Building a Retrieval-Augmented Generation (RAG) Customer Support Chatbot with Gemini and HNSW

Introduction

In today’s digital landscape, businesses face increasing demands for efficient, scalable, and secure customer service solutions. Traditional customer support systems often struggle with high operational costs, slow response times, and the inability to handle large volumes of inquiries. AI-powered customer service solutions offer a transformative approach by automating responses, improving accuracy, and enhancing user experience. The ability to switch between multiple large language models (LLMs) ensures flexibility and adaptability across different industries, making it a valuable tool for enterprises in e-commerce, technical support, education, and community forums.

In this project, an intelligent chatbot is implemented and evaluated for customer support using a Retrieval-Augmented Generation (RAG) architecture. The goal was to enhance the ability of large language models (LLMs) to respond with contextual accuracy by grounding them in a knowledge base. By leveraging Google’s Gemini 1.5 Flash model for text generation and embeddings, and HNSW (Hierarchical Navigable Small World) for fast similarity search, the system delivers context-aware responses while maintaining enterprise-grade scalability. This solution reduces reliance on human agents, cuts response times, and adapts seamlessly to diverse use cases, which can be ideal for e-commerce and technical support platforms.

Methods

In this project, the Bitext Customer Support dataset from Kaggle with 24,635 labeled prompt-response pairs was pre-selected to train the model. For pre-processing, the dataset was cleaned by removing missing values and standardizing the column names, which ensured that the data would be suitable for both embedding and evaluation.

The Gemini model’s text-embedding-004 API was utilized to facilitate the running environment on the Jupyter Notebook, which generates 768-dimensional embeddings for each customer prompt and can be replaced by any feasible embedding models. A caching mechanism was implemented to optimize model performance by avoiding redundant computations. The embeddings were indexed using HNSW, which provides a fast and memory-efficient approximate nearest-neighbor’s search capability. The index was configured to allow high-recall semantic similarity lookups for the chatbot's retrieval module.

For each incoming query, the top three semantically similar past responses were retrieved from the HNSW index, which were used as a grounding context in the prompt passed to the Gemini 1.5 Flash generative model. A few-shot learning section was included with sample question-answer pairs, along with structured instructions to produce a JSON response. Robust error handling was implemented, including retry logic for quota exhaustion, and fallback responses for incomplete or non-JSON outputs.

For the performance measurement, semantic and text-based metrics between the model’s response and the ground truth were evaluated. The cosine similarity (semantic similarity), BLEU (n-gram overlap), and ROUGE (recall-based textual similarity) were calculated, for each of these measures provides a different perspective on the model’s accuracy and relevance.

Results

The manual inspection of the outputs revealed high relevance and coherence, which confirmed the model validity. All generated responses were labeled as “low” confidence by default, which may be a conservative default output from Gemini. The average cosine similarity was 0.786, which suggests strong semantic alignment between generated and reference answers. BLEU was 0.048, and ROUGE scores were 0.170 (ROUGE-1), 0.085 (ROUGE-2), and 0.133 (ROUGE-L), implying relatively low literal overlap but good contextual relevance.

A closer analysis of the cosine similarity distribution provides deeper insights into the model's semantic understanding. Among the 200 test samples, 135 achieved a cosine similarity of 0.7 or higher, while the remaining 65 scored between 0.5 and 0.7. No responses fell below the 0.5 threshold, indicating that all generated responses maintained at least a moderate level of semantic similarity. This result underscores the strength of the RAG architecture in retrieving meaningful context that significantly informs the generated output.

In contrast, the BLEU and ROUGE scores were relatively low across the board. A BLEU score of 0.048 suggests that the surface-level n-gram overlap between generated and reference responses is minimal. Similarly, the ROUGE-1, ROUGE-2, and ROUGE-L scores being below 0.2 reflect a consistent pattern of low literal matches. However, this does not necessarily indicate poor performance. Rather, it highlights the model’s tendency to paraphrase or rephrase while maintaining the same intent and meaning. In customer support scenarios, semantic fidelity often takes precedence over verbatim responses, making high cosine similarity a more appropriate success metric.

Threshold analysis on ROUGE-L further supports this view. Only one response exceeded a ROUGE-L score of 0.7, and the remaining 199 were below that threshold. While this may appear unfavorable from a traditional NLP standpoint, it reinforces the conclusion that the generative model prioritizes contextual response quality over textual mimicry.

No technical errors or generation failures were detected throughout the experiment, which speaks to the robustness of the retry and fallback mechanisms. The chatbot was able to successfully handle all 200 queries, offering a complete output in each case, and custom queries yielded coherent and structured replies as well. These answers, evaluated manually, demonstrated clear and actionable steps and were rated with high confidence levels by the Gemini model (0.95), suggesting that the model performs especially well when the context aligns with frequently occurring support topics.

These findings indicate that the RAG chatbot built in this project is semantically robust, practically useful, and suitable for real-world deployment scenarios where precise wording is less critical than correct and contextually appropriate assistance.

Conclusion

This project proves RAG’s transformative potential for customer service by combining Retrieval-Augmented Generation with fast vector indexing and a capable LLM such as Gemini to build a responsive and intelligent customer service assistant. The chatbot showed high semantic accuracy (78%) and was able to retrieve relevant context and provide appropriate answers, even in varied phrasing situations. The evaluation metrics confirmed the robustness, especially in semantic similarity, although there is room for improvement in literal matching scores such as BLEU and ROUGE.

The adapted Gemini model can be updated with locally hosted LLMs supported by Ollama for increased privacy and reliability. Future work may include adjusting the confidence labeling mechanism by mapping low-confidence outputs to human agent escalations and integrating user feedback loops to enable dynamic learning through the auto-update of knowledge bases. Overall, the RAG-empowered chatbot provides a scalable and effective solution to intelligent customer support automation and demonstrates the blueprints for enterprises in the shortage of customer service with high accuracy and scalability.