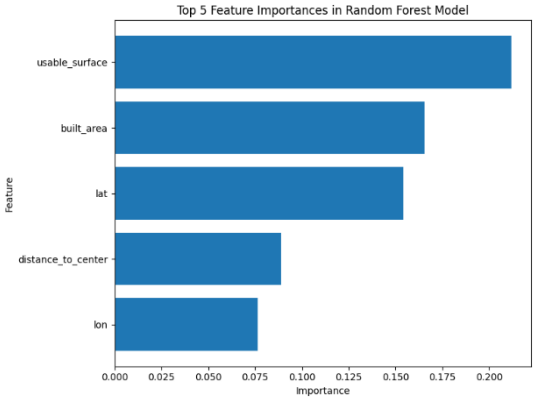


Fig. 5. Top 5 feature importances in Random Forest Regressor

Figure 5 illustrates the top 5 features by importance used by the Random Forest model. The barplot also suggests that distance to city center, geospatial location, and the surface of the apartment are among the most important features used for price prediction. Regarding the Decision tree model, the Random forest model uses the built area of the apartment instead of the year built as one of the top 5 features.

For the XGBoost ensemble model, the optimal parameters after training using the grid search algorithm are:

- Learning_rate: 0.1
- Max_depth: 7
- N_estimators: 800
-

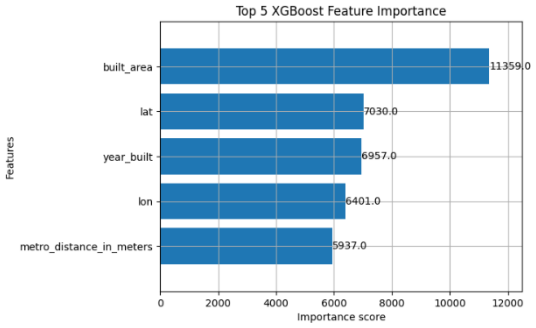


Fig. 6. Top 5 XGBoost feature importance

Figure 6 presents the top 5 features used by the XGBoost model in predictions. In the same manner as the last models, the surface of the residential unit, the location, and the distance to the center are the main factors used in predictions.

4.3 Model Comparison

Table 1. Performance of the models based on key indicators

	MAE	RMSE	MAPE	R2
Decision Tree	20487.65	32589.67	15.86	0.77
XGBoost	15089.36	24296.89	11.88	0.87
Random Forest	17134.86	26573.65	13.7	0.85

This chapter will analyze and compare the performance of the studied models presented in Table 1. The single Decision Tree model, although very interpretable, performs the worst in terms of prediction among the three methods. It has a Mean Absolute Error (MAE) of approximately 20,488 and a Root Mean Squared Error (RMSE) of 32,590 on the test set, meaning its point-predictions tend to be short of actual values by a significant margin. Its MAPE of 15.9 % shows that, on average, predictions are different from actual prices by nearly one-sixth, and its R^2 of 0.77 shows that only 77% of the variability in sale prices is explained. This result illustrates the model's tendency to overfit patterns in the training data and not to smooth out noise between different regions of the feature space.

By comparison, the Random Forest ensemble substantially reduces the bias and variance through the application of bootstrap aggregation. Its MAE is reduced to approximately 17,135 (a decrease of 16 % relative to the Decision Tree), and its RMSE is reduced to 26,574, while the MAPE is reduced to 13.7 %. Such reductions yield an R^2 equal to 0.85, indicating that 85 % of residential prices' variance is accounted for by the model. The improved performance comes from averaging the predictions of 200 decorrelated trees, each with a maximum depth of 12, which stabilizes estimates and lowers over-fitting at the expense of less interpretability.

Finally, the XGBoost model achieves the highest overall accuracy by sequentially correcting the residuals of its predecessor. It gives an MAE of 15,089 (26 % lower than the Decision Tree), an RMSE of 24,297, and a MAPE of just 11.9 %, which corresponds to an R^2 of 0.87. Practically, this means the gradient-boosted ensemble reduces average percentage error by nearly one-quarter compared to the tree baseline and explains 87 % of price variation. Though slightly more complex to train and to tune, XGBoost's ability to learn subtle feature interactions and to penalize overly complex trees makes it the go-to when minimizing prediction error is top priority.

5 Conclusion

In this research, a thorough comparison of tree-based regression algorithms Random Forest, Decision Tree, and XGBoost for predicting apartment prices in residential apartments in Bucharest has been performed. Using a true transactional dataset collected in March 2025, which was subject to detailed preprocessing, missing-value imputation, one-hot encoded, and creating new spatial indicators, each model was optimized using grid search combined with five-fold cross-validation to reduce the mean squared error (MSE) as the goal function. Among the independent variables involved, geospatial variables like proximity to the city center, longitude, latitude, distance to metro station, and unit size measurements (usable area and built area) were the most critical indicators of the price.

The results reveal a clear and noteworthy trend in the predictive accuracy. The single Decision Tree, while illustrating higher interpretability, incurred the largest testing errors, with the mean absolute percentage error being 15.9%. Application of the bootstrap averaging in the Random Forest approach reduced the mean error by 16%,

thereby attesting to the importance of variance reduction via ensemble methods. Finally, the gradient-boosted XGBoost model reported the lowest errors, demonstrating its ability to accurately make incremental corrections and discern complex, non-linear relationships.

Practically, these findings suggest that ensemble methods, particularly XGBoost, are significantly better than single trees when reducing pricing error is the goal. Random Forest is a good compromise if interpretability and computational cost need to be balanced against performance. In the meantime, the Decision Tree remains a good benchmark for rapid prototyping and stakeholder communication because it has interpretable decision rules.

Despite these advancements, some of these limitations remain to be explored in future research. First, our spatial attributes depend on fixed coordinates and do not account for dynamic urban factors such as traffic congestion or planned infrastructure development. Second, temporal dynamics (for instance, seasonality, macroeconomic data) were not explicitly modeled; employing time-series methods or adding lagged market indicators might enhance predictions further. Investigating these directions will contribute to real-time appraisal systems and support data-informed decision-making in Bucharest's transitioning real estate market.

References

[1] S. Rosen, "Hedonic Prices and Implicit Markets: Product Differentiation in Pure Competition," *Journal of Political Economy,* vol. 82, no. 1, 1974.

[2] Z. Yang, "An application of the hedonic price model with uncertain attribute - The case of the People's Republic of China," *Property Management,* vol. 19, no. 1, pp. 50-63, 2001.

[3] R. P. M. &. S. S. Maurer, "Hedonic price indices for the Paris housing market," *Allgemeines Statistisches Archiv,* vol. 88, pp. 303-326, 2004.

[4] Y. C. N. X. R. Z. K. C. S. H. Linchuan Yang, "Place-varying impacts of urban rail transit on property prices in Shenzhen, China: Insights for value capture," *Sustainable Cities and Society,* vol. 58, p. 102140, 2020.

[5] M. a. J. a. Y. S. a. P. R. a. N. P. a. Indervati, "House Price Prediction Using Machine Learning," in *1st International Conference on Advances in Computing, Communication and Networking (ICAC2N),* Greater Noida, India, 2024.

[6] Y. L. M.-H. L. a. S.-Y. H. Jeonghyeon Kim, "A Comparative Study of Machine Learning and Spatial," *Sustainability,* vol. 14, no. 15, p. 9056, 2022.

[7] O. N. A. F. A. A. O. O. Y. F. A. G. O. Abigail Bola Adetunji, "House Price Prediction using Random Forest Machine Learning Technique," *Procedia Computer Science,* vol. 199, pp. 806-813, 2022.

[8] B.-S. T. S. W. W. Winky K.O. Ho, "Predicting property prices with machine learning algorithms," *Journal of Property Research,* vol. 38, no. 1, pp. 48-70, 2021.

[9] P. V. &. S. A. Grybauskas, "Predictive analytics using Big Data for the real estate market during the COVID-19 pandemic," *Journal of Big Data,* vol. 8, p. 105, 2021.

[10] R. G. a. N. S. N. J. Manasa, "Machine Learning based Predicting House Prices using Regression Techniques," in *2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA),* Bangalore, India, 2020.

[11] Y. L. Z. W. M. Z. T. W. C. Choujun Zhan, "A hybrid machine learning framework for forecasting house price," *Expert Systems with Applications,* vol. 233, p. 120981, 2023.

[12] T. D. Phan, "Housing Price Prediction Using Machine Learning Algorithms: The Case of Melbourne City, Australia," in *2018 International Conference on Machine Learning and Data Engineering (iCMLDE)*, Sydney, NSW, Australia, 2018.

[13] A. a. A. H. a. S. R. Rai, "Predicting Housing Sale Prices Using Machine Learning with Various Data Split Ratios," *Data and Metadata*, vol. 3, 2024.

[14] M. N. H. D. B. M. Quang Truong, "Housing Price Prediction via Improved Machine Learning Techniques," *Procedia Computer Science*, vol. 174, pp. 433-442, 2020.

[15] J. K. B. Byeonghwa Park, "Using machine learning algorithms for housing price prediction:," *Expert Systems with Applications*, vol. 42, no. 6, pp. 2928-2934, 2015.

[16] B. a. S. Dutta, "Predicting the housing price direction using machine learning techniques," in *2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)*, Chennai, India, 2017.

[17] F. Z. W. P. S. G. J. R. F. D. C. R. Yuhao Kang, "Understanding house price appreciation using multi-source big geo-data and machine learning," *Land Use Policy*, vol. 111, 2021.

[18] Foryś, "Machine learning in house price analysis: regression models versus neural networks," *Procedia Computer Science*, vol. 207, pp. 435-445, 2022.

[19] R. A. A. a. F. Darnis, "Use of Haversine Formula in Finding Distance Between Temporary Shelter and Waste End Processing Sites," *Journal of Physics: Conference Series*, vol. 1500, 2020.

[20] J. &. H. S. Carroll, "Using a video camera to measure the radius of the Earth," *Physics Education*, vol. 48, no. 6, p. 731., 2013.

[21] P.-B. G. W. Iwan Syarif, "SVM Parameter Optimization Using Grid Search and Genetic Algorithm to Improve Classification Performance," *TELKOMNIKA*, vol. 14, no. 4, pp. 1502-1509, 2016.

[22] Y. W. Y. Z. J. Liu, "New Machine Learning Algorithm: Random Forest," in *Information Computing and Applications. ICICA 2012.*, 2012.

[23] G. Tianqi Chen, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKD International Conference on Knowledge Discovery and Data Mining*, 2016.

Nicușor Andrei earned his bachelor’s degree in Economic Informatics in 2023 at the Academy of Economic Studies from Bucharest. He is a master's student at the same university and is currently working as a Data Analyst with expertise in ETL processes, dashboard designs, and database optimizations. His fields of interest are: data analysis, machine learning, deep learning, and LLM models.

Detection of Fake News Using Deep Learning and Machine Learning

Gabriela CHIRIAC, Ada Maria CATINA
Bucharest University of Economic Studies
Faculty of Cybernetics, Statistics and Economic Informatics
Bucharest, Romania
chiriacgabriela20@stud.ase.ro; catinaada20@stud.ase.ro

Automatically identifying fake news is a complex challenge, involving detailed knowledge of how fake news is propagated and advanced data processing technologies. The use of Machine Learning and Deep Learning algorithms for misinformation detection involves a continuous learning process as manipulation methods are constantly evolving. Effective detection of fake news requires constant adaptation of algorithms to keep up with new disinformation methods. While these technologies offer promising solutions, the challenge is to calibrate them properly so that they work optimally in different contexts.
This paper explores automated methods for detecting fake news, analyzing the effectiveness of the various techniques and how they can be improved to face the challenges of data quality, domain variability and the continuous evolution of disinformation strategies.

Keywords: *misinformation, Machine Learning, Deep Learning, fake news.*

1 Introduction

In the age of digitalization and social media platforms, the spread of fake news has become a real challenge. Today, online platforms provide us with access to information, but this freedom also comes with the risk of falling prey to cybercriminals who spread fake news.

Fake news is defined as text, images, videos or audio files intentionally created to misinform public opinion. Social media platforms enable misinformation to spread very quick, exposing thousands of users just in few minutes. Their purpose is to influence people's opinions about an individual or group of individuals, to create social unrest and to undermine trust in institutions.

Because of the quick spread and big volume of information, distinguishing between real and false news has become very difficult. Manually verifying information can be a slow and difficult process due to the large amount of data being produced every day.

Social media is a key way to consume news, but it can have both advantages and disadvantages. It is great for access, quick and cheap, as well, but it also makes the spread of fake news easier. In many cases, the quality of information can be unreliable, as some information are intentionally falsified. Detecting this type of deception has become a research topic, attracting increased interest in recent years [1]. According to one of the EU Agency for Cybersecurity reports [2], disinformation and misinformation are a critical challange to global security. In this report we can find that the use of cloud computing technologies and AI algorithms contributes to the creation of misinformation, accentuating challenges related to the *security and integrity of the online environment*.

Another important aspect highlighted by the European data Protection Supervisor [3] is the use of complex applications called *bots*, which are organized into networks and used heavily to amplify disinformation content. Bot networks are organized by foreign actors attempting to

mask the true initiators, thus further complicating the detection process. However, recent research indicates that while automated bots contribute to the spread of fake news, the influence of human behavior, especially *word of mouth* marketing, has an even greater impact. As such, the issue of how to combat the spread of fake news can't come down to how to identify the people who produce and spread it. We must equip people to spot fake news and teach them critical thinking so they can keep the danger away. Thus, media education becomes the key factor in the fight against fake news, helping to build a society that is better prepared to deal with manipulation [3].

This phenomenon has serious consequences for society, and various advanced methods have been developed to combat it, applying machine learning and deep learning algorithms. The algorithms are trained to verify news content by exposing them to large volumes of data. This training process involves the algorithms learning to distinguish between real and fake news based on a set of properties extracted from the text, images, or metadata associated with the news. Since the spread of fake news is constantly changed, these methods need to continuous update to keep up with sophisticated tactics used by disinformation creators to mislead both users and detection systems. [3]

This paper aims to explore advanced *Machine Learning and Deep Learning* methods used in the automatic detection of fake news. This study will examine different techniques and algorithms that can be applied to classify informational content, aiming to evaluate their effectiveness in detecting fake news and combating them.

2. Literature Review

Numerous studies have explored solutions for identifying and classifying fake news from various data sources, using different approaches to increase the accuracy of predictions. In this section, we will analyze significant contributions from the research literature that address various techniques and methods for detecting this problem.

In 2015, several researchers [4] adopted a text-based approach, dividing fake news into three typologies: serious inventions, large-scale jokes, and parodies. They explore the difficulties encountered in developing a false news detection system, highlighting the importance of taking into account the typology and context in which they are generated. Perez-Rosas and other researchers [5] have developed an automated algorithm that combines lexical, syntactic and semantic information to detect fake news, bringing more complexity and efficiency to the detection process. In a similar study [6] proposed a hybrid method for detecting fake news, which combines linguistic analysis with social networks evaluation techniques. This method has proven to be successful in identifying fake news, given that social network analysis can reveal the ways in which false information is propagated and amplified on online platforms.

In 2016, Dadgar and collaborators [7] applied feature-extracting techniques such as TF-IDF, and implemented machine learning algorithms such as SVM (support Vector machines) to classify fake news into various categories. Ruchansky, Seo, and Liu [8] proposed a hybrid algorithm named CSI, which integrates three essential components: capture, score, and integration to improve predictions about fake news. This approach underlines the importance of integrating information from multiple sources to build a more robust detection system.

Regarding social network-based techniques, Shu, Wang and Liu [9] studied another model that analyzes factors such as the position of news and user interactions on social networks. This approach is extremely relevant, given how much fake

news and manipulation rely on social media platforms. Similarly, a study by Jin, Cao, Zhang, and Luo [10] proposed an approach based on identifying conflicting points of view in social networks. By applying this method to real data sets, the authors demonstrated how differences of opinion can be indicators of the presence of misleading information.

Several researchers participating in a conference on natural language processing [11] used an attention-based long short-term memory (LSTM) network to detect fake news, demonstrating the use of advanced technologies, such as neural networks, to improve the detection process.

Recent studies, such as that of Janze Christian [12], have applied these techniques in the context of the 2016 US election to observe the impact of fake news on election campaigns. Butain and Golbeck [13] have created an automated system that detects fake news on Twitter, a platform where its dissemination is a major problem.

To reduce the impact of fake news, several researchers [14] have proposed a competitive model that analyzes the relationship between the original and updated false information, with the aim of minimizing its effect on the public.

On the other hand, Tschiatschek and collaborators [15] proposed a method based on the human signals, using Bayesian inference to improve detection accuracy, and Guacho, Abdali and Papalexakis [16] introduced a semi-supervised false news detection technique, which combines human signals with machine learning algorithms.

These innovative research and solutions highlight the complexity and diversity of approaches in the field of detecting fake news. While technologies have advanced in detection of fake news, the challenges remain due to the quick spread of fake news and to the development of new methods and tools used for disinformation.

3. Methodology

This paper is based on a comparative literature review methodology, aiming to highlight the most efficient methods for automatic detection of fake news using Machine Learning and Deep Learning.

The study was conducted by identifying and selecting relevant scientific papers, published in specialized journals and conference in the field of artificial intelligence and natural language processing. This selection criteria included topic relevance, types od algorithms used, datasets used and evaluation metrics reported. The analysis covered both classical models, such as Logic Regresion, SVM, Random Forest, XGBoost, KNN, and advanced deep learning models, such as CNN, RNN, BERT. The models were compared according to the performance obtained in the analyzed papers.

4. Evaluation Metrics

Various metrics are used to evaluate the performance of algorithms, most of which are based on the confusion matrix. The confusion matrix is a table that shows and compares the actual values with the predicted outcomes of a classification model (Table 1).

Table 1. Confusion matrix

		Current values	
		1	0
Predicted values	1	True positive (TP)	False positive (FP)
	0	False negative (FN)	True negative (TN)

Accuracy measures the percentage of correctly classified instances out of the total number of instances analyzed. It is calculated as the ratio between the sum of True Positives and True Negatives and the

total number of instances. This indicator is particularly relevant in situations where the distribution of classes is balanced, when the number of examples in each category (e.g., fake news and real news) is approximately equal [17].

Precision indicates the proportion of instances correctly classified as positive, relative to the total number of instances predicted as positive. It is calculated as True Positives / (True Positives + False Positives). This indicator shows how accurate the positive predictions are and to what extent the model avoids misclassifying negative instances as positive [17].

The Recall (or True Positive Rate) measures the proportion of correctly identified positive instances in the total existing positive instances. It is calculated as True Positives / (True Positives + False Negatives). This indicator reflects the model's ability to detect all positive cases in the dataset, being especially useful when it is important to minimize omissions [17].

The F1 Score is the harmonic mean of precision and recall and gives a balance between precision and recall. This metric is particularly useful in unbalanced datasets, as it takes into account both the precision and the model's ability to detect all positive instances. In binary classification, True Positives (TP) is the number of events correctly classified as positive and False Positives (FP) is the number of events incorrectly classified as positive, True Negatives (TN) is the number of events correctly classified as negative and False Negatives (FN) is the number of events incorrectly classified as negative [17].

5. Machine Learning in detecting fake news

Machine learning is a branch of artificial intelligence which is concerned with the development of systems that can learn and adapt based on the data that they process. Algorithms are capable of generating predictions, recognizing patterns in data, and making decisions based on the information extracted. They are classified into supervised, unsupervised, and semi-supervised machine learning algorithms, with the first two categories being used most frequently [18].

Supervised learning algorithms are trained on labeled datasets, where the results are known, in order to be able to correctly estimate future events. They learn to correctly associate input data with expected outcomes so that they can make accurate predictions based on new datasets.

Logistic Regression is a binary classification algorithm that estimates the probability that an instance belongs to one of two classes (in this case, true or false). This model is often used for simple classification tasks, it predicts whether an item is true or false based on features extracted from the text. Logistic Regression gives efficient results when there is a linear relationship between the input features and the output label. Although this model is not extremely complex, it is often efficient for data with simple relationships and has been used as a benchmark for comparisons with other more advanced algorithms. The mathematical functions of the hypothesis of logistic regression and cost to obtain an optimal probability are represented as follows:

$$h_\theta(X) = \frac{1}{1+e^{-(\beta_0+\beta_1 X)}} \qquad [19]$$

$$Cost(h_\theta(x), y) = \begin{cases} \log(h_\theta(x)), & y = 1 \\ -\log(1 - h_\theta(x)), & y = 0 \end{cases}$$

Support Vector Machines (**SVM**) is a popular algorithm for classification problems, using a mathematical concept called *maximum edge*. It aims to find a hyperplane that separates the data in the two classes (true/false) by as large a margin as possible. SVM maximizes the

distance between the separating hyperplane and the nearest data points in each class (called support vectors). This large margin improves the generalization of the model. For cases where the data is not linearly separable, SVM uses a technique called kernel trick, which transforms the data into a higher dimensional space where linear separation becomes possible.

Random Forests are an ensemble of decision trees that help improve performance and reduce the risk of overtraining. Each tree in the forest is trained on a random subset of the dataset, obtained by the bootstrap method. At each node, a random subset of features is selected to determine the splitting criterion. The final prediction of the Random Forest model is made by majority vote (for classification) or by averaging the outputs (for regression) of all trees in the forest.

K-Nearest Neighbors (**KNN**) is an instance-based algorithm that classifies an article based on the nearest points in the dataset. KNN has been used in various studies to learn patterns of classification of fake news based on features extracted from the text. This is a simple and efficient algorithm, but it can be slower for large datasets because of the need to calculate distances between instances. Distances between two points can be calculated using the following formulas [19]:

$$Euclidean\ distance = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

$$Manhattan\ distance = \sum_{i=1}^{k}|x_i - y_i|$$

$$Minkowski\ distance = \left(\sum_{i=1}^{k}|x_i - y_i|^q\right)^{1/q}$$

AdaBoost (Adaptive Boosting) **and XGBoost** (Extreme Gradient Boosting) are two of the most popular boosting methods used to enhance the performance of classification and regression models. Boosting is a technique that combines several weak models (usually simple decision trees) to build a strong and robust model. The basic idea is that the models are trained sequentially, with each new model focusing on correcting errors made by previous models.

AdaBoost increases the weight of instances that have been misclassified by previous models, making them more influential in the next iteration. XGBoost, on the other hand, applies a more advanced, gradient-based boosting technique that trains decision trees in a highly efficient and parallelized manner. In a study [19], XGBoost had a significant impact on improving overall accuracy and performance due to its ability to manipulate complex data and reduce prediction errors. XGBoost was one of the algorithms that contributed to the outstanding results for fake news classification.

Decision Trees (**CART**) is an intuitive machine learning method that builds a tree-like model, where each internal node represents a feature-based question, each branch corresponds to a possible answer, and the leaves indicate a classification or a numerical value (in the case of regression). In classification tasks, the tree splits the data based on features to create subsets that are as "pure" as possible. Splitting decisions are typically based on measures such as entropy or the Gini index, which assess the quality of the partitions.

In the research [19], decision trees (CART) have been used as part of an ensemble, playing a significant role in creating accurate classifications, especially for categorical or textual data. Although decision trees are easy to interpret, they can be less effective in handling data with multiple meanings or in detecting complex relationships.

A study conducted by two researchers from India [20] compared the performance of Logistic Regression and SVM in

detecting fake news distributed on social platforms. The results of applying the SVM and LR models are:

Table 2. Results from applying SVM and LR models on a fake news dataset [20]

Model	**Accuracy**	**Precision**
SVM	0.91	0.89
LR	0.95	0.93

Model	***Recall***	***F1-Score***
SVM	0.73	0.75
LR	0.79	0.83

On a dedicated dataset, LR achieved 95.12% accuracy and 93.62% precision, while SVM had 91.68% accuracy and 89.20% precision, showing a slight superiority of LR. The authors proposed a novel framework called Novel Fake News Detection (NFND) based on Logistic Regression and emphasized the need to extend the datasets and integrate advanced technique such as POS tagging, word2vec, and topic modeling to improve the performance.

In another study, some researchers [5], created two distinct datasets: one generated by crowdsourcing, covering six diverse domains (sports, business, entertainment, politics, technology and education) and another collected from the web, focusing on celebrity news. The true news articles originated from trusted sources such as ABCNews, CNN and The New York Times, while the fake articles were written by trained workers on Amazon Machanical Turk to mimic journalistic style. For classification, a linear SVM was used with five-fold cross-validation, and the model performed exceptionally well using linguistic features such as readability, punctuation and the LIWC lexicon. The results showed high accuracy, sometimes even outperforming human annotators, although cross-domain generalization remained difficult, suggesting that structural and linguistic differences exist between fake news across different content areas.

On the other hand, Conroy and collaborators [6] proposed a complementary approach, integrating SVM into a hybrid system that combines linguistic technique with social network analysis. In this context, SVM vas applied to datasets containing both real and fake articles, including satirical and manipulated news, and demonstrated strong performance in identifying deceptive patterns. The authors concluded that this integration significantly improves fake news detection by leveraging both textual content and social context.

Both studies confirm the usefulness of SVM in automatic detection, as an efficient classification model based on linguistic features and as a part of complex hybrid system that combine different typed of data.

Acording to the study by Ahmad and others [19], the performance of several machine learning algorithms on four different datasets were evaluated. The obtained results were analyzed in order to compare the efficiency of each algorithm, depending on the characteristics of the datasets and their typology. The datasets used are open source and include both real and fake articles from various domains.

The first dataset, DS1 (ISOT Fake News Dataset), contains almost 45.000 articles, half are real, mainly from Reuters.com, and half are fake, sourced from disinformation websites, mostly political. The second dataset, DS2, available on Kaggle, includes more than 25.000 articles from various domains, split into training and test sets. The third dataset, DS3, also from Kaggle, has 3.300 articles from trusted sources such as CNN and The New York Times, as well as from untrusted sources focusing on sports, entertainment and politics. To allow for a more comprehensive evaluation, a combined dataset (DS4) was created, integrating all the articles from the previous sets.

Table 3 summarizes the average performance scores calculated across all four datasets to compare the overall effectiveness of the evaluated classification algorithms.

Figure 1 is a graphical representation of the average performance of the algorithms on all datasets [19].

Table 3. Average performance of learning algorithms [19]

Model	Average Precision	Average Recall	Average F1-Score
LR	0.93	0.92	0.92
LSVM	0.68	0.79	0.72
RF	0.80	0.80	0.79
KNN	0.70	0.67	0.68
AdaBoost	0.92	0.92	0.92
XGBoost	0.95	0.94	0.94
Decision Tree (CART)	0.94	0.94	0.94

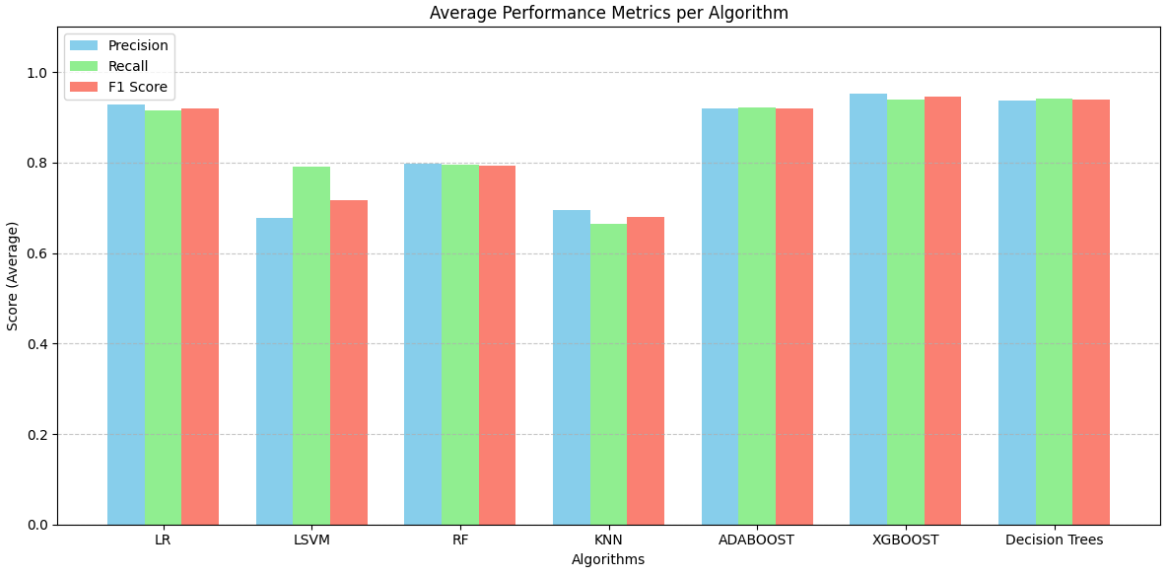


Fig. 1. Average performance of learning algorithms [19]

XGBoost ranked as the best performing algorithm, achieving an average F1 score of 0.95. Its superior performance is due to its ensemble boosting method and advanced regularization techniques that minimize classification errors systematically. AdaBoost and Decision Trees also demonstrated strong results, with F1 scores of 0.92 and 0.94 respectively, benefiting from adaptive learning mechanisms and hierarchical decision structures [19]. Logistic Regression maintained a constant performance (F1-score = 0.93), showing a particular efficiency on homogeneous datasets, while it showed a moderate decrease (to 0.87) on more heterogeneous data. This suggests its reliability for linearly separable problems, but also limitations in handling complex nonlinear relations. The analysis revealed suboptimal performance for linear SVM (F1 score = 0.73) and KNN (F1 score = 0.68). Linear SVM demonstrated a significant compromise between precision (0.68) and recall (0.79), while KNN performance was

affected by sensitivity to data noise and dimensionality [19].

A relevant example of the application of machine learning algorithms is presented in the study [21], published in the IOP Conference Series: Materials Science and Engineering. The authors used the LIAR-PLUS Master dataset, which contains veracity-labeled claims from the politifact.com platform. The dataset was preprocessed using the NLTK and SAFAR v2 libraries, applying operations such as text cleaning, tokenization, POS tagging and linguistic features extraction (e.g. average word length and adjective frequency). The study compares the performance of several classification algorithms, including Naïve Bayes, Random Forest, XGBoost, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Decision Tree and Linear Regression.

The experimental results showed that XGBoost algorithm achieved the best performance, with an accuracy of 75.30%, an F1-score of 77%, a precision of 76% and a recall of 0.75%. It was followed by SVM, which obtained an accuracy of 73.20% and Random Forest with an accuracy if 72.50%.

Simpler models, like Naïve Bayes and KNN, performed worse, with accuracies of approximately 65% and 62%, while Linear Regression gave modest results, confirming its limitations when addressing complex text classification tasks.

Figure 2 is a graphical representation of the average results of all the algorithms [21]:

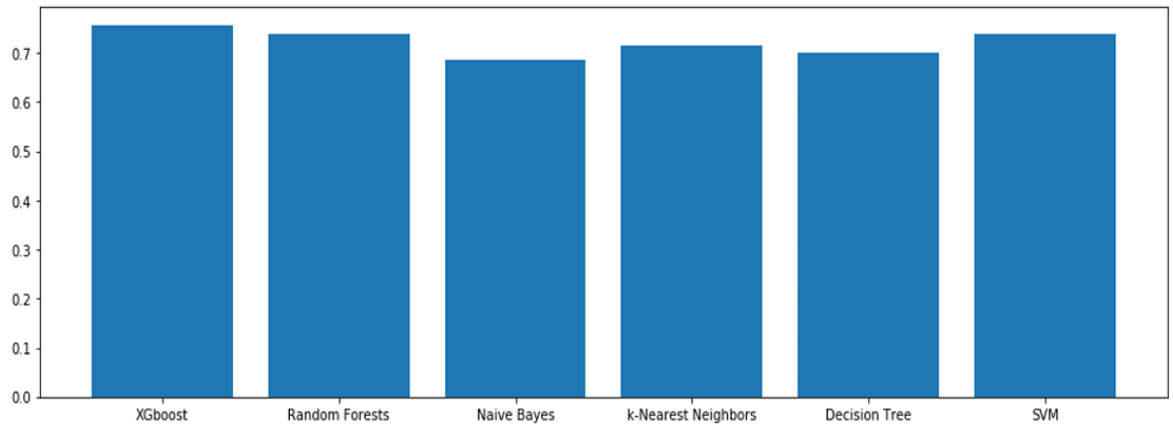


Fig.2. Accurancy Results of all the Algorithms

In addition to individual model testing, the authors also explored hybrid models by combining multiple classifiers through majority voting strategies. The results indicated an improvement in performance: the hybrid model composed of XGBoost, SVM and Random Forest achieved an accuracy of 81.20%, demonstrating increased robustness and reduced classification errors.

The comparative analysis underscored the importance of selecting an appropriate algorithm and complementary models can lead to superior performance in fake news detection tasks.

While supervised learning algorithms have shown promising results in the automatic detection of fake news, several studies have also highlighted their limitation, particularly regarding generalizability in novel contexts or across varied domains. For this reason, recent research has explored complementary methods that combine linguistic analysis and structural information.

Algorithms with unsupervised learning are trained on unlabeled datasets, where the correct results are not known. Their goal is to identify hidden patterns, structures or relationships in the data without knowing a previously provided output. These algorithms learn to organize and classify data on their own, discovering clusters or similar features, which can then be used to understand and interpret new data. This approach is useful when manual labeling is impossible. Although less precise than supervised methods, clustering provides valuable information in the absence of labels.

A key factor in one of the success factors behind machine learning algorithms is *parameter tuning*. This means that you will have to tune important parameters like the number of trees in a random forest, the level of regularization in a logistic regression or the number of neighbors in the KNN. The optimal choice of these parameters allows algorithms to become much more efficient in identifying false news, especially when using complex and large datasets.

Feature selection techniques, such as dimensionality reduction (e.g. PCA), can help improve results by eliminating irrelevant variables and preventing over-training. Appropriate adjustments of these parameters and features can lead to better performance, increasing the effectiveness of systems designed to detect fake news.

A relevant example of the application of unsupervised learning in fake news detection is presented in the study by Yang and colleagues [22], where they develop an original method based on probabilistic modelling. This research tackles the challenge of identifying fake news without using manually labeled datasets, which differentiates it from most existing approaches. The proposed model, called **UFD** (Unsupervised Fake News Detection), is based on the idea that users' interactions on social media, such as likes, retweets or comments on posts, may reflect their perceptions of a news story's veracity. They consider both news veracity and user credibility as latent variables and incorporate them into a generative model built on Bayesian networks. To validate the method's performance, they used two real datasets: LIAR and BuzzFeed News.

The UFD model demonstrated notably strong results. Compared to other unsupervised methods (Majority Voting, TruthFinder, LTM, CRH), the proposed model performed better.

For **Majority Vote** each news item gathers the opinions of verified users, and the most common version is considered true.

TruthFinder tries to find out which information is true, even if user opinions contradict each other. It analyzes the conflicts between opinions and calculates which information is most likely to be correct, even without knowing in advance what is true or false.

LTM is a model that recognizes that users can be wrong. It tries to discover the truth even if some people give wrong information, but it needs a simple structure in which each source says something about particular news item.

CRH assesses how correct each user is in general. If a user has given correct information many times, their opinion will count more in the final decision. This calculates how credible everyone is and determines which information is true.

On the LIAR set, UDF achieved an accuracy of 75.90%, while the other methods ranged between 58% and 64%. Additionally, the model attained an F1-score of 74.1%, exceeding the performance of the majority voting method of over 20 percentage points.

An important advantage of this approach is its dual capability: it not only estimates the veracity of news content but also assesses the credibility of users. This dual output enhances the model's reliability and its

capacity to extract valuable insights from unstructured data.

Machine Learning has shown considerable potential in detecting fake news, with choosing the right algorithms being the most important step in obtaining accurate results. Algorithms such as XGBoost, AdaBoost and logistic regression have proven effective in correctly classifying articles, while unsupervised learning can uncover hidden patterns in the data, bringing additional value to the detection process. Combining multiple algorithms can also help create a more robust model.

By integrating multiple algorithms and evaluating them across diverse datasets, it becomes possible to identify the most suitable models for the automatic detection, depending on the type of data, the complexity of the relationships between variables and the computational resources available. The reviewed studies confirm that ensemble models, such as XGBoost and Random Forest, offer an excellent balance between accuracy and robustness, like Logistic Regression, and can also yield competitive results in more constrained scenarios, with the added benefit of interpretability. Moreover, the adoption of hybrid models that combine linguistic features, contextual signals and meta-information extracted from social media represents a promising direction for enhancing the reliability of fake news detection systems.

However, the ultimate success of a system is highly dependent on continuous optimization of algorithms and hyperparameter tuning, which can make the difference between an accurate model and a less performing one. Future research should incorporate these aspects to increase the accuracy and adaptability of counter-disinformation systems.

6. Deep Learning in fake news detection

Another branch of Artificial Intelligence, related to Machine Learning, is *Deep Learning*, abbreviated DL. Deep Learning consists of learning methods that allow computers to learn independently, without the intervention of a human factor to define rules or knowledge. These models are structured as an artificial neural network because of their architecture which is made of interconnected nodes across multiple layers, similar to the neurons of a biological brain. The difference between DL and ML is that the latter involves the use of neural networks that have an input layer of neurons, an output layer and sometimes 1-2 hidden layers, whereas Deep Learning uses *deeper* neural networks, as the name suggests, with multiple hidden layers.

An example of a deep neural network can be seen in the figure below:

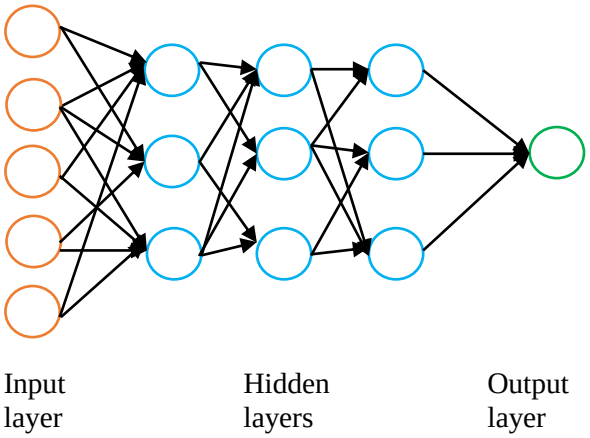


Fig. 3. Example neural network with multiple hidden layers

Natural Language Processing, or **NLP**, is another branch of AI, like ML and DL. Using NLP, computers can understand human language [23]. This technology is the basis of intelligent assistants. With NLP, these assistants can understand and reproduce the human language. NLP provides a number of data *pre-processing techniques* that are essential for ML and DL. These methods necessary especially for the analysis of a group of sentences, such as a news story. An NLP method is

tokenization in which a text is split into a sequence of tokens, consisting of words or parts of words. Another approach is the *Bag-of-Words* model, in which a data set is viewed as a multitude of words put together, ignoring their order. This calculates the frequency of occurrence of words in the text. One use of Bag-of-Words is searching for certain information on the Internet or other search engines. Bag-of-Words technique along with **TF-IDF** are vectorization methods often used in ML and DL models for fake news detection.

Another vector-based NLP technique is the **word embeddings** technique, which assigns different real number vectors to words, with the resulting representations being distributed, with a reduced dimension. This method is considered a more improved version of Bag-of-Words, the vector space being continuous and multidimensional. Models often used in fake news detection, pre-trained with word embeddings, are *GloVe* and *Word2Vec*. These have the ability to train very large datasets. Word2Vec represents performance in detecting syntactic relations between words, while GloVe does better with global semantic relations.

A first and common DL model is **CNN** (*Convolutional Neural Network*), first introduced by Kunihiko Fukushima [24]. Convolutional Neural Network implies the use of convolutional layers within the model. These layers filter the input data so that only relevant and useful information is extracted. CNN is basically used for image classification and recognition as input data. Convolutional layers may have other pooling layers attached to them. These layers reduce the size of images to decrease the number of parameters in a network. With other words, a pooling layer is a small portion of the input image and the convolutional operation helps extract its most important features.

A study by M. F. Mridha [25] highlights that *max pooling and average pooling* are the most commonly used functions in CNNs. Their research identified best parameters values for a CNN model in order to perform in detection of the fake news: the dense layer has 100 units, there are 100 filters, and the filter size is 5. *GlobalMaxPooling1D*, which performs global pooling, has the highest score among the CNN approaches, making it the most effective solution for detecting fake news.

As with any other learning model, certain problems arise, such as error reduction or *overfitting*, which occurs when model performance is poor on new data. One of the solutions to the overfitting problem is *regularization*, which is most common in detecting false news, according to Mridha's research.

Another often used model of Deep Learning is **RNN** (*Recurrent Neural Networks*), used for processing sequential data, such as speech and language. The structure of these neural networks includes recurrent links that *memorize* the information from previous execution of the same computation process. In this way, the final result depends on the results obtained during the whole process [24]. A problem encountered in the RNN application is the *disappearance of the gradient in time*.

A variation of the RNN model is RNN with Long Short Term Memory (**LSTM-RNN**) [17]. The added memory (LSTM) is intended to retain information from previous computations over a longer period. In detecting fake news it is important to understand the context, to look at the information as a whole for a correct classification, that is why the LSTM-RNN model is often used in this case. Nowadays, most of the fake news are appearing on social networks. In their study, Sahoo and Gupta [25] tried to detect such news items on the Facebook platform. They considered both the data of the posts and the account information of the people who published them, applying the LSTM-

RNN model. However, due to too much data, the runtime was too long.

Another variation of the RNN model is **GRU** (*Gated Reccurent Unit*), similar to LSTM, but with a simpler architecture. GRU needs fewer parameters, making the train process more efficient. The main difference between GRU and LSTM model is how they manage the memory. LSTM has a separated memory part, which is updated through three gates (input, output and forget). On the other hand, GRU combines the memory and the hidden parts into one, using only two gates (reset and update). Through these gates, GRU model chooses what information is modified. At a conference from 2024, Elfaik and Nfaoui [26] did a comparative study between GRU and other Deep Learning methods, LSTM and RNN, used to detect fake news. They used ISOT Fake News Dataset and after cleaning the data and applying selected methods to identify the fake news, they observed that GRU model had a higher accuracy compared to the others, as we can observe in Table 4 bellow. This means that GRU model is highly effective for detecting fake news with minimal manual features extraction.

Table 4. GRU vs. LSTM and RNN – accuracy values [26]

Model	**Accuracy**
LSTM	0.9969
RNN	0.7448
GRU	0.9983

The LSTM of a recurrent neural network can also be bidirectional, related to the **BiLSTM-RNN** model. This is a variant of the traditional RNN model, but which provides a memory that allows data to be processed from the beginning to the end and from the end to the beginning. BiLSTM-RNN is used when both the past and the future need to be analyzed, it is more powerful than the classical LSTM, but also more expensive.

One pre-trained DL model is the **BERT** model, short for Bidirectional Encoder Representation from Transformers. This model is based on transformers, as the name suggests, and was introduced by Google in 2018 [17]. Transformers have the ability to process a word by taking into account its relationships with other words in the text, allowing the model to understand its context. Thus, BERT is used for natural language processing (**NLP**) in many situations, like translating languages.

As with RNN, BERT has several variations, one of which is **ALBERT**, which is derived from the *A Lite BERT model*, considered to be efficient to use for false news detection. In contrast to BERT, ALBERT uses a smaller projection layer and applies the *Weight Sharing principle*, which divides the weights among all layers, thus reducing the number of parameters.

RoBERTa (Robustly Optimized BERT Approach) is another improved variant of the BERT model, introduced in 2019 by Facebook AI researchers. The difference between the two models is that RoBERTa can process a much larger volume of data and uses improved training procedures. This makes RoBERTa a more powerful tool to use, especially for fake news detection. However, in 2020, following a study, the **FakeBERT** model specifically designed for fake news detection was introduced. FakeBERT is based on the architecture of the BERT model, and the difference costs in the training set containing fake and real news [27]. Another specific feature of FakeBERT is the use of the back-translation technique. This technique involves translating a real news text into a language, and then translating it back into the original language, generating synthetic data that is added to the training set.

7. Deep Learning vs. Machine Learning in fake news detection

Machine Learning and Deep Learning methods show performance in identifying fake news, as we can observe in this paper. ML utilizes classical algorithms such as SVM or Random Forest, while DL uses the principle of deep neural networks such as CNN and RNN. In contrast to ML, DL models have the ability to process large datasets containing raw texts, without human intervention, learning automatically. However, due to its higher performance, Deep Learning involves higher costs in terms of resources and processing time. In contrast, traditional models can be trained and deployed much faster on modest hardware, making them a more practical solution in low-resource environments.

Both branches of Artificial Intelligence have performed well in fake news detection, but their performance differs depending on the details of the problem. For example, if one wants to classify a text as fake or not, then an ML model can be used, but if one wants to analyze a large dataset, such as all the news posted on a web page, then an DL model is better to use, due to its improved ability to work on large datasets.

Several studies have been conducted over time comparing different ML and DL models applied on the same datasets. The Deep Learning methods presented earlier in this paper are the most common methods used in different studies on fake news detection. An example of such work is the comparative study of Deep Learning and Machine Learning methods applied to identify fake news, published in the *Carol* National Defense University Bulletin [17]. The datasets underlying the research include fake news from ISOT, BuzzFeed and PolitiFact websites. DL models, also described in this paper, were applied and results obtained as follows:

Table 5. Results from applying DL models on a fake news dataset [17]

Model	***Accuracy***	***F1 Score***
RoBERTa	0.99	0.99
LSTM-RNN	0.96	0.97
BiLSTM-RNN	0.98	0.97
ALBERT	0.97	0.97
FakeBERT	0.98	0.99
BERT	0.98	0.98
CNN	0.96	0.96

In the table above, we can observe that after applying the models, the best performing model in identifying fake news is RoBERTa, with accuracy and F1 score values very close to 1 and 0.99, respectively. The difference between RoBERTa and the other models is small, which shows that all models were able to correctly identify almost all fake news, the worst performing model being CNN, with an accuracy and F1 score of 0.96.

A case study is the work of S. Repede [17], mentioned earlier in the paper. He applied on 4 datasets several machine learning models. Analyzing the evaluation metrics, it was found that the most efficient model, with the highest performance, is the *RoBERTa* model, with values above 0.99, very close to the maximum value 1.

Another research conducted by Arshad Ali and Maryam Gulzar [28] was based on the detection of fake news on social networks related to the COVID-19 pandemic. In this research, the authors tried to combine two machine learning models, **BERT** (DL) and **SVM** (ML), along with an evolutionary algorithm, **NGSA-II** (Non-dominated Sorting Genetic Algorithm II), to obtain better results, thus creating a new hybrid model. After preprocessing the COVID-19 news dataset, different traditional ML and DL models were applied, including:

Table 6. Results applying classical ML and DL models on the COVID-19 dataset [28]

Model	*Accuracy*	*Precision*
SVM	0.60	0.65
Random Forest	0.68	0.63
K-Neighbour	0.64	0.69
BERT	0.63	0.72
CNN	0.72	0.73
Model	***Recall***	***F1 Score***
SVM	0.71	0.55
Random Forest	0.7	0.66
K-Neighbour	0.65	0.7
BERT	0.63	0.67
CNN	0.82	0.77

We can observe that the applied models have an average performance, the best being CNN, with the highest evaluation metrics values. After the application of the proposed hybrid model, BERT+NSGA-II+SVM, it was found that it performs much better than the traditional models applied individually. The F1 score (0.83) and accuracy (0.8) have values above 0.8, recall has a value of 0.9, and precision is 0.76. All metrics have higher values than those of the CNN model, the best performing of the traditional models applied.

At a conference, Chang [29] conducted a study on different Machine Learning and Deep Learning algorithms used to combat fake news. He used the ISOT dataset containing fake and real news. After pre-processing it, the author compared 15 algorithms, and the results showed that Deep Learning models performed the best, especially the BERT model with 99.95% accuracy. In second and third place were the BiLSTM and LSTM models, showing the superiority of Deep Learning on this dataset. Regarding Machine Learning, the best result belongs to the SVM model, with an accuracy of 98.65%.

A comparative study was carried out by Kishwar and Zafar [30] based on a dataset of news about Pakistan. The two researchers used Google Fact Checker along with a series of words to extract relevant news. In addition to Google Fact Checker, data was also extracted from other sources such as Kaggle. After applying different Machine Learning models and technologies, Deep Learning was found to perform better in identifying fake news. The CNN model in conjunction with GloVe achieved an F1 score of 0.93, while LSTM scored 0.94. An interesting thing introduced in the study is a questionnaire completed by 57 people who had to categorize 10 news stories as fake or real. The results of the questionnaire showed that most of the fake news stories correctly identified by those individuals were also classified as fake by the models applied. However, there were many misclassifications of real news stories as fake because they had a similar writing style to fake news, with information that appeared to be false. Thus, it was shown that ML models can overcome human judgment.

Similar to the previously mentioned studies, Alghamdi, Lin and Luo [31] did a study in which, on several fake news

datasets, LIAR, PolitiFact, GossipCop and COVID-19, they applied different models of both ML and DL. LIAR is a rather voluminous dataset used for identifying fake news, composed of US political statements. Similarly, PolitiFact is a dataset of political statements available on the website of with the same name. On the other hand, GossipCop contains data on celebrity news stories, and COVID-19, as the name suggests, contains news about the COVID-19 pandemic, both sets categorized as real or fake. Their study becomes even more interesting because they applied the same model several times, each time using different pre-processing techniques. Their results showed that the classical ML models, applied together with TF-IDF, perform best on the LIAR dataset, compared to more advanced models such as DL or hybrid models. In contrast, on the PolitiFact dataset, RoBERTabase performs best with a high F1 score of 0.93. Regarding the GossipCop dataset, classical ML models performed better, while on the COVID-19 dataset the BERTbase model, belonging to Deep Learning, performed best. This study shows that there is no universal ideal method that gives the same performance on all datasets, as context as well as other factors (resources, data volume and so on) matter a lot.

8. Conclusion

The problem of the spread of fake news today is growing. People spend much of their time on social media platforms like Facebook, Instagram and TikTok. For various reasons, certain individuals or *bots* post various news, ads or videos containing false information with the aim of mass manipulation or monetization. Therefore, the use of a mechanism to identify false information is essential. Both Machine Learning models, such as Random Forest and SVM, and Deep Learning models, such as RNN and FakeBERT, can be used for this purpose, as we have seen in this paper.

Machine Learning and Deep Learning perform well on this false news topic. However, depending on the size of the dataset, the complexity of the situation and the available resources, an ML model may be more efficient than a DL model or vice versa. It has been shown that for large datasets, in particular fake news datasets, Deep Learning models perform better due to their LSTM, BiLSTM and BERT methods. However, technologies are evolving, and hybrid solutions that have appeared in recent years, such as the BERT+NSGA-II+SVM model proposed and analyzed by Ali and Gulzar [28], which combines ML models with DL models, NLP techniques and other technologies, perform much better and represent the future of combating online misinformation.

References

[1] K. S. A. W. S. T. J. &. L. H. Shu, „Fake News Detection on Social Media: A Data Mining Perspective," *ACM SIGKDD Explorations Newsletter,* vol. 19, nr. 1, pp. 22-36, 2017.

[2] U. A. f. C. ENSIA, "ENISA THREAT LANDSCAPE 2022," *European Union Agency for Cybersecurity,* 2022.

[3] D. P. Supervisor, „Fake News Detection," 2023. [Interactiv]. Available: https://www.edps.europa.eu/press-publications/publications/techsonar/fake-news-detection_en. [Accesat 27 3 2025].

[4] Y. C. N. J. C. V. L. Rubin, "Deception detection for news: three types of fakes," *Proceedings of the Association for Information Science and Technology,* vol. 52, no. 1, pp. 1-4, 2015.

[5] K. A. L. R. M. V. Perez-Rosas, "Automatic Detection of Fake News,"

arXiv preprint arXiv:1708.07104, 2017.

[6] V. L. R. Y. C. N. J. Conroy, "Automatic deception detection: Methods for finding fake news," *Proceedings of the Association for Information Science and Technology*, vol. 52, no. 1, pp. 1-4, 2015.

[7] M. S. A. M. F. SM H. Dadgar, "A novel text mining approach based on TF-IDF and Support Vector Machine for news classification," *Proceedings of IEEE International Conference on Engineering and Technology* , pp. 112-116, 2016.

[8] S. S. Y. L. N. Ruchansky, "CSI: A hybrid deep model for fake news detection," *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 797-806, 2017.

[9] S. W. H. L. K. Shu, "Understanding User Profiles on Social Media for Fake News Detection," *IEEE Conference on Multimedia Information Processing and Retrieval*, 2018.

[10] J. C. Y. Z. J. L. Z. Jin, "News Verification by Exploiting Conflicting Social Viewpoints in Microblogs," *Thirtieth AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, pp. 2972-2978, 2016.

[11] Q. L. R. X. M. L. C. R. H. Y. Long, "Fake news detection through multi-perspective speaker profiles," *Proceedings of the Eighth International Joint Conference on Natural Language Processing*, pp. 252-256, 2017.

[12] M. R. C. Janze, "Automatic detection of fake news on social media platforms," *Proceedings of 21st Pacific-Asia Conference on Information Systems*, 2017.

[13] J. G. C. Buntain, "Automatically identifying fake news in popular twitter threads," *Proceedings of the IEEE International Conference on Smart Cloud, New York*, pp. 208-215, 2017.

[14] W. H. C. J. F. G. L. H. Zhu H., "Information dissemination model for social media with constant updates," *Physica A: Statistical Mechanics and its Applications*, vol. 502, pp. 469-482, 2018.

[15] S. M. G. R. A. M. A. K. S. Tschiatschek, "Fake news detection in social networks via crowd signals," *Proceedings of the World Wide Web Conferences, France*, pp. 517-524, 2018.

[16] S. A. E. E. P. G. B. Guacho, "Semi-supervised content-based fake news detection using tensor embeddings and label propagation," *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 322-325, 2018.

[17] R. B. Ș. E. Repede, "O comparație a modelelor de inteligență artificială folosite pentru detectarea știrilor false," *Buletinul Universităţii Naţionale de Apărare »Carol I«,* no. 1, pp. 81-99, 2023.

[18] Oracle, „Ce este machine learning?," 2025. [Interactiv]. Available: https://www.oracle.com/ro/artificial-intelligence/machine-learning/what-is-machine-learning/. [Accesat 27 Martie 2025].

[19] M. Y. S. Y. M. O. A. I. Ahmad, "Fake News Detection Using Machine Learning Ensemble Methods," *Complexity*, vol. 2020, no. 1, 2020.

[20] M. A. N. Leela Siva Rama Krishna, "Fake News Detection System using Logistic Regression and Compare Textual Property with Support Vector Machine Algorithm," *Proceedings of the International Conference on Sustainable Computing and Data Communication Systems* , 2022.

[21] N. A. H. S. M. R. Z Khanam, "Fake News Detection Using Machine Learning Approaches," *IOP Conference Series: Materials Science and Engineering*, 2021.

[22] S. S. K. W. S. G. R. W. F. &. L. H. Yang, "Unsupervised Fake News Detection on Social Media: A Generative Approach," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 33, no. 1, pp. 5644-5651, 2019.

[23] Eppright, „Ce este NLP (procesarea limbajului natural)?," 2021. [Interactiv]. Available: https://www.oracle.com/ro/artificial-intelligence/what-is-natural-language-processing/.

[24] O. Y. A. U. a. Y. D. M. Coskun, "An Overview of Popular Deep Learning Methods," *European Journal of Technic, Vol. 7, Number 2,* pp. 165-176, 2017.

[25] J. K. M. A. H. M. M. M. a. M. S. R. M. F. Mridha, "A Comprehensive Review on Fake News," *IEEE Access,* pp. 156151-156170, 2021.

[26] H. N. H. Elfaik, "Automatic Detection of Fake News Using Gated Recurrent Unit Deep Model," in *5th International Conference on Innovative Data Communication Technologies and Application*, 2024.

[27] G. a. P. N. R. K. Kaliyar, „FakeBERT: Fake news detection in social media with a BERT-based deep learning approach," *Multimedia Tools and Applications,* pp. 11766-11788, 2021.

[28] M. G. A. Ali, "An Improved FakeBERT for Fake News Detection," *Applied Computer Systems, Vol. 28, No. 2,* pp. 180-188, 2023.

[29] L. Chang, "Comparison of Machine Learning and Deep Learning Algorithms in Detecting Fake News," in *Proceedings of the 28th World Multi-Conference on Systemics, Cybernetics and Informatics*, 2024.

[30] Z. A. Kishwar, "FakenewsdetectiononPakistaninewsusingmachinelearninganddeep learning," *Expert Systems With Applications,* vol. 211, pp. 1-10, 2023.

[31] Y. L. S. L. J. Alghamdi, "A Comparative Study of Machine Learning and Deep Learning Techniques for Fake News Detection," *Information,* pp. 1-28, 2022.

Gabriela Chiriac graduated in 2020 from the Faculty of Accounting and Management Information Systems and she is currently a final year student in the master's program Databases – Business Support at the Faculty of Cybernetics, Statistics and Economic Studies, Bucharest University of Economic Studies. Professionally, she works as a Data Engineer, focusing on developing and maintaining ETL flows, managing databases and providing technical support for internal stakeholders.

Ada Maria Catina graduated in 2020 from the Faculty of Cybernetic, Statistics and Economic Studies, specializing in Economic Informatics. She is currently in her final year of the master's program Databases – Business Support at the Faculty of Cybernetics, Statistics and Economic Studies, Bucharest University of Economic Studies. Her academic interests focus on databases, data science and the application of machine learning techniques.