ChatBot com IA

UNIVERSIDADE NOVE DE JULHO - UNINOVE

São Paulo
2024

ChatBot com IA

Projeto apresentado a Universidade Nove de Julho - UNINOVE, como parte dos requisitos obrigatórios para obtenção do título de Bacharel em Ciência da Computação.

Prof. Orientador: Edson Melo de Souza, Dr.

São Paulo
2024

Abstract

This document describes the development of a chatbot for the confectionery Doces Vermelhos, using the ChatGPT API and integration with WhatsApp via Twilio. The chatbot is capable of interacting with customers, answering frequently asked questions, and assisting with orders. This project aims to improve customer service by providing quick and accurate responses, optimizing the time of human attendants.

Contents

1. Introdução

1.1 Contexto

Com o aumento da demanda por atendimento ao cliente automatizado, as pequenas empresas podem se beneficiar do uso de chatbots para melhorar a experiência do cliente. Este projeto visa desenvolver um chatbot para empresas pequenas.

1.2 Objetivo

O objetivo deste projeto é criar um chatbot capaz de interagir com clientes através do WhatsApp, respondendo a perguntas comuns e assistindo em pedidos, utilizando a API do ChatGPT.

1.3 Estrutura do Documento

Este documento está organizado da seguinte forma:

- Capítulo 1: Introdução
- Capítulo 2: Fundamentação Teórica
- Cap´ıtulo 3: Metodologia
- Capítulo 4: Análise dos Resultados
- Cap´ıtulo 5: Chatbots Baseados em Regras
- Capítulo 6: Conclusões
- Referências
- Apêndices

2. Fundamentação Teórica

2.1 Chatbots

Chatbots são programas de computador projetados para simular conversas humanas. Eles são amplamente utilizados em atendimento ao cliente para fornecer respostas rápidas e eficientes.

2.2 APIs Utilizadas

2.2.1 OpenAI GPT-4

A API do OpenAI GPT-4 é utilizada para processar e gerar respostas às mensagens dos clientes. A tecnologia GPT-4 oferece respostas contextualizadas e relevantes, melhorando a experiência do usuário.

2.2.2 Twilio

O Twilio é utilizado para integrar o chatbot com o WhatsApp, permitindo a comunicação com os clientes. A API do Twilio facilita o envio e recebimento de mensagens de maneira programática.

Figure 2.1: Exemplo de integração com a API do OpenAI GPT-4.

Figure 2.2: Logo do Twilio.

3. Metodologia

3.1 Configuração do Ambiente

Para configurar o ambiente de desenvolvimento, siga os passos abaixo:

1. Instalar as bibliotecas necessárias:

```
pip install openai flask twilio
```

2. Configurar as variáveis de ambiente para as APIs do OpenAI e do Twilio.

3.2 Implementação

3.2.1 Estrutura do Código

O código do chatbot é estruturado em três partes principais:

- Configuração das APIs
- Funções principais do chatbot
- Rota do Flask para o webhook do Twilio

```
# Configurar API do OpenAI
openai.api_key = 'SUA_CHAVE_API_OPENAI'

# Configurar API do Twilio
twilio_account_sid = 'SUA_CONTA_SID_TWILIO'
twilio_auth_token = 'SEU_AUTH_TOKEN_TWILIO'
twilio_whatsapp_number = 'SEU_NUMERO_WHATSAPP_TWILIO'
client = Client(twilio_account_sid, twilio_auth_token)
```

4. Análise dos Resultados

4.1 Interação com Clientes

Durante os testes, o chatbot foi capaz de responder eficientemente a várias perguntas frequentes e auxiliar em pedidos. A interação é iniciada com uma saudação e seguida por respostas contextuais.

4.2 Desempenho do Chatbot

O desempenho do chatbot foi avaliado em termos de rapidez e precisão das respostas. Em média, o chatbot respondeu às mensagens em menos de 2 segundos, mantendo um alto nível de precisão nas respostas.

5. Chatbots Baseados em Regras

Antes de adotar a tecnologia de inteligência artificial, a nossa empresa confeccionava chatbot baseado em regras. Este tipo de chatbot funcionava através de um conjunto predefinido de regras e respostas, limitando a flexibilidade e a eficácia do atendimento ao cliente.

5.1 Limitações dos Chatbots Baseados em Regras

5.1.1 Flexibilidade e Escalabilidade Limitadas

Chatbots baseados em regras têm dificuldades em lidar com variações nas perguntas dos usuários e requerem constante atualização manual das regras. Isso limita a capacidade de escalar o atendimento de forma eficaz.

5.1.2 Experiência do Usuário Inferior

A falta de compreensão contextual pode levar a interações frustrantes para os clientes, que recebem respostas r´ıgidas e muitas vezes irrelevantes.

5.1.3 Manutenção Intensiva

Manter e atualizar um chatbot baseado em regras é um processo manual e intensivo, que exige tempo e recursos significativos. Qualquer alteração nas regras ou no conteúdo do chatbot precisa ser feita manualmente, o que não é prático em um ambiente de negócios dinâmico.

5.1.4 Capacidades de Aprendizado

Chatbots baseados em IA, como os que utilizam o GPT-4, podem aprender e melhorar com o tempo, fornecendo respostas mais precisas e personalizadas. Essa capacidade de aprendizado contínuo é uma vantagem significativa em relação aos chatbots baseados em regras, que permanecem estáticos a menos que sejam atualizados manualmente.

5.2 Transição para Chatbots com IA

Devido às limitações mencionadas, a nossa empresa decidiu adotar um chatbot com inteligência artificial. A transição para um chatbot baseado em IA trouxe vários benefícios, incluindo:

- **Maior Flexibilidade:** Capacidade de lidar com uma ampla gama de perguntas e fornecer respostas contextualmente relevantes.
- **Melhora na Experiência do Usuário:** Interações mais naturais e satisfatórias, resultando em maior satisfação do cliente.
- **Redução da Manutenção:** Menor necessidade de atualizações manuais frequentes, economizando tempo e recursos.
- **Capacidade de Aprendizado:** O chatbot pode aprender com as interações anteriores e melhorar continuamente suas respostas.

6. Conclusões

6.1 Resumo

O desenvolvimento do chatbot para pequenas empresas mostrou-se eficiente para automatizar o atendimento ao cliente via WhatsApp, utilizando a API do ChatGPT e a integração com o Twilio. O chatbot é capaz de fornecer respostas rápidas e precisas, melhorando a experiência do cliente e liberando tempo dos atendentes humanos para tarefas mais complexas.

6.2 Problema Resolvido

O principal problema resolvido foi a necessidade de atendimento ao cliente rápido e eficiente. Antes do chatbot, os clientes enfrentavam longos tempos de espera para obter respostas. Com a automacão, os clientes recebem atendimento imediato, o que aumenta a satisfação e a fidelização.

6.3 Trabalhos Futuros

Para futuras melhorias, sugere-se:

- Integração com um sistema de pedidos online.
- Personalização de respostas com base no histórico do cliente.
- Implementação de um sistema de feedback para melhorar continuamente o serviço.
- Expansão das funcionalidades do chatbot para incluir sugestões de produtos baseadas nas preferências do cliente.

Referências

- OpenAI. OpenAI API Documentation. Disponível em: https://beta.openai.com/docs/
- Twilio. Twilio API for WhatsApp Documentation. Disponível em: https://www.twilio.com/docs/whatsapp
- Abbasi, A.; Altmann, J.; Hossain, L. Identifying the effects of co-authorship networks on the performance of scholars: A correlation and regression analysis of performance measures and social network analysis measures. Journal of Informetrics, Elsevier Ltd, v. 5, n. 4, p. 594–607, 2011. ISSN 17511577. doi:10.1016/j.joi.2011.05.007.
- D'Uggento, A. M.; Ricci, V.; Toma, E. An indicator proposal to evaluate research activities based on Scimago institutions ranking (SIR) data: An application for italian high education institutions. Electronic Journal of Applied Statistical Analysis, v. 9, n. 4, p. 655–674, 2016. ISSN 20705948. doi:10.1285/i20705948v9n4p655.
- Mitchell, T. M. Machine learning. McGraw-hill, New York, 1997. 432 p.
- Villaseñor-Almaraz, M. et al. Impact factor correlations with Scimago Journal Rank, Source Normalized Impact per Paper, Eigenfactor Score, and the CiteScore in Radiology, Nuclear Medicine Medical Imaging journals. La radiologia medica, Springer Milan, v. 124, n. 6, p. 495–504, jun. 2019. ISSN 0033-8362. doi:10.1007/s11547-019-00996-z.

7. Código Completo

```
import os
import openai
from flask import Flask, request, jsonify
from twilio.twiml.messaging_response import MessagingResponse
from twilio.rest import Client

openai.api_key = 'SUA_CHAVE_API_OPENAI'
twilio_account_sid = 'SUA_CONTA_SID_TWILIO'
twilio_auth_token = 'SEU_AUTH_TOKEN_TWILIO'
twilio_whatsapp_number = 'SEU_NUMERO_WHATSAPP_TWILIO'
client = Client(twilio_account_sid, twilio_auth_token)

app = Flask(__name__)

def send_greeting():
    greeting_message = (
        "Olá! Bem-vindo à Doces Vermelhos! \n"
        "Eu sou seu assistente virtual. Como posso ajudar você hoje?\n"
        "1. Ver nosso cardápio\n"
        "2. Fazer um pedido\n"
        "3. Saber sobre nossas promoções\n"
        "4. Falar com um atendente"
    )
    return greeting_message

def get_response_from_chatgpt(message):
    response = openai.Completion.create(
        engine="text-davinci-003",
        prompt=message,
        max_tokens=150
    )
    return response.choices[0].text.strip()

@app.route('/webhook', methods=['POST'])
def webhook():
    incoming_msg = request.values.get('Body', '').strip()
    response = MessagingResponse()
```

```
    message = response.message()

    if incoming_msg.lower() in ['oi', 'olá', 'bom dia', 'boa tarde', 'boa noite'
        message.body(send_greeting())
    else:
        chatgpt_response = get_response_from_chatgpt(incoming_msg)
        message.body(chatgpt_response)

    return str(response)

if __name__ == '__main__':
    app.run(debug=True)
```