Bilingual AI-Powered Customer-Support WhatsApp Chatbot for KSFE

Anurag P. , Anvin C. Shaju , Darshana Das K. , Abdul Majid , and Sai Nath A.

Department of Computer Science & Engineering; Government Engineering College, Thrissur

***Abstract*—*The development and implementation of the Bilingual AI-Powered Customer-Support WhatsApp Chatbot for Kerala State Financial Enterprises Ltd. (KSFE) have been a significant milestone in enhancing customer service and engagement for the organization. The chatbot, designed to provide support in both English and Malayalam languages, offers a range of features that streamline customer interactions and improve overall user experience. This report outlines the development process, key features, and impact of the chatbot on KSFE's customer service operations. It also discusses future prospects for further enhancing the chatbot's capabilities and integration with other customer service channels.*

Keywords*—*WhatsApp API, OpenAI API, Fine-tuning, Conversational Memory

1. Introduction

The Kerala State Financial Enterprises Ltd (KSFE) is a leading public sector financial institution in Kerala, fully owned by the Government of Kerala. KSFE offers a wide range of financial services, including chit funds, deposits, and loans. With over 670 branches and more than 8300 employees, KSFE is one of the most profitable public sector undertakings in Kerala, boasting a revenue of over ₹73,000 crore ($8.8 billion).

Effective customer support mechanisms are essential for the successful operation of any company. In today's digital age, firms use websites and social media platforms to disseminate information and enhance their visibility. WhatsApp, being a widely used messaging platform in India, offers a unique opportunity for companies to connect with their customers on a more personal level. By leveraging WhatsApp, companies can provide instant and personalized support to their customers.

This WhatsApp bot is an AI-powered tool that allows customers to interact with a computer program in a conversational manner. These bots can understand and respond to user queries, providing information and assistance. By implementing a customer-support WhatsApp bot, companies can reduce the workload on their support teams and improve the efficiency of their customer service operations.

As part of our internship at KSFE, we, Computer Science Engineering undergraduate students from Government Engineering College, Thrissur, developed a bilingual Customer-Support WhatsApp bot for KSFE. This bot, available in both English and Malayalam languages, allows customers to access a wide range of services offered by KSFE directly through the messaging platform.

This report provides a detailed overview of the technical and functional aspects of the project, highlighting the key features and impact of the WhatsApp bot on KSFE's customer service operations.

2. WhatsApp API

Meta for Developers offers the WhatsApp Business API[1], a tool that empowers medium and large enterprises to engage with customers on a large scale. This API enables businesses to create systems that connect numerous customers with agents or bots, facilitating both automated and manual communication.

With the WhatsApp Business API, businesses can send and receive WhatsApp messages using a dedicated phone number. Interacting with this API requires a server to handle API requests and responses. This server can be developed using any backend technology. In our case, we utilized Node.js, a popular JavaScript runtime environment, to build the server for our WhatsApp bot.

The Cloud API is constructed on Graph API, which dictates that requests are formulated using the HTTP protocol, along with a combination of URL parameters, headers, and request bodies.

To send requests from the server to the WhatsApp API, we employed Axios, a widely used HTTP client for Node.js. Axios facilitates the transmission of requests, enabling the bot to engage with users on the WhatsApp platform.

```
await axios({
    method: "POST",
    url:
        "https://graph.facebook.com/v13.0/" +
        phone_no_id +
        "/messages?access_token=" +
        access_token,
    data: {
        messaging_product: "whatsapp",
        to: from,
        type: "text",
        text: {
            body: "Hey, I'm Customer-Support bot
     by KSFE. How can I help you today?",
        },
    },

    headers: {
        "Content-Type": "application/json",
        Authorization: `Bearer ${access_token}`,
    },
});
```

Code 1. *Example of Axios POST request to send a message*

Webhook configuration is necessary on the Meta for Developers website to allow the forwarding of customer requests to our server. This setup ensures seamless communication between customers and our WhatsApp bot.

3. Fine-Tuning Large Language Model

Large Language Models are trained on large, diverse datasets to learn general language patterns. Fine-tuning[2] allows you to adapt the model to your specific task or domain, improving

its performance and relevance. By fine-tuning on domain-specific data, the model can learn specialized knowledge and terminology, making it more effective in understanding and generating content relevant to that domain. Leveraging pre-trained models for fine-tuning can be more cost-effective than developing a custom model from scratch, especially for smaller-scale projects or limited resource environments.

Fine-tuning allows you to adapt the model to changing requirements or new tasks without the need to retrain the entire model, making it flexible and adaptable to evolving needs. Fine-tuning on top of state-of-the-art pre-trained models like GPT-3.5 can help you achieve state-of-the-art performance in various NLP tasks, giving your chatbot a competitive edge.

Steps involved in fine-tuning process are:

3.1. Preparing the dataset. The dataset used for fine-tuning must be in JSONL format. It consists of conversations between three roles: "system," "user," and "assistant." The "system" role provides the initial prompt to the assistant on how to handle the user's prompt and how the assistant should respond. The "user" role contains potential user prompts that need to be answered by the assistant. The "assistant" role contains the expected reply from the assistant in a real-world scenario. This methodology allows LLM models to be trained for specific topics, making them more domain-specific. Increasing the number of examples used for training can improve the accuracy of fine-tuning.

```
{
        "messages": [
            {
                "role": "system",
                "content": "You're KSFE bot, a
    responsible WhatsApp bot AI to serve
    customers of Kerala State Financial
    Enterprises Ltd. You should answer users'
    questions in a polite and effective way."
            },
            {
                "role": "user",
                "content": "How can I download
    my previous online transaction receipts?"
            },
            {
                "role": "assistant",
                "content": "You can go to https
    ://ksfeonline.com/payment to download your
    previous online transaction receipts."
            }
        ]
    },
```

Code 2. *Example of dataset in JSON format*

3.2. Validating the data. Data validation involves several activities that are performed using a Python notebook. These activities include data loading, format validation, token counting utilities, data warnings, and token counts. The tiktoken Python package is used to provide a fast and reversible method for converting text into tokens for use with OpenAI's models. Additionally, cost estimation is conducted to estimate the number of tokens needed for fine-tuning using the dataset.

```
Dataset has ~12022 tokens that will be charged
    for during training
By default, you'll train for 3 epochs on this
    dataset
By default, you'll be charged for ~36066 tokens
```

Code 3. *Result of cost estimation*

3.3. Creating a fine-tuning job. To fine-tune a model, a job can be created on the OpenAI website. The process involves selecting a base model for fine-tuning, uploading training data in JSONL format, adding validation data, specifying a custom suffix to be appended to the output model name, and then creating the job. Once the fine-tuning job is complete, a fine-tuned model name is provided, which can be used in the project instead of the base model like "gpt-3.5-turbo".

4. Technologies Used

4.1. Node.js. The chatbot is a Node.js application written in JavaScript. Node.js enables us to send and receive API requests and process them according to our requirements. JavaScript, known for its user-friendly nature, allows for smooth development and maintenance of the application. Axios is used to send HTTP requests.

4.2. Railway.app. Railway.app is the platform that simplified the deployment and hosting of our Node.js application. It provides tools and services to manage the infrastructure required to run Node.js applications, making it easier for developers to deploy their projects and keep them running smoothly.

4.3. Google Translate API. The Cloud Translate API is a premium language translation service offered by Google. We utilize this service to manage Malayalam-English translations within the chatbot. Due to the AI model's proficiency being higher in English than in Malayalam, we translate user conversations from Malayalam to English, pass them to the AI model, and then translate the AI model's responses back to Malayalam before sending them to the user.

5. Features

5.1. Bilingual Support. The chatbot is available in both English and Malayalam languages to cater to the diverse customer base of KSFE. Bilingual support is essential as Malayalam is the preferred language for a significant portion of KSFE's customers. When customers start interacting with the bot, they are prompted to choose their preferred language. Additionally, customers have the flexibility to switch languages during their interaction. The bot allows customers to chat in both English and Malayalam, receiving responses in their chosen language.

5.2. Enhanced AI Conversations. The chatbot utilizes the GPT-3.5-turbo AI model from OpenAI, which has been fine-tuned with KSFE data to provide higher-quality answers to customer queries. A custom model has been created by fine-tuning with a dataset of questions and answers sourced from Internet forums. This enhancement enables the bot to provide more intelligent responses, especially regarding matters related to KSFE.

5.3. Conversational Memory. The chatbot incorporates a context memory.[3] feature that remembers the latest conversation between the customer and the AI model. This allows

the bot to recall and build upon previous interactions, providing more relevant and personalized responses. The context memory feature enhances the flow of the conversation, making the chat experience more seamless and engaging for the user.

5.4. Request a Call. Customers have the option to request a call from customer support to address their queries, similar to the "Request a Call" feature on the KSFE website. The chatbot prompts users to provide their name, email, and district. This information is then stored in a database, enabling customer care personnel to follow up and assist further based on the provided details.

5.5. Products & Services Information. The Products Services section provides a comprehensive overview of the diverse range of financial products and services offered by KSFE. It includes detailed descriptions of each product, highlighting key features, eligibility criteria, and benefits. Additionally, the section provides links to the respective webpages on the KSFE website, allowing users to access more detailed information and explore the various offerings available.

6. Source Code

The source code and other related information for the project can be found at: github.com/anuragpoolakkal/ksfebot

7. Conclusion

The development and implementation of the Bilingual AI-Powered Customer-Support WhatsApp Chatbot for Kerala State Financial Enterprises Ltd. (KSFE) have been a significant milestone in enhancing customer service and engagement for the organization. The chatbot, designed to provide support in both English and Malayalam languages, offers a range of features that streamline customer interactions and improve overall user experience.

7.1. Achievements.

- **Enhanced Customer Support:** The chatbot provides customers with instant access to information and assistance, reducing the need for human intervention in routine queries.
- **Improved Efficiency:** By automating responses to frequently asked questions, the chatbot has helped KSFE customer care personnel focus on more complex and critical customer issues, improving overall efficiency.
- **Personalized Interaction:** The chatbot's contextual memory feature allows it to remember previous interactions, enabling more personalized responses and enhancing the overall user experience.
- **Increased Accessibility:** The chatbot's availability on WhatsApp, a widely used messaging platform, has made it more accessible to a larger audience, including those with limited internet connectivity.

7.2. Impact.

- **Customer Satisfaction:** The chatbot has contributed to higher customer satisfaction levels by providing quick and accurate responses to customer queries, leading to a more positive customer experience.
- **Cost-Effectiveness:** By automating customer support, KSFE has been able to reduce operational costs associated with traditional customer service methods.
- **Scalability:** The chatbot's design allows for easy scalability to accommodate a growing customer base and evolving customer needs.

7.3. Future Prospects.

- **Continuous Improvement:** KSFE can further enhance the chatbot by continuously updating its knowledge base and incorporating feedback from users to improve its responses and functionality.
- **Integration with Other Channels:** The chatbot can be integrated with other customer service channels, such as email and phone support, to provide a seamless omnichannel experience for customers.
- **Advanced AI Capabilities:** Future enhancements could include integrating more advanced AI capabilities, such as natural language processing (NLP), to enable the chatbot to understand and respond to more complex queries.

In conclusion, the Bilingual AI-Powered Customer-Support WhatsApp Chatbot has been a valuable addition to KSFE's customer service ecosystem, providing customers with a more efficient and personalized support experience. As technology continues to evolve, the chatbot will play an increasingly important role in meeting the growing demands of KSFE's customer base.

References

[1] M. for Developers. "Get started - cloud api". (), [Online]. Available: https://developers.facebook.com/docs/whatsapp/cloud-api/get-started.

[2] OpenAI. "Fine-tuning - openai api". (), [Online]. Available: https://platform.openai.com/docs/guides/fine-tuning.

[3] A. Radzinski. "Short-term memory: Maintaining conversation context - chatbots life". (), [Online]. Available: https://chatbotslife.com/short-term-memory-maintaining-conversation-context-a62986fff183.