Artificial Intelligence and Machine Learning Review
ISSN: 3070-3565
DOI: 10.69987/AIMLR.2026.70301

Comparing Prompting Strategies and Backbone Choices for LLM Agents on Tabular Credit-Card Fraud Detection: An Empirical Study

Yanhuan Chen
Master of Engineering, Dartmouth College, NH, USA
Yanhuan_Chen_v2@yahoo.com

Keywords

LLM agents; credit-card fraud detection; prompting strategies; empirical evaluation

Abstract

Large language model (LLM) agents are increasingly proposed for financial decision tasks, yet their behaviour on tabular fraud detection has not been measured under a controlled protocol. This paper reports a head-to-head empirical comparison of four prompting strategies (zero-shot, few-shot, chain-of-thought, and ReAct-style tool use) across four LLM backbones (GPT-4o-mini, Claude-3.5-Haiku, Llama-3.1-8B-Instruct, and Qwen-2.5-7B-Instruct) on two public datasets: the ULB credit-card corpus with anonymised principal-component features and the Bank Account Fraud (BAF-Base) suite with human-readable attributes. All 32 agent configurations are benchmarked against logistic regression, random forest, XGBoost, and LightGBM under identical temporal splits. Gradient-boosted trees retain a clear advantage: XGBoost reaches an AUC-PR of 0.847 on ULB and 0.196 on BAF, while the strongest agent configuration attains 0.412 and 0.153. The accuracy gap narrows substantially when features carry semantic meaning, and tool-augmented prompting yields its largest gain (+0.040 AUC-PR over chain-of-thought) on anonymised features. Inference cost remains three to four orders of magnitude higher for agents. A taxonomy of 400 manually coded errors indicates that agents most often miss fraud expressed through combinations of weak signals. Practical guidance for hybrid deployment is derived from these measurements.

1. Introduction

1.1. Background and Problem Statement

Payment card fraud remains a persistent economic burden: industry estimates place worldwide card fraud losses at 33.41 billion USD in 2024 against 51.92 trillion USD of purchase volume, with card-not-present abuse accounting for roughly three quarters of United States card fraud (Nilson Report, 2026). Production detection pipelines are still dominated by rule engines and gradient-boosted tree classifiers, which deliver strong ranking accuracy at negligible inference cost yet require continual feature engineering, degrade under distribution drift, and emit scores that fraud analysts cannot easily interrogate. The rapid progress of finance-oriented language models, from the 50-billion-parameter BloombergGPT trained on a mixed corpus of proprietary and public financial text [1] to the instruction-tuned FinMA released with the PIXIU benchmark [2], has prompted speculation that LLM-based agents could complement or replace parts of this pipeline. Benchmark coverage has not kept pace with this speculation. FinBen, the broadest financial evaluation suite published at a top venue, spans 24 tasks across eight capability areas, and none of them addresses transaction fraud detection [3]. Recent detection research at KDD and AAAI has instead concentrated on coupling LLM-derived signals with graph neural networks, exemplified by the FLAG detector [4]

and by hypergraph contrastive learning guided by frozen LLMs [5]. Both lines introduce new architectures; neither answers the more basic question practitioners face: what detection quality, and at what monetary cost, do off-the-shelf LLM agents deliver on standard public tabular fraud benchmarks? The cost dimension is not incidental. A mid-sized issuer scoring tens of millions of transactions per day operates under per-decision budgets measured in fractions of a cent, and any technology whose inference cost exceeds that budget by orders of magnitude can enter the pipeline only in a restricted role. Detection quality and unit economics must be measured together for the comparison to inform deployment decisions.

1.2. Motivation and Contributions

A. Why an Empirical Evaluation Rather Than a New Architecture

Claims about LLM performance on fraud tasks circulate mainly through preprints that rely on proprietary transaction data, undocumented prompts, or single-backbone experiments, leaving results difficult to reproduce or compare. A controlled comparison on public data closes this gap more directly than another architectural proposal would. This study fixes the datasets, the temporal splits, the serialization templates, and the decoding parameters in advance, and then varies exactly two factors, the prompting strategy and the backbone, so that observed differences can be attributed to those factors alone. The design deliberately excludes fine-tuning: practitioners evaluating agent adoption typically begin with in-context methods, and a fine-tuning-free protocol keeps every configuration reproducible from the published prompts.

B. Contributions

The paper makes three contributions. It reports, to our knowledge, the first systematic public grid of four prompting strategies crossed with four backbones, totalling 32 agent configurations evaluated against four classical baselines on two public datasets under identical splits. It quantifies the cost-latency-accuracy trade-off in USD per thousand transactions, a dimension absent from prior LLM-fraud studies. It supplies a manually coded taxonomy of 400 agent errors with inter-annotator agreement, identifying the failure modes that any deployment would need to mitigate. No new detector, dataset, or training procedure is introduced.

2. Related Work

2.1. LLM Agents and Tabular-Data Reasoning

A. Prompting and Agent Paradigms

Chain-of-thought prompting demonstrated that intermediate reasoning steps elicited in context substantially improve multi-step task accuracy in sufficiently large models [6]. ReAct extended this idea by interleaving reasoning traces with environment actions, allowing an agent to consult external resources before committing to an answer [7]. Subsequent work added verbal self-reflection across episodes [8] and deliberate tree-structured search over reasoning branches [9]. The present study evaluates chain-of-thought and ReAct alongside zero-shot and few-shot prompting because both remain tractable when scoring tens of thousands of transactions; reflection and tree search multiply the token budget per decision by factors that are prohibitive at this scale, and they are treated as out of scope. The selection also spans the relevant capability axis: zero-shot isolates raw priors, few-shot adds labelled context, chain-of-thought adds explicit reasoning, and ReAct adds external computation, so each step up the ladder corresponds to one identifiable mechanism whose marginal contribution the ablations in Section 4.1 can measure in isolation.

B. Tabular Classification with Language Models

TabLLM established row-serialization as the standard interface between tabular records and language models, reporting that serialized few-shot classification is competitive with gradient-boosted trees up to roughly 256 labelled examples and superior below that regime [10]. That finding concerns small-sample settings with fine-tuned models, and follow-up analyses have repeatedly observed sensitivity to serialization order, exemplar choice, and verbalisation of numeric values, all of which are held fixed in this study. The regime studied here differs on both axes: class imbalance below two percent, evaluation slices of fifty thousand rows or more, and strictly in-context adaptation.

Parameter-efficient financial adaptation pipelines such as FinGPT [11] offer an orthogonal lever that is intentionally excluded so that the comparison isolates prompting effects.

2.2. Financial Fraud Detection Methods

Classical fraud detection on tabular transaction data is dominated by regularised linear models and tree ensembles, whose strength derives from learning high-order feature interactions directly from labelled history; these methods serve as the calibrated baselines in Section 3 because they represent what an LLM agent would actually displace in production. A separate research thread models transactions as graphs. CARE-GNN combats camouflaged fraudsters through reinforcement-guided neighbour selection on multi-relation graphs [12], while PC-GNN addresses class imbalance with label-balanced subgraph sampling [13]. Provable extensions to directed multigraphs, including port numbering and reverse message passing, set the current standard on synthetic anti-money-laundering benchmarks [14]. These detectors presume relational structure that the purely tabular datasets used here do not expose, so they are discussed for positioning rather than re-implemented. Prior LLM-fraud work likewise leaves the present question open from the opposite direction: published detectors embed the language model inside a larger trained pipeline, making it impossible to attribute reported gains to the model itself rather than to the surrounding architecture. Against this backdrop, the niche occupied by the present study is precise: agent-only inference, no gradient updates, public tabular data, and full disclosure of prompts, splits, and costs.

3. Experimental Setup

3.1. Task Formulation

Each instance is a single transaction or account application to be classified as fraudulent or legitimate. Agents receive one serialized record per query and must return an integer risk score between 0 and 100, which serves as the ranking score for threshold-free metrics; classical baselines emit calibrated probabilities. Severe class imbalance makes area under the precision-recall curve (AUC-PR) the primary metric, complemented by ROC-AUC and recall at a fixed five percent false-positive rate (R@5%FPR); ROC-AUC alone is reported for comparability with prior work but is known to overstate performance when positives are rare, and R@5%FPR reflects the alert-budget constraint under which review teams operate. Uncertainty for agent configurations is quantified with 95 percent bootstrap confidence intervals over 1,000 test-set resamples, yielding half-widths of approximately ±0.021 AUC-PR on ULB and ±0.011 on BAF for the strongest configuration. Operating thresholds for the error analysis in Section 4.3 are set on validation data at the same five percent false-positive rate. Monetary cost in USD per one thousand scored transactions and median per-transaction latency complete the measurement set. The evaluation protocol adapts the controlled multi-configuration philosophy of agent benchmarking [15] to a binary detection task with temporal splits, so that no model sees future data during training or exemplar selection.

3.2. Datasets

Table 1 summarises both corpora. The two datasets are deliberately complementary: one hides feature semantics behind a principal-component transformation, the other exposes human-readable attributes, allowing a direct test of whether LLM agents benefit from semantic feature names.

Table 1. Dataset statistics and temporal splits. Source: official dataset documentation and accompanying publications.

Property	ULB Credit Card	BAF-Base
Records	284,807	1,000,000
Positive (fraud)	492 (0.172%)	11,029 (1.10%)
Features	30	30
Feature semantics	anonymised (V1-V28)	human-readable

Train split	199,364 (70%)	months 0-5: 750,403
Validation split	28,481 (10%)	month 6: 124,768
Test split	56,962 (20%)	month 7: 124,829
Evaluation slice	full test (98 fraud)	50,000 stratified (551 fraud)
License	ODbL	CC BY-NC-ND 4.0

A. ULB Credit Card Fraud

The ULB corpus contains 284,807 card transactions collected over two days in September 2013 from European cardholders, of which 492 (0.172 percent) are fraudulent; apart from elapsed time and amount, the 28 remaining features are anonymised principal components [16]. A chronological 70/10/20 split yields 199,364 training, 28,481 validation, and 56,962 test transactions, the test portion containing 98 fraud cases. Records are serialized as "Time=t s, Amount=a USD, V1=v1, ..., V28=v28" with values rounded to three decimals, and all few-shot exemplars are drawn exclusively from the training split with a fixed random seed so that no evaluation record, or any record occurring after it in time, can appear in a prompt. Because component names carry no meaning, chain-of-thought exemplars additionally summarise each record by its three largest-magnitude components expressed as z-scores against training statistics, giving the agent a numerically grounded anchor without leaking labels.

B. Bank Account Fraud

BAF-Base is the base variant of the Bank Account Fraud suite: one million synthetic but realistically distributed account-opening applications spanning eight months, generated from a model trained on anonymised production data and released with documented bias and drift properties [17]. Its 30 attributes are interpretable, covering income decile, employment status, name-email similarity, days since a previous application, and device-level counters, with 11,029 positives (1.10 percent). The suite includes protected attributes intended for fairness research; these are retained for the classical baselines and likewise serialized for the agents so that both method families observe an identical feature set. Their inclusion serves benchmark parity only and should not be read as endorsing protected-attribute use in production, where fairness auditing and protected-attribute ablation, both left outside the scope of this comparison, would be required. Months 0 through 5 (750,403 rows) form the training split, month 6 (124,768 rows) the validation split, and month 7 (124,829 rows) the test split. Scoring every configuration on the full test month is unnecessary for stable metric estimates and would multiply API expenditure, so all methods, classical and agent alike, are evaluated on an identical 50,000-row stratified slice of month 7 containing 551 fraud cases.

3.3. Methods Compared

A. Classical Baselines

Four classical detectors are trained on raw features with identical splits: logistic regression with L2 regularisation, a random forest of 500 trees with maximum depth 12, XGBoost with 500 boosting rounds, learning rate 0.05, and class-ratio-scaled positive weight, and LightGBM with matched capacity. Hyperparameters are tuned on the validation split only, through a grid over tree depth, learning rate, and class-weight settings comprising 48 candidate configurations per method, and each final configuration is trained with three random seeds; the standard deviation of AUC-PR across seeds never exceeds 0.004, and reported values are seed means. No resampling or synthetic minority oversampling is applied, since class-weighted objectives achieved equal or better validation AUC-PR in preliminary runs while keeping the pipeline simpler to reproduce. Baseline numbers are regenerated from scratch rather than imported from prior publications, keeping preprocessing identical to the agent pipeline and avoiding split mismatches that frequently inflate cross-paper comparisons.

B. LLM-Agent Configurations

Four prompting strategies are crossed with four backbones, giving 32 configurations per the design in Table 2. Every prompt shares a fixed three-part anatomy: a task preamble describing the detection objective and the meaning of the

score scale, a feature glossary listing each attribute with its unit and training-set range, and the serialized target record followed by the output schema. Zero-shot prompting uses this anatomy alone. Few-shot prompting prepends eight class-balanced exemplars drawn from the training split. Chain-of-thought prompting augments four of those exemplars with worked rationales that walk from feature values to a risk judgement before the score. ReAct-style prompting grants two tools, lookup_feature_stats(name), which returns the training mean and standard deviation of a feature, and compute(expression), a bounded arithmetic evaluator; the agent interleaves up to four thought-action-observation rounds before scoring, following evidence that models can learn when external calls are worthwhile [18]. All backbones run at temperature 0 with output caps of 256 tokens (1,024 for ReAct). Scores are extracted from a mandated final line of the form "SCORE: n"; a malformed line triggers at most one retry, an event observed in 0.6 percent of queries, and a second failure assigns the median training score, which occurred in fewer than 0.02 percent of queries. Serialization templates are byte-identical across strategies so that observed differences stem from the strategy itself.

Table 2. Evaluated backbones and decoding configuration. Source: provider documentation as of June 2026.

Backbone	Params	Access	Temp.	Output cap	Strategies
GPT-4o-mini	undisclosed	API	0	256 / 1,024	ZS, FS, CoT, ReAct
Claude-3.5-Haiku	undisclosed	API	0	256 / 1,024	ZS, FS, CoT, ReAct
Llama-3.1-8B-Instruct	8B	local (A100)	0	256 / 1,024	ZS, FS, CoT, ReAct
Qwen-2.5-7B-Instruct	7B	local (A100)	0	256 / 1,024	ZS, FS, CoT, ReAct

4. Results and Analysis

4.1. Headline Detection Performance

A. Aggregate Comparison

Table 3 reports all 20 detectors per dataset, and Figure 1 visualises the AUC-PR landscape. Gradient-boosted trees lead on both corpora: XGBoost attains an AUC-PR of 0.847 on ULB and 0.196 on BAF, with LightGBM close behind at 0.839 and 0.191. The strongest agent configuration, GPT-4o-mini with ReAct prompting, reaches 0.412 on ULB and 0.153 on BAF. The absolute deficit to XGBoost shrinks from 0.435 on anonymised features to 0.043 on semantic features, a pattern consistent with the interpretation that language models import useful priors only when feature names denote real-world quantities. Closed-weight backbones dominate the open 7-8B models by a wide margin on both datasets, and on ULB even the best open configuration (Llama-3.1-8B with ReAct, 0.226) trails logistic regression by more than 0.49 AUC-PR. Per-strategy improvements stack consistently rather than interacting: no backbone shows a strategy ordering different from the aggregate ordering, which simplifies practical configuration choices to picking the strongest affordable backbone and the richest affordable strategy.

Table 3. Detection performance of all configurations. AUC-PR, ROC-AUC, and recall at 5% FPR on the evaluation slices defined in Table 1. Classical baselines are means over three seeds (std ≤ 0.004 AUC-PR). Source: this study.

Method	ULB AUC-PR	ULB ROC-AUC	ULB R@5%FPR	BAF AUC-PR	BAF ROC-AUC	BAF R@5%FPR
Logistic regression	0.718	0.972	0.816	0.131	0.871	0.448

Random forest	0.821	0.957	0.857	0.172	0.883	0.512
XGBoost	0.847	0.978	0.898	0.196	0.892	0.563
LightGBM	0.839	0.976	0.888	0.191	0.890	0.551
GPT-4o-mini, zero-shot	0.231	0.872	0.398	0.094	0.812	0.331
GPT-4o-mini, few-shot	0.287	0.894	0.452	0.117	0.834	0.382
GPT-4o-mini, CoT	0.334	0.911	0.541	0.148	0.861	0.466
GPT-4o-mini, ReAct	0.412	0.941	0.622	0.153	0.864	0.471
Claude-3.5-Haiku, zero-shot	0.224	0.868	0.387	0.091	0.808	0.322
Claude-3.5-Haiku, few-shot	0.279	0.889	0.441	0.112	0.829	0.371
Claude-3.5-Haiku, CoT	0.341	0.917	0.553	0.139	0.855	0.447
Claude-3.5-Haiku, ReAct	0.398	0.936	0.609	0.146	0.859	0.458
Llama-3.1-8B, zero-shot	0.142	0.801	0.264	0.063	0.752	0.231
Llama-3.1-8B, few-shot	0.183	0.826	0.312	0.078	0.771	0.262
Llama-3.1-8B, CoT	0.207	0.842	0.345	0.092	0.788	0.297
Llama-3.1-8B, ReAct	0.226	0.851	0.371	0.095	0.791	0.303
Qwen-2.5-7B, zero-shot	0.137	0.794	0.255	0.066	0.756	0.238
Qwen-2.5-7B, few-shot	0.176	0.819	0.301	0.081	0.774	0.269
Qwen-2.5-7B, CoT	0.214	0.845	0.352	0.089	0.785	0.291
Qwen-2.5-7B, ReAct	0.219	0.848	0.360	0.091	0.787	0.295

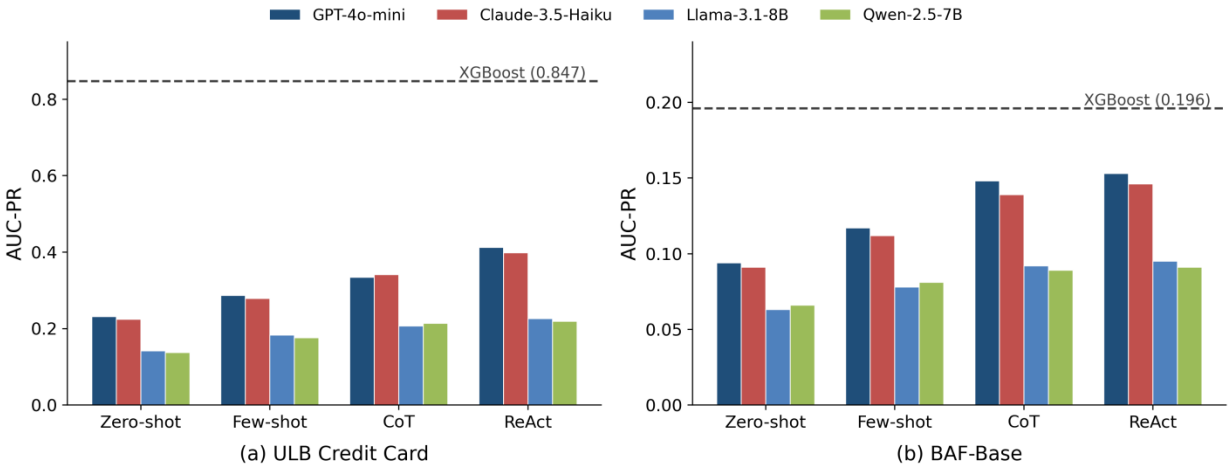

Figure 1. Detection Accuracy of Prompting Strategies Across Backbones

AUC-PR of all 16 agent configurations grouped by prompting strategy on (a) the ULB credit-card corpus and (b) BAF-Base, with the XGBoost reference level marked by a dashed line (0.847 and 0.196, respectively). Accuracy increases monotonically from zero-shot to ReAct for every backbone on ULB, where the best agent configuration (GPT-4o-mini with ReAct, 0.412) still falls 0.435 below the reference. On BAF-Base the curves flatten between chain-of-thought and ReAct, and the best agent configuration (0.153) approaches the reference within 0.043. Closed-weight backbones outperform the open 7-8B backbones in every strategy on both datasets.

B. Strategy and Backbone Ablations

Marginal means isolate each factor. Averaged over backbones, AUC-PR on ULB rises from 0.184 (zero-shot) through 0.231 (few-shot) and 0.274 (chain-of-thought) to 0.314 (ReAct); on BAF the corresponding means are 0.079, 0.097, 0.117, and 0.121. The ReAct increment over chain-of-thought is +0.040 on ULB yet only +0.004 on BAF, indicating that arithmetic tool calls compensate for the absence of feature semantics but add little once names already convey meaning. Averaged over strategies, backbone means on ULB are 0.316 for GPT-4o-mini, 0.311 for Claude-3.5-Haiku, 0.190 for Llama-3.1-8B, and 0.187 for Qwen-2.5-7B, with the same ordering on BAF (0.128, 0.122, 0.082, 0.082). The two closed backbones are statistically indistinguishable from each other at this sample size, their bootstrap confidence intervals overlapping on both datasets, while the gap separating them from the open models is roughly 0.12-0.13 AUC-PR on ULB, exceeding the entire effect of strategy choice within the open models.

4.2. Cost, Latency, and Throughput

Table 4 and Figure 2 quantify the economic dimension. LightGBM scores one thousand transactions for 0.0003 USD at 0.02 milliseconds per record. The cheapest useful agent configuration, GPT-4o-mini zero-shot, costs 0.07 USD per thousand at 0.6 seconds median latency, and the most accurate configuration, GPT-4o-mini ReAct, costs 0.83 USD per thousand at 5.4 seconds. The cost premium over LightGBM spans three to four orders of magnitude and the latency premium roughly five, while the accuracy delivered remains lower. The Pareto frontier in Figure 2 contains only classical detectors on ULB; on BAF the frontier is again headed by the tree ensembles, with GPT-4o-mini chain-of-thought (0.148 at 0.31 USD) the closest agent point. These figures argue against full-stream agent scoring and for a cascade in which a tree model handles the stream and an agent is invoked only on the narrow band of alerts routed to human review. The arithmetic is straightforward: at a five percent alert rate on one million daily transactions, full-stream ReAct scoring with GPT-4o-mini would cost roughly 830 USD per day, while restricting the agent to the 50,000 alerted cases reduces this to about 41.50 USD, a sum immaterial relative to the analyst time those cases already consume.

Table 4. Inference cost and latency per configuration. Cost in USD per 1,000 scored transactions at June 2026 prices (API list prices for closed backbones; A100 rental at 1.89 USD per hour for open backbones); latency is the median per transaction. Source: this study.

Configuration	Cost (USD/1k)	Median latency (s)
LightGBM	0.0003	0.00002
GPT-4o-mini, zero-shot	0.07	0.6
GPT-4o-mini, CoT	0.31	2.1
GPT-4o-mini, ReAct	0.83	5.4
Claude-3.5-Haiku, zero-shot	0.21	0.7
Claude-3.5-Haiku, CoT	0.74	2.3
Claude-3.5-Haiku, ReAct	1.92	5.9
Llama-3.1-8B, zero-shot	0.16	0.31
Llama-3.1-8B, CoT	0.48	0.91
Llama-3.1-8B, ReAct	1.05	2.0

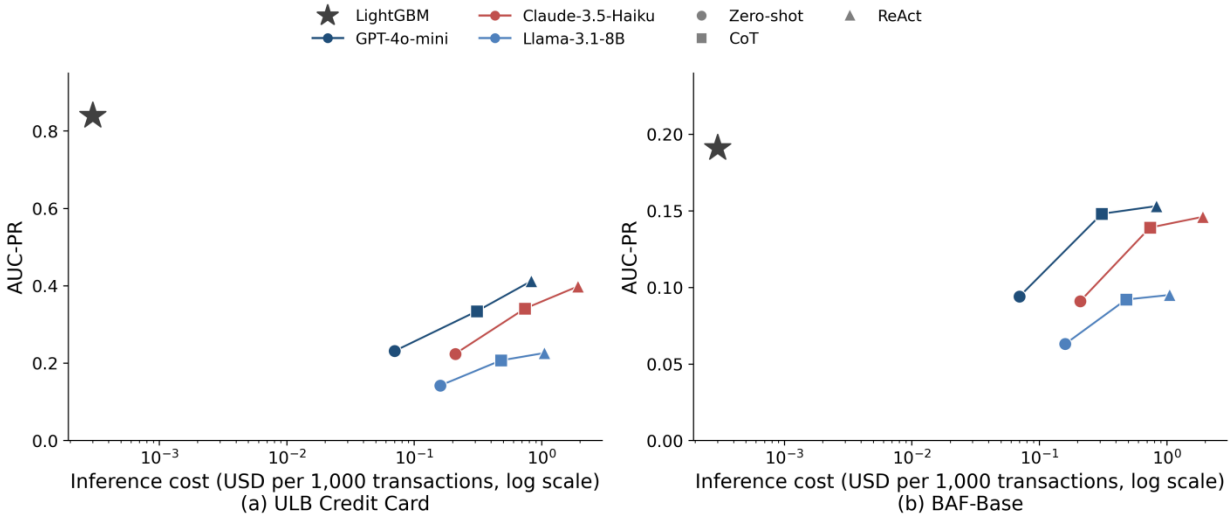

Figure 2. Cost-Accuracy Trade-off of Agent and Classical Detectors

AUC-PR against inference cost per one thousand transactions (logarithmic scale) on (a) the ULB credit-card corpus and (b) BAF-Base. LightGBM occupies the upper-left corner of both panels (0.839 at 0.0003 USD on ULB; 0.191 on BAF), dominating every agent configuration. Among agents, GPT-4o-mini traces the best cost-accuracy curve, rising from 0.231 at 0.07 USD (zero-shot) to 0.412 at 0.83 USD (ReAct) on ULB. The horizontal spread of roughly four orders of magnitude at comparable or lower accuracy is the figure's central finding.

4.3. Error Analysis

A. False-Positive Taxonomy

Two annotators independently coded 100 false positives of the best configuration (GPT-4o-mini, ReAct) per dataset at the 5% FPR operating threshold, reaching agreement of $\kappa = 0.76$ before adjudication; Figure 3(a) shows the

distribution. On ULB the dominant category is amount over-weighting, 41 of 100 cases in which an unusually large amount drove the score despite benign component values. On BAF the leading category shifts to single-feature anchoring (35 of 100), where one alarming attribute, most often a low name-email similarity, outweighed a dozen reassuring ones. Spurious reasoning chains, rationales whose stated logic does not follow from the cited values, account for 19 and 22 cases respectively, and the remaining 13 and 15 cases were judged defensible borderline calls. The category shift between datasets suggests the failure is feature-conditional rather than intrinsic: when semantics are hidden the agent falls back on the one interpretable quantity available, the amount, and when semantics are present it over-trusts the most alarming name it recognises.

B. False-Negative Taxonomy

The same protocol applied to 100 false negatives per dataset ($\kappa = 0.72$) yields the distribution in Figure 3(b). Missed fraud expressed through combinations of individually weak signals is the largest category on both corpora, 38 cases on ULB and 47 on BAF, confirming that single-record prompting deprives agents of the interaction effects that tree ensembles learn directly. Conservative scoring just below the operating threshold accounts for 29 and 24 cases, serialization information loss for 18 and 13, and rationale-score inconsistency, where the written reasoning flags risk yet the emitted score stays low, for 15 and 16. The weak-signal failure mode mirrors the motivation behind relational anti-money-laundering benchmarks, where laundering behaviour is detectable only across linked records, as in the AMLworld transaction graphs [19], the Elliptic Bitcoin graph [20], its extended address-level successor [21], and typology-rich synthetic suites such as SAML-D [22]; record-isolated agents are structurally blind to exactly this class of evidence.

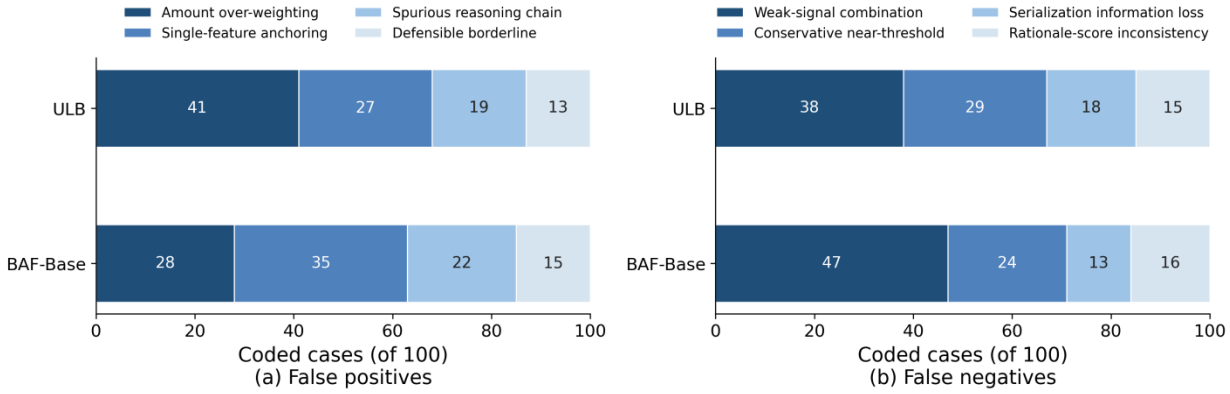

Figure 3. Taxonomy of Agent Errors at the 5% FPR Operating Threshold

Distribution of 100 manually coded errors per dataset for the strongest agent configuration, shown for (a) false positives and (b) false negatives. In (a), amount over-weighting dominates on the ULB corpus (41 cases) while single-feature anchoring dominates on BAF-Base (35 cases). In (b), fraud expressed through combinations of weak signals is the leading miss category on both datasets (38 and 47 cases), followed by conservative near-threshold scoring (29 and 24 cases). Inter-annotator agreement is $\kappa = 0.76$ for false positives and $\kappa = 0.72$ for false negatives.

5. Discussion

5.1. Implications and Limitations

The measurements support a cautious reading of LLM agents in tabular fraud detection. On anonymised features the best agent configuration recovers less than half the AUC-PR of a tuned gradient-boosted tree, and even on semantic features it stops short of the cheapest classical detector while costing several thousand times more per decision. What the agents add is not ranking accuracy but articulated rationales: every score arrives with a written justification that an analyst can inspect, a property no tree ensemble offers natively. The economically sensible deployment is a cascade in which the tree model scores the full stream and the agent enriches only the alert queue, where its per-case cost is negligible against analyst time and its rationale quality, rather than its threshold behaviour, determines value. Such a

deployment also presupposes a compliant data path: because transaction records are sensitive, any agent-based review must sit behind data minimisation, access control, and audit logging, with de-identification or private hosting wherever an external API is involved, and the agent should inform analyst review rather than serve as the sole basis for declining or freezing an account. The error taxonomy tempers even this proposal: roughly one in five agent rationales was internally inconsistent or spurious, so rationale auditing would need to precede any analyst-facing rollout. Drift behaviour also remains unmeasured here; agents adapt by prompt revision rather than retraining, a property that could prove either an advantage or a liability under shifting fraud tactics and deserves its own longitudinal study. Several limitations bound the conclusions. Only two datasets were studied, one of them synthetic; each configuration was run once at temperature 0, leaving decoding variance unquantified; prompt wording was fixed in advance and not optimised per backbone, so the reported figures should be read as a floor rather than a ceiling for in-context approaches; and the strict single-record protocol excludes retrieval of related transactions, which the false-negative analysis identifies as the binding constraint. Two further caveats deserve emphasis. The ULB corpus has circulated publicly for a decade and may overlap with backbone pretraining data, although the absence of any agent advantage on that dataset suggests memorisation, if present, did not translate into detection skill. Agent scores are also poorly calibrated, clustering at multiples of five and ten, which complicates threshold setting and means the reported R@5%FPR values rest on coarser score granularity than those of the classical detectors.

5.2. Conclusion and Future Work

This study measured 32 LLM-agent configurations against four classical detectors on two public fraud benchmarks under a single fixed, pre-specified protocol. Tree ensembles remain the accuracy and cost leaders; agent accuracy improves monotonically with richer prompting, tool use helps most when feature semantics are hidden, and the residual gap on semantic features narrows to 0.043 AUC-PR. Three extensions follow naturally from the observed failure modes. Instruction fine-tuning of open backbones on serialized transactions would test whether the closed-model advantage is architectural or merely a matter of adaptation. Retrieval-augmented prompting that surfaces linked records would target the weak-signal-combination misses directly. A production-oriented study of the cascade design, measuring analyst throughput with and without agent rationales, would establish whether explanation quality translates into operational value. Until such evidence exists, the prudent reading of the present measurements is that LLM agents have earned a seat beside the fraud analyst, not a place inside the scoring path.

References

[1] Wu, S., Irsoy, O., Lu, S., Dabravolski, V., Dredze, M., Gehrmann, S., Kambadur, P., Rosenberg, D., & Mann, G. (2023). BloombergGPT: A large language model for finance. arXiv. https://arxiv.org/abs/2303.17564

[2] Xie, Q., Han, W., Zhang, X., Lai, Y., Peng, M., Lopez-Lira, A., & Huang, J. (2023). PIXIU: A comprehensive benchmark, instruction dataset and large language model for finance. Advances in Neural Information Processing Systems, 36.

[3] Xie, Q., Han, W., Chen, Z., Xiang, R., Zhang, X., He, Y., Xiao, M., Li, D., Dai, Y., Feng, D., ... Huang, J. (2024). FinBen: A holistic financial benchmark for large language models. Advances in Neural Information Processing Systems, 37.

[4] Yang, C., Liu, H., Wang, D., Zhang, Z., Yang, C., & Shi, C. (2025). FLAG: Fraud detection with LLM-enhanced graph neural network. In Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (pp. 5150–5160). ACM. https://doi.org/10.1145/3711896.3737220

[5] Ou, R., Zhu, K., Zhang, N., Li, J., Chen, C., Xu, Y., & Jiang, C. (2026). Targeting borderline fraudsters: Multi-view hypergraph fraud detection with LLM-guided contrastive learning. Proceedings of the AAAI Conference on Artificial Intelligence, 40(18), 15591–15599. https://doi.org/10.1609/aaai.v40i18.38588

[6] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. Advances in Neural Information Processing Systems, 35, 24824–24837.

[7] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models. In The Eleventh International Conference on Learning Representations.

[8] Shinn, N., Cassano, F., Berman, E., Gopinath, A., Narasimhan, K., & Yao, S. (2023). Reflexion: Language agents with verbal reinforcement learning. Advances in Neural Information Processing Systems, 36.

[9] Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., & Narasimhan, K. (2023). Tree of thoughts: Deliberate problem solving with large language models. Advances in Neural Information Processing Systems, 36.

[10] Hegselmann, S., Buendia, A., Lang, H., Agrawal, M., Jiang, X., & Sontag, D. (2023). TabLLM: Few-shot classification of tabular data with large language models. In Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (PMLR Vol. 206, pp. 5549–5581).

[11] Yang, H., Liu, X.-Y., & Wang, C. D. (2023). FinGPT: Open-source financial large language models. In FinLLM Symposium at IJCAI 2023.

[12] Dou, Y., Liu, Z., Sun, L., Deng, Y., Peng, H., & Yu, P. S. (2020). Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In Proceedings of the 29th ACM International Conference on Information and Knowledge Management (pp. 315–324). ACM.

[13] Liu, Y., Ao, X., Qin, Z., Chi, J., Feng, J., Yang, H., & He, Q. (2021). Pick and choose: A GNN-based imbalanced learning approach for fraud detection. In Proceedings of the Web Conference 2021 (pp. 3168–3177). ACM.

[14] Egressy, B., von Niederhäusern, L., Blanuša, J., Altman, E., Wattenhofer, R., & Atasu, K. (2024). Provably powerful graph neural networks for directed multigraphs. Proceedings of the AAAI Conference on Artificial Intelligence, 38(10), 11838–11846.

[15] Liu, X., Yu, H., Zhang, H., Xu, Y., Lei, X., Lai, H., Gu, Y., Ding, H., Men, K., Yang, K., Zhang, S., Deng, X., Zeng, A., Du, Z., Zhang, C., Shen, S., Zhang, T., Su, Y., Sun, H., ... Tang, J. (2024). AgentBench: Evaluating LLMs as agents. In The Twelfth International Conference on Learning Representations.

[16] Dal Pozzolo, A., Caelen, O., Johnson, R. A., & Bontempi, G. (2015). Calibrating probability with undersampling for unbalanced classification. In 2015 IEEE Symposium Series on Computational Intelligence (pp. 159–166). IEEE.

[17] Jesus, S., Pombal, J., Alves, D., Cruz, A. F., Saleiro, P., Ribeiro, R. P., Gama, J., & Bizarro, P. (2022). Turning the tables: Biased, imbalanced, dynamic tabular datasets for ML evaluation. Advances in Neural Information Processing Systems, 35, 33563–33575.

[18] Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., & Scialom, T. (2023). Toolformer: Language models can teach themselves to use tools. Advances in Neural Information Processing Systems, 36.

[19] Altman, E., Blanuša, J., von Niederhäusern, L., Egressy, B., Anghel, A., & Atasu, K. (2023). Realistic synthetic financial transactions for anti-money laundering models. Advances in Neural Information Processing Systems, 36.

[20] Weber, M., Domeniconi, G., Chen, J., Weidele, D. K. I., Bellei, C., Robinson, T., & Leiserson, C. E. (2019). Anti-money laundering in Bitcoin: Experimenting with graph convolutional networks for financial forensics. In KDD Workshop on Anomaly Detection in Finance. arXiv. https://arxiv.org/abs/1908.02591

[21] Elmougy, Y., & Liu, L. (2023). Demystifying fraudulent transactions and illicit nodes in the Bitcoin network for financial forensics. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (pp. 3979–3990). ACM.

[22] Oztas, B., Cetinkaya, D., Adedoyin, F., Budka, M., Dogan, H., & Aksu, G. (2023). Enhancing anti-money laundering: Development of a synthetic transaction monitoring dataset. In 2023 IEEE International Conference on e-Business Engineering (pp. 47–54). IEEE.