Introduction to Artificial Intelligence and Generative Learning
CPS 769

Segundo Trimestre de 2024

Professores:
Edmundo de Souza e Silva (PESC/COPPE/UFRJ)
Rosa M. Leão (PESC/COPPE/UFRJ)
Participação Especial: Gaspare Bruno (Diretor Inovação, ANLIX)

Lista de Exercícios 6

ATENÇÃO! Faça as listas de forma que TODAS AS RESPOSTAS sejam DEVIDAMENTE COMENTADAS (passos para se chegar a resposta). Ser claro e objetivo facilitará organizar as ideias para as discussões em classe.

Para facilitar escrever a lista de forma clara, é possível traduzir equações escritas a mão para LaTex:
`https://mathpix.com/`, ver também
`https://www.overleaf.com/learn/latex/Questions/Are_there_any_tools_to_help_transcribe_mathematical_formulae_into_LaTeX%3F`.

Objetivo

O objetivo deste trabalho é desenvolver uma aplicação que: (a) aceite perguntas dos usuários em linguagem natural e sobre a Qualidade de Experiência (QoE) na transmissão de vídeo e; (b) forneça respostas em linguagem natural. Para realizar o trabalho, poderão ser utilizadas as APIs da OpenAI. O raciocínio *Chain of Thought* (CoT) e *function calling* devem ser usados para processar os dados fornecidos, analisar a qualidade de experiência (QoE) e responder dinamicamente às perguntas dos usuários. A aplicação deve utilizar os arquivos de dados fornecidos, que contêm medições de *bitrate* de vídeo e latência.

Background:
A tarefa foi motivada por um *data challenge* do Comitê Técnico de Monitoramento (CT-Mon) da RNP. Um serviço de streaming deve proporcionar a recepção do vídeo sem interrupções e com boa imagem. Para termos uma boa imagem a taxa de transmissão deve ser alta. A latência (RTT) entre o servidor e o cliente tem um papel importante na transmissão, e ela deve ser, de forma geral, baixa. Uma forma de atingir a qualidade desejada, é segmentar um vídeo em resoluções diferentes e armazená-lo em vários servidores. Por conseguinte, cada requisição do cliente pode ser direcionada para o servidor que esteja com as condições necessárias para transmitir o vídeo com a melhor qualidade (QoE), isto é, com alta taxa de transmissão e baixa latência. (Na verdade, essa é uma simplificação do problema para o trabalho.)

Dados Fornecidos

Dois arquivos CSV são fornecidos:

- **bitrate_train.csv**: Contém medições da taxa de transmissão (*bitrate*) de vídeo (em kbps).
  - Coluna 1: ID do Cliente (string)
  - Coluna 2: ID do Servidor (string)
  - Coluna 3: Timestamp (inteiro)
  - Coluna 4: *Bitrate* (em kbps)

- **rtt_train.csv**: Contém medições de latência (RTT, em ms).
  - Coluna 1: ID do Cliente (string)
  - Coluna 2: ID do Servidor (string)
  - Coluna 3: Timestamp (está em Unix time em segundos desde 1/1/1970).
  - Coluna 4: Latência (em ms)

Tarefas do Trabalho

Sua tarefa é desenvolver uma aplicação que integre o entendimento de linguagem natural e a análise de QoE utilizando as APIs da OpenAI. A aplicação deve:

- Aceitar perguntas em linguagem natural dos usuários sobre os dados fornecidos.
- Fornecer respostas em linguagem natural, analisando os dados usando *Chain of Thought*.
- Utilizar o recurso de *function calling* da OpenAI para processar os dados e calcular os resultados quando necessário.

Você deve completar as seguintes tarefas:

1. Desenvolver uma Interface de Linguagem Natural

Desenvolva uma interface de usuário que permita que os usuários façam perguntas em linguagem natural. Exemplos de perguntas incluem:

- "Qual cliente tem a pior qualidade na aplicação de vídeo streaming?´´
- "Qual é a melhor estratégia de troca de servidor para maximizar a qualidade de experiência do cliente X?"
- "Qual servidor tem a qualidade de experiência mais consistente?"

2. Implementar o Raciocínio *Chain of Thought* (CoT)

Incorpore o raciocínio CoT em sua aplicação para processar e responder às perguntas dos usuários passo a passo:

1. Analise e compreenda a pergunta do usuário.
2. Divida a pergunta em uma série de etapas intermediárias.
3. Chame as funções relevantes para processar os dados e calcular a resposta.
4. Gere uma explicação em linguagem natural dos resultados com base nas etapas executadas.

3. Utilizar o *Function Calling* da OpenAI para Processamento de Dados

Integre o *function calling* com a API da OpenAI para lidar com tarefas específicas de processamento de dados. Por exemplo:

- Chamadas de funções para calcular a QoE usando a fórmula:

$$QoE = \frac{\text{Bitrate normalizado}}{\text{Latência normalizada}}$$

  Você deverá então, como parte do CoT, normalizar todo o dataset por exemplo usando normalização *max-min*.
- Chamadas de funções para identificar os clientes e servidores com a pior QoE.
- Chamadas de funções para calcular as melhores estratégias de troca de servidor para os clientes.

As chamadas de função devem calcular as respostas com base no conjunto de dados fornecido e, em seguida, retornar esses resultados como parte de uma resposta em linguagem natural.

Exemplos de Perguntas que a Aplicação Deve Responder

Sua aplicação deve ser capaz de responder aos seguintes tipos de perguntas utilizando o raciocínio CoT e o *function calling* da OpenAI:

1. Cliente com a Pior QoE

Exemplo de Pergunta: "Qual cliente tem a pior qualidade de recepção de vídeo ao longo do tempo?"

1. Analise a pergunta e reconheça que ela envolve a identificação de uma definição sobre o que é qualidade (QoE) e como é calculada.
2. Analise a pergunta e reconheça que ela envolve a identificação do cliente com a menor QoE.
3. Calcule a QoE para cada cliente usando a fórmula e determine qual cliente tem a menor média de QoE.
4. Retorne o ID do cliente com uma explicação de como o resultado foi calculado.

2. Servidor com a QoE Mais Consistente

Exemplo de Pergunta: "Qual servidor fornece a QoE mais consistente?"

1. Analise a pergunta e entenda que consistência se refere à baixa variação de QoE ao longo do tempo.
2. Calcule a variância da QoE para cada servidor com base em todos os clientes conectados a ele.
3. Identifique e retorne o servidor com a menor variância de QoE.

3. Melhor Estratégia de Troca de Servidor para um Cliente

Exemplo de Pergunta: "Qual é a melhor estratégia de troca de servidor para maximizar a qualidade de experiência do cliente X?"

1. Analise a pergunta para entender que ela envolve calcular a QoE para vários servidores em diferentes momentos.
2. Use CoT para determinar a QoE de todos os servidores aos quais o cliente pode se conectar.
3. Recomende a melhor estratégia de troca identificando o servidor com a melhor QoE em cada timestamp.
4. Retorne o plano de troca recomendado e explique como ele foi derivado.

4. Efeito da Mudança de Latência na QoE

Exemplo de Pergunta: "Se a latência aumentar 20%, como isso afeta a QoE do cliente Y?"

1. Analise a pergunta e identifique que é necessário ajustar os valores de latência para um cliente específico.
2. Recalcule a QoE usando os valores de latência modificados e retorne os resultados.
3. Explique como a mudança na latência impacta a QoE do cliente.

Entregáveis

- Uma aplicação funcional que aceite perguntas em linguagem natural relacionadas à QoE e forneça respostas em linguagem natural utilizando raciocínio CoT e *function calling* (por exemplo, da OpenAI).

- Indique também como tratar uma pergunta que não pode ser respondida pelos seus cálculos, ou estiver fora de contexto.
- Um relatório breve resumindo sua abordagem para o raciocínio CoT, *function calling* e as APIs usadas, incluindo exemplos de perguntas e respostas que sua aplicação pode lidar.
- Seu código, incluindo documentação explicando como o raciocínio CoT e o *function calling* foram implementados.

Critérios de Avaliação

- **Correção:** A aplicação deve responder corretamente às perguntas relacionadas à QoE utilizando os dados fornecidos.
- **Implementação de CoT:** O raciocínio Chain of Thought deve ser claro e lógico nas respostas.
- ***Function Calling*:** Uso adequado do *function calling* da OpenAI para lidar com tarefas de processamento de dados.
- **Interação em Linguagem Natural:** A aplicação deve lidar com uma variedade de perguntas em linguagem natural e fornecer respostas claras e coerentes.
- **Documentação e Qualidade do Código:** Seu código deve ser bem documentado e fácil de seguir, com explicações claras de sua abordagem.

Código de um Exemplo (não inclui normalização dos dados)

Abaixo está um exemplo básico de como configurar uma chamada de função com a API da OpenAI para lidar com cálculos de QoE:

```
import pandas as pd

# Carregar os dados
bitrate_data = pd.read_csv('bitrate.csv')
latency_data = pd.read_csv('latency.csv')

# Mesclar os dados com base nos pares cliente-servidor e timestamp
merged_data = pd.merge(bitrate_data, latency_data, on=['ClientID', 'ServerID', 'Timestamp'])

# Definir uma função para calcular a QoE (não inclui normalização)
def calculate_qoe(row):
    return row['Bitrate'] / row['Latency']

# Aplicar o cálculo de QoE ao conjunto de dados
merged_data['QoE'] = merged_data.apply(calculate_qoe, axis=1)

# Exemplo de função para encontrar o cliente com a pior QoE
def worst_qoe_client(data):
    return data.groupby('ClientID')['QoE'].mean().idxmin()

# Exemplo de chamada de API para responder a uma pergunta sobre o cliente com a pior QoE
client_with_worst_qoe = worst_qoe_client(merged_data)
print(f'O cliente com a pior QoE é {client_with_worst_qoe}')
```