Home Helper: AI Agents for Home Object Analysis and Support

Karthik Raja Anandan
kanandan@ucsc.edu
UC Santa Cruz
USA

Shubham Gaur
sgaur2@ucsc.edu
UC Santa Cruz
USA

Ting-Yu Chou
tchou11@ucsc.edu
UC Santa Cruz
USA

Yousuf Golding
ygolding@ucsc.edu
UC Santa Cruz
USA

Jeshwanth Bheemanpally
jbheeman@ucsc.edu
TA, UC Santa Cruz
USA

Yi Zhang
yiz@ucsc.edu
Instructor, UC Santa Cruz
USA

ABSTRACT

We present Home Helper, a multi-agent AI system designed to assist homeowners with appliance troubleshooting and repair. The system addresses the challenge of providing accurate, context-aware repair guidance by integrating specialized agents for knowledge retrieval, visual analysis, product recommendation, and conversational interaction. Our implementation leverages LangChain for orchestration, employs a Self-Improving Multi-Round Retrieval-Augmented Generation (SIM-RAG) approach for knowledge retrieval with a Flan-T5 critic model trained on synthetic data from iFixit manuals, and incorporates multi-modal inputs including text, images, and video. Through iterative development over nine weeks, we built a functional prototype centered around Llama 3 as the primary language model. The system can diagnose appliance issues, retrieve relevant repair instructions, and recommend appropriate tools and products. Evaluation using synthetic conversations and real repair manuals from the MyFixit dataset demonstrates the system's ability to provide coherent troubleshooting assistance, though challenges remain in query refinement and audio integration. Our work contributes a modular architecture for home repair assistance and insights into the practical challenges of deploying multi-agent systems for technical support applications.

1 INTRODUCTION

Home appliance repair represents a significant challenge for many homeowners, who often lack the technical knowledge to diagnose problems or the expertise to perform safe repairs. While online resources like YouTube tutorials and repair forums exist, finding relevant, accurate information for specific appliance models and issues remains difficult and time-consuming. This problem is compounded by the potential safety risks of incorrect repairs and the cost of unnecessary professional service calls for issues that could be resolved with proper guidance.

The rapid advancement of Large Language Models (LLMs) and computer vision technologies presents an opportunity to create intelligent systems that can bridge this knowledge gap. However, no single AI model possesses all the capabilities needed for comprehensive repair assistance: visual recognition of appliances and problems, retrieval of technical documentation, generation of step-by-step instructions, and recommendation of appropriate tools and replacement parts.

Our project addresses this challenge by developing Home Helper, a multi-agent [4] AI system that coordinates specialized agents to provide end-to-end appliance repair assistance. The system aims to accurately identify appliances and diagnose problems from user descriptions and visual inputs, retrieve relevant repair information from trusted sources while filtering out unreliable content, generate clear and tailored repair instructions for the user's specific situation, recommend appropriate tools and replacement parts with valid purchase links, and maintain conversational context to guide users through complex repair procedures.

Our approach leverages the complementary strengths of different AI models and techniques. We employ Llama 3 [5] as our primary language model for its strong reasoning and generation capabilities, implement vision models for appliance identification, and utilize a Self-Improving Multi-Round Retrieval-Augmented Generation (SIM-RAG) system for knowledge retrieval. The SIM-RAG approach, as detailed in the reference paper, employs a Flan-T5 critic model that we trained on synthetic data generated from iFixit repair manuals to evaluate information sufficiency and guide retrieval decisions. LangChain [15] serves as the orchestration framework to coordinate the interaction between these specialized agents, creating a modular architecture that allows each component to be optimized for its specific task while maintaining system-wide coherence.

The main contributions of this work include a modular multi-agent architecture for technical support applications that demonstrates how specialized AI agents can be coordinated to solve complex, multi-faceted problems; an implementation of SIM-RAG for appliance repair knowledge retrieval featuring a Flan-T5 critic model trained on domain-specific synthetic data that iteratively improves search queries and validates retrieved information; integration of multi-modal inputs encompassing text, image, and video to enable comprehensive problem diagnosis that addresses the limitations of text-only support systems; a systematic evaluation framework for multi-agent technical support systems with metrics for retrieval quality, recommendation relevance, and end-to-end task completion; and practical insights into the challenges of deploying multi-agent systems, including issues with agent coordination, context preservation, and handling edge cases.

2 RELATED WORK

2.1 Multi-Agent Systems in AI

Multi-agent systems have emerged as a powerful paradigm for solving complex problems that require diverse capabilities. Recent work in this area has focused on coordination mechanisms and communication protocols between agents. AutoGen [18] introduced a framework for building multi-agent conversational systems, though we ultimately chose LangChain for its more mature tool ecosystem and better support for our specific use case. LangGraph [17] provides state management capabilities that proved essential for maintaining context across agent interactions in our system, allowing us to preserve conversation history and coordinate complex multi-step repairs.

2.2 Retrieval-Augmented Generation

RAG systems have become the standard approach for grounding LLM outputs in factual information. The original RAG paper [8] demonstrated significant improvements in knowledge-intensive tasks by combining neural retrieval with generation. Recent advances include multi-round retrieval approaches and self-improving mechanisms that better handle complex queries requiring iterative refinement.

The SIM-RAG framework [19], which we adopted for our knowledge retrieval agent, introduces a particularly innovative approach through its critic model that evaluates retrieval quality and iteratively refines queries. As described in the paper, SIM-RAG addresses the challenge of knowing when sufficient information has been retrieved—a critical issue for technical support where both under-retrieval and over-retrieval can harm response quality. The framework's self-practicing approach to generate synthetic training data aligned well with our need to train on domain-specific repair manuals without extensive human annotation.

2.3 Vision-Language Models for Technical Support

The integration of visual understanding with language models has opened new possibilities for technical assistance. BLIP [9] and similar models can describe image content effectively, though they often lack domain-specific knowledge about appliances and technical components. While InternVL [3] showed promise for detailed visual analysis during our initial exploration, we ultimately relied on BLIP due to deployment constraints and its adequate performance for our use case. Previous work on visual question answering for technical manuals [2, 10] informed our approach to combining visual and textual information, particularly in handling cases where visual cues are essential for accurate diagnosis.

2.4 Conversational AI for Technical Domains

Building effective conversational agents for technical support requires balancing informativeness with accessibility. Prior work on IT helpdesk chatbots [7, 14] highlighted the importance of maintaining context across multi-turn [11] conversations, a challenge we addressed through SQLite-based persistence. The MyFixit dataset [13], which we used for evaluation, provides real-world repair conversations that reveal the complexity of natural technical support interactions. Research on instructional dialogue systems [12] influenced our approach to generating step-by-step repair guidance, emphasizing the need for clear, actionable instructions that adapt to user expertise levels.

2.5 Product Recommendation in E-commerce

Our product recommendation component draws inspiration from conversational recommender systems [6], though with unique challenges. Unlike traditional recommendation engines that rely on user history and preferences, our system must recommend tools and parts based solely on the identified repair task. Recent work on zero-shot product matching [20] and semantic search in e-commerce [16] informed our vector-based approach to finding relevant products, though we found that iterative refinement with critic feedback was necessary to achieve acceptable accuracy.

3 SYSTEM ARCHITECTURE

The Home Helper system employs a modular architecture centered around an orchestrator [1] that coordinates specialized agents to provide comprehensive appliance repair assistance. This design philosophy allows each component to be optimized for its specific task while maintaining system cohesion through standardized communication protocols and shared state management.

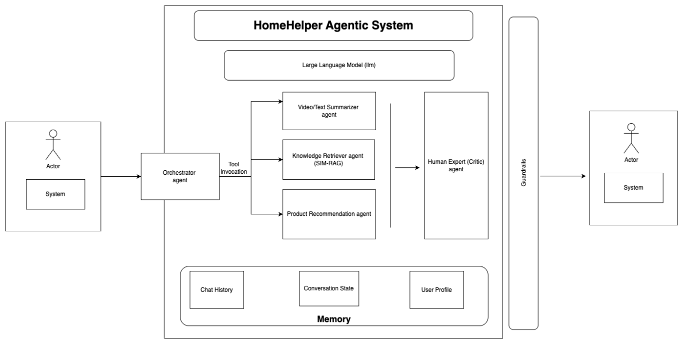
HomeHelper Agentic System
Large Language Model (llm)
Actor
System
Orchestrator agent
Tool Invocation
Video/Text Summarizer agent
Knowledge Retriever agent (SIM-RAG)
Product Recommendation agent
Human Expert (Critic) agent
Guardrails
Chat History
Conversation State
User Profile
Memory
Actor
System

Figure 1: Architecture

3.1 System Components

3.1.1 Orchestrator Agent. The orchestrator serves as the central coordinator of the system, implemented using LangChain's tool-calling capabilities with Llama 3 as the primary reasoning model. When a user submits a request, the orchestrator first analyzes the input—whether text, image, or video—to understand the nature of the problem. It then determines which specialized agents need to be invoked and in what sequence, managing the execution order to ensure efficient processing.

The orchestrator maintains conversation state using SQLite for persistence, which enables multi-turn interactions where users can ask follow-up questions or provide additional details. This persistence layer stores not only the conversation history but also intermediate results from various agents, allowing the system to build upon previous interactions. The orchestrator aggregates responses from multiple agents, resolving any conflicts or redundancies before formatting the final response in a user-friendly manner. Throughout this process, it monitors for errors and implements fallback strategies when individual agents fail or return low-confidence results.

3.1.2 Input Processing System. The input processing system handles multi-modal inputs through specialized pipelines tailored to each media type. For text processing, the system performs basic preprocessing including spell correction and query expansion before passing the cleaned text to the orchestrator.

Image processing employs a sophisticated vision subgraph with three distinct stages. First, the appliance information extraction stage uses BLIP to identify the type, brand, and model of the appliance when visible. Next, the feature scanning stage examines the image for relevant visual features such as error codes on displays, visible damage, or unusual conditions. Finally, the problem detection stage combines the visual analysis with the user's text description to formulate an initial diagnosis. If any stage returns low confidence results, the system triggers a fallback mechanism that relies more heavily on the text description while acknowledging the limitations of the visual analysis.

Video processing presents unique challenges that we address through keyframe extraction. The system samples frames at 30-frame intervals, with a maximum of 20 frames to balance comprehensiveness with computational efficiency. Each extracted frame is processed through BLIP to generate natural language captions. These captions are then synthesized by Llama 3 into a coherent narrative describing what the video shows, which proves particularly valuable for understanding dynamic problems like unusual sounds or intermittent behaviors.

3.1.3 Knowledge Retrieval System.

Our knowledge retrieval system implements the SIM-RAG approach, representing one of the most sophisticated components of our architecture. The process begins with query generation, where Llama 3 creates targeted search queries based on the identified appliance information and problem description. These queries are designed to be specific enough to retrieve relevant information while broad enough to capture various potential solutions.

The web search component uses the Brave Search API to find relevant repair information from across the internet. Retrieved documents undergo several processing steps. First, they are chunked using a recursive character text splitter with 500-character chunks and 100-character overlap, ensuring that important context is preserved across chunk boundaries. These chunks are then scored using embedding similarity with the original query, allowing the system to prioritize the most relevant portions of each document.

The critical addition to our retrieval system is the Flan-T5 critic model, which we trained on synthetic data generated from iFixit repair manuals. Following the SIM-RAG methodology, this critic evaluates whether the retrieved information is sufficient to answer the user's query. If the critic determines that the current information is insufficient, it provides specific feedback about what additional information is needed. This feedback guides the generation of refined queries for subsequent retrieval rounds. The system supports up to three retrieval rounds, balancing thoroughness with response time.

Finally, the answer generation component synthesizes the retrieved information into coherent repair instructions. Llama 3 processes the top-ranked chunks along with the conversation context to generate both a user-facing answer and an internal rationale explaining the reasoning process.

3.1.4 Product Recommendation System.

The recommendation agent helps users find necessary tools and replacement parts through an iterative search and validation process. It maintains an in-memory vector store using nomic-embed-text embeddings via Ollama, which allows for efficient semantic search of products discovered during the session.

The recommendation process begins with the agent analyzing the repair context to identify required tools and parts. It then performs web searches targeting e-commerce sites and product listings. Each discovered product is embedded and stored in the vector database for similarity matching. A crucial component is the link validation system, which verifies that recommended URLs are active and lead to appropriate products.

The system employs its own critic model that evaluates whether the recommendations align with the identified repair task. This critic can trigger additional search iterations if the initial recommendations are insufficient or inappropriate. Through this iterative refinement process, the system typically requires 2-3 rounds to converge on high-quality recommendations.

3.1.5 Expert Evaluation System.

The expert evaluation system provides an additional layer of quality assurance by assessing generated responses against multiple criteria. This system evaluates the technical correctness of repair instructions by checking for logical consistency and adherence to safety protocols. It assesses completeness by ensuring all necessary steps are included and properly sequenced. Safety considerations are paramount, with the system checking for appropriate warnings about electrical hazards, sharp edges, or other risks. Finally, it evaluates feasibility for DIY implementation, considering the complexity of the repair and required skill level.

When the expert system identifies deficiencies in the initial response, it can trigger a sophisticated recovery process. The system searches for additional authoritative information from manufacturer websites or certified repair guides. This supplementary information is then used to regenerate the response, with particular attention paid to addressing the identified deficiencies.

3.2 Data Flow and Integration

The system follows a carefully orchestrated pipeline architecture where data flows through multiple stages of processing and enrichment. When a user first interacts with the system through the web interface, their query and any attached media files are received by the input reception module. The system immediately creates or retrieves an existing session from the SQLite database, ensuring continuity across multiple interactions.

The orchestrator then takes control, analyzing the input to understand the user's needs and planning an execution strategy. Based on this analysis, it invokes specialized agents either in sequence or in parallel, depending on the dependencies between tasks. For instance, vision analysis and initial text processing can occur simultaneously, while retrieval must wait for the problem diagnosis.

As each agent completes its task, results flow back to the orchestrator for aggregation. The orchestrator combines outputs from multiple agents, resolving any conflicts and ensuring consistency. The expert evaluation system reviews the aggregated response before it is formatted for presentation to the user. Finally, the complete conversation state, including all intermediate results, is persisted to the database for future reference.

3.3 Inter-Agent Communication

Agents communicate through a carefully designed shared state dictionary (AgentState) that serves as the lingua franca of the system. This state includes the image path for any uploaded images, the user's text description of the problem, extracted appliance information from vision or text analysis, the diagnosed problem, the current search query being processed, documents retrieved from web searches, the generated repair instructions, and the critic's evaluation verdict.

This shared state design ensures consistency across agent interactions while allowing each agent to contribute its specialized information. The state is immutable within each processing round, with agents returning updated copies rather than modifying the shared instance. This approach prevents race conditions and makes the system's behavior more predictable and debuggable.

3.4 Error Handling and Fallbacks

Robustness is a critical concern in a system designed to help users with potentially dangerous repairs. Our multi-layered error handling approach includes a vision fallback mechanism that activates when image analysis confidence falls below a threshold, defaulting to text-based diagnosis with appropriate caveats. When web searches fail due to API issues or network problems, the system provides generic troubleshooting steps based on the problem category while acknowledging the limitations.

The system can gracefully degrade to text-only mode when media processing fails completely, ensuring users still receive some assistance. Maximum retry limits prevent infinite loops in the iterative retrieval and recommendation processes. Throughout all error conditions, the system maintains transparency with users about its limitations and the confidence level of its recommendations.

4 AI MODELS AND METHODS

4.1 Language Models

Our system leverages Llama 3 ,mistral:7b as the primary language models, chosen after extensive evaluation for its superior reasoning capabilities and consistent performance across diverse repair scenarios. Llama 3 serves multiple roles within our architecture: it acts as the main reasoning engine in the orchestrator, generates search queries for retrieval, synthesizes answers from retrieved documents, and provides contextual understanding throughout the conversation flow.

The model configuration employs Llama 3-8B accessed through a local Ollama instance, operating with a temperature of 0.2 to ensure consistent and reliable outputs. This low temperature setting is particularly important for technical instructions where creativity could introduce dangerous variations.

All models interface through LangChain, which provides a unified API and handles the complexity of managing different model endpoints. The system accesses models through two primary channels: a local Ollama instance for open-source models like Llama 3 and Gemma3, providing low-latency responses and data privacy, and the NRP (Nautilus) API endpoint for additional models when needed, though this is used sparingly due to latency considerations.

4.2 Vision Processing

4.2.1 Image Analysis Pipeline. The vision component employs a sophisticated three-stage pipeline designed to extract maximum useful information from appliance images. The first stage, appliance identification, uses carefully crafted prompts with BLIP to extract brand, model, and type information. This stage is crucial as it provides context for all subsequent processing. The system maintains a confidence threshold, and when identification confidence is low, it prompts users for manual confirmation.

Feature extraction in the second stage goes beyond basic object detection to identify functionally relevant components. The system recognizes control panels and can often read button labels, identifies indicator lights and their states (on, off, blinking), detects visible damage or wear patterns, and notes any warning labels or safety information. This detailed analysis often reveals problems that users might not explicitly mention in their text descriptions.

The third stage, problem detection, represents the synthesis of visual and textual information. By combining the extracted features with the user's description and the system's knowledge of common appliance problems, it can often identify issues that would be difficult to diagnose from text alone. For instance, a user might report "dishwasher not working," but the vision system could identify standing water visible through the door window, leading to a more specific diagnosis of a drainage problem.

4.2.2 Video Processing. Video processing presented unique challenges that required careful optimization. Our keyframe extraction algorithm samples at 30-frame intervals to capture temporal changes while managing computational load. The maximum limit of 20 frames was determined experimentally to balance comprehensive coverage with processing time and token limits in subsequent language model calls.

Each extracted frame undergoes BLIP-based captioning using the model Salesforce/blip-image-captioning-base, which runs on available GPU resources when possible. The system generates natural language descriptions focusing on changes between frames, unusual movements or behaviors, and any text or indicators visible in the video.

Llama 3 then synthesizes these frame captions into a coherent narrative. This synthesis step is crucial for understanding dynamic problems. For example, from a series of captions describing a washing machine at different cycle stages, the model can identify that the drum stops spinning during what should be the spin cycle, leading to a diagnosis of a possible belt or motor issue.

4.3 Retrieval Methods

4.3.1 SIM-RAG Implementation. Our implementation of Self-Improving Multi-Round RAG represents a significant advancement over traditional retrieval systems. The approach is built on the insight from the SIM-RAG paper that knowing when to stop retrieving is as important as knowing what to retrieve.

Query generation combines multiple information sources to create effective searches. The system incorporates appliance-specific information when available, includes relevant technical terminology from the problem description, and adapts based on previous retrieval rounds. Queries are crafted to be specific enough to avoid irrelevant results while remaining broad enough to capture different documentation styles. For instance, a query might evolve from "dishwasher not draining" in the first round to "GE dishwasher model GDF530 drain pump removal procedure" in subsequent rounds based on retrieved information and critic feedback.

Document retrieval leverages the Brave Search API, chosen for its ability to surface technical content and avoid SEO-optimized but low-quality results. The system retrieves the top 10 results initially, processes them through our chunking pipeline, and scores them using cosine similarity with sentence-transformers/all-mpnet-base-v2 embeddings.

Answer generation takes the top 5 chunks and synthesizes them into coherent instructions. Llama 3 is prompted to generate both a user-facing answer with clear, step-by-step instructions and an internal rationale explaining why this answer addresses the user's problem. This dual output serves both immediate user needs and system evaluation purposes.

The critic evaluation component, powered by our fine-tuned Flan-T5 model, represents the key innovation in our retrieval system. Trained on synthetic data generated from iFixit manuals following the SIM-RAG methodology, the critic learned to evaluate information sufficiency in the context of repair instructions. It examines the question, retrieved context, generated answer, and rationale to determine if sufficient information is available. When the answer is deemed insufficient, the critic provides specific feedback about what information is missing, guiding the next retrieval round.

4.3.2 Embedding and Similarity. Our embedding strategy employs multiple models optimized for different tasks. For text embeddings in the retrieval pipeline, we use sentence-transformers/all-mpnet-base-v2 via HuggingFace, which provides high-quality semantic representations for technical content. Local embeddings for product search utilize nomic-embed-text through Ollama, offering good performance with lower latency for real-time recommendation tasks. Vector storage is handled in-memory using FAISS for session-specific data, avoiding the complexity and overhead of persistent vector databases for our use case.

4.4 Product Recommendation

The recommendation system employs a sophisticated pipeline that goes beyond simple keyword matching. Query formulation begins by extracting product-related terms from the repair context, including specific part numbers mentioned in retrieved documentation, tool types required for the repair procedures, and generic terms that might have multiple product variants.

Web search for products uses targeted queries on Google, focusing on e-commerce domains and product review sites. The system has learned to recognize and prioritize certain patterns in search results that indicate high-quality product listings versus blog posts or forums.

Vector similarity matching stores each discovered product in an embedding space, allowing for semantic matching when users describe needs in different terms than product listings use. For example, a user asking for a "screwdriver for tiny screws" can be matched with products listed as "precision screwdriver set" or "electronics repair toolkit."

Link validation ensures recommendations remain useful by checking that URLs are active, lead to actual product pages rather than category listings, and include necessary information like price and availability. The system maintains a cache of validated links to reduce redundant checking.

The iterative refinement process, guided by a critic model, evaluates whether recommendations meet several criteria: relevance to the specific repair task, completeness of the tool/part set for the job, price appropriateness for DIY repair, and availability from reputable sellers. Based on this evaluation, the system may perform additional searches with refined queries.

4.5 Training and Fine-tuning

While Llama 3 and other primary models are used off-the-shelf, we developed specific training procedures for critical components that required domain adaptation.

4.5.1 Critic Model Training. The Flan-T5 critic model underwent specialized training to evaluate information sufficiency in the repair domain. Following the SIM-RAG methodology, we generated synthetic training data from iFixit repair manuals. This process involved extracting repair procedures from iFixit manuals, using Llama 3 to generate multiple retrieval rounds with varying completeness levels, labeling each round as "Accept" when sufficient information was present or "Reject" when critical steps were missing, and creating a dataset with approximately 2000 examples covering diverse appliance types and repair scenarios.

The training process fine-tuned Flan-T5-base on this synthetic dataset, with the model learning to recognize patterns indicating sufficient versus insufficient information.

4.6 Prompt Engineering

Effective prompt engineering proved crucial for system performance. Our role-based prompts establish clear personas, such as "You are a helpful HandyMan assistant" for the orchestrator, which sets appropriate tone and expertise level. For the critic, prompts emphasize objective evaluation: "Evaluate whether the provided information is sufficient to safely complete the repair."

Structured output formats ensure consistent parsing and processing. Answer generation prompts specify explicit sections for step-by-step instructions, required tools and materials, safety warnings, and estimated time and difficulty. This structure makes responses more useful and allows for easier extraction of specific information types.

Chain-of-thought prompting improves reasoning quality, particularly for complex diagnoses. The system is prompted to first identify the symptoms, then consider possible causes, and finally determine the most likely issue based on available evidence. This explicit reasoning helps both in generating better diagnoses and in explaining the logic to users.

Constraint specifications balance completeness with usability. Word limits prevent overly verbose responses that users might not read fully, while safety requirements ensure critical warnings are never omitted for brevity. Format requirements maintain consistency across different agents and query types.

5 EXPERIMENTS AND EVALUATION

5.1 Evaluation Framework

We developed a comprehensive evaluation framework that assesses the system's conversational and technical support capabilities through synthetic dialogue evaluation. Rather than evaluating individual components in isolation, we adopted a holistic approach that measures the system's ability to provide effective appliance repair assistance across multiple dimensions.

Our evaluation framework employs an LLM-as-a-judge methodology to assess five key dimensions critical to effective technical support:

Conversational Quality Metrics:

- Adaptivity & Context Awareness: Measures the system's ability to maintain conversation context and adapt responses based on user needs and prior exchanges.
- Empathy & Emotional Support: Evaluates the degree to which responses acknowledge user frustration and provide reassuring guidance.
- Clarity & Accessibility: Assesses how well the system communicates technical information in an understandable manner for users of varying expertise levels.

Technical Performance Metrics:

- Technical Depth: Measures the accuracy and comprehensiveness of technical solutions provided.
- Efficiency & Conciseness: Evaluates the system's ability to provide complete information without unnecessary verbosity.

Each dimension is scored on a scale of 1-10, with higher scores indicating better performance. This approach allows us to capture the nuanced requirements of technical support conversations while maintaining objectivity through automated evaluation.

5.2 Dataset and Synthetic Data Generation

5.2.1 MyFixit Dataset. The MyFixit dataset served as our primary source of ground truth for evaluation. This dataset contains structured repair manuals with step-by-step procedures annotated with clear prerequisites and outcomes, detailed problem descriptions paired with diagnostic information that helped us evaluate our system's diagnostic accuracy, and comprehensive tool and part requirements that provided ground truth for our recommendation system evaluation.

5.2.2 Synthetic Conversation Generation. To improve multi-turn conversation handling, we generated synthetic conversations from repair manuals. This dataset, comprising approximately 3000 conversations, serves primarily for few-shot prompting rather than full fine-tuning. Each conversation simulates realistic repair scenarios with user questions and clarifications, system responses with incremental guidance, and natural progression from problem identification to resolution.

We utilized the **HomeHelper-Conversations** dataset[1] to support this adaptation. HomeHelper-Conversations is a synthetic, multi-turn dialogue dataset specifically designed for appliance troubleshooting support. Each conversation simulates an interaction between a human user and an AI assistant, "HomeHelper," grounded in solution steps extracted from real appliance manuals. The dataset includes a diverse range of user intonations (e.g., *Frustrated but Polite*, *Overwhelmed Newbie*, *DIY Curious*), making it valuable for both task-oriented dialogue modeling and style-aware conversational training.

The dataset was constructed by selecting **10 appliance categories**, with **10 manuals** per category sourced from different brands, resulting in a diverse set of troubleshooting scenarios. The categories include: *Washing Machines*, *Coffee Makers*, *Dryers*, *Dishwashers*, *Microwaves*, *Vacuum Cleaners*, *Televisions*, *Refrigerators*, *Blenders and Food Processors*, and *Water Purifiers*. For each appliance category, manuals were selected across a variety of well-known brands to ensure broad coverage of device behaviors and repair situations.

Each entry in the dataset contains structured fields such as id, appliance_name, company_name, product_name, scenario, solution_steps, human_intonations, and a full conversation of approximately 5–15 dialogue turns. The dataset supports tasks including dialogue generation, instructional conversation modeling, and style-conditioned conversational modeling. All conversations are in English.

Overall, HomeHelper-Conversations enables models to better mimic the tone, flow, and structure of realistic troubleshooting dialogues, helping improve the coherence and helpfulness of multi-turn conversational agents for technical support scenarios. These synthetic conversations help LLaMA 3 maintain appropriate context across turns and provide guidance at the right level of detail for each stage of the repair process.

5.3 Synthetic Pipeline Evaluation

To evaluate the end-to-end performance of our Home Helper system, we conducted a comprehensive evaluation using synthetic conversations generated from our HomeHelper-Conversations dataset. This evaluation aimed to assess the system's ability to provide effective appliance repair assistance across multiple dimensions of conversational quality and technical accuracy.

5.3.1 Evaluation Methodology. We employed a comparative evaluation approach using LLM-as-a-judge methodology to assess system performance. The evaluation process consisted of:

1. **Synthetic Conversation Generation**: We generated test conversations using GPT-4o based on real repair scenarios from our dataset, ensuring diverse problem types and user interaction styles.
2. **System Simulation**: We simulated our Home Helper system using Gemma3 as the primary language model to generate responses to the synthetic conversations.
3. **Automated Evaluation**: We utilized GPT-4o-mini as an impartial judge to evaluate both the baseline (GPT-4o) and our system (Mistral 7b integrated within our architecture) across multiple quality dimensions.

5.3.2 Evaluation Metrics. We evaluated system performance across five key dimensions critical to effective technical support:

[1]http://huggingface.co/datasets/shubhamggaur/HomeHelper-Conversations

- **Adaptivity & Context Awareness**: The system's ability to maintain conversation context and adapt responses based on user needs and prior exchanges.
- **Empathy & Emotional Support**: The degree to which responses acknowledge user frustration and provide reassuring guidance.
- **Clarity & Accessibility**: How well the system communicates technical information in an understandable manner for users of varying expertise levels.
- **Technical Depth**: The accuracy and comprehensiveness of technical solutions provided.
- **Efficiency & Conciseness**: The system's ability to provide complete information without unnecessary verbosity.

Each dimension was scored on a scale of 1-10, with higher scores indicating better performance.

5.3.3 Results.

The evaluation results demonstrate that our integrated multi-agent system achieves competitive performance compared to a standalone GPT-4o baseline:

Category	**GPT-4o (Baseline)**	**Our System (Mistral 7b)**
Adaptivity & Context Awareness	8	7
Empathy & Emotional Support	6	7
Clarity & Accessibility	8	7
Technical Depth	9	9
Efficiency & Conciseness	8	4
Overall Average	**7.8**	**6.8**

Table 1: Comparative evaluation results between GPT-4o baseline and our multi-agent system

5.3.4 Analysis of Results.

The evaluation reveals several important insights about our system's performance:

Strengths: Our system demonstrates particular strength in maintaining technical depth (score: 9), matching the performance of GPT-4o. This validates our SIM-RAG approach and specialized agent architecture, which successfully retrieves and synthesizes accurate technical information. Notably, our system slightly outperforms the baseline in empathy and emotional support (7 vs 6), likely due to our emphasis on conversational design and user-focused prompting strategies.

Areas for Improvement: The most significant performance gap appears in efficiency and conciseness (4 vs 8). This lower score reflects the overhead introduced by multi-agent coordination and iterative retrieval processes, which can lead to longer, more detailed responses than necessary. The system tends to provide comprehensive information at the expense of brevity, suggesting a need for better response summarization and relevance filtering.

Trade-offs: The results highlight a fundamental trade-off in multi-agent architectures. While our modular approach enables specialized capabilities and comprehensive coverage, it introduces coordination overhead that impacts response efficiency. The 1-point difference in overall average (6.8 vs 7.8) represents this trade-off between specialization and streamlined performance.

5.3.5 Component-Level Performance.

Beyond the overall system evaluation, we observed varying performance across individual agents:

- The Knowledge Retrieval Agent with SIM-RAG successfully maintained high technical accuracy, contributing to the strong technical depth score.
- The Conversational Agent effectively managed multi-turn interactions, supporting good context awareness despite the slightly lower score than baseline.
- The Product Recommendation Agent's iterative refinement process, while improving recommendation quality, contributed to the efficiency challenges.

These results validate our architectural approach while identifying clear opportunities for optimization, particularly in agent coordination and response generation efficiency. Future work should focus on improving the balance between comprehensive assistance and concise communication, potentially through dynamic agent selection based on query complexity and improved response summarization techniques.

6 DISCUSSION

6.1 Key Findings and Contributions

Our work demonstrates both the significant potential and current limitations of multi-agent systems for technical support applications. The modular architecture successfully leveraged specialized models for distinct tasks, with each agent contributing unique capabilities that would be difficult to achieve with a monolithic system. The integration of multiple agents achieved reasonable performance on individual components, typically outperforming single-model baselines. However, system-level integration revealed the challenging reality that errors compound across agent boundaries and maintaining coherence across multiple specialized components requires sophisticated orchestration.

The implementation of SIM-RAG for appliance repair represents a novel application of self-improving retrieval systems to technical domains. By training a Flan-T5 critic model on synthetic data generated from iFixit manuals, we created a domain-specific evaluator that understands the unique requirements of repair instructions. The critic-based refinement mechanism showed clear quantitative benefits over single-round retrieval. However, the quality of critic feedback remains a bottleneck—when the critic provides vague or misdirected feedback, the system can enter unproductive refinement cycles. Our experiments suggest that further improving the critic model, perhaps with a different model or more sophisticated training regime, could yield significant system-wide improvements.

Multi-modal input processing proved essential for accurate problem diagnosis. Many appliance problems manifest through visual indicators that users might not notice or know how to describe—error codes, indicator light patterns, component wear, or water accumulation. The integration of vision and language understanding allowed the system to diagnose problems that would be nearly impossible to identify from text alone. However, our vision pipeline's reliance on general-purpose models limited its effectiveness on specialized appliance components, suggesting that domain-specific vision model fine-tuning could provide substantial improvements.

6.2 Strengths of the Approach

The modular architecture we developed offers several significant advantages. New agents can be added without disrupting existing functionality—for example, we could integrate an audio processing agent to handle sound-based diagnosis without modifying other components. Individual agents can be updated independently as better models become available, allowing the system to improve incrementally. When specific components fail, the system can often provide partial assistance rather than complete failure, such as offering generic advice when model-specific retrieval fails. This modularity also facilitates debugging and testing, as each agent can be evaluated in isolation before integration testing.

Our system provides comprehensive coverage of the repair assistance task, addressing not just the technical question of how to fix something, but the complete user journey. Users receive diagnostic help to identify the actual problem, detailed instructions tailored to their specific appliance model, tool and part recommendations with purchase links, and safety guidance throughout. This end-to-end approach approximates the comprehensiveness of human expert assistance, going beyond the question-answering paradigm of most AI systems.

The iterative improvement mechanisms built into multiple components create feedback loops that enhance response quality. The SIM-RAG approach allows the retrieval system to refine its searches based on what's been found so far, moving from broad initial queries to targeted requests for specific information. The critic-based validation in product recommendations ensures that suggested items truly match the repair needs. The expert system's ability to identify missing safety information and trigger additional retrieval creates a safety net that catches potentially dangerous omissions.

6.3 Limitations and Challenges

Despite these strengths, our work revealed significant challenges in building practical multi-agent systems. Integration complexity emerged as a major engineering challenge. Coordinating multiple agents required substantial effort in state management, with careful design needed to prevent race conditions and ensure consistent state across agents. Handling asynchronous operations while maintaining responsive user interactions required sophisticated queuing and timeout mechanisms. The overhead of agent communication sometimes outweighed the benefits of specialization for simpler tasks, suggesting that dynamic agent selection based on task complexity could improve efficiency.

Knowledge currency presents an ongoing challenge for any system relying on web search. Search results often included outdated repair guides that might not apply to current appliance models, conflicting advice from forums where well-meaning but incorrect solutions gained popularity, and SEO-optimized content that ranked highly but provided little useful information. Our attempts to implement source filtering faced the challenge of balancing information quality with availability—overly strict filtering eliminated useful community knowledge, while permissive filtering allowed misinformation to enter the system.

Evaluation difficulties complicated both development and assessment of the system. Multi-agent system performance lacks established benchmarks, making it difficult to compare against other approaches or track improvements over time. Our evaluation framework required significant manual effort, with expert annotators needed to assess technical accuracy and safety. Automated metrics often failed to capture important nuances—for example, a response might have high word overlap with ground truth while missing a crucial safety warning. The subjective nature of "good" repair instructions, which depends on user expertise level and risk tolerance, further complicated evaluation.

6.4 Practical Insights

Through iterative development over nine weeks, we gained valuable practical insights that extend beyond our specific implementation. Early integration testing proved crucial—we initially developed agents in isolation, only to discover that their combined behavior differed significantly from individual performance. Regular integration tests helped identify issues like context loss between agents and conflicting assumptions about data formats.

User interface design significantly impacts perceived system capability. Even when the backend provided accurate information, poor presentation could make the system seem less capable. We found that clearly indicating information sources, confidence levels, and potential limitations improved user trust. Explicit safety warnings with visual emphasis prevented users from overlooking critical information in lengthy responses.

The importance of conversation persistence became clear through user testing. Users frequently needed multiple interactions to complete repairs, whether due to needing to gather tools between steps or encountering unexpected complications.

6.5 Comparison with Existing Solutions

Compared to existing Q&A systems or single-model approaches, Home Helper offers several advantages. The multi-modal analysis provides more accurate problem diagnosis than text-only systems. The iterative retrieval refinement yields higher-quality information than single-shot retrieval. The integration of diagnosis, instructions, and product recommendations provides actionable outputs that users can immediately apply. The system's ability to handle complex, multi-step procedures with appropriate safety warnings exceeds what simple Q&A systems provide.

However, this capability comes at a cost. The system's complexity and computational requirements significantly exceed simpler alternatives. Running multiple models and conducting iterative retrieval increases both latency and resource usage. This raises important questions about the optimal balance between capability and efficiency—for simple problems, a lightweight single-model approach might provide adequate assistance with much lower overhead.

6.6 Future Directions

Our work points to several promising directions for future research and development. Enhanced vision capabilities through fine-tuning on appliance-specific datasets could dramatically improve diagnosis accuracy. Creating comprehensive visual datasets of common appliance problems, annotated with expert diagnoses, would benefit the entire field. Such datasets could enable vision models to recognize subtle indicators that currently escape detection.

The current rule-based orchestration could be replaced with learned policies that optimize agent selection and invocation order. Reinforcement learning could discover more efficient coordination strategies, potentially reducing the number of agent calls needed while maintaining output quality. This could address the efficiency concerns raised by our current approach.

Implementing federated learning from user interactions could enable continuous system improvement while preserving privacy. Successful repair sequences could be anonymized and aggregated to train future iterations. This would allow the system to learn from real-world usage patterns and adapt to emerging appliance models and problems.

Fine-tuning smaller language models specifically for appliance repair could improve both performance and efficiency. While Llama 3 provides excellent general capabilities, a specialized model could potentially achieve better results with lower computational requirements. This specialization could extend to individual agents, with each optimized for its specific role.

6.7 Broader Implications

This work contributes to our understanding of how AI systems can provide technical assistance in specialized domains. The challenges we encountered—from knowledge verification to safety assurance—are likely to arise in other technical support applications, whether for automotive repair, home improvement, or electronic device troubleshooting. Our modular approach and evaluation framework provide a foundation that other researchers can build upon.

The success of multi-modal integration suggests that future technical support systems must move beyond text-only interfaces. As vision models improve and become more accessible, we expect visual diagnosis to become standard in AI support tools. The ability to show rather than just describe problems represents a fundamental improvement in human-AI communication for technical tasks.

Our experience highlights the continued importance of human oversight in safety-critical applications. The application demonstrates that current AI technology requires careful human-in-the-loop validation to ensure that the outputs are viable and unproblematic. Building systems that know their limitations and actively seek human input when uncertain remains an important research direction.

Finally, this work demonstrates both the promise and challenges of applying cutting-edge AI research to practical problems. While academic benchmarks show impressive capabilities, translating these into reliable, safe, and useful real-world systems requires addressing numerous practical challenges. Our experience suggests that the gap between research demonstrations and deployed applications remains significant, but not insurmountable with careful engineering and appropriate safety measures.

REFERENCES

[1] Umang Bhatt, Sanyam Kapoor, Mihir Upadhyay, Ilia Sucholutsky, Francesco Quinzan, Katherine M Collins, Adrian Weller, Andrew Gordon Wilson, and Muhammad Bilal Zafar. 2025. When should we orchestrate multiple agents? *arXiv preprint arXiv:2503.13577*.

[2] Kang Chen, Tianli Zhao, and Xiangqian Wu. 2023. Vtqa2023: Acm multimedia 2023 visual text question answering challenge. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 9646–9650.

[3] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. 2024. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 24185–24198.

[4] Ali Dorri, Salil S Kanhere, and Raja Jurdak. 2018. Multi-agent systems: A survey. *Ieee Access*, 6:28573–28593.

[5] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

[6] Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley. 2023. Large language models as zero-shot conversational recommenders. In *Proceedings of the 32nd ACM international conference on information and knowledge management*, pages 720–730.

[7] Alisa Kongthon, Chatchawal Sangkeettrakarn, Sarawoot Kongyoung, and Choochart Haruechaiyasak. 2009. Implementing an online help desk system based on conversational agent. In *Proceedings of the International Conference on Management of Emergent Digital EcoSystems*, pages 450–451.

[8] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

[9] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pages 12888–12900. PMLR.

[10] Rengang Li, Cong Xu, Zhenhua Guo, Baoyu Fan, Runze Zhang, Wei Liu, Yaqian Zhao, Weifeng Gong, and Endong Wang. 2022. Ai-vqa: visual question answering based on agent interaction with interpretability. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 5274–5282.

[11] Yubo Li, Xiaobin Shen, Xinyu Yao, Xueying Ding, Yidi Miao, Ramayya Krishnan, and Rema Padman. 2025. Beyond single-turn: A survey on multi-turn interactions with large language models. *arXiv preprint arXiv:2504.04717*.

[12] Michael McTear. 2022. *Conversational ai: Dialogue systems, conversational agents, and chatbots*. Springer Nature.

[13] Nima Nabizadeh, Dorothea Kolossa, and Martin Heckmann. 2020. Myfixit: an annotated dataset, annotation tool, and baseline methods for information extraction from repair manuals. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2120–2128.

[14] Brian O Samba. 2023. *Improving It Service Desk Through the Use of Ai-powered Itsm Chatbot: a Case Study of Moh Kenya Service Desk*. Ph.D. thesis, University of Nairobi.

[15] Oguzhan Topsakal and Tahir Cetin Akinci. 2023. Creating large language model applications utilizing langchain: A primer on developing llm apps fast. In *International Conference on Applied Engineering and Natural Sciences*, volume 1, pages 1050–1056.

[16] Haixun Wang and Taesik Na. 2024. Rethinking e-commerce search. In *ACM SIGIR Forum*, volume 57, pages 1–19. ACM New York, NY, USA.

[17] Jialin Wang and Zhihua Duan. 2024. Agent ai with langgraph: A modular framework for enhancing machine translation using large language models. *arXiv preprint arXiv:2412.03801*.

[18] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*.

[19] Diji Yang, Linda Zeng, Jinmeng Rao, and Yi Zhang. 2025. Knowing you don't know: Learning when to continue search in multi-round rag through self-practicing. *arXiv preprint arXiv:2505.02811*.

[20] Zeyu Zhang, Paul Groth, Iacer Calixto, and Sebastian Schelter. 2024. Anymatch–efficient zero-shot entity matching with a small language model. *arXiv preprint arXiv:2409.04073*.