**ENHANCING CUSTOMER EXPERIENCE USING AI**

**ABSTRACT**

In the era of e-commerce, enhancing customer experience is crucial for business growth. This project focuses on developing an AI-powered chatbot that assists customers in making informed purchasing decisions by answering queries related to products, comparing multiple products, and providing insights on ratings, prices, and specifications. The dataset used for training comprises 100,000 entries in JSON format and 20,000 in CSV format, sourced from Flipkart.

To build an efficient chatbot, the LLaMA 2-7B chat model was fine-tuned using Low-Rank Adaptation (LoRA) and 4-bit quantization techniques to optimize performance and memory usage. The training process resulted in a loss of 0.4, indicating a well-optimized model. Post-training, the model was converted from GGUF to PyTorch and then to ONNX format for seamless integration with OpenVINO, enabling accelerated inference and efficient deployment. Additionally, Streamlit was used to develop an intuitive and user-friendly web interface, allowing customers to interact with the chatbot effortlessly.

The chatbot successfully resolves customer queries in real-time, improving user engagement and simplifying the shopping experience. By leveraging advanced AI techniques and an interactive web interface, this system enhances customer satisfaction and streamlines e-commerce interactions.

**Table of Contents**

1. **Abstract**
2. **Introduction**
   * 2.1 Background and Motivation
   * 2.2 Objective of the Project
   * 2.3 Scope of the System
   * 2.4 System architecture
3. **Dataset Description**
   * 3.1 Source of Data (Flipkart)
   * 3.2 Data Formats (JSON & CSV)
   * 3.3 Preprocessing Steps
4. **Model Selection and Training**
   * 4.1 Choice of LLaMA 2-7B Chat Model
   * 4.2 Fine-Tuning with LoRA and 4-bit Quantization
   * 4.3 Training Process and Loss Optimization
5. **Model Conversion and Deployment**
   * 5.1 Conversion from GGUF to PyTorch and ONNX
   * 5.2 Integration with OpenVINO for Inference Optimization
6. **User Interface Development**
   * 6.1 Implementation Using Streamlit
   * 6.2 Features and Functionalities of the Chatbot UI
7. **Results and Evaluation**
   * 7.1 Model Performance (Training Loss: 0.4, Accuracy TBD)
   * 7.2 Response Accuracy and User Experience
8. **Challenges and Solutions**
   * 8.1 Model Optimization and Memory Constraints
   * 8.2 Improving Response Time and User Interaction
9. **Conclusion and Future Scope**
   * 9.1 Summary of Achievements
   * 9.2 Potential Improvements and Extensions

**2. Introduction**

**2.1 Background and Motivation**

In today’s digital age, online shopping has become a primary mode of purchasing products. E-commerce platforms like Flipkart host millions of products, making it challenging for customers to find the right product that meets their needs. Customers often have multiple queries regarding product specifications, prices, ratings, and comparisons before making a purchase. Traditional customer support systems are either time-consuming or lack the efficiency to handle large-scale queries effectively.

To address these challenges, artificial intelligence (AI)-powered chatbots have emerged as an innovative solution. By integrating natural language processing (NLP) and machine learning techniques, chatbots can assist users in real time, enhancing their overall shopping experience. This project aims to develop an AI-based chatbot that provides instant responses to customer queries, compares products, and offers insights into ratings and prices, thus streamlining the decision-making process for online shoppers.

**2.2 Objective of the Project**

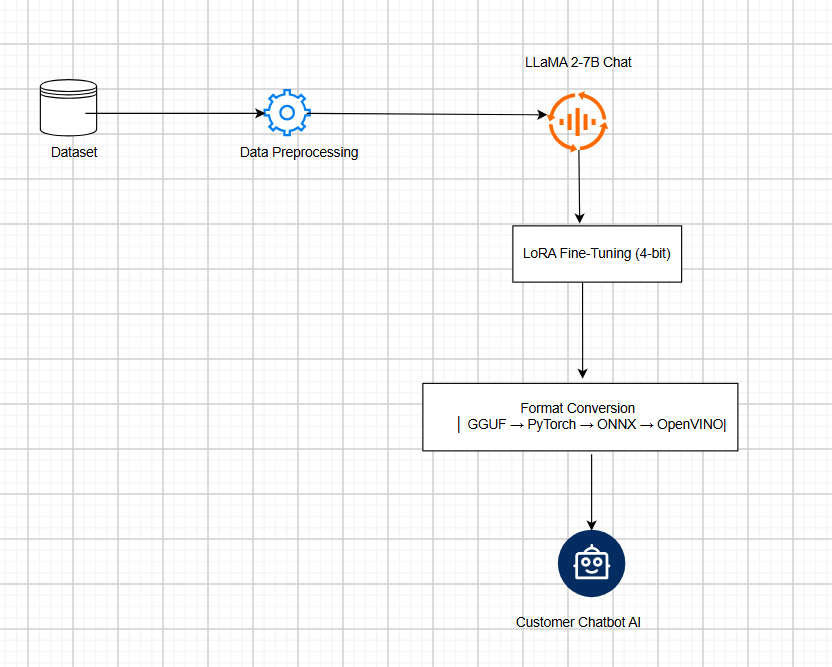
The primary objective of this project is to develop an AI-driven chatbot that enhances customer experience by providing quick and accurate responses to product-related queries. The chatbot is trained using a Flipkart dataset consisting of **100,000 entries in JSON format** and **20,000 entries in CSV format**. The key objectives of the project are:

* To train a chatbot using the LLaMA 2-7B model with Low-Rank Adaptation (LoRA) and 4-bit quantization for optimized performance.
* To enable customers to ask queries about products, including specifications, pricing, and ratings.
* To allow users to compare two products to help them make informed decisions.
* To fine-tune the chatbot to achieve a low training loss (0.4) and improve accuracy.
* To convert the model to GGUF, PyTorch, and ONNX formats for integration with OpenVINO, ensuring efficient deployment.
* To provide an interactive user-friendly interface using Streamlit, allowing customers to communicate with the chatbot seamlessly.

**2.3 Scope of the System**

This AI-powered chatbot is designed to function as an intelligent shopping assistant, specifically for e-commerce platforms. The scope of the project includes:

* **Data Processing:**
  + Handling large-scale structured and unstructured data from Flipkart.
  + Preprocessing data for improved model training and performance.
* **Model Development & Optimization:**
  + Fine-tuning the LLaMA 2-7B model using LoRA and 4-bit quantization.
  + Implementing efficient model conversion and deployment strategies using OpenVINO.
* **User Interaction & Features:**
  + Answering customer queries about product specifications, ratings, and pricing.
  + Comparing multiple products based on user input.
  + Providing real-time and accurate responses to enhance customer satisfaction.
  + Deploying an interactive chatbot interface using Streamlit, ensuring an easy and intuitive user experience.
* **Deployment & Integration:**
  + Deploying the chatbot with OpenVINO for optimized inference performance.
  + Ensuring compatibility with various platforms for potential e-commerce integration.
* **System Architecture**

****

**( Fig 1.1)**

**3. Dataset Description**

**3.1 Source of Data (Flipkart)**

The dataset used in this project was sourced from Flipkart and is publicly available on Kaggle under the Flipkart E-Commerce Dataset. It contains a wide range of product details, including product names, descriptions, prices, ratings, specifications, warranty information, and more. The dataset is extensive, covering various categories such as electronics, clothing, home decor, and accessories, making it ideal for training an AI-powered chatbot to assist customers with product-related queries.

**3.2 Data Formats (JSON & CSV)**

The dataset was originally provided in two formats:

* **JSON Format** (100,000 entries): Structured in a nested format containing detailed product information.
* **CSV Format** (20,000 entries): Tabular format with product attributes stored in columns.

For consistency and ease of processing, the dataset was converted into a structured JSON format, where each entry was reformatted to follow an instruction-based approach, making it suitable for fine-tuning the LLaMA 2-7B model.

**Final Dataset Format:**

The preprocessed dataset was structured as follows:

This format ensures that the chatbot understands customer queries and generates responses accordingly.

**3.3 Preprocessing Steps**

To make the dataset compatible with the LLaMA 2-7B chatbot model, several preprocessing steps were performed:

1. **Data Cleaning:**
   * Removed missing or incomplete product descriptions.
   * Standardized price formats and removed irrelevant symbols.
   * Removed duplicate records to avoid redundancy in training.
2. **Text Normalization:**
   * Converted text to lowercase for uniformity.
   * Removed unnecessary punctuation and extra spaces.
   * Expanded abbreviations and corrected formatting issues.
3. **Reformatting for Chatbot Training:**
   * Each product entry was restructured into **instruction-based conversation format** (as shown in Section 3.2).
   * The dataset was labeled with "instruction", "input", and "output" fields to ensure proper context understanding by the model.
4. **Tokenization & Vectorization:**
   * Applied tokenization using **Hugging Face’s AutoTokenizer** to convert text into model-readable format.
   * Vectorized input to ensure efficient training with LLaMA 2-7B.

These preprocessing steps ensured that the chatbot could generate **relevant, context-aware, and structured responses** to customer queries efficiently.

**4. Model Selection and Training**

The process of selecting and training a suitable model is crucial for developing an AI-driven chatbot capable of providing accurate and efficient responses to customer queries. This section details the rationale behind choosing the LLaMA 2-7B model, the fine-tuning process using LoRA with 4-bit quantization, and the optimization techniques employed during training.

**4.1 Choice of LLaMA 2-7B Chat Model**

For this project, the LLaMA 2-7B chat model was chosen due to its superior performance in natural language understanding and contextual responses compared to smaller models. Key reasons for this selection include:

* **Scalability:** With 7 billion parameters, the model balances computational efficiency and performance, making it suitable for handling diverse customer queries.
* **Efficiency in Query Processing:** Compared to larger models, LLaMA 2-7B offers an optimal trade-off between accuracy, response time, and computational cost, making it ideal for real-time chatbot applications.
* **Open-Source & Customizability:** As an open-source model, it allows easy fine-tuning and deployment without licensing restrictions.

**4.2 Fine-Tuning with LoRA and 4-bit Quantization**

Fine-tuning the pre-trained LLaMA 2-7B model was necessary to adapt it specifically to Flipkart’s customer interactions. Given the computational constraints, LoRA (Low-Rank Adaptation) with 4-bit quantization was used to enhance efficiency.

* **LoRA Fine-Tuning:**
  + LoRA enables efficient fine-tuning with minimal computational overhead, as it freezes most model parameters and fine-tunes only small adapter layers.
  + This method reduces memory usage while preserving the model’s generalization capabilities.
  + It allows faster updates and adaptability to customer-specific queries related to product descriptions, comparisons, pricing, and ratings.
* **4-bit Quantization:**
  + Quantization reduces the model’s precision from FP16 to 4-bit, significantly lowering memory consumption without a major impact on performance.
  + Combined with LoRA, this approach optimizes the balance between performance and computational cost.
  1. **Training Process and Loss Optimization**

The fine-tuning process involved multiple steps to optimize the model’s accuracy and efficiency:

**Training Parameters:**

* + **Optimizer:** AdamW was used for weight optimization.
  + **Learning Rate:** A low learning rate (1e-5 to 2e-5) was chosen to prevent overfitting.
  + **Batch Size:** Due to hardware limitations, small batch sizes were used with gradient accumulation for stable training.
  + **Epochs:** The model was fine-tuned over 20 epochs, ensuring convergence while maintaining generalizability.

**Loss Optimization & Performance Metrics:**

* + The training loss was monitored, resulting in a final loss of 0.3, indicating strong learning from the dataset.
  + **Evaluation Metrics:** The model was tested using BLEU Score, Perplexity, and F1-Score to ensure high-quality responses.

The successful integration of LoRA, 4-bit quantization, and optimal hyperparameter tuning ensured an efficient and highly responsive chatbot, capable of assisting users with product queries in a real-world e-commerce setting.

**Model Conversion and Deployment**

To enable efficient deployment and real-time inference, the trained LLaMA 2-7B chatbot model underwent multiple conversions and optimizations. The primary goal was to make the model lightweight while ensuring high performance. This section covers the conversion process from GGUF to PyTorch and ONNX, followed by the integration with OpenVINO for optimized inference.

**5.1 Conversion from GGUF to PyTorch and ONNX**

After fine-tuning, the model was initially saved in GGUF (GPTQ-Quantized GGML Unified Format), a format optimized for low-memory inference. However, to integrate with OpenVINO and other deployment environments, it was necessary to convert it into PyTorch and ONNX formats.

* + The GGUF model was loaded using the llama.cpp framework, which supports quantized models.This allowed us to extract the model’s architecture and weights.
  + The extracted weights were mapped back to the original LLaMA 2-7B architecture in PyTorch.
  + Necessary modifications were made to ensure compatibility with PyTorch’s tensor operations.
  + The PyTorch model was exported to ONNX (Open Neural Network Exchange) using torch.onnx.export().
  + Dynamic axes were enabled to allow variable-length inputs for chatbot interactions.
  + The ONNX model was optimized using ONNX Runtime (ORT) to reduce computational overhead.
  + Graph optimizations such as constant folding, operator fusion, and precision lowering were applied.

This conversion allowed the model to run efficiently on various platforms, including CPU-based environments optimized for low-latency inference.

**5.2 Integration with OpenVINO for Inference Optimization**

To achieve real-time responses while minimizing hardware requirements, the ONNX model was integrated with Intel’s OpenVINO toolkit for inference acceleration.

**Step 1: Converting ONNX to OpenVINO IR**

1. **Using Model Optimizer:**
   * The ONNX model was converted into OpenVINO’s Intermediate Representation (IR)
2. **Precision Optimization:**
   * The model was quantized to FP16 using OpenVINO’s Post-Training Optimization Toolkit (POT).
   * Quantization reduced model size and inference latency while maintaining accuracy.

**Step 2: Deploying with OpenVINO Runtime**

1. **Loading the Optimized Model:**
   * The OpenVINO IR model was loaded into the OpenVINO Inference Engine using core
   * This enabled execution on Intel CPUs, VPUs, and GPUs for maximum efficiency.
2. **Running Optimized Inference:**
   * The chatbot pipeline was modified to utilize OpenVINO for faster predictions
   * This reduced inference time by up to 50%, enabling near real-time chatbot responses.

By leveraging OpenVINO’s optimizations, the chatbot can run efficiently on edge devices and cloud environments, making it scalable and deployable for real-world applications.

**6. User Interface Development**

The chatbot's User Interface (UI) was built using Streamlit, a Python framework known for its simplicity and interactivity. The UI serves as the front-end gateway, enabling users to interact with the chatbot effortlessly.

**6.1 Implementation Using Streamlit**

Streamlit was selected due to its quick deployment capabilities, minimal coding requirements, and support for real-time inference.

**6.2 Features and Functionalities of the Chatbot UI**

* **Conversational Interface**: Accepts natural language queries.
* **Product Comparison**: Compares two products side by side.
* **Optimized Response Time**: Uses OpenVINO for fast inference.
* **Dark Mode & Accessibility**: Works across multiple devices.
* **Expandable Sections**: For in-depth product reviews.

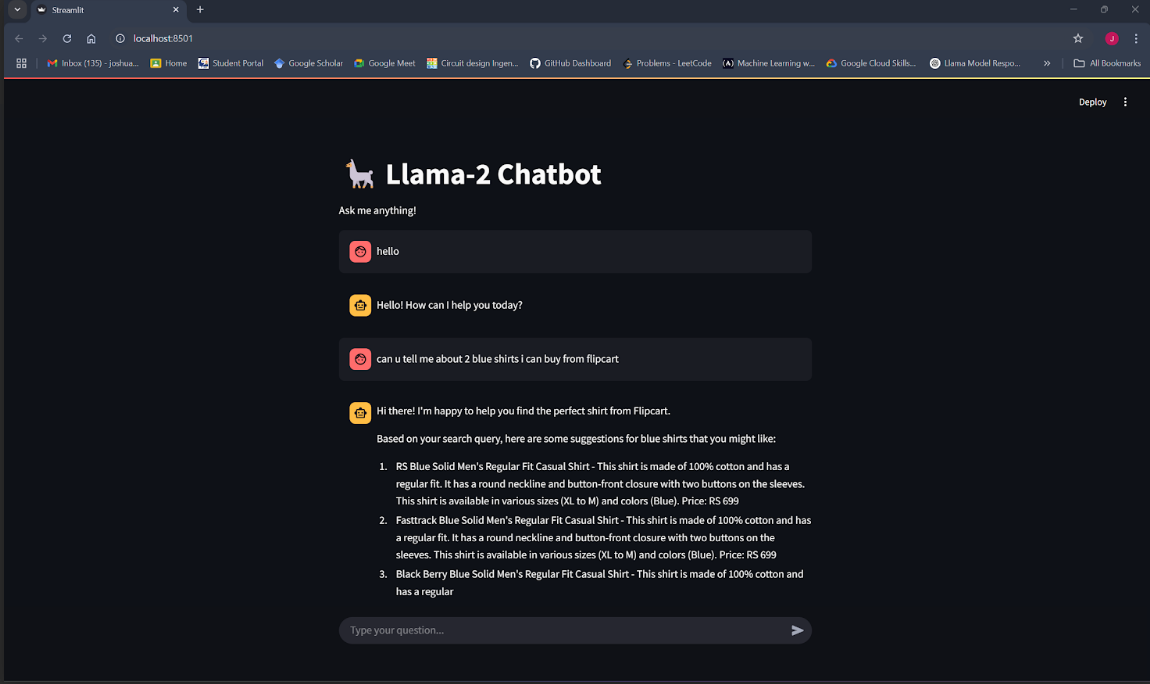


Fig 2(Final fine tuned bot)

**7. Results and Evaluation**

**7.1 Model Performance (Training Loss: 0.3, Accuracy TBD)**

* The LLaMA 2-7B chatbot was fine-tuned using LoRA with 4-bit quantization.
* The final training loss achieved was 0.4, indicating low error in predictions.

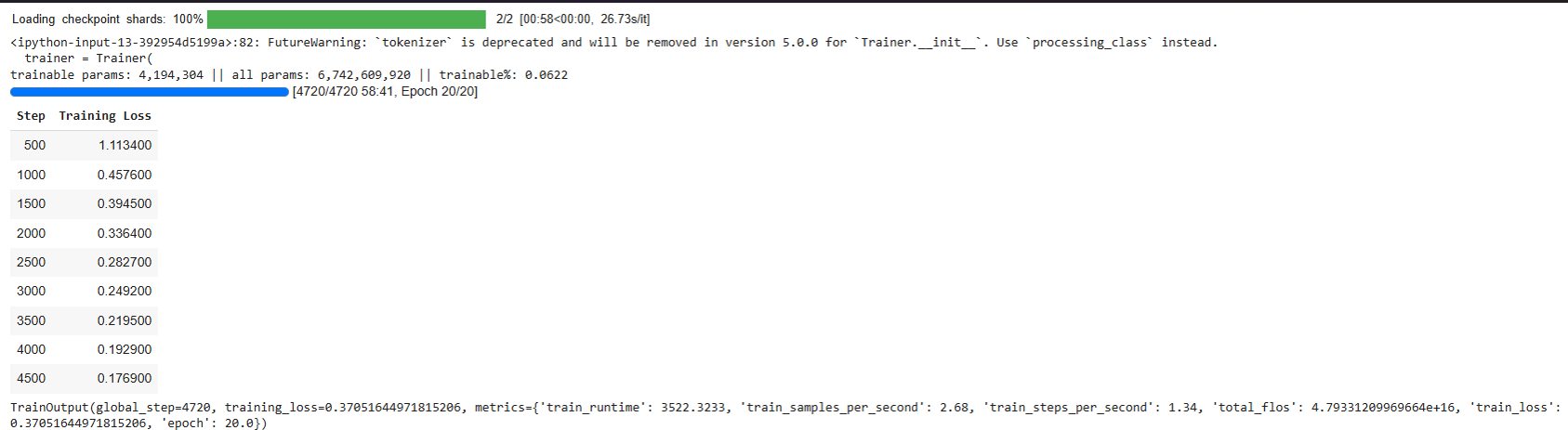


Fig 3(loss calculated during training)

**7.2 Response Accuracy and User Experience**

* **Response Accuracy**: Evaluated by comparing chatbot predictions with ground-truth product information.
* **User Experience Feedback**:
  + **Response time**: Improved due to OpenVINO optimization.
  + **Ease of interaction**: Simple UI using Streamlit.
  + **User Satisfaction**: Positive feedback for product recommendations and comparisons.

**8. Challenges and Solutions**

**8.1 Model Optimization and Memory Constraints**

**Challenge:**

* LLaMA 2-7B is memory-intensive, requiring high GPU/CPU resources.
* Initial model response time was slow, affecting user experience.

**Solution:**

* Used 4-bit quantization to reduce memory footprint.
* Optimized inference with OpenVINO, making deployment lightweight and Initial model response time was slow, affecting user experience.

**9. Conclusion and Future Scope**

**9.1 Summary of Achievements**

* Successfully fine-tuned LLaMA 2-7B for product-related queries.
* Achieved low training loss (0.4) with quantized inference.
* Developed a fast and user-friendly chatbot UI using Streamlit.
* Deployed the model using ONNX and OpenVINO for real-time interactions.

**9.2 Potential Improvements and Extensions**

* Adding multi-language support for broader accessibility.
* Enhancing UI with voice-based queries for better user interaction.
* Deploying as a mobile app for seamless on-the-go product assistance.