International Journal of Advanced Artificial Intelligence Research (IJAAIR)
eISSN: 3087-4270
Volume03 Issue11 | Page No: - 104-112

Received: 20 Sep. 2026, Revised: 30 Sep. 2026, Accepted: 11 November 2026, Published: 19 November 2026

DOI: https://doi.org/10.55640/ijaair-v03i11-07

Privacy-Aware Multi-Lender Routing for Point-of-Sale Finance using Reinforcement Learning: A Simulation Study

Amit Gamot
Independent Researcher Jersey City, NJ, USA

ABSTRACT: Many point-of-sale lending platforms run a credit application through a waterfall: prime lender first, then near-prime and sub-prime lenders if declined. Each fallback means a call to the checkout by a lender and another institution receiving the applicant's data. This paper investigates if a reinforcement learning policy can select the lender or choose not to route at all, solely based on the credit score and the requested amount. A Proximal Policy Optimization (PPO) agent is trained in a synthetic three-lender simulator to trade conversion against latency, lenders contacted and declines, where routing is modeled as a Markov decision process. Over 5 seeds, each evaluated on 10,000 held-out simulated applications, PPO raised net conversion from 62.7% to 70.8% relative to the waterfall, cut mean lender latency from 2,944 to 1,476 ms, and reduced lenders contacted from 2.41 to 0.74 per application. The conversion gain depends on the abandonment of slow sessions, which is assumed by the simulator. Without abandonment the waterfall converts more than PPO. The gains are also dampened by stronger baselines. 73.2% change for single-lender routing and parallel broadcast. The learned policy never takes into account the prime or near-prime lender. It sends applicants to the sub-prime lender or declines 26.2% of them outright It acts very much like a single-lender rule with a tuned score cut-off. Solving the same decision process exactly shows that the reward does favor matching applicants to lender tiers, yet neither longer training nor a price penalty led PPO to find that behavior. The paper also discusses the limitations of the simulator and the consumer protection issues raised by early declines and tier selection.

KEYWORDS: Point-of-sale finance, embedded lending, reinforcement learning, proximal policy optimization, data minimization, multi-lender routing.

1. Introduction

Point-of-sale financing lets a shopper apply for an installment loan at checkout. Many platforms that offer it do not lend from their own balance sheet. They pass the application to partner lenders, and a common way to do that is the waterfall. The application goes to a prime lender first. If that lender declines, it moves to a near-prime lender and then to a sub-prime one, until someone approves or the list runs out.

The waterfall is easy to build and easy to explain to merchants, but it has two costs. Each fallback is another network round trip while the shopper waits, and a shopper who waits too long may leave. Each fallback also sends the shopper's application to another institution, including ones that are going to decline it. Data protection law asks organizations to limit personal data to what a purpose actually needs [1],

[2], and sending an application to a lender that will almost certainly decline it is hard to defend on that test.

If a platform could know which lender will say yes, it wouldn't have to make the calls that are destined to fail. This paper asks if a reinforcement learning (RL) policy [3] can learn that prediction along with a rule for when to stop from the little information available before contacting any lender: a soft-pull credit score and the requested amount. The work is a simulation study. All lender, shopper and timing parameters are synthetic, and the conclusions are about what the method does in this environment, not any production system.

The paper makes four contributions:

- A Markov decision process (MDP) formulation of lender routing with an explicit Halt action and a reward that prices conversion, latency, the number of lenders that receive the application, and declines (Section III).

- An evaluation of a PPO agent [4] against six baselines and an exact reward-optimal policy, over 5 training seeds with a held-out evaluation environment in which every method sees the same applicants (Sections IV and V).
- An analysis of what the agent learns, with ablations, a sensitivity study on the abandonment assumption, and a price-aware variant. The gains over the waterfall come from two simple behaviors, using one lender and declining low scores early. Matching applicants to the right lender tier is not one of them, even though the exact solution shows the reward would pay for it.
- A discussion of the consumer-protection and regulatory questions these behaviors raise (Section VI).

2. BACKGROUND AND RELATED WORK

A. Routing designs in POS lending

Three designs act as examples. The sequential waterfall tests lenders in a fixed order from prime to sub-prime. It reaches out to an third lender only after a rejection so prime applicants get processed quickly while applicants who get rejected early go through several back and forths. Parallel broadcast sends the application to every lender at the time and holds on to the best approval. Its speed depends on the lender but every lender gets every application. Static rules send applications based on fixed limits, such, as sending scores above 700 to the lender. They are quick and clear. Someone has to update the limits manually as lenders change their requirements.

B. Learning-based routing

Choosing among endpoints based on the situation is similar to payment routing. In payment routing a platform selects a gateway, for every transaction. Agrawal and Patil describe a feedback-control approach to gateway selection in a production payment orchestrator and report success-rate gains of up to 1.15% over rule-based routing [5]. When each decision is a single step, the problem is a contextual bandit; Li et al. gave both a practical algorithm and a way to evaluate bandit policies offline from logged random traffic [6]. Lender routing with fallbacks is sequential, because a decline changes what the platform knows and what it can do next. This paper therefore uses the general RL setting [3] and PPO, a widely used policy-gradient method [4]. Any learned router depends on predicting approval from applicant attributes, which is the credit-scoring problem; Lessmann et al. benchmark the classifiers used for it [7]. Deep RL results can vary a lot across random seeds and implementations [8], so the experiments below report several seeds.

C. Regulatory context

The EU General Data Protection Regulation requires personal data to be "adequate, relevant and limited to what is necessary" for the purpose of processing [1]. California law similarly requires that collection and use of personal information be "reasonably necessary and proportionate" to the purpose [2]. In the United States, a platform's soft inquiry needs a permissible purpose under the Fair Credit Reporting Act, such as the consumer's written instructions [9]. According to Regulation B, creditors have a time limit of 30 days from when a completed application is submitted to inform applicants of the actions that have been taken. In situations where an application is submitted on behalf of a person with more than one creditor and no credit is granted in the end, the creditors that are taking adverse actions against the applicant would also be responsible for informing the applicant. This is important here because a policy that denies an applicant the chance to go through the application process before any of the lenders looks at them is actually making a decision on credit. There's no legal analysis in the paper; Section VI only points out the questions involved.

3. PROBLEM FORMULATION

A. Pre-qualification and state

When the shopper starts an application, the platform runs one consent-based soft inquiry. It also makes use of the credit score 'c' and the amount 'm' that is requested while omitting the personal details which include the name of the person, address, and social security number. The state at step t is

$$s_t = [\, c/850,\ m/5000,\ u_1,\ u_2,\ u_3 \,] \in [0, 1]^5, \quad (1)$$

where uk ∈ {0, 1} records whether lender k has already been contacted in this session. The vector is pseudonymous than anonymous and that helps keep the identity hidden A score and an amount, combined with the merchant and the time of purchase, could identify a person, and pseudonymised data remains personal data under the GDPR [1]. Only the routing decision uses the reduced state; a lender that is contacted receives the full application.

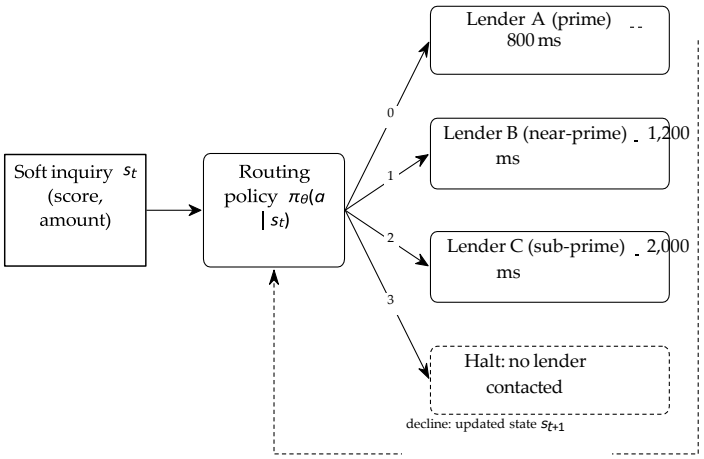


Fig. 1. Routing setup. After a decline the policy sees the updated state and may try another lender or halt. Latencies are those of the simulator.

TABLE I

Simulator Parameters

Quantity	Value
Credit score c	$N(632, 85^2)$, clipped to [300, 850]
Requested amount m	Uniform on [\$100, \$5,000]
Thresholds $\theta_A, \theta_B, \theta_C$	705, 635, 565 (effective score, Eq. 2)
Underwriting noise	$N(0, 18^2)$, independent per call
Lender latency A, B, C	800, 1,200, 2,000 ms
Abandonment	$p_{ab} = 0.35$ once total latency > 2,800 ms

B. Actions

The action set is $a_t \in \{0, 1, 2, 3\}$. Actions 0, 1, and 2 send the application to Lender A (prime), B (near-prime), or C (sub-prime). Action 3, Halt, ends the session without contacting any further lender, so if Halt is chosen first, no lender receives the application. Choosing a lender that was already contacted also ends the session, with a penalty. A session ends automatically once all three lenders have been tried. Fig. 1 shows the setup.

C. Simulated applicants, lenders, and abandonment

Table I lists the simulator's parameters. Scores are drawn from a normal distribution with mean 632 and standard deviation 85, clipped to [300, 850]. Amounts are uniform between \$100 and \$5,000. Each lender approves when an effective score clears its threshold θ_k:

$$c - 0.012\,(m - 1500) + \varepsilon > \theta_k, \qquad \varepsilon \sim N(0, 182), \tag{2}$$

with $\theta = (705, 635, 565)$ for A, B, and C and a fresh ε on every call. Larger amounts therefore need higher scores. After each call, if the session's cumulative lender latency exceeds 2,800 ms, the shopper abandons with probability $p_{ab} = 0.35$, even if that call was an approval.

There are two aspects of the simulator that set up the eventual results. First of all, the lenders have exactly the same thresholds, which means they are arranged in a hierarchy, so the applicants who are supposed to have loans from Lender A will most probably get money from Lender C as well. Secondly, due to those latencies, the abandonment can occur only when both Lender B and Lender C are contacted during the third call of the waterfall session.

The simulator also has no prices, so on its own it gives no reason to prefer a prime approval to a sub-prime one. All parameters were chosen by hand for illustration; none were fitted to real lender data.

D. Reward

The per-step reward is

$$R_t = \alpha\, \mathbb{1}[\text{conv}_t] - \beta\, \ell_t - \gamma\, \mathbb{1}[\text{contact}_t] - \lambda_{\text{rej}}\, \mathbb{1}[\text{decline}_t], \tag{3}$$

where conv_t means the lender approved and the shopper stayed, ℓ_t is the lender's latency in milliseconds, contact_t means a lender received the application, and decline_t means the lender declined. The weights are $\alpha = 100$, $\beta = 0.01$ per ms, $\gamma = 20$, and $\lambda_{\text{rej}} = 50$. Halt earns zero. Repeating a lender earns $-\lambda_{\text{rej}}$ and ends the episode, which makes it a penalized way to stop; this only matters when Halt is removed (Section V-C). For reference, a conversion at Lender A is worth $100 - 8 - 20 = 72$, a conversion at Lender C is worth 60, and a decline at Lender A costs 78.

The γ term counts lenders contacted, which serves as a proxy for data exposure. It discourages extra calls through the reward. It is not a constraint and does not guarantee that data is shared only when necessary.

4. EXPERIMENTAL SETUP

A. Training and evaluation

PPO is trained with Stable-Baselines3 [11] using the default multilayer-perceptron policy and default hyperparameters (learning rate $3 \times 10{-}4$, 2,048 steps per update, minibatch 64, 10 epochs, discount 0.99, GAE λ = 0.95, clip range 0.2, no entropy bonus) for 50,000 environment steps (51,200 in practice, since training stops after a whole update), on a Gymnasium [12] implementation of the simulator. Each of the 5 training seeds $s \in \{0, \ldots, 4\}$ trains on an environment seeded with s. The deterministic policy is then evaluated on a separate environment seeded with 10,000 + s, over 10,000 new applications. Every baseline is evaluated on the same five evaluation seeds.

Each environment draws applicants from one random stream and underwriting noise and abandonment from a second stream, both derived from its seed. Every method therefore sees the same 10,000 applicants for a given seed, and differences between methods come from routing and from underwriting noise, not from different applicant samples. Results are reported as mean ± sample standard deviation across seeds. The experiments used Python 3.13.16, Gymnasium 1.4.0, Stable-Baselines3 2.9.0, and PyTorch 2.14.1.

B. Baselines

- Sequential waterfall: A, then B, then C.
- Parallel broadcast: all three lenders at once. Latency is that of the slowest lender (2,000 ms), and the approval from the most prime lender is kept (A over B over C).
- Static score rules: one call to the most prime lender whose threshold the raw score clears, ignoring the amount. Scores at or below 565 are declined without a call.
- Single lender C: every applicant goes to Lender C.
- Single lender C with a tuned cut-off : applicants with a raw score above a cut-off go to Lender C once; the rest are declined without a call. For each seed the cut-off is chosen from 500 to 650 in steps of 5 to maximize the reward in Eq. 3 on the training environment, and the chosen value (580–585) is then evaluated on the held-out one.
- Score-aware cascade†: computes the noise-free effective score $e\hat{} = c - 0.012(m - 1500)$, starts at the most prime lender with $e\hat{} > \theta k$ (or at Lender C if there is none), falls back one tier after each decline, and declines up front when $e\hat{} < \theta C - 2\sigma = 529$.
- Exact optimum†: the reward-maximizing policy for Eq. 3.

Because the approval probability of each lender follows in closed form from Eq. 2, the MDP can be solved exactly for each applicant by backward induction over the eight possible sets of contacted lenders, treating approval, abandonment, and Halt as terminal.

The dagger marks provided the initial data. They make use of the approval rule, which is explained in this paper. These marks can be treated as points of reference rather than a competitive advantage; the exact optimum is the maximum value that can be achieved by any particular policy acting under this reward function.

C. Metrics

Net conversion means the percentage of applications that are accepted and there is no abandonment. Latency is the total of the times taken by the lender to respond in a session in terms of sequential policies, and for simultaneous broadcasting, the longest response time. A halted application counts as zero, and the platform's own soft inquiry is left out for every method. Lenders per application counts how many lenders received the application. Halted is the share of applications that no lender saw. The lender index is the mean tier of the approving lender among conversions (0 for A, 1 for B, 2 for C). Since sub-prime credit generally costs the borrower more, a higher index means more expensive credit, although the simulator itself has no prices. Reward is the mean per-application return under Eq. 3; it is not defined for parallel broadcast, which is not a sequential policy. Forgone conversions is the share of all applicants who were halted but would have been approved by a single call to Lender C, estimated with an independent noise draw.

5. RESULTS

A. Comparison with baselines

Table II and Fig. 2 compare the methods. Against the waterfall, PPO raises mean net conversion by 8.1 percentage points, lowers mean latency by 1,468 ms (50%), and makes 69% fewer lender contacts per application. PPO is better on all three measures in each of the five seeds.

TABLE II

MAIN COMPARISON (MEAN ± STANDARD DEVIATION OVER 5 SEEDS, 10,000 HELD-OUT APPLICATIONS EACH)

Method	Conversion (%)	Latency (ms)	Lenders / app	Halted (%)	Lender index	Reward (Eq. 3)
Sequential waterfall	62.7 ± 0.3	2,944 ± 5	2.41 ± 0.00	0.0 ± 0.0	1.05 ± 0.00	−98.6 ± 0.6
Parallel broadcast	73.2 ± 0.3	2,000 ± 0	3.00 ± 0.00	0.0 ± 0.0	1.19 ± 0.01	—
Static score rules	59.2 ± 0.4	1,105 ± 5	0.79 ± 0.00	21.4 ± 0.3	1.14 ± 0.01	22.8 ± 0.6
Single lender (C)	73.2 ± 0.4	2,000 ± 0	1.00 ± 0.00	0.0 ± 0.0	2.00 ± 0.00	19.7 ± 0.6
Single lender (C) + tuned cut-off	68.8 ± 0.6	1,442 ± 21	0.72 ± 0.01	27.9 ± 1.1	2.00 ± 0.00	38.3 ± 0.3
PPO orchestrator	70.8 ± 0.4	1,476 ± 18	0.74 ± 0.01	26.2 ± 0.9	2.00 ± 0.00	39.8 ± 0.4
Score-aware cascade†	71.9 ± 0.6	1,381 ± 8	0.90 ± 0.01	14.9 ± 0.4	1.25 ± 0.01	31.5 ± 0.8
Exact optimum†	69.6 ± 0.4	1,158 ± 5	0.72 ± 0.00	28.2 ± 0.3	1.43 ± 0.00	42.4 ± 0.4

†Informed baseline that uses the simulator's approval rule. Lender index: mean tier of the approving lender among conversions (0 = A, 1 = B, 2 = C); lower means lower-cost credit. Reward is the mean return per application under Eq. 3; it is not defined for parallel broadcast.

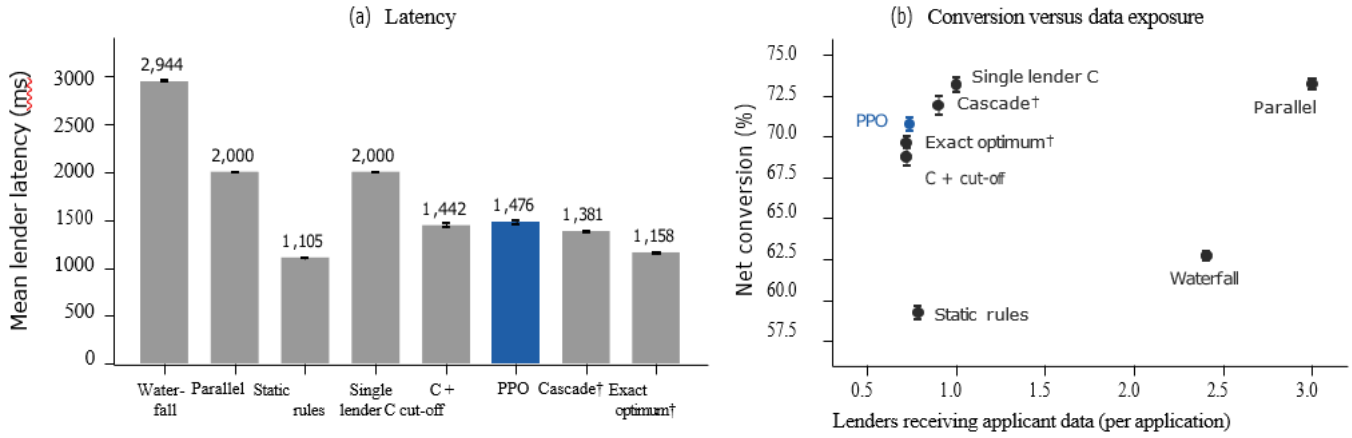


Fig. 2. (a) Mean lender latency per application. (b) Net conversion against the number of lenders that receive the applicant's data (price-aware PPO variants are in Table VI). Error bars show one standard deviation across seeds. †Informed baseline.

TABLE III

CONVERSION BY CREDIT-SCORE BAND (POOLED OVER 5 SEEDS)

Credit score	Share (%)	Conversion (%) Wfall	Par.	One C	Casc.†	PPO	PPO halted (%)
<565	21.4	4.5	7.1	6.7	6.5	2.9	95.0
565–634	29.9	53.0	77.1	77.2	76.4	72.0	19.6
635–704	29.4	90.6	99.9	99.9	96.7	99.9	0.0
≥705	19.3	100.0	100.0	100.0	100.0	100.0	0.0

Wfall = waterfall, Par. = parallel broadcast, One C = single lender C, Casc. = informed score-aware cascade. All methods see the same applicants, so the band shares are common to every column.

Against the other baselines the picture is less favorable. Parallel broadcast and the single-lender policy both convert 73.2%, 2.4 points more than PPO. The conversion rates are virtually the same because of the process, but their backgrounds of data exposure and price of credit are totally different. The lenders are hierarchical; thus, any candidate approved by lender A or B will likely get the approval from lender C as well; hence making contacts with all three does not make sense when making inquiries from lender C alone. In fact, their conversion rates in regard to the same loan applications

differ by less than half of a percentage point in both directions, which is within the underwriting water margin.

The single-lender rule with a tuned cut-off comes close to PPO on every measure: 68.8% conversion, 1,442 ms, and 0.72 lenders per application, against 70.8%, 1,476 ms, and 0.74 for PPO. PPO's 2.0-point edge in conversion has two sources: it halts fewer applicants (26.2% against 27.9%), and its halting decision depends on the amount as well as the score, which the cut-off ignores. The informed cascade converts 71.9% with 1,381 ms of latency. PPO is 1.1 points below it in conversion and 95 ms slower, and makes 18% fewer lender contacts. The static rules have the lowest latency but also the lowest conversion (59.2%), because they make one call on the score alone and never retry.

The rewards column presents a different image compared to the conversion. With respect to Eq. 3, PPO obtains 39.8 units of currency/application, which is more than the tuned cutoff value, which yields 38.3, the informed cascade method, and the sole lender policy. On the objective on which it was trained, PPO beats all methods presented in Table II excluding the exact optimum and is only 0.1 units of currency away from the optimum informed single call rule (Section V-B). The optimum achieves 42.4 units of the currency, which means that PPO achieves 94% of the reward achievable. However, the optimum in terms of the conversion from the point of view of conversions is worse than PPO (69.6%) but has a larger speed of execution (1,158 ms) and lower cost for the lender (lender index 1.43). Section V-B shows where the remaining gap comes from..

TABLE IV

ABLATIONS OF THE REWARD AND ACTION SPACE (MEAN ± STANDARD DEVIATION OVER 5 SEEDS)

Configuration	Conversion (%)	Latency (ms)	Lenders / app	Halted (%)	Lender index
Full reward, with Halt	70.8 ± 0.4	1,476 ± 18	0.74 ± 0.01	26.2 ± 0.9	2.00 ± 0.00
Full reward, 250k training steps	70.3 ± 1.2	1,459 ± 35	0.73 ± 0.02	27.1 ± 1.8	2.00 ± 0.00
No Halt action	73.2 ± 0.4	2,000 ± 0	1.00 ± 0.00	0.0 ± 0.0	2.00 ± 0.00
No privacy penalty ($\gamma = 0$)	72.1 ± 0.9	1,555 ± 42	0.78 ± 0.02	22.3 ± 2.1	2.00 ± 0.00
$\gamma = 0$ and no Halt	73.2 ± 0.4	2,000 ± 0	1.00 ± 0.00	0.0 ± 0.0	2.00 ± 0.00
No latency penalty ($\beta = 0$)	71.7 ± 0.4	1,534 ± 18	0.77 ± 0.01	23.3 ± 0.9	2.00 ± 0.00
No rejection penalty ($\lambda_{rej} = 0$)	72.4 ± 0.5	1,771 ± 24	0.89 ± 0.01	11.5 ± 1.2	2.00 ± 0.00
Conversion reward only	73.2 ± 0.4	2,000 ± 0	1.00 ± 0.00	0.0 ± 0.0	2.00 ± 0.00

TABLE V

SENSITIVITY TO THE ABANDONMENT PROBABILITY (MEAN OVER 3 SEEDS)

	Conversion (%)				PPO only		
p_{ab}	Wfall	Par.	One C	PPO	Lat. (ms)	Len./app	Halt (%)
0.00	73.5	73.5	73.4	71.2	1,492	0.75	25.4
0.20	67.6	73.5	73.4	70.0	1,445	0.72	27.7
0.35	62.9	73.5	73.4	71.0	1,483	0.74	25.9
0.50	58.3	73.5	73.4	70.9	1,479	0.74	26.0

The 10.5-point gap between the waterfall and parallel broadcast comes almost entirely from abandonment. Both methods can reach every lender, but the waterfall reaches Lender C only after 2,000 ms of earlier calls. Section V-D confirms this by removing abandonment.

B. What the policy learned

Across the five seeds the PPO policy never sends an application to Lender A or B as its first action. It sends 73.8% of applicants to Lender C and halts the rest. In effect, it is the single-lender policy plus a score cut-off below which it declines. Table III breaks this down by score band. The policy halts 95.0% of applicants scoring below 565, where approval is rare, and 19.6% of the 565–634 band, where most applicants would have been approved. Nobody scoring 635 or above is halted.

Halting has a cost. Among the halted are 2.4% of all applicants (±0.4) who would have been approved by a single call to Lender C. Since PPO only ever calls Lender C, this figure closely matches its 2.4-point conversion gap to the single-lender policy. Most of the loss falls in the 565–634 band, where PPO converts 72.0% against 77.2% for single-lender routing.

The tier choice has a cost too. Applicants scoring 705 or above make up 19.3% of the population, and on average 75.9% of them would be approved by Lender A on a single call (the rest fall below A's threshold because of a large amount or underwriting noise). PPO sends every one of them to the sub-prime lender. It is tempting to read this as PPO leaving 12 points on the table, since a conversion at A is worth 72 and one at C is worth 60. The

exact solution shows that the trade is closer than that. For these applicants, calling A first is worth 52.2 on average once the best follow-up is taken into account,

against 60.0 for calling C first, because a call to A carries a 24.1% chance of a decline that costs 78. Calling A first beats calling C first for only 58% of them. Making the initial contact with B ends up being a more successful method than using either alternative on average (67.9), whilst the optimum initial call under these circumstances is A in 46% of cases and B in 54% of cases, and never C. The difference of 12 points between A and C in Eq. 3 can be entirely attributed to the latency term, so the payment relies on the speed of the response, rather than on its being cost-effective.

The PPO approach does not succeed in this strategy at this level of training as the strategy accounts for almost the totality of losses experienced. The best option giving just a single call to Lender C brings the average per application to 39.9, which is only 0.1 point higher than that of the PPO. The remaining 2.5 points between the strategy applied and the optimal performance are attributed strictly to calls to A and B. The optimal decision level is almost equal to that of the PPO (28.2% versus 26.2%), but 13% of approvals go to Lender A and 32% to B. In contrast, the cascade places 20% with A, 36% with B, and 45% with C.

C. Ablations

Table IV varies the reward and the action space. Removing the Halt action reproduces the single-lender policy (73.2%, 2,000 ms, 1.00 lenders per application). After a decline the agent ends the session by repeating Lender C. Repeating costs λrej = 50, less than the expected loss from a second call, so without Halt it becomes a penalized way to stop. Training on the conversion reward alone gives the same single-lender policy. Removing the privacy penalty makes the policy somewhat less strict: the halt rate falls to 22.3% and conversion rises to 72.1%. The rejection penalty matters more. Without it, the halt rate drops to 11.5% and the policy contacts 0.89 lenders per application. Of PPO's total reduction against the waterfall (1.67 fewer lenders per application), 1.41 comes from making a single call and 0.26 from halting. Relative to the single-lender policy, the remaining saving comes from declining applicants early, which the rejection penalty encourages more than the γ term does. Removing the latency penalty has a similar effect (halt rate 23.3%, conversion 71.7%). For a single call to Lender C the latency and contact terms each cost 20, so dropping either one lowers the bar for routing by the same amount.

Training for five times as long (250,000 steps; 251,904 in practice) does not change the behavior. The policy still sends every routed applicant to Lender C (lender index 2.00) and converts 70.3% (±1.2).

TABLE VI

PRICE-AWARE REWARD VARIANTS (MEAN OVER 5 SEEDS)

Method	Conv. (%)	Lat. (ms)	Lenders / app	Halted (%)	Approved by A / B / C (%)	Lender index	Reward
PPO, no price term	70.8	1,476	0.74	26.2	0 / 0 / 100	2.00	39.8
PPO, price weight 15	68.9	1,417	0.71	29.1	0 / 0 / 100	2.00	18.9
PPO, price weight 30	23.1	282	0.24	76.5	0 / 100 / 0	1.00	8.5
Exact optimum†, no price term	69.6	1,158	0.72	28.2	13 / 32 / 56	1.43	42.4
Exact optimum†, price weight 15	67.4	1,064	0.70	31.2	17 / 34 / 48	1.31	28.4
Exact optimum†, price weight 30	37.5	424	0.40	61.0	34 / 66 / 0	0.66	16.6
Sequential waterfall (reference)	62.7	2,944	2.41	0.0	26 / 42 / 32	1.05	−98.6
Parallel broadcast (reference)	73.2	2,000	3.00	0.0	23 / 36 / 41	1.19	—

Approval shares and lender index are pooled over the conversions of all five seeds. Reward is the mean return under the reward used to train or solve that row (Eq. 3 with the stated price weight); it is not defined for parallel broadcast. †Informed policy that uses the simulator's approval rule.

D. Sensitivity to the abandonment assumption

The least solid part of the simulator is the principle of abandonment. As shown in Table V, pab varies from 0 to 0.5 by increments of 0.25. Each of these experiments utilized three seeds which were reused in the cases where pab was equal to 0.35 (i.e., seeds 0-2 of the main experiments). Assuming no abandonment, the conversion benefits of the waterfall are at 73.5% similar to the conversion achieved by the parallel broadcast and higher than the conversion of PPO (71.2%). Meanwhile, the time of the waterfall is not influenced by pab, therefore, the trick of PPO is still outstripping it in terms of speed and the amount of data used. However, even in this case, the conversion achieved by PPO originates from the assumption of abandonment. The results for PPO remain unchanged in different experiments, which is what can be expected from a policy that never makes a

second move.

E. Adding a price term

To see if the tier issue is merely absent from the goal, the reward will gain an additional penalty of w . k for a tier-k conversion. Table VI provides the results obtained in this case and compares these results to the true optimum for every weight. For w = 15 the policy is the same: there is only Lender C with 68.9% conversions. The optimum for the same reward forwards 17% of the applicants to Lender A and achieves a profit of 28.4 compared to PPO's 18.9. For w = 30 the conversion at Lender C becomes valueless (100 - 20 - 20 - 60 = 0), resulting in unstable training. The model stops using C in 4 of 5 seeds, only uses Lender B, rejecting somewhere between 63% and 82% of applications, and converts from 17.7% to 35.7% of applicants.

Averaged over seeds, conversion falls to 23.1 ± 14.6%. The optimum for this reward converts 37.5% and places 34% of its approvals with Lender A. At neither weight does PPO use Lender A, even though a conversion there is worth 72 under both. A price term may be necessary for an RL policy to prefer cheaper credit, but at this training budget it was not sufficient. The difficulty does not seem to be learning where approvals happen: without a price term PPO comes within 0.1 of the best single-call rule for Lender C, so it has learned that lender's boundary well. What it never does is try Lenders A or B often enough to learn that they pay off, which looks like an exploration problem rather than a modeling one.

6. Discussion

A. What the results support

In this simulator, avoiding sequential fallbacks and declining very unlikely applicants early cuts latency and data exposure sharply compared with a waterfall. The PPO agent finds both behaviors, but simple policies capture them too. The single-lender policy already removes the fallbacks, and adding a score cut-off tuned for the same reward comes within 2.0 points of PPO's conversion at similar latency and exposure. On the reward itself PPO does a little better than every baseline that could be built without the approval model, and better than the informed cascade, but it stays 2.6 points below the exact optimum. That gap is almost all tier matching, which PPO never learned.

Two points follow. The informed cascade beats PPO on conversion, latency, and credit cost while contacting a few more lenders, and the exact optimum shows that the reward itself rewards sending strong applicants to the faster, more prime lenders. The optimum, however, needed no learning: it is a fixed function of score and amount that follows from known approval probabilities. A case for RL here would need an environment in which those probabilities are unknown and drift, or lenders differ in more than one dimension, so that the optimum cannot simply be written down.

B. Consumer outcomes

The act of sending eligible candidates to lenders specializing in sub-prime loans could result in them receiving loans with higher interest rates compared to their actual qualifications. In equation 3, this matter is overlooked by the reward function as well as the measurements made by latency, conversion rates or the concept of data exposure. Therefore, in order to implement any criterion of evaluation in real life, it should take into account a loan price or a clear ban for sending an applicant to a lender expected to give a go ahead. The issue of decline by score band is also a matter of fairness.

C. Halting is a credit decision

If the policy stops short of reaching out to any lender, then the system has actually rejected the application without informing the applicant about it. If that does entail some notification obligations under Regulation B [10], then both the notification giver and notification reasons would need to be checked with legal advisers. All these should be resolved before implementing a solution like this one. The same is true for the legitimate purpose of conducting soft inquiries [9].

D. Data minimization

It is convenient to count lenders approached, but the legal principles state whether doing so is needed and proportionate [1], [2], and counting on its own cannot answer that question. The diminished routing state is also personal data, and the process of not proceeding reduces exposure by declining people, which incurs its own costs. A smaller number of lenders is a good indication; it does not prove compliance.

E. Limitations

The data used in this experiment is synthetic, which means that the parameters are predetermined and not fitted in any way. The underwriting problem is one-dimensional; hence, the lenders' processes are hierarchical and independent across calls. But lenders in real life operate in a correlated manner. There is also a lack of price points, lender capacity constraints, or terms of the offer. The abandonment process employs an arbitrary threshold. The latencies are static instead of random. PPO was conducted with default hyperparameters without any sort of tuning, and there were only five seeds per setting used for experiments.

7. Conclusion And Future Work

In a three-lender model developed through PPO

routing policy, it has proven itself to be more efficient than sequential waterfall in latency, data exposure, and conversion speed in instances where slow sessions do not contribute to abandonment. For this purpose, PPO routing policy uses just one lender and turns down many low scored applications at the very beginning. However, tuned single-lender cut-off technique almost achieves the same result. It, indeed, gives up some conversion compared to parallel and singly routed lenders. At the same time, PPO routing stands out from the previous models as it has the award equal to 94% of the exact optimum. What PPO routing cannot do is tier matching, since it has not learned sending applicants to the correct lender tier no matter what length of training was applied, or whether there is charge in the prices or not.

The question would become more definitive with several extensions. It is required to fit per-lender approval models and latencies to the actual decision logs. The goal must include the offer price along with strict restrictions against an applicant being routed to a more expensive lender than he/she can afford. Since each session includes no more than three callbacks from lenders, contextual-bandit methods with offline assessment on logged data [6] are fitting for this task. Lastly, any policy that rejects applicants at the initial stage requires a fairness assessment along with a clear delineation of notice obligations..

Code Availability

The simulator along with the experiment scripts needed for the generation of tables and figures besides the raw results for each seed appear in the supplementary materials.

References

1. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 (General Data Protection Regulation), Art. 5(1)(c) and Recital 26, Official Journal of the European Union, L 119, pp. 1–88, May 2016.
2. California Consumer Privacy Act of 2018, as amended by the California Privacy Rights Act of 2020, Cal. Civ. Code § 1798.100(c).
3. R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.
4. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," arXiv:1707.06347, 2017.
5. A. Agrawal and H. Patil, "A control-theoretic approach to dynamic payment routing for success rate optimization," arXiv:2510.16735, 2025.
6. L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in Proc. 19th Int. Conf. World Wide Web (WWW), Raleigh, NC, USA, 2010, pp. 661–670, doi: 10.1145/1772690.1772758.
7. S. Lessmann, B. Baesens, H.-V. Seow, and L. C. Thomas, "Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research," Eur. J. Oper. Res., vol. 247, no. 1, pp. 124–136, 2015, doi: 10.1016/j.ejor.2015.05.030.
8. P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," in Proc. AAAI Conf. Artif. Intell., vol. 32, no. 1, 2018, pp. 3207–3214, doi: 10.1609/aaai.v32i1.11694.
9. Fair Credit Reporting Act, 15 U.S.C. § 1681b(a)(2)–(3), "Permissible purposes of consumer reports."
10. Equal Credit Opportunity Act (Regulation B), 12 C.F.R. § 1002.9(a)(1)(i) and (g), "Notifications."
11. A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-Baselines3: Reliable reinforcement learning implementations," J. Mach. Learn. Res., vol. 22, no. 268, pp. 1–8, 2021.
12. M. Towers et al., "Gymnasium: A standard interface for reinforcement learning environments," in Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track, 2025, arXiv:2407.17032.