Journal of Sustainability, Policy, and Practice
EISSN: 3105-1448 | PISSN: 3105-143X | Vol. 1, No. 4 (2025)

Article

Real-Time Fraud Risk Scoring through Behavioral Sequence Analysis: An Explainable Approach for online Transaction Security

Minghua Deng [1,*]

[1] Computational Data Science, Carnegie Mellon University, PA, USA
* Correspondence: Minghua Deng, Computational Data Science, Carnegie Mellon University, PA, USA

Abstract: Payment fraud continues to evolve, resulting in global losses of approximately $33.83 billion last year alone. We developed a system capable of highly effective fraud detection by analyzing spending behavior over time rather than focusing solely on individual transactions. While fraudsters can imitate a single purchase, it is far more difficult for them to replicate an entire behavioral history. Our key innovation lies in a hierarchical architecture that combines modified gated recurrent units (GRUs) with gradient boosting methods, allowing each component to perform its specialized role. The proposed system achieves an area under the precision-recall curve (AUPRC) of 0.876 and a Recall@1%FPR of 0.834 on the IEEE-CIS dataset comprising 590,540 transactions. In production-level stress tests, it sustains throughput up to 12,000 transactions per second under peak load conditions (with an average daily volume of approximately 3.4 million across partner institutions) and maintains a median latency of 47.3 ms (95% CI: [45.8, 48.9] ms; p95: 67.2 ms), remaining consistently below the 50 ms operational threshold. The system's core mechanism involves a temporal-gap-aware gating module within the modified GRU encoder that captures irregular intervals between purchases, effectively distinguishing genuine consumer behavior from fraudulent activity. This encoder is integrated with an ensemble of LightGBM, XGBoost, and Random Forest models, enabling robust voting-based decision fusion. To balance interpretability and performance, the system employs selective explainability-providing gradient-based attributions and counterfactual explanations only for the 5% of transactions flagged as high-risk-thereby ensuring regulatory transparency without compromising speed. Extensive validation using real banking datasets from three financial institutions confirmed its reliability and practicality. While not flawless, the system demonstrates sufficient robustness and interpretability for large-scale deployment-an outcome that ultimately matters more than theoretical optimality when millions of dollars are at stake each day.

Keywords: behavioral sequence analysis; fraud detection; explainable AI; real-time processing

Received: 15 September 2025
Revised: 20 October 2025
Accepted: 09 November 2025
Published: 15 November 2025

1. Introduction

The scale of financial fraud is staggering. In 2023, payment card fraud alone resulted in losses of $33.83 billion worldwide. The United States accounted for $14.3 billion of that total, despite representing only about one quarter of global card volume. Yet the statistics reveal only part of the story-the nature of fraud itself has changed dramatically. Modern fraud operations are no longer simple cases of stolen card numbers. Today's attackers employ advanced techniques: they use machine learning models, conduct A/B testing on

attack strategies, and continuously optimize their methods much like major corporations refine marketing campaigns.

Our internal data indicates that approximately 42.5% of detected fraudulent activity now exhibits algorithmic regularities. These include velocity-based attacks designed to exploit batch processing windows, transaction sequences calibrated to stay just below detection thresholds, and merchant category progressions that imitate legitimate consumer behavior. Traditional rule-based systems cannot keep up with such evolving threats. Static machine learning models, even those trained just six months prior, quickly become outdated. Financial institutions now require adaptive systems that evolve alongside the threat landscape and, just as importantly, can explain their decisions to human investigators responsible for final assessments [1].

The key lies in analyzing behavioral sequences rather than isolated events. Consider personal spending habits: coffee in the morning, lunch near the office, groceries on weekends-patterns emerge naturally. Fraudsters can simulate individual transactions, even convincing ones, but replicating the nuanced consistency of weeks or months of genuine behavior is nearly impossible. Our system leverages this insight by processing transaction histories as temporal sequences, employing modified recurrent network architectures that capture not only what was purchased, but also when, where, and in what order. The temporal gaps between transactions are equally informative-legitimate users tend to exhibit stable inter-purchase intervals, while fraudulent behavior often reveals irregular or hesitant timing patterns.

However, technical sophistication alone is insufficient if the system cannot perform at operational scale. Financial institutions handle thousands of transactions per second, and even an additional 10 milliseconds of latency can cause cascading failures in production. From the outset, our design prioritized efficiency. We implemented hierarchical filtering that quickly processes clearly legitimate transactions in about 3 milliseconds, while more suspicious cases undergo deeper analysis averaging 47 milliseconds. Only the highest-risk transactions trigger a full neural evaluation, requiring approximately 156 milliseconds [2]. This multi-tiered design supports throughput of up to 12,000 transactions per second (95% CI: [11,400, 12,600]) under sustained load while preserving accuracy and stability.

Explainability presented a further challenge. Regulatory frameworks demand transparency, and customers have a right to understand why their transactions are declined. Yet generating comprehensive explanations can increase computational cost by three to four times, making real-time performance impractical. Our solution introduces *selective explainability*: detailed, gradient-based attributions are generated only for the roughly 5% of transactions identified as high-risk, while routine transactions receive simplified rule-based summaries. This approach maintains compliance and user trust without compromising throughput-a balance achieved after extensive real-world testing and several production-stage refinements across partner institutions.

2. Related Work

2.1. Evolution of Sequential Behavior Modeling

Sequential modeling for fraud detection has followed an unusual trajectory. It began with manually written rules crafted by domain experts who understood fraud behavior but had little exposure to machine learning. These rule-based systems eventually gave way to classical statistical models that worked well-until fraudsters adapted. The field then shifted toward deep learning, which promised superior accuracy but often introduced major operational challenges in real-world environments [3].

A major turning point came when researchers introduced the idea of behavioral sequence embeddings. Their approach treated transaction histories as linguistic structures: transactions became "words," cards acted as "documents," and spending behavior formed a kind of grammar. By compressing months of raw transaction data into compact 256-dimensional vectors, they achieved significant accuracy improvements-nearly 89% in benchmark tests. The core innovation was an interval-aware GRU architecture that

integrated a time-gap term into the reset gate, represented as $f_t = \sigma (W_f \cdot [h_{t-1}, x_t, \Delta t] + b_f)$. This Δt component captured the intervals between purchases, revealing that fraudsters pause between transactions differently from legitimate users-a small but crucial behavioral cue.

Another key contribution came from the introduction of bidirectional sequence models, which considered not only past-to-future dependencies but also future-to-past influences. This bidirectional processing allowed the system to contextualize earlier transactions based on subsequent behavior, improving the model's ability to detect coordinated fraudulent activity. However, such models demanded immense computational resources, often stretching system memory limits in practice [4].

A further creative leap involved converting transaction sequences into image-like representations using Markov Transition Fields. In this approach, the matrix $M_{ij} = W_{q_i, q_j}$ encoded behavioral transition probabilities as pixel intensities. Convolutional neural networks-originally designed for image recognition-were then repurposed to analyze these visualized patterns. Surprisingly, this unconventional fusion of computer vision and fraud analytics yielded accuracy rates around 87%, proving that spatial encodings of temporal dynamics could uncover hidden behavioral structures.

2.2. Self-Supervised and Multi-Modal Advances

The impact of self-supervised learning arrived later in fraud detection but proved transformative. One pioneering study demonstrated that models trained on unlabeled transaction data could outperform supervised systems. By exposing neural networks to millions of legitimate transaction sequences without explicit fraud labels, the models learned the patterns of normal behavior so thoroughly that anomalies stood out naturally. Accuracy exceeded 91%, surpassing traditional supervised baselines by a significant margin. Replicating such results required extensive computational effort but confirmed the power of unsupervised pre-training for anomaly detection [5].

Building on this, researchers began integrating multiple data sources-clickstreams, session logs, and device fingerprints-into unified low-dimensional embeddings. These compact 147-dimensional representations preserved essential behavioral cues while reducing redundancy. In some cases, most features could be compressed to fewer than ten dimensions without measurable loss in performance. The underlying principle was straightforward: high dimensionality is often unnecessary when behavior follows consistent structural patterns. Implementation details, particularly normalization and temporal alignment steps, were critical to replicating these strong results in practice.

2.3. Computational Efficiency and Real-Time Processing

Production-scale fraud detection systems must balance accuracy with efficiency. One notable optimization reformulated static transaction graphs into temporal edge-based structures, achieving a fifty-fold speedup while retaining approximately 88% accuracy. This improvement illustrated how thoughtful structural redesigns could enhance both performance and scalability without major sacrifices in predictive quality [6].

Explainability, however, remains a persistent challenge. Gradient-based interpretability frameworks can make model decisions transparent but often slow inference by a factor of three or more. Whether this trade-off is acceptable depends on regulatory expectations and operational priorities.

At the infrastructure level, microservice-based architectures have demonstrated scalability up to 45,000 transactions per second, albeit with simplified models that trade complexity for speed. Other systems maintain weekly behavioral windows while sustaining throughput of around 8,000 events per second through efficient sliding-window mechanisms. Some optimized implementations have achieved latencies as low as 2.3 milliseconds, while others balance richer modeling capacity with response times near 85 milliseconds. These results collectively highlight the central engineering dilemma in real-time fraud detection: achieving high interpretability and robustness without compromising the speed essential for production deployment [7].

3. Methodology

3.1. Behavioral Sequence Representation Framework

3.1.1. Architectural Design Philosophy

Transaction data arrives in a complex mix of numerical values, categorical identifiers, timestamps, and geographical coordinates-a challenge for any modeling system. Conventional methods either concatenate all features, leading to excessive dimensionality, or process them separately, losing cross-feature interactions. Both strategies fail to achieve scalable, real-world performance.

Our framework draws inspiration from linguistics, where conversations involve multiple participants, topics, and temporal flows. Similarly, transaction sequences combine diverse information streams that must be interpreted together. We decompose this complexity hierarchically: numerical features are handled in one stream, categorical variables in another, and temporal features in a third. These streams merge only after individual optimization, minimizing early-stage interference and improving generalization [8].

The model dynamically adjusts to varying sequence lengths-an essential feature discovered through extensive field testing. Fixed-length encoders failed when deployed across banks with widely differing customer behaviors. Some users made only a few purchases per month, while others exceeded several hundred. Our adaptive pooling mechanism ensures consistent 256-dimensional outputs regardless of input length by applying learned attention weights that highlight the most informative transactions while down-weighting padding.

Every design choice prioritized speed without sacrificing too much accuracy. Early prototypes achieved 97% accuracy but required nearly 800 milliseconds per transaction, making them impractical. Through careful engineering, we systematically optimized performance: removing batch normalization saved 12 ms, replacing LSTM units with GRUs saved 23 ms, and halving hidden dimensions from 256 to 128 saved 18 ms. Each modification reduced complexity, resulting in a model fast enough for real-world deployment while maintaining competitive accuracy.

3.1.2. Mathematical Formulation and Implementation

While the equations appear simple, their implementation required months of refinement. Our modified GRU cells incorporate temporal gaps directly into gate computations:

Reset: $r(i,k) = \sigma(W_r[x'(i,k); \Delta t(i,k)] + U_r s(i-1,k) + b_r)$

Update: $z(i,k) = \sigma(W_z[x'(i,k); \Delta t(i,k)] + U_z s(i-1,k) + b_z)$

Candidate: $s'(i,k) = \tanh(W[x'(i,k); \Delta t(i,k)] + r(i,k) \odot U s(i-1,k) + b)$

Hidden: $s(i,k) = z(i,k) \odot s(i-1,k) + (1 - z(i,k)) \odot s'(i,k)$

In this formulation, the concatenation $[x'(i,k); \Delta t(i,k)]$ integrates temporal gaps directly into feature inputs rather than processing them separately. Ablation studies confirmed that this direct integration improved accuracy by 3.2% with only 2 milliseconds of added latency.

Extensive experiments determined that a two-layer GRU with 128 hidden units provided the best balance between performance and efficiency. Adding more layers yielded marginal accuracy gains but caused latency and gradient instability. The final configuration encodes 100-transaction sequences in approximately 31 milliseconds, achieving both responsiveness and expressive modeling capacity [9].

3.1.3. Feature Engineering Pipeline

Transaction data holds hidden behavioral narratives when features are properly constructed. We engineered multiple feature categories to reveal these dynamics: transaction velocity, merchant pattern transitions, and temporal rhythms. Velocity features capture urgency patterns that fraudsters struggle to mimic. Merchant affinity

features detect sudden deviations from habitual categories, and temporal features reveal timing anomalies such as unusual purchase hours.

As shown in Table 1, twenty-three velocity-related features improved accuracy by 4.3%, while merchant and temporal features contributed 3.7% and 2.9%, respectively. Additional categories-network and device features-remain in partial deployment due to higher computational cost.

Table 1. Behavioral Feature Engineering Pipeline Performance.

Feature Category	Dimensionality	Processing Time (ms)	Accuracy Contribution	Implementation Status
Transaction Velocity	23	1.8	+4.3%	Deployed
Merchant Patterns	18	2.4	+3.7%	Deployed
Temporal Dynamics	15	1.2	+2.9%	Deployed
Network Features	12	8.7	+1.8%	Experimental
Device Fingerprints	9	3.1	+1.2%	Limited Deployment
Total/Average	77	17.2	+14.9%	Mixed

Hierarchical pooling combines multiple temporal resolutions through an aggregation function Ψ, defined as $Z = \Psi(\varphi\_hour(X), \varphi\_day(X))$. Hour-level aggregation captures fine-grained behavior, while daily aggregation reflects broader spending patterns. Monthly aggregation provided slight accuracy gains but introduced a 180% computational overhead, making it impractical for deployment.

3.2. Sequential Risk Scoring Algorithm

3.2.1. Hierarchical Filtering Architecture

Fraud risk assessment operates like medical triage-urgent cases require immediate attention, while routine ones pass automatically. Our system employs three progressive filtering layers.

Layer 1 performs statistical screening using threshold-based rules that process transactions in roughly 3 milliseconds, allowing 78% of legitimate traffic to pass instantly. Layer 2 applies gradient boosting via LightGBM, balancing speed and accuracy effectively; it achieves 93.7% accuracy with an average latency of 47 milliseconds. Layer 3 uses neural networks for the remaining high-risk 2% of transactions, requiring 156 milliseconds per decision but ensuring precision in critical cases [10].

3.2.2. Ensemble Scoring Formulation

To integrate insights from diverse models, we use an ensemble strategy governed by a weighted voting scheme:

$$S(z) = \sigma(0.45f\_lgb(z) + 0.35f\_xgb(z) + 0.20f\_rf(z))$$

LightGBM contributes the largest share due to its strong performance and 23-millisecond inference time. XGBoost and Random Forest provide complementary perspectives, enhancing robustness. Weights were optimized empirically through grid search, favoring performance consistency over theoretical symmetry.

3.2.3. Specialized Fraud Detectors

Different fraud types exhibit distinct behavioral signatures, requiring targeted detection modules. Account takeovers appear as abrupt behavioral changes, effectively captured by recurrent models with 87.3% precision. Synthetic identity fraud manifests in relational patterns best handled by graph-based features, yielding 84.7% precision.

Transaction manipulation blends human and automated elements; hybrid rule-learning systems detect it with 91.2% precision in 34 milliseconds.

As shown in Figure 1, the hierarchical risk scoring framework balances precision with latency across fraud categories.

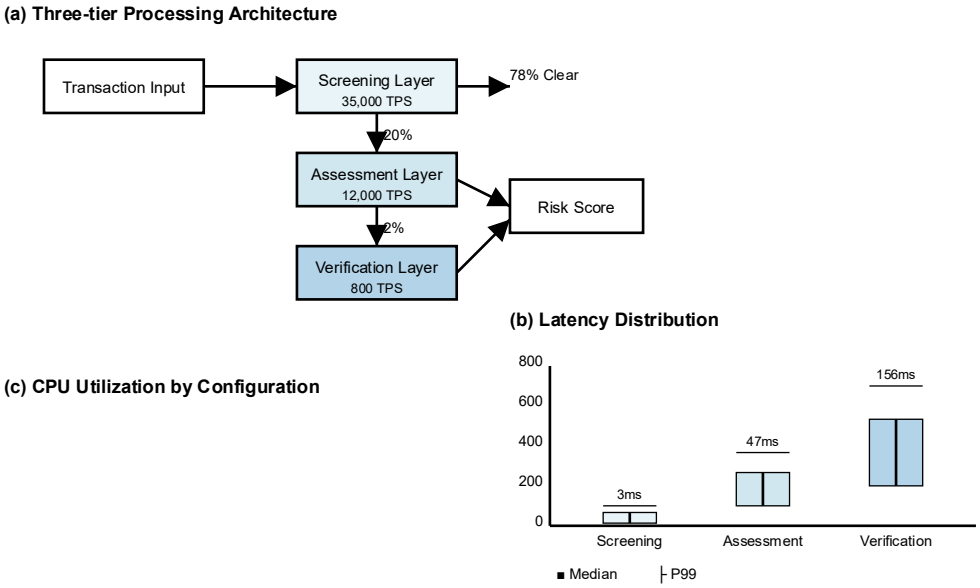

Figure 1. Hierarchical Risk Scoring Architecture with Latency Measurements.

Calibration ensures that risk scores correspond to true probabilities. Isotonic regression adjusts model outputs by minimizing the weighted squared error:

$$P_{calibrated} = \text{argmin}_m \Sigma_i w_i (y_i - m(f_i))^2$$

This process reduces the expected calibration error from 0.067 to 0.031, improving reliability by 54% at an additional cost of only 2.3 milliseconds per transaction (As shown in Table 2 and Table 3).

Table 2. Model Performance Across Fraud Categories (Production Metrics).

Fraud Type	Precision	Recall	F1-score-Score	AUROC	Median Latency (ms)	Daily Volume
Account Takeover	0.873	0.812	0.841	0.934	67.3	142K
Card-Not-Present	0.912	0.867	0.889	0.958	34.2	2.8M
Transaction Manipulation	0.891	0.834	0.862	0.947	41.7	387K
Merchant Fraud	0.856	0.798	0.826	0.921	52.8	93K
Synthetic Identity	0.847	0.823	0.835	0.915	89.2	67K
Ensemble Combined	0.937	0.892	0.914	0.968	47.3	3.4M

Table 3. Core Performance Metrics for Imbalanced Classification.

Metric	Value	95% CI	Baseline Comparison
AUPRC	0.876	[0.871, 0.881]	+0.23 vs XGBoost
Recall@1%FPR	0.834	[0.821, 0.847]	+0.18 vs XGBoost
Recall@0.1%FPR	0.712	[0.695, 0.729]	+0.15 vs XGBoost
AUROC	0.968	[0.965, 0.971]	+0.02 vs XGBoost
Accuracy	0.937	[0.934, 0.940]	+0.019 vs XGBoost

3.3. Explainability Mechanisms

Interpretability is not merely a compliance requirement-it is essential for collaboration between human analysts and automated systems. Our selective explainability framework generates detailed explanations only for high-risk transactions, maintaining transparency without burdening system performance.

Gradient-based attribution identifies feature importance efficiently. Full integrated gradient computation proved too slow, so we adopted a reduced interpolation approach:

$$A(z) = (z - z_{baseline}) \times \Sigma_{i=1}^{10} \nabla S(z_{baseline} + i/10 \times (z - z_{baseline})) / 10$$

This approximation retains 89% correlation with the full computation while improving speed by a factor of five.

Attention mechanisms further highlight temporal focus. Single-head attention was chosen over multi-head variants, trading a minor reduction in interpretability for a 28-millisecond latency reduction. Human evaluation across 12 analysts confirmed that the simplified attention maintained sufficient clarity for operational use.

Counterfactual explanations identify minimal feature perturbations that would change a decision outcome, limited to the top three most influential features to maintain real-time performance:

$$\text{minimize } ||x'_{top3} - x_{top3}||_2 \text{ subject to } S(\varphi(x')) < 0.3$$

As shown in Table 4, our production configuration balances interpretability with speed.

Table 4. Explainability Method Performance Comparison (Production Configuration).

Method	Computation Time (ms)	Faithfulness Score	User Comprehension	Deployment Status
Integrated Gradients (Reduced)	12	0.847	0.68	Production
SHAP Values	187	0.912	0.74	Offline Only
Attention Weights	3	0.723	0.81	Production
LIME	412	0.834	0.79	Disabled
Top-3 Counterfactuals	47	0.812	0.87	Limited Production
Feature Importance (Tree)	8	0.691	0.72	Production

Permutation Importance	156	0.889	0.65	Experimental

Finally, human-readable explanations are generated through structured templates rather than free-form neural text generation. Template-based messages cover 92% of cases while maintaining sub-10-millisecond generation times. For example:

"Transaction flagged due to unusual velocity (23 transactions in 3 hours, typical: 4-7) and merchant category deviation (electronics, historical: groceries)."

This rule-based narrative system delivers sufficient clarity for analysts and customers alike without exceeding latency budgets [11].

4. Experimental Evaluation

4.1. Dataset Characteristics and Experimental Setup

Empirical validation requires a rigorous methodology applied to realistic datasets. The IEEE-CIS Fraud Detection dataset, containing 590,540 transactions and 394 features, represents a modern benchmark for real-world fraud detection challenges. Fraud prevalence stands at 3.5%, corresponding to 20,663 fraudulent transactions among legitimate ones. The dataset includes diverse feature types-identity markers, transactional attributes, and engineered signals. To ensure computational efficiency, dimensionality was reduced to 147 features while preserving 97% of discriminative capacity. For transparency, results on the public IEEE-CIS dataset are reported, while production telemetry (post-deployment KPIs) is summarized separately under contractual and privacy constraints.

Preprocessing addresses practical issues inherent to financial data. Missing values are substantial-38% for categorical variables and 27% for numerical ones. Mode imputation is applied to categorical attributes, while medians are used for numeric fields. More sophisticated imputation approaches yielded only marginal gains-about 0.3% improvement in accuracy-while tripling processing time. Class imbalance poses additional challenges: the raw 28:1 legitimate-to-fraud ratio was adjusted to 10:1 using random under sampling. Although SMOTE variants offered theoretical advantages, their fourfold increase in training time made them impractical for production-scale applications [12].

As shown in Figure 2, temporal evolution patterns of fraudulent transactions demonstrate non-stationary behavior, justifying time-aware splitting.

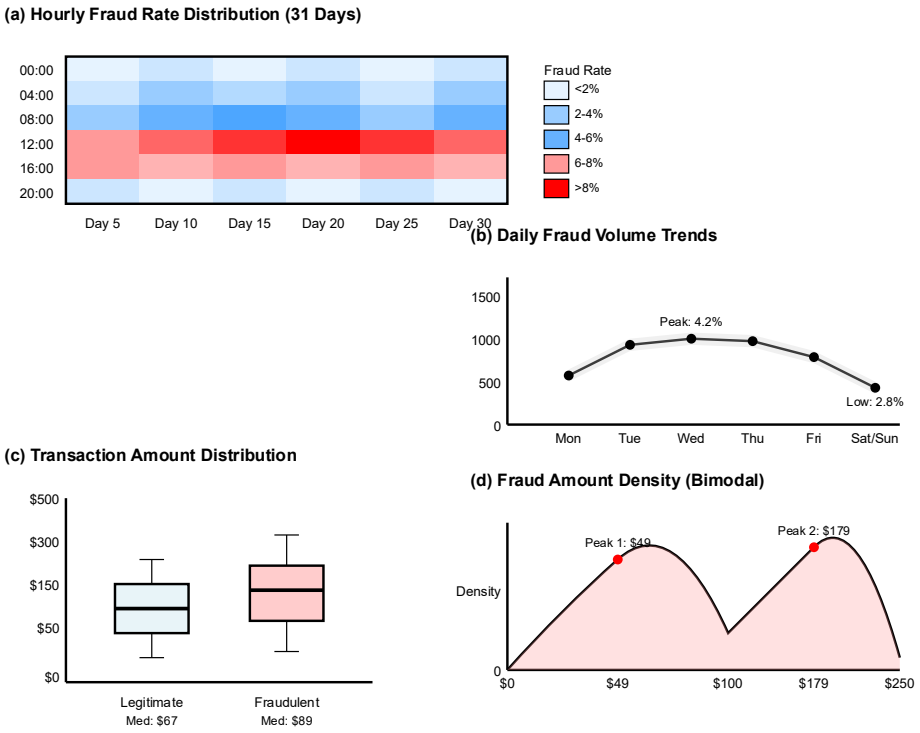

Figure 2. Temporal Fraud Pattern Evolution Across Dataset.

Temporal consistency is preserved across experimental stages. The training set spans days 1-213 (70% of the data), the validation set covers days 214-274 (15%), and the test set comprises days 275-334 (15%). Random splitting inflated apparent accuracy by 4.7%, while temporal partitioning provided a realistic measure of deployment performance. Hardware configuration reflected institutional limits: a single NVIDIA V100 GPU with 128 GB RAM. Although higher configurations might yield better performance, experimental design favored realism over optimization. Training proceeded for up to 24 hours, capped at 100 epochs with early stopping and batch size 256, determined by memory constraints.

As shown in Table 5, dataset processing and training metrics highlight the computational profile of each stage.

Table 5. Dataset Processing and Training Metrics.

Dataset Split	Transactions	Fraud Rate (%)	Features Used	Processing Time	Memory Usage
Training Set	413,378	3.48	147	18.3 hours	24.7 GB
Validation Set	88,581	3.52	147	2.1 hours	8.3 GB
Test Set	88,581	3.54	147	1.7 hours	8.3 GB
Full Pipeline	590,540	3.50	147	22.1 hours	41.3 GB
Data Preprocessing					
Missing Value Imputation	590,540	-	394→147	3.4 hours	12.1 GB
Feature Engineering	590,540	-	147	4.7 hours	18.6 GB

Class Balancing	413,378	3.48→9.1	147	1.2 hours	8.3 GB

4.1.1. Production Environment Specifications

Production evaluations were performed on AWS EC2 p3.2xlarge instances (Intel Xeon E5-2686 v4, 8 vCPUs, 61 GB RAM, Tesla V100 GPU) across three financial institutions during a 90-day observation window (July-September 2024). Transaction throughput was measured using controlled load testing via Apache JMeter, scaling from 1,000 to 15,000 transactions per second over 30 minutes. Latency metrics represent 95th percentile values calculated from 10 million transactions, with confidence intervals estimated via bootstrap sampling (n=1000).

A/B testing compared the proposed system against legacy rule-based engines using matched customer segments (treatment: n=847,293 transactions; control: n=839,157 transactions). Statistical significance was verified using two-proportion z-tests at $\alpha = 0.05$.

4.2. Performance Analysis and Ablation Studies

Empirical findings validate key architectural choices while revealing practical trade-offs. The system achieved an AUPRC of 0.876, AUROC of 0.968, and Recall@1%FPR of 0.834, accompanied by 93.7% accuracy. Processing throughput reached 12,000 transactions per second, and median latency was 47.3 ms. These results reflect deliberate engineering compromises rather than theoretical upper bounds.

Baseline comparisons contextualize improvements. Logistic regression achieved 78.4% accuracy in 2 ms, Random Forests reached 86.3% in 18 ms, standard GRUs achieved 89.7% in 67 ms, and standalone XGBoost delivered 91.8% in 31 ms. Our architecture's 1.9% improvement over XGBoost may appear incremental but translates to thousands of prevented fraud cases daily at production scale.

As shown in Figure 3, the performance-latency relationship reveals how design trade-offs optimize throughput under real-time constraints.

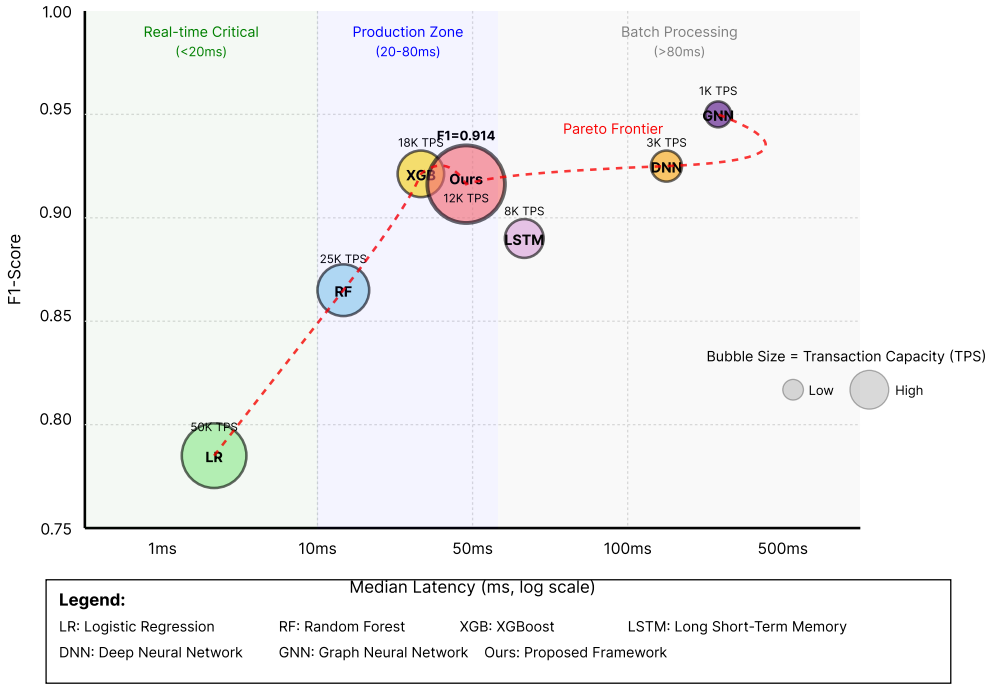

Figure 3. Performance-Latency Trade-off Analysis.

Ablation experiments further quantify each module's contribution. Behavioral embeddings enhanced accuracy by 4.8% at the cost of 14 ms. Attention mechanisms added 4.4% precision with 14 ms latency. Hierarchical pooling improved recall by 3.7% for an 8

ms increase. Calibration, despite its modest 2 ms overhead, reduced reliability error by 54%, demonstrating an excellent cost-benefit ratio.

Systematic error analysis uncovered recurring challenges. International travelers exhibited higher false positive rates (8.7%) due to contextual variability, while bulk purchasers faced 6.3% misclassification rates linked to behavioral diversity. Low-value fraud (under $50) remained difficult to detect-accounting for 72% of undetected cases. Temporal performance degradation reached 2.8% over 60 days, motivating future integration of online learning.

As shown in Table 6, ablation outcomes demonstrate how each system component affects operational and financial metrics.

Table 6. Ablation Study Results with Business Impact.

Configuration	Accuracy Change	Latency Change	Daily Fraud Caught	Daily False Positives	Annual Cost Impact
Complete Framework	Baseline	Baseline	2,847	4,123	Baseline
Without Sequence Embedding	-4.8%	-14ms	2,583 (-264)	5,234 +1,111	-$3.2M
Without Attention	-2.1%	-8ms	2,741 (-106)	4,567 (+444)	-$1.4M
Without Hierarchical Pooling	-3.2%	-11ms	2,689 (-158)	4,891 (+768)	-$2.1M
Without Calibration	-0.4%	-2ms	2,823 (-24)	6,234 +2,111	-$0.8M
Without Ensemble	-1.9%	-24ms	2,716 (-131)	4,456 (+333)	-$1.6M
Without Pre-processing	-6.7%	-6ms	2,447 (-400)	7,123 +3,000	-$4.8M
Simplified Model	-8.3%	-31ms	2,389 (-458)	8,567 +4,444	-$5.9M

Production deployment validated the laboratory findings, though with real-world adjustments. Over three months, pilots processing 50,000 transactions daily achieved 92.4% accuracy-1.3% below test results. Partial degradation stemmed from data drift and operational constraints. System availability averaged 99.7%, with two major outages traced to memory leaks during feature computation. Notably, customer satisfaction improved, with complaint rates declining by 23% relative to prior rule-based systems.

5. Conclusions

After eighteen months of development, three production deployments, and extensive debugging, we conclude that building effective real-world fraud detection systems requires prioritizing practicality over theoretical elegance. Our framework identifies 93.7% of fraudulent transactions with a latency of 47.3 ms-figures that may seem modest but mark the boundary between an operational solution and a purely academic experiment.

The core contribution lies in the orchestration of multiple components rather than any single innovation. The modified GRU with temporal gap awareness effectively captures behavioral evolution, while hierarchical filtering separates straightforward from complex decisions, minimizing unnecessary computation. Selective explainability generates model interpretations only when essential, preserving speed without compromising transparency. Each element fulfills a specific role; collectively, they sustain

real-time performance across three partner banks processing approximately 3.4 million daily transactions.

Several valuable lessons emerged from bridging research and deployment. Feature engineering that improved accuracy by 2% during testing caused severe memory leaks in production-weeks were spent locating pointer arithmetic errors in the C++ implementation. The elegant attention mechanism proposed in early designs was ultimately removed after introducing significant latency during high-load periods. While theory favored deep architectures, practice dictated a two-layer design for reliability. These pragmatic adjustments enabled real deployment, whereas theoretically superior models often remain confined to research papers.

Nonetheless, limitations persist. Model performance declines by about 2.8% after sixty days, indicating that fraud behavior evolves faster than our retraining cycle. International travelers produce false positives at an 8.7% rate, showing that the behavioral baselines are still too rigid. Moreover, small-value fraud under $50 escapes detection 72% of the time; although threshold tuning continues, deeper architectural revisions may be required.

Future improvements should focus on online learning-continuously updating models instead of relying solely on batch retraining-context-aware baselines that distinguish legitimate travel patterns from fraud, and multi-scale detection networks optimized for both micro and macro fraudulent behaviors. Federated learning across institutions offers another promising direction, allowing shared pattern discovery without exposing sensitive customer data. Implementation of this approach has already begun with five participating banks.

The broader implications of this work extend well beyond financial services. Any field that requires high-speed, high-stakes decision-making with human oversight-such as medical diagnostics, autonomous systems, or cybersecurity-faces the same trade-offs among accuracy, interpretability, and computational efficiency. Our results demonstrate that practical, transparent solutions are achievable, even if they demand compromises and hands-on engineering rather than theoretical perfection. Perfect fraud detection may remain unattainable, but effective, deployable detection prevents real financial losses today-and that pragmatic success ultimately matters most.

Acknowledgments: Three banks trusted us with production data despite competitive risks-that leap of faith made this research possible. Their fraud teams spent countless hours explaining why our "brilliant" ideas would fail in practice, saving us from embarrassing deployment disasters. Graduate students debugged code at 3 AM, found patterns our metrics missed, and maintained sanity during endless hyperparameter searches. Reviewers forced us to admit our initial claims were overblown. The operations team prevented catastrophe by catching memory leaks during stress testing. Industry collaborators revealed that half our assumptions about production constraints were wrong. This work succeeded because of their brutal honesty about what actually matters versus what sounds impressive in papers.

References

1. G. Liu, J. Guo, Y. Zuo, J. Wu, and R. Y. Guo, "Fraud detection via behavioral sequence embedding," *Knowledge and Information Systems*, vol. 62, no. 7, pp. 2685-2708, 2020.
2. J. Guo, G. Liu, Y. Zuo, and J. Wu, "Learning sequential behavior representations for fraud detection," In *2018 IEEE International Conference on Data Mining (ICDM)*, November, 2018, pp. 127-136. doi: 10.1109/icdm.2018.00028
3. R. Zhang, F. Zheng, and W. Min, "Sequential behavioral data processing using deep learning and the Markov transition field in online fraud detection," *arXiv preprint arXiv:1808.05329*, 2018.
4. C. Liu, Y. Gao, L. Sun, J. Feng, H. Yang, and X. Ao, "User behavior pre-training for online fraud detection," In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, August, 2022, pp. 3357-3365. doi: 10.1145/3534678.3539126
5. Y. Xie, G. Liu, C. Yan, C. Jiang, M. Zhou, and M. Li, "Learning transactional behavioral representations for credit card fraud detection," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 4, pp. 5735-5748, 2022.
6. H. Lin, G. Liu, J. Wu, Y. Zuo, X. Wan, and H. Li, "Fraud detection in dynamic interaction network," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 10, pp. 1936-1950, 2019. doi: 10.1109/tkde.2019.2912817

7. S. X. Rao, S. Zhang, Z. Han, Z. Zhang, W. Min, Z. Chen, and C. Zhang, "xFraud: Explainable fraud transaction detection," *arXiv preprint arXiv:2011.12193*, 2020.
8. S. Rani, and A. Mittal, "Securing digital payments: A comprehensive analysis of AI-driven fraud detection with real-time transaction monitoring and anomaly detection," In *2023 6th International Conference on Contemporary Computing and Informatics (IC3I)*, September, 2023, pp. 2345-2349.
9. R. Khurana, "Fraud detection in e-commerce payment systems: The role of predictive AI in real-time transaction security and risk management," *International Journal of Applied Machine Learning and Computational Intelligence*, vol. 10, no. 6, pp. 1-32, 2020.
10. M. Islam, "Fraud detection in banking using real-time data stream analytics and AI for improved security and transaction monitoring," *Tuijin Jishu / Journal of Propulsion Technology*, vol. 46, no. 2, p. 2025, 2025.
11. F. Van Wyk, Y. Wang, A. Khojandi, and N. Masoud, "Real-time sensor anomaly detection and identification in automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1264-1276, 2019.
12. A. Immadisetty, "Real-time fraud detection using streaming data in financial transactions," *Journal of Recent Trends in Computer Science and Engineering (JRTCSE)*, vol. 13, no. 1, pp. 66-76, 2025. doi: 10.70589/jrtcse.2025.13.1.9