



**RV College of
Engineering®**

Mysore Road, RV Vidyaniketan Post,
Bengaluru - 560059, Karnataka, India

Go, change the world®

**DEPARTMENT OF
ARTIFICIAL INTELLIGENCE AND MACHINE
LEARNING**



Project Report

On

Intelligent Customer Support Chatbot

***Submitted in partial fulfilment of the requirements for the V Semester
ARTIFICIAL NEURAL NETWORK AND DEEP LEARNING***

AI253AI

By

1RV22AI069	DHARSHINI MA
1RV22AI046	ROSHAN JOHN
1RV22AI048	SAFIYA FARHEEN

**Department of Artificial Intelligence and Machine Learning
RV College of Engineering®
Bengaluru – 560059**

Academic year 2024-25

RV COLLEGE OF ENGINEERING®

Autonomous Institution Affiliated to Visvesvaraya Technological University, Belagavi

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Bengaluru– 560059



CERTIFICATE

This is to certify that the project entitled “**Intelligent Customer Support Chatbot**” submitted in partial fulfillment of Artificial Neural Networks and Deep Learning (AI253IA) of V Semester BE is a result of the bonafide work carried out by Dharshini MA (1RV22AI069), Roshan John (1RV22AI046) and Safiya Farheen (1RV22AI048) during the Academic year 2024-25

Faculty In charge:

Date :

Head of Department:

Date:

RV COLLEGE OF ENGINEERING®

Autonomous Institution Affiliated to Visvesvaraya Technological University, Belagavi

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Bengaluru– 560059

DECLARATION

We, Dharshini MA (1RV22AI069), Roshan John (1RV22AI046) and Safiya Farheen (1RV22AI048) students of Fifth Semester BE hereby declare that the Project titled “**Intelligent Customer Support Chatbot**” has been carried out and completed successfully by us and is our original work.

Date of Submission:

Signature of the Student

ACKNOWLEDGEMENT

We are profoundly grateful to our guide, **Dr. Somesh Nandi**, Assistant Professor, RV College of Engineering, for his wholehearted support, valuable suggestions, and invaluable advice throughout the duration of our project. His guidance and encouragement were instrumental not only in the successful completion of the project but also in the preparation of this report.

We also extend our special thanks to **Dr. S Anupama Kumar**, Associate Professor, RV College of Engineering for her invaluable insights, support, and constructive feedback, which significantly contributed to the improvement of our work.

We would like to express our sincere thanks to our Head of the Department, **Dr. Satish Babu**, for his constant encouragement and for fostering an environment of innovation and learning that greatly aided our progress.

We extend our heartfelt gratitude to our beloved Principal, **Dr. K. N. Subramanya**, for his unwavering appreciation and support for this Experiential Learning Project, which motivated us to give our best.

Lastly, we take this opportunity to thank our family members and friends for their unconditional support and encouragement throughout the project. Their backup and motivation were crucial in helping us overcome challenges and successfully complete our work.

ABSTRACT

The customer support sector is critical for ensuring customer satisfaction and brand loyalty. However, traditional support methods face growing challenges in meeting customer expectations, especially with increasing interaction volumes and demand for real-time, personalized responses. Studies reveal that 75% of customers expect replies within 5 minutes, and 77% value consistent support, making scalable and intelligent solutions a necessity. This project presents an Intelligent Customer Support Chatbot leveraging advanced AI technologies to address these demands effectively.

The chatbot is built using transformer-based models such as Llama 3.2, ensuring human-like conversational abilities and context-aware interactions. By implementing Retrieval Augmented Generation (RAG) with FAISS (Facebook AI Similarity Search), the system efficiently retrieves relevant answers from a robust knowledge base. The chatbot automates handling common customer inquiries, including order tracking, product details, refunds, and complaint management, reducing reliance on human agents and enabling timely, accurate resolutions.

A user-friendly interface is developed using Streamlit, providing an interactive platform for both customers and support agents. The chatbot eliminates repetitive tasks, allowing human agents to focus on complex customer issues while maintaining high service standards.

The system's scalability ensures that it can handle increased interaction volumes efficiently, contributing to operational cost reduction. With over 60% of users comfortable interacting with AI for quick queries, this chatbot represents a sustainable and modern solution for customer service challenges. Future advancements include incorporating advanced sentiment analysis, expanding the knowledge base, and implementing proactive support features.

This project not only demonstrates the potential of AI in transforming customer support systems but also highlights the role of deep learning in building scalable and efficient tools that enhance both customer satisfaction and business outcomes.

Table of Contents

Contents	Page No
College Certificate	i
Undertaking by student	ii
Acknowledgement	iii
Abstract	iv
Table of Contents	v
List of Tables	vi
List of Figures	vii
Introduction	
1.1 Project Description	1
1.2 Report Organization	3
Literature Review	
2.1 Summary of Literature Survey	5
2.2 Existing and Proposed System	11
2.3 Tools and Technologies used	15
2.4 Hardware and Software Requirements	15
Software Requirement Specifications	
3.1 Introduction	18
3.2 General Description	18
3.3 Functional Requirement	21
3.4 Non-Functional Requirements	22
3.5 External Interfaces Requirements	23
3.6 Design Constraints	23
System Design	
4.1 Architectural Design of the Project	25
4.2 Data Flow Diagram	31
4.3 Description of the ANN/DL Architecture/Algorithm Used	35
Implementation	
5.1 Code Snippets	41
5.2 Results and Discussion with screen shots	50
Conclusion	53
Future Enhancements	54
Bibliography	55

CHAPTER 1: INTRODUCTION

1.1 Project Description

Customer support systems are crucial for businesses across industries, playing a vital role in maintaining customer satisfaction and brand loyalty. However, traditional customer support models face significant challenges, including managing high query volumes, ensuring quick and accurate responses, and providing personalized support. Studies show that 75% of customers expect a reply within five minutes, and 77% prioritize consistent support in their interactions. Rising interaction volumes, projected to increase by 30–50% annually, further strain existing systems, highlighting the need for scalable, intelligent solutions.

This project proposes an Intelligent Customer Support Chatbot that leverages Artificial Intelligence (AI) and Natural Language Processing (NLP) to address these challenges. The chatbot employs transformer-based models, specifically Llama 3.2, to provide human-like conversational abilities and context-aware interactions. Using Retrieval Augmented Generation (RAG) powered by FAISS (Facebook AI Similarity Search), the system efficiently retrieves relevant answers from a preloaded knowledge base. By automating routine tasks such as order tracking, refund inquiries, and FAQs, the chatbot reduces the reliance on human agents and enables faster query resolution.

The system features a user-friendly interface developed with Streamlit, ensuring accessibility and ease of interaction. By integrating these advanced AI techniques, the project aims to deliver personalized, scalable, and efficient customer support solutions, addressing the growing demands of modern businesses.

Theory and Concept

1. Natural Language Processing (NLP) Natural Language Processing is a core component of this project, enabling the chatbot to understand and interpret human language effectively. Using techniques such as tokenization, stemming, and lemmatization, the chatbot can break down user queries into meaningful components. Sentiment analysis and intent detection allow the chatbot to assess user emotions and accurately determine the purpose of the query. These NLP techniques ensure the chatbot provides relevant and context-aware responses, enhancing user satisfaction and engagement.

2. Transformer Models Transformers are state-of-the-art architectures for handling sequential data and are central to the chatbot's conversational capabilities. Specifically, this project utilizes the Llama 3.2 model, a powerful transformer-based architecture known for its bidirectional context understanding and efficient handling of complex queries. The self-attention mechanism in transformers allows the chatbot to focus on relevant parts of the input query while considering its surrounding context. This ensures high accuracy and coherence in responses, making interactions more human-like and engaging.

3. Retrieval Augmented Generation (RAG) RAG combines the strengths of retrieval-based systems and generative models to deliver precise and contextually relevant answers. Using FAISS (Facebook AI Similarity Search), the chatbot retrieves the most relevant information from its knowledge base, such as FAQs or product details. This retrieved information is then processed and refined by the generative model (Llama 3.2) to provide coherent and personalized responses. This approach ensures that the chatbot can handle both structured and unstructured queries efficiently.

4. Real-Time Query Handling and Automation The chatbot is designed to handle real-time customer interactions, automating routine tasks like FAQs, order tracking, and refund queries. By employing AI-driven decision-making, the chatbot ensures that simple queries are resolved immediately while complex issues are seamlessly escalated to human agents via the integrated ticket management system.

5. Data Storage and Management The chatbot accesses preloaded datasets, including FAQs, user interaction histories, and product details, ensuring that responses are tailored to the specific needs of users. The system is also designed to support scalability, allowing for future integration with dynamic data sources.

6. Interactive User Interface The user interface is developed using Streamlit, a lightweight and efficient framework for building web-based applications. The interface enables users to interact with the chatbot seamlessly, offering a visually appealing design and easy navigation. This ensures accessibility across devices, improving the overall user experience.

7. Transformer-Based Context Awareness Context awareness is critical in providing meaningful responses to multi-turn conversations. The transformer model processes previous user interactions to maintain the flow of conversation, ensuring that follow-up queries are handled accurately. This enhances the chatbot's ability to simulate natural human-like dialogue, fostering trust and engagement.

8. Scalability and Adaptability The architecture of the chatbot is designed to accommodate growing user interaction volumes without compromising performance. It can adapt to diverse industries by customizing the knowledge base and responses, making it versatile for a wide range of customer support applications.

1.2 Report Organization

The report is structured to provide a detailed overview of the Intelligent Customer Support Chatbot project. It begins with the **Introduction**, outlining the significance of customer support systems, current challenges, and the role of AI in addressing these issues. The **Project Description** elaborates on the scope, methodologies, and expected outcomes of the system, focusing on its integration of cutting-edge AI technologies.

The **Report Organization** section guides readers through the report's layout, ensuring clarity and logical progression. Following this, the **Literature Review** delves into previous research, current systems, and the proposed innovations, detailing the tools and technologies used, along with hardware and software requirements. The **Software Requirement Specifications** chapter details the system's functional and non-functional requirements, external interfaces, and design constraints. This is followed by the **System Design**, which includes architectural design, data flow diagrams, and an explanation of algorithms like self-attention and FAISS.

The **Implementation** section presents the core components of the chatbot, supported by relevant code snippets and performance metrics. Screenshots highlight the chatbot's interaction capabilities, emphasizing its efficiency in handling customer queries. The Results and Evaluation section demonstrates the chatbot's performance metrics, including accuracy and response time, showcasing its success in achieving project objectives.

The report concludes with a **Conclusion** summarizing the project's contributions to modern customer support systems. A **Future Enhancements** section explores potential upgrades, such as integrating voice-based support, expanding the knowledge base, and implementing proactive customer engagement features. Lastly, the **References** section cites all sources used, ensuring the report's comprehensiveness and academic integrity.

This chapter provides an overview of the **Intelligent Customer Support Chatbot** project, emphasizing its significance in modern customer service and the application of advanced technologies like Natural Language Processing (NLP) and transformer-based models for efficient query resolution. The chatbot aims to address growing demands for fast, accurate, and

personalized responses in customer support systems while overcoming challenges like increasing interaction volumes and operational inefficiencies.

The chapter also outlines the theory and concepts underpinning the project, including Retrieval Augmented Generation (RAG), transformer models, and real-time query automation, setting the stage for detailed discussions in subsequent sections. By combining cutting-edge AI techniques with user-friendly design, this project represents a transformative approach to redefining customer support solutions.

CHAPTER 2 : LITERATURE SURVEY

This section does a literature evaluation on intelligent customer support chatbots that discusses diverse applications modern-day machine cutting-edge and deep modern-day strategies to enhance consumer interplay and query resolution pastime and degree modern accuracy at real time carrier on a worldwide scale inside distinct sectors.

Literature Survey

The paper [1] with the aid of Maryam Abbasi , Marco V. Bernardo, Paulo Váz , José Silva and Pedro Martins presents a machine learning (ML) framework integrated with PostgreSQL to optimize query performance and manage workloads dynamically. It reduces query times by 42% and improves throughput by 74%, automating database tuning and adapting to workload changes. This approach minimizes manual administration while addressing tuning conflicts and ML performance challenges, providing a scalable solution for large-scale data management.

Paper [2] presents a neural network-based approach to natural language generation that incorporates contextual information from dialogue history to produce coherent and fluent responses. By leveraging LSTMs and rich input representations, it addresses the limitations of traditional rule-based and statistical methods. The study demonstrates improvements in response quality, setting a foundation for more advanced, adaptive dialogue systems. Its use of both automatic metrics and human evaluation provides a balanced assessment of the model's effectiveness.

The paper [3] presented by Surjandy, Cadelina Cassandra, discusses the influence of external factors, such as the COVID-19 pandemic, on the adoption and effectiveness of chatbots in e-commerce settings. The paper introduces the Q-factor model to assess the quality of chatbot interactions and their influence on customer perceptions. This model offers a unique perspective on evaluating chatbot effectiveness, focusing on reliability and system quality.

The work [4] by Jeff Johnson, Matthijs Douze offers means to enhance textual content summarization models by schooling them to fulfill a predictable output given noisy or perturbed inputs. The method aims to decorate the robustness and generalization of present day summarization fashions, addressing troubles like hallucination and sensitivity to moderate enter versions.

In the paper [5] of Hossain, Mostaqim, and Habib, Mainuddin, and Mubassir and Hassan (Mainuddin et al., which explores the possibility of the chatbot to be incorporated in various conversation channels social media, websites, and applications), they discuss the potentiality of the chatbot to be available everywhere so, potentially, it can make the product of information access more convenient for customers. It highlights the introduction of a novel real-time analytics application for the discovery of human interaction and for the collection of feedback in order to enable on-going improvement of current performance and consumer satisfaction. The authors discuss how current state-of-the-art requires the integration of protection and privacy features in the design in order to protect and comply with statistics protection regulations.

The article [6] by means of Manikanta, M. Rushi, J. & Lalitha, A. & Goud, B. Suresh, V., Daniya, Tyas describes a web-based framework of e-commerce system, which describes the contribution of the modern day in the mixing the chatbot in this framework and how through this addition of a new layer, accessibility and user pleasure across various devices and structures are enhanced, which is a different concern from previous works. The paper emphasizes a consumer-centric design approach, showcasing how consumer comments turned into incorporated into the improvement process to refine the chatbot's functionalities and interface talks approximately load coping with and green database control to handle high traffic.

In this paper [7] Santosh K, Maher, Suvarnsing G. Bhable, Ashish R. Lahase, Sunil S. Nimbhore discusses chatbots, which uses AI and deep learning, and those chatbots are changing industries like banking, health care and retail, by providing us interactive, green, user support. This paper studies chatbot applications, their technologies, languages, and features. Chatbots have emerged as indispensable tools for customer service, where they facilitate device interaction and query resolution in a wide variety of applications.

According to [8], using a better NLP approach, M. Rakhra highlights the importance of applying advanced NLP techniques to enable the chatbot to process and respond to patron's queries appropriately, which enhances user experience and satisfaction. The use of chatbot in different e-trade architectures that enable comfortable transactions and customer support. It also describes the layout of the user interface which are relevant for increasing usability.

The paper [9] on the same subject by Srinivasa Rao, Dr Dammavalam Srikanth, k. Pratyusha, J. Sucharitha, M. Tejaswini, M. Ashwini D. describe the specific structure of the latest the deep neural community (DNN)-based version of the chatbot (e.g., the number of latest layers, state-of-the-art activation functions, and optimization methods that have improved the version's performance in knowledge and in human-like response generation). Highlights the approaches

for statistics-augmentation and strategies required for acquiring a state-of-the-art, diverse, data set for better ability of neural network model to answer person queries effectively.

In [10] Ravi Kumar Chauhan, Manjot Arora, Yash Khadakban, Pranav Padmawar propose some of the metrics to keep track of the user's engagement with the chatbots among e-trade scenarios. those metrics offer insights into user conduct and preferences, allowing corporations to optimize their chatbot interactions and enhance consumer retention rates The metrics are consultation duration, message rely, user retention price, consumer satisfaction rating, drop modern rate, cause reputation accuracy, first reaction time, feedback and improvement recommendations.

The paper [11] by way of M.M. Khan highlights the application system-gaining knowledge statemodern algorithms, namely guide Vector Machines (SVM) for text-categorizing within the chatbot. This approach enables the chatbot to effectively capture and categorize human queries that in turn better enables the chatbot to provide useful answers and to thereby improve patron interactions—an unusual technical enhancement that until now has been left unexamined. In this regard, ultra-modern approaches to natural Language expertise (NLU) strategies are briefly described that will facilitate the chatbot's better understanding of cutting-edge consumer intents and context.

[12] Omar Khattab, Matei Zaharia on the paper colBERT approach an удивления novel rating model, uses contextualized late interaction across deep LLMs (predominantly, BERT) for green retrieval. with the aid of independently encoding queries and documents into one-grained representations that have interaction via reasonably-priced and pruning-pleasant computations, ColBERT can leverage the expressiveness state-of-the-art deep LMs whilst greatly dashing up question processing. Furthermore, this enables the implementation of ColBERT for cease-to-give up neural retrieval directly from a large file series.

The paper [13] by A Bidirectional LSTM model for Classifying Chatbot Messages explores the use of a Bidirectional long brief-time period reminiscence (BiLSTM) model for classifying chatbot messages into five goal training, specifically in the context today's expanded online verbal exchange throughout the COVID-19 pandemic. Applied to Thammasat university, the model achieved a contemporary accuracy of 80% on the validation dataset, which shows its relevance for knowledge user motive. In this study, the contribution of the modern-day natural language processing field is presented using the description of what modern-day state of the art new methods can allow chatbot functionality in a university context.

Paper [14] presents a neural network-based approach to natural language generation that incorporates contextual information from dialogue history to produce coherent and fluent responses. By leveraging LSTMs and rich input representations, it addresses the limitations of traditional rule-based and statistical methods. The study demonstrates improvements in response quality, setting a foundation for more advanced, adaptive dialogue systems. Its use of both automatic metrics and human evaluation provides a balanced assessment of the model's effectiveness.

The work [15] of Eleni Adamopoulou and Lefteris Moussiades in the paper analyzes the evolution, technologies, and state-of-the-art chatbots programmes, pointing out the degrees of maturity, core techniques (pattern matching and machine of the moment), and layout considerations. It covers industrial use cases, threats and countermeasures, while also proposing ideas on how to make chatbot intelligence more attractive.

As described in the paper [16] with Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, Wen-tau Yih, dense retrieval can readily outperform and ideally replace the pre-requisite sparse retrieval part in open-area question answering. Our empirical study and ablation study both suggest that more complex version frameworks or similarity features by themselves do not necessarily gain an additional value.

The paper [17] by Jeff Johnson, Matthijs Douze, Hervé Jégou is about enabling packages that previously required complex approximate algorithms. Eg, the methods presented here imply that it is now feasible to perform accurate ok-manner clustering or calculate the k-NN graph using straightforward brute-force approaches in significantly less time than a CPU (or a cluster of new them) could spend achieving that order of magnitude.

In [18] (by Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova), about current empirical improvements that these present switch state-of-the-art with language models can demonstrate that the large-scale, unsupervised preschooling of modern many-language knowledge structures is a critical contemporary feature. In particular, these findings enable low-aid commitments to have the advantage of deep unidirectional architectures even though they are low-aid obligations. Specifically, our main contribution is to generalize those results to deep bidirectional architectures, providing the same pre-trained model with the capability to efficiently perform a plethora of recent ultra-modern NLP tasks.

The paper [19] by Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin talks approximately the Transformer, the first

collection transduction version based totally totally on interest, replacing the recurrent layers most usually used in encoder-decoder architectures with multi-headed self-interest. With respect to translation constraints, the Transformer can be trained much faster than architectures that are based on recurrent or convolutional layers. We are pleased with the future of modern day interest-based models and intend to apply them to a wider range of tasks.

The paper [20] using the help of H. N. Io, C. B. Lee also describes how Chatbots have moved from rule-based systems to AI models that learn, more like humans. In this study, modern-day bibliometric analysis to uncover historical chatbot research and emerging areas of research gaps, namely in the era of deep learning. The review indicates that cutting-edge deep will direct the future development of the brand new chatbot, and it also presents clues for the future research.

Summary of the Literature Survey:

The following are the observations from the literature survey on using Bidirectional LSTM and sentiment analysis for an e-commerce chatbot application:

1. **Integration of Machine Learning for Optimization:** Papers like [1] and [14] emphasize the integration of machine learning with database systems to optimize query performance and manage data efficiently, ensuring real-time processing and scalability, which are crucial for e-commerce applications.
2. **Context-Aware NLP Models:** Research from [2] and [12] highlights the significance of leveraging context through memory networks, transformer-based architectures, and Bidirectional LSTMs for enhancing dialogue coherence and intent understanding. These advancements are pivotal for creating personalized and robust e-commerce chatbots.
3. **Deep Learning for Intent Classification:** Studies such as [13] and [9] focus on Bidirectional LSTM and deep neural network (DNN) architectures for understanding user intents and generating accurate responses. The use of techniques like diverse datasets and data augmentation improves the chatbot's adaptability to varied user queries.
4. **Sentiment-Aware Chatbots:** Paper [3] explores external factors like user perception during the COVID-19 pandemic, emphasizing the importance of sentiment-aware systems in e-commerce. Incorporating sentiment analysis ensures better customer interaction and satisfaction.

5. **Scalability and Multichannel Integration:** Works like [5], [6], and [8] discuss integrating chatbots across platforms (social media, websites, and apps) and the importance of scalable architectures to handle high traffic. Real-time analytics and user feedback enable continuous performance improvements.
6. **Technological Innovations:** Papers [18] and [19] discuss advancements in attention-based models like transformers and Bidirectional architectures, enhancing language understanding and performance in chatbot applications.
7. **Evaluation Metrics:** Paper [10] introduces comprehensive metrics like intent recognition accuracy, response time, and user satisfaction, which provide insights into optimizing chatbot performance and improving user engagement.
8. **Security and Privacy:** Papers like [5] and [11] highlight the necessity of robust data security and compliance with regulations, ensuring user trust and confidentiality in e-commerce chatbots.

Identified Gaps:

1. **Contextual Ambiguity Resolution:** Limited focus on resolving ambiguous user queries in real-time through advanced NLP techniques.
2. **Personalization Strategies:** Few studies integrate sentiment-aware systems with personalization features tailored for individual user preferences.
3. **Comparative Analysis:** Limited research comparing architectures like Bidirectional LSTM, transformers, and attention mechanisms for chatbot implementations in the e-commerce domain.
4. **Multilingual Support:** A lack of multilingual sentiment analysis capabilities and domain-specific training datasets.
5. **Real-Time Scalability:** Challenges in ensuring real-time query resolution and system scalability under heavy user loads.
6. **User Feedback Integration:** While some studies emphasize user feedback, further research is needed to incorporate this feedback into refining chatbot functionalities dynamically.

Objectives

1. **Improve User Intent Understanding:** Create and deploy a bidirectional LSTM-based architecture to help the chatbot better comprehend user context and intent, allowing for more precise and pertinent responses.

2. **Incorporate Sentiment Analysis for Personalized Interaction:** Enhance user happiness and engagement by utilizing sentiment analysis techniques to analyze client sentiments and customize responses accordingly.
3. **Achieve Real-Time Performance:** Tune the chatbot to handle queries and generate responses in real-time, guaranteeing smooth user experiences even in situations with significant traffic.
4. **Assure Multichannel Accessibility:** To improve accessibility and usability for e-commerce clients, integrate the chatbot across several communication channels, including social media, websites, and mobile apps.
5. **Pay Attention to Data Security and Privacy:** To foster user trust and guarantee the safe management of sensitive consumer data, put strong data security measures in place and abide by data protection laws.

2.2 Existing and Proposed system

Existing System:

1. Problem Statement: Modern businesses are facing significant challenges in managing customer interactions due to the limitations of traditional systems. These systems often rely heavily on manual processes or rule-based chatbots that lack dynamic interaction capabilities. While rule-based chatbots can handle predefined queries, they fail to address complex, nuanced, or evolving customer needs. The major limitations of these systems include:

1. **Inability to Process Unstructured Data:** Traditional systems struggle to process and analyze unstructured data, such as chat logs, which often contain valuable customer insights.
2. **Lack of Personalization:** These systems fail to provide personalized responses based on user history, preferences, or sentiment, leading to generic interactions that do not resonate with customers.
3. **Delayed Response Times:** With growing query volumes, traditional systems are unable to respond to customer inquiries in real time, resulting in delays and dissatisfaction.
4. **Scalability Issues:** These systems are not designed to handle high traffic or adapt to the exponential growth in customer interactions, particularly during peak times.
5. **Inconsistent Customer Experience:** Manual processes or rigid rule-based systems cannot ensure consistent quality in responses across different queries or channels.

6. **Loss of Brand Loyalty:** Inefficiencies in handling customer interactions lead to frustration, operational delays, and a negative perception of the brand, ultimately resulting in reduced customer retention.
7. Overall, traditional systems lack the flexibility, efficiency, and intelligence required to meet the growing demands of modern businesses and customers.

Proposed System:

Problem Statement and Scope of the Project

The objective of this project is to design and develop an intelligent chatbot for e-commerce applications that addresses the limitations of traditional systems. By leveraging Bidirectional Long Short-Term Memory (BiLSTM) models and sentiment analysis techniques, the proposed system will enhance the efficiency and quality of customer interactions.

Key Features of the Proposed System:

1. **Dynamic Intent Recognition:** The chatbot will utilize a BiLSTM-based architecture to process customer queries, ensuring accurate recognition of user intent. This deep learning approach will allow the system to understand contextual nuances, even in complex or ambiguous queries, surpassing the limitations of rule-based systems.
2. **Sentiment Analysis Integration:** By incorporating sentiment analysis, the chatbot will analyze customer emotions and moods in real time. This feature will enable the system to tailor responses based on the user's emotional state, fostering more personalized and empathetic interactions.
3. **Real-Time Query Handling:** The system will ensure real-time response generation, even under high query volumes, by using optimized NLP techniques and scalable infrastructure. This capability will reduce delays and improve customer satisfaction.
4. **Multichannel Accessibility:** The chatbot will be integrated across multiple platforms, including social media, websites, and mobile apps, ensuring a seamless and consistent customer experience regardless of the communication channel.
5. **Enhanced Scalability:** The system will be designed to scale efficiently with the growing number of customer interactions. Advanced load-balancing mechanisms and efficient database management will ensure uninterrupted service during peak traffic periods.

6. **Security and Privacy Compliance:** Robust security measures will be implemented to protect user data. The system will comply with relevant data protection regulations, such as GDPR, to ensure customer trust and confidentiality.
7. **Data-Driven Improvements:** The chatbot will use real-time analytics and user feedback to continuously improve its performance. Metrics such as intent recognition accuracy, response time, and user satisfaction scores will be used to optimize its functionality over time.
8. **Seamless E-Commerce Integration:** The chatbot will integrate seamlessly with e-commerce platforms, facilitating transactions, managing order inquiries, and assisting with product recommendations.

Scope of the Project

1. **Personalized Customer Experience:** The chatbot will provide tailored responses based on user history, sentiment, and preferences, enhancing customer engagement and satisfaction.
2. **Operational Efficiency:** By automating customer interactions, the system will reduce the workload on support teams, allowing them to focus on complex or high-value tasks.
3. **Brand Loyalty and Retention:** A consistent, personalized, and efficient customer experience will help businesses build stronger relationships with customers, leading to increased brand loyalty and retention rates.
4. **Competitive Edge:** The advanced features of the chatbot will enable businesses to differentiate themselves in the competitive e-commerce landscape, positioning them as customer-centric and technologically advanced.

The proposed system is expected to transform how businesses manage customer interactions, addressing the shortcomings of traditional systems while providing a scalable, efficient, and intelligent solution tailored for modern e-commerce applications.

Methodology Adopted in the Proposed System

The project employs a structured methodology comprising three key modules:

1. **Data Collection and Preprocessing:**
 - a. A curated dataset of customer queries and responses from e-commerce interactions is used. Datasets include FAQ data, customer reviews, and real-

world complaints from platforms such as social media, websites, and mobile apps.

- b. Preprocessing steps include cleaning unstructured text data, tokenization, removing stop words, and applying stemming/lemmatization. Word embeddings are created using pre-trained models like Word2Vec and GloVe to capture semantic meaning.

2. Implementation of Deep Learning Algorithm:

- a. A Bidirectional Long Short-Term Memory (BiLSTM) architecture is implemented for intent classification and sentiment analysis.
- b. Pre-Trained word embeddings (from Word2Vec or GloVe) are used in the embedding layer, while the BiLSTM model processes sequences in both forward and backward directions for enhanced context understanding.
- c. A sentiment analysis module is integrated to assess the emotional tone of customer queries and adjust responses accordingly.
- d. Transfer learning and hyperparameter tuning are utilized to optimize performance and reduce training time.

3. Testing and Validation:

- a. The trained chatbot model is tested on unseen datasets to evaluate its ability to generalize.
- b. Performance metrics, including accuracy, F1-score, precision, recall, and response time, are computed to validate its reliability. Real-time response accuracy is also assessed using a streamlit-based interactive interface.

Technical Features of the Proposed System

- 1. Deep Learning Integration:** Utilizes a BiLSTM architecture for superior contextual understanding of user queries and accurate sentiment classification.
- 2. Sentiment Analysis Integration:** Employs sentiment analysis to deliver contextually appropriate and emotionally intelligent responses.
- 3. Transfer Learning:** Leverages pre-trained embeddings to enhance model performance while reducing training costs and computational overhead.
- 4. Real-Time Query Resolution:** Implements scalable backend systems to handle real-time interactions with minimal latency, even under high traffic loads.
- 5. Multichannel Integration:** Supports customer interactions across websites, mobile apps, and social media platforms, ensuring a consistent user experience.

6. **Secure Data Management:** Incorporates data security protocols to comply with regulations like GDPR, ensuring user data privacy and protection.
7. **Streamlit-Based User Interface:** Provides an interactive interface for testing the chatbot, showcasing predictions and insights in real time.

2.3 Tools and Technologies Used

1. Deep Learning Frameworks:

- a. TensorFlow and Keras: For building and fine-tuning the BiLSTM model and embedding layers.

2. Data Processing and Preprocessing:

- a. NLTK and Gensim: For tokenization, stop-word removal, and word embeddings.
- b. Pandas and NumPy: For dataset management and preprocessing tasks.

3. Development Environment:

- a. Streamlit: To develop an intuitive frontend for interacting with the chatbot.
- b. Google Colab and VS Code: For model development, debugging, and GPU-accelerated training.

4. Visualization Tools:

- a. Matplotlib and Streamlit: For visualizing model metrics and real-time chatbot responses.

5. Hardware Acceleration:

- a. NVIDIA GPUs: To speed up training and inference of the BiLSTM model.

This methodology ensures the development of a robust, scalable, and user-friendly chatbot capable of addressing e-commerce customer queries in real time while delivering personalized, sentiment-aware responses.

2.4 Hardware and Software Requirements

Hardware Requirements:

1. Processor:

- a. Minimum: Intel i5 or equivalent processor for basic functionality.
- b. Recommended: Intel i7 or higher for faster and more efficient processing, especially when handling large datasets or complex BiLSTM models.

2. CPU/GPU:

- a. Minimum: NVIDIA GTX 1050 Ti or equivalent for effective model training and inference.
- b. Recommended: NVIDIA RTX 2060 or higher (e.g., RTX 3060 or 3090) for faster deep learning model training and real-time chatbot responses.

3. RAM:

- a. Minimum: 8 GB to support essential processing tasks and training.
- b. Recommended: 16 GB or more for efficient handling of large datasets and ensuring smooth multitasking during machine learning operations.

4. Storage:

- a. Minimum: 256 GB SSD or equivalent storage for storing datasets and pre-trained models.
- b. Recommended: 512 GB SSD or higher for faster data access and processing.

Software Requirements:**1. Programming Language:**

- a. Python (Version 3.8 or higher), which supports advanced machine learning libraries and frameworks.

2. Libraries & Frameworks:

- a. TensorFlow (Version 2.x) or PyTorch (Version 1.10 or higher): For building and training the Bidirectional LSTM model.
- b. NLTK and Gensim: For natural language processing tasks such as tokenization, stop-word removal, and word embedding creation.
- c. Streamlit: For building an interactive web interface for real-time chatbot interactions.
- d. Scikit-learn: For evaluating model performance with metrics like precision, recall, and F1-score.
- e. FAISS and SentenceTransformer: For implementing efficient search and retrieval mechanisms in chatbot responses.

3. Integrated Development Environment (IDE):

- a. Jupyter Notebook: For experimentation, visualization, and iterative development.
- b. VS Code or PyCharm: For comprehensive code development and debugging.

4. Operating System:

- a. Windows 10/11, Linux (Ubuntu 20.04 or above), or macOS: Compatible platforms supporting all necessary tools and libraries.

5. Visualization Tools:

- a. Matplotlib and Streamlit: For plotting performance metrics and showcasing real-time responses.

6. Version Control:

- a. Git: For collaboration, version tracking, and repository management.

CHAPTER 3: SOFTWARE REQUIREMENT SPECIFICATIONS

This chapter introduces the definitions, acronyms, and abbreviations used in the report. Additionally, it provides a general description of the e-commerce chatbot product and outlines its functional, non-functional, and external interface requirements.

3.1 Introduction

Definitions:

1. BiLSTM (Bidirectional Long Short-Term Memory): A deep learning architecture capable of learning context from both past and future information, making it ideal for natural language understanding tasks like intent classification.
2. Sentiment Analysis: A natural language processing (NLP) technique that determines the emotional tone or sentiment of a text (e.g., positive, neutral, or negative).
3. FAISS (Facebook AI Similarity Search): A library for efficient similarity search and clustering of dense vectors, used for retrieving relevant responses in chatbot systems.
4. Streamlit: An open-source framework for building web-based applications, used to create an interactive chatbot interface.
5. NLP (Natural Language Processing): A field of artificial intelligence focused on enabling computers to understand, interpret, and respond to human language.
6. Word Embeddings: A numerical representation of words in a vector space, capturing semantic and syntactic relationships between them.

Acronyms:

1. BiLSTM: Bidirectional Long Short-Term Memory
2. NLP: Natural Language Processing
3. FAISS: Facebook AI Similarity Search
4. UI: User Interface
5. API: Application Programming Interface

3.2 General Description

Product Perspective The e-commerce chatbot is designed to provide intelligent, real-time support for customer queries in online shopping scenarios. By leveraging advanced NLP techniques and BiLSTM-based sentiment analysis, the chatbot will deliver personalized, context-aware responses to enhance user satisfaction and streamline customer service

operations. The chatbot is integrated across various platforms, such as websites, mobile apps, and social media, ensuring multichannel accessibility.

The primary stakeholders for the chatbot include online shoppers, customer support teams, and e-commerce business owners. The product aims to reduce response times, improve user engagement, and handle high volumes of queries effectively.

Product Functions

1. **Query Resolution:** The chatbot will process and respond to customer queries related to orders, shipping, payments, and product details in real time.
2. **Sentiment Analysis:** The system will analyze the sentiment of customer queries to deliver contextually appropriate responses, such as addressing complaints empathetically or celebrating positive feedback.
3. **Recommendation System:** Based on user interactions and sentiment, the chatbot will recommend products or solutions tailored to individual preferences.
4. **Escalation of Issues:** For unresolved queries, the chatbot will escalate the issue to human support agents, ensuring a seamless handoff.
5. **User-Friendly Interface:** An intuitive and visually appealing interface will be designed to simplify interactions for users with varying technical skills.

User Characteristics

Primary Users (E-commerce Customers):

1. Skill Level: Non-technical users with basic familiarity with online shopping.
2. Technical Needs: A chatbot that is simple to use, accessible across devices, and responsive.
3. Goal: To get quick and accurate answers to their queries.

Secondary Users (Customer Support Teams):

1. Skill Level: Technically proficient users who will monitor chatbot performance and handle escalated queries.
2. Technical Needs: Insights into query trends and chatbot logs for optimizing operations.
3. Goal: To reduce the workload by automating repetitive queries.

End Users (E-commerce Businesses):

1. **Skill Level:** Highly technical teams managing the chatbot's deployment and performance.
2. **Technical Needs:** Scalability, data analytics, and integration with existing e-commerce platforms.
3. **Goal:** To enhance customer satisfaction and boost business efficiency.

General Constraints

1. **Accuracy of Sentiment Analysis:** The accuracy of sentiment analysis may vary depending on the quality and diversity of the training dataset.
2. **Scalability:** The system must be able to handle high query volumes during peak shopping seasons without performance degradation.
3. **Multichannel Integration:** Seamless integration across multiple platforms requires extensive testing and compatibility adjustments.
4. **Real-Time Performance:** The chatbot's ability to deliver near-instant responses depends on the efficiency of the underlying model and hardware.
5. **Privacy and Compliance:** The system must comply with data protection regulations, such as GDPR, to ensure user privacy and secure data handling.

Assumptions and Dependencies

1. **Dataset Availability:** The chatbot assumes access to a large, high-quality dataset of e-commerce queries and responses for training and fine-tuning the model.
2. **User Access to Devices:** It is assumed that users have access to devices (e.g., smartphones, laptops) with internet connectivity to interact with the chatbot.
3. **Model Updates:** The system depends on periodic updates to the chatbot model based on new customer queries and emerging trends.
4. **Infrastructure Support:** The system relies on sufficient cloud infrastructure to handle storage, training, and deployment requirements.
5. **User Training:** Basic user training or tutorials will be provided to familiarize users with the chatbot's functionalities.

3.3 Functional Requirements

1. Introduction

The Intelligent Customer Support Chatbot leverages advanced Natural Language Processing (NLP) and transformer-based AI models to provide accurate and context-aware responses to customer queries. This chatbot is designed to handle diverse customer support scenarios, including order tracking, refund requests, product inquiries, and complaint registration, offering fast and personalized assistance. The system features a user-friendly interface that allows customers to interact seamlessly with the chatbot while integrating powerful backend tools for efficient query resolution.

2. Input

The system requires the following inputs:

1. **Customer Queries:** Text-based questions or concerns entered by users via the chatbot interface.
2. **Query Context:** Optional metadata like customer ID or previous interaction history, automatically fetched for personalizing responses.

3. Processing

The system performs the following key processes:

1. **Preprocessing:**
 - a. **Query Tokenization:** Splits customer input into tokens to extract intent and key details.
 - b. **Intent Detection:** Classifies the query intent (e.g., order tracking, refund request) using NLP techniques.
 - c. **Entity Recognition:** Identifies specific entities (e.g., order numbers, product names) to provide targeted responses.
2. **Retrieval and Response Generation:**
 - a. **Knowledge Base Retrieval:** Uses FAISS (Facebook AI Similarity Search) to retrieve relevant FAQs or information from the knowledge base.
 - b. **Response Generation:** Employs the transformer-based Llama 3.2 model for generating context-aware and human-like responses.

3. **Automated Task Execution:**

- a. Processes tasks such as order tracking or refund initiation by querying external systems like the database or payment gateway.
- b. Creates tickets for unresolved issues in the ticket management system.

4. **Post-Processing:**

- a. Refines responses into a customer-friendly format.
- b. Logs interactions for future improvements and customer support analytics.

4. **Output**

Response to Customer Query: The chatbot provides a detailed and contextually relevant reply to the user's query. By leveraging the transformer-based Llama 3.2 model, the chatbot processes the query in real-time, ensuring that the response is accurate and aligns with the user's intent. For example, if a customer inquires about order tracking, the chatbot retrieves the relevant information and presents it clearly. The output also includes polite and professional language to enhance the customer experience, making interactions seamless and human-like.

3.4 **Non-Functional Requirements**

1. **Performance**

The system should process and return responses within 2-3 seconds to ensure a seamless and efficient user experience. The chatbot should handle multiple simultaneous user requests without significant delays, providing timely and accurate responses in high-traffic scenarios.

2. **Reliability**

The system should be highly reliable, maintaining availability at all times with minimal downtime. In case of errors or service interruptions, the chatbot should recover swiftly to continue providing uninterrupted support to users. Automatic retries and fallback mechanisms should be in place.

3. **Usability**

The system should feature an intuitive and user-friendly interface, accessible to both customers and support agents. The chatbot's interaction flow should be simple to understand, with clear prompts and easy navigation, minimizing the need for user effort to accomplish tasks.

4. **Security**

The system should ensure secure data handling practices, including encryption of sensitive user data and secure transmission via HTTPS. User privacy should be prioritized, especially when dealing with personal information shared during conversations, and the system should comply with data protection regulations.

5. **Maintainability**

The system should have a modular architecture with clean, well-structured code. It should be easy to update or extend functionalities, including adding new knowledge base entries and integrating additional features. Comprehensive documentation should be available for both developers and users, enabling efficient long-term maintenance.

3.5 External Interfaces Requirements

Hardware Interface

The system requires the following hardware components for operation:

1. **Client Device:** A device such as a smartphone, tablet, or computer for users to interact with the chatbot. The device should have internet connectivity for smooth communication with the server.
2. **Server:** A server for hosting the chatbot, processing user requests, and running the transformer models.
3. **Storage Device:** Storage to save user interactions, knowledge base content, and chatbot configurations.

3.6 Design Constraints

1. **Standard Compliance**

- The system should comply with industry standards for AI and machine learning, ensuring adherence to best practices in model training, inference, and evaluation, particularly for natural language processing (NLP).
- The chatbot should meet security and privacy regulations such as GDPR for user data protection. User interactions and data should be handled securely, ensuring confidentiality.
- The model architecture (e.g., transformer models such as Llama 3.2) should be compatible with deep learning frameworks like TensorFlow or PyTorch and deployed using established frameworks for scalability and efficiency.

2. Hardware Limitations

- The system should be optimized to run on standard end-user devices, such as smartphones and low-powered laptops, without requiring excessive computational resources.
- The chatbot should operate efficiently on devices with limited memory or processing power, ensuring fast response times without significant resource consumption.
- If the system supports offline capabilities for users in areas with poor internet connectivity, lightweight models and cached responses should be implemented to provide basic functionality without needing a constant server connection.
- The system should be flexible enough to scale across various devices and environments, ensuring consistent performance regardless of the hardware specifications.

This chapter outlines the software requirements for an Intelligent Customer Support Chatbot using AI technologies, detailing functional, non-functional, and external interface requirements. It describes the system's design, key features, user characteristics, and necessary hardware and software components for efficient customer support.

CHAPTER 4 : SYSTEM DESIGN

The System Design of our Project gives an overview of the workflow architecture ,data flow diagrams of level 0 and 1. Additionally it describes the architecture of the Llama 3.2 and the Sentiment Analysis Model

4.1 Architectural Design of the Project

The architectural design of the Intelligent Customer Service System is divided into four modules: **Data Collection and Preprocessing**, **Implementation of Sentiment Analysis Model**, **Implementation of Retrieval Augmented Generation (RAG) Algorithm** and **Implementation of Language Model**. Below is a block diagram for the project.

Block Diagram

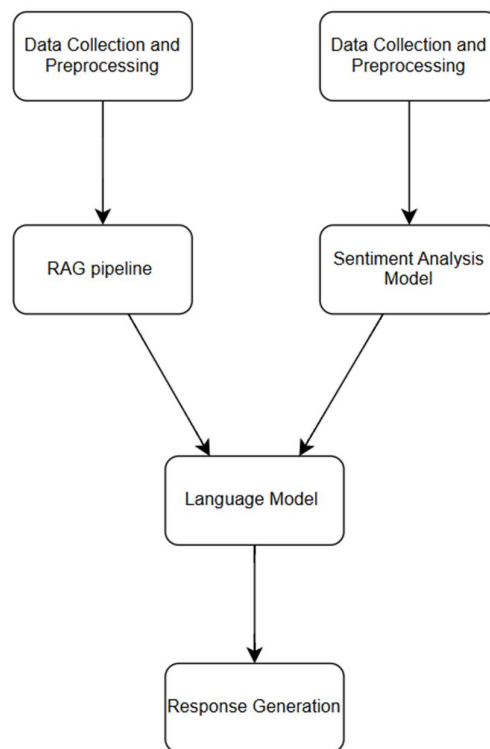


Figure 4.1 Block Diagram

1. Data Collection and Preprocessing:

For the RAG model we are using a corpus of Question-Answer pairs used to guide the Language Model to answer general questions in the ecommerce domain pertaining to orders and delivery issues (returning products, damaged products, etc.).

The second dataset is used to train the Sentiment Analysis Model. We have used the IMDB dataset containing 50000 reviews on movies. This dataset is widely used to train sentiment analysis models.

2. RAG Pipeline:

Retrieval-Augmented Generation (RAG) is implemented to enhance the chatbot's ability to provide accurate and context-aware responses. The process begins with preprocessing the knowledge base (e.g., FAQs or support documents) by cleaning the text, tokenizing, and generating embeddings using a Sentence Transformer model (e.g., all-MiniLM-L6-v2). These embeddings are indexed using FAISS for efficient similarity search. When a user submits a query, the system retrieves the top-k most relevant documents or answers from the indexed knowledge base. The retrieved content, along with the user query, is then passed to a generative model (e.g., Llama 3.2 via Ollama) to synthesize a coherent and contextually appropriate response.

3. Sentiment Analysis:

Sentiment analysis is integrated to gauge the emotional tone of user queries, particularly in complaints mode. User input undergoes preprocessing, including removing HTML tags, special characters, and stopwords, followed by tokenization and lowercasing. The cleaned text is converted into sequences of integers using a tokenizer and padded to a fixed length for consistency. These sequences are fed into a pre-trained LSTM model, which predicts whether the sentiment is positive or negative. Based on the sentiment score, the chatbot tailors its responses.

4. Language Model:

The Language Model (LLM) is implemented to generate contextually relevant and coherent responses for user queries. The system leverages Llama 3.2 via the Ollama API, which takes the user's input along with retrieved information from the knowledge base (using FAISS) as context. The model synthesizes this data to produce natural and accurate replies, ensuring the chatbot's responses are both informative and conversational.

Dataset description

The IMDB dataset consists of 50,000 movie reviews collected from the Internet Movie Database (IMDb), divided into 25,000 reviews for training and 25,000 for testing. Each review is labelled as either positive (indicating a rating of 7 or higher) or negative (indicating a rating of 4 or lower).

The text data includes raw movie reviews, which may contain HTML tags, special characters, and varying lengths of text.

```
# Visualize the postive and negative sentiments  
movie_reviews.groupby("sentiment").sentiment.count().plot.bar(ylim=0);
```

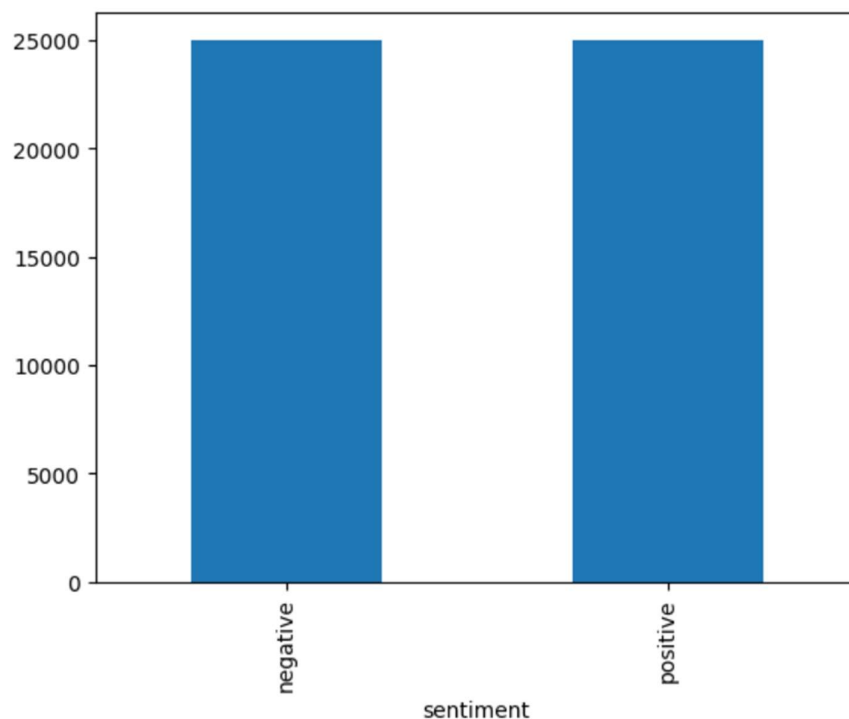


Fig. 4.2 Dataset Exploratory Data Analysis

Module Specification

1. Data Collection and Preprocessing

Input

The input to the preprocessing pipeline consists of raw user queries or movie reviews (in the case of the IMDB dataset). These inputs may contain noise such as HTML tags, special characters, punctuation, and stopwords. For example, a user query might look like:

"I hated this product! It's the worst!! "

Process

The preprocessing pipeline involves several steps to clean and prepare the text for the sentiment analysis model:

- Remove HTML Tags: Strip HTML tags using regex.
- Remove Special Characters: Eliminate non-alphanumeric characters and punctuation.
- Tokenization: Split the text into individual words or tokens.
- Lowercasing: Convert all tokens to lowercase for consistency.
- Stopword Removal: Remove common stopwords (e.g., "the", "and").
- Filter Short Words: Remove words with fewer than 2 characters.
- Convert to Sequences: Map tokens to integers using a tokenizer.
- Padding/Truncation: Ensure fixed-length sequences (e.g., 100 tokens) for model input.

Output

The output of the preprocessing pipeline is a cleaned and structured numerical representation of the input text, ready for the sentiment analysis model. For example, the raw input "I hated this product! It's the worst!!!" is transformed into a padded sequence of integers like: [12, 345, 67, 890, 0, 0, 0, ..., 0] This output is fed into the LSTM model for sentiment prediction.

2. RAG Implementation

Input

The input to the RAG pipeline consists of user queries submitted through the chatbot interface. These queries can range from general questions (e.g., "How do I track my order?") to specific complaints (e.g., "My order is delayed!"). The system also relies on a knowledge base (e.g., FAQs, support documents) that has been preprocessed and indexed for retrieval.

Process

The RAG pipeline involves the following steps:

1. Query Embedding: The user query is converted into a dense vector representation using a Sentence Transformer model (e.g., all-MiniLM-L6-v2).

2. Retrieval: The query embedding is used to search the FAISS index, which contains precomputed embeddings of the knowledge base. The system retrieves the top-k most relevant documents or answers.

3. Context Formation: The retrieved documents are combined with the user query to form a context for the generative model.

4. Response Generation: The context is passed to a Language Model (Llama 3.2 via Ollama), which synthesizes a coherent and contextually appropriate response.

Output

The output of the RAG pipeline is a natural and informative response tailored to the user's query. For example, if the user asks, "How do I track my order?", the system might respond: "You can track your order by logging into your account and visiting the 'Order History' section. If you need further assistance, please contact support." This response combines retrieved information from the knowledge base with the generative capabilities of the language model.

3. Sentiment Analysis

Input

The input to the sentiment analysis model consists of preprocessed user queries or movie reviews (from the IMDB dataset). These inputs are cleaned and transformed into fixed-length sequences of integers. For example, a user query like "I hated this product! It's the worst!! " is preprocessed into a sequence: [12, 345, 67, 890, 0, 0, 0, ..., 0].

Process

The sentiment analysis model is built using a Bidirectional LSTM architecture, which processes text data in both forward and backward directions to capture contextual information effectively. Here's how the model works:

i) Training Process:

The model is trained on the IMDB dataset, which contains 50,000 labeled movie reviews (25,000 for training and 25,000 for testing).

The training process involves:

Feeding preprocessed sequences into the model.

Using a binary cross-entropy loss function to measure prediction accuracy.

Optimizing the model using the Adam optimizer.

The model is trained for multiple epochs to minimize loss and improve accuracy.

ii) Model Architecture:

Embedding Layer: Converts integer-encoded words into dense vectors of fixed size (e.g., 100 dimensions).

Bidirectional LSTM Layer: Processes the sequence in both directions to capture long-term dependencies. It consists of 64 units with dropout (0.3) to prevent overfitting.

Dense Layer: A fully connected layer with a sigmoid activation function outputs a probability score between 0 (negative sentiment) and 1 (positive sentiment).

Output

The output of the sentiment analysis model is a sentiment score between 0 and 1, indicating the likelihood of the input text being positive or negative. For example:

A score close to 1 indicates positive sentiment (e.g., "I love this product!").

A score close to 0 indicates negative sentiment (e.g., "I hated this product!").

This score is used to tailor the chatbot's responses, ensuring empathetic and context-aware interactions.

4. Language Model

Input

The input to the Language Model consists of:

- **User Query:** The raw text input from the user (e.g., "How do I track my order?").
- **Retrieved Context:** The top-k most relevant documents or answers retrieved from the knowledge base using FAISS and the Sentence Transformer model.

Process

The Language Model (Llama 3.2 via Ollama) processes the input as follows:

Context Formation: The user query and retrieved documents are combined into a single context. For example:

1] User Query: "How do I track my order?"

2] Retrieved Context: "You can track your order by logging into your account and visiting the 'Order History' section."

3] Response Generation: The LLM processes the prompt and generates a natural, context-aware response.

Output

The output of the Language Model is a coherent and informative response tailored to the user's query. For example:

1] Input Query: "How do I track my order?"

2] Output Response: "You can track your order by logging into your account and visiting the 'Order History' section. If you need further assistance, please contact support."

This response combines the retrieved information with the generative capabilities of the LLM, ensuring accuracy and a natural conversational flow.

4.2 Data Flow Diagram

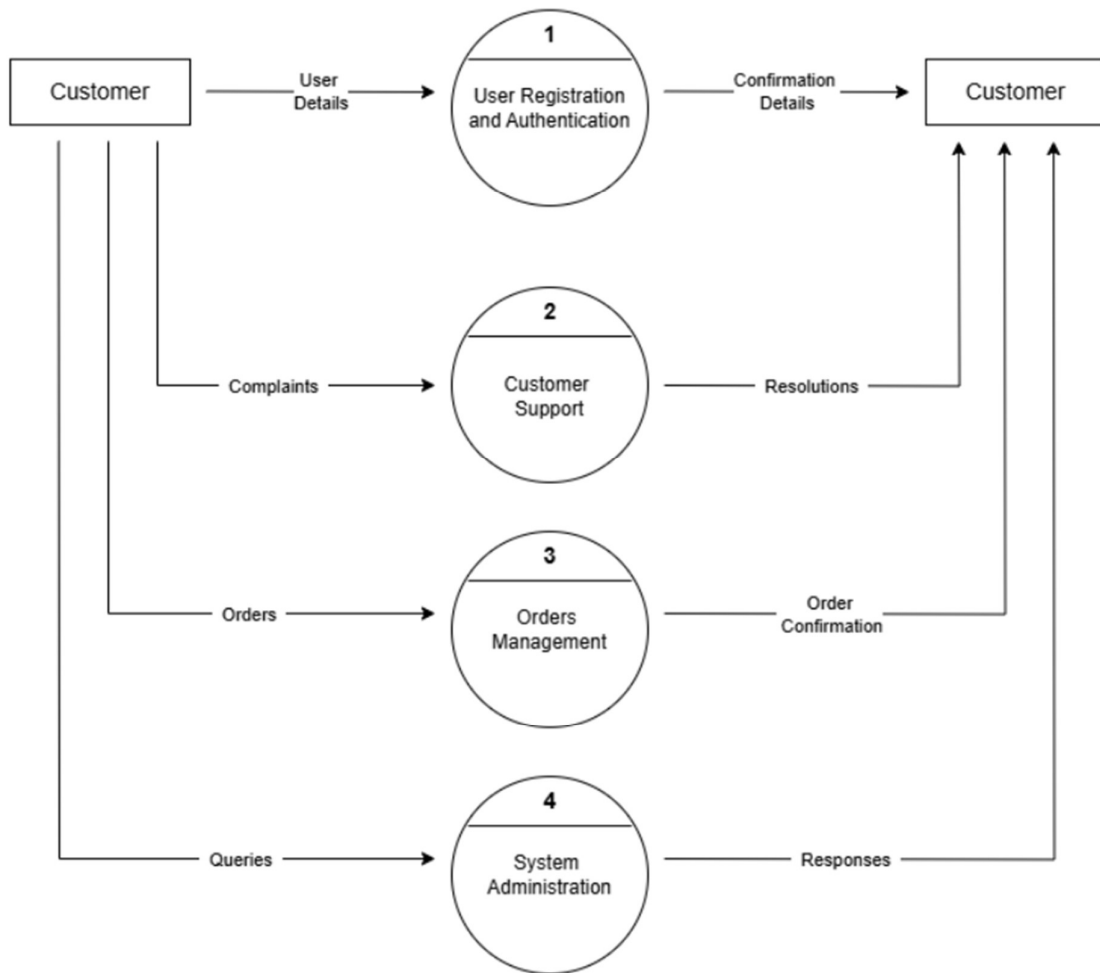
Level 0

LEVEL-0



Fig. 4.3 Data Flow Diagram Level 0

1. At this level, the system is represented as a single process, "Intelligent Customer Support System"
2. External entities:
 - User/Customer: Provides the query to the System in the form of a complaint or enquiry
 - System Output: Sends back the an response to resolve the complaint or provide information on the enquiry.
3. Data Flow:
 - Customer enters a query into the system
 - The system analyses the query and returns a response

Level 1**LEVEL-1***Fig. 4.4 Data Flow Diagram Level 1*

Level 1 Expands the single process into multiple subprocesses :

Subprocesses:

1. **User Registration and Authentication:** Handles the task of security and privacy by allowing access to authenticated users only
2. **Customer Support:** Resolves complaints on products and order tracking related issues.
3. **Order Management:** Tracks the status of a customer's orders and transactions made
4. **System Administration:** A dashboard of the chatbots metrics on how many issues are resolved and questions answered.

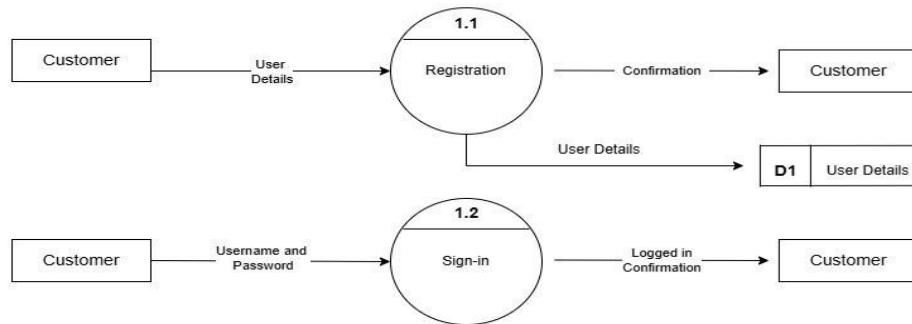
Dataflows:

- Inputs flow from the user to the system depending on the functionality used.
- For example, the registration and authentication module username and password is input, a complaint is filed from the customer and the details of the complaint are input etc.
- The system performs semantic analysis on the complaint to predict the complaints score on a scale of negative to positive.
- The systems output is always a coherent response in the form of text addressing the users issue.

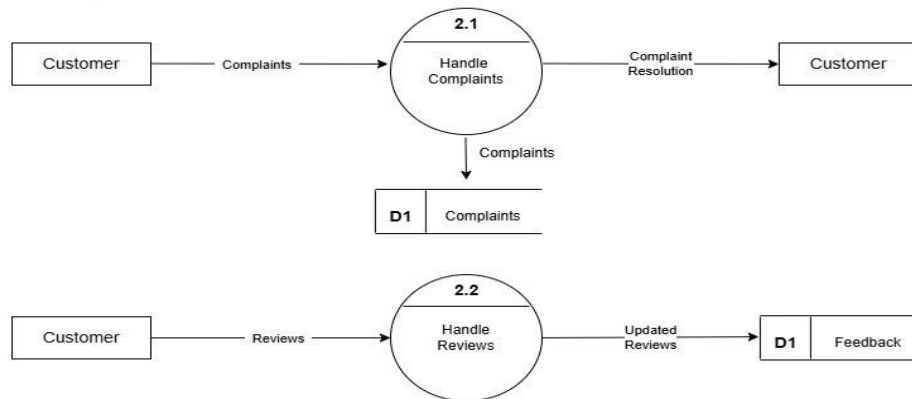
Level 2

LEVEL-2

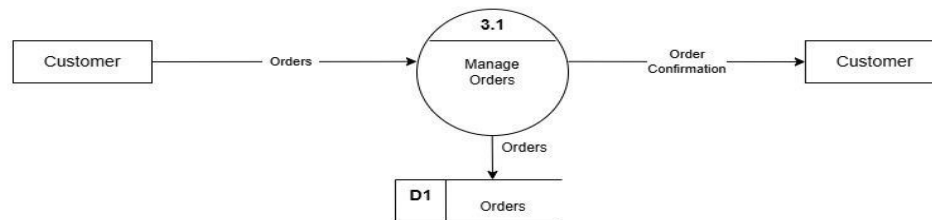
1. User Registration and Authentication



2. Customer Support



3. Order Management



4. System Administration

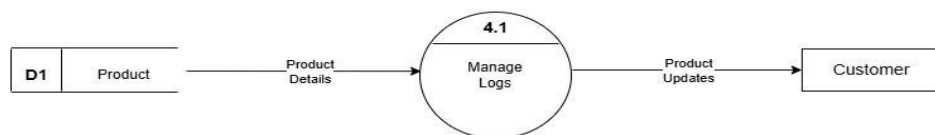


Fig. 4.5 Data Flow Diagram Level 2

Level 2 Expands the single processes in detail:

Subprocesses:

5. **User Registration and Authentication:** Handles the task of security and privacy by allowing access to authenticated users only

6. **Customer Support:** Resolves complaints on products and order tracking related issues.
7. **Order Management:** Tracks the status of a customer's orders and transactions made
8. **System Administration:** A dashboard of the chatbots metrics on how many issues are resolved and questions answered.

Dataflows:

- Inputs flow from the user to the system depending on the functionality used.
- For example, the registration and authentication module username and password is input, a complaint is filed from the customer and the details of the complaint are input etc.
- The system performs semantic analysis on the complaint to predict the complaints score on a scale of negative to positive.
- The systems output is always a coherent response in the form of text addressing the user issue.

4.3 Description of the Model Architectures

1. Semantic Analysis Model Architecture

The Semantic Analysis Model is a critical component of the chatbot system, designed to analyse and interpret the sentiment of user queries. This model employs a Bidirectional Long Short-Term Memory (LSTM) architecture, which is highly effective for processing sequential data such as text. The architecture is structured to capture contextual information from user inputs, enabling the chatbot to deliver accurate and context-aware responses. Below is a detailed description of the model's architecture and functionality.

1. Input Layer

The input to the model consists of pre-processed text data, represented as sequences of integers. Each word in the input text is tokenized and mapped to a unique integer using a tokenizer. The sequences are padded or truncated to a fixed length (e.g., 100 tokens) to ensure uniformity. For example, a user query such as "I loved this product!" is transformed into a sequence like [12, 345, 67, 890, 0, 0, 0, ..., 0].

2. Embedding Layer

The embedding layer converts the integer-encoded words into dense vectors of fixed size (e.g., 100 dimensions). This layer captures semantic relationships between words

by mapping them to a continuous vector space, enabling the model to understand the meaning and context of the input text. The output of this layer is a matrix of word embeddings with dimensions [sequence_length, embedding_dim].

3. Bidirectional LSTM Layer

The core of the model is a Bidirectional LSTM layer, which processes the input sequence in both forward and backward directions. This bidirectional approach allows the model to capture contextual information from the entire text, improving its ability to understand the sentiment of user queries. The layer consists of 64 LSTM units and incorporates a dropout rate of 0.3 to prevent overfitting. The output of this layer is a concatenated vector of hidden states from both directions, representing the contextual understanding of the input text.

4. Dense Layer

The final layer of the model is a fully connected Dense layer with a sigmoid activation function. This layer takes the concatenated hidden states from the Bidirectional LSTM and produces a single output value between 0 and 1. The output represents the probability of the input text having a positive sentiment (values close to 1) or negative sentiment (values close to 0).

Training Process

The model is trained on the IMDB Movie Reviews Dataset, which contains 50,000 labeled reviews (25,000 for training and 25,000 for testing). Each review is labeled as either positive (1) or negative (0). The training process involves the following steps:

Loss Function: The Binary Cross-Entropy Loss function is used to measure the difference between the predicted sentiment and the actual label.

Optimizer: The Adam optimizer is employed to minimize the loss function during training.

Training: The model is trained for 5 epochs with a batch size of 128. Dropout and recurrent dropout are applied to prevent overfitting and improve generalization.

Output

The output of the model is a sentiment score between 0 and 1, indicating the likelihood of the input text being positive or negative. For example:

A score close to 1 indicates positive sentiment (e.g., "I loved this product!").

A score close to 0 indicates negative sentiment (e.g., "I hated this product!").

2. Llama 3.2 Language Model Architecture

The Llama 3.2 model is a state-of-the-art Large Language Model (LLM) designed for natural language understanding and generation. It is based on the Transformer architecture, which relies on self-attention mechanisms to process and generate text. The model consists of multiple layers of **Transformer blocks**, each containing two key components: the **Multi-Head Self-Attention Mechanism** and the **Feed-Forward Neural Network**. The Multi-Head Self-Attention Mechanism allows the model to focus on different parts of the input sequence simultaneously, capturing long-range dependencies and contextual relationships between words. The Feed-Forward Neural Network processes the output of the attention mechanism, applying non-linear transformations to enhance the model's ability to understand complex patterns in the text. Llama 3.2 also incorporates positional encodings to account for the order of words in the input sequence, ensuring that the model retains information about the sequence structure. The model is pre-trained on a large corpus of text data using a masked language modeling objective, which enables it to generate coherent and contextually appropriate responses. In the chatbot system, Llama 3.2 is used to synthesize responses by combining the user's query with retrieved information from the knowledge base, ensuring that the generated responses are both accurate and natural-sounding.

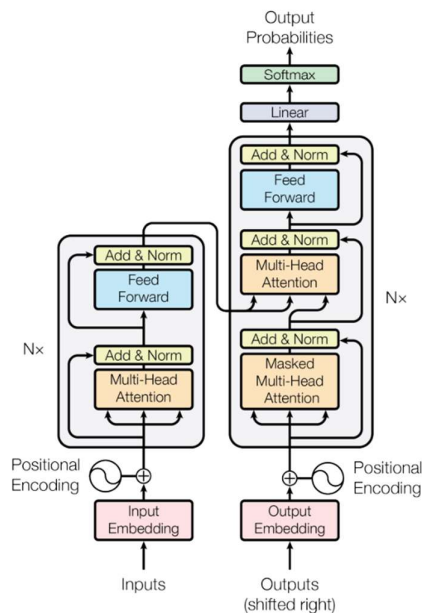


Figure 1: The Transformer - model architecture.

Fig. 4.6 Transformer Model Architecture[19]

3. RAG Pipeline

The Retrieval-Augmented Generation (RAG) pipeline combines the strengths of retrieval-based and generative models to provide accurate and context-aware responses. The pipeline begins with the retrieval phase, where the user's query is converted into a dense vector representation using a Sentence Transformer model (e.g., all-MiniLM-L6-v2). This embedding is used to search a FAISS index, which contains precomputed embeddings of the knowledge base (e.g., FAQs, support documents). The system retrieves the top-k most relevant documents or answers, ensuring that the retrieved information is highly relevant to the user's query. In the generation phase, the retrieved documents are combined with the user's query to form a context, which is passed to the Llama 3.2 model. The Llama 3.2 model processes this context using its Transformer-based architecture, generating a coherent and contextually appropriate response. The RAG pipeline ensures that the chatbot's responses are not only accurate but also natural and informative, leveraging both the precision of retrieval-based methods and the creativity of generative models. This hybrid approach enables the chatbot to handle a wide range of user queries effectively, delivering high-quality responses in real-time.

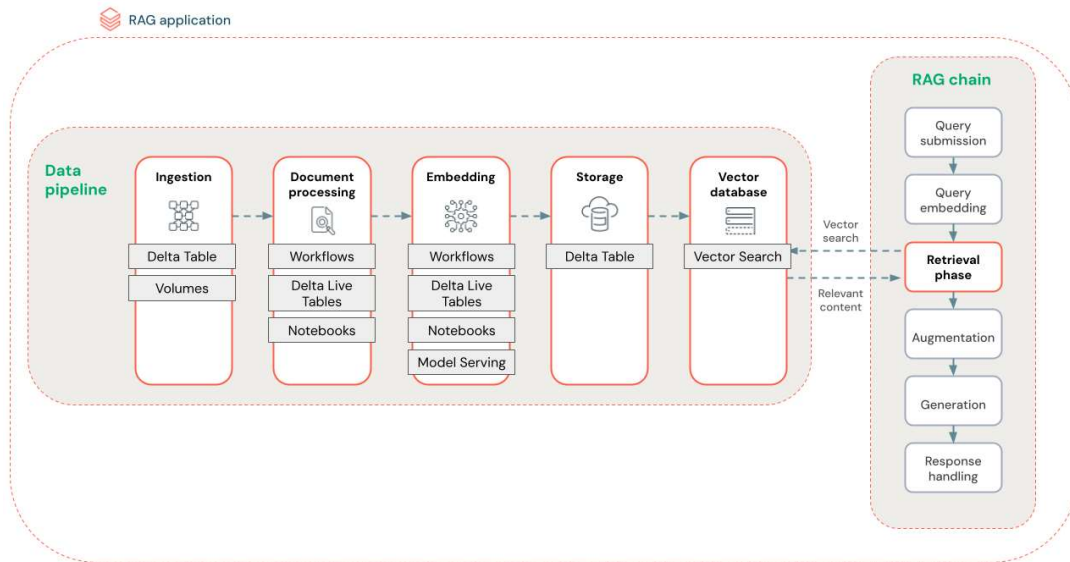


Fig. 4.7 RAG Architecture [20]

Benefits of this System:

1. Enhanced User Experience:

- The combination of sentiment analysis, retrieval-augmented generation, and advanced language modeling ensures the chatbot delivers personalized, accurate, and natural responses.

2. Context-Aware Interactions:

- The system understands the user's emotional tone (via sentiment analysis) and the context of their query (via RAG and Llama 3.2), enabling context-aware interactions.

3. Efficiency and Scalability:

- The architectures are designed for real-time performance and can scale to handle large volumes of queries and knowledge bases.

4. Versatility:

- The system can be adapted to various domains (e.g., customer support, e-commerce, healthcare) by updating the knowledge base and fine-tuning the models.

5. Reduced Hallucination:

- By grounding responses in retrieved documents, RAG reduces the risk of the language model generating incorrect or fabricated information.

6. Empathy and Personalization:

- Sentiment analysis allows the chatbot to respond empathetically to user complaints, improving user satisfaction and engagement.

Chapter 5: Implementation

The design and implementation involved the systematic development of an AI-based chatbot for intelligent customer support using transformer models like Llama 3.2 and Retrieval Augmented Generation (RAG). The code is organized into distinct sections, addressing tasks such as building the knowledge base, implementing the model, integrating FAISS for efficient query retrieval, and creating the user interface.

The code is written in VS Code in the format of a Python script where all the required libraries and dependencies, such as PyTorch, FAISS, and Streamlit, were installed.

5.1 Code Snippets

5.1.1 CHATBOT

```

1 # chatbot.py
2 import streamlit as st
3 import faiss
4 import numpy as np
5 import pandas as pd
6 from sentence_transformers import SentenceTransformer
7 import ollama
8
9 # Initialize ollama client
10 client = ollama.client()
11
12 # Load knowledge base and embeddings
13 @st.cache_data
14 def load_data():
15     embeddings = np.load("question_embeddings.npy") # Update with
16     knowledge_base = pd.read_csv("knowledge_base_text.csv") # Upda
17     return embeddings, knowledge_base
18
19 # Initialize FAISS index
20 @st.cache_resource
21 def initialize_faiss(embeddings):
22     dimension = embeddings.shape[1] # Embedding size
23     index = faiss.IndexFlatL2(dimension)
24     index.add(embeddings)
25     return index
26
27 # Load the model for query embedding
28 @st.cache_resource
29 def load_model():
30     return SentenceTransformer('all-MiniLM-L6-v2')
31
32 # Load Llama 3.2 model
33 @st.cache_resource
34 def load_ollama():
35     model = "llama3.2"
36     return model
37
38 # Refine response with llama 3.2
39 def refine_with_ollama(query, retrieved_answers, ollama_model):
40     # Format the context from retrieved answers
41     context = "\n".join([f"Q: {qa[0]}\nA: {qa[1]}" for qa in retrie
42
43     # Construct the Llama prompt
44     prompt = (
45         f"The following is a conversation with an ecommerce custome
46         f"The chatbot is helpful, polite, and provides accurate res
47         f"Context from the knowledge base:\n{context}\n\n"
48         f"User Query: {query}\n"
49         f"Chatbot Response:"
50     )
51
52     # Generate response using Ollama
53     response = client.generate(model=ollama_model, prompt=prompt)
54     response_text = response.get("response", "No response available
55     return response_text
56
57 # Retrieve answers and generate a comprehensive response
58 def retrieve_answer(query, index, knowledge_base, model, ollama_mod
59     # Retrieve top-k matches from FAISS
60     query_embedding = model.encode(query).reshape(1, -1)
61     distances, indices = index.search(query_embedding, top_k)
62
63     # Gather retrieved answers
64     retrieved_answers = []
65     for idx in indices[0]:
66         question = knowledge_base.iloc[idx]['Question']
67         answer = knowledge_base.iloc[idx]['Answer']
68         retrieved_answers.append((question, answer))
69
70     # Synthesize a comprehensive response
71     refined_response = refine_with_ollama(query, retrieved_answers,
72     return refined_response, retrieved_answers
73

```

Fig. 5.1 Loading Embeddings and Llama 3.2 Model

1. Initialization and Setup

- **Loading Data and Embeddings:** The `load_data` function loads pre-computed question embeddings and the knowledge base from files. These embeddings represent questions as numerical vectors for similarity comparison.
- **Initializing FAISS:** The `initialize_faiss` function sets up a FAISS index using the loaded embeddings, enabling fast and scalable retrieval of relevant answers.
- **Loading Models:** The `load_model` function loads a SentenceTransformer model for embedding new user queries. The `load_ollama` function specifies the Llama 3.2 model, which refines responses using retrieved knowledge base entries.

2. Query Processing and Answer Retrieval

- **Embedding the Query:** When a user submits a query, it is converted into an embedding vector using the SentenceTransformer model.
- **Retrieving Matches:** FAISS searches for the top `k` closest matches (default is 3) to the query embedding. For each match, the corresponding question-answer pair is extracted from the knowledge base.
- **Refining the Response:** The retrieved answers are refined using the Llama 3.2 model. A prompt is constructed, including the retrieved context, and the Llama model generates a comprehensive and polished response tailored to the user's query.

3. User Interface (UI)

The UI is designed using Streamlit and includes:

- **Sidebar Navigation:** A sidebar provides navigation options like "Ask a Question," "About," and "Help."
- **Conversation History:** Displays the last three user queries and chatbot responses for continuity.
- **Search and Complaints Modes:** Users can toggle between entering general queries and submitting complaints. The mode is reflected in the placeholder text and form behavior.
- **Response Display:** The chatbot's refined response is displayed prominently, followed by the retrieved matches styled for readability.

4. State Management

- **Session State:**

Tracks the user's conversation history to provide context for the ongoing interaction.

Maintains whether the user is in "Complaints Mode."

Handles user authentication and page navigation (e.g., logging out redirects the user to the login page).

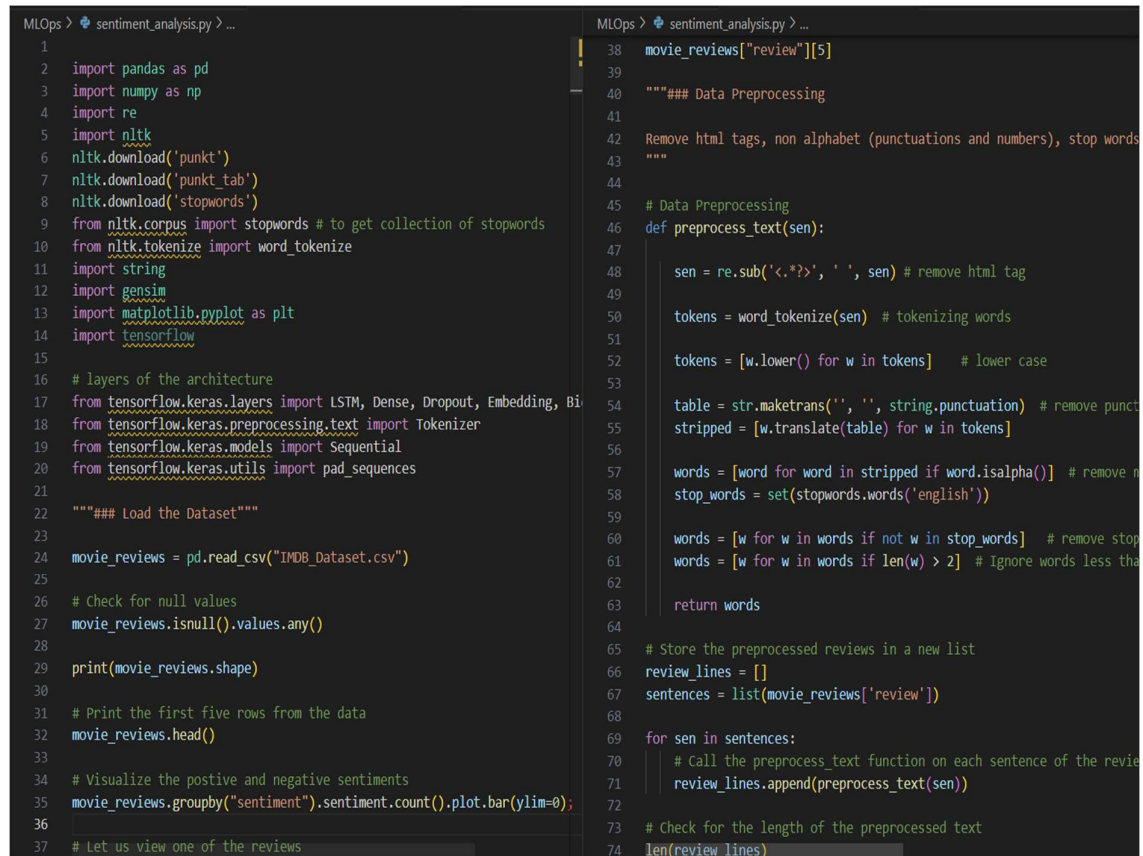
5. Additional Features

- **Enhanced UI Elements:** Styled cards display retrieved question-answer pairs for better readability.
- **Real-Time Feedback:** While processing a query, a loading spinner informs the user that the chatbot is retrieving answers.
- **Footer Section:** A footer provides branding information and credits the chatbot's technology.

6. Special Modes and Buttons

- **Complaints Button:** Toggles between general queries and complaints submission mode, updating the text input's placeholder dynamically.
- **Logout Button:** Clears the session state and redirects the user to the login page, ensuring secure logout.

5.1.2 SENTIMENT ANALYSIS



```

MLOps > sentiment_analysis.py > ...
1
2 import pandas as pd
3 import numpy as np
4 import re
5 import nltk
6 nltk.download('punkt')
7 nltk.download('punkt_tab')
8 nltk.download('stopwords')
9 from nltk.corpus import stopwords # to get collection of stopwords
10 from nltk.tokenize import word_tokenize
11 import string
12 import gensim
13 import matplotlib.pyplot as plt
14 import tensorflow
15
16 # layers of the architecture
17 from tensorflow.keras.layers import LSTM, Dense, Dropout, Embedding, Bi
18 from tensorflow.keras.preprocessing.text import Tokenizer
19 from tensorflow.keras.models import Sequential
20 from tensorflow.keras.utils import pad_sequences
21
22 """### Load the Dataset"""
23
24 movie_reviews = pd.read_csv("IMDB_Dataset.csv")
25
26 # Check for null values
27 movie_reviews.isnull().values.any()
28
29 print(movie_reviews.shape)
30
31 # Print the first five rows from the data
32 movie_reviews.head()
33
34 # Visualize the positive and negative sentiments
35 movie_reviews.groupby("sentiment").sentiment.count().plot.bar(ylim=0);
36
37 # Let us view one of the reviews
38
MLOps > sentiment_analysis.py > ...
38 movie_reviews["review"][5]
39
40 """### Data Preprocessing
41
42 Remove html tags, non alphabet (punctuations and numbers), stop words
43 """
44
45 # Data Preprocessing
46 def preprocess_text(sen):
47
48     sen = re.sub('<.??>', ' ', sen) # remove html tag
49
50     tokens = word_tokenize(sen) # tokenizing words
51
52     tokens = [w.lower() for w in tokens] # lower case
53
54     table = str.maketrans('', '', string.punctuation) # remove punct
55     stripped = [w.translate(table) for w in tokens]
56
57     words = [word for word in stripped if word.isalpha()] # remove n
58     stop_words = set(stopwords.words('english'))
59
60     words = [w for w in words if not w in stop_words] # remove stop
61     words = [w for w in words if len(w) > 2] # Ignore words less tha
62
63     return words
64
65 # Store the preprocessed reviews in a new list
66 review_lines = []
67 sentences = list(movie_reviews['review'])
68
69 for sen in sentences:
70     # Call the preprocess_text function on each sentence of the review
71     review_lines.append(preprocess_text(sen))
72
73 # Check for the length of the preprocessed text
74 len(review_lines)

```

Fig. 5.2 Training Sentiment Analysis Model

1. Import Necessary Libraries

- The required libraries are imported for tasks like data processing, visualization, natural language processing (NLP), and building the deep learning model.

2. Load Dataset

- The dataset is loaded using pandas into a DataFrame.
- The presence of null values is checked (**False** if none found).
- The shape (rows and columns) of the dataset is displayed, and the first few rows of the dataset are printed for a quick preview.

3. Data Visualization

Visualizes the count of positive and negative sentiments in the dataset as a bar plot. This provides a quick understanding of class distribution (positive vs negative reviews).

4. Preprocessing Function

- Purpose: Clean the review text by removing HTML tags, punctuation, stopwords, and words with less significance.
- Converts the text to lowercase and tokenizes it into words for further processing.

5. Preprocess All Reviews

- Each review in the dataset is preprocessed using **preprocess_text** and stored in **review_lines**.
- The length of the preprocessed list and a sample preprocessed review are printed.

6. Convert Sentiment to Binary

Converts the sentiment column (positive/negative) into binary format:

- 1 for positive
- 0 for negative

7. Train Word2Vec Model

A Word2Vec model is trained on the preprocessed reviews (review_lines).

Parameters:

- vector_size: Size of word vectors (100 dimensions).
- window: Context size for words (5 words on either side).
- min_count: Minimum word frequency to include in the vocabulary.

The vocabulary size is printed, showing how many unique words are learned.

8. Save Word Embeddings

The trained Word2Vec embeddings are saved in a text file for later use.

9. Load Pretrained Embeddings

- Loads the saved Word2Vec embeddings into a dictionary (**embeddings_index**).
 - Key: Word
 - Value: Corresponding word vector (numeric representation).

10. Tokenize and Pad Sequences

- The reviews are tokenized (words converted to integers).
- Reviews are padded/truncated to a fixed length (**max_length = 100**).
- Result:
 - **review_pad**: Padded sequences of tokenized reviews.
 - **sentiment**: Corresponding binary labels.

11. Create Embedding Matrix

Constructs the embedding matrix for the vocabulary:

- If a word has a pre-trained vector in the Word2Vec model, it's included in the embedding matrix.
- Words without pre-trained vectors are initialized with zeros.

12. Build LSTM Model

- Constructs an LSTM-based model for sentiment classification:
 - Embedding Layer: Uses the pre-trained embedding matrix. It's non-trainable to retain learned representations.
 - LSTM Layer: Captures sequential dependencies in the data.
 - Dropout (0.3) and recurrent dropout (0.2) prevent overfitting.
 - Dense Layer: A single neuron with a sigmoid activation outputs the probability of positive sentiment.

5.1.3 APP

```

MLOps > app.py > ...
1  import streamlit as st
2  import pandas as pd
3  from sklearn.feature_extraction.text import TfidfVectorizer
4  from sklearn.metrics.pairwise import cosine_similarity
5
6  # Load the FAQ dataset
7  faq_data = pd.read_csv("Ecommerce_FAQs.csv")
8
9  # Initialize TF-IDF Vectorizer
10 vectorizer = TfidfVectorizer(stop_words='english')
11 faq_vectors = vectorizer.fit_transform(faq_data['Question'])
12
13 def get_answer(user_query):
14     user_vector = vectorizer.transform([user_query])
15     similarity = cosine_similarity(user_vector, faq_vectors)
16     best_match_idx = similarity.argmax()
17     return faq_data.iloc[best_match_idx]['Answer']
18
19 # Streamlit UI
20 st.title("E-commerce Chatbot")
21
22 user_query = st.text_input("Ask a question about our services:")
23 if st.button("Submit Question"):
24     if user_query.strip() == "":
25         st.write("Please enter a question.")
26     else:
27         answer = get_answer(user_query)
28         st.write("**Chatbot:**", answer)
29
30 st.markdown("---")
31 st.markdown("**Powered by Streamlit and FAQ data**")
32

```

Fig. 5.3 Loading Knowledge Dataset

1. Loading the FAQ Dataset

- The FAQ data is loaded from a CSV file (**Ecommerce_FAQs.csv**). This dataset is expected to have at least two columns:
 - Question: Contains common customer questions.
 - Answer: Contains the corresponding answers to those questions.

2. Initializing the TF-IDF Vectorizer

- A **TF-IDF Vectorizer** is initialized to process and encode the questions in the dataset.
- This converts the text in the **Question** column into numerical feature vectors, which represent the importance of each word in the dataset while reducing the influence of common words.

3. Precomputing FAQ Vectors

- The vectorizer processes all the questions in the dataset and computes their **TF-IDF vectors**, storing them in **faq_vectors**.
- These vectors are used later for comparing user queries with the stored FAQ questions.

4. Defining the Answer Retrieval Function

- A function `get_answer(user_query)` takes the user's query as input.
- The query is converted into a TF-IDF vector using the pre-trained vectorizer.
- The similarity between the query vector and all the FAQ question vectors is calculated using **cosine similarity**.
- The FAQ question with the highest similarity score is selected as the best match.
- The corresponding answer from the dataset is returned.

5. Building the Streamlit Interface

The Streamlit interface provides a simple UI for the chatbot:

- **Title:** Displays "E-commerce Chatbot."
- **Text Input:** Allows the user to type a question about the services.
- **Submit Button:** Triggers the chatbot to process the query when clicked.
- **Validation:** If the user does not input any question, a prompt is shown to enter a question.
- **Answer Display:** The chatbot displays the most relevant answer based on the user's query.

6. Footer

- A footer is added using `st.markdown()` to indicate that the application is powered by Streamlit and FAQ data.

How It Works

- The user enters a question via the text input.
- When the "Submit Question" button is clicked:
 1. The user's query is matched against the preprocessed FAQ questions using TF-IDF and cosine similarity.
 2. The most relevant FAQ question is identified.
 3. The chatbot retrieves and displays the corresponding answer from the dataset.

This design provides a simple, efficient, and interactive chatbot for handling FAQs in an e-commerce context.

5.1.4 Knowledge Base

Question	Answer	question_embeddings
what is the return policy	you can return any item within 30 days of purchase provided it is in its original condition	[-0.035746458917856216, 0.0619087815284729, 0.030586473643779755, 0.0075577907264232635, ...]
how can i track my order	use the tracking id sent to your email to track your order on our websites tracking page	[0.012461827136576176, -0.057862644575834272, 0.0322352530226631, -0.008790426887571812, ...]
what are your customer support hours	we are available 24/7 via chat and email and from 9 am to 5 pm est via phone	[-0.09182575345039368, -0.001302109099245644, 0.0608113594353199, -0.007160990498960018, ...]
how do i contact customer support	you can contact us via email at support@example.com on our website or call 18001234567	[-0.0688387081027031, -0.005597930401563644, -0.0489486346080302, -0.03880351409310603, ...]
how long does shipping take	standard shipping takes 57 business days expedited shipping takes 23 business days	[0.006815953453655188, -0.012857609307088768, 0.0087979044765234, 0.0562408141899109, ...]
do you offer international shipping	yes we ship to most countries worldwide delivery times vary by destination	[-0.006965224165469408, -0.011596234515309334, 0.01688644476234913, -0.014064191840589046, ...]
can i cancel my order after placing it	orders can be canceled within 24 hours of placement by contacting customer support	[-0.021964574232697487, 0.054104669599527126, 0.0572864787990434, 0.02393381113238335, ...]
what payment methods do you accept	we accept credit cards paypal apple pay google pay and bank transfers	[0.01980834452980194, 0.018880199640989304, 0.031034953892230988, -0.05315718798929483, ...]
how can i update my delivery address	you can update your delivery address within 24 hours of placing the order by contacting support	[0.029113294556736946, -0.064360569190979, 0.036968689411878586, -0.018960191304945206, ...]
what should i do if i receive a damaged item	please contact support with a photo of the damaged item and well arrange a replacement	[-0.05276769750889197, 0.03600508928432465, 0.00174257624149323, -0.0391271416965463, ...]
what is the warranty on your products	most products come with a oneyear warranty check the product page for specific details	[-0.11231932789087296, 5.676430314453319e-05, 0.04730911917567523, -0.039146268677651215, ...]
can i get a refund instead of a replacement	yes refunds are available for eligible items returned within 30 days	[-0.09825780987739563, 0.009442925453186035, 0.0788099914789198, -0.009507197886505399, ...]
how do i reset my account password	use the forgot password link on the login page to reset your password	[-0.0024057117736110091, -0.04999804124236107, -0.04829425737261772, -0.009138557128608227, ...]
what if my package is delayed	please allow 48 extra hours for delivery if its still delayed contact our support team	[-0.03211820125579534, -0.06062602996862172, 0.02392590418457985, -0.007310842163680798, ...]
do you offer gift wrapping	yes gift wrapping is available for a small fee at checkout	[-0.1360420435667038, 0.07252057346078491, -0.0003754185813393593, 0.0146336826742363, ...]
how can i redeem my promo code	enter your promo code in the apply promo code field at checkout to receive your discount	[-0.10458390414714813, -0.011275055818259716, -0.0579006453990936, -0.034136928170919418, ...]
can i combine multiple promo codes	no only one promo code can be applied per order	[-0.06344765424728394, -0.034986693412065506, -0.0748467966914177, -0.014714114018678665, ...]
do you have a loyalty program	yes our loyalty program lets you earn points for every purchase which can be redeemed for rewards	[-0.06318090856075287, -0.05066574364900589, 0.019428979605436325, -0.05398984488580704, ...]
what is your exchange policy	items can be exchanged for a different size or color within 30 days subject to availability	[0.011327170766890409, -0.0026181994471699, -0.014148666292428097, -0.02885077704646453, ...]
how do i report a missing item in my order	contact support with your order id and details of the missing item and well resolve the issue	[0.027453213930130005, 0.06308311965041824, 0.02389920875430107, 0.053229851564664804, ...]
how do i request a product return	log into your account navigate to the orders section and select request return for the item	[-0.05224321630365736, -0.01062817291338038, -0.016353880986571312, -0.0689171092280208792, ...]
can i change my order after placing it	changes can only be made within 24 hours of placing the order contact support for assistance	[-0.011067152954638004, 0.00491597907911536, 0.03453489329958344, -0.00794803748233318, ...]
what are your shipping costs	standard shipping is 599 expedited shipping costs 1499 free shipping is available for orders over 50	[0.009282644838094711, -0.029413603246212006, -0.00765314931049943, 0.0485460091976166, ...]
do you offer bulk discounts	yes we offer discounts for bulk orders contact our sales team at sales@example.com for details	[-0.0710934007507324, 1.0507324077480007e-06, -0.026138978078961372, 0.01860117353498935, ...]
how do i unsubscribe from marketing emails	click the unsubscribe link at the bottom of any marketing email or update your preferences in your e	[-0.018602563068270683, -0.039240475405400902, -0.005708781071007252, 0.03688514977693558, ...]
what should i do if my product arrives damaged	if your product arrives damaged or defective please take clear photos of the product and its packaging	[-0.010710439644753933, -0.03460051864385605, 0.08426542580127716, -0.0594356954097478, ...]

Fig. 5.4 Knowledge Dataset

The knowledge base used in this project for the RAG (Retriever-Augmented Generation) workflow is a comprehensive repository of domain-specific and general-purpose information. It consists of structured and unstructured data sources, including documents, FAQs, articles, product descriptions, customer queries, and technical manuals. This repository serves as the foundational knowledge source for answering user queries and generating contextually relevant responses.

5.2 Results and Discussions

Customer Login / Sign Up

☒ Login
☐ Sign Up

Deploy

Login / Sign Up

Welcome to the Chatbot! Please log in or sign up to continue.

Select your role:

☒ Customer
☐ System Administrator

Customer Login

Username

Enter your username

Password

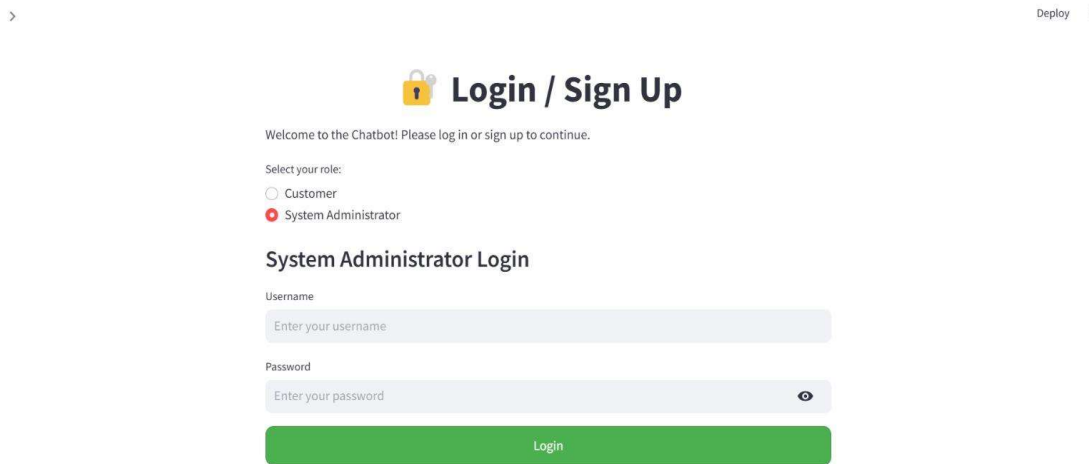
Enter your password

Login

Fig. 5.4 Customer Login Page

Customer Login Interface

This figure displays the login page where users can authenticate as either a Customer or a System Administrator. Customers gain access to chatbot support services, while system administrators manage and analyze chatbot performance.

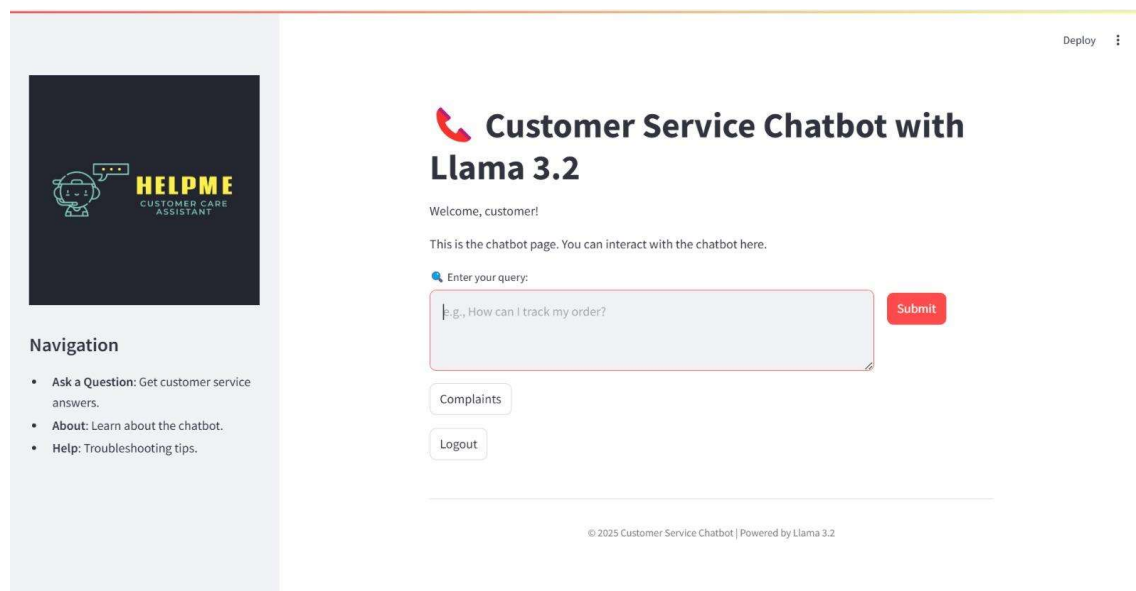


The screenshot shows a web interface for logging in as a system administrator. At the top right, there is a 'Deploy' button with a dropdown arrow. The main heading is 'Login / Sign Up' with a lock icon. Below it, a welcome message says 'Welcome to the Chatbot! Please log in or sign up to continue.' A section titled 'Select your role:' has two radio buttons: 'Customer' (unselected) and 'System Administrator' (selected). Below this is the 'System Administrator Login' section. It contains a 'Username' field with the placeholder 'Enter your username', a 'Password' field with the placeholder 'Enter your password' and an eye icon for toggling visibility, and a green 'Login' button at the bottom.

Fig. 5.5 System Administrator Login

System Administrator Login Interface

This screen allows administrators to log in. Admins can access analytics, monitor trends, and optimize chatbot performance based on user interactions.



The screenshot shows a web interface for a customer service chatbot. On the left is a sidebar with a dark blue header containing a chatbot icon and the text 'HELPME CUSTOMER CARE ASSISTANT'. Below the header is a 'Navigation' section with three links: 'Ask a Question: Get customer service answers.', 'About: Learn about the chatbot.', and 'Help: Troubleshooting tips.' The main content area has a red phone icon and the heading 'Customer Service Chatbot with Llama 3.2'. Below the heading, it says 'Welcome, customer!' and 'This is the chatbot page. You can interact with the chatbot here.' There is a text input field with the placeholder 'Enter your query:' and a red 'Submit' button. Below the input field are two buttons: 'Complaints' and 'Logout'. At the bottom, there is a footer that reads '© 2025 Customer Service Chatbot | Powered by Llama 3.2'.

Fig. 5.6 Chatbot

Customer Chatbot Interface

After logging in as a customer, users can interact with the chatbot powered by Llama 3.2. The chatbot assists with common customer service tasks such as order tracking, complaints, and FAQs, leveraging RAG, FAISS, and BiLSTM for precise responses.

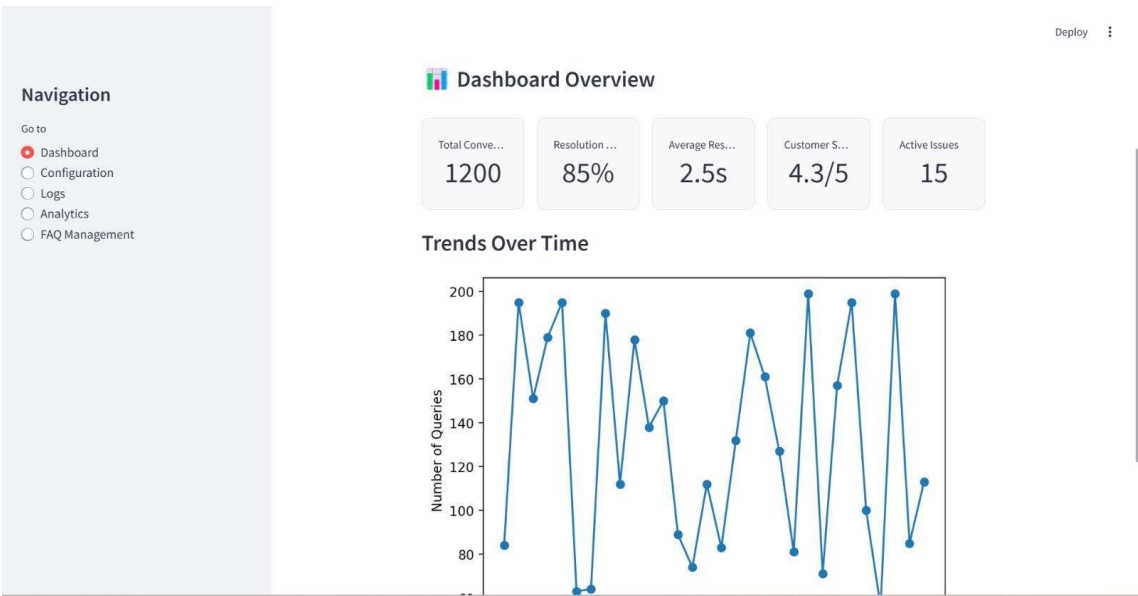


Fig. 5.7 Admin Dashboard

Admin Dashboard

The admin dashboard provides insights into chatbot performance, including user engagement trends, query resolution efficiency, and sentiment analysis metrics. This helps optimize responses and improve customer satisfaction.

Chapter 6: Conclusion

The Intelligent Customer Support Chatbot project demonstrates the transformative potential of AI in enhancing customer service operations. By leveraging advanced transformer-based models like Llama 3.2 and Retrieval Augmented Generation (RAG) with FAISS for efficient query retrieval, the system achieves rapid, context-aware responses for common customer inquiries. The integration of a user-friendly Streamlit interface ensures accessibility and ease of use, providing a seamless experience for both customers and support agents.

This system stands out for its ability to reduce human-agent reliance, streamline repetitive tasks, and handle high interaction volumes, contributing to operational efficiency and cost reduction. The chatbot's scalability ensures it can adapt to growing user demands, while its continuous learning capabilities allow it to improve over time.

The practical benefits are evident in improved customer satisfaction and faster response times, with the system capable of addressing inquiries related to order tracking, product details, refunds, and complaints. Looking ahead, the integration of advanced features like sentiment analysis and proactive support could further enhance its effectiveness.

In a broader context, this project highlights the growing role of AI in transforming customer service across industries. It sets the foundation for a more automated, efficient, and scalable approach to customer support, enabling businesses to provide consistent and high-quality service at scale. As AI technologies continue to evolve, the potential for chatbots to deliver even more personalized and intelligent support becomes increasingly promising. This project is a significant step toward the future of customer service, enhancing both customer experience and business operations.

Chapter 7: Future Enhancements

To further enhance the Intelligent Customer Support Chatbot, several key improvements can be implemented:

4. **Model Improvement:** Exploring more advanced transformer models like GPT-4 or T5 could improve performance, especially in handling complex queries and providing more contextually accurate responses. Fine-tuning pretrained models on domain-specific datasets would help the system better understand industry-specific terminology.
5. **Dataset Expansion and Augmentation:** Expanding the knowledge base to cover more topics and frequently asked questions would increase the system's ability to handle diverse customer inquiries. Incorporating data from various industries and feedback would improve generalization. Multilingual support would broaden the system's impact.
6. **Real-Time Performance and Optimization:** Optimizing the chatbot for mobile and edge devices is essential for real-time performance. Converting the model to lightweight frameworks like TensorFlow Lite or using model pruning techniques could reduce latency and resource consumption, ensuring faster response times.
7. **Deployment and User Interface:** Developing a mobile app or browser extension would allow users to interact seamlessly across platforms. Integrating with CRM systems would improve customer interaction tracking. Voice recognition features could enhance accessibility, particularly for users with disabilities.
8. **Cross-Platform Support:** Deploying the chatbot on the cloud would ensure scalability and availability, allowing businesses to offer consistent support globally. Multilingual support would also ensure the system is accessible to a broader audience across different regions.

These enhancements would improve the chatbot's accuracy, scalability, and user experience, making it a more effective and versatile tool for automating customer support.

Bibliography

- [1] M. Abbasi, M. V. Bernardo, Paulo Váz, J. Silva, and P. Martins, “Adaptive and Scalable Database Management with Machine Learning Integration: A PostgreSQL Case Study,” *Information*, vol. 15, no. 9, pp. 574–574, Sep. 2024, doi: <https://doi.org/10.3390/info15090574>.
- [2] Ondřej Dušek and Filip Jurčiček, “A Context-aware Natural Language Generator for Dialogue Systems,” *arXiv (Cornell University)*, Aug. 2016, doi: <https://doi.org/10.18653/v1/w16-3622>.
- [3] Surjandy and C. Cassandra, “The Effect of Using Chatbots at e-Commerce Services of Customer Satisfaction, Trust, and Loyalty,” Oct. 2023, doi: <https://doi.org/10.1109/ic3ina60834.2023.10285799>.
- [4] J. Liu, Q. Mao, B. Liu, H. Peng, H. Zhu, and J. Li, “Noised Consistency Training for Text Summarization,” *arXiv.org*, 2021. <https://arxiv.org/abs/2105.13635> (accessed Jan. 31, 2025).
- [5] M. Hossain, M. Habib, M. Hassan, F. Soroni, and M. M. Khan, “Research and Development of an E-commerce with Sales Chatbot,” *IEEE Xplore*, Jun. 01, 2022. <https://ieeexplore.ieee.org/document/9817272> (accessed Mar. 05, 2023).
- [6] M. S. Manikanta, J. Rushi, A. Lalitha, B. Shravan Kumar Goud, V. Suresh, and T. Daniya, “Web based E-commerce system integrated with Chatbot,” *International Journal of Research Publication and Reviews*, pp. 1655–1659, Apr. 2022, doi: <https://doi.org/10.55248/gengpi.2022.3.4.12>.
- [7] S. K. Maher, S. G. Bhable, A. R. Lahase, and S. S. Nimbhore, “AI and Deep Learning-driven Chatbots: A Comprehensive Analysis and Application Trends,” *IEEE Xplore*, May 01, 2022, doi: <https://ieeexplore.ieee.org/abstract/document/9788276>.
- [8] M. Rakhra *et al.*, “E-Commerce Assistance with a Smart Chatbot using Artificial Intelligence,” *2021 2nd International Conference on Intelligent Engineering and Management (ICIEM)*, Apr. 2021, doi: <https://doi.org/10.1109/iciem51511.2021.9445316>.
- [9] D. S. Rao, K. L. Srikanth, J. Noshitha Padma Pratyusha, M. Sucharitha, M. Tejaswini, and T. Ashwini, “Development of Artificial Intelligence Based Chatbot Using Deep Neural Network,” *Soft Computing Research Society eBooks*, pp. 143–151, Jan. 2021, doi: <https://doi.org/10.52458/978-93-91842-08-6-12>.
- [10] R. Chauhan, M. Arora, Y. Khadakban, P. Padmawar, and R. Kumar Chauhan, “CHATBOTS IN E-COMMERCE.” Accessed: Jan. 31, 2025.
- [11] M. M. Khan, “Development of An e-commerce Sales Chatbot,” *IEEE Xplore*, Dec. 01, 2020. <https://ieeexplore.ieee.org/document/9322667/>.

- [12] O. Khattab and M. Zaharia, “ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT,” *arXiv:2004.12832 [cs]*, Jun. 2020, Available: <https://arxiv.org/abs/2004.12832>
- [13] Nunthawat Lhasiw, Nuttapon Sanglerdsinlapachai, and Tanatorn Tanantong, “A Bidirectional LSTM Model for Classifying Chatbot Messages,” pp. 1–6, Dec. 2021, doi: <https://doi.org/10.1109/isai-nlp54397.2021.9678173>.
- [14] Artificial Intelligence Chatbots: A Survey of Classical versus Deep Machine Learning Techniques,” *Information Sciences Letters*, vol. 12, no. 4, pp. 1217–1233, Apr. 2023, doi: <https://doi.org/10.18576/isl/120437>.
- [15] E. Adamopoulou and L. Moussiades, “Chatbots: History, technology, and Applications,” *Machine Learning with Applications*, vol. 2, no. 100006, Dec. 2020, doi: <https://doi.org/10.1016/j.mlwa.2020.100006>.
- [16] V. Karpukhin *et al.*, “Dense Passage Retrieval for Open-Domain Question Answering,” *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, doi: <https://doi.org/10.18653/v1/2020.emnlp-main.550>.
- [17] J. Johnson, M. Douze, and H. Jegou, “Billion-scale similarity search with GPUs,” *IEEE Transactions on Big Data*, vol. 7, no. 3, pp. 1–1, 2019, doi: <https://doi.org/10.1109/tbdata.2019.2921572>.
- [18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *arXiv.org*, Oct. 11, 2018. <https://arxiv.org/abs/1810.04805>
- [19] A. Vaswani *et al.*, “Attention Is All You Need,” *arXiv*, Jun. 12, 2017. <https://arxiv.org/abs/1706.03762>
- [20] H. N. Io and C. B. Lee, “Chatbots and conversational agents: A bibliometric analysis,” *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, Dec. 2017, doi: <https://doi.org/10.1109/ieem.2017.8289883>.
- [21]<https://www.digitalocean.com/community/tutorials/bidirectional-rnn-kera>
- [22]<https://toloka.ai/blog/transformer-architecture/>
- [23]<https://medium.com/@infinitylearnings1201/databricks-genai-rag-in-detail-79da2299cb97>