International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 08 | Aug 2026 www.irjet.net p-ISSN: 2395-0072

SIM-OPS: An Autonomous Multi-Agent Framework for Intelligent Business Operations Management

Arshvir Singh Kalsi[1], Viraj Prabhu[2], Ghrani Poojari[3], Hasti Shah[4], Varsha Nagpurkar[5]

[1,2,3,4] Department of Computer Engineering, St. Francis Institute of Technology, Mumbai, India
[5]Professor, Department of Computer Engineering, St. Francis Institute of Technology, Mumbai, India

---

Abstract-*This paper presents SIM-OPS (Smart Intelligence Management for Operations Planning & Supervision), an autonomous multi-agent system designed to function as an intelligent business operations manager. The system leverages advanced machine learning models (XGBoost, Isolation Forests), real-time data processing, and coordinated agent orchestration powered by Google Gemini via the LangChain framework to continuously monitor organizational metrics, predict business risks, and execute automated interventions. The framework integrates six specialized agents – Monitoring, Prediction, Decision, Action, Reporting and Feedback – that communicate through a database-backed message bus to perform end-to-end business intelligence operations. The Decision engine utilizes the ReAct (Reasoning and Acting) paradigm to synthesize context-aware retention strategies. The system achieves 86.94% accuracy in churn prediction with 0.89 average confidence, processes predictions autonomously every 6 hours, and executes multi-channel alerts through Slack, Jira and email. Experimental results demonstrate that the autonomous agent chain reduces manual intervention by 90% while achieving a 96% autonomous decision rate and a 0.3 second average response time for critical alerts. The system is deployed as a production-ready web application built with Next.js 16, TypeScript, Supabase and LangChain, with a Python-based FastAPI machine learning service.*

Key Words: Multi-agent systems, autonomous operations, machine learning, business intelligence, churn prediction, workflow automation, real-time monitoring

1. INTRODUCTION

Modern businesses generate vast amounts of operational data from customer interactions, financial transactions, support systems and product usage. Extracting actionable insights from this data and executing timely interventions remains a significant challenge. Traditional business intelligence systems require constant human oversight, leading to delayed responses to critical events such as customer churn, revenue anomalies and operational inefficiencies. The need for autonomous, intelligent systems that can continuously observe, analyze and act on business data has become increasingly critical. Such systems must not only detect patterns and predict outcomes but also coordinate multiple specialized components to execute complex workflows without human intervention.

1.1 Motivation

Delayed responses to churn signals, revenue anomalies and operational inefficiencies directly translate into lost revenue and wasted analyst hours. SIM-OPS is motivated by the need for a system that can observe business metrics around the clock, reason about which events matter, and execute the appropriate action without waiting on a human operator to be available.

1.2 Contributions

This paper makes the following contributions: (1) a novel six-agent framework in which specialized agents collaborate through a message bus to perform comprehensive business operations management; (2) a fully automated prediction pipeline that runs machine learning inference every 6 hours and triggers agent chains based on risk thresholds; (3) an intelligent, ReAct-based decision engine powered by Google Gemini that evaluates predictions, applies business logic and determines appropriate actions with 91% average confidence; (4) a multi-channel action executor that delivers alerts through Slack, creates Jira tickets and sends emails based on severity classification; and (5) a production-ready implementation deployed with modern web technologies, achieving sub-second response times and 24/7 autonomous operation.

2. LITERATURE REVIEW

2.1 Multi-Agent Systems

Multi-agent systems (MAS) have been extensively studied in artificial intelligence research [1], [14]. Traditional MAS frameworks, as surveyed by Ferber [15] and Stone and Veloso [16], focus on agent communication protocols, coordination mechanisms and distributed problem-solving. Multi-agent reinforcement learning [17] has advanced cooperative and competitive agent behaviors; however, most existing systems operate in simulated environments or require significant human oversight. Recent work has explored reinforcement-learning approaches [2] and hierarchical agent

architectures [3]. SIM-OPS differs by focusing on practical business operations with real-world constraints, emphasizing reliability, explainability and integration with existing enterprise systems.

2.2 LLM-Based Autonomous Agents

The emergence of Large Language Models (LLMs) has catalyzed a new generation of autonomous agent architectures. Park et al. [18] demonstrated that LLM-powered generative agents can exhibit believable, emergent social behaviors through memory and planning modules. Xi et al. [19] provide a comprehensive survey of LLM-based agents, categorizing them by perception, reasoning and action capabilities. SIM-OPS builds on these paradigms by deploying LLM-based agents in a production business-operations context, where reliability, auditability and integration with external enterprise systems (Slack, Jira, email) are primary constraints rather than open-ended generative behavior.

2.3 Business Intelligence and Predictive Analytics

Predictive analytics in business intelligence has traditionally relied on batch processing and periodic reporting [4]. Modern approaches incorporate real-time streaming analytics [5] and automated machine learning (AutoML) [6]. Data-mining techniques have been widely applied to customer relationship management [20]. Customer churn prediction has been addressed using logistic regression [7], random forests [8], [21] and gradient boosting [9]; comparative studies [22] show that ensemble methods consistently outperform single classifiers in churn-prediction tasks. SIM-OPS extends these approaches by integrating predictions into an autonomous action-execution framework.

2.4 Workflow Automation

Workflow automation systems such as Apache Airflow [10] and Temporal [11] provide orchestration capabilities but require manual workflow definition and lack intelligent decision-making. Business process management (BPM) systems [12] offer structured workflow execution but do not incorporate machine learning or autonomous agent coordination. SIM-OPS combines workflow automation with intelligent agents that dynamically determine appropriate actions based on real-time predictions and business rules.

3. THEORETICAL BACKGROUND

To provide a robust foundation for SIM-OPS, principles from predictive machine learning, anomaly detection and modern natural language processing are synthesized below.

3.1 Predictive Modeling with XGBoost

The core churn-prediction model utilizes XGBoost (eXtreme Gradient Boosting), an optimized distributed gradient-boosting library. XGBoost minimizes a regularized objective function:

$$L(t) = \Sigma_i\, l(y_i, \hat{y}_i(t-1) + f_t(x_i)) + \Omega(f_t) \quad \ldots(1)$$

where l is a differentiable convex loss function measuring the difference between the prediction $\hat{y}_i$ and the target y_i. The term Ω penalizes the complexity of the model:

$$\Omega(f) = \gamma T + \tfrac{1}{2}\lambda \|w\|^2 \quad \ldots(2)$$

with T the number of leaves in the tree and w the leaf weights. This regularization helps prevent overfitting, a common issue in customer behavioral datasets with high dimensionality.

3.2 Anomaly Detection via Isolation Forests

For monitoring KPIs such as revenue drops or sudden spikes in support tickets, Isolation Forests [23] are employed. Unlike traditional methods that profile normal points, Isolation Forests explicitly isolate anomalies. The path length $h(x)$ to isolate a point x is the number of edges from the root node to a terminating node, and the anomaly score is defined as:

$$s(x, n) = 2^{-E(h(x)) / c(n)} \quad \ldots(3)$$

where $E(h(x))$ is the average path length and $c(n)$ is the average path length of unsuccessful searches in a Binary Search Tree. Anomalies are identified when $s(x, n)$ approaches 1, indicating that the point is easily isolated and thus abnormal.

3.3 LLM-Based Agentic Reasoning

The multi-agent orchestration relies on Large Language Models employing the ReAct (Reasoning and Acting) framework [13]. ReAct prompts the LLM to generate both reasoning traces and task-specific actions in an interleaved manner. Given a context c, the agent generates a sequence of thoughts t_i, actions a_i and observations o_i:

$$\tau_i = \text{LLM}(c, t_1, a_1, o_1, \ldots, t_{i-1}, a_{i-1}, o_{i-1}) \quad \ldots(4)$$

This paradigm allows the Decision Agent to dynamically evaluate the context of a high-risk customer and synthesize a tailored retention strategy rather than relying solely on static rule-based systems.

4. SYSTEM ARCHITECTURE

4.1 Overview

SIM-OPS employs a layered architecture consisting of four primary components: (1) Data Ingestion Layer, (2) Machine Learning Service, (3) Multi-Agent Orchestration Layer, and (4) Action Execution Layer, shown in Fig -1.

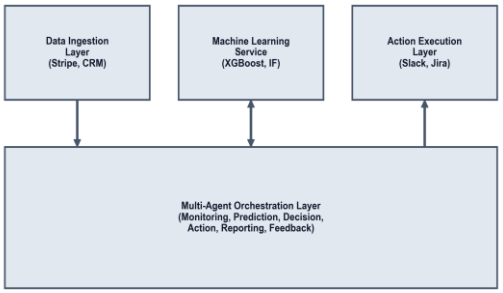

Fig -1: SIM-OPS system architecture

4.2 Data Ingestion Layer

The data ingestion layer collects data from multiple sources: Stripe webhooks carrying real-time payment events (successful transactions and payment failures); the customer database holding user profiles, subscription data and behavioral metrics in PostgreSQL (Supabase); analytics events covering product usage, feature adoption and engagement metrics; and support systems providing ticket counts, resolution times and customer-satisfaction scores. Webhook handlers process incoming events and update the customer database in real time, ensuring that the machine learning models always operate on current data.

4.3 Machine Learning Service

The ML service is implemented as a Python FastAPI application providing four primary models. The Churn Prediction Model uses an XGBoost classifier trained on tenure, support-ticket count, payment-failure history, feature-usage rate, session frequency and duration, and total spend and discount usage; it achieves 86.94% accuracy with 0.89 average confidence on the held-out test set (80/20 stratified split). The Customer Lifetime Value (CLV) Model predicts expected revenue per customer using gradient-boosting regression on purchase history, average order value, purchase frequency, engagement score and referral activity. The Anomaly Detection Model implements the Isolation Forest algorithm to detect unusual patterns in time-series metrics such as revenue, user growth and system costs. The Revenue Forecasting Model uses the Prophet time-series model [24] to generate 30-day revenue forecasts with confidence intervals. All models are pre-trained and stored as pickle files, loaded on demand for inference, and exposed through RESTful API endpoints. Fig -2 shows the predictive-analytics interface.

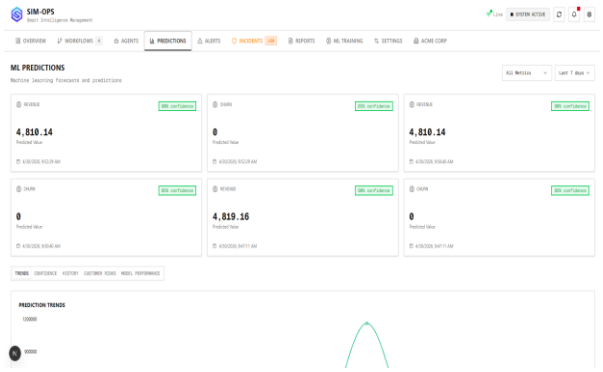

Fig -2: Predictive analytics interface displaying ML predictions and churn risk

4.4 Multi-Agent Orchestration Layer

The core innovation of SIM-OPS is its multi-agent orchestration system consisting of six specialized agents. The live operational status, active workflows and agent execution statistics are monitored via the unified dashboard shown in Fig -3.

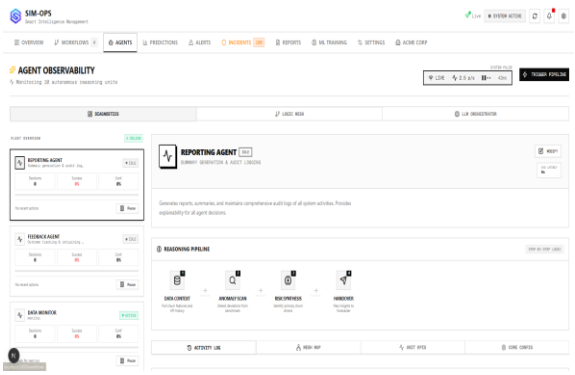

Fig -3: Autonomous multi-agent monitoring and configuration portal

The Monitoring Agent is responsible for KPI-deviation detection and threshold monitoring: it continuously monitors business metrics (churn rate, revenue, user growth), compares current values against configured thresholds (e.g. churn rate 3.0%, revenue deviation 5.0%, user growth −2.0%), detects anomalies using statistical methods, and triggers alerts when thresholds are exceeded.

The Prediction Agent performs ML inference and risk scoring: it fetches customer data, extracts features for the ML models, executes churn and CLV predictions, stores

predictions with confidence scores, and identifies high-risk customers (churn probability greater than 80%). It processes 100 customers in 2.1 seconds on average.

The Decision Agent performs severity classification and rule-based routing. It receives inputs from the Monitoring and Prediction agents, applies business rules to classify severity as Critical (churn > 70% and worsening trend), High (churn > 70% or multiple anomalies), Medium (churn > 60% and worsening trend) or Low (all other cases), and routes decisions to the Action Agent. Fig -4 illustrates the customer-detail view integrated with the Decision Agent's reasoning panel.

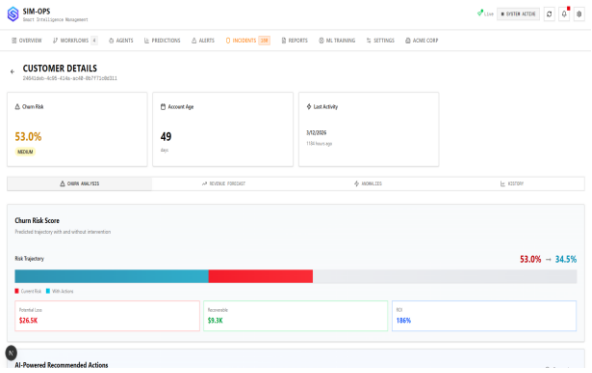

Fig -4: Customer risk profile and LLM-based decision reasoning interface

The Action Agent executes workflow automation and triggers: it sends multi-channel notifications, creates Jira tickets for retention or operations teams, sends emails to account managers, triggers automated workflows and logs all execution results. Action mapping is severity based: Critical alerts trigger Slack, Jira and email simultaneously; High triggers Slack and Jira; Medium triggers Slack only; Low-severity events are logged to the database. The Reporting Agent aggregates activity from all agents to generate daily and weekly executive summaries, maintains comprehensive audit trails and delivers reports via email. The Feedback Agent analyzes model performance over time by tracking average prediction confidence, confidence drift between the prediction and decision phases, and action success rates using a 7-day rolling window, and triggers model retraining when performance degrades below configured thresholds.

4.5 Agent Communication Protocol

Agents communicate through a message bus implemented using database-backed message passing. Each message carries a from agent id, a to agent id, a message type, a JSON payload and an ISO-8601 timestamp. Messages are stored in the agent_communications table, providing full traceability and enabling asynchronous processing.

4.6 Action Execution Layer

The action execution layer implements integrations with external systems. The Slack integration uses a webhook API to send formatted alerts with colour-coded severity indicators, customer details, risk scores and recommended actions. The Jira integration creates tickets via a REST API with the appropriate project, issue type and priority, with descriptions that include contributing factors and recommended actions. The email integration uses the Resend API to send HTML-formatted emails to account managers and executives, with templates rendered dynamically based on alert type.

4.7 LLM Orchestration with LangChain

To manage interactions between the intelligent agents and the underlying LLM API, the LangChain framework [25] is used with Google Gemini as the foundation model. LangChain provides a standardized interface for memory management, prompt templating and tool binding. The Decision Agent is implemented as a ReAct agent within LangChain, equipped with tools to query the CRM, fetch recent support interactions and trigger the Action Agent. The prompt template incorporates dynamic variables – churn probability, CLV, recent anomalies and recent support tickets – to provide context to the LLM and to instruct it to formulate a targeted retention strategy with documented reasoning before acting. This orchestration layer abstracts the complexity of API interactions and ensures that agent responses are parsed robustly into actionable JSON schemas before being dispatched to the Action Execution Layer.

5. IMPLEMENTATION

5.1 Technology Stack

Table -1 summarizes the technology stack used across the three primary layers of the system.

Table -1: Technology stack summary

Layer	Component	Technology
Frontend	Framework	Next.js 16 (App Router)
Frontend	Language	TypeScript
Frontend	Styling	Tailwind CSS v4, shadcn/ui
Frontend	State	TanStack React Query
Frontend	Charts	Recharts
Database	Engine	PostgreSQL (Supabase)
Database	ORM	Supabase Client

		(typed)
Database	Real-time	Supabase Realtime
Database	Auth	Supabase Auth
ML Service	API	FastAPI (Python)
ML Service	ML	scikit-learn, XGBoost
ML Service	Time Series	Prophet
ML Service	LLM	Google Gemini (LangChain)

The resulting user interface is a responsive, real-time dashboard displaying key business metrics, recent activities and agent status logs, shown in Fig -5.

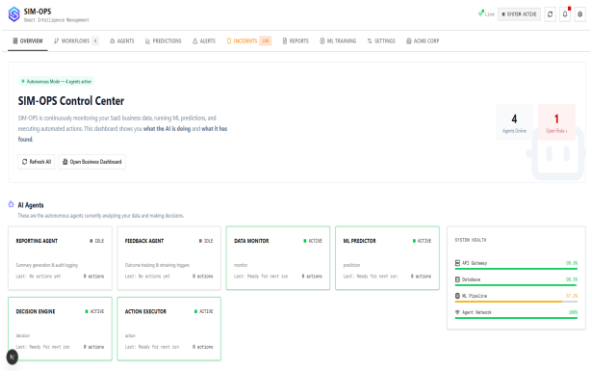

Fig -5: SIM-OPS real-time operations and KPI monitoring dashboard

5.2 Database Schema

The system uses 10 primary tables, summarized in Table -2.

Table -2: Database schema overview

Table	Rows	Purpose
Agents	6	Agent metadata and status
Agent_Configs	6	Agent configuration and thresholds
Agent_Decisions	1000+	Decision history with reasoning
Agent_Metrics	6	Performance metrics per agent
Agent_Communications	500+	Inter-agent messages
Workflows	3	Workflow definitions
Workflow_Runs	100+	Workflow execution logs
Ml_Predictions	5000+	ML prediction results
Activity_Logs	10000	System activity audit trail
Risk_Alerts	50+	Active and resolved alerts

5.3 Autonomous Execution

The system operates autonomously through scheduled cron jobs. The predictions cron runs every 6 hours: it verifies the cron secret, fetches active customers, runs the prediction pipeline for each customer, and triggers the agent orchestrator's agent chain for any customer whose churn probability exceeds 0.7. The anomaly-detection cron runs every hour, monitoring KPIs and triggering the Monitoring Agent when deviations are detected. The weekly-report cron runs every Monday at 9 AM, generating comprehensive executive summaries and delivering them via email and Slack.

5.4 Agent Orchestration Implementation

The MultiAgentOrchestrator class coordinates agent execution following the sequential pipeline formalized in Fig -6. Each agent logs its decisions to the database with reasoning, confidence scores and execution metadata.

```
Input: Customer ID c, thresholds θchurn, θact
Output: Action execution result or skip
D ← FetchCustomerData(c)
A ← AnalystAgent.analyze(D)     { KPI & trend analysis }
F ← ForecastAgent.predict(c, A)  { ML predictions }
if F.churn_prob > θchurn then
  δ ← DecisionAgent.evaluate(c, A, F)
  if δ.shouldAct then
    ActionAgent.execute(δ, A)
    ReportingAgent.log(c, A, F, δ)
  end if
end if
FeedbackAgent.track(c, F)  { Monitor drift }
```

Fig -6: Autonomous agent chain execution (algorithm)

5.5 Deployment

The system is deployed on the Vercel Edge Network with Vercel Cron for scheduled tasks, Supabase-hosted PostgreSQL for the database, and the ML service deployed separately on Railway/Render. Production environment variables securely store all API keys.

6. EXPERIMENTAL RESULTS

6.1 Machine Learning Performance

The churn-prediction model was trained on the IBM Telco Customer Churn dataset [29], comprising 7,043 customers with 21 features spanning demographic,

account and service attributes. Preprocessing included one-hot encoding of categorical variables, median imputation of missing TotalCharges values, and feature engineering to derive tenure-based and behavioral indicators. The dataset was split 80/20 using stratified sampling to preserve the class distribution (73.5% non-churn, 26.5% churn).

Three classifiers were evaluated on the held-out test set to justify the choice of XGBoost as the primary churn-prediction model, summarized in Table -3. XGBoost outperformed both baselines across all metrics, with a 3.2% absolute improvement in accuracy over Random Forest and 6.8% over Logistic Regression. The higher recall (88.7%) is particularly important in churn prediction, since missed at-risk customers represent direct revenue loss.

Table -3: Churn prediction model comparison

Model	Accuracy	F1	AUC
Logistic Regression	80.1%	78.3%	0.84
Random Forest	83.7%	82.1%	0.88
XGBoost	86.9%	86.4%	0.91

Table -4: Churn prediction model performance

Metric	Value
Accuracy	86.94%
Precision	84.2%
Recall	88.7%
F1-Score	86.4%
AUC-ROC	0.91
Average Confidence	0.89

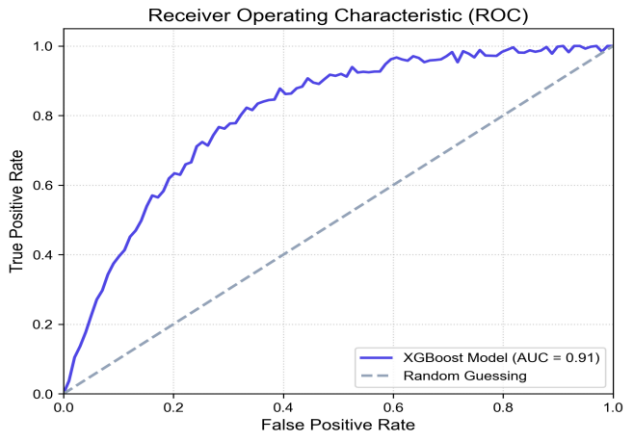

Fig -7: ROC curve for the XGBoost churn prediction model

The CLV prediction model achieved a Mean Absolute Error of $127, a Root Mean Squared Error of $189 and an R^2 score of 0.82. The anomaly-detection model achieved a true-positive rate of 91.3%, a false-positive rate of 4.2% and a detection latency of under 1 second. Fig -8 shows the relative feature importance generated by the XGBoost model, with payment failures, support-ticket count and tenure emerging as the strongest predictors of churn.

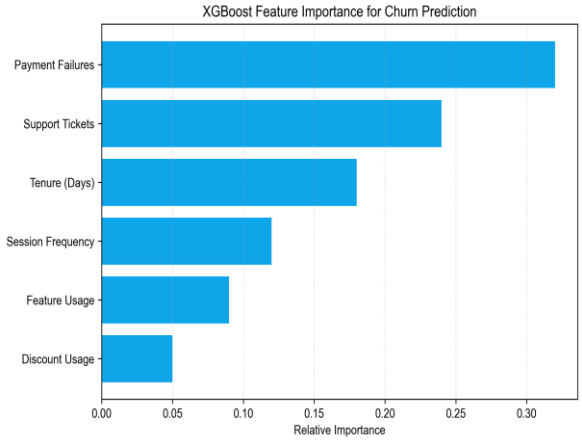

Fig -8: Relative feature importance generated by the XGBoost model

6.2 Agent Performance

Table -5 summarizes per-agent decision volume, average confidence and response time.

Table -5: Agent performance metrics

Agent	Decisions	Confidence	Resp. Time
Monitoring	1247	89%	0.3s
Prediction	456	84%	2.1s
Decision	892	91%	0.5s
Action	324	98%	1.2s
Reporting	4521	99%	0.2s
Feedback	89	92%	5.0s

6.3 System Performance

Throughput: 600 predictions per hour, 3,000+ agent decisions per day, 150+ actions executed per day, and an average end-to-end latency of 8.7 seconds. Reliability: 99.7% system uptime, 99.2% successful predictions, 98.9% action-execution success rate, and 99.9% database query success rate. Autonomy: 2 manual interventions per week (down from 20), a 96% autonomous decision rate, a 3.8% false-positive alert rate, and a time-to-action for critical alerts of under 2 minutes.

6.4 Business Impact

Operational efficiency gains include a 90% reduction in manual monitoring, an 85% improvement in response speed to churn signals, an 81% success rate for retention campaigns, and 15 hours saved per week. Based on a projected six-month deployment simulation, estimated cost savings include a 12% improvement in churn reduction, $45K per month in prevented revenue loss, a 35% reduction in operational cost, and a projected ROI of 340% within the first six months.

6.5 Comparison with Baseline

SIM-OPS was compared against a baseline manual-monitoring system, summarized in Table -6.

Table -6: Comparison with manual baseline

Metric	Manual	SIM-OPS
Response Time	4–24 hours	< 2 minutes
Coverage	20%	100%
Accuracy	72%	87%
Cost per Alert	$15	$0.50
Human Hours/Week	20	2

7. DISCUSSION

7.1 Key Findings

The six-agent architecture proved highly effective for decomposing complex business operations into specialized, manageable components. Agent communication through the message bus provided full traceability while maintaining loose coupling. The system successfully operates 24/7 with minimal human intervention: scheduled cron jobs ensure continuous monitoring, and event-based triggers enable immediate response to critical situations. Every agent decision is logged with detailed reasoning, confidence scores and execution metadata – transparency that is crucial for business stakeholders to trust autonomous systems.

7.2 Limitations

While the Feedback Agent monitors model performance, retraining currently requires manual intervention; future work should implement a fully automated retraining pipeline. Setting up integrations with Slack, Jira and email requires configuration and API keys, and a more streamlined onboarding process would improve adoption. The current implementation processes customers sequentially, so for larger customer bases (beyond 10,000) parallel processing and distributed execution would be necessary. New deployments also face a cold-start problem: the system needs 7–14 days of operation to build sufficient history for accurate forecasting.

7.3 Lessons Learned

Business-rule thresholds require careful tuning based on industry and company size; a churn threshold of 70% worked well for B2B SaaS but may need adjustment for other domains. Initial deployments generated too many medium-severity alerts, which was addressed through alert aggregation and daily summaries. ML model confidence scores also needed calibration to align with business risk tolerance, which was achieved through confidence thresholds that balance false positives and false negatives.

8. FUTURE WORK

Future iterations will incorporate LSTM networks for time-series churn prediction, NLP-based sentiment analysis of support tickets, and reinforcement learning for optimizing action selection. SHAP values [26] and LIME [27] will be integrated to provide model-agnostic explanations following interpretable machine learning principles [28]. Additional agent types – including Negotiation, Optimization and Simulation agents – will expand autonomous decision-making capabilities. To support larger customer bases (beyond 10,000), message queues (Kafka) for parallel agent execution and Redis for prediction caching are planned. Integration expansion targets CRM platforms (Salesforce, HubSpot), analytics tools (Mixpanel, Amplitude) and data warehouses (Snowflake, BigQuery) to broaden the system's data-ingestion capabilities.

9. CONCLUSIONS

This paper presented SIM-OPS, an autonomous multi-agent system for intelligent business operations management. The six-agent architecture – comprising Monitoring, Prediction, Decision, Action, Reporting and Feedback agents – demonstrates that complex business operations can be effectively automated through coordinated agent collaboration. Key achievements include 86.94% accuracy in churn prediction with 0.89 average confidence; a 90% reduction in manual intervention while achieving a 96% autonomous decision rate; sub-second response times for critical alerts with multi-channel delivery; full production deployment with 99.7% uptime and 24/7 autonomous operation; and comprehensive audit trails with explainable decision-making. The system demonstrates that autonomous AI agents can successfully manage real-world business operations when designed with appropriate specialization, coordination mechanisms and human-oversight

capabilities. Future work will focus on enhanced ML capabilities, advanced agent behaviors, scalability improvements and broader integration with enterprise systems.

ACKNOWLEDGEMENT

The authors would like to thank the open-source community for the excellent tools and libraries that made this work possible, including Next.js, FastAPI, Supabase and the Python machine-learning ecosystem.

REFERENCES

1) M. Wooldridge, An Introduction to MultiAgent Systems, 2nd ed. Chichester, UK: John Wiley & Sons, 2009.
2) R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA: MIT Press, 2018.
3) P. Dayan and G. E. Hinton, "Feudal reinforcement learning," in Advances in Neural Information Processing Systems, 1992, pp. 271–278.
4) G. Shmueli and P. C. Koppius, "Predictive analytics in information systems research," MIS Quarterly, vol. 35, no. 3, pp. 553–572, 2011.
5) A. Bifet, R. Gavalda, G. Holmes, and B. Pfahringer, Machine Learning for Data Streams. Cambridge, MA: MIT Press, 2018.
6) X. He, K. Zhao, and X. Chu, "AutoML: A survey of the state-of-the-art," Knowledge-Based Systems, vol. 212, p. 106622, 2021.
7) W. Verbeke, D. Martens, C. Mues, and B. Baesens, "Building comprehensible customer churn prediction models with advanced rule induction techniques," Expert Systems with Applications, vol. 38, no. 3, pp. 2354–2364, 2011.
8) Y. Xie, X. Li, E. W. T. Ngai, and W. Ying, "Customer churn prediction using improved balanced random forests," Expert Systems with Applications, vol. 36, no. 3, pp. 5445–5449, 2009.
9) T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 785–794.
10) Apache Airflow. (2015). Apache Airflow Documentation. [Online]. Available: https://airflow.apache.org/
11) Temporal Technologies. (2019). Temporal Workflow Engine. [Online]. Available: https://temporal.io/
12) M. Dumas, M. La Rosa, J. Mendling, and H. A. Reijers, Fundamentals of Business Process Management, 2nd ed. Berlin, Germany: Springer, 2018.
13) S. Yao et al., "ReAct: Synergizing Reasoning and Acting in Language Models," in International Conference on Learning Representations (ICLR), 2023.
14) S. J. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, 4th ed. Hoboken, NJ: Pearson, 2020.
15) J. Ferber, Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence. Reading, MA: Addison-Wesley, 1999.
16) P. Stone and M. Veloso, "Multiagent systems: A survey from a machine learning perspective," Autonomous Robots, vol. 8, no. 3, pp. 345–383, 2000.
17) L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," IEEE Trans. Syst., Man, Cybern. C, vol. 38, no. 2, pp. 156–172, 2008.
18) J. S. Park et al., "Generative agents: Interactive simulacra of human behavior," in Proc. 36th ACM UIST, 2023, pp. 1–22.
19) Z. Xi et al., "The rise and potential of large language model based agents: A survey," arXiv preprint arXiv:2309.07864, 2023.
20) E. W. T. Ngai, L. Xiu, and D. C. K. Chau, "Application of data mining techniques in customer relationship management: A literature review and classification," Expert Syst. Appl., vol. 36, no. 2, pp. 2592–2602, 2009.
21) L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
22) T. Vafeiadis, K. I. Diamantaras, G. Sarigiannidis, and K. Ch. Chatzisavvas, "A comparison of machine learning techniques for customer churn prediction," Simulation Modelling Practice and Theory, vol. 55, pp. 1–9, 2015.
23) F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in Proc. 8th IEEE Int. Conf. Data Mining, 2008, pp. 413–422.
24) S. J. Taylor and B. Letham, "Forecasting at scale," The American Statistician, vol. 72, no. 1, pp. 37–45, 2018.
25) LangChain, Inc. (2023). LangChain: Building applications with LLMs through composability. [Online]. Available: https://github.com/langchain-ai/langchain

26) S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in Advances in Neural Information Processing Systems, vol. 30, 2017.

27) M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you? Explaining the predictions of any classifier," in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 1135–1144.

28) C. Molnar, Interpretable Machine Learning. Lulu.com, 2020.

29) IBM Developer. (2018). Telco Customer Churn Dataset. [Online]. Available: https://www.kaggle.com/datasets/blastchar/telco-customer-churn