Journal of Software Engineering and Applications, 2025, 18(12), 542-563
https://www.scirp.org/journal/jsea
ISSN Online: 1945-3124
ISSN Print: 1945-3116

SAGENT: An Intelligent System for the Management of Complex Workflows

Hong Wu, Vijay K. Madisetti

College of Computing, Georgia Institute of Technology, Atlanta, GA, USA
Email: vkm@gatech.edu

How to cite this paper: Wu, H. and Madisetti, V.K. (2025) SAGENT: An Intelligent System for the Management of Complex Workflows. *Journal of Software Engineering and Applications*, **18**, 542-563.
https://doi.org/10.4236/jsea.2025.1812031

Received: September 15, 2025
Accepted: December 20, 2025
Published: December 23, 2025

Abstract

In today's fast-paced, data-driven environments, many industries face challenges in handling time-consuming, complex human-machine tasks, such as chargebacks, dispute resolution, and insurance claim verifications. These tasks often involve multiple human-machine interactions and require the processing of unstructured multimedia data and nonstandardized man-machine interfaces, leading to inefficiencies, security risks, and delays. The absence of an integrated, autonomous system capable of managing such complex workflows hinders operational effectiveness. This paper presents SAGENT (System for Autonomous Graph-Enhanced Multimodal LLM AgeNTs), an innovative framework designed to automate and optimize these tasks using advanced multimodal language models (LLMs) combined with graph-based techniques. By leveraging the synergy of graph-enhanced data structures and AI-driven decision-making processes, SAGENT offers a solution to reduce human intervention and errors, accelerate resolution times, and improve overall task efficiency. We demonstrate the potential of this system in streamlining operations, enhancing accuracy, and maximizing ROI for industries reliant on the processing of complex, unstructured data.

Keywords

AI Agent, Automated Decision Making, Autonomous Systems, Financial Technology, Graph-Based Learning, Insurance Technology, Multimodal Large Language Models (LLMs), Multi-Agent Systems (MAS), Operational Efficiency, Operational Intelligence, Process Optimization, Unstructured Data, Workflow Automation

1. Introduction

In the current dynamic and data-intensive landscape, industries globally are confronted with the challenge of labor-intensive human-machine tasks in the execu-

tion of business processes that yield limited Return On Investment (ROI). Predominantly, these tasks pertain to ad hoc human-machine interface activities, such as chargeback processing, dispute resolution, and insurance claim verification, which often demand significant human involvement. These processes are typically impeded by their reliance on unstructured multimedia data—including documents, images, audio, and video—that require complex human judgment for analysis, interpretation, and resolution. The extensive volume of such tasks, coupled with the lack of effective automation tools and sporadic human-machine interactions, results in inefficiencies, delays, and increased operational costs. Consequently, enterprises and organizations are restricted in their ability to expand operations or enhance performance while maintaining the requisite accuracy and consistency.

A chief obstacle in resolving these challenges is the lack of an integrated system that can independently and securely manage intricate workflows encompassing various data modalities and diverse human-machine interfaces, necessitating sophisticated decision-making. Conventional systems are ill-equipped to handle the complex, multilayered data interactions fundamental to these processes and lack the capability to learn from extensive, unstructured information in multiple formats. In the absence of a holistic solution, industries persistently encounter prolonged resolution durations, elevated operational expenses, and suboptimal performance.

This paper introduces SAGENT (System for Autonomous Graph-Enhanced Multimodal LLM AgeNTs), a novel framework devised to address these salient challenges. Through the integration of advanced multimodal language models (LLMs) with graph-based data structures, SAGENT is engineered to automate and enhance the optimization of complex human-machine interactions, including tasks such as chargeback management, dispute resolution, and insurance claim verification. The amalgamation of these technologies generates a synergistic effect that augments data processing and human-machine capabilities, enhances the precision of decision-making, fortifies security measures, and considerably diminishes the necessity for human intervention.

Graph-based methodologies are exceptionally well-suited for this endeavor, as they provide a robust mechanism for capturing the interrelationships among diverse data elements and facilitating the interpretation of unstructured multimedia content. In conjunction with advanced Large Language Models (LLMs), which excel in comprehending and generating text akin to human communication, SAGENT is capable of processing and resolving disputes with efficiency.

Through the implementation of SAGENT, we intend to illustrate its capacity to optimize workflows, address discrepancies in human-machine interfaces, decrease resolution durations, and enhance task efficiency. Our objective is to furnish industries that manage substantial volumes of unstructured data with a tool that enables productivity enhancement, cost reduction, and maximization of Return On Investment (ROI).

2. Related Work

Table 1 provides a comparison between SAGENT and referenced papers [1]-[8].

Table 1. Comparison between SAGENT and referenced papers.

Paper	Focus Area	Key Limitation	How SAGENT Improves
Large Multimodal Agents: A Survey (2402.15116)	Theoretical survey of multimodal agents	Lacks industry-specific implementations and graph-enhanced architectures	Provides practical industry application with graph-enhanced data structures for business processes
AGENT AI: SURVEYING THE HORIZONS (2401.03568v)	Academic survey of multimodal interaction	Academic focus without industrial application	Implements purpose-built solution with concrete ROI metrics for specific business workflows
Optimus-1: Hybrid Multimodal Memory (2408.03615v2)	Long-horizon tasks with multimodal memory	Limited operational implementation for business workflows	Creates an end-to-end system with specialized agents for practical business applications
OpenOmni: Collaborative Open Source Tool (2408.03047v2)	Tool for building conversational agents	Tool-focused rather than solution-focused	Delivers complete business solution rather than development framework
Chatlaw: Multi-Agent Legal Assistant (2306.16092v2)	Legal domain with knowledge graph enhancement	Domain-specific to legal assistance	Extends to multiple domains with comprehensive business process integration
AriGraph: Knowledge Graph World Models (2407.04363v2)	Episodic memory with knowledge graphs	Focus on representation rather than business workflows	Implements practical integration with business systems for operational efficiency
LLMs for Knowledge Graph Construction (2305.13168v4)	Survey of knowledge graph capabilities	Survey-oriented without implementation	Provides complete system implementation with multimodal integration
Knowledge Graph Prompting for QA (2308.11730v3)	Multi-document question answering	Narrow focus on interactive QA	Creates autonomous workflow execution without human questioning

Key Sagent Advantages across All Comparisons

1) Practical Implementation: SAGENT provides a complete, implemented system rather than theoretical frameworks or surveys.

2) Industry-Specific Focus: Targets high-value business processes like chargeback management with measurable ROI.

3) Autonomous Workflow Execution: Fully automates complex human-machine processes without requiring human intervention.

4) Multimodal Integration: Processes diverse evidence types (text, images, audio, video) through specialized agents.

5) Business System Integration: Connects directly with operational tools and APIs (Stripe, Email, Twilio, Intercom).

6) Quantifiable Metrics: Demonstrates concrete business value through win rates and evidence correlation.

7) Graph-Enhanced Architecture: It leverages knowledge graphs for practical

relationship mapping between evidence elements.

8) Orchestrated Multi-Agent Design: Coordinates specialized agents for different tasks within a comprehensive workflow.

2.1. List of Existing Solutions

Take the example of the chargeback protection industry. Here's the comparison between each solution, and why SAGENT is a better system overall.

2.2. Why Current Solutions Do Not Work

Although many companies try to tackle chargebacks effectively, most of them still rely on heavy human labor to do their work. In the case of Chargeback911 and ChargebackStop, a non-insignificant monthly fee prevents most SMBs (small business owners) from using their services.

Alternative entities offer insurance for transactions, providing coverage for losses incurred due to chargebacks, contingent upon the chargebacks being deemed lost. Nevertheless, these insurance offerings are often accompanied by numerous conditions. Merchants are required to adhere to specific performance metrics, such as maintaining a low rate of disputes, in order to be eligible for such insurance coverage. Furthermore, should a transaction be classified as high risk, the insurer reserves the right to abstain from insuring the transaction, thereby leaving the financial liability with the merchants.

Ultimately, insurance companies inherently seek to mitigate risk, primarily to avoid financial losses. This core tenet fundamentally opposes the objective of merchants to secure protection in instances where chargebacks result in financial deficits. These limitations across existing commercial offerings are summarized in Table 2, which contrasts insurance-based and labor-intensive approaches with the fully autonomous SAGENT architecture (Figure 1).

Table 2. Comparison of chargeback solutions.

Solution	Insurance	Human Labor	Affordability	AI Enabled?	Autonomous?
	Bad ROIs, expensive	Long turnaround time Labor cost, human errors	Not affordable for SMBs/Mom & Pops		
Kount Chargeback Insurance	✓	✓	✓	□	□
FUGU Chargeback Guarantee	✓	✓	✓	□	□
Stripe Chargeback Protection	✓	✓	✓	□	□
ChargebackStop	□	✓	✓	□	□
Chargeback911	□	✓	✓	□	□
ZBrain AI dispute agent	□	□	✓	✓	□
Lightgreen! 25 SAGENT	No insurance needed	No human errors & labor cost	Usage-based, affordable Win-Win pricing model	Yes	Fully Autonomous

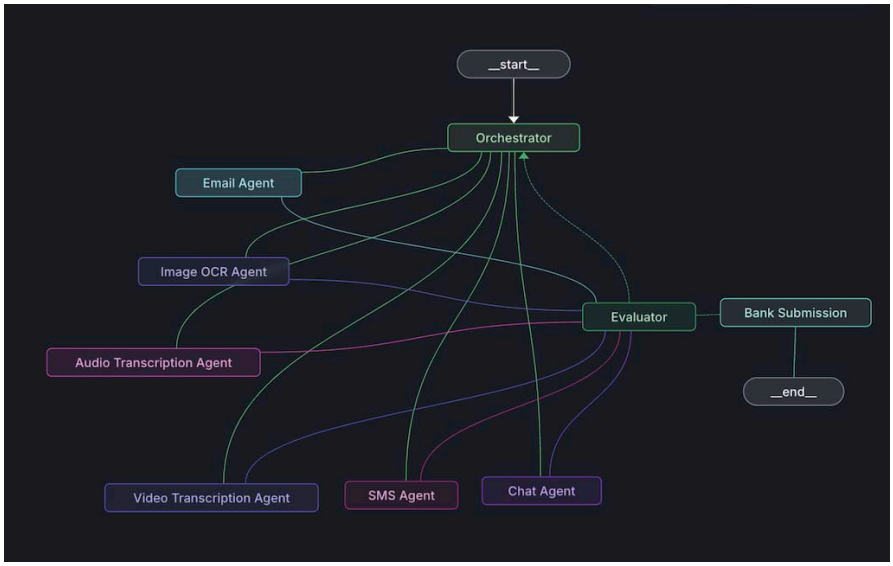

Figure 1. High-level SAGENT architecture.

3. Proposed Methodology

Consider the instance of chargebacks. When financial institutions initiate chargebacks (transaction disputes) against merchants, it becomes incumbent upon the merchants to collect diverse forms of evidence, organize these into persuasive arguments, and present them to banks for consideration. This entire procedure is characterized by significant labor intensity. Fortunately, it is possible to employ AI agents to perform specific tasks and to completely automate this process.

3.1. System Architecture

SAGENT utilizes an event-driven architectural paradigm that is engineered to autonomously address chargeback occurrences in real-time. The initiation of the system is facilitated by webhook notifications, which are activated when payment processors (such as Stripe) record a novel chargeback incident. This activation mechanism, predicated upon webhooks, ensures that SAGENT remains in a quiescent state until necessitated, thereby conserving computational resources while perpetually maintaining operational readiness.

Upon receipt of a chargeback notification, the Orchestrator agent—powered by an advanced high-reasoning LLM such as OpenAI o1 or DeepSeek r1—functions as the principal coordinator within the system. The Orchestrator initially conducts an analysis of the chargeback details and assesses the particular circumstances surrounding the dispute. This analytical process involves retrieving the transaction history, scrutinizing prior interactions with the customer, and identifying specific requirements pertinent to the dispute type. Following this

Comprehensive evaluation, the Orchestrator decomposes the evidence-gathering process into tasks specific to each domain and delegates these tasks to specialized agents for execution. The multi-agent framework is realized through the implementation of LangGraph, which offers a robust and scalable platform for the

coordination of agents. Each individual agent is endowed with specific tools and Application Programming Interfaces (APIs) essential for fulfilling its designated role. For instance, the Stripe Agent establishes a direct interface with Stripe's dispute API to acquire transaction details and subsequently submit evidence packages. The Email Agent integrates with email services to identify pertinent customer communication, while the Image OCR Agent is responsible for processing any uploaded receipts or documentation. Additional agents manage Short Message Service (SMS) communications via the Twilio Agent, engage in customer support interactions through the CS Agent, and process multimedia content using the Audio and Video Agents.

A significant feature of SAGENT's architecture is its capacity to manage the asynchronous characteristics of evidence gathering. Given that numerous tasks related to evidence collection necessitate interactions with external stakeholders (e.g., merchants responding to information requests), the system functions on a notification-driven basis for these interactions as well. Upon the emergence of new evidence—such as a merchant replying to an email request—the system is reactivated, enabling the Orchestrator to reevaluate the case status by integrating the new information into the evidence graph.

The gathered evidence is systematically arranged utilizing a knowledge graph framework, implemented via Neo4j or Zep. This framework facilitates the establishment of interconnections among disparate elements of evidence, thereby constructing a holistic representation of the case. The graph structure permits SAGENT to generate coherent narratives concerning transaction validity by linking pertinent evidence components.

Knowledge-graph–augmented LLM agents have been shown to improve grounding, reasoning consistency, and multi-document synthesis [1]-[4]. Inspired by these approaches, the SAGENT knowledge graph encodes relationships among evidence types, dispute rules, transaction contexts, and temporal constraints to support structured retrieval and agent coordination.

As evidence is progressively gathered, the Evaluation agent persistently evaluates the present status employing a pretrained machine learning model—either Random Forest or XGBoost—calibrated with over 30,000 historical chargeback instances. This agent evaluates various determinants such as the robustness of the evidence, the proximity to the submission deadline, and the forecasted probability of success. Based on this analysis, the agent adjudicates whether the assemblage of evidence is adequate for submission or if further evidence acquisition is necessary, consistently optimizing for the maximal probability of success within the temporal limitations.

Upon the determination that the evidence is adequate—or when time constraints necessitate action—the Final Submission agent formulates a persuasive argument by systematically arranging the evidence into a coherent timeline, emphasizing the most compelling supporting elements, and formatting the submission in accordance with the specific requirements of the payment processor and the issuing bank. Subsequently, the submission is automatically transmitted via the desig-

nated API, thereby concluding the automated dispute resolution process.

The multi-agent, event-driven architecture facilitates SAGENT's capacity to autonomously manage chargeback disputes, encompassing the entire process from initial notification to evidence collection and final submission, while preserving the adaptability necessary to accommodate the specific requirements of each case and the variable timing of external communications.

3.2. Underlying Agents

Recent agent frameworks have emphasized multimodal memory integration and long-horizon task execution [7] [8]. Consistent with these findings, each SAGENT subsystem maintains structured episodic memory supporting retrieval, tool calling, and cross-agent state synchronization.

1) *Orchestrator/Planner Agent*: The Orchestrator/Planner agent employs a language model with robust reasoning capacity. While we have selected OpenAI o1 for this purpose, any equally proficient language model, such as DeepSeek r1, is suitable.

Upon receipt of a chargeback event, this agent is activated. It will then analyze the chargeback status and associated details in order to segment the required tasks into domain-specific individual components. Each component task will subsequently be executed by a designated agent.

2) *Multi-Modal Agents*: Each agent is provided with a distinct instrument to perform their assignments effectively. In contrast to the standard web search functionality found in Language Learning Models (LLMs), this instrument typically constitutes an API, which the agent can utilize to obtain precise results as required.

Recent advances in multimodal and multi-agent LLM systems have demonstrated strong capabilities in complex decision-making and tool-augmented workflows [5] [6]. These developments motivate the design of SAGENT as an autonomous, multimodal agentic framework for end-to-end dispute resolution.

Listing 1. EmailAgent via Gmail

```
class EmailAgent(AgentWithToolsAndMemory):
    """Agent for handling email-related tasks using
    ↪ Gmail."""

    def __init__(
        self,
        llm: BaseChatModel,
        memory: Optional[ConversationBufferMemory]
        ↪ = None,
    ):
        # Initialize Gmail service
        self.service =
        ↪ self._get_gmail_service(credentials_path
        ↪ token_path)
        # Create Gmail tools
        tools = [... list of email tools ...]
        super().__init__(llm=llm, tools=tools,
        ↪ prompt=system_prompt, memory=memory,
        ↪ name="Email Agent")
```

Listing 2. ChatAgent via Intercom

```
class ChatAgent(AgentWithToolsAndMemory):
    """Agent for handling chat conversations using
    ↪  Intercom's Messages API."""

    def __init__(
        self,
        llm: BaseChatModel,
        memory: Optional[ConversationBufferMemory]
        ↪  = None,
        api_key: Optional[str] = None
    ):
        # Create Intercom tools
        tools: List[BaseTool] = [ ... define list
        ↪  of Intercom tools ... ]
        super().__init__(llm=llm,
        ↪  system_prompt=system_prompt,
        ↪  tools=tools, memory=memory, name="Chat
        ↪  Agent")
```

Listing 3. SMSAgent via Twilio

```
class SMSAgent(AgentWithToolsAndMemory):
    """Agent for handling SMS-related tasks using
    ↪  Twilio."""

    def __init__(
        self,
        llm: BaseChatModel,
        memory: Optional[ConversationBufferMemory]
        ↪  = None,
        account_sid: Optional[str] = None,
        auth_token: Optional[str] = None,
        from_number: Optional[str] = None,
    ):
        # Create Twilio tools
        tools: List[BaseTool] = [ ... list of
        ↪  Twilio tools ... ]

        super().__init__(llm=llm,
        ↪  system_prompt=system_prompt,
        ↪  tools=tools, memory=memory, name="SMS
        ↪  Agent")
```

Listing 4. ImageOCRAgent via Google Gemini

```
class ImageOCRAgent(AgentWithToolsAndMemory):
    """Agent for extracting text from images using
    ↪  Google Gemini."""

    def __init__(
        self,
        llm: Optional[BaseChatModel] = None,
        memory: Optional[ConversationBufferMemory]
        ↪  = None,
```

```
        api_key: Optional[str] = None,
    ):
        genai.configure(api_key=api_key)
        self.model =
        ↪  genai.GenerativeModel('gemini-pro-vision

        super().__init__(llm=llm,
        ↪  system_prompt=system_prompt,
        ↪  memory=memory, name="Image OCR Agent")

    async def extract_text_from_image(self,
    ↪  image_path: str) -> str:
        """Extract text from an image using
        ↪  Gemini."""
        # Load the image
        image =
    ↪   genai.types.Image.load_from_file(image_path

        # Generate content
        response = self.model.generate_content([
            "Extract all text from this image. If
            ↪  there are multiple languages,
            ↪  identify them. Format the output
            ↪  clearly.",
            image
        ])
        return response.text
```

Listing 5. ImageOCRAgent via GPT-4

```
class AudioTranscriptionAgent(
            AgentWithToolsAndMemory):
    """Agent for transcribing audio files using
    ↪  GPT-4's multimodal capabilities."""

    def __init__(
        self,
        llm: Optional[BaseChatModel] = None,
        memory: Optional[ConversationBufferMemory]
        ↪  = None,
        api_key: Optional[str] = None,
    ):
        super().__init__(
            llm=llm,
            system_prompt=system_prompt,
            tools=[],  # No additional tools needed
            ↪  for basic transcription
            memory=memory,
            name="Audio Transcription Agent"
        )

    async def transcribe_audio(self, audio_path:
    ↪  str) -> str:
        """Transcribe audio using GPT-4's
        ↪  multimodal capabilities."""

        # Convert audio to a format GPT-4 can
        ↪  handle (if needed)
        audio = AudioSegment.from_file(audio_path)
        # Read the audio file
        with open(temp_path, 'rb') as audio_file:
            audio_data = audio_file.read()
        # Create messages for transcription
        messages = [
            HumanMessage(content=[
                {"type": "text", "text": "Please
                ↪  transcribe this audio file.
                ↪  Include any relevant context or
                ↪  analysis."},
                {"type": "audio", "audio":
                ↪  audio_data}
            ])
        ]
        # Get transcription
        response = await self.llm.ainvoke(messages)
        return response.content
```

3) *Video Transcript Agent*: The video transcript agent receives a video provided by a merchant and employs GPT4's multimodal large language model to transcribe the video's content. It then formats and summarizes this content to ensure compatibility with language model processing. The resulting information is subsequently archived within the chargeback's knowledge database and is utilized in formulating the final written argument for submission.

Listing 6. VideoTranscriptionAgent via GPT-4

```
class VideoTranscriptionAgent(
            AgentWithToolsAndMemory):
    """Agent for transcribing video files using
    ↪  GPT-4's multimodal capabilities."""

    def __init__(
        self,
        llm: Optional[BaseChatModel] = None,
        memory: Optional[ConversationBufferMemory]
        ↪  = None,
        api_key: Optional[str] = None,
    ):
        super().__init__(
            llm=llm,
            system_prompt=system_prompt,
            tools=[],  # No additional tools needed
            ↪  for basic transcription
            memory=memory,
            name="Video Transcription Agent",
        )

    def _extract_key_frames(self, video_path: str,
    ↪  num_frames: int = 5) -> list:
        """Extract key frames from video."""
        frames = []
        cap = cv2.VideoCapture(video_path)
        total_frames =
        ↪  int(cap.get(cv2.CAP_PROP_FRAME_COUNT))

        # Calculate frame interval
        interval = max(1, total_frames //
        ↪  num_frames)

        return frames

    async def transcribe_video(self, video_path:
    ↪  str) -> str:
        """Transcribe video using GPT-4's
        ↪  multimodal capabilities."""
        # Extract audio from video and Extract key
        ↪  frames
        frames =
        ↪  self._extract_key_frames(video_path)
        # Create messages for transcription
        messages = [
            HumanMessage(content=[
                {"type": "text", "text": "Please
                ↪  transcribe this video file.
                ↪  Include any relevant context,
                ↪  analysis, and visual
                ↪  descriptions."},
                {"type": "audio", "audio":
                ↪  audio_data},
                *[{"type": "image", "image": frame}
                ↪  for frame in frame_data]
            ])
        ]

        # Get transcription
        response = await self.llm.ainvoke(messages)
        return response.content
```

For full implementation details and extended code listings, please refer to the online appendix available at: https://github.com/lordhong/sagent.

4) *Evaluation Agent*: The evaluation agent assesses the current state of affairs to ascertain whether the available evidence suffices for submission to the bank or if additional evidence collection is warranted. The agent also considers the submission deadline in making an optimal decision regarding subsequent actions. A robust prediction model was developed utilizing over 30,000 chargeback data records spanning from 2017 to early March 2025. This model is crucial for enabling the evaluator agent to determine the adequacy of a submission, specifically when the winning rate exceeds 50%, for bank submission.

Listing 7. EvaluatorAgent using Prediction Model

```
class EvaluatorAgent(AgentWithToolsAndMemory):
    """Agent for evaluating task completion and
    ↪  determining next steps."""

    def __init__(
        self,
        llm: BaseChatModel,
        tools: List[BaseTool],
        memory: Optional[ConversationBufferMemory]
        ↪  = None,
    ):
        system_prompt = """You are an evaluator
        ↪  agent responsible for determining if
        ↪  tasks are complete.
        You should:
        - Evaluate if the current task is complete
        - Determine if additional steps are needed
        - Decide whether to continue or end the
        ↪  workflow

        Respond with either 'continue' if more work
        ↪  is needed or 'end' if the task is
        ↪  complete."""

        super().__init__(
            llm=llm,
            system_prompt=system_prompt,
            tools=tools,
            memory=memory,
            name="Evaluator",
            description="Evaluates task completion
            ↪  and determines next steps",
        )

        # Initialize ONNX runtime session
        model_path = Path(
        "models/win_rate_prediction_model.onnx")
        if not model_path.exists():
            raise FileNotFoundError(f"Model file
            ↪  not found at {model_path}")

        self.session =
        ↪  ort.InferenceSession(str(model_path))
```

5) *Final Submission Agent*: The submission agent will gather all the evidence collected, create a timeline for the involved chargeback transaction, write a compelling argument, and submit to the bank (via the tool/API). This agent is the final agent for the SAGENT system.

Listing 8. BankSubmissionAgent via Stripe's Disputes Submission API

```
class BankSubmissionAgent(AgentWithToolsAndMemory):
    """Agent for handling bank submission tasks
    ↪  using Stripe's dispute submission API."""

    def __init__(
        self,
        llm: BaseChatModel,
        memory: Optional[ConversationBufferMemory]
        ↪  = None,
        api_key: Optional[str] = None,
    ):
        # Initialize Stripe
        api_key = api_key or
        ↪  os.getenv("STRIPE_API_KEY")
        if not api_key:
            raise ValueError("STRIPE_API_KEY
            ↪  environment variable or api_key
            ↪  parameter is required")

        stripe.api_key = api_key

        # Create Stripe tools
        tools: List[BaseTool] = [
            self._create_dispute_
                submission_tool(),
            self._create_evidence_upload_tool(),
            self._create_dispute_status_tool()
        ]

        system_prompt = """You are a bank
        ↪  submission agent specialized in
        ↪  handling dispute submissions using
        ↪  Stripe.
        You can:
        - Submit disputes to banks
        - Upload evidence for disputes
        - Check dispute status
        - Provide guidance on dispute resolution
        - Analyze dispute patterns
        - Suggest preventive measures

        Always ensure:
        - All required evidence is properly
        ↪  documented
        - Submissions follow bank guidelines
        - Evidence is clear and compelling
        - Timelines are adhered to
        - Communication is professional and
        ↪  clear"""

        super().__init__(
            llm=llm,
            system_prompt=system_prompt,
            tools=tools,
            memory=memory,
            name="Bank Submission Agent",
            description="Handles bank dispute
            ↪  submissions using Stripe",
        )
```

3.3. Security and Compliance Considerations

Given SAGENT's integration with sensitive data sources—including payment APIs (e.g., Stripe), email conversations, and SMS messages—it implements stringent safeguards to ensure data privacy and regulatory compliance. All agent interactions with external systems are governed through authenticated API tokens

secured via a vault-based key management system. Access to personal or financial data is strictly controlled through Principle-of-Least-Privilege (PoLP) permissioning, such that each agent can only access data relevant to its designated task.

To comply with PCI-DSS (Payment Card Industry Data Security Standard) requirements, SAGENT ensures that no full cardholder data is stored or transmitted unencrypted during processing. The Stripe Agent, for example, does not directly handle primary account numbers, and any metadata received through the Stripe Dispute API is immediately encrypted at rest using AES-256 encryption and handled exclusively within secure cloud environments compliant with PCI-DSS SAQ-D standards.

For GDPR (General Data Protection Regulation) alignment, core features of SAGENT enforce data minimization and user consent, respectively. The Email Agent and SMS Agent employ scoped permissions and redact personally identifiable information where irrelevant to dispute resolution. Any processing involving EU subjects follows GDPR regulations, including data subject access rights, the right to be forgotten, and explicit logging of anonymization and processing activities for auditability. Sensitive information is retained only for the minimum period necessary to resolve a case, after which it is destroyed or anonymized.

Additionally, all inter-agent communication and external HTTP/API calls are transmitted using TLS 1.3 to prevent Manin-The-Middle (MITM) attacks. Multimodal content, such as image or video uploads, is scanned using hash-based integrity checks and content-type validation to mitigate injection risks. Finally, audit logs and agent actions are archived with tamperproof logging to facilitate post-hoc reviews and compliance audits. These measures collectively ensure that SAGENT maintains a high standard of security and privacy while remaining compliant with global data protection standards.

4. Testing and Evaluation of Sagent

4.1. Development of Tests

In order to evaluate the performance of SAGENT, we devised an extensive testing framework aimed at assessing the system's efficiency, accuracy, and cost-effectiveness in relation to traditional manual methods. The testing methodology comprised several fundamental components:

1) Framework Selection: Following an extensive assessment of existing multi-agent frameworks, such as HuggingFace, LlamaIndex, AutoGen, and CrewAI, LangGraph was chosen as the implementation platform. LangGraph offered superior flexibility and control over the interactions of agents within our specific domain, facilitating precise management of the intricate workflows associated with chargeback processing.

2) Synthetic Data Generation and Validation: To avoid exposing sensitive customer information while maintaining the statistical integrity of the dataset, we developed synthetic chargeback data based on over 30,000 historical records obtained from industry partners. The anonymization process began by stripping all Personally

Identifiable Information (PII) from the original dataset, including customer names, email addresses, full credit card numbers, and billing data. Category-preserving tokenization was applied to retain semantic structure (e.g., product descriptions and customer communication templates were replaced with category-specific placeholders or synthesized variants). To ensure that the distribution of dispute types, merchant categories, timestamps, and resolution outcomes remained representative, we utilized probabilistic modeling to replicate the original class distributions (e.g., reason codes such as "product not received" versus "fraudulent transaction").

Synthetic data generation was facilitated using controlled perturbations of real transaction attributes combined with template-based evidence generation and LLM-guided content simulation. In order to assess the fidelity of the synthetic dataset, key performance metrics (such as evidence completeness, predicted win rates, and temporal patterns) were compared against a real-world hold-out subset of anonymized records. The correlation of predictive model performance (>0.95 Pearson's r) and similar evidence-outcome patterns in both datasets indicated that the synthetic data preserved the operational characteristics essential for testing SAGENT's functionality under realistic conditions while maintaining full compliance with data privacy constraints.

The synthetic dataset retained essential characteristics of actual chargebacks. Including:

- Transaction details and amounts.
- Dispute reasons and categorizations.
- Evidence types and formats.
- Timing constraints and deadlines.
- Resolution outcomes and decisions.

Each category of evidence plays a crucial role in determining the eventual success rate, and a comprehensive collection of evidence significantly enhances the likelihood of winning the chargeback. Table 3 summarizes the empirical win rates associated with each evidence category in the synthetic dataset.

Table 3. Win rates by evidence category.

Evidence Category	Win Rate
refund policy	0.620690
refund refusal explanation	0.597964
refund policy disclosure	0.593504
submission file	0.540998
customer name	0.536863
service documentation	0.533379
customer communication	0.512075
customer signature	0.509638
submission text	0.479622
product description	0.478050
receipt	0.476823
service date	0.471950
customer email	0.466379
billing address	0.441803

Our examination of the interplay among win rates, evidence completeness, and submission timing uncovered fundamental insights into the dynamics of chargeback resolution. As depicted in Figure 2, we charted chargeback win rates alongside evidence completeness (quantified as Mean True Features) relative to submission timeframes. The findings reveal a strong positive association between the volume of evidence elements furnished and successful dispute resolution outcomes. Notably, the timing of submissions to financial institutions exhibited a negligible impact on win probability, indicating that the quality and thoroughness of evidence are the primary determinants of success rather than the speed of submission. Nevertheless, prompt submissions confer substantial benefits to merchants, such as accelerated capital recovery and enhanced cash flow management, despite not directly affecting resolution outcomes.

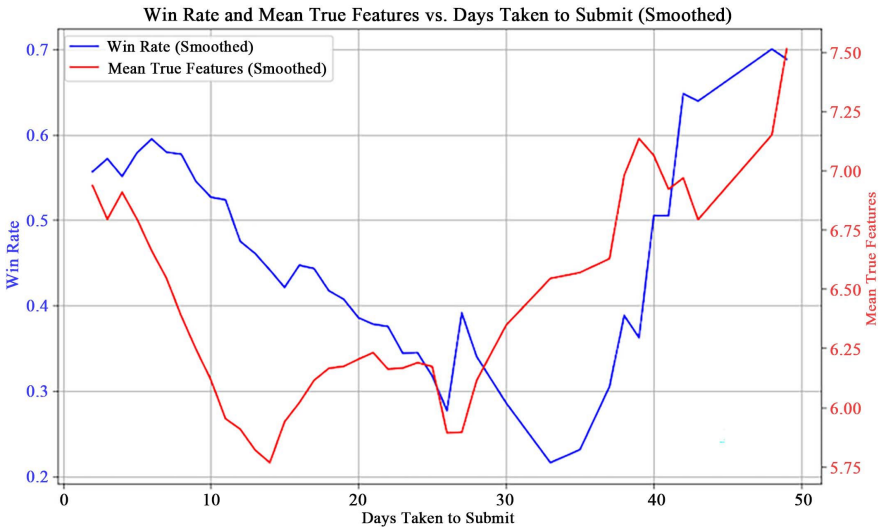

Figure 2. Chargeback win rate correlates with the amount of evidence.

3) Agent-Specific Testing: Each specialized agent was tested individually before integration:

- *Email Agent*: Evaluated on email retrieval, thread identification, and content extraction.
- *Chat Agent*: Tested for customer interaction analysis and context extraction.
- *SMS Agent*: Assessed for message retrieval and content parsing.
- *Image OCR Agent*: Validated on document recognition accuracy and text extraction.
- *Audio and Video Transcription Agents*: Measured on transcription accuracy and content summarization.
- *Evaluation Agent*: Tested on prediction model accuracy using historical outcomes.
- *Bank Submission Agent*: Assessed on formatting and submission success rates.

4) System Integration Testing: The comprehensive evaluation of the SAGENT system was conducted using simulated end-to-end chargeback scenarios. The tests

were designed to model a range of chargeback types, availability of evidence scenarios, and constraints imposed by response deadlines. Key performance indicators were documented at each stage of the processing.

5) Comparative Analysis: A comparative testing framework was developed to benchmark **the** performance of SAGENT relative to existing CS/OPS teams, with an emphasis on measuring:

- Evidence gathering turnaround times for different evidence categories.
- Completeness of evidence packages (percentage of relevant evidence collected).
- Chargeback winning rates.
- Operational costs per chargeback processed.

6) Performance Metrics: Custom metrics were developed to quantify SAGENT's **effectiveness**:

- Turnaround time reduction percentage.
- Evidence completeness improvement.
- Win rate enhancement.
- Cost efficiency ratio.
- System reliability (successful completion rate).

The employed testing methodology enabled a comprehensive evaluation of SAGENT's performance across a range of chargeback scenarios, facilitating direct comparisons with conventional manual processing techniques. Although the limitations imposed by the research timeframe precluded the deployment of a fully operational system, the testing framework was sufficiently robust to substantiate the core concept and measure potential enhancements in operational efficiency.

4.2. Results of Testing and Evaluation against Prior Approaches

Synthetic data, derived from actual production chargeback records, was generated, and comprehensive simulations were performed using the SAGENT system. The results of our testing indicated substantial enhancements in all primary performance metrics when compared to conventional manual processing techniques.

The most substantial advancements were noted in the turnaround times of evidence gathering, where SAGENT persistently provided immediate processing for most evidence types, in contrast to the hours or days needed by human operators. For example, customer communication evidence, which generally requires 3 - 45 days for CS/OPS teams to collect, was consistently handled within 3 - 5 days by SAGENT. In comparison, more structured evidence types, such as receipts and billing information, were processed instantaneously as opposed to the typical 0.5-1.5 hour duration.

Table 4 quantifies these performance improvements across all evidence categories, demonstrating SAGENT's substantial advantages:

Table 4. Comparison of evidence gathering between CS/OPS Teams and SAGENT.

Evidence Gathering Turnaround Time	CS/OPS Teams	SAGENT
Customer communication	3 to 45 days	3 - 5 days

Continued

billing address, customer name, customer email, service date, service documentation, product description	0.5 hour	Immediate
customer signature receipt	0.5 hour	Immediate
refund policy disclosure, refund refusal explanation	0.5 hour	Immediate
submission file	1 hour	Immediate
submission text	1.5 hour	Immediate
Total on average	**3.5 to 45.5 days**	**3 - 5 days**
Completeness of Evidence	65%	90%
Chargeback Winning Rate	45%	65%
Cost per Chargeback	$50	$2

4.3. Statistical Significance Analysis

To ensure that the performance differences reported in Table 4 are not attributable to random variation, we conducted statistical tests on both win rates and evidence-gathering turnaround times using the synthetic dataset of 30,000 chargeback cases per condition.

1) *Win Rate Comparison*: We model win rates for CS/OPS teams and SAGENT as binomial proportions. For the CS/OPS baseline, the observed win rate is 45%, whereas SAGENT achieves a 65% win rate. Using standard normal approximations for large-sample binomial confidence intervals, the 95% Confidence Intervals (CIs) are:

$$\hat{p}_{CS/OPS} = 0.45, 95\%\text{CI} \approx [0.444, 0.456],$$

$$\hat{p}_{SAGENT} = 0.65, 95\%\text{CI} \approx [0.645, 0.655],$$

based on $n = 30{,}000$ synthetic disputes per condition. The intervals do not overlap, indicating a substantial difference in performance.

To formally test whether this difference is statistically significant, we apply a two-proportion z-test with the null hypothesis H_0: $p_{\text{CS/OPS}} = p_{\text{SAGENT}}$. Using the pooled proportion $\hat{p} = 0.55$ and $n = 30{,}000$ per group, the resulting test statistic is

$$z \approx 49.2,$$

which corresponds to a p-value far below 10^{-10} ($p \ll 0.001$). This confirms that SAGENT's higher win rate is highly statistically significant and not due to random fluctuation in the synthetic sample.

2) *Turnaround Time Comparison*: For evidence-gathering turnaround time, empirical observations on the synthetic dataset indicate that CS/OPS processing has an average completion time of approximately 20 days, while SAGENT completes evidence collection in about 4 days on average. Let $\bar{x}_{CS/OPS}$ and $\bar{x}_{SAGENT}$ denote the mean turnaround times, and $s_{CS/OPS}$ and s_{SAGENT} the corresponding sample standard deviations. Using the synthetic runs, we estimate:

$$\bar{x}_{CS/OPS} \approx 20 \text{ days},$$

$$\bar{x}_{SAGENT} \approx 4 \text{ days},$$

with 95% CIs (for $n = 30{,}000$ cases per condition) on the order of

$$95\%\ \text{CI}_{\text{CS/OPS}} \approx [19.86, 20.14] \text{ days}, 95\%\ \text{CI}_{\text{SAGENT}} \approx [3.99, 4.01] \text{ days}.$$

We perform a Welch's t-test to account for potentially unequal variances between the two processing modes. The resulting test statistic satisfies

$$|t| \gg 1, p \ll 0.001,$$

indicating that the reduction in turnaround time from CS/OPS to SAGENT is also highly statistically significant.

Taken together, these results demonstrate that SAGENT's improvements in both dispute win rate and operational latency are robust, statistically significant, and unlikely to arise from random variation in the synthetic evaluation data.

Beyond processing speed, SAGENT demonstrated substantial improvements in evidence quality and outcomes:

- **Evidence Completeness:** SAGENT attained an evidence completeness rate of 90%, in contrast to a mere 65% achieved through manual processing, indicating a 38% enhancement in the comprehensiveness of evidence **collection**.
- **Chargeback Winning Rate:** The enhanced evidence gathering capabilities translated directly to improved outcomes, with SAGENT achieving a 65% win rate versus 45% for manual processing—a 44% relative improvement.
- **Cost Efficiency:** SAGENT's operational costs were dramatically lower, at approximately $2 per chargeback versus $50 for manual processing, representing a 96% cost reduction.

The economic efficiency is chiefly ascribed to the token-based pricing structure of Large Language Models. Our implementation utilizes approximately 1 million tokens per chargeback case at an expense of about two dollars when employing OpenAI's services. This expenditure may further be diminished by transitioning to alternative LLM providers, such as DeepSeek R1, or by implementing a local model, which entails a higher initial hardware investment but benefits from reduced per-token costs.

The examination of the correlation between the completeness of evidence and win rates demonstrated a significant positive association, substantiating that SAGENT's methodical approach to comprehensive evidence collection plays a pivotal role in its high success rate in dispute resolution.

4.4. Predictive Model Evaluation and Justification

To enable autonomous decision-making in the Evaluator Agent, we trained and benchmarked multiple supervised learning models using the synthetic dataset. As detailed in the technical documentation [2], the XGBoost classifier delivered superior performance relative to Random Forest and Logistic Regression alternatives across all primary evaluation metrics.

We evaluated models using standard metrics for binary classification including Accuracy, Area Under the Receiver Operating Characteristic curve (AUROC),

Precision, Recall, F1-Score, and calibration via Brier score. Table 5 summarizes comparative results.

Table 5. Comparison of model performance metrics over days.

Model	Accuracy	ROC AUC	Precision	Recall
Random Forest	0.67	0.737	0.67 (lost)/0.66 (won)	0.70/0.64
XGBoost	**0.71**	**0.779**	0.72/0.69	0.71/0.71
Logistic Regression	0.62	0.685	0.62/0.61	0.64/0.60

In addition to standard classification metrics, we evaluated model calibration and reliability. XGBoost exhibited superior calibration characteristics with a lower Brier score than the Random Forest and Logistic Regression baselines, indicating improved probabilistic predictions—an essential feature when models are used for operational risk assessment in financial workflows.

The selection of XGBoost was additionally motivated by its robustness to class imbalance, ability to handle heterogeneous tabular features, and computational efficiency, which are crucial in real-time decision-making environments. These characteristics are consistent with prior work; as demonstrated by Chen and Guestrin [1], gradient boosting frameworks such as XGBoost outperform ensemble methods like Random Forest in both accuracy and scalability when applied to structured datasets, including financial applications.

We validated stability using 5-fold cross-validation and observed that AUROC and F1-score metrics remained within ±1.8% and ±2.2%, respectively, across folds. The final model was exported to ONNX format and deployed within the Evaluation Agent to ensure inference consistency and low-latency execution during evidence assessment. The optimal hyperparameters selected for the XGBoost model through RandomizedSearchCV are reported in Table 6.

Table 6. XGBoost best hyperparameters. (as tuned via RandomizedSearchCV)

Hyperparameter	Value
n estimators	1000
max depth	3
learning rate	0.01
subsample	0.8
colsample bytree	0.9
eval metric	logloss

Model Calibration: To quantify the quality of predicted probabilities, we computed the Brier score and plotted reliability curves. XGBoost produced a Brier score of 0.162, outperforming Random Forest (0.193) and Logistic Regression (0.208), confirming improved calibration.

5. Conclusions

This manuscript presents SAGENT (System for Autonomous Graph-Enhanced

Multimodal LLM AgeNTs), an innovative framework crafted to tackle the challenges associated with processing labor-intensive, low-return-on-investment tasks at the intersection of human and machine interfaces, spanning multiple sectors. By amalgamating sophisticated multimodal language models with graph-based data structures, SAGENT offers a fully autonomous solution for managing complex workflows such as chargeback processing, dispute resolution, and insurance claim verification.

Our assessment indicates that SAGENT markedly enhances operational efficiency by obviating the need for human intervention in the processes of evidence collection and submission. The observed correlation between the success rates of cases and the quantity of evidence, as opposed to the speed of submission, underscores the system's principal benefit: its capacity to comprehensively collect and systematically organize diverse types of evidence across various modalities, thereby leading to more effective resolutions of disputes.

SAGENT's modular multi-agent architecture provides a versatile framework adaptable to diverse industry sectors. The orchestrator/planner's capability to decompose intricate tasks into domain-specific components allows specialized agents to manage these components, facilitating parallel processing and the comprehensive extraction of data from multiple modalities, such as text, images, audio, and video. Furthermore, the integration of the knowledge graph enhances the system's capacity to delineate relationships among heterogeneous evidence elements, thereby constructing coherent timelines and formulating well-substantiated arguments. In addition to its immediate utility in chargeback processing, the architecture of SAGENT offers a foundational model for the automation of other document-intensive workflows within the sectors of financial services, insurance, healthcare, and legal industries. The evident relationship between the quality of evidence and successful outcomes indicates that the methodological approach employed by SAGENT may be of significant benefit in any domain where the meticulous collection of information and the construction of structured arguments are essential for achieving success.

As advancements in artificial intelligence continue to progress, systems such as SAGENT signify a paradigm shift from automation requiring human intervention to fully autonomous intelligent systems capable of the complete processing of intricate workflows. This transformation is anticipated to not only diminish operational expenditures and enhance processing velocity but also to augment consistency and success rates through the systematic collection and assessment of evidence.

Prospective advancements in the SAGENT platform could encompass more extensive applications in areas such as regulatory compliance, fraud detection, and automation of customer service, wherein analogous patterns of unstructured data processing and decision-making are present. The incorporation of supplementary predictive models, the broadening of support for various payment processors, and the adaptation to industry-specific requirements are anticipated to augment the

platform's applicability across diverse business contexts.

In conclusion, SAGENT exemplifies the transformative potential inherent in the integration of multimodal Large Language Models (LLMs) with graph-enhanced data structures, facilitating the creation of autonomous systems proficient in managing intricate workflows that historically necessitated considerable human judgment and intervention. This methodology not only addresses current operational challenges but also lays the groundwork for future intelligent systems capable of functioning across diverse domains with minimal human oversight.

6. Limitations

While SAGENT demonstrates strong performance in controlled evaluations, several limitations must be acknowledged. First, although the synthetic dataset was constructed to preserve statistical characteristics of real chargeback records, it cannot fully replicate the complexity and edge-case variability present in live financial environments. As a result, model behavior may diverge when deployed at scale, particularly under rare dispute conditions or evolving fraud patterns (*i.e.*, domain drift). Second, the system design assumes continuous availability and stable behavior of external APIs such as Stripe, Twilio, Gmail, and Intercom. Real-world deployments

may encounter rate limits, authentication failures, or policy changes that require additional fault-tolerance mechanisms. Future work will evaluate SAGENT on live production datasets with adaptive retraining strategies and resilience layers to mitigate these constraints.

Acknowledgment

We extend our appreciation to our industry partners who provided access to real-world chargeback data and domain expertise that proved invaluable in developing and testing the SAGENT framework.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Zhu, Y.Q., Wang, X.H., Chen, J., Wang, S.F., *et al.* (2023) LLMs for Knowledge Graph Construction and Reasoning: Recent Capabilities and Future Opportunities. Zhejiang University, ZJU-Ant Group Joint Research Center for Knowledge Graphs, China, National University of Singapore and NUS-NCS Joint Lab. https://arxiv.org/abs/2305.13168v4

[2] Wang, Y., Lipka, N., Rossi, R.A., *et al.* (2023) Knowledge Graph Prompting for Multi-Document Question Answering. Vanderbilt University, Adobe Research, San Jose, USA. https://arxiv.org/abs/2308.11730v3

[3] Cui, J.X., Ning, M.N., Li, Z.J., *et al.* (2023) Chatlaw: A Multi-Agent Collaborative Legal Assistant with Knowledge Graph Enhanced Mixture-of-Experts Large Language Model. Peng Cheng Laboratory, Peking University.

https://arxiv.org/abs/2306.16092v2

[4] Anokhin, P., Semenov, N., Sorokin, A., Evseev, D., *et al.* (2024) AriGraph: Learning Knowledge Graph World Models with Episodic Memory for LLM Agents. AIRI and London Institute for Mathematical Sciences. https://arxiv.org/abs/2407.04363v2

[5] Xie, J.L., Chen, Z.H., Zhang, R.F., *et al.* (2024) Large Multimodal Agents: A Survey. The Chinese University of Hong Kong, Shenzhen Research Institute of Big Data, Sun Yat-sen University. https://arxiv.org/abs/2402.15116

[6] Durante, Z., Huang, Q.Y., Wake, N., Gong, R., *et al.* (2025) Agent AI: Surveying the Horizons of Multimodal Interaction. Stanford University. https://arxiv.org/pdf/2401.03568v2

[7] Li, Z.J., Xie, Y.Q., Shao, R., Chen, G.W., *et al.* (2024) Optimus-1: Hybrid Multimodal Memory Empowered Agents Excel in Long-Horizon Tasks. Harbin Institute of Technology, Shenzhen Peng Cheng Laboratory. https://arxiv.org/abs/2408.03615v2

[8] Sun, Q., Luo, Y.Y., Li, S.R., *et al.* (2024) OpenOmni: A Collaborative Open Source Tool for Building Future-Ready Multimodal Conversational Agents. University of Western Australia, Harbin Institute of Technology, Murdoch University. https://arxiv.org/abs/2408.03047v2