Desenvolvimento de uma Aplicação Java para Análise de Sentimentos

Usando API do ChatGPT

Este artigo apresenta um procedimento técnico completo para o desenvolvimento de uma aplicação Java que utiliza a API da OpenAI (ChatGPT) para realizar análise de sentimentos em textos fornecidos. O objetivo é classificar sentenças como **Positivas**, **Negativas** ou **Neutras**, por meio de requisições à API com prompts personalizados.

Estrutura do Projeto

O projeto é composto por três classes Java principais:

- Credentials.java: Armazena a chave da API.
- ChatGPT_AnaliseSentimento.java: Responsável por interagir com a API.
- ChatGPT_Run.java: Classe executável com entrada de dados e exibição de resultado.

Requisitos Técnicos

Pré-requisitos:

- Java 8 ou superior
- Biblioteca JSON (org.json)
- Acesso à internet
- Chave da API da OpenAI (modelo GPT-3.5-turbo ou GPT-4)

Dependência:

Para projetos Maven:

xml

```xml
<dependency>
    <groupId>org.json</groupId>
    <artifactId>json</artifactId>
    <version>20210307</version>
</dependency>
```

Estrutura de Diretórios

```
/ChatGPTSentimento/
├── src/
│   ├── Credentials.java
│   ├── ChatGPT_AnaliseSentimento.java
│   └── ChatGPT_Run.java
```

1. Classe Credentials.java

Esta classe centraliza o armazenamento da chave da API da OpenAI, evitando exposição direta em múltiplos arquivos.

```
public class Credentials {
    public static final String OPENAI_API_KEY = "sua-chave-api-aqui";
}
```

2. Classe ChatGPT_AnaliseSentimento.java

Classe responsável pela comunicação com a API ChatGPT e análise da resposta.

Principais responsabilidades:

- Montar o prompt
- Configurar a requisição HTTP
- Enviar e receber dados da API
- Extrair o sentimento da resposta

Exemplo (resumo do código):

```
import org.json.JSONArray;
import org.json.JSONObject;

import java.io.*;
import java.net.HttpURLConnection;
import java.net.URL;

public class ChatGPT_AnaliseSentimento {

    public static String analisarSentimento(String texto) throws IOException {
        URL url = new URL("https://api.openai.com/v1/chat/completions");
        HttpURLConnection conexao = (HttpURLConnection) url.openConnection();
        conexao.setRequestMethod("POST");
        conexao.setRequestProperty("Authorization", "Bearer " + Credentials.OPENAI_API_KEY);
        conexao.setRequestProperty("Content-Type", "application/json");
        conexao.setDoOutput(true);

        // Construção do prompt
        JSONObject mensagem = new JSONObject();
        mensagem.put("role", "user");
        mensagem.put("content", "Analise o sentimento do seguinte texto e responda com apenas uma palavra: Positivo,
Negativo ou Neutro. Texto: '" + texto + "'");

        JSONArray mensagens = new JSONArray();
        mensagens.put(mensagem);

        JSONObject requisicao = new JSONObject();
        requisicao.put("model", "gpt-3.5-turbo");
        requisicao.put("messages", mensagens);

        try (OutputStream os = conexao.getOutputStream()) {
            byte[] entrada = requisicao.toString().getBytes("utf-8");
            os.write(entrada, 0, entrada.length);
        }

        BufferedReader br = new BufferedReader(new InputStreamReader(conexao.getInputStream(), "utf-8"));
        StringBuilder resposta = new StringBuilder();
        String respostaLinha;
        while ((respostaLinha = br.readLine()) != null) {
            resposta.append(respostaLinha.trim());
        }
```

```
        JSONObject respostaJson = new JSONObject(resposta.toString());
        return respostaJson
                .getJSONArray("choices")
                .getJSONObject(0)
                .getJSONObject("message")
                .getString("content")
                .trim();
    }
}
```

3. Classe ChatGPT_Run.java

Classe principal do projeto, executa a aplicação via console.

Responsabilidades:

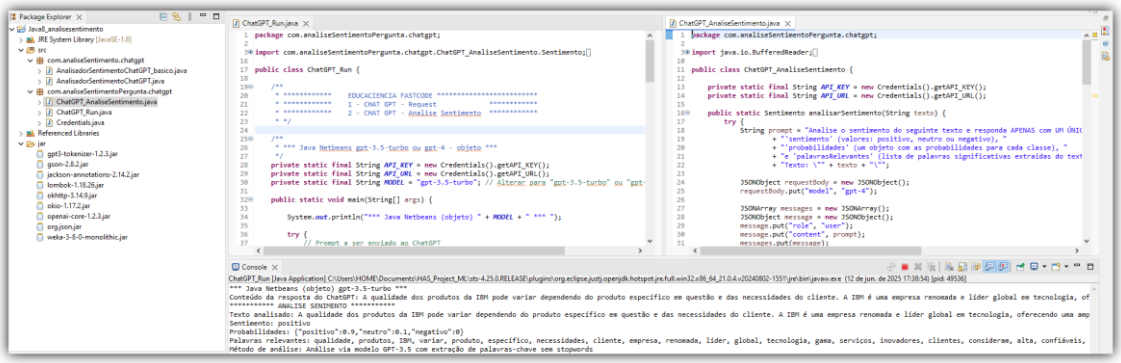
- Receber entrada do usuário (texto)
- Chamar o método de análise
- Exibir o resultado ao usuário

```
import java.util.Scanner;

public class ChatGPT_Run {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Digite o texto para análise de sentimento:");
        String texto = scanner.nextLine();

        try {
            String resultado = ChatGPT_AnaliseSentimento.analisarSentimento(texto);
            System.out.println("Sentimento detectado: " + resultado);
        } catch (Exception e) {
            System.err.println("Erro ao analisar sentimento: " + e.getMessage());
        }
    }
}
```

Testes e Validação

Exemplos de entrada e saída:

Texto	Resultado esperado
"Estou muito feliz hoje!"	Positivo
"Esse dia está horrível."	Negativo
"Hoje foi um dia comum."	Neutro



Educa indica como melhorias Futuras

- Suporte a múltiplos idiomas.
- Interface gráfica (JavaFX/Swing).
- Geração de relatórios CSV.
- Classificação com percentual (ex: 70% positivo).
- Fila de textos em lote.

Este procedimento demonstra como criar uma aplicação Java robusta e funcional para análise de sentimentos com a API do ChatGPT.

A modularização facilita manutenções, testes e expansões futuras.

O uso do modelo da OpenAI garante alta qualidade na análise sem necessidade de treinar modelos próprios.

1. Interface Gráfica com JavaFX (exemplo simples)

```
Label label = new Label("Digite o texto:");
TextField input = new TextField();
Button btn = new Button("Analisar");
Label resultado = new Label();

btn.setOnAction(e -> {
    String texto = input.getText();
    String sentimento = ChatGPT_AnaliseSentimento.analisarSentimento(texto);
    resultado.setText("Sentimento: " + sentimento);
});
```

2. Leitura de Textos em Lote (ex: de um .txt)

```
List<String> linhas = Files.readAllLines(Paths.get("textos.txt"));
for (String linha : linhas) {
    String sentimento = ChatGPT_AnaliseSentimento.analisarSentimento(linha);
    System.out.println("Texto: " + linha);
    System.out.println("Sentimento: " + sentimento);
}
```

3. Salvando os resultados em CSV

```
try (PrintWriter writer = new PrintWriter("resultados.csv")) {
    writer.println("Texto,Sentimento");

    for (String linha : textos) {
        String sentimento = ChatGPT_AnaliseSentimento.analisarSentimento(linha);
        writer.println("\"" + linha + "\",\"" + sentimento + "\"");
    }
}
```

4. Percentual de Sentimento (prompt ajustado)

```
mensagem.put("content", "Analise o sentimento do seguinte texto e responda com uma porcentagem de positividade, negatividade ou neutralidade. Texto: '" + texto + "'");
```

Resposta esperada do ChatGPT:

Positivo: 75%, Neutro: 20%, Negativo: 5%

5. Expor como API REST com Spring Boot (controller básico)

```
@RestController
public class SentimentoController {

    @PostMapping("/sentimento")
    public String analisar(@RequestBody String texto) {
        return ChatGPT_AnaliseSentimento.analisarSentimento(texto);
    }
}
```

Você agora tem uma aplicação Java capaz de analisar sentimentos com a API do ChatGPT.

Com estrutura modular e simples, este projeto pode ser facilmente adaptado para aplicações reais em negócios, educação, suporte ao cliente e muito mais.

EducaCiência FastCode para a comunidade