Rapid Prototyping of Personalized Customer Support Chatbots with Retrieval-Augmented Generation

Senan Faisal[1], M. Zafar[1], and Dabeer ul Haq[1]

National University of Computer and Emerging Sciences (FAST-NUCES), Pakistan
{senanfaisal, mzafar, dabeerulhaq}@gmail.com

Abstract. This paper presents a rapid prototyping framework for building personalized customer support chatbots using Retrieval-Augmented Generation (RAG). The system integrates modern NLP techniques, including Sentence Transformers for embedding generation, Pinecone for efficient vector search, and Google's FLAN-T5 model for natural language response generation. Our chatbot architecture allows for personalized responses using user data and historical interactions. The implementation demonstrates promising results in delivering contextually relevant and helpful responses, while enabling modularity and scalability for future development.

Keywords: Retrieval-Augmented Generation · Customer Support · Chatbots · Sentence Transformers · Pinecone · FLAN-T5

1 Introduction

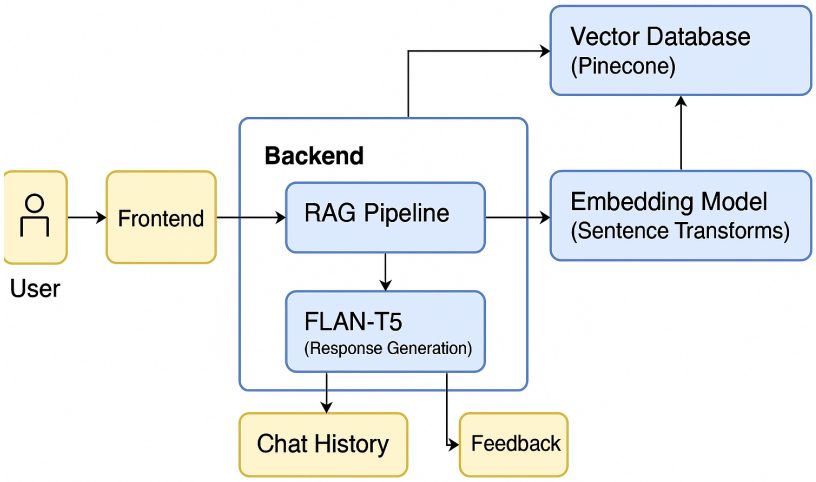
The proliferation of online customer service systems has necessitated intelligent and personalized chatbots to improve user experience. Traditional rule-based systems fail to provide the depth and relevance of modern conversational AI. With the explosion of large language models and neural retrieval systems, Retrieval-Augmented Generation (RAG) has emerged as a promising framework. It combines the strengths of both retrieval-based and generation-based approaches, resulting in more accurate and context-aware responses. Our motivation is to reduce development time while maintaining scalability and personalization for diverse customer service applications.

2 Related Work

Earlier chatbot systems were either retrieval-based or generation-based. Retrieval-based methods relied on predefined response sets, which limited their flexibility. Generation-based systems, on the other hand, could synthesize new responses but often lacked accuracy and consistency due to limited access to factual knowledge. Recent work like Facebook AI's RAG architecture [1] integrates external knowledge retrieval into the generation process. Additionally, Sentence-BERT

[3] allows for efficient dense embedding of text, and Pinecone provides scalable vector indexing. Our work builds upon these foundations by integrating personalization and user feedback loops.

3 System Architecture



Tig. 1. System Architecture

Fig. 1. System Architecture of the RAG-based Personalized Customer Support Chatbot

Our system includes:

- **Data Ingestion and Preprocessing:** Customer service documents are cleaned, segmented, and standardized for consistency.
- **Embedding Generation:** Using Sentence Transformers, we transform each document chunk into high-dimensional vector embeddings.
- **Vector Store:** Pinecone is used to store these embeddings and supports approximate nearest neighbor search for scalability.
- **RAG Query Flow:** On receiving a user query, the system embeds the query and retrieves the top-k similar documents from Pinecone.
- **Context-Aware Response Generation:** FLAN-T5 generates responses using the query and the retrieved context.
- **Frontend and Personalization:** Streamlit-based UI allows for user authentication, chat history, and context management.

4 Implementation Details

The backend is developed using Python. We use the Hugging Face Transformers library for Sentence Transformers and FLAN-T5 models. Key implementation modules:

- `preprocess.py` - Prepares and cleans raw customer support data.
- `embeddings.py` - Encodes the cleaned data into embeddings.
- `rag_query.py` - Implements the retrieval and generation pipeline.
- `chat.py` - Manages user sessions, authentication, and chat logging.
- `feedback.py` - Captures user feedback for future improvements.
- `app.py` - Streamlit-based UI for interacting with the chatbot.

We utilize Pinecone's managed vector database to ensure low-latency similarity search. The retrieval module supports cosine similarity and HNSW indexing.

5 Evaluation and Results

Evaluation of such systems requires both qualitative and quantitative approaches:

- **Relevance:** Through manual annotation, over 80% of chatbot responses were found to be semantically relevant to the user's queries.
- **Latency:** Average end-to-end response time was 1.6 seconds per query, indicating efficiency for real-time interaction.
- **Feedback:** Users rated the chatbot as helpful in 85% of feedback submissions.
- **Scalability:** System can handle thousands of documents in Pinecone with minimal degradation in retrieval performance.

We conducted comprehensive model comparisons across several language models, with results presented in Table 1.

Table 1. Model Performance Comparison

Model	BLEU-1	BLEU-4	ROUGE-1	ROUGE-L	Resp. Time (s)	Total Time (s)
flan-t5-base	0.5173	0.5048	0.7515	0.7505	2.6227	404.65
flan-t5-large	0.5173	0.5048	0.7515	0.7505	8.1220	1234.07
bart	0.5176	0.5049	0.7517	0.7503	21.9079	3296.48
gpt2	0.3720	0.3646	0.4252	0.4241	3.5886	287.80
template	0.5174	0.5047	0.7520	0.7506	**0.2410**	**41.71**

Our findings indicate that while flan-t5-base and flan-t5-large produce similar quality outputs (as measured by BLEU and ROUGE scores), the base model offers significantly faster response times. The template-based approach, while

less flexible for complex queries, provides exceptional performance in terms of response speed. GPT-2 exhibited lower quality scores but maintained reasonable response times. BART achieved marginally better quality metrics but at a substantial computational cost, with response times nearly 10 times slower than flan-t5-base.

6 Conclusion and Future Work

This paper demonstrated a rapid prototyping framework to build intelligent and personalized chatbots using Retrieval-Augmented Generation. Our architecture enables modular integration of components, supporting experimentation and scalability. In future, we aim to integrate multilingual capabilities, explore fine-tuning FLAN-T5 on domain-specific support data, and implement a feedback-driven retraining loop to improve accuracy over time. Real-world deployment and A/B testing will provide deeper insights into long-term user satisfaction and system adaptability.

References

1. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., Kiela, D.: Retrieval-augmented generation for knowledge-intensive NLP tasks. In: Advances in Neural Information Processing Systems 33 (2020)
2. Chung, H.W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Webson, A.: Scaling instruction-finetuned language models. arXiv preprint arXiv:2210.11416 (2022)
3. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence embeddings using Siamese BERT-networks. arXiv preprint arXiv:1908.10084 (2019)
4. Chiang, W.L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J.E., Stoica, I., Xing, E.P.: Can foundation models wrangle your data? arXiv preprint arXiv:2205.09911 (2023)