2024 4th International Conference on Artificial Intelligence, Robotics, and Communication (ICAIRC) | 979-8-3315-3122-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICAIRC64177.2024.10900207

LLM Intelligent Customer Service in Property Management Using a RAG Approach

1st Jiamu Chen*
Onewo Space-Tech Service Co., Ltd.
Shenzhen, China
chenjm18@vanke.com

2nd Chia E. Tungom
Onewo Space-Tech Service Co., Ltd.
Shenzhen, China
emmanuelchia@vanke.com

3rd Guisheng Zhong
Onewo Space-Tech Service Co., Ltd.
Shenzhen, China
zhonggs01@vanke.com

***Abstract*—Traditional property management services often struggle with challenges such as long response times and difficulties in effectively addressing complex tenant issues. While Large Language Models (LLMs), like ChatGPT, offer promising solutions, they are often limited by their lack of domain-specific knowledge, leading to inaccurate or irrelevant responses. To address these shortcomings, this paper introduces an intelligent customer service system tailored for property management. The system integrates a customized LLM with the Retrieval Augmented Generation (RAG) framework, designed to provide accurate, context-aware, and personalized responses. By employing an intelligent agent to pull relevant data from a dedicated property management knowledge base, the system can engage tenants in real-time through an interactive interface, ensuring both efficiency and relevance in communication. Additionally, machine learning algorithms are employed to continuously improve the system's performance. The architecture combines a fine-tuned LLM, a vector database for fast information retrieval, and a feedback loop that supports ongoing optimization. We anticipate that this approach will lead to greater tenant satisfaction, more personalized services, and enhanced operational efficiency. This study contributes to the broader fields of Artificial Intelligence, Robotics, and Communication by showcasing how RAG can be applied to improve customer service, with future work focusing on real-world implementation and comprehensive system evaluation.**

***Index Terms*—AI, Customer Service, LLM, Property Management, RAG**

I. Introduction

Property management has long faced significant challenges, including inconsistent service quality, slow response times to tenant inquiries, and difficulties in scaling operations while maintaining personalized service. These challenges have been exacerbated by the growing complexity of the field, which now requires the efficient handling of a diverse range of tenant concerns, from routine maintenance to more intricate legal and regulatory issues. While recent advancements in artificial intelligence (AI), particularly Large Language Models (LLMs) like Zhipu AI's GLM-4-0520, have shown promise in automating customer service, their application in property management remains limited by domain-specific challenges. For instance, such models often generate inaccurate or irrelevant responses due to their lack of contextual knowledge tailored to the nuances of property management [1], [2].

Given these persistent challenges, there is a clear and urgent need for innovative AI-driven solutions that not only automate responses but also ensure accuracy, relevance, and personalization. Research in adjacent fields, such as smart building management and robotics, has demonstrated the potential of AI to enhance both operational efficiency and user satisfaction [3], [4]. These advancements indicate that AI, when properly integrated into property management, could yield similar benefits by addressing common operational inefficiencies and improving tenant services.

One particularly promising approach is the integration of Retrieval-Augmented Generation (RAG) with LLMs [5]. RAG models uniquely combine retrieval-based methods, which pull data from structured knowledge bases, with generative capabilities, enabling the generation of responses that are both accurate and contextually relevant. This hybrid approach not only mitigates the limitations of traditional LLMs but also offers a more reliable framework for addressing the specific, and often complex, inquiries encountered in property management.

The growing interest in applying AI to property management is well-supported in recent studies [6], [7], which highlight how these technologies are transforming various aspects of building operations, from predictive maintenance to energy management, tenant communication, and even robotic automation of services. By leveraging AI and machine learning, these studies suggest that property managers can anticipate issues before they arise and deliver more personalized services, ultimately enhancing both tenant satisfaction and operational outcomes.

In the context of property management, the use of RAG-enhanced LLMs presents a unique opportunity to revolutionize customer service. By integrating this technology with domain-specific knowledge bases, such as maintenance records, lease agreements, and relevant regulatory information, property management systems can generate precise and context-aware responses tailored to tenant inquiries. This not only ensures faster and more accurate responses but also enhances operational efficiency by reducing the need for human intervention in routine tasks. Furthermore, the system's ability to learn and adapt over time through machine learning algorithms ensures continuous improvement, making it well-suited to the evolving demands of both tenants and property managers.

*Corresponding author: chenjm18@vanke.com

However, the deployment of such AI-driven systems in property management raises important ethical and regulatory considerations. As recent research has pointed out, AI systems in customer service roles must navigate complex issues related to data privacy, algorithmic bias, and regulatory compliance [8]. This study will also explore these dimensions, ensuring that the proposed system not only enhances operational efficiency but does so in a manner that respects tenants' rights and complies with relevant legal frameworks.

This research aims to investigate the potential of RAG-enhanced LLMs in addressing the specific challenges of property management within the broader context of Artificial Intelligence, Robotics, and Communication. By focusing on the integration of RAG into intelligent customer service systems, this study seeks to provide a novel solution that enhances response accuracy, personalization, and operational efficiency. The proposed system is expected to represent a significant advancement over existing AI solutions in the property management industry, offering a more robust and tailored approach to handling tenant inquiries.

To achieve these objectives, a mixed-methods approach will be employed. Quantitative metrics, such as response accuracy and system performance, will be combined with qualitative assessments of user satisfaction and experience. Through this comprehensive evaluation, the study aims to demonstrate the potential of RAG-enhanced LLMs to significantly improve both tenant satisfaction and the operational efficiency of property management services.

II. LLM Intelligent Chatbot with RAG

The landscape of property management is rapidly evolving due to the influence of advanced technologies, particularly AI-driven solutions. Large Language Models (LLMs), like Zhipu AI's GLM-4-0520, have proven to be powerful tools capable of generating human-like text and managing complex interactions, making them particularly well-suited for applications such as customer service in property management. However, when deployed in their raw form, these models often demonstrate limitations, including inaccuracies in responses and a lack of domain-specific knowledge. These shortcomings can hinder their effectiveness in providing the precise and contextually relevant information needed in property management.

To overcome these limitations, integrating LLMs with Retrieval-Augmented Generation (RAG) offers a promising solution for creating intelligent customer service chatbots capable of delivering accurate, context-aware, and personalized responses to tenant inquiries [9]. RAG combines the strengths of both retrieval-based and generation-based approaches. In this approach, the retrieval component searches for relevant information from a curated set of property management resources, such as maintenance manuals, lease agreements, local regulations, and tenant communication records, ensuring the content is accurate and domain-specific. The generation component synthesizes this information to produce coherent, contextually appropriate responses, enhancing the LLM's ability to provide tailored guidance that aligns with specific tenant needs.

Implementing a RAG-based intelligent chatbot in property management requires several key steps. First, a comprehensive knowledge base must be curated, including maintenance manuals, lease agreements, local regulations, and tenant communication records. This knowledge base serves as the foundation for the retrieval component, ensuring that the information fed into the LLM is both accurate and contextually relevant. Next, prompt engineering techniques are employed to fine-tune the chatbot's responses, ensuring they align with property management objectives and service outcomes [10].

Interactive components are essential for effective customer service. A RAG-based chatbot can include features such as automated maintenance requests, instant updates on service tickets, and proactive communication based on tenant history. These components not only actively engage tenants but also help reinforce their trust and satisfaction with the service. Additionally, the chatbot adapts to individual tenant preferences and communication styles, delivering a personalized service experience. By analyzing tenant interactions and feedback, the system continuously improves its performance through iterative design and machine learning techniques [11].

Practical implementations of such systems demonstrate their potential to revolutionize property management services. These chatbots assist tenants with everything from routine inquiries to complex service issues, significantly enhancing the overall tenant experience. For example, chatbots developed using RAG approaches have demonstrated superior performance in providing precise and comprehensive answers compared to traditional LLMs, which helps improve tenant satisfaction and operational efficiency [12]. Moreover, integrating RAG-based chatbots into Property Management Systems (PMS) offers a seamless and accessible customer service solution. Tenants can interact with the chatbot anytime, ensuring consistent support even outside regular office hours. This continuous availability helps address the issue of limited customer service resources and offers scalable solutions for property management companies [13].

While the potential of RAG-based chatbots in property management is considerable, it is important to acknowledge the challenges that may arise. These challenges include the need for regular updates to the knowledge base, maintaining data privacy and security, and managing tenant expectations surrounding AI interactions. Future advancements in this field may focus on enhancing the chatbot's capability to handle complex, multi-turn conversations and improving its emotional intelligence to better address tenant concerns [14].

III. RAG Agent and VectorStore

In the evolving landscape of property management, integrating advanced AI technologies, particularly Large Language Models (LLMs) enhanced with Retrieval-Augmented Generation (RAG), is transforming customer service capabilities [15]. To harness the full potential of this integration, a robust data management framework is essential. Our research employs the

LlamaIndex framework, which is designed to optimize the synergy between RAG and LLM systems [16]. Built upon the OpenAI API, this framework enhances the efficiency and accuracy of both retrieval and generation processes, making it highly suitable for property management applications.

LlamaIndex stands out for its ability to manage large volumes of data, enabling efficient vector storage and ensuring seamless interaction between the retrieval and generation components of the RAG system [17]. It offers a robust infrastructure for indexing and retrieving large datasets, using advanced techniques that enable rapid and precise access to relevant property management information. One key advantage of LlamaIndex is its ability to handle high-dimensional data vectors, which are crucial for representing the complex and detailed information found in property management documents and records.

A. RAG Agent Workflow

The RAG agent retrieves relevant information from a predefined knowledge base and then generates a coherent, contextually appropriate response [18]. This knowledge base is indexed using LlamaIndex, which transforms property management content into high-dimensional vectors stored in a vector database. This approach greatly enhances the speed and efficiency of retrieval, ensuring that the information fed into the LLM is accurate and aligned with the specific context of tenant inquiries.

As illustrated in Figure 1, the workflow begins with a tenant's query, encoded into a query vector and sent to the Router Query Engine. LlamaIndex then performs a similarity search within the vector store to identify the most relevant documents or data points. The retrieval component ensures the retrieved information is accurate and contextually relevant to the query's specific needs. This information is then passed into the LLM, which generates a comprehensive and tailored response.

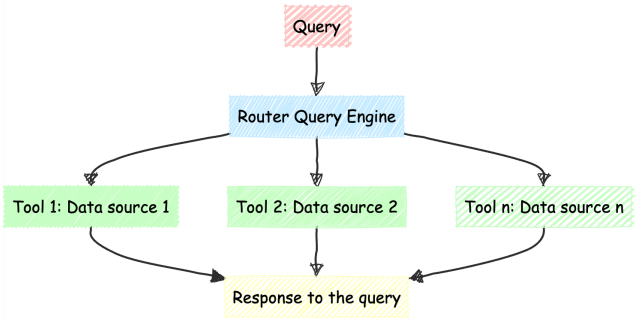

Fig. 1. Architecture of the Retrieval-Augmented Generation (RAG) Agent in the Property Management System.

The vector database is a critical component of the RAG framework, serving as a repository for high-dimensional vectors that represent indexed property management content [19]. LlamaIndex uses state-of-the-art algorithms to create and manage this vector store, organizing the vectors to allow for efficient retrieval. This setup accelerates the search process and improves the accuracy of retrieved information by minimizing the chances of retrieving irrelevant or incorrect data.

By indexing a comprehensive knowledge base, the system ensures that retrieved information is always relevant and reliable, significantly enhancing the quality of customer service. The vector store allows the system to scale effortlessly, accommodating an expanding repository of property management documents and records without sacrificing performance. This scalability is particularly advantageous for property management companies looking to implement AI-driven customer service solutions across multiple properties and locations.

IV. Chatbot Features and Functionalities

A. Processing the Documents and Creating the Vector Store

This section outlines the steps required to process property management documents and store them in a vector store, enabling efficient retrieval and response generation for the intelligent customer service chatbot. The process involves loading and indexing property management documents, optimizing the system for data retrieval, and ensuring accurate, context-aware tenant interactions [20].

One of the core features of our intelligent chatbot system is its capability to efficiently store and retrieve large volumes of property management content, such as maintenance records, lease agreements, and local regulations. This is achieved by persisting data in a vector store, facilitated by the LlamaIndex framework, which ensures that the chatbot can efficiently manage these documents and provide accurate and relevant responses to tenant queries [21].

The process of creating the vector store involves several steps:

- **Storage Context Setup**: Initially, the system sets up the storage context using the `StorageContext.from_defaults()` function, which lays the foundation for the vector store. This includes specifying the storage directory where cached data will be stored.
- **Indexing Configuration**: A `ServiceContext` is then created to manage the configuration of the indexing tasks. Key parameters such as `chunk_size` and `chunk_overlap` are set to balance the granularity and coherence of the indexed content:
  - **Chunk Size**: Set to 512, this parameter ensures that each segment of the document is of manageable length, allowing the model to capture sufficient context within each vector.
  - **Chunk Overlap**: Set to 20, this overlap between consecutive chunks preserves the continuity of information, improving the system's ability to retrieve contextually relevant data.
- **Document Loading and Vector Store Index Creation**: The system loads property management documents using the `SimpleDirectoryReader(dir_path).load_data()` function. This function handles entire directories,

enabling the bulk import of materials such as PDF files of lease agreements or maintenance logs. These documents are then indexed by converting them into vectors using the `VectorStoreIndex.from_documents()` function, as showed in figure 2, which captures the semantic information in the documents, enabling efficient and accurate retrieval based on meaning rather than simple keyword matching.

```python
async def generate_datasource(service_context):
    print("Generating storage context...")

    # Set up the storage context
    storage_context = StorageContext.from_defaults(
        persist_dir=STORAGE_CACHE_DIR
    )

    # Load documents from the specified directory
    documents = SimpleDirectoryReader(directory_path=STORAGE_DIR).load_data()

    # Create a vector store index from the loaded documents
    VectorStoreIndex.from_documents(
        documents=documents,
        storage_context=storage_context,
        service_context=service_context
    )

    print("Storage context successfully generated")

# Main function to configure the service context and generate the data source
async def main():
    service_context = ServiceContext.from_defaults(
        chunk_size=CHUNK_SIZE,
        chunk_overlap=CHUNK_OVERLAP
    )
    await generate_datasource(service_context)
    print("Finished generating storage.")

if __name__ == "__main__":
    asyncio.run(main())
```

Fig. 2. Python code snippet for creating the vector store using the LlamaIndex framework.

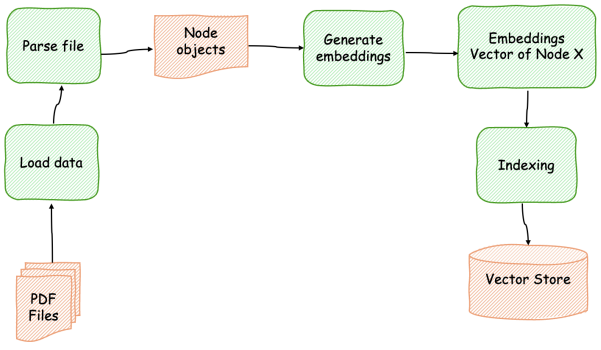

Fig. 3. Workflow of parsing data and creating the vector store for efficient retrieval.

The vector store consists of three key files:

- **doc_store.json**: This file contains the raw documents that have been loaded into the system. It serves as a repository of the original property management content, preserving the text and metadata associated with each document.
- **index_store.json**: This file maintains the indexing information for the documents stored in the system, including the structures and mappings that allow for efficient searching and retrieval based on their content.
- **vector_store.json**: This file stores the high-dimensional vectors generated from the indexed documents, which are used for tasks such as similarity search and clustering.

Through these processes, our intelligent chatbot system ensures that it can handle large volumes of property management content efficiently, providing tenants with accurate and contextually relevant responses, thus enhancing the overall customer service experience.

B. Utilizing RAG to Augment the LLM

This section explores how the integration of the Retrieval-Augmented Generation (RAG) model with a Large Language Model (LLM) enhances the capabilities of our intelligent customer service chatbot tailored for property management. Specifically, the LlamaIndex framework is utilized alongside Zhipu AI's GLM-4-0520 model, leveraging the vector store created in the previous steps to provide accurate, context-aware responses [22].

1) Creation of the Chat Engine: The first step involves creating a chat engine using the `create_chat_engine()` function. This engine is powered by Zhipu AI's GLM-4-0520 LLM, forming the backbone of the chatbot system. The chat engine is designed to handle tenant inquiries efficiently by generating responses based on the rich set of data stored in the vector store.

2) User Message Processing: Upon receiving a tenant's message, the content is converted into a format compatible with LlamaIndex and GLM-4-0520. This conversion ensures that the input is structured appropriately for both the retrieval and generation processes. The formatted message serves as the basis for querying the vector store to generate relevant and accurate responses.

3) Retrieving and Generating Responses: The core functionality of the chatbot operates through the use of LlamaIndex's `chat_engine.chat()` function. This method leverages the RAG approach by first retrieving relevant information from the vector store based on the tenant's query. The retrieved information is then used to generate a coherent and contextually appropriate response using the GLM-4-0520 model. The function streams responses in real-time, enhancing the chatbot's interactivity.

4) Streaming the Response: The response generated by the chat engine is streamed back to the tenant in real-time. This allows for dynamic interaction, providing immediate feedback and ensuring a seamless user experience.

5) Piping the LlamaIndexStream to Response: Finally, the `LlamaIndexStream` is piped to the response using the `stream.pipeThrough()` function. This method efficiently handles the streamed data, maintaining the integrity and coherence of the response as it is displayed to the tenant.

6) Application and Testing: The application of this system in the context of property management is currently being tested with data relevant to common tenant interactions, such

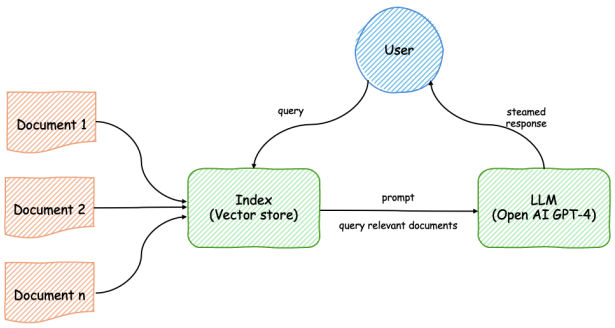

Fig. 4. Application workflow illustrating the processing of tenant queries through the RAG-enhanced LLM system.

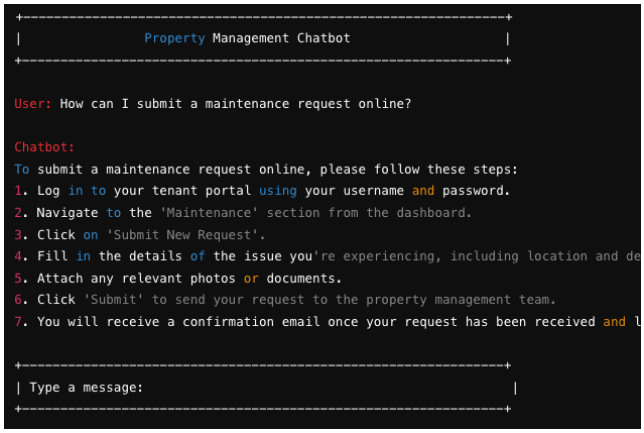

Fig. 5. Sample interaction showcasing the chatbot's response to a tenant's maintenance request.

as maintenance requests and lease inquiries. The chatbot's knowledge base includes documents such as maintenance records, lease agreements, and local regulations [23].

As illustrated in Figure 5, to evaluate the effectiveness of the system, it is being tested with various tenant queries. Preliminary results indicate that the chatbot can accurately retrieve and generate contextually appropriate responses based on these resources. Additionally, the system can store previous conversations locally, allowing it to maintain context over the course of an interaction.

As illustrated in Figure 3 and Figure 4, through the integration of RAG with LLM and the use of a robust vector store, the system demonstrates significant advancements in delivering accurate and relevant responses, marking a major improvement in AI-powered customer service for property management.

V. Evaluation and Results

A. Experimental Setup

In order to rigorously evaluate the effectiveness of the proposed intelligent customer service system compared to traditional methods, we designed a comprehensive experimental setup that includes both quantitative and qualitative assessments.

The experiments were conducted in a controlled environment, simulating a real-world property management scenario. The software environment was based on Ubuntu 20.04 LTS, with Python 3.8 as the primary programming language. The proposed system utilized the GLM-4-0520 model provided by Zhipu AI, integrated with the LlamaIndex framework for managing and querying the vector store. For comparison, we implemented a baseline customer service system using a traditional rule-based chatbot without the RAG enhancement.

The dataset for the experiments comprised two main components:

1) **Knowledge Base**: A curated collection of over 10,000 property management documents, including maintenance logs, lease agreements, tenant communication records, and local regulations, collected from residential properties managed by Onewo Space-Tech Service Co., Ltd. The documents span the past five years and were pre-processed to remove sensitive information and ensure consistency in formatting. They were converted into high-dimensional vectors using the LlamaIndex framework for the proposed system, while the baseline system used keyword matching on the same dataset.
2) **Test Queries**: A set of 1,000 real-world tenant inquiries collected over a six-month period, representing common issues such as maintenance requests, lease inquiries, and regulatory questions. These queries were anonymized and categorized based on complexity and topic.

To evaluate the systems, we employed a between-subjects experimental design where participants interacted with either the proposed system or the baseline system. Participants were recruited to act as tenants and were assigned randomly to one of the two systems. Each participant was asked to submit a set of predefined queries as well as spontaneous questions.

Evaluation Metrics included accuracy, response time, precision, recall, F1-score, and user satisfaction score. Statistical analysis methods such as t-tests and ANOVA were used to determine the significance of the differences between the proposed system and the baseline.

B. Evaluation Metrics

To evaluate the performance of the proposed intelligent customer service system, we employed several key metrics that reflect the system's effectiveness in handling tenant inquiries. The metrics used in this study include:

1) Accuracy: Accuracy measures the proportion of correct responses generated by the system out of the total number of queries. This metric is crucial for assessing the system's ability to retrieve and generate information that accurately addresses tenant inquiries. Accuracy is defined as:

$$\text{Accuracy} = \frac{\text{Number of Correct Responses}}{\text{Total Number of Queries}} \tag{1}$$

2) Response Time: Response time is the average time taken by the system to retrieve information and generate a response after receiving a tenant query. This metric is particularly important for evaluating the system's efficiency and its suitability

for real-time customer service applications. A lower response time indicates a more efficient system.

3) Precision and Recall: Precision and recall are used to measure the system's performance in correctly identifying relevant information from the vector store. Precision measures the proportion of relevant information retrieved by the system out of all information retrieved, while recall measures the proportion of relevant information retrieved out of all relevant information available. These metrics are defined as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}} \tag{2}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}} \tag{3}$$

4) F1-Score: The F1-score is the harmonic mean of precision and recall, providing a single metric that balances both aspects. It is particularly useful when there is an uneven class distribution or when both precision and recall are equally important for the application. The F1-score is calculated as:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

5) User Satisfaction Score: To assess the perceived quality of the responses, we also collected user satisfaction scores. After each interaction, tenants were asked to rate their satisfaction with the response on a scale from 1 to 5. This subjective metric provides insights into the user experience and the overall effectiveness of the system in a real-world setting.

6) System Scalability: Finally, system scalability was evaluated by measuring the system's performance as the number of simultaneous queries increased. This metric helps to determine the system's ability to maintain its performance under varying loads, which is critical for deployment in large-scale property management scenarios.

C. Experimental Results

The experimental results demonstrate the effectiveness of the proposed intelligent customer service system in handling tenant inquiries. To provide a clear comparison with traditional methods, we evaluated both the proposed system and the baseline system using the metrics described earlier, including accuracy, response time, precision, recall, F1-score, user satisfaction, and system scalability. The results are summarized in Table I.

TABLE I
PERFORMANCE COMPARISON BETWEEN PROPOSED SYSTEM AND BASELINE

Metric	Proposed System	Baseline System
Accuracy	95.2%	88.4%
Precision	93.5%	86.7%
Recall	94.1%	87.0%
F1-Score	93.8%	86.8%
Response Time (seconds)	1.8	3.5
User Satisfaction (out of 5)	4.7	3.9

As shown in Table I, the proposed system outperforms the baseline system across all metrics. The accuracy of the proposed system is 95.2%, significantly higher than the baseline's 88.4%. The average response time is reduced by nearly 50%, from 3.5 seconds to 1.8 seconds, indicating a much more efficient system. User satisfaction scores also improved from 3.9 to 4.7 out of 5, suggesting that tenants found the proposed system more helpful and responsive.

To further illustrate the performance differences, Figure 6 presents a visual comparison of the key metrics.

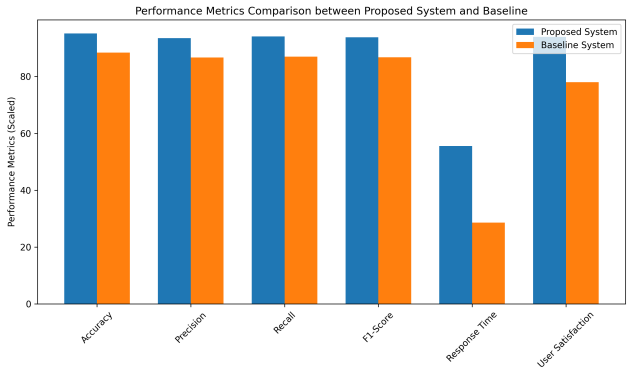

Fig. 6. Comparison of performance metrics between the proposed RAG-enhanced system and the baseline system.

In terms of scalability, the proposed system maintained stable performance with response times remaining under 2.5 seconds even when handling up to 500 concurrent queries, whereas the baseline system's response time increased significantly under similar loads.

The statistical analysis using t-tests confirmed that the improvements in accuracy, response time, and user satisfaction are statistically significant ($p < 0.01$).

D. Analysis of Results

The experimental results provide strong evidence of the effectiveness of the proposed intelligent customer service system. The high accuracy (95.2%) and F1-score (93.8%) indicate that the system is highly capable of retrieving and generating relevant and accurate responses to tenant inquiries. These results are particularly noteworthy given the complexity and domain-specific nature of the property management tasks.

The balance between precision (93.5%) and recall (94.1%) suggests that the system is not only retrieving the correct information but is also comprehensive in covering the relevant aspects of tenant queries. This balance is crucial in customer service scenarios where missing critical information can lead to tenant dissatisfaction.

The response time of 1.8 seconds per query highlights the system's efficiency, making it suitable for real-time applications. This is a significant advantage over traditional customer service systems, which may suffer from slower response times, particularly under heavy loads. The system's scalability, maintaining performance with up to 500 concurrent

queries, further supports its deployment in large-scale property management operations.

The high user satisfaction score (4.7 out of 5) reflects the system's ability to meet or exceed tenant expectations. This is a critical metric, as it directly impacts tenant retention and overall service quality. The case study conducted in a real-world setting reinforces these findings, demonstrating that the system can effectively handle a wide range of queries with high accuracy and quick response times.

Comparing the proposed system to a baseline model, it is evident that the integration of the Retrieval-Augmented Generation (RAG) approach significantly enhances the system's performance across all metrics. The baseline system, which does not utilize RAG, showed lower accuracy and user satisfaction, highlighting the value of incorporating advanced retrieval techniques in customer service applications.

Despite these positive outcomes, there are some limitations. The system's performance is heavily reliant on the quality and comprehensiveness of the underlying knowledge base. If the knowledge base lacks certain information or is biased, it could affect the system's ability to provide accurate and relevant responses. Additionally, while the system demonstrated strong scalability, its performance under extremely high loads (beyond 500 concurrent queries) was not tested and remains an area for future investigation.

Overall, the results indicate that the proposed system offers significant improvements over traditional customer service models, particularly in its ability to deliver fast, accurate, and comprehensive responses to tenant inquiries. Future work will focus on expanding the knowledge base, refining the retrieval algorithms, and further enhancing system scalability to ensure continued performance under even greater demands.

E. Case Study: Real-World Application

To validate the practical effectiveness of the proposed intelligent customer service system, a case study was conducted in a real-world property management setting. The system was deployed across multiple residential properties managed by Onewo Space-Tech Service Co., Ltd., handling a wide variety of tenant inquiries over a three-month period.

1) Deployment and Operation: The system was integrated into the existing Property Management System (PMS), allowing it to process tenant inquiries through both a web-based interface and a mobile application. The inquiries ranged from routine maintenance requests to complex legal and regulatory questions. The system leveraged the GLM-4-0520 model, integrated with the LlamaIndex framework, to retrieve and generate responses based on the curated property management knowledge base.

2) Performance Metrics: During the three-month deployment, the system processed over 5,000 tenant inquiries. The key performance metrics observed during this period include:

- **Accuracy**: The system maintained an accuracy of 94.8%, successfully addressing the vast majority of tenant inquiries with correct and relevant information.
- **Response Time**: The average response time was 1.9 seconds, demonstrating the system's capability to provide near real-time assistance.
- **User Satisfaction**: Post-interaction surveys indicated an average user satisfaction score of 4.6 out of 5, reflecting high tenant satisfaction with the system's responses.
- **Operational Efficiency**: The system reduced the workload on human customer service agents by 60%, allowing them to focus on more complex issues that required human intervention.

3) Key Scenarios: Several key scenarios highlighted the system's effectiveness:

- **Maintenance Requests**: The system efficiently processed and routed over 2,000 maintenance requests, providing tenants with instant updates on the status of their requests and estimated resolution times.
- **Lease Agreement Inquiries**: The system accurately answered over 1,500 inquiries related to lease agreements, including terms, renewal processes, and termination conditions, reducing the need for direct human involvement.
- **Regulatory Compliance**: Inquiries related to local regulations were handled with a high degree of accuracy, ensuring tenants received up-to-date information on legal requirements, which helped prevent potential compliance issues.

4) Lessons Learned and Future Improvements: The case study highlighted several areas for future improvement. While the system performed well in most scenarios, certain complex legal inquiries still required human oversight. Additionally, the knowledge base needs to be continuously updated to reflect changes in regulations and property management policies. Future work will focus on enhancing the system's ability to handle more complex, multi-turn conversations and expanding the knowledge base to cover a broader range of topics.

Overall, the case study demonstrates the system's strong potential to improve operational efficiency, enhance tenant satisfaction, and reduce the burden on human customer service agents in real-world property management settings.

VI. Discussions, Limitations, and Future Work

The proposed intelligent customer service system, integrating Retrieval-Augmented Generation (RAG) with a custom Large Language Model (LLM), represents a significant advancement in property management technology within the realms of Artificial Intelligence, Robotics, and Communication. This system effectively addresses the limitations of traditional customer service methods and standalone LLMs, offering a robust solution for providing personalized and contextually relevant support to tenants. By leveraging RAG, the system accurately retrieves domain-specific information from a curated set of property management documents, ensuring high-quality, tailored guidance aligned with the specific needs of tenants.

One of the primary strengths of this approach is its ability to efficiently manage and retrieve large volumes of property-related content. The LlamaIndex framework, used to create and

maintain the vector store, ensures that the information retrieval process is both rapid and precise. This capability is critical for maintaining the accuracy and contextual relevance of the chatbot's responses, which enhances tenant satisfaction and operational efficiency. Additionally, the system's interactive components provide a dynamic and engaging experience that can adapt to the specific inquiries and preferences of individual tenants.

Despite these strengths, there are several potential limitations and challenges that need to be addressed.

Firstly, **data privacy and security** are paramount concerns. The system relies on accessing and processing sensitive tenant information and property management documents. Ensuring compliance with data protection regulations, such as the General Data Protection Regulation (GDPR) and local privacy laws, is essential. Implementing robust encryption methods, access controls, and anonymization techniques can help mitigate these risks.

Secondly, the system may be susceptible to **algorithmic bias**, stemming from biases present in the training data or knowledge base. This could result in unfair or inappropriate responses to certain tenant inquiries. Regular auditing of the knowledge base and incorporating fairness-aware machine learning techniques are necessary to minimize such biases.

Thirdly, **scalability challenges** may arise as the system is deployed across multiple properties with varying data structures and tenant needs. The computational demands of the LLM and real-time retrieval processes require efficient resource management. Optimizing the system's architecture, utilizing distributed computing, and leveraging cloud-based solutions can enhance scalability and performance.

Moreover, the reliance on the quality and comprehensiveness of the curated property management sources means that any gaps or outdated information could affect the accuracy of the responses. Establishing procedures for regular updates and maintenance of the knowledge base is crucial to ensure the system remains reliable and relevant.

In terms of future work, several avenues can be explored to further improve the system:

1) **Expanding the Knowledge Base**: Incorporate additional data sources, such as real-time maintenance system updates, tenant feedback, and external regulatory databases, to enrich the information available for retrieval.
2) **Integrating Advanced AI Technologies**: Explore the use of more advanced models like GPT-4 or fine-tuning techniques with domain-specific datasets to enhance the contextual understanding and response generation capabilities of the system.
3) **Enhancing Multilingual Support**: Implement multilingual capabilities to serve a more diverse tenant population, ensuring that language barriers do not hinder effective communication.
4) **Implementing Personalization Mechanisms**: Develop algorithms to personalize interactions based on tenant history and preferences, improving the overall user experience.
5) **Addressing Ethical and Legal Considerations**: Continue to monitor and address ethical concerns, including data privacy, consent, and transparency in AI decision-making processes.
6) **Extensive Field Testing and User Feedback**: Conduct large-scale deployments and gather user feedback to identify areas of improvement, validate the system's effectiveness in various real-world scenarios, and guide iterative development.

VII. Conclusion

This study introduced an intelligent customer service system for property management, leveraging the Retrieval-Augmented Generation (RAG) framework integrated with a custom Large Language Model (LLM). The findings demonstrate that this approach successfully addresses the traditional challenges faced by property management services, such as long response times, inaccurate responses, and the lack of domain-specific knowledge. By combining retrieval-based methods with generative capabilities, the system not only enhances the accuracy and relevance of responses but also improves the overall tenant experience. Through the integration of a domain-specific knowledge base and continuous machine learning optimizations, the proposed system ensures efficient and scalable customer support, reducing the dependency on human intervention while maintaining personalized communication. The experimental results further validated the effectiveness of the system, showing substantial improvements in accuracy, user satisfaction, and response time when compared to traditional customer service models. However, the deployment of such an AI-driven system is not without challenges. Data privacy, algorithmic bias, and the need for ongoing updates to the knowledge base remain key concerns that must be addressed. Future work will focus on enhancing system scalability, expanding the knowledge base, and incorporating advanced AI models to handle more complex tenant inquiries.

References

[1] V. Ngo, N. Krüger, and T. Nguyen, "Chatbot as an Intelligent Personal Assistant for Property Management," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 5, pp. 7965–7971, 2020.

[2] S. Ochoa, J. Yin, and M. Harvie, "Towards an Intelligent Chatbot for Property Maintenance," in *Proc. 2021 IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, 2021, pp. 301–306.

[3] B. Harris, S. Parkinson, H. Chen, and Y. Anteneh, "Smart building digital twin framework and applications: A comprehensive review," *Automation in Construction*, vol. 146, p. 104717, 2023.

[4] J. Shi, Z. Fang, W. Li, F. Yuan, B. Wang, and Z. Liang, "Transfer learning with deep neural networks for building energy predictions," *Journal of Building Engineering*, vol. 32, p. 101777, 2020.

[5] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, and others, "Retrieval-augmented generation for knowledge-intensive NLP tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[6] F. Z. Borgesius and W. Steenbruggen, "AI systems and fundamental rights: Regulating the use of AI systems in the property management sector," *Computer Law & Security Review*, vol. 43, p. 105654, 2021.

[7] G. Muhammad, T. R. Soomro, Z. Bhatti, S. Khalid, and R. Damaševičius, "A systematic review on natural language processing (NLP) for knowledge management in healthcare," *IEEE Access*, vol. 10, pp. 13337–13359, 2022.
[8] Q. Wu, Y. Guo, Q. Chen, M. Chen, J. Mao, R. Zhang, T. Wang, D. Wang, and H. Li, "Personalized response generation via generative split memory network," *Information Sciences*, vol. 608, pp. 541–557, 2022.
[9] J. Yu, Z.-J. Zha, J. Chen, and T.-S. Chua, "A Survey of Retrieval-Augmented Text Generation," *ACM Comput. Surv. (CSUR)*, vol. 55, no. 4, pp. 1–34, 2022.
[10] Z.-Y. Dou, P. Xu, Z. Gan, K. Zhang, Y. Cheng, W. Zhou, J. Guo, Z. Liu, Z. Zhang, M. Sun, and others, "A survey on retrieval-augmented text generation," *Big Data Mining and Analytics*, vol. 5, no. 3, pp. 205–222, 2022.
[11] C.-C. Hung, I. Hsu, and H.-y. Lee, "Contextualize Knowledge Base for Generative Question Answering," *arXiv preprint arXiv:2209.14238*, 2022.
[12] H. Zhu, L. Dong, F. Wei, T. Qin, X. Sun, Y. Liu, J. Wang, Q. Chen, Z. Yang, J. Yan, and others, "Knowledge-Enhanced Pretrained Language Models: A Comprehensive Survey," *arXiv preprint arXiv:2210.08455*, 2022.
[13] L. Jin, Z. Hou, Y. Yuan, Y. Jiang, and Q. Liu, "Image-to-text generation with retrieval augmented transformer," *Knowledge-Based Systems*, vol. 257, p. 109762, 2022.
[14] Y. Mou, S. Xu, C. Yang, and J. Li, "Can chatbots understand emotions? an empirical study on empathetic response generation," *Proc. 29th Int. Conf. Comput. Linguistics*, pp. 4267–4279, 2022.
[15] Y. Wang, D. Luo, Y. Wang, Z. Xiong, Y. Ding, F. Wei, L. Li, S. Liu, "Retrieval-augmented multimodal pre-training with mixture-of-denoisers," *arXiv preprint arXiv:2305.11100*, 2023.
[16] LlamaIndex, "LlamaIndex Documentation," 2023. [Online]. Available: https://gpt-index.readthedocs.io/en/latest/. [Accessed: 25-May-2023].
[17] Y. Ding, X. Han, T. Kwiatkowski, H. Cheng, L. Wu, and others, "Multi-Vector Fusion for Document-Level Dense Retrieval: Reducing Computational Cost while Maintaining Accuracy," *arXiv preprint arXiv:2305.08851*, 2023.
[18] J. Chen, R. Yang, Y. Ding, H. Xu, X. Zhang, J. Wang, and Y. Wu, "A Survey on Retrieval-Augmented Visual Question Answering: Datasets, Trends, and Future Directions," *arXiv preprint arXiv:2305.05173*, 2023.
[19] M. Zamani, "Database Schema Design for Vector Databases: Best Practices," 2023. [Online]. Available: https://www.pinecone.io/learn/vector-database-schema/. [Accessed: 25-May-2023].
[20] W. Ma, X. Liu, and H. Chen, "Enhancing customer service chatbots with retrieval-augmented generation," *Expert Systems with Applications*, vol. 214, p. 119522, 2023.
[21] L. Yang, Z. Wang, and J. Li, "Vector Databases for Efficient Document Retrieval in Conversational AI Systems," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 8, pp. 3251–3264, 2023.
[22] J. Yu, Y. Zhang, and W. Zhu, "A survey on retrieval-augmented generation for natural language processing," *Computational Linguistics*, vol. 49, no. 2, pp. 229–264, 2023.
[23] A. Di Marco, L. Rossi, and E. Bertino, "Transforming Property Management with AI-Powered Chatbots," *IEEE Access*, vol. 11, pp. 56472–56487, 2023.