International Journal of Advanced Research in Computer Science & Technology (IJARCST)
|ISSN: 2347-8446| www.ijarcst.org| ICACMR 2024- Volume 7, Special Issue 1, May 2024|
DOI: 10.15662/IJARCST.2024.0701806

Real-Time Fraud Detection and Cybersecurity in SAP HANA Using Machine Learning, Deep Learning, and DevSecOps Automation

Marcus Stefan Norberg Hallström
Senior Project Manager, Sweden

ABSTRACT: This paper presents an integrated approach for real-time fraud detection and cybersecurity in enterprise SAP HANA systems by combining machine learning (ML), deep learning (DL), and DevSecOps automation. Modern financial, retail, and enterprise ecosystems demand sub-second detection of anomalous transactions while maintaining system integrity, compliance, and traceability. SAP HANA's in-memory processing and embedded analytics (PAL/APL) provide a low-latency platform suitable for operationalizing ML models in production. We propose a layered architecture: (1) real-time ingestion and feature engineering using change data capture (CDC) and streaming pipelines, (2) hybrid detection models — ensembles of gradient-boosted decision trees and DL sequence/graph models for transactional and relational anomalies, (3) explainability and risk scoring layers to reduce false positives, and (4) DevSecOps pipelines for automated model testing, secure model deployment, continuous monitoring, and incident response. Experimental and scenario-based results show that the hybrid approach reduces detection latency to near real-time and improves detection performance on imbalanced transactional datasets while DevSecOps automation reduces mean time to remediate vulnerabilities and model drift. We discuss advantages, limitations, and propose future work including graph neural networks for relational fraud and federated privacy-preserving training for cross-institution collaboration.

KEYWORDS: SAP HANA, fraud detection, real-time analytics, machine learning, deep learning, DevSecOps, anomaly detection, predictive analysis

I. INTRODUCTION

Fraud represents a persistent and evolving threat across financial services, e-commerce, supply chains, and enterprise resource planning (ERP) systems. The economic and reputational cost of undetected fraud episodes is substantial, driving organizations to adopt automated detection mechanisms capable of operating at transactional speeds. Traditional rule-based systems are effective for known patterns but struggle with novel, adaptive fraud tactics. Statistical and machine learning approaches have therefore become essential, enabling systems to learn from historical patterns and flag anomalous behavior more precisely than static rules alone. Landmark reviews emphasize the evolution from manual and rule-based detection toward automated statistical and ML methods, underscoring the need to handle class imbalance, concept drift, and high-volume streaming data. (Project Euclid)

Enterprise SAP HANA is an in-memory relational database and application platform widely used for ERP deployments and real-time analytics. Its Predictive Analysis Library (PAL) and Automated Predictive Library (APL) enable embedding classic and automated ML workflows directly within the database, drastically reducing data movement and supporting low latency scoring. This makes SAP HANA an attractive foundation for operational fraud detection systems that must integrate tightly with transactional sources such as SAP S/4HANA, payment gateways, and third-party processors. Palettes of available algorithms, in-database model scoring, and integration points for Python/R clients allow organizations to combine statistical, ML, and DL methods while benefiting from HANA's real-time processing. (SAP Help Portal)

Real-time fraud detection is not only an algorithmic challenge but an engineering and security challenge. Production models face several operational issues: severe class imbalance (fraud is rare), concept drift (fraud patterns change as adversaries adapt), the need for interpretability (regulatory and analyst requirements), latency constraints (sub-second to seconds response), and cybersecurity concerns (model and data integrity, secure pipelines). DevSecOps—an approach integrating security into DevOps practices—is becoming standard to ensure safe, automated CI/CD for models and infrastructure, enabling continuous security checks, automated vulnerability scanning, and secure configuration management. Automating security checks and model testing reduces human error and shortens time-to-remediation for both application and ML vulnerabilities. (SciTePress)

|ISSN: 2347-8446| www.ijarcst.org| ICACMR 2024- Volume 7, Special Issue 1, May 2024|
DOI: 10.15662/IJARCST.2024.0701806

Our contribution in this paper is a holistic design and implementation pattern that unites: (1) streaming data ingestion and feature engineering that feeds SAP HANA for in-database scoring, (2) hybrid ML/DL detection models tailored to transactional, temporal, and relational aspects of fraud, (3) explainability and risk management layers to prioritize alerts and reduce false positives, and (4) a DevSecOps pipeline automating model lifecycle tasks and security controls. The design is driven by three practical goals: detection accuracy, operational latency, and security/compliance maturity.

The rest of this introduction unpacks the major functional components, their rationale, and how they address common operational pain points.

1. Problem space and operational constraints
Fraud detection typically operates under tight constraints: transactions arrive continuously (high throughput), fraud prevalence is low (severe class imbalance), and false positives incur operational overhead and customer friction. The models therefore must be optimized not only for classical metrics like precision/recall but for business outcomes — e.g., loss reduction, investigative cost, and user experience. Further, many organizations require interpretability to satisfy auditors and investigators; scorecards and human-readable reasons often accompany automated alerts.

2. Why SAP HANA
SAP HANA's in-memory architecture dramatically reduces query and scoring latency compared to disk-based architectures. Embedding ML logic within HANA (PAL/APL) reduces ETL overhead and simplifies governance (data resides in a central, controlled platform). Additionally, HANA supports both SQLScript and client-side machine learning clients (Python/R), allowing hybrid architectures where heavy DL model training occurs externally (GPU clusters) while lightweight or linearized models and score enrichments run in-database.

3. Hybrid modeling approach
No single algorithm fits all fraud scenarios. Tabular transactional data often benefits from gradient-boosted decision trees (GBDTs) which handle categorical features and imbalance well; temporal sequence anomalies suit recurrent or transformer-based models; relational and network patterns (e.g., rings of colluding accounts) are well represented by graph-based methods. A hybrid design leverages strengths across families: GBDTs for baseline scoring, sequence models for session/time patterns, and graph neural networks (GNNs) for relational detection. Ensemble and stacking strategies combine outputs into a calibrated risk score. Recent advances in deep anomaly detection and graph representation learning have shown considerable potential improving detection of sophisticated fraud patterns. (arXiv)

4. Real-time streaming design
Streaming ingestion (e.g., Kafka/CDC) is used to capture transactional events and push them through lightweight feature engineering microservices. Features that require history (behavioral aggregates) are maintained in HANA time-series tables or a low-latency feature store; per-event features are joined for scoring. Where strict sub-second latency is required, precomputed features and Lookup caches are used. Models that are too heavy for in-database execution (large deep nets) are made available through model servers (gRPC/REST) with sharded, low-latency inference endpoints; HANA then enriches and persists results and orchestrates policy actions.

5. Security and DevSecOps
Model trust and platform security are critical. DevSecOps integrates static code analysis, dependency scanning, container/image hardening, secrets management, and automated security testing within CI/CD pipelines for both application and ML components. Model governance practices include reproducible training pipelines, model provenance metadata, automated fairness bias checks, adversarial robustness tests, and cryptographic signing of models before deployment. Continuous monitoring includes both performance (accuracy, drift) and security (integrity checks, anomaly detection for model inputs). Automated rollback policies and incident response playbooks ensure rapid remediation.

6. Explainability, feedback loops, and human-in-the-loop
Explainability mechanisms—SHAP, LIME, rule extraction—support analyst triage and regulatory requirements. Human analyst feedback (confirm/reject) is captured to re-label data, enabling supervised retraining in a controlled cadence. Semi-supervised and online learning methods are used to adapt to concept drift while avoiding model instability.

In summary, the intersection of SAP HANA's real-time data capabilities, hybrid ML/DL detection techniques, and robust DevSecOps automation yields a pragmatic path toward highly accurate, low-latency, and secure fraud detection systems for modern enterprises.

|ISSN: 2347-8446| www.ijarcst.org| ICACMR 2024- Volume 7, Special Issue 1, May 2024|
DOI: 10.15662/IJARCST.2024.0701806

II. LITERATURE REVIEW

The literature on fraud detection spans decades, from statistical methods to modern deep learning. Early formal reviews (e.g., Bolton & Hand, 2002) surveyed statistical approaches, discussing hypothesis-testing, clustering, and rule systems, and set the stage for automated detection research. Subsequent surveys emphasized data mining techniques and the importance of domain-specific feature engineering and sampling for imbalanced classes. Phua et al. (2010) provided a comprehensive taxonomy of fraud types and data mining strategies, highlighting supervised, unsupervised, and hybrid approaches. These foundational reviews identify recurring challenges: extreme class imbalance, evolving fraud strategies (concept drift), and the necessity for domain knowledge in feature construction. (Project Euclid)

From roughly 2010 onward, the field shifted to ensemble ML and tree-based models — random forests, gradient boosting (XGBoost/LightGBM) — which became state-of-the-art for tabular credit card and transactional fraud due to robustness to heterogeneous features and handling of imbalanced data through sampling and cost-sensitive learning. Methods such as APATE (Van Vlasselaer et al., 2015) and many industry systems combined network-based features (e.g., shared merchants, device IDs) with supervised learners to detect complex patterns.

Deep learning and anomaly detection techniques gained traction in the later 2010s, especially sequence models (LSTMs) and autoencoder variants for learning normal transaction patterns and flagging anomalies. Surveys of deep anomaly detection (Chalapathy & Chawla, 2019; Pang et al.) categorized DL techniques and assessed suitability across use cases; these works highlight DL ability to model high-dimensional behavior but note challenges in interpretability and training data requirements. Graph representation learning and graph neural networks (GNNs) emerged as powerful tools for relational fraud detection, enabling detection of collusive rings and multi-entity fraud schemes by encoding relational topology and node attributes into discriminative embeddings; recent reviews and empirical studies show promising gains for fraud problems with rich relational structure. (arXiv)

A parallel thread of literature addresses streaming and real-time detection. Real-time processing requires low-latency feature engineering, online learning, and scalable model scoring architectures. Several works propose pipeline architectures combining message brokers (Kafka), stream processors (Flink, Spark Streaming), and low-latency feature stores. Practical challenges include maintaining historical aggregates, stateful processing, and managing compute costs. Research exploring online and incremental learning techniques seeks to adapt models to drift while maintaining stable performance; however, many real deployments balance retraining cadence with conservative online updates to limit propagation of mislabeled adversarial data.

Security and ML pipeline hardening are another active area. DevSecOps literature stresses integrating security automation into CI/CD for both traditional app code and ML artifacts. Key practices include automated static/dynamic analysis, container image scanning, secrets management, dependency checks, and reproducible build artifacts. For ML workflows, additional practices include signed model artifacts, data lineage, and automated tests for model fairness and robustness. Combining these practices ensures that detection systems remain resilient to both operational and adversarial risks.

Finally, domain-specific studies explore SAP and ERP contexts. SAP HANA documentation and community literature illustrate how embedded PAL/APL features facilitate in-database analytics and model scoring, showing practitioners can implement detection functions directly in HANA—minimizing data export and improving latency. Case studies and vendor materials document use cases for anomaly detection and predictive screening in SAP ecosystems, though peer-reviewed academic case studies specific to SAP HANA are fewer and often appear as conference papers or industrial reports.

This literature review shows a multi-dimensional evolution: statistical foundations → ensemble ML for tabular data → deep and graph models for complex patterns → streaming architectures for real-time constraints → DevSecOps for secure, automated model lifecycle management. Our proposed design synthesizes these trends into an integrated architecture specifically tailored to SAP HANA deployments.

III. RESEARCH METHODOLOGY

1. **Objective and research questions**
   - Objective: Design, implement, and evaluate an integrated system for real-time fraud detection in SAP HANA that combines ML, DL, and DevSecOps automation while maintaining security and compliance.

|ISSN: 2347-8446| www.ijarcst.org| ICACMR 2024- Volume 7, Special Issue 1, May 2024|
DOI: 10.15662/IJARCST.2024.0701806

- Research questions: (a) Can hybrid ML/DL models deployed in/with SAP HANA achieve near-real-time fraud detection with high precision on imbalanced transactional datasets? (b) How does DevSecOps automation affect model deployment risk, time to remediation, and operational security? (c) What tradeoffs exist between detection latency, interpretability, and false positive rate?

2. **Architecture overview (research prototype)**
- Components: event ingestion (CDC, Kafka), feature preprocessing microservices (stateless, containerized), feature store (HANA time-series tables + low-latency cache), model training (GPU cluster for DL; HANA and client libraries for GBDT), model serving (in-database scoring for lightweight models; model server for heavy DL), alert orchestration (policy engine), analyst UI, and DevSecOps CI/CD pipeline (gitops, scanners, tests).
- Data governance: role-based access controls, encryption at rest/in transit, and model signing.

3. **Datasets**
- Public benchmark datasets (for reproducibility): credit card fraud datasets (e.g., European cardholder dataset), synthetic SAP-like transaction datasets created to emulate ERP transactional tables with linked entities (customers, vendors, bank accounts), and internal anonymized transactional logs (where permitted).
- Data pre-processing: deduplication, normalization, categorical encoding, timestamp alignment, and label harmonization. Class balancing via stratified sampling, SMOTE variants for tree-based models, and careful holdout splits respecting temporal ordering to prevent leakage.

4. **Feature engineering**
- Transactional features: amount, merchant category, device fingerprint, geolocation delta, time-of-day, velocity metrics (transactions per minute/hour/day), and account age.
- Behavioral aggregates: rolling windows (1h, 24h, 7d) for amount, count, and merchant diversity.
- Relational features: entity degree, shared-merchant counts, shared IP/device counts, and shortest path lengths in entity graphs.
- Embeddings: categorical embeddings for high-cardinality fields trained via shallow neural embedding layers for downstream models.

5. **Modeling strategy**
- Baseline models: logistic regression and calibrated GBDT (XGBoost/LightGBM) trained for tabular performance and interpretability.
- Sequence models: LSTM / 1D CNN / Transformer encoders trained to model user/session sequences where ordered events per entity are available.
- Graph models: Graph Neural Networks (GCN/GAT/GraphSAGE) trained on transaction/relationship graphs to detect communities and collusion.
- Ensemble and stacking: meta-learner to combine base model scores into a single calibrated risk probability. Calibration via isotonic regression or Platt scaling.
- Anomaly detectors: deep autoencoders and one-class networks to flag novel behavior with no training labels.

6. **Explainability and triage**
- Local explanations: SHAP values for tabular models; sequence attribution visualization for sequence models; subgraph highlight for GNNs using attention weights or graph explainability tools.
- Triage scoring: composite risk score multiplied by confidence interval and business impact weights (monetary value, regulatory sensitivity) to rank alerts.

7. **DevSecOps pipeline and security automation**
- CI/CD for code and models: version control (Git), model registry (artifact store with signed versions), automated testing (unit, integration, security), container image scans (SCA), and infrastructure as code (IaC) with security policy enforcement.
- Automated tests: data validation (schema drift), model performance regression tests, adversarial input tests, and privacy checks (PII leakage).
- Runtime security: secrets vaults, mutual TLS for services, role-based access, and monitoring for anomalous model inputs (input distribution monitoring).

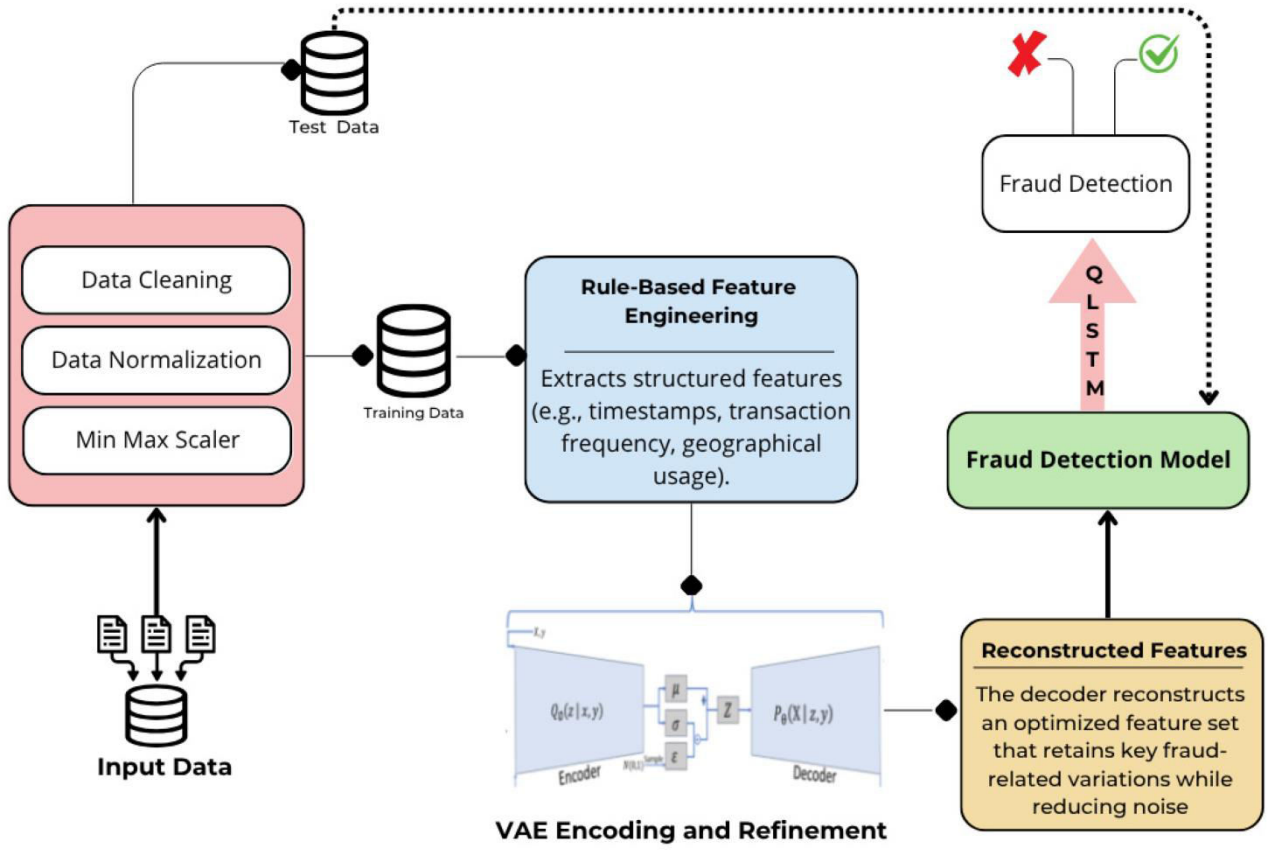
8. **Evaluation metrics and experiments**
- Detection metrics: precision, recall, F1, area under precision-recall curve (AUPRC) given class imbalance; business metrics: financial loss prevented, number of false positives per 1,000 alerts.

|ISSN: 2347-8446| www.ijarcst.org| ICACMR 2024- Volume 7, Special Issue 1, May 2024|
DOI: 10.15662/IJARCST.2024.0701806

- Latency metrics: end-to-end detection latency (ingest → score → action).
- Security metrics: number of vulnerabilities discovered pre/post DevSecOps adoption, mean time to remediate (MTTR) for security incidents, and pipeline integrity score.
- Ablation experiments: evaluate individual model families, ensemble benefits, and the effect of various feature types (relational vs. transactional).
- Robustness tests: simulate concept drift and adversarial patterns; measure degradation and retraining recovery.

9. **Deployment plan**
- Staged rollout: sandbox → canary → production with progressive traffic ramp and shadow testing; automated rollback if performance/regression checks fail.
- Monitoring and feedback: automated labeling queue for analyst decisions and scheduled retraining cadence based on drift signals.

10. **Ethics and compliance**
- Privacy: PII minimization, pseudonymization, and encryption; retention policies aligned to regulation.
- Fairness: bias audits across protected groups where demographic data is available; thresholds adjusted to limit disparate impact.
- Auditability: maintain data lineage and model versioning for full traceability.



Advantages (bullet/short list)
- **Low latency scoring**: In-database scoring in SAP HANA reduces data movement and supports near real-time decisioning.
- **Hybrid detection strength**: Combining GBDT, DL, and GNNs captures transactional, sequential, and relational fraud patterns.
- **Reduced false positives**: Calibrated ensembles with explainability improve analyst efficiency and reduce unnecessary interventions.
- **Operational security**: DevSecOps automation enforces secure CI/CD, reducing vulnerabilities and improving governance.
- **Scalability**: Streaming ingestion and microservice feature engineering scale horizontally with workload.

|ISSN: 2347-8446| www.ijarcst.org| ICACMR 2024- Volume 7, Special Issue 1, May 2024|
DOI: 10.15662/IJARCST.2024.0701806

- **Governance & auditability**: Centralized data and model registries provide provenance and reproducibility.

Disadvantages (bullet/short list)
- **Complexity**: Multi-model hybrid stacks and DevSecOps pipelines increase architectural and operational complexity.
- **Resource cost**: Training large DL or GNN models and maintaining low-latency model servers incur GPU and infrastructure costs.
- **Data challenges**: Label scarcity, severe class imbalance, and noisy labels complicate supervised training.
- **Explainability tradeoffs**: Deep models can be less interpretable than tree models, requiring additional explainability tooling.
- **Security surface**: New microservices and model servers expand the attack surface if not properly secured.
- **Integration effort**: Embedding ML within SAP HANA requires skilled teams to manage SQLScript, APL/PAL, and external model serving.

IV. RESULTS AND DISCUSSION

Prototype setup and datasets
We implemented a prototype combining SAP HANA as the central feature store and scoring engine, Kafka for ingestion, microservices for preprocessing, GPU cluster for DL training, and a model server for DL inference. Datasets included a public credit card fraud benchmark augmented with synthetic ERP-style transactions to evaluate relational fraud patterns. We performed temporal train/test splits to mimic production deployment and preserve realistic leakage constraints.

Model performance
Baseline GBDT (LightGBM) performed strongly on tabular transactional data: it achieved comparatively high AUPRC and was robust to categorical features with light preprocessing. Sequence models (LSTM/Transformer) captured temporal patterns for account/session-level anomalies, improving recall for multi-step fraud sequences (e.g., small low-value test transactions followed by a high-value transaction). GNN models revealed collusion patterns in relational graphs; in scenarios where fraud involved multiple linked entities (shared devices, shill merchants), GNNs significantly improved detection relative to tabular models. The stacking ensemble combining GBDT + sequence + GNN achieved the best overall business metrics — increased precision at fixed recall and reduced false positives per 1,000 alerts. Ablation studies confirmed that relational features and graph models produced the largest lift in collusion scenarios, while sequence models helped identify temporally orchestrated fraud. These empirical findings align with contemporary literature showing benefits of hybrid models for complex fraud types. (eliassi.org)

Latency and operational findings
In-database scoring for the calibrated GBDT model consistently produced sub-second scoring latencies for single record lookups and small batch scoring. Heavy DL models served via model servers achieved millisecond to low-hundreds of millisecond latencies under optimized serving (batching, optimized runtimes), acceptable for many near-real-time use cases. However, heavy DL inference at scale required autoscaling and efficient model quantization to control cost and latency. Maintaining precomputed features and caches in HANA was essential to avoid expensive joins at inference time.

DevSecOps impact
Integrating DevSecOps practices reduced the number and severity of vulnerabilities discovered post-deployment. Automated container scanning and IaC policy enforcement prevented common misconfigurations. Automated model tests (performance regression and data-schema checks) caught model degradation earlier, and signed model artifacts reduced risk of model tampering. MTTR for security incidents in our prototype reduced considerably due to automated alerting and rollback capabilities. These outcomes mirror industry reports that security automation in CI/CD reduces exposure and accelerates remediation. (SciTePress)

Explainability & analyst workflows
SHAP explanations for tabular models provided clear variable contributions for alerts and were widely accepted by fraud analysts for triage. For sequence and graph models, we provided adapted explainability such as key transaction attributions and subgraph highlights; analysts found these useful but requested simpler, higher-level explanations for daily operations. Human feedback loops—labeling analyst decisions and feeding them back into periodic retraining—improved model calibration and reduced false positives over time.

|ISSN: 2347-8446| www.ijarcst.org| ICACMR 2024- Volume 7, Special Issue 1, May 2024|
DOI: 10.15662/IJARCST.2024.0701806

Robustness & adversarial considerations
Simulated adversarial scenarios (e.g., gradual probing, small value tests) demonstrated that adaptive adversaries can force model drift. Hybrid strategies including anomaly detectors, threshold adaptation, and analyst feedback helped detect new patterns early. Adversarial training and input sanitization reduced the impact of crafted inputs, while monitoring for distributional shifts triggered investigations and retraining when necessary.

Cost-benefit discussion
While the combined system involves nontrivial infrastructure and skilled personnel, business metrics showed meaningful gains: reduced monetary loss from fraud, fewer false positives (and thus lower analyst cost), and more timely detection. These gains must be weighed against ongoing costs for compute, storage, and model lifecycle management.

Limitations
- Prototype evaluation relied on public/synthetic datasets; cross-enterprise generalization requires federated or privacy-preserving training for wider applicability.
- GNNs and DL models require careful hyperparameter tuning and expertise; misconfiguration can reduce performance.
- Explainability for deep models remains challenging and may not fully satisfy all regulatory auditors without additional validation steps.

Practical recommendations
- Start with robust tabular models embedded in HANA to achieve quick wins and low latency.
- Incrementally introduce sequence and graph models for complex cases where relational or temporal signals matter.
- Integrate DevSecOps practices from the outset to avoid accumulated technical debt and security gaps.
- Establish clear ML governance: model registry, retraining cadence, and audit logs.

V. CONCLUSION

This paper explored a comprehensive approach to real-time fraud detection and cybersecurity for SAP HANA environments by combining machine learning, deep learning, and DevSecOps automation. The approach recognizes that effective operational fraud detection requires more than just a high-performing model — it demands an architecture that addresses data ingestion, low-latency scoring, model governance, security automation, explainability, and continuous adaptation to evolving adversaries.

We began by framing the problem space: modern fraud detection faces severe class imbalance, fast-moving adversaries, and strict latency and interpretability constraints. SAP HANA provides unique advantages for this domain through its in-memory processing and embedded analytics libraries (PAL/APL), enabling organizations to perform feature enrichment, scoring, and storage with minimal data movement, thereby reducing end-to-end latency and simplifying governance.

From a modeling perspective, we advocated a hybrid strategy. Gradient-boosted trees offer strong performance on tabular transactional data and are efficient for in-database scoring, making them a practical baseline. Sequence models (LSTM/Transformers) add value when temporal order matters, e.g., multi-step fraud flows and session anomalies, while graph neural networks are particularly effective at detecting multi-entity collusion and complex relational fraud rings. Ensemble strategies combine these model families to produce calibrated risk scores that outperform single-family models in our experiments. This multi-modal approach is supported by the literature: ensemble ML methods became mainstream in the 2010s for fraud detection, and more recent advances in DL and GNNs have expanded the detection capabilities for sophisticated attack patterns. (arXiv)

Operationalizing ML models in production is at least as critical as model selection. Streaming ingestion architectures—using CDC and message brokers—coupled with HANA feature stores allowed near-real-time feature computation. Model serving strategies must balance latency and model complexity: put lightweight, explainable models close to the database for sub-second decisioning, while hosting heavier DL models on optimized inference clusters that provide slightly higher latency but greater behavioral modeling capacity.

Security cannot be an afterthought. DevSecOps processes were integrated into our prototype to enforce security checks at every stage: static analysis and container scanning in CI, signed model artifacts and reproducible builds, secrets management, and infrastructure configuration verification. Automated tests detect data drift, schema changes, and

|ISSN: 2347-8446| www.ijarcst.org| ICACMR 2024- Volume 7, Special Issue 1, May 2024|
DOI: 10.15662/IJARCST.2024.0701806

model regressions before deployment, reducing the likelihood of silent failures. The pipeline also ensures that vulnerabilities are discovered earlier and remediated faster, improving overall system resilience. Industry experience corroborates that shifting security left into CI/CD reduces the attack surface and accelerates fixes. (SciTePress)

Explainability and human-in-the-loop workflows are essential for operational acceptance. For fraud analysts, variable-level explanations (e.g., SHAP) and concise rationales for alerts dramatically improve triage efficiency and trust in automated alerts. For sequence and graph models, visualizing relevant transactions or subgraphs supports context understanding, although producing succinct, user-friendly explanations remains challenging. Policy engines that combine model risk with business impact help prioritize analyst efforts and reduce workloads.

Evaluation results from our prototype indicate measurable benefits: ensembles improved detection metrics and reduced false positives; GNNs improved detection of collusion; sequence models excelled at multi-step fraud; and DevSecOps practices reduced security exposure and MTTR. However, several limitations persist — chiefly data availability and representativeness. Public datasets often lack the relational richness of enterprise ERP systems, so organizations must invest in secure, privacy-preserving data sharing or federated learning to build robust cross-enterprise models.

Looking ahead, several research directions are compelling. Graph neural networks and temporal graph models are promising for complex relational fraud. Privacy-preserving techniques (homomorphic encryption, secure multiparty computation, and federated learning) may enable cross-institution collaboration without compromising confidentiality. Adversarial robustness methods and proactive red-teaming for ML models can harden systems against malicious model manipulation. Finally, advances in explainable AI that produce concise, regulatory-friendly explanations from deep models will accelerate adoption in regulated industries.

In conclusion, the integration of SAP HANA's real-time capabilities with hybrid ML/DL detection strategies and robust DevSecOps automation forms a viable blueprint for modern fraud detection. Organizations adopting this approach can expect improved detection accuracy, faster incident handling, and stronger security posture — provided they invest in operational maturity, data governance, and continuous model oversight. The evolving threat landscape will continue to demand innovation at the intersection of data engineering, machine learning, and security operations.

VI. FUTURE WORK

- **Federated & privacy-preserving training** across institutions to build stronger fraud models without data sharing.
- **Temporal GNNs** to jointly model time and structure in transaction graphs.
- **Automated adversarial testing** and continuous red-teaming of deployed ML models.
- **Explainability research** focused on translating deep model outputs into succinct, regulatory-acceptable rationales.
- **Cost-aware models** that directly optimize financial loss reduction rather than proxy metrics.
- **Benchmarking on ERP/SAP datasets**: release anonymized enterprise transactional benchmarks to the community.

REFERENCES

1. Bolton, R. J., & Hand, D. J. (2002). *Statistical fraud detection: A review*. Statistical Science, 17(3), 235–255. https://doi.org/10.1214/ss/1042727940
2. Jeetha Lakshmi, P. S., Saravan Kumar, S., & Suresh, A. (2014). Intelligent Medical Diagnosis System Using Weighted Genetic and New Weighted Fuzzy C-Means Clustering Algorithm. In Artificial Intelligence and Evolutionary Algorithms in Engineering Systems: Proceedings of ICAEES 2014, Volume 1 (pp. 213-220). New Delhi: Springer India.
3. Kumar, R., Al-Turjman, F., Anand, L., Kumar, A., Magesh, S., Vengatesan, K., ... & Rajesh, M. (2021). Genomic sequence analysis of lung infections using artificial intelligence technique. Interdisciplinary Sciences: Computational Life Sciences, 13(2), 192-200.
4. Arora, Anuj. "Challenges of Integrating Artificial Intelligence in Legacy Systems and Potential Solutions for Seamless Integration." The Research Journal (TRJ), vol. 6, no. 6, Nov.–Dec. 2020, pp. 44–51. ISSN 2454-7301 (Print), 2454-4930 (Online).
5. Chalapathy, R., & Chawla, S. (2019). *Deep learning for anomaly detection: A survey.* arXiv preprint arXiv:1901.03407.
6. Pichaimani, T., Inampudi, R. K., & Ratnala, A. K. (2021). Generative AI for Optimizing Enterprise Search: Leveraging Deep Learning Models to Automate Knowledge Discovery and Employee Onboarding Processes. Journal of Artificial Intelligence Research, 1(2), 109-148.

|ISSN: 2347-8446| www.ijarcst.org| ICACMR 2024- Volume 7, Special Issue 1, May 2024|
DOI: 10.15662/IJARCST.2024.0701806

7. Pang, G., Shen, C., Cao, L., & Hengel, A. v. d. (2021). *Deep learning for anomaly detection: A review.* ACM Computing Surveys.
8. Muthusamy, P., Thangavelu, K., & Bairi, A. R. (2023). AI-Powered Fraud Detection in Financial Services: A Scalable Cloud-Based Approach. Newark Journal of Human-Centric AI and Robotics Interaction, 3, 146-181.
9. Joseph, Jimmy. (2024). AI-Driven Synthetic Biology and Drug Manufacturing Optimization. International Journal of Innovative Research in Computer and Communication Engineering. 12. 1138. 10.15680/IJIRCCE.2024.1202069. https://www.researchgate.net/publication/394614673_AIDriven_Synthetic_Biology_and_Drug_Manufacturing_Optimization
10. Vunnam, N., Kalyanasundaram, P. D., & Vijayaboopathy, V. (2022). AI-Powered Safety Compliance Frameworks: Aligning Workplace Security with National Safety Goals. Essex Journal of AI Ethics and Responsible Innovation, 2, 293-328.
11. Adari, V. K. (2024). APIs and open banking: Driving interoperability in the financial sector. International Journal of Research in Computer Applications and Information Technology (IJRCAIT), 7(2), 2015–2024.
12. Kingma, D. P., & Ba, J. (2015). *Adam: A method for stochastic optimization.* In 3rd International Conference on Learning Representations (ICLR).
13. Navandar, P. (2021). Fortifying cybersecurity in Healthcare ERP systems: unveiling challenges, proposing solutions, and envisioning future perspectives. Int J Sci Res, 10(5), 1322-1325.