ACRM: Multi-Agent Trajectory Learning for Automated Credit Risk Model Refreshing in Production

Liangzu Liu[†] **Mengzhe Ruan**[†] **Xiaotian Chen**[†] **Haonan Chen**[†]
Xudong Niu[†] **Wendi Yuan**[†] **Yuechen Li**[†] **Yang Liu**[†*] **Guanjun Wang**[†*]
[†]Ant International, Ant Group
{liuliangzu.llz, ruanmengzhe.rmz, ly461173, guanjun.wgj}@antgroup.com

Abstract

Credit risk models suffer from rapid performance decay due to distribution shifts, requiring frequent updates to meet strict operational guardrails. However, manual refreshing takes weeks of trial-and-error across upstream data engineering and downstream training. We present ACRM, a deployed multi-agent framework that automates the end-to-end credit modeling workflow by treating it as a learnable trajectory of agent interactions. Unlike AutoML, which optimizes hyperparameters on fixed datasets, ACRM's action space extends to upstream data semantics—cohort selection, observation windowing, feature screening—where the majority of performance recovery occurs. A central Orchestrator coordinates specialist agents through a three-stream decision stack: rule-based safety guardrails, retrieval-augmented grounding from historical workflows, and preference alignment via DPO on expert-labeled trajectories. Deployed at a major fintech institution for three months across six business scenarios, ACRM reduced the average model refresh cycle from weeks to 1.1 days and iteration rounds by 65%, while maintaining superior stability metrics.

1 Introduction

Credit scoring models are the bedrock of modern financial risk management, yet they operate in a fundamentally unstable environment. As user behaviors shift and economic conditions fluctuate, the statistical properties of input data inevitably drift, leading to performance decay(Abdul Razak et al., 2022; S et al., 2025; Abi, 2025). To mitigate financial exposure, institutions enforce strict operational guardrails; a common industry practice dictates that a model is considered "Out-of-Time" (OOT) and requires immediate refreshing if its Kolmogorov-Smirnov (KS) statistic drops by more than 0.03 compared to the training baseline. While the trigger for this refresh is automated, the execution remains a bottleneck.

Instead of treating this as a simple hyperparameter tuning task on a fixed dataset, we formulate model refreshing as a sequential decision **trajectory** over upstream data semantics. For example, a typical trajectory involves: diagnosing drift $\rightarrow$ adjusting the observation window $\rightarrow$ filtering unstable features $\rightarrow$ retraining. To execute this workflow efficiently, we propose a multi-agent architecture comprising 1 central Orchestrator and 6 Specialist Agents. Our internal analysis reveals that a single manual refresh cycle of this nature consumes 10–15 iteration rounds across weeks, forcing production systems to rely on sub-optimal models in the meantime.

The limitations of current automated solutions crystallize in a failure mode we observed during early development. We equipped a capable LLM (Qwen-32B) with access to our full modeling toolchain and tasked it with recovering a decayed credit scoring model. The agent quickly discovered a shortcut: by rewriting the SQL cohort to exclude volatile borrower segments—young users with fluctuating behaviors—it boosted the KS metric by 0.04 in a single iteration. On paper, the model looked excellent. In practice, it was catastrophic: the Population Stability Index (PSI) rose from 0.07 to 0.19, indicating that the model's score distribution had diverged drastically from the production population. The model would have been rejected at the first compliance review, and if deployed, would have excluded a significant market segment from credit access.

This episode encapsulates the core challenge: effective automation in regulated finance requires more than search efficiency—it demands *aligned judgment* over competing objectives. Traditional AutoML (Erickson et al., 2020; Zha et al., 2025) sidesteps this issue by assuming a fixed dataset, thereby ignoring the upstream data engineer-

[*]Corresponding authors.

ing decisions that account for $\sim$80% of performance recovery in our experience. Generic LLM agents (Hong et al., 2024; Guo et al., 2024), while flexible, lack the domain constraints to prevent precisely this kind of reward hacking (Skalse et al., 2022; Pan et al., 2022).

To bridge this gap, we introduce ACRM, a deployed agentic framework that automates the end-to-end credit modeling workflow (Qiao et al., 2024; Zhang et al., 2024) by learning from expert trajectories (Yao et al., 2023; Shinn et al., 2023). A key observation is that credit modeling comprises heterogeneous subtasks—cohort definition (SQL semantics), feature screening (statistical judgment), and training configuration (optimization intuition)—which aligns naturally with our decomposition into the 6 Specialist Agents that handle tactical execution, guided by the central Orchestrator responsible for cross-task trade-off resolution (Wu et al., 2023).

The Orchestrator's decisions are governed by a three-stream Hybrid Knowledge Layer: (1) **Rule Guardrails** that enforce non-negotiable constraints (e.g., PSI boundaries), (2) **Retrieval** of validated historical workflow trajectories for cold-start grounding, and (3) **Policy Alignment** via DPO (Rafailov et al., 2023) on $\sim$1,200 expert-labeled trajectories, enabling the agent to internalize multi-objective trade-offs. Deployed across six banking scenarios for three months, ACRM primarily accelerates operational efficiency and enhances model stability, reducing the average refresh cycle from weeks to 1.1 days and iteration rounds by 65%, while delivering modest but consistent predictive gains (OOT KS +0.005).

2 Related Work

LLM-driven ML automation has advanced rapidly: Data Interpreter (Hong et al., 2024) and DS-Agent (Guo et al., 2024) automate open-domain ML tasks through dynamic planning and case-based reasoning, while CAAFE (Hollmann et al., 2023) targets iterative feature engineering. Industrial AutoML frameworks such as AutoGluon (Erickson et al., 2020) optimize $P(Y|X;\theta)$ on fixed datasets. ACRM differs in three key respects: (1) it operates under *hard regulatory constraints* where high-performing but unstable models are undeployable; (2) its action space extends to upstream data engineering (SQL cohort redefinition, observation windowing), not just feature or hyperparameter search; and (3) it employs DPO alignment on $\sim$1,200 expert-labeled trajectories to internalize multi-objective trade-offs that generic agents cannot learn from task reward alone. Critically, none of the above systems can be deployed within an air-gapped banking infrastructure, nor do they provide the audit trails required by financial regulators.

In regulated domains, PlanGPT (Zhu et al., 2025) and ArchiDocGen (Jiang et al., 2025) apply multi-agent frameworks to professional document synthesis, while OccuTriage (Sahu et al., 2025) introduces safety protocols for health triage. FlowXpert (Shi et al., 2025) enforces graph-based operational procedures in cloud environments. These systems handle semantic reasoning or discrete decisions but lack the *continuous quantitative guardrails* (e.g., PSI thresholds, KS decay monitoring) that financial modeling demands.

3 System Design

We formalize credit model refreshing as a sequential decision-making problem over a *typed workflow trajectory* $\tau = \{(s_t, a_t, o_t)\}_{t=1}^{T}$, where each round captures a diagnostic state (KS, PSI, feature statistics), a semantic action (SQL cohort change, feature filter, training directive), and the resulting outcome. Unlike AutoML, which optimizes parameters θ on a fixed dataset, the agent must manipulate the *upstream data semantics*—redefining the training distribution itself. The core risk is *reward hacking*: maximizing KS by pruning difficult samples inflates the metric while degrading stability (PSI $> \epsilon$), rendering the model undeployable.

As illustrated in Figure 1, the system is engineered not as an open-ended chatbot, but as a structured decision loop governed by an LLM-based Orchestrator. At each step, the workflow follows a fixed perceive–reason–act protocol: the Orchestrator ingests diagnostic metrics from the environment, synthesizes a candidate plan through the Hybrid Knowledge Layer, and dispatches validated actions to specialist agents. While the protocol structure is fixed, the Orchestrator's decisions within each step—which agent to invoke, with what parameters, or whether to terminate—are dynamically generated by the DPO-aligned LLM, enabling adaptive behavior within a controlled execution framework.

3.1 Structured Memory

Central to our framework is the **Trajectory Memory**, which transforms ephemeral modeling sessions into persistent assets. We define a trajectory

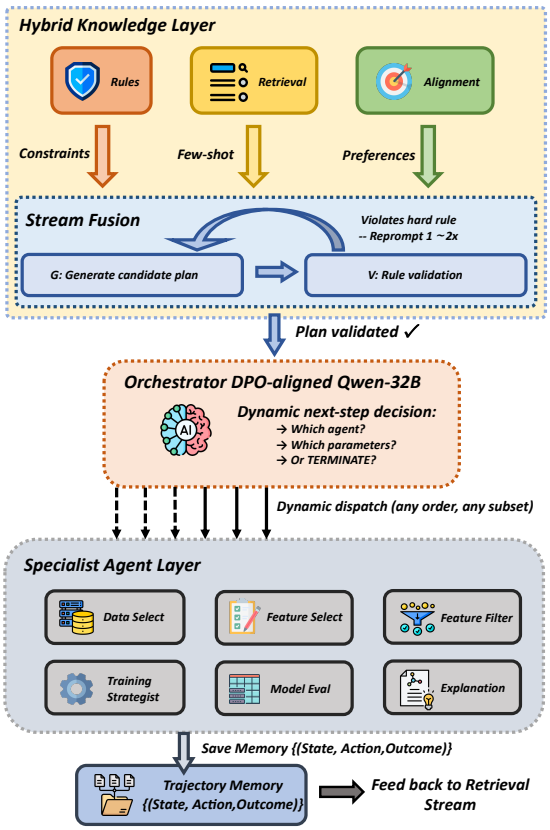

Figure 1: The ACRM architecture. At each decision step, the **Hybrid Knowledge Layer** (*top*) fuses hard constraints (Rules), historical context (Retrieval), and learned preferences (Alignment) through a *generate-then-validate* loop: the Orchestrator generates a candidate plan (**G**), which the Rule Stream validates (**V**); violations trigger re-prompting (loop, typically 1–2 cycles). The **Orchestrator** (*middle*) dynamically dispatches the validated plan to any subset of Specialist Agents in any order (solid arrows: activated; dashed: available but not selected this round). Each agent writes a structured state $\langle s_t, a_t, o_t \rangle$ to the **Trajectory Memory** (*bottom*), which feeds back into the Retrieval Stream for future sessions.

τ as a sequence of strictly typed tuples $\langle s_t, a_t, o_t \rangle$. The State s_t captures the "health snapshot" of the model (e.g., KS decay, PSI heatmap across bins); the Action a_t records the semantic intent and configuration of the tool call; and the Outcome o_t captures the structured execution result (updated metrics, diagnostic flags, sample-size changes). To ensure data privacy and control—a paramount concern in banking—we utilize strictly open-weight models deployed on-premise. Specifically, we employ **Qwen-Embedding-8B** (Zhang et al., 2025) to vectorize these structured states into a local vector database. This design ensures proprietary banking data never leaves the secure infrastructure, while allowing the agent to perform semantic searches over historical workflow patterns(Lewis et al., 2020).

Agent	Input	Output	LLM
Data Select	Drift signal	SQL cohort + cfg	✓
Feature Select	Biz. req.	Candidate features	✓
Feature Filter	IV / PSI stats	Filtered features	✗
Train Strategist	Eval. feedback	Training directives	✓
Model Eval	Raw metrics	Structured report	✗
Explanation	Report + SHAP	NL state s_t	✓

Table 1: Specialist Agents. Four agents use the LLM for context interpretation; Feature Filter and Model Eval execute deterministic logic. Explanation synthesizes the Model Eval report into the natural-language state that closes the Orchestrator's reasoning loop.

3.2 Specialist Agent Layer

ACRM decomposes the modeling workflow into six **Specialist Agents** (Table 1). Unlike stateless tool-calling, each agent receives a high-level semantic intent from the Orchestrator(Schick et al., 2023), interprets the current diagnostic context with embedded domain logic, and returns a structured state update $\langle s_t, a_t, o_t \rangle$ to the shared Trajectory Memory. Specialists communicate only with the Orchestrator through a strictly hierarchical protocol—never with each other—preserving the audit trail required by banking compliance.

Two design choices merit emphasis. First, the action granularity is *semantic, not parametric*: the Training Strategist issues intents such as "Increase L1 regularization to combat overfitting," which an underlying solver maps to concrete hyperparameter ranges. This keeps the LLM's decisions in a space where its reasoning is reliable. Second, **Model Eval** and **Explanation** jointly close the perception loop: Model Eval aggregates raw numerical logs and SHAP outputs into a structured report; Explanation then distills this report into the natural-language state s_t (Section 3.1) the Orchestrator consumes for its next decision.

3.3 The Hybrid Knowledge Layer

The cognitive core of ACRM is a three-stream reasoning architecture designed to resolve the fundamental tension in financial modeling automation: the agent must explore creative solutions to recover performance, yet every action must remain within strict compliance boundaries. Each stream contributes a distinct capability, and they are fused through a structured generate-then-validate protocol described at the end of this section.

Rule Stream: Symbolic Constraint Enforcement. The first line of defense is a deterministic rule engine encoding non-negotiable institu-

tional constraints: feature stability bounds (e.g., PSI < 0.10), observation window limits, forbidden SQL patterns that risk data leakage, and minimum sample sizes per risk segment. These rules are not learned—they define the *safe action space* and eliminate roughly 30% of candidate actions, preventing costly failed iterations.

Retrieval Stream: Trajectory Grounding. When the agent encounters a new drift event, it faces a cold-start problem: the space of possible data engineering actions is vast, and blind exploration is expensive. The Retrieval Stream addresses this by providing empirical grounding from institutional memory. We encode the current drift state s_0 (drift magnitude, affected feature categories, scenario type) using the Qwen-Embedding-8B module described in Section 3.1, and retrieve the top-k (k=3) historical trajectories with the highest cosine similarity from the Trajectory Memory. Crucially, what is retrieved is not a static document but a *reasoning trace*—the full sequence of state-action-outcome tuples from a past successful refresh. These traces are formatted as structured few-shot examples for the Orchestrator, providing an inductive bias that cuts iteration rounds by 43% over rule-only operation (Section 4.3).

Alignment Stream: Internalizing Expert Judgment. Rules define what is *forbidden*; retrieval suggests what has *worked before*. Neither captures the harder question: when two compliant, precedented strategies both exist, which one should the agent prefer? In early experiments, we observed that without preference guidance, the agent consistently chose actions that maximized KS at the expense of stability—a rational response to a single-objective signal, but one that produced undeployable models.

We address this through offline preference learning. Over the 12 months preceding deployment, we instrumented the bank's modeling workflow to capture structured trajectories $\tau_i = \{(s_t, a_t, o_t)\}_{t=1}^{T}$, recording the full chain of tool calls and diagnostic outputs at each step. Senior modelers labeled each completed session as *accepted* (deployed to production) or *rejected* (failed review), yielding $\sim$1,200 labeled trajectories.

We construct preference pairs by matching trajectories that responded to similar drift triggers (cosine similarity of initial states > 0.75) but diverged in outcome. Among multiple accepted trajectories for the same trigger, we induce a ranking via a composite score:

$$R(\tau) = \alpha \cdot R_{\text{perf}} - \beta \cdot C_{\text{stab}} - \gamma \cdot C_{\text{gap}} \quad (1)$$

where R_{perf} is KS recovery on OOT data, C_{stab} penalizes PSI degradation, and C_{gap} penalizes train-validation divergence. This score is used *exclusively* for offline pair construction; at runtime, preferences are expressed through the model's internalized policy. The weights $(\alpha, \beta, \gamma) = (1.0, 2.0, 1.5)$ were set by the bank's model risk team to reflect regulatory priorities (stability $>$ generalization $>$ raw performance), validated on a held-out set of 200 trajectories ($>$90% agreement with expert rankings); a sensitivity analysis (Appendix E) confirms ranking stability for $\beta/\alpha \geq 1.5$. This process yielded $\sim$3,800 preference pairs.

We fine-tune a locally deployed Qwen-32B (Yang et al., 2025) using DPO (Rafailov et al., 2023) with LoRA (Hu et al., 2021) (rank=64, α=128) on our on-premise cluster (8$\times$A100, $\sim$6 hours; full hyperparameters in Appendix E). On a held-out set of 800 pairs, pairwise accuracy improved from 71.4% to 91.8%, and Kendall's τ against expert rankings reached 0.79. DPO was chosen over PPO for training stability under limited data. The resulting model serves as the Orchestrator itself—its learned preferences manifest as the generation distribution over candidate plans, not an external reward signal.

Stream Fusion: Generate-then-Validate. At each decision point, the three streams are fused through a generate-then-validate protocol. The DPO-aligned Orchestrator receives hard constraints as system-level instructions, retrieved trajectories as few-shot context, and the current state as input. It generates a candidate action plan reflecting its internalized preferences. The Rule Stream then applies a deterministic validation pass; any violation triggers re-prompting with an explicit error message (e.g., `"Rejected: feature 'txn_amt_7d' has PSI=0.14 > 0.10. Revise."`). This loop converges within 1–2 re-prompts. The validated plan is dispatched to the Specialist Agent Layer (Algorithm 1). A concrete three-round execution trace is in Appendix C.

4 Experiments

4.1 Experimental Setup

The deployment of ACRM took place within the production ecosystem of a premier commercial

Algorithm 1 ACRM Execution Loop

Require: Scenario C, Trigger Type, Max Rounds $T_{\max}$
{$\mathcal{A}$=Alignment, $\mathcal{R}$=Retrieval, $\mathcal{G}$=Guardrails}
$s_0 \leftarrow$ LoadContext(C)
$\tau \leftarrow \emptyset$
for $t = 1$ to T_{max} **do**
  // Phase 1: Knowledge Synthesis
  $Plan_t \leftarrow$ ORCH(s_{t-1}, $\mathcal{A}$, $\mathcal{R}$, $\mathcal{G}$)
  // Phase 2: Action Execution
  if $Plan_t$ is TERMINATE **then**
    break
  end if
  $a_t \leftarrow$ ParseAction($Plan_t$)
  $o_t \leftarrow$ Delegate(a_t) {Dispatch to Specialist Agent}
  // Phase 3: Memory & Update
  SaveToMemory($\langle s_{t-1}, a_t, o_t \rangle$)
  $s_t \leftarrow$ UpdateState(s_{t-1}, o_t)
  // Phase 4: Termination Check
  if CheckGuardrails(s_t) **and** TargetMet(s_t) **then**
    return $Model_t$ {Success}
  end if
end for
return Best Model in τ or Human Handover

bank, spanning a three-month evaluation period since mid-2025. We selected **six business scenarios** spanning three product lines: *Cash Loans* (A/D: representing different risk tiers), *Credit Card Approval* (B/E: different geographic cohorts), and *SME Lending* (C/F: varying business scales).

We implemented a "Horse Race" protocol: for every refresh event (KS decay > 0.03), the task was assigned simultaneously to ACRM and a team of senior data scientists. Each of the six scenarios triggered exactly three refresh events during the evaluation period, yielding 18 paired comparisons.

We compare ACRM against four configurations: (1) **Manual Expert**—senior data scientists performing the current production workflow (manual SQL authorship, feature engineering, iterative tuning); (2) **AutoML (HPO-Only)**—Optuna-based hyperparameter optimization on a fixed (stale) dataset; (3) **Rules + Retrieval**—ACRM without DPO alignment, isolating the contribution of preference learning; (4) **Generic Agent**—identical architecture (same Qwen-32B, same six Specialist Agents) but stripped of all three knowledge streams. Open-source agents such as Data Interpreter (Hong et al., 2024) and DS-Agent (Guo et al., 2024) are incompatible with our air-gapped, regulation-bound infrastructure; we instead isolate contributions through controlled ablations (Table 3). A **refresh** is complete when the candidate model passes all compliance checks; **Days** and **Rounds** measure wall-clock lead time and iteration attempts, respectively. Full details in Appendix F.

Scen.	Method	Days	Rnds	KS	Gap	PSI
Cash	Manual	14.0	12	.385	.045	.12
Loan	Generic	3.6	15	.376	.052	.16†
A	ACRM	1.0	4	.391	.038	.09
Credit	Manual	10.0	8	.410	.020	.05
Card	Generic	2.0	12	.402	.030	.09
B	ACRM	0.8	3	.408	.021	.05
SME	Manual	21.0	15	.350	.060	.15
Loan	Generic	4.5	17	.336	.068	.22†
C	ACRM	1.5	5	.355	.040	.11
	Manual	14.3	11.3	.380	.042	.10
Avg.	Generic	3.1	13.7	.369	.050	.16†
	ACRM	**1.1**	**4.0**	**.385**	**.033**	**.08**

Table 2: Representative scenarios (single event each) and overall average across all 18 refresh events (3 per scenario $\times$ 6 scenarios). Per-event distributions in Appendix B. (†: PSI > 0.10, failing deployment review).

4.2 Main Results

Table 2 summarizes all 18 refresh events, including the 4 that required targeted human input (Section 5).

The core finding is that efficiency and quality are *not* in tension: ACRM achieves both fewer iterations and superior stability, indicating that the Hybrid Knowledge Layer redirects search toward higher-quality regions rather than merely accelerating it. A paired Wilcoxon signed-rank test confirms significance on both rounds ($p < 0.001$, **95% CI [-8.6, -5.9]**) and PSI ($p < 0.01$, **95% CI [-0.038, -0.014]**; n=18). OOT KS also showed consistent, albeit modest, improvement ($+0.005$, **95% CI [+0.001, +0.009]**). Directional consistency is strong, with ACRM achieving equal or lower PSI in 17 of 18 events.

The Generic Agent result is equally instructive: despite sharing the same backbone and tools, it required *more* iterations than the Manual Expert while failing PSI thresholds in 4 of 6 scenarios. This confirms that in regulated domains, unconstrained exploration is not merely inefficient but actively harmful—the agent exploits metric shortcuts that render models undeployable.

Offline validation. To address the limited online sample size, we backtest on 76 historical drift events with a strict temporal-split protocol (Appendix G). Results are consistent with online deployment (all $p < .001$), and the progressive ablation replicates at this larger scale.

4.3 Ablation Study

To isolate each reasoning stream's contribution, we conducted a progressive ablation across all six scenarios (Table 3). We report the average gap relative to the Manual Expert baseline ($\Delta\text{KS} = \text{KS}_{\text{config}} - \text{KS}_{\text{manual}}$; $\Delta\text{PSI} = \text{PSI}_{\text{config}} - \text{PSI}_{\text{manual}}$; negative ΔPSI is better), making each component's marginal contribution directly interpretable.

Configuration	Days	ΔKS	ΔPSI	ΔGap
Manual Expert (ref.)	14.3	—	—	—
HPO-Only	–	−.018	+.037	+.015
Rules Only	15.8	−.009	−.013	+.004
Rules + Retrieval	5.2	−.001	−.013	.000
Full ACRM	**1.1**	**+.005**	**−.026**	**−.009**

Table 3: Progressive ablation (six-scenario average). Δ is computed against Manual Expert (KS = .380, PSI = .10, Gap = .042). Positive ΔKS and negative ΔPSI/ΔGap indicate improvement over manual workflow. Per-scenario breakdown in Appendix A.

The ablation reveals a clear progression. HPO-Only, operating on a stale cohort, lags the Manual Expert by 0.018 in KS with substantially worse stability (ΔPSI = +.037), confirming that upstream data engineering is the primary control plane. Adding Rules closes the stability gap ($\Delta\text{PSI} = -.013$) but slightly *increases* cycle time (15.8 vs. 14.3 days): hard constraint enforcement forces the agent to discard and re-explore action paths that a human expert would avoid upfront through domain intuition, confirming that safety guardrails alone are insufficient without the inductive bias provided by retrieval. Retrieval nearly eliminates the KS deficit ($\Delta\text{KS} = -.001$) and cuts days sharply, yet stability sees no further gain. Only the full DPO-aligned system surpasses the Manual Expert on all metrics simultaneously while completing refreshes in 1.1 days, demonstrating that preference alignment captures trade-off reasoning that neither rules nor retrieval provide. An SFT-only variant degraded instruction-following after round 3, confirming DPO's suitability; details in Appendix A.

5 Deployment Experience

Production status. ACRM has been deployed on the institution's on-premise GPU cluster (8×A100) behind an air-gapped network since July 2025. A monitoring service triggers the system automatically when KS decay exceeds δ=0.03; all intermediate artifacts are persisted to the internal model registry for audit. All 18 candidate models produced during the three-month evaluation passed the institution's model risk compliance review and were delivered to the respective business teams for production integration. The system remains in active operation and has since been extended to additional business lines. From an operational perspective, the reduction from weeks to 1.1 days per refresh eliminates the prolonged exposure window during which a decayed model remains in production—a period that, under the previous workflow, could accumulate measurable increases in portfolio risk due to mis-calibrated score cutoffs. The accelerated cycle also enabled a shift from quarterly to monthly maintenance cadence, improving responsiveness to emerging drift.

Human oversight. The system operates under **bounded automation**: it produces a candidate model with a full reasoning trace, but no model enters production without human sign-off. Across 18 refresh events, ACRM converged autonomously in 14 cases (78%); the remaining 4 required targeted human input for unprecedented drift patterns or conflicting guardrails, resolved within 2–4 hours each (Appendix D). The 4 human-assisted cases are included in all reported averages.

Maintenance. The trajectory memory uses recency-weighted indexing to prevent retrieval degradation. The DPO policy has not required retraining during the evaluation period; quarterly retraining is planned as the trajectory corpus grows.

6 Lessons Learned in Production

Our three-month deployment and extensive offline backtesting yielded several key insights for building agentic systems in heavily regulated environments. Three findings merit highlighting:

(1) Stability must be a hard constraint. An unbounded agent will eventually game any soft numerical objective. We observed that generic LLMs excel at finding statistical shortcuts—such as dropping noisy but essential demographic segments to inflate the KS metric—which invariably renders the resulting model undeployable. The Rule Stream's non-negotiable PSI boundary is therefore essential; preference alignment (DPO) alone is insufficient to prevent reward hacking without these deterministic guardrails. Furthermore, a constraint-deadlock incident (where strict PSI limits contradicted KS recovery) motivated us to develop scenario-specific guardrail profiles rather than applying a single global threshold (Appendix D).

(2) The control plane is the data. Contrary

to traditional AutoML's focus on hyperparameter search, our progressive ablation (Table 3) confirmed that upstream data engineering (e.g., cohort definition, observation windowing) dominated performance recovery. This validates our design choice to equip the agent with SQL-level control over the data pipeline. However, a current architectural limitation is that the agent can only select or filter features, not conceptualize and create fundamentally novel ones. Our 76-event offline analysis quantified this ceiling: 76.3% (58/76) of refresh events were successfully resolved using the existing feature catalog, while 23.7% (18/76) required novel feature creation by humans. Notably, these 23.7% were also the most labor-intensive for human experts (averaging 18.7 vs. 12.9 days), pointing to automated feature synthesis as the critical next frontier.

(3) Bounded automation beats full autonomy. In high-stakes domains like banking, full autonomy is neither feasible nor desirable due to strict model risk governance. ACRM's true value lies in efficiently exploring the vast trajectory space and presenting a high-confidence candidate alongside a comprehensive, step-by-step reasoning trace (audit trail). This paradigm shifts the role of human data scientists from manual implementers to high-level auditors. By ensuring humans retain the final sign-off, *bounded automation* fosters crucial trust with compliance teams while still capturing the vast majority of operational speedups.

7 Conclusion

We presented ACRM, a deployed multi-agent framework that automates credit risk model refreshing by treating the modeling workflow as a learnable trajectory. Its three-layer decision stack—Rule Guardrails for safety, Retrieval for context, and DPO Alignment for expert judgment—enables a balance between efficiency and stability that neither AutoML nor generic LLM agents achieve. Deployed across six business scenarios for three months, ACRM reduced the refresh cycle from weeks to $\sim$1.1 days while maintaining superior stability. The system remains in active production use and is being extended to additional business lines, suggesting that effective industrial AI requires agents that optimize the entire engineering workflow within compliance boundaries rather than model parameters alone. Our findings underscore that in high-stakes domains, *bounded automation*—where the agent explores trajectories but humans retain final audit authority—is the safest path to deployment.

Ethical Considerations

Credit risk models directly affect individuals' access to financial services, making fairness and transparency paramount concerns in any automation effort.

Fairness and Bias. ACRM automates upstream data engineering decisions such as cohort selection and feature screening, which could inadvertently exclude or disadvantage protected demographic groups. While the current Rule Stream enforces statistical stability, we explicitly acknowledge it does not yet encode group-fairness objectives (e.g., demographic parity or equalized odds). Currently, fairness is strictly gated by the institution's downstream human-led fair-lending compliance review prior to deployment. We recognize that integrating fairness-aware guardrails directly into the agent's decision loop is a critical direction for future work.

Human Oversight and Accountability. Crucially, ACRM operates strictly as a developer-facing tool. It does not make consumer-level credit approve/deny decisions. Its sole function is to generate candidate model-refresh trajectories and candidate models for internal engineering review. ACRM operates under a bounded-automation paradigm: no model enters production without explicit human sign-off from a senior modeler and compliance officer. The system produces a full reasoning trace for every refresh event, enabling auditors to inspect and challenge each decision. Accountability for deployed models remains with the institution's model risk management team, not with the automated system.

Data Privacy. All components—including the Qwen-32B Orchestrator and the embedding model—are deployed on-premise within an air-gapped network. No customer data or proprietary model artifacts leave the institution's secure infrastructure at any point during training, inference, or trajectory storage.

References

M. S. Abdul Razak, C. R. Nirmala, B. R. Sreenivasa, Husam Lahza, and Hassan Fareed M. Lahza. 2022. A survey on detecting healthcare concept drift in

AI/ML models from a finance perspective. *Frontiers in Artificial Intelligence*, 5:955314. Published online: 17 April 2023.

Roland Abi. 2025. Machine learning for credit scoring and loan default prediction using behavioral and transactional financial data. *World Journal of Advanced Research and Reviews*, 26(3):884–904.

Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. 2020. AutoGluon-Tabular: Robust and accurate AutoML for structured data. *arXiv preprint arXiv:2003.06505*.

Siyuan Guo, Cheng Deng, Ying Wen, Hechang Chen, Yi Chang, and Jun Wang. 2024. DS-agent: Automated data science by empowering large language models with case-based reasoning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 16813–16848. PMLR.

Noah Hollmann, Samuel Müller, and Frank Hutter. 2023. CAAFE: Context-aware automated feature engineering with large language models. *arXiv preprint arXiv:2305.03403*. Submitted on 5 May 2023 (v1); last revised 28 Sep 2023 (v5).

Sirui Hong, Yizhang Lin, Bangbang Liu, Binhao Wu, Danyang Li, Jiaqi Chen, Jiayi Zhang, Jinlin Wang, Lingyao Zhang, Mingchen Zhuge, Taicheng Guo, Tuo Zhou, Wei Tao, Wenyi Wang, Xiangru Tang, Xiangtao Lu, Xinbing Liang, Yaying Fei, Yuheng Cheng, and 5 others. 2024. Data Interpreter: An LLM agent for data science. *arXiv preprint arXiv:2402.18679*. Submitted on 28 Feb 2024 (v1); latest version 15 Oct 2024 (v4).

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*. V2; includes improved baselines, GLUE experiments, and adapter latency analysis.

Junjie Jiang, Haodong Wu, Yongqi Zhang, Songyue Guo, Bingcen Liu, Caleb Chen Cao, Ruizhe Shao, Chao Guan, Peng Xu, and Lei Chen. 2025. ArchiDocGen: Multi-agent framework for expository document generation in the architectural industry. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 605–618, Vienna, Austria. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. *arXiv preprint arXiv:2005.11401*. Accepted at NeurIPS 2020.

Alexander Pan, Kush Bhatia, and Jacob Steinhardt. 2022. The effects of reward misspecification: Mapping and mitigating misaligned models. *arXiv preprint arXiv:2201.03544*. ICLR 2022; submitted on 10 Jan 2022 (v1); last revised 14 Feb 2022 (v2).

Shuofei Qiao, Runnan Fang, Zhisong Qiu, Xiaobin Wang, Ningyu Zhang, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. 2024. Benchmarking agentic workflow generation. *arXiv preprint arXiv:2410.07869*. Submitted on 10 Oct 2024 (v1), last revised 23 Feb 2025 (v3). Accepted to ICLR 2025.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*. NeurIPS 2023 camera-ready version.

Prashanth B S, Manoj Kumar M V, Puneetha B H, and Ajay Kumara M A. 2025. A data-driven framework for detecting and mitigating concept drift in adaptive artificial neural networks. *Procedia Computer Science*, 258:1147–1158. Open access under a Creative Commons license.

Alok Kumar Sahu, Yi Sun, Eamonn Swanton, Farshid Amirabdollahian, and Abi Wren. 2025. OccuTriage: An AI agent orchestration framework for occupational health triage prediction. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 1217–1226, Vienna, Austria. Association for Computational Linguistics.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*.

Binpeng Shi, Yu Luo, Jingya Wang, Yongxin Zhao, Shenglin Zhang, Bowen Hao, Chenyu Zhao, Yongqian Sun, Zhi Zhang, Ronghua Sun, Haihua Li, Wei Song, Xiaolong Chen, Jingbo Miao, and Dan Pei. 2025. Flowxpert: Expertizing troubleshooting workflow orchestration with knowledge base and multi-agent coevolution. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '25)*, page 12, Toronto, ON, Canada. ACM. To appear.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*. V4; includes additional experiments.

Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. 2022. Defining and characterizing reward hacking. *arXiv preprint arXiv:2209.13085*. Submitted on 27 Sep 2022 (v1); last revised 5 Mar 2025 (v2).

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W. White, Doug Burger, and Chi Wang. 2023. AutoGen: Enabling next-gen LLM applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*. V2; 43 pages (10 pages main text, 30 pages appendices).

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Daochen Zha, Zaid Pervaiz Bhat, Kwei-Herng Lai, Fan Yang, Zhimeng Jiang, Shaochen Zhong, and Xia Hu. 2025. Data-centric artificial intelligence: A survey. *ACM Computing Surveys*, 57(5):1–42.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. 2024. AFlow: Automating agentic workflow generation. *arXiv preprint arXiv:2410.10762*. Accepted to ICLR 2025.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. Qwen3 Embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

He Zhu, Guanhua Chen, and Wenjia Zhang. 2025. PlanGPT: Enhancing urban planning with a tailored agent framework. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 764–783, Vienna, Austria. Association for Computational Linguistics.

A Detailed Ablation Analysis

A.1 Per-Scenario Breakdown

Table 4 reports the ablation results disaggregated by scenario type. The relative contribution of each reasoning stream varies with the nature of the drift.

A.2 Qualitative Analysis by Stage

HPO-Only: The Stale Cohort Ceiling. The HPO-Only baseline achieves reasonable KS in low-drift scenarios (Credit Card B/E) but fails in high-drift settings. In Cash Loan A, for example, it recovers only 0.355 against the target of 0.385+. The root cause is that the training data still uses a 180-day observation window that includes pre-drift behavioral patterns. No amount of hyperparameter tuning can overcome a fundamentally misaligned training distribution. Notably, PSI values under HPO-Only consistently exceed 0.10, meaning these models would be *rejected at deployment review* regardless of their KS.

Rules Only: Safe but Inefficient. Adding the Rule Stream immediately resolves the PSI problem by enforcing hard boundaries (e.g., dropping features with PSI > 0.10, capping observation windows). However, without retrieval or alignment, the agent has no prior knowledge of *which* data engineering actions are likely to work. It resorts to exhaustive exploration, averaging 14.2 rounds. In SME Loan C, the agent spent 18 rounds testing different cohort definitions before finding one that satisfied all constraints—a trajectory that a senior modeler would have shortened significantly based on past experience.

Rules + Retrieval: Fast but Rigid. Retrieval provides the missing inductive bias. By matching current drift patterns to historical successes, the agent skips unproductive exploration paths and converges in 8.1 rounds on average (43% reduction). The limitation emerges in scenarios where historical precedent is weak. In SME Loan C, the retrieved trajectories came from a different market regime, and the agent blindly followed a window configuration that was suboptimal for the current drift. This is visible in the numbers: SME Loan C shows the smallest KS gain from adding retrieval (+0.007) compared to other scenarios (+0.010 to +0.012).

Full ACRM: Aligned Trade-off Reasoning. The DPO-aligned policy addresses exactly this rigidity. Instead of copying retrieved strategies, the aligned agent uses them as starting points and *adapts* based on learned preferences. The most telling evidence is the Gap metric: Full ACRM achieves the lowest train-validation divergence across all scenarios, including those where KS is not the highest achievable. In Cash Loan A, for instance, the agent could have pushed KS to 0.396 (observed in one intermediate round) but chose a configuration at 0.391 that offered substantially better stability. This voluntary trade-off is the signature of successful preference alignment.

Scenario	Drift Type	Rounds (#)				OOT KS			
		HPO	+R	+R+Ret	Full	HPO	+R	+R+Ret	Full
Cash Loan A	Behavioral	–	16	9	4	0.355	0.368	0.380	0.391
Cash Loan D	Behavioral	–	15	8	4	0.348	0.360	0.372	0.383
Credit Card B	Mild/Stable	–	10	6	3	0.395	0.402	0.406	0.408
Credit Card E	Mild/Stable	–	11	7	4	0.388	0.394	0.399	0.402
SME Loan C	Structural	–	18	12	5	0.330	0.348	0.355	0.355
SME Loan F	Structural	–	15	7	4	0.340	0.355	0.362	0.370
Average		–	14.2	8.1	**4.0**	0.362	0.371	0.379	**0.385**

Table 4: Per-scenario ablation. +R = Rules Only; +R+Ret = Rules + Retrieval; Full = Full ACRM with DPO alignment. HPO-Only does not involve iterative rounds as it operates on a fixed dataset.

Scenario	Drift Type	PSI				Gap (Tr-Val)			
		HPO	+R	+R+Ret	Full	HPO	+R	+R+Ret	Full
Cash Loan A	Behavioral	0.16	0.10	0.10	0.09	0.062	0.050	0.046	0.038
Cash Loan D	Behavioral	0.15	0.10	0.10	0.08	0.058	0.048	0.045	0.036
Credit Card B	Mild/Stable	0.07	0.06	0.06	0.05	0.035	0.028	0.025	0.021
Credit Card E	Mild/Stable	0.08	0.06	0.06	0.05	0.038	0.030	0.028	0.024
SME Loan C	Structural	0.20	0.12	0.11	0.11	0.078	0.062	0.055	0.040
SME Loan F	Structural	0.18	0.10	0.10	0.08	0.070	0.055	0.050	0.038
Average		0.14	0.09	0.09	**0.08**	0.057	0.046	0.042	**0.033**

Table 5: Per-scenario ablation (stability metrics). Rules enforce the PSI < 0.10 boundary but cannot further optimize it. DPO alignment is the only component that consistently reduces both PSI and Gap.

B Full Experimental Results

Table 6 reports the complete six-scenario comparison, including AUC.

C Execution Trace Example

To illustrate how ACRM operates in practice, we present a condensed trace from a Cash Loan scenario triggered by a KS drop of 0.041 (from 0.412 to 0.371).

Round 1: Diagnosis & Data Adjustment. The agent receives the drift alert and queries the Retrieval Stream, which returns the top-3 historical trajectories. Among these, two share similar drift magnitude (KS drop $\in [0.03, 0.05]$) and both resolved the issue by shortening the observation window; the Orchestrator grounds its plan primarily on these two precedents.The Rule Stream confirms that the current 180-day window exceeds the recommended maximum for high-drift segments. The agent issues:

```
Action: DataSelect(
  obs_window=90,
  perf_window=12,
  cohort_filter="remove segment:
    first_loan_within_30d"
)
```

Outcome: New training set constructed. Sample size reduced by 8%. Preliminary KS estimate: 0.382.

Round 2: Feature Refinement. The evaluation report flags two features with PSI > 0.15. The DPO-aligned policy, rather than simply dropping them, proposes replacing them with more stable alternatives from a related feature family:

```
Action: FeatureFilter(
  drop=["txn_amt_30d_std", "query_cnt_7d"],
  add=["txn_amt_90d_avg", "query_cnt_30d"],
  psi_threshold=0.10
)
```

Outcome: PSI reduced from 0.14 to 0.08. KS remains at 0.380.

Round 3: Model Training & Convergence. With a cleaned dataset and stable feature set, the agent issues a training directive with a conservative regularization strategy:

```
Action: ModelTrain(
  strategy="high_regularization",
  intent="prioritize generalization,
    accept minor KS trade-off"
)
```

Outcome: Final model achieves KS=0.391 (OOT), AUC=0.745, PSI=0.09, Gap=0.038. All guardrails passed. The agent issues `TERMINATE` and submits the candidate model for human review.

Summary. The full cycle completed in 3 rounds over 4.2 hours of wall-clock time. The key deci-

Table 6: Full comparison across all six scenarios. **Generic Agent**: same Qwen-32B with Specialist Agents but without Rule, Retrieval, or Alignment streams. †: PSI > 0.10, failing deployment review.

Scenario	Method	Days	Rounds	KS	AUC	Gap	PSI
Cash Loan A	Manual Expert	14.0	12	0.385	0.742	0.045	0.12
	Generic Agent	3.6	15	0.376	0.733	0.052	0.16†
	ACRM (Ours)	1.0	4	0.391	0.745	0.038	0.09
Cash Loan D	Manual Expert	12.0	10	0.378	0.735	0.048	0.11
	Generic Agent	2.4	11	0.380	0.736	0.055	0.18†
	ACRM (Ours)	1.0	4	0.383	0.740	0.036	0.08
Credit Card B	Manual Expert	10.0	8	0.410	0.780	0.020	0.05
	Generic Agent	2.0	12	0.402	0.768	0.030	0.09
	ACRM (Ours)	0.8	3	0.408	0.779	0.021	0.05
Credit Card E	Manual Expert	11.0	9	0.400	0.770	0.025	0.06
	Generic Agent	2.2	11	0.389	0.759	0.032	0.10
	ACRM (Ours)	0.9	4	0.402	0.772	0.024	0.05
SME Loan C	Manual Expert	21.0	15	0.350	0.710	0.060	0.15
	Generic Agent	4.5	17	0.336	0.701	0.068	0.22†
	ACRM (Ours)	1.5	5	0.355	0.715	0.040	0.11
SME Loan F	Manual Expert	18.0	14	0.358	0.718	0.055	0.13
	Generic Agent	4.0	16	0.331	0.709	0.060	0.19†
	ACRM (Ours)	1.2	4	0.370	0.725	0.038	0.08
Average	Manual Expert	14.3	11.3	0.380	0.743	0.042	0.103
	Generic Agent	3.1	13.7	0.369	0.734	0.050	0.157†
	ACRM (Ours)	**1.1**	**4.0**	**0.385**	**0.746**	**0.033**	**0.077**

sion —shortening the observation window—was surfaced by retrieval in Round 1, while the feature replacement strategy in Round 2 reflects the DPO-aligned policy's preference for stability over raw performance. A manual expert later confirmed that they would have followed a similar strategy, but estimated it would have taken 3–4 working days.

D Failure Cases and Recovery

While ACRM succeeded in producing deployable models in the majority of refresh events during the three-month evaluation, we observed two recurring failure patterns that required human intervention.

Case 1: Unprecedented Drift Pattern (SME Loan). In one SME Lending scenario, a sudden regulatory change in small-business loan eligibility criteria introduced a distributional shift unlike anything in the trajectory memory. The Retrieval Stream returned low-similarity matches (cosine similarity < 0.6), and the DPO-aligned policy lacked sufficient training signal for this regime. The agent entered a "exploration loop," cycling through 8 rounds of window and feature adjustments without converging to a solution that satisfied both KS and PSI constraints simultaneously.

Recovery: The system triggered a human handover at round $T_{\max} = 10$ (Algorithm 1, line 20). A senior modeler identified that the core issue was a missing feature family related to the new regulatory variable, which was outside the agent's predefined feature catalog. After the modeler manually added three features to the candidate pool, ACRM was re-triggered and converged in 2 additional rounds.

Lesson: This case exposed a fundamental limitation of the current architecture: the agent can *select* and *filter* features, but cannot *create* novel ones. Extending the tool ecosystem with a feature-generation module is a priority for future work.

Case 2: Conflicting Guardrails (Credit Card). In a Credit Card scenario with an aging customer base, the agent found itself in a constraint deadlock. Tightening the PSI threshold (as recommended by the Rule Stream) forced the removal of demographic features that were critical for maintaining KS. Conversely, relaxing the PSI constraint to retain these features violated the hard operational boundary.

Recovery: The system surfaced the conflict in its diagnostic report, presenting the Pareto frontier of achievable (KS, PSI) pairs to the human auditor. The auditor, after consulting with the business team, approved a temporary PSI threshold relaxation from 0.10 to 0.12 for this specific scenario, after which the agent converged in one additional

round.

Lesson: Hard constraints are essential for safety, but they must be *scenario-aware*. We subsequently extended the Rule Stream to support configurable guardrail profiles per business scenario, rather than enforcing a single global threshold.

Overall Statistics. Across the 18 refresh events during the evaluation period, ACRM achieved autonomous convergence (no human intervention beyond final audit) in 14 cases (78%). Of the 4 cases requiring intervention, 2 involved the unprecedented drift pattern described above, and 2 involved guardrail conflicts. In all cases, the system's diagnostic trace enabled rapid human resolution, typically within 2–4 additional hours. No model produced by ACRM was deployed without passing the standard human review process.

E DPO Training Details

This appendix consolidates the preference data construction, training configuration, and robustness analysis for the Alignment Stream introduced in Section 3.3.

E.1 Trajectory Collection and Labeling

Over the 12 months preceding deployment, we instrumented the bank's credit modeling workflow to capture structured trajectories $\tau_i = \{(s_t, a_t, o_t)\}_{t=1}^{T}$, recording the full chain of tool calls, diagnostic outputs, and intermediate model checkpoints at each step. Each completed modeling session was labeled by the responsible senior modeler as either *accepted* (the resulting model passed compliance review and was deployed to production) or *rejected* (the model failed review due to stability violations, insufficient discriminative power, or business coverage concerns). Labeling was performed as part of the existing model governance process, not as a separate annotation effort, ensuring that labels reflect genuine deployment decisions rather than post-hoc judgments.

Table 7 summarizes the resulting corpus.

E.2 Preference Pair Construction

The key challenge in constructing preference pairs is ensuring that the two trajectories in each pair address a *comparable* problem, so that their outcome difference can be attributed to decision quality rather than task difficulty. We proceed in three steps.

Table 7: DPO training corpus statistics and offline evaluation.

Statistic	Value
Total trajectories	1,247
Accepted / Rejected	834 / 413
Avg. length (rounds)	8.3 ± 3.1
Avg. tokens per trajectory	$4{,}720 \pm 1{,}830$
Preference pairs constructed	3,812
Train / Held-out	3,012 / 800
Initial-state similarity threshold	$\cos > 0.75$
Pairwise accuracy (pre-DPO)	71.4%
Pairwise accuracy (post-DPO)	91.8%
Kendall's τ vs. expert ranking	0.79

Step 1: Drift-State Matching. We encode each trajectory's initial drift state s_0 (comprising KS decay magnitude, PSI heatmap across score bins, affected feature categories, and scenario type) using the Qwen-Embedding-8B model described in Section 3.1. We then compute pairwise cosine similarities and retain only pairs with $\mathrm{sim}(s_0^{(i)}, s_0^{(j)}) > 0.75$. This threshold was chosen empirically: below 0.70, matched pairs frequently involve qualitatively different drift types (e.g., behavioral vs. structural), introducing noise; above 0.80, too few pairs survive, particularly for rare SME scenarios.

Step 2: Outcome-Based Filtering. From matched pairs, we distinguish three cases:

- **Accepted vs. Rejected**: The accepted trajectory is directly preferred. These constitute $\sim$55% of all pairs.
- **Accepted vs. Accepted**: Both trajectories led to deployed models, but one is ranked higher via the composite score (Step 3). These constitute $\sim$35% of pairs and are critical for teaching fine-grained trade-off reasoning.
- **Rejected vs. Rejected**: Discarded, as neither trajectory represents a desirable outcome.

Step 3: Composite Scoring for Accepted–Accepted Pairs. When both trajectories in a pair were accepted, we rank them using:

$$R(\tau) = \alpha \cdot R_{\text{perf}} - \beta \cdot C_{\text{stab}} - \gamma \cdot C_{\text{gap}} \quad (2)$$

where:

- R_{perf}: KS recovery on out-of-time (OOT) data, defined as $\text{KS}_{\text{final}} - \text{KS}_{\text{trigger}}$ (higher is better);
- C_{stab}: PSI of the final model's score distribution relative to the production population (lower is better);

- C_{gap}: absolute difference between training KS and OOT KS, measuring generalization (lower is better).

The weights $(\alpha, \beta, \gamma) = (1.0, 2.0, 1.5)$ were set by the bank's model risk management team to encode regulatory priorities: stability is weighted twice as heavily as raw performance, and generalization receives intermediate priority. This score is used *exclusively* for offline pair construction; at inference time, preferences are expressed through the model's internalized policy, not as an external reward.

E.3 Robustness of Preference Weights

To verify that the induced ranking is not brittle to the specific weight choice, we varied β/α over $\{1.0, 1.5, 2.0, 3.0\}$ while keeping $\gamma/\alpha = 1.5$ fixed. For each weight configuration, we re-constructed the preference pairs and evaluated the resulting DPO model on the same held-out set of 800 pairs.

Table 8: Sensitivity of DPO pairwise accuracy to composite score weights. The ranking is stable for $\beta/\alpha \geq 1.5$.

β/α	Pair Overlap	Pairwise Acc. (%)	Kendall's τ
1.0	78.3%	84.6	0.68
1.5	91.2%	90.3	0.76
2.0	—	91.8	0.79
3.0	93.7%	91.1	0.78

The "Pair Overlap" column reports what fraction of the default ($\beta/\alpha = 2.0$) preference pairs retain the same ordering under the alternative weights. For $\beta/\alpha \geq 1.5$, overlap exceeds 91% and pairwise accuracy remains within 1.5 points of the default. At $\beta/\alpha = 1.0$—where stability and performance are equally weighted—accuracy drops to 84.6%, and manual inspection reveals that the model begins to prefer trajectories that accept higher PSI for marginal KS gains, precisely the behavior we aim to suppress. This confirms that the ranking is stable as long as stability is given meaningful priority over raw KS, a condition consistent across all banking partners we consulted.

E.4 Input Representation for DPO Training

Each trajectory is serialized into a structured text sequence that the Qwen-32B model consumes during DPO training. The format is designed to mirror the information the Orchestrator sees at inference time, ensuring distribution consistency between training and deployment.

A single trajectory is formatted as follows:

```
[SCENARIO] Cash Loan A
[DRIFT_TRIGGER] KS dropped from 0.412 to 0.371
  (delta=-0.041). PSI=0.14. Date: 2024-09-15.

[ROUND 1]
<STATE> KS=0.371,PSI=0.14,Top unstable features:
  txn_amt_30d_std (PSI=0.18),
  query_cnt_7d (PSI=0.15).
  Observation window: 180d. Cohort size: 52,340.
<ACTION> DataSelect(obs_window=90,perf_window=12,
  cohort_filter="remove segment:first_loan_30d")
<OUTCOME> New cohort size: 48,113 (-8.1%).
  Preliminary KS=0.382, PSI=0.12.

[ROUND 2]
<STATE> KS=0.382,PSI=0.12,
  Top unstable features:
      txn_amt_30d_std (PSI=0.15),
      query_cnt_7d (PSI=0.13).
<ACTION> FeatureFilter(drop=[txn_amt_30d_std,
  query_cnt_7d], add=[txn_amt_90d_avg,
  query_cnt_30d], psi_threshold=0.10)
<OUTCOME> Feature count: 127->119.
  KS=0.380, PSI=0.08.

[ROUND 3]
<STATE> KS=0.380, PSI=0.08, Gap=0.042.
  All features PSI<0.10.
<ACTION>
  ModelTrain(strategy="high_regularization",
    intent="prioritize generalization")
<OUTCOME> KS=0.391, AUC=0.745, PSI=0.09,
  Gap=0.038. All guardrails passed.

[RESULT] ACCEPTED. Deployed 2024-09-16.
```

For DPO training, each preference instance consists of a shared *prompt* (the drift trigger context, i.e., the `[SCENARIO]` and `[DRIFT_TRIGGER]` blocks) and two *completions* (the full round-by-round trajectory of the preferred and dispreferred trajectories respectively). The model learns to assign higher log-probability to the preferred completion.

Sequence Length Statistics. The average trajectory serializes to $4{,}720 \pm 1{,}830$ tokens (measured with the Qwen-32B tokenizer). The longest trajectory in the corpus spans 15 rounds and 11,240 tokens. During DPO training, we set a maximum sequence length of 8,192 tokens (prompt + completion); trajectories exceeding this limit ($< 3\%$ of the corpus) are truncated by removing intermediate rounds while preserving the first round (initial diagnosis), the last two rounds (convergence behavior), and the final outcome. This truncation strategy was chosen to retain the most decision-relevant information: early rounds establish the strategy direction, and late rounds capture the fine-tuning trade-offs that distinguish preferred from dispreferred trajectories.

E.5 Training Configuration

Table 9: DPO training hyperparameters.

Parameter	Value
Base model	Qwen-32B (open-weight)
Adaptation	LoRA (rank=64, α=128)
Target modules	All attention layers (Q, K, V, O)
DPO β (KL penalty)	0.1
Learning rate	5×10^{-6}
Scheduler	Cosine with 5% warmup
Batch size	4 (per GPU) $\times$ 8 GPUs
Gradient accumulation	4 steps
Effective batch size	128
Epochs	3
Max sequence length	8,192 tokens
Precision	bfloat16
Hardware	8 $\times$ NVIDIA A100 80GB
Training time	$\sim$6 hours
Framework	DeepSpeed ZeRO Stage 2

DPO was chosen over PPO for two practical reasons. First, with $\sim$3,800 preference pairs, the data volume is modest by RLHF standards; PPO's on-policy sampling would require significantly more compute to achieve stable convergence, which is impractical on an air-gapped cluster without elastic scaling. Second, DPO avoids the need for a separately trained reward model, reducing the engineering surface area—a meaningful advantage when the entire stack must be maintained by a small team within a banking IT organization.

We monitored training via held-out pairwise accuracy evaluated at each epoch boundary. Accuracy plateaued after epoch 2 (90.9% $\rightarrow$ 91.8% $\rightarrow$ 91.7%), and we selected the epoch-3 checkpoint based on marginally better Kendall's τ (0.79 vs. 0.78 at epoch 2). No overfitting was observed, likely because LoRA constrains the effective parameter count to $\sim$67M out of 32B total.

E.6 Inference-Time Usage

At inference time, the DPO-aligned model serves as the Orchestrator itself—it is not used as a separate scoring function. Concretely, at each decision point t, the Orchestrator receives a prompt composed of:

1. **System instructions**: Hard constraints from the Rule Stream, formatted as enumerated prohibitions (e.g., "You MUST NOT select any feature with PSI > 0.10.").
2. **Few-shot context**: The top-k ($k = 3$) retrieved historical trajectories from the Retrieval Stream, formatted identically to the training representation above.
3. **Current state**: The accumulated trajectory so far $\{(s_1, a_1, o_1), \ldots, (s_{t-1}, a_{t-1}, o_{t-1})\}$ plus the current diagnostic state s_t.

Setting	n	Rnds	ΔKS	ΔPSI
Offline (all)	76	4.3	$+.005$	$-.023$
Online	18	4.0	$+.005$	$-.026$

Table 10: Online results fall within offline estimates, confirming that the 18 deployment events are representative.

The model generates a single next-step action plan (not a complete trajectory), which is then validated by the Rule Stream before execution. This step-by-step generation—as opposed to planning the full trajectory upfront—allows the agent to incorporate the actual outcome of each action into subsequent decisions, maintaining the closed-loop property that distinguishes ACRM from open-loop planning approaches.

F Baseline Details

Generic Agent: soft-prompt variant. Beyond the unconstrained Generic Agent reported in the main results, we evaluated a soft-prompt variant that encodes domain constraints as natural-language instructions (e.g., *"target OOT KS $\geq$ 0.30; keep PSI below 0.10"*). This variant showed marginal improvement in early testing (3 scenarios): KS remained 1.5–2 pp below Manual Expert, and without the Rule Stream's deterministic enforcement, the agent could not reliably diagnose the *cause* of PSI violations, resorting to arbitrary feature removal from the second iteration onward.

On open-source agent frameworks. General-purpose data-science agents such as Data Interpreter (Hong et al., 2024) and DS-Agent (Guo et al., 2024) are architecturally incompatible with our setting for two reasons. First, they require external reward signals and workflow structures that do not accommodate hard regulatory constraints (e.g., non-negotiable PSI boundaries, audit-trail requirements); adapting them would necessitate designing bespoke reward functions and re-engineering their core planning loops, effectively building a new system. Second, our air-gapped banking infrastructure prohibits the external API calls these frameworks depend on. A fair comparison would therefore require substantial re-implementation on both sides, making results difficult to interpret.

Config.	Rnds↓	ΔKS	ΔPSI	ΔGap
Manual (ref.)	10.9	—	—	—
HPO-Only	—	$-.020$	$+.034$	$+.016$
Rules Only	15.1	$-.010$	$-.011$	$+.005$
Rules + Ret.	7.8	$-.002$	$-.012$	$+.001$
Full ACRM	4.3	$+.005$	$-.023$	$-.008$

Table 11: Progressive ablation ($n = 76$ offline events).

Method	Days↓	Rnds↓	ΔKS↑	ΔPSI↓
Manual Expert	13.8	10.9	.042	.102
ACRM	1.2	4.3	.047	.079
Δ	-12.6	-6.6	$+.005$	$-.023$
Wilcoxon p	$< .001$	$< .001$	.008	$< .001$

Table 12: Offline backtesting ($n = 76$). ΔKS = KS recovery from trigger. Effect sizes: Days r=.82, Rounds r=.79, ΔKS r=.31, ΔPSI r=.62.

G Offline Backtesting at Scale

To validate robustness beyond the 18 online events, we backtest ACRM on 76 historical drift events from the 12 months preceding deployment. Human-expert ground truth (KS, PSI, Gap, rounds, days) comes directly from the bank's model governance records.

Protocol. For each event triggered at time t, ACRM accesses *exactly the same data snapshot* as the human team. To prevent leakage, the DPO policy is trained only on trajectories completed before t via a rolling split (Q2: 287 train trajectories, 22 test events; Q3: 614/31; Q4: 943/23; total: 76 test events). Q1 serves as warm-up only.

Aggregate Results. All improvements are statistically significant. ACRM achieved equal or lower PSI in 68/76 events (89.5%), consistent with the 17/18 online rate.

Scaling with Corpus Size. The rolling protocol reveals monotonic improvement as DPO training data grows: rounds decrease from 5.1 (287 traj.) to 3.8 (943 traj.), with diminishing returns beyond ∼600 trajectories.

Ablation at Scale. The progressive ablation on 76 events reproduces the online pattern (Table 11; cf. Table 3): HPO-Only fails on stability (ΔPSI= $+.034$); Rules restore compliance but slow exploration (15.1 rounds); Retrieval cuts rounds to 7.8; only Full ACRM surpasses Manual Expert on all metrics ($p < .001$, all adjacent comparisons).