SBSeg 2026 Armação dos Búzios - RJ
VEJA OS SELOS DE ARTEFATOS

PixGuard-Sim: A Deadline-Aware Testbed for Pix Fraud Detectors

Cristhian Kapelinski[1], **Diego Kreutz**[1]

[1]AI Horizon Labs, Federal University of Pampa (UNIPAMPA)
cristhianavila.aluno@unipampa.edu.br, diegokreutz@unipampa.edu.br

Abstract. *Brazilian regulation gives a paying institution about a second and a half to decide whether a suspect Pix transfer should be diverted for review before it settles, on a system used by 170 million people where settlement is irreversible. Yet every openly released payment-fraud dataset and baseline we surveyed is scored in batch, with no reference to that window. PixGuard-Sim is an open testbed that scores a detector against the window from the latency it measures, and releases a Pix-native generator with a settlement clock and two scenarios open data omits: multi-hop MED-2.0 refund tracing and payment under coercion. Scoring accuracy and timing together pulls them apart. The most accurate detectors we evaluate are cloud reasoning models that never saw a labeled event from this stream: the stronger finds 112 of 150 frauds where a random forest fitted to the data finds 93, and neither answers in time. Ranking by accuracy alone selects what cannot be deployed.*

1. Introduction

Pix, the instant-payment system launched by the Central Bank of Brazil (BCB) in 2020, became the country's dominant payment method within five years, moving **63.8 billion transactions worth R$ 26.9 trillion in 2024**[1]. It also accounts for **54.7% of all payment transactions** in Brazil (second half of 2025)[2], and reaches roughly 170 million people through about 890 million registered keys[3]. A *key* is an entry in DICT (Diretório de Identificadores de Contas Transacionais), the directory that resolves a phone number, tax id, or random token to a destination account. What makes Pix attractive is that a transfer clears in seconds and, once cleared, is final: the payer cannot unilaterally reverse it the way a cardholder disputes a card charge. That same finality is what makes a scam expensive. The moment a coerced or deceived victim confirms the transfer, the funds are the recipient's, and the only recourse is a regulator-defined refund procedure that has to locate the money in accounts the fraudster controls and is already moving it out of. Prevention therefore has to happen in the seconds *before* the transfer clears, and that interval is what this paper measures detectors against.

That risk is now material, and the loss it produces is measured as money that could not be given back. Victims and institutions asked for **R$ 4.94 billion** to be returned in 2024 after a fraud was established, and did not get it, **up 70% year over year**. The refunds failed because the receiving account had been closed or emptied first, across 3.45 million rejected requests[4]. Every one of those requests was made after the transfer had

[1] Febraban, Pix was the most-used payment method in Brazil in 2024, 2025. [2] IstoÉ Dinheiro, Pix now accounts for 54.7% of transactions in Brazil, 2026. [3] Central Bank of Brazil, Pix Statistics, 2026. [4] Estadão/InfoMoney, Pix fraud losses rise 70% to R$ 4.9 billion in 2024, 2025, on Central Bank data.

settled, since a refund presupposes that the money moved, and in each case the after-the-fact mechanism failed to bring it back. How many of them a decision taken before settlement would have caught is not something these figures can say, and we make no such estimate; what they do establish is that recovery after settlement is where the losses are concentrated, which is the reason to measure detection before it. The same report quotes the operational manual of DICT defining a fraud to include a transfer the payer initiated *under coercion or extortion*[4], which is why coercion appears as one of our scenarios rather than as an exotic case. A recent taxonomy of Pix fraud [Pizzolato et al. 2025] was built from a literature review and practitioner consultations. It classifies fifteen scam methodologies into four thematic groups, and explicitly recommends building fraud simulators for controlled testing. The regulatory response has shifted from reactive refunds toward real-time prevention. The MED-2.0 mechanism (Mecanismo Especial de Devolução, the BCB special refund mechanism for fraud victims) was introduced by Resolução BCB número 493 of 28 August 2025, whose article 121 makes the value-recovery functionality **mandatory from 2 February 2026** for participants acting as transactional-account providers or special settlement agents[5]. Its stated advance over the mechanism it replaces is reach: where blocking and return applied only to the first account that received the fraudulent funds, MED-2.0 traces beyond that account, and the directory itself sends infraction notices to the institutions holding the transactions most likely to lie on the drain path, which then freeze the balance and review[6]. In other words, the regulator has committed institutions to reasoning about chains of accounts, not single transfers, which is precisely the structure no open Pix dataset contains and which our multi-hop scenarios instantiate. No regulation fixes a blocking latency, but the regulator does publish how long each participant may take, and grants a far wider window once a transfer is already under suspicion. Section 3 derives the decision deadline from those published budgets.

How should we measure a Pix fraud detector when the regulation demands not just accuracy but a decision *in time*? Conventional precision and recall are computed in batch, after the fact, and say nothing about whether a flag would have arrived before the irrevocable decision point. The closest open artifacts leave that question open. Anti-money-laundering (AML) simulators model laundering topology rather than consumer-scam dynamics and are written to be jurisdiction-agnostic: PaySim advances time in coarse hourly steps [Lopez-Rojas et al. 2016], and Tide, the closest analog to layered refund tracing, chains illicit transfers through intermediary accounts at hour-scale delays with no instant-payment semantics [van den Beukel et al. 2026]. The one open Pix-native dataset, pix-fraud-br, stops at the first transfer to a single mule[7]. None scores a detector against the pre-transaction window the regulation now centers on.

We close that gap with PixGuard-Sim, an open-source testbed agnostic to both the detector and the source of the data. It takes a detector and a labeled stream, times how long the detector actually needs to score one event, and reports the share of true frauds it both flags and decides on within a configurable pre-transaction deadline, alongside precision, recall, F1, and PR-AUC, the area under the precision-recall curve, with 95% confidence intervals. It ships reference definitions of the two Pix-native cases no open dataset carries, layered MED-2.0 refund tracing and coercion, and adapters that let one detector be scored

[5] Central Bank of Brazil, Resolução BCB 493, 28 August 2025, art. 121. [6] Central Bank of Brazil, Special Refund Mechanism (MED 2.0), Pix circuit, 2025. [7] A. Messina, pix-fraud-br: synthetic Pix fraud dataset, 2026.

across three independently produced sources without code changes. The deadline matters when detection moves to heavier models, where one decision costs what the institution has to make it, an axis batch accuracy cannot see.

Contributions. We make three, all released as an open artifact (Sections 3, 4, 5): a deadline-aware evaluation protocol combining detector scores, measured per-event latency, and a configurable decision deadline; a released Pix-native generator carrying that settlement clock, with reference definitions for the two scenarios open prior work omits, multi-hop MED-2.0 refund tracing and coercion, each mapped onto the published Pix-fraud taxonomy; and a comparison of the same detectors across three independently produced sources, our generator and the released Tide high-illicit and low-illicit (HI/LI) and pix-fraud-br sets. These are one contribution, not three: the deadline is measurable only because the generator carries a settlement clock, and interpretable only because three independent sources score the same detector.

2. Related Work

We group prior work into four themes: the Pix and instant-payment fraud landscape, fraud datasets and simulators, detection methods, and real-time and cross-distribution evaluation. Table 1 positions PixGuard-Sim against the closest comparable approaches on the dimensions that matter for pre-transaction Pix blocking.

Table 1. Closest comparable approaches versus PixGuard-Sim. The last column marks whether detectors are scored against a decision deadline.

Work	Payment domain	Fraud cases modeled	Chain depth	Time modeled	Deadline scoring
PaySim [Lopez-Rojas et al. 2016]	Mobile money	None; the paper states that injecting fraud is future work	–	Hourly steps	✗
pix-fraud-br[7]	Pix	Account takeover, then one transfer to one mule	Single-hop	Transaction timestamps	✗
Tide [van den Beukel et al. 2026]	Bank transfers	Overseas transfers, rapid movement, front business, synchronized transactions, U-turn	2 to 5 hops	Timestamps, 1 to 48 h between hops	✗
IBM AMLworld [Altman et al. 2023]	Bank, crypto	Fan-in, fan-out, gather-scatter, scatter-gather, cycle, random, bipartite, stack	Multi-hop	Transaction timestamps	✗
[Dal Pozzolo et al. 2018]	Credit card	–	–	Delay before the label arrives (δ days)	✗
FraudTransformer [Aminian et al. 2025]	Bank payments	–	–	Event timestamps and inter-event gaps	✗
PixGuard-Sim (this work)	**Pix**	**Account takeover, coercion, multi-hop MED-2.0 refunds, mule chains**	**Single and multi-hop**	**Measured per-event decision latency**	✓

Pix and instant-payment fraud. The recent Pix-fraud taxonomy [Pizzolato et al. 2025] characterizes the Brazilian landscape, but as an analytical study it instantiates no mule chains, MED layers, or DICT; our mapping of scenarios onto its four groups operationalizes it. The same authorized-push-payment (APP) problem appears outside Brazil: a regulatory analysis of victim-authorized transfers and the mandatory-reimbursement deadline [Braithwaite 2024] examines time-bound responses to fraud *claims*, after the fact, and contributes neither data nor detector. The largest APP dataset we know of is the UK regulator's synthetic set, reachable by application rather than open download[8]. Its seven

[8] Financial Conduct Authority, Authorised Push Payment synthetic data, 2024.

scam typologies are all deception, a victim persuaded to authorize; none is a payment made under duress, and it carries no settlement clock.

Fraud datasets and simulators. Agent-based simulators are the established way to release privacy-safe fraud data. PaySim [Lopez-Rojas et al. 2016] advances time one hour per step and BankSim [Lopez-Rojas and Axelsson 2014] one day, neither with a real-time decision point; PaySim's published design covers legitimate traffic only, so the fraud labels later work inherits, pix-fraud-br's among them, come from the released dataset rather than from that design. The AML lineage adds laundering topology: AML-Sim [Weber et al. 2018] injects known patterns, AMLworld calibrates an agent-based model to real statistics and labels every laundering transaction [Altman et al. 2023], and Tide [van den Beukel et al. 2026] adds explicit timing to it, which makes it the closest open analog to MED-2.0 layering. Their scope is complementary rather than competing, since instant-payment semantics, MED refund tracing, and duress fall outside jurisdiction-agnostic goals. AMLworld does list extortion and kidnapping, but as crimes that *produce* funds later laundered: it captures the criminal's proceeds, not the coerced victim's own payment. We therefore build on Tide as an independent temporal base rather than reimplementing a generator. That leaves pix-fraud-br[7], a PaySim derivative reframed in Pix terms with one gradient-boosted baseline we reproduce; its own documentation notes that real mule networks chain two to four accounts, where it stops at the first, and its baseline, like every other, is scored in batch.

Detection methods. The detectors PixGuard-Sim scores come from two families. The tabular family is XGBoost [Chen and Guestrin 2016], a strong tabular-fraud baseline, plus logistic regression, random forest, and gradient boosting from scikit-learn [Pedregosa et al. 2011]. The graph family treats accounts and transfers as a network [Cheng et al. 2025]: graph convolutional networks for illicit-transaction forensics [Weber et al. 2019], EvolveGCN for graphs that change over time [Pareja et al. 2020], CARE-GNN against fraudsters that camouflage their neighborhoods [Dou et al. 2020], and heterogeneous networks fitted to real bank transactions [Johannessen and Jullum 2023]. Our graph baseline is GraphSAGE [Hamilton et al. 2017], which aggregates a neighborhood and so generalizes to unseen accounts. Every one of these is judged on how well it ranks fraud in batch; none is scored against a pre-transaction deadline.

Real-time and cross-distribution evaluation. A practitioner line argues that batch accuracy is unrealistic for streaming fraud, from an operational account of realistic measures, class imbalance, and drift [Dal Pozzolo et al. 2014] to a formalization of stream detection under verification latency [Dal Pozzolo et al. 2018], a quantification of distribution shift over time [Lucas et al. 2019], ROSFD's on-demand retraining under drift detectors [Yelleti 2025], and FraudTransformer's use of timestamps and inter-event gaps [Aminian et al. 2025]. The latency all of these model is *verification* latency, the days before a transaction's true label reaches the learner, not the milliseconds before settlement becomes irrevocable; and each reports on a single data distribution. Spade [Jiang et al. 2022] is the closest work measuring the second clock: it finds fraud rings on an evolving graph in microseconds and scores a prevention ratio, the share of a fraudster's transactions arriving after the ring is identified. That fits a rail where blocking an account stops its next transfer. Pix settles per transfer and irrevocably, so the question

is whether the decision on *this* payment arrives before it settles, and our metric is that analog.

That is the gap Table 1 summarizes: across the generators, the detectors, and the streaming line, no released artifact both carries the moment a transfer becomes irrevocable and scores a detector against it. PixGuard-Sim consumes those datasets through thin adapters and adds one of its own, to our knowledge the only openly available payment stream that carries a per-event settlement clock.

3. Architecture and Method

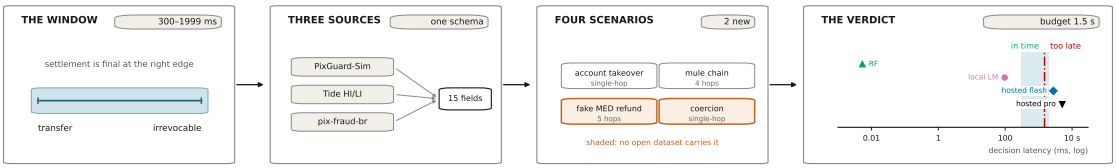


Figure 1. The window that closes, the three sources feeding one schema, the four scenarios with the chain depth each carries, and where each detector's measured decision lands against the 1.5 s authorization budget.

Architecture. PixGuard-Sim is a four-stage pipeline (Figure 1). A per-generator *adapter* maps each source into one normalized Pix-native event schema; a *scenario library* contributes the missing fraud cases, multi-hop MED-2.0 refunds and coercion among them; a thin *detector interface* exposes only the numeric features and the event timeline to a pluggable detector; and a *deadline-aware scorer* trains the detector, times its real per-event latency, and reports accuracy together with the pre-deadline flag fraction. Both the detector and the generator are swappable without touching the harness, which is what lets the same evaluation run across three independent sources.

Event schema. Every generator maps into one normalized Pix-native event record, which carries a relative timeline, a MED-layer index for refund chains, and the Pix-native signals a detector may read. The external sources supply only the amount; the three behavioral columns they do not carry are filled by the harness from a fixed, label-correlated noise process, which the paper calls *injected signals*. No detector ever sees the label or the scenario name: a tabular detector is handed the four numeric feature columns and the timeline, and the graph detector additionally the payer and payee identifiers, without which it cannot build an account network. Appendix C lists all fifteen fields and where each one comes from.

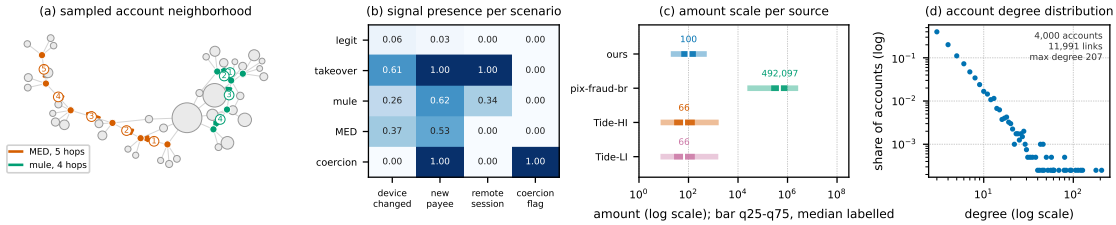


Figure 2. Account graph with one real MED and one real mule chain, signal presence, amount scale per source, and account degrees.

Three independent sources. The harness scores detectors on our own generator and on two externally released datasets, reached through thin adapters (Table 2). Our generator

builds its account network as a Barabási–Albert graph, in which a few accounts accumulate disproportionately many links, as real payment networks do (Figure 2d). It then emits the four Pix-native scenarios on top of it. It is the only source that carries coercion and multi-hop MED-2.0, so its role is to supply those scenarios, not to lend external credibility. That role falls to the two external sources, built by other groups for other purposes. Tide's HI and LI datasets [van den Beukel et al. 2026] contribute multi-hop laundering at illicit rates far below ours. pix-fraud-br[7] contributes Pix-framed single-hop fraud, together with a published baseline we can reproduce. Every input is synthetic and fully labeled.

Table 2. The three synthetic sources, their licenses, and their role in the evaluation.

Source	License	Illicit rate	Role in this evaluation
PixGuard-Sim generator (ours)	MIT	1.19%	Emits 40 482 events on a scale-free account graph; the only source of the coercion and multi-hop MED-2.0 scenarios
pix-fraud-br[7]	ODC-BY	0.77%	Externally released Pix-framed dataset of 2 000 000 single-hop transfers; anchors our results against its published baseline
Tide HI/LI [van den Beukel et al. 2026]	CC BY 4.0	0.19% / 0.10%	Externally released multi-hop laundering chains; tests the detectors at illicit rates six to twelve times rarer than ours

Scenarios. PixGuard-Sim instantiates four scenarios. Three map onto groups named by the taxonomy [Pizzolato et al. 2025]: *account-takeover* (remote-access "ghost hand"), *coercion* (robbery, express kidnapping, and extortion), and *fake-MED refund* (the refund/urgency group). We extend the last with multi-hop MED-2.0 tracing of up to five layers and add a *mule-chain* dispersal scenario. The two chained scenarios are not one case under two names. A MED chain is the abused refund flow that the regulator's tracing follows: no remote session at any layer, every signal fading by a fixed factor per hop. A mule chain disperses money already taken, so its first hop still looks like the takeover behind it, remote session and all, and then goes cold at once. Multi-hop tracing, MED layering, and mule chains are our additions, named by no taxonomy group. Coercion is deliberately hard to detect: the victim uses their own device in a genuine session, so of the four columns a tabular detector sees the event carries no device change and no unusual velocity, leaving an amount and a new payee. Each layer of a chain is a separately scored event, and Figure 2a traces one chain of each kind through the account graph.

The deadline metric. Each detector returns a fraud score and the per-event decision latency the harness *measures* for it by timing real scoring, not a configured budget. For the set F of true-fraud events, score s_i, decision threshold τ, measured latency ℓ_i, and deadline d, the pre-deadline flag fraction is

$$\mathrm{predeadline}(d) = \frac{1}{|F|} \sum_{i \in F} \mathbf{1}[s_i \geq \tau]\, \mathbf{1}[\ell_i \leq d].$$

A fraud counts only if the detector both flags it and reaches that decision inside the window; this is recall restricted to the decisions that arrive in time, a second condition on a familiar quantity rather than a new estimator. Where ℓ_i comes from matters. A detector that scores one event per call, as a language model does, contributes each event's own generation time; a detector that scores a whole frame at once contributes that frame's mean time per event. Because a lightweight tabular model decides in well under a millisecond, it clears every deadline we evaluate and its pre-deadline fraction is simply its recall. The metric separates detectors only once measured latency approaches the window, which is the regime a heavier model enters.

The deadline is not ours to invent. The Brazilian regulator publishes a time manual that bounds each participant's share of a Pix transfer, and gives the paying institution 1.5 s at the 95th percentile to authorize the payment it is asked to send[9]. The same manual widens that budget to 30 minutes, or 60 outside business hours, once the order is already suspected of fraud[9]. Those two numbers together define the job. A detector does not have to complete an investigation before settlement; it has to decide, inside the ordinary 1.5 s budget, whether this transfer is worth diverting into the wide one. That is why the deadline we score against is of the order of a second and not of the order of an hour. Appendix D lists the budgets bounding each leg of a transfer, so a reader can see which one this deadline is. Our generator carries its own settlement clock, marking each transfer irrevocable between 300 and 1999 ms after initiation. That clock is a generator parameter, tighter than the regulator's cycle, which reaches finality in seconds; it never enters the deadline metric, which compares measured latency against the authorization budget. We report the deadline, the latency distribution, and the reference machine together.

4. Evaluation

We evaluate eight detectors. Four are tabular, meaning they see one row of numeric features per transfer: a fixed rule threshold (RULE) as a floor, logistic regression (LR), random forest (RF), and the gradient-boosted XGBoost (XGB). Three are language models prompted with one transfer at a time: a small off-the-shelf instruct model run on our own machine, which we prompt in two regimes and which therefore contributes two rows, and two reasoning models called over the network in the cloud. XGBoost is scored on the external sources, where it wins (Table 5); Table 3 covers the other three. A graph detector is in Appendix B. Every detector's per-event latency is measured by the harness rather than assumed, and every detector within a given comparison sees exactly the same events.

Our generator emits 40 482 events, of which 482 are fraud, a rate of 1.19%. Detectors are trained and scored on a label-stratified split whose evaluation set holds 16 193 events (193 frauds); the Tide and pix-fraud-br runs use a label-stratified subsample of 400 000 transactions each. Confidence intervals are written *value* [lo, hi]: Wilson intervals for proportions, and bootstrap intervals over 1000 resamples for F1 and PR-AUC. Everything that runs locally runs on one machine, an AMD Ryzen 5 8600G (6 cores, 12 threads) with 32 GB of memory and an NVIDIA GeForce RTX 5060 Ti (16 GB), so those latencies are comparable with each other. The two cloud models run on the provider's hardware and are reached over a residential connection, so their latency is a round trip and not a property of this machine; that is stated wherever they appear, because it is what an institution calling such a model would also pay.

4.1. Among sub-millisecond detectors, the deadline changes nothing

Table 3 scores the three cheapest detectors on our own generator.

The random forest is the strongest of the tabular detectors, at PR-AUC 0.743 [0.685, 0.799] and recall 0.622 [0.552, 0.687] over 193 frauds (Table 3). Logistic regression trails it at 0.706 and the rule threshold marks the floor at 0.431, which is where it is by construction: the rule fires on a fixed amount or a fixed velocity, and only about a third of the frauds in this stream clear either threshold, so the gap says nothing about

[9] Central Bank of Brazil, Manual de Tempos do Pix, version 7.0, 2026.

Table 3. Accuracy and measured per-event latency of the tabular detectors on 16 193 events from our generator.

Detector	F1	PR-AUC	Recall	Latency (ms/event)
RULE	0.443	0.431	0.342	< 0.01
LR	0.639	0.706	0.513	< 0.01
RF	**0.684**	**0.743** [0.685, 0.799]	**0.622** [0.552, 0.687]	< 0.01

tuning. The last column is what matters for the deadline. All three decide in well under a millisecond, so each one's pre-deadline fraction is exactly its recall. Among detectors this cheap the deadline does not discriminate; it becomes an axis only when scoring gets expensive, which the next experiment tests.

4.2. The detectors that work are the ones that answer too late

Figure 3a shows the window the events close in, and Figure 3b where each decision lands in it.

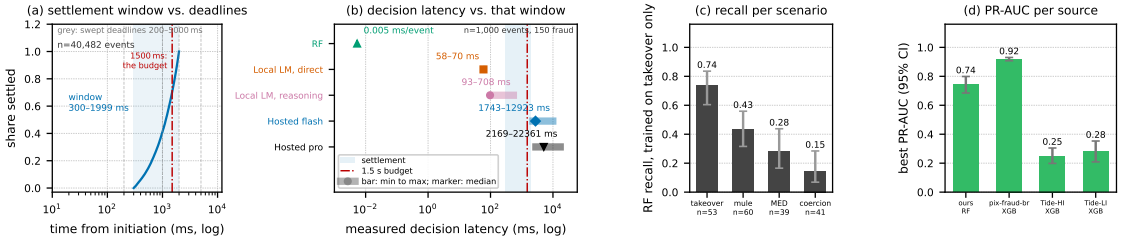


Figure 3. Settlement window, measured decision latency, per-scenario recall, and PR-AUC across the three sources.

We prompt Qwen2.5-1.5B-Instruct with one transfer per call, in two regimes: *direct*, which asks only for a fraud probability and caps generation at 8 tokens, and *reasoning*, which asks it to weigh the four signals in at most three sentences first and caps at 320. Both take the highest-probability token at each step, so runs agree; Appendix E gives the prompts verbatim. Both regimes and the random forest score the same 1000 events, of which 150 are fraud (Table 4), a slice enriched above the pool's 1.19% rate so the fraud-side intervals rest on more than a handful of cases.

Table 4. Accuracy and latency on 1000 identical events, both columns read against the 1.5 s the regulator's time manual gives a paying institution to authorize: *share of budget* is p95 latency over it, *in time* the share of the 150 frauds flagged and decided inside it. Latency is in ms.

Detector	PR-AUC	Latency mean / p95	Output tokens	Share of budget	Decisions / s	In time
RF	**0.931**	< 0.01	none	$< 0.001\%$	187 126	0.620
Direct LLM	0.201	60 / 62	5	4.1%	16	0.627
Reasoning LLM, local	0.160	109 / 103	8	6.8%	9.7	0.993
DeepSeek v4-flash, cloud	0.846	3108 / 5025	171	**335%**	0.2	**0.000**
DeepSeek v4-pro, cloud	**0.849**	5704 / 10 329	284	**689%**	0.1	**0.000**

Two detectors run in the cloud, called one transfer per request the way an institution would call them, and they are the ones the deadline decides. They detect, at PR-AUC 0.846 and 0.849 against 0.150 for a detector flagging at random, without ever seeing a labeled event from this stream. In counts, the stronger flags 112 of the 150 frauds at 30 false alarms among 850 legitimate transfers, where the random forest trained on 16 193 labeled events flags 93 at 1: a zero-shot model catches more fraud here than one fitted to the distribution. It also never arrives in time. At the 95th percentile the two spend 5025 and 10 329 ms, 335% and 689% of the budget, so the share of frauds both flagged and

decided inside it is 0.000 and 0.000 (Table 4): every fraud they catch, they catch after the money is gone. The small model on our own machine is the control showing this is about deliberation, not language models as such: asked for the same reasoning it emits a bare verdict and stops, answering in 109 ms at PR-AUC 0.160, chance level, flagging almost everything at precision 0.149. What separates them is not speed but how much they say: the budget buys about 122 generated tokens on this machine, and Table 4 shows what each spends against it. A reader can place their own model by counting the tokens it needs. Two cautions. Only the random forest was fitted to this distribution, so the language models judge an amount without knowing the legitimate median is 100; and the slice being enriched, these figures describe this comparison, not a production flag rate.

4.3. Training on single-hop fraud degrades with every unseen scenario

Legitimate traffic plus account takeover, all of it single-hop, is the modeling scope of the one open Pix dataset. A detector trained on that case alone keeps most of its recall on the scenario itself, at 0.736 [0.604, 0.836] over 53 frauds, and loses it monotonically as the scenario moves away from what it saw, through mule chains and multi-hop MED-2.0 refunds down to 0.146 [0.069, 0.284] over 41 coercion events (Figure 3c, with per-scenario counts and 95% intervals). The ordering is the point: the further a scenario sits from the one the detector saw, the less accuracy survives, and an institution that has labeled only takeover cannot see this from its own data. Coercion is the far end, and the reason is visible in the signals rather than in the model. The generator emits a coerced payment as a genuine session on the victim's own device, so of the four columns a tabular detector sees, the two the takeover-trained detector relies on push it the wrong way: the device never changes, which Figure 2b shows, and the coerced payer's velocity sits *below* the legitimate average rather than above it. The other two do separate the case, since the payee is always new and the amount well above the legitimate median, but a detector that learned to read a takeover as a device change on a busy account does not weigh them enough. A synthetic scenario is only as informative as the signals it carries.

4.4. Both the source and the surviving feature set move the score

Table 5 scores the external sources twice: under their own features, and under the shared columns.

Table 5. Accuracy on the external sources, under each source's own features and under the four-column shared schema.

Source (illicit rate)	Features	Detector	F1	PR-AUC	Recall
pix-fraud-br (0.77%)	its own	XGB	0.851	0.920 [0.907, 0.930]	0.817
pix-fraud-br (0.77%)	its own	RULE	0.015	0.014	0.938
pix-fraud-br (0.77%)	shared schema	RF	0.080	0.137	0.045
Tide-HI (0.19%)	shared schema	XGB	0.303	0.250 [0.198, 0.305]	0.213
Tide-LI (0.10%)	shared schema	XGB	0.281	0.280 [0.209, 0.353]	0.186

Two things move a detector's score, and the harness separates them: which source the data comes from, and how much of it survives the mapping into the shared schema. On pix-fraud-br scored with its own published features, XGBoost reaches PR-AUC 0.920 [0.907, 0.930] (Table 5), against 0.865 for the dataset's own baseline on a different split and feature set[7]: not directly comparable, but both say the data is comfortably separable. Reduced to the four columns every source shares, the same dataset falls to 0.137, because most of what made it separable lived in fields the shared schema does not carry, chief

among them the receiving account's prior balance, which the dataset's own analysis puts at two thirds of its model's importance. Tide is reachable only through that shared schema: at illicit rates of 0.19% and 0.10%, XGBoost reaches 0.250 and 0.280 at recall 0.213 and 0.186, and the rule floor is near useless there, at 0.036. Figure 3d puts the best detector on each source side by side, naming which model won.

Training on one source and testing on another is harder still. Under the shared schema a random forest scoring 0.743 on our own data falls to 0.021 on pix-fraud-br; trained the other way it scores 0.137 there and 0.281 on ours. We attribute that collapse to no single cause: the sources differ at once in fraud rarity, in which fields survive, in the scale of the one that always does, and in graph shape, and this experiment separates none of the four. The third is the easiest to overlook and the largest: the median amount, in each source's own units, is 100 in our data and 66 in Tide-HI against 492 097 in pix-fraud-br, so the column a detector learns on barely overlaps the one it later meets (Figure 2c). A PR-AUC is a statement about a detector, a dataset, and a feature set together, and changing any one moved our numbers by more than 0.7.

A graph detector. Reading the account network does not help on the multi-hop scenarios: a two-layer GraphSAGE lands below every tabular detector at sub-millisecond latency, so the deadline leaves the ranking untouched (Appendix B).

5. Limitations and Conclusion

Limitations. Real Pix data is private, so every input here is synthetic and no fraud rate or accuracy estimates what an institution would see. The two Pix-native scenarios are our reference definitions, mapped onto a taxonomy naming neither MED layering nor mule chains, so that mapping is a proposal others may draw differently. Our coercion scenario also raises the amount and always uses a new payee, so it is not signal-free in general, only free of the two behavioral columns the single-hop detector had leaned on. The deadline verdict is a property of a detector *on a stated machine, network, and software stack*: a quantized checkpoint or a server-class GPU would move the model on our own machine, and running the same cloud model inside the institution would remove the network hop and the provider's queue. That correction is too small to change the verdict, since the two spend 3.4 and 6.9 times the budget and what must shrink is the deliberation itself, but any such verdict should be re-measured, not inherited. Finally, the graph baseline is one small architecture. Appendix A turns these results into an order of work for a team choosing a detector under the MED-2.0 regime.

Conclusion. We asked what a detector score should mean once the regulator gives an institution a second to decide whether a transfer is worth holding. PixGuard-Sim reports what a detector catches and when its answer arrives, and on our data the two disagree: the most accurate detectors are the two cloud reasoning models, which never answer inside the budget, while the one that always answers in time catches less. Ranking by accuracy alone selects what cannot be deployed. The scenarios and the sources give two further reasons to distrust a bare score: accuracy on single-hop takeover degraded with every unseen scenario, and reducing a detector to the shared columns cost more than changing it did. A score is interpretable only next to the deadline, the machine, the scenario, the feature set, and the data.[10] Real Pix traffic and cheaper deliberation come next.

[10] Artifact: `https://doi.org/10.5281/zenodo.22664944`

Acknowledgments

This work was partially supported by Capes, CNPq (Grant No. 409743/2025-9), and FAPERGS (Grant Nos. 24/2551-0001368-7 and 24/2551-0000726-1). We thank the anonymous reviewers of SBSeg 2026 for their suggestions and comments. The authors also acknowledge the use of Generative AI as a supporting tool for reviewing and improving the code and manuscript. Every figure is plotted by a script in the artifact, from the same result files the tables read.

References

Altman, E., Blanuša, J., et al. (2023). Realistic synthetic financial transactions for anti-money laundering models. In *NeurIPS Datasets & Benchmarks*.

Aminian, G., Elliott, A., et al. (2025). FraudTransformer: Time-aware GPT for transaction fraud detection. In *Workshop on AI for Financial Fraud Detection and Prevention at ACM ICAIF*.

Braithwaite, J. (2024). 'Authorized Push Payment' bank fraud: What does an effective regulatory response look like? *Journal of Financial Regulation*, 10(2):174–193.

Chen, T. and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *KDD*, pages 785–794.

Cheng, D., Zou, Y., et al. (2025). Graph neural networks for financial fraud detection: A review. *Front. Comput. Sci.*, 19(9):199609.

Dal Pozzolo, A., Boracchi, G., et al. (2018). Credit card fraud detection: A realistic modeling and a novel learning strategy. *IEEE TNNLS*, 29(8):3784–3797.

Dal Pozzolo, A., Caelen, O., et al. (2014). Learned lessons in credit card fraud detection from a practitioner perspective. *Expert Syst. Appl.*, 41(10):4915–4928.

Dou, Y., Liu, Z., et al. (2020). Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *CIKM*.

Hamilton, W. L., Ying, R., and Leskovec, J. (2017). Inductive representation learning on large graphs. In *NeurIPS*.

Jiang, J., Li, Y., et al. (2022). Spade: A real-time fraud detection framework on evolving graphs. *PVLDB*, 16(3):461–469.

Johannessen, F. and Jullum, M. (2023). Finding money launderers using heterogeneous graph neural networks. arXiv:2307.13499.

Lopez-Rojas, E. A. and Axelsson, S. (2014). BankSim: A bank payments simulator for fraud detection research. In *EMSS*.

Lopez-Rojas, E. A., Elmir, A., and Axelsson, S. (2016). PaySim: A financial mobile money simulator for fraud detection. In *EMSS*, pages 249–255.

Lucas, Y., Portier, P.-E., et al. (2019). Dataset shift quantification for credit card fraud detection. IEEE AIKE, arXiv:1906.06977.

Pareja, A., Domeniconi, G., et al. (2020). EvolveGCN: Evolving graph convolutional networks for dynamic graphs. In *AAAI*, pages 5363–5370.

Pedregosa, F., Varoquaux, G., et al. (2011). Scikit-learn: Machine learning in Python. *JMLR*, 12:2825–2830.

Pizzolato, G. L., Lopes, B. M., et al. (2025). A taxonomy of pix fraud in brazil: Attack methodologies, ai-driven amplification, and defensive strategies.

van den Beukel, M., Rožanec, J. M., and Varbanescu, A.-L. (2026). Tide: A customisable dataset generator for anti-money laundering research. arXiv:2603.01863.

Weber, M., Chen, J., et al. (2018). Scalable graph learning for anti-money laundering: A first look. arXiv:1812.00076.

Weber, M., Domeniconi, G., et al. (2019). Anti-money laundering in Bitcoin: Experimenting with graph convolutional networks for financial forensics. KDD Workshop on Anomaly Detection in Finance, arXiv:1908.02591.

Yelleti, V. (2025). ROSFD: Robust online streaming fraud detection with resilience to concept drift in data streams. arXiv:2504.10229.

A. Applying this to a detector choice

A team choosing a detector now that MED-2.0 is in force should measure candidate models on its own hardware and network, and treat the resulting latency as a selection criterion of the same standing as accuracy. The two point in opposite directions in our results: the two most accurate detectors we measure are the two that never answer in time, so a shortlist ranked by PR-AUC would put them first and a shortlist ranked by deployability would drop them. The order to work in follows from that. First measure per-event latency against the deadline the institution can actually enforce, and drop candidates whose slowest decisions fall outside it, since a flag that arrives after settlement cannot stop the transfer. Then compare accuracy among the candidates that survive. Doing it the other way round wastes the accuracy comparison on candidates that were never eligible. Two further habits follow from the scenario and cross-source results. A detector trained on the fraud an institution has already labeled, which in practice means single-hop account takeover, should not be assumed to carry over to layered refund fraud under MED-2.0; it needs its own evaluation set, and if the relevant signals are not collected at all, no amount of training substitutes for instrumenting them. And an accuracy figure quoted from a synthetic dataset should be treated as specific to that dataset until it is reproduced on another.

B. The graph detector

We train GraphSAGE with two layers of mean aggregation over payer-payee edges. It reaches PR-AUC 0.247 on our own data and 0.011 on the Tide-HI graph, below every tabular detector, and fits in about 2.4 s. We report it as a working graph detector inside the harness rather than as a competitive result. Neither graph gives a two-layer network much to aggregate over: an account averages 20 counterparties in our data and 29 in Tide-HI, and in both the pairs that transact are a small fraction of those that could. Its inference latency is sub-millisecond, like the tabular detectors, so here too the deadline leaves the ranking untouched. The detectors whose measured latency makes the deadline bind remain the language models, and among those only the two cloud ones are accurate enough for that to cost anything.

C. The normalized event schema

Section 4 reports that pix-fraud-br falls from PR-AUC 0.920 under its own features to 0.137 under the shared schema. Table 6 is why: it lists every field the harness normalizes to, and where that field comes from when the source is external.

Of the four columns a tabular detector sees, only the amount survives the mapping unchanged. The other three are the injected signals, drawn from a fixed, label-correlated noise process. Each is drawn from both classes: a signal reachable only from fraud would

Table 6. The fifteen fields of the normalized event record. The last column says where the value comes from when the stream is Tide or pix-fraud-br: taken *from the source*, a *filler* that carries no fraud signal, an *injected signal* drawn to correlate with the label, or left *always zero*.

Field	What it holds	From an external source
***Identity and ground truth**: never exposed to a detector*		
`event_id`	Event identifier	filler
`scenario`	Scenario name, or `legit`	from the source
`is_fraud`	Ground-truth label	from the source
***Accounts**: seen only by the graph detector*		
`payer_account`	Paying account	from the source
`payee_account`	Receiving account	from the source
`payee_dict_key`	Resolved DICT alias of the payee	filler
***Timeline and chain**: the deadline is measured on this clock*		
`t_init_ms`	Transfer initiated (relative ms)	from the source
`t_settle_ms`	Settlement becomes irrevocable	filler, a fixed 1000 ms later
`med_layer`	0 for the originating transfer, 1..N per chain hop, refund or mule	always zero
The four numeric features a tabular detector sees		
`amount_brl`	Transfer amount	**from the source**
`device_changed`	Payer device differs from its usual one	**injected**
`new_payee`	Payee alias never paid before	**injected**
`payer_velocity_1h`	Payer transfers in the trailing hour	**injected**
Pix-native signals our generator alone emits		
`is_remote_session`	Remote-access ("ghost hand") session	always zero
`coercion_flag`	Event initiated under duress	always zero

be a label in disguise, and a detector reading it would be scored on our injection rather than on the source. Together the three carry little: on the Tide slices a detector using only them reaches PR-AUC 0.04 or below. That is the whole of what an external source contributes to a tabular detector: one real number and three injected signals. The fall from 0.920 to 0.137 on the same dataset is the price of that reduction. It is why a score quoted for a detector is also a statement about the schema it was measured under.

D. The published Pix time budgets

The deadline in Section 3 is one line of a table the regulator publishes, and the other lines are what make it the right line. Table 7 lists the service levels of the settlement cycle[9], using the manual's own timestamps: t'_0 is the moment the paying institution receives the payer's confirmation, t_1 the moment it creates the settlement message, and t_4 is what the manual calls *finality*, the instant the national instant-payment system swaps the two balances and the transfer can no longer be undone.

Table 7. Service levels for one Pix transfer, from the regulator's time manual. The first row is the budget this paper scores detectors against.

Interval	p50	p95 / p99	What it bounds
Paying institution's own authorization ($t_1 - t'_0$)	**0.9 s**	**1.5 s** (p95)	**The time this paper gives a detector: the payer's institution has decided by the end of it**
Receiving institution's authorization ($t'_3 - t_2$)	1.4 s	2.3 s (p95)	A different participant's share, after the payer's decision is already made
Time inside the settlement system, primary channel	2.8 s	4.6 s (p99)	The clearing leg, which the paying institution does not control
Whole payer experience ($t_{6a} - t'_0$)	6.0 s	10.0 s (p99)	Confirmation reaching the payer's screen, after finality
Hard limit to settle, primary channel	40 s		A transfer not settled by then is rejected outright
Authorization of an order already suspected of fraud	30 min / 60 min		Business hours and otherwise: the wide window a diverted transfer moves into

Two consequences follow, and together they are the reason the paper measures what it measures. First, the deadline is the *first* row and not the last: a detector is not being asked to finish an investigation before the money moves, only to decide, inside the paying institution's own 1.5 s, whether this transfer is worth diverting into the 30-to-60-minute window. A detector that needs five seconds has not merely been slow, it has spent

the whole budget of a leg that is not its own. Second, none of these rows is a detection deadline that the regulation names as such; the manual bounds participants, not models. What it gives is the interval inside which the decision has to exist for the transfer to be held, which is what the metric in Section 3 scores.

E. The language-model prompts

Four language-model measurements appear in Table 4, and what separates them is what the prompt asks for, so the prompts belong next to the numbers. All four see the same event, rendered as one line from the four behavioral features the shared schema carries. The reasoning cap is 320 tokens on our own machine and 1024 in the cloud. The cap is a ceiling, not a schedule, and it binds on none of them: asked to deliberate in at most 320 tokens, the local model's median completion is 8, so it answers in 93–708 ms, inside the budget. Given the same reasoning prompt, the cloud models spend 171 and 284 tokens. That difference in how much they say, not any difference in what they were asked, is what puts their decisions outside the window.

Three details matter for reading the result. The verdict is parsed from the final `Probability:` line; a reply that never reaches one falls back to the threshold value, which the harness counts as a flag, so the harness reports how many completions fell back, and in the runs reported here none did. The reasoning prompt names the four signals explicitly, so the local model's low accuracy is not a failure to find the features but a failure to weigh them: given the same named features, the cloud models reach PR-AUC 0.846 and 0.849. And the cloud requests are issued with at most eight in flight, which shortens the run without changing what is measured, since each event's latency is its own round trip; the harness verifies this by timing the same requests serially and in parallel and reporting the ratio. Both prompts are reproduced in full below, with the event line as the harness renders it.

Both prompts open with the same rendered event, one line built from the four shared columns, shown here as ⟨event⟩: `Pix transfer: amount R$1250.00; payer device changed: yes; new payee (never paid before): yes; payer velocity: 7 transfers in the last hour.`

The *direct* prompt, verbatim:

```
You are a Brazilian Pix fraud-detection system. Assess one instant
transfer and output only a fraud probability between 0 and 1. ⟨event⟩ Is
this transfer fraudulent? Reply with a single probability between 0 and
1 (e.g. 0.85). Probability:
```

The *reasoning* prompt, verbatim, used both on our own machine and in the cloud:

```
You are a Brazilian Pix fraud-detection analyst. Decide whether one
instant transfer is fraudulent. ⟨event⟩ Think step by step about each
signal (a high amount, a changed device, a brand-new payee, and an
unusually high velocity each raise fraud risk), weigh them, then conclude.
Keep the reasoning to at most three short sentences, then STOP and output
the verdict on its own final line, written exactly as "Probability:
X" where X is a decimal number between 0 and 1 with two decimals (for
example, Probability: 0.85). Do not write anything after that line.
```