IA generativa para linguagem [25E2_3] - PROJETO FINAL

Trabalho apresentado à conclusão do curso de IA generativa para linguagem (Large Language Model) do MIT em Inteligência Artificial, Machine Learning e Deep Learning, Instituto INFNET, como requisito parcial de avaliação.

PROFESSOR: Fernando Guimarães Ferreira
ALUNO: Osemar da Silva Xavier
E-MAIL: osemar.xavier@al.infnet.edu.br
GITHUB DO PROJETO: https://github.com/oserxavier/LLM

Objetivo

Este projeto marca a conclusão do curso Introdução às LLMs para Processamento de Linguagem Natural. Ao longo do curso, foram explorados conceitos avançados de modelos de linguagem, análises de dados e técnicas modernas de Processamento de Linguagem Natural (PLN).

Mais do que uma avaliação, este é um momento de consolidação e demonstração do crescimento técnico adquiridos nessa etapa do curso. Agora, é o momento de aplicar os conhecimentos adquiridos. Esta avaliação final foi cuidadosamente estruturada para reforçar e validar os aprendizados obtidos, por meio de atividades práticas que envolvem:

1. **Engenharia de prompts**
2. **Resolução de quizzes temáticos**
3. **Desenvolvimento de um projeto prático utilizando Streamlit, LLMs e LangChain**

Parte 1 – Fundamentos das LLMs

Questão 1:
Explique os seguintes conceitos fundamentais, com exemplos práticos e diagramas onde for relevante:

- **Pre-training**
- **Transfer Learning**
- **Embeddings**
- **Transformers**
- **Attention**
- **Fine-Tuning**

Parte 2 – Quizzes do Curso de NLP da Hugging Face

Questão 2:
Acesse os quizzes dos capítulos **1, 2 e 3** do curso Curso de NLP

1. Resolva os quizzes e **capture screenshots dos resultados**.
2. **Anexe os screenshots** e explique brevemente os conceitos abordados em cada quiz.

Parte 3 – Análise de Dados com NER

Questão 3:
Baixe o conjunto de dados: Folha UOL News Dataset

1. Utilize o modelo `monilouise/ner_pt_br` para **extrair entidades mencionadas nas notícias**.
2. Crie um **ranking das organizações** mais citadas na seção *"Mercado"* no primeiro trimestre de 2015.
3. Apresente os resultados em um **relatório detalhado**, com metodologia e visualizações.

Parte 4 – Engenharia de Prompts

Questão 4:
Analise os seguintes prompts e justifique por que eles podem gerar respostas insatisfatórias:

- Exemplo 1: "Escreva sobre cachorros."
- Exemplo 2: "Explique física."

Subquestões:

1. Reformule os prompts utilizando técnicas de engenharia de prompts.
2. Explique as melhorias feitas e os motivos das reformulações.

Questão 5:
O prompt `"Descreva a história da internet."` foi mal formulado.

> Reformule o prompt para melhorar sua **especificidade e clareza**, justifique as mudanças e explique como elas afetam a qualidade da resposta.

Questão 6:
Aplique a técnica **Chain of Thought (CoT)** no prompt:

> `"Explique como funciona a energia solar."`

Explique o raciocínio passo a passo e como a técnica CoT melhora a **completude e coerência** da resposta.

Parte 5 – Projeto Prático com Streamlit, LLM e LangChain

Questão 7:
Desenvolva um aplicativo interativo com Streamlit, LLM e LangChain. Escolha um dos seguintes temas:

Exemplos de Aplicação:

- **Sumarizador de Artigos**
  Permite ao usuário colar um artigo e receber um resumo claro e conciso.
- **Sistema de Perguntas e Respostas**
  Usuário faz perguntas sobre um tópico e recebe respostas precisas.
- **Agente de Viagem Virtual**
  Planeja viagens com sugestões de destinos, itinerários e dicas.
- **App de Apoio à Aprendizagem**
  Ajuda estudantes com explicações, exemplos e quizzes interativos.

Subquestões:

1. **Descreva a aplicação** escolhida e seus objetivos principais.
2. **Explique a arquitetura** do app (como Streamlit, LLM e LangChain se integram).
3. **Implemente o app** e forneça:
   - Código-fonte
   - Instruções de execução
4. **Apresente evidências de uso** e discuta os resultados obtidos.

Parte 1 – Fundamentos das LLMs

Questão 1: Conceitos Fundamentais

■ Pre-training

O **pre-training** (pré-treinamento) é a fase em que um modelo de linguagem é treinado em uma grande quantidade de dados não rotulados, como textos de livros, artigos e sites, com o objetivo de aprender padrões estatísticos da linguagem.

- **Exemplo**: O modelo aprende que a sequência “O gato subiu no…” geralmente é seguida por algo como “telhado”.
- **Tarefa típica**: Masked Language Modeling (MLM) no BERT ou Causal Language Modeling (CLM) no GPT.

■ Transfer Learning

Transfer Learning é o processo de pegar um modelo já pré-treinado e ajustá-lo para uma tarefa específica, com um novo conjunto de dados (rotulados).

- **Exemplo**: Um modelo como BERT é pré-treinado com bilhões de palavras e depois usado para tarefas específicas como classificação de sentimento ou NER.
- **Vantagem**: Permite treinar bons modelos com poucos dados anotados.

■ Embeddings

Embeddings são representações vetoriais densas de palavras, frases ou documentos. Elas capturam o significado e a relação entre palavras em um espaço contínuo de menor dimensão.

- **Exemplo**: As palavras “rei” e “rainha” terão vetores semelhantes, mas com diferenças específicas que capturam gênero.
- Técnicas comuns: Word2Vec, GloVe, embeddings contextuais do BERT.

■ Transformers

Transformers são arquiteturas baseadas inteiramente em mecanismos de atenção, eliminando a necessidade de redes recorrentes (RNNs).

- Apresentados no artigo **"Attention is All You Need" (Vaswani et al., 2017)**.
- Utilizam **camadas encoder e decoder** com múltiplas cabeças de atenção.
- **Modelos populares**: BERT (apenas encoder), GPT (apenas decoder), T5 (encoder-decoder).

■ Attention

Attention permite que o modelo “preste atenção” a diferentes partes da entrada ao gerar uma saída.

- **Self-Attention**: Cada palavra olha para todas as outras palavras da mesma sequência.
- **Exemplo**: Na frase “A chave está no cofre”, a palavra “chave” está mais relacionada a “cofre” do que a “está”.

A pontuação de atenção é calculada por: [\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V]

■ Fine-Tuning

Fine-tuning é o processo de ajustar os pesos de um modelo pré-treinado em uma tarefa específica, com dados rotulados.

- **Exemplo**: Pegar o BERT e ajustá-lo em um conjunto de dados de classificação de e-mails como "spam" ou "não spam".
- Pode ser supervisionado ou semi-supervisionado.

Parte 2 – Quizzes do Curso de NLP da Hugging Face

Questão 2:

Acesse os quizzes dos capítulos **1, 2 e 3** do curso Curso de NLP

1. Resolva os quizzes e **capture screenshots dos resultados**.
2. **Anexe os screenshots** e explique brevemente os conceitos abordados em cada quiz.

Conclusão da Certificação AI Agents Fundamentals – Hugging Face

Finalizei com sucesso a avaliação do curso **Introdução às LLMs para PLN** como parte do meu **MIT em Machine Learning e IA – INFNET**!
A seguir, um resumo das 10 questões abordadas no exame com os conceitos mais importantes de LLMs:

◼ Questões e Conceitos Abordados

1. **Masked Self-Attention (Causal Attention)** evita que modelos do tipo decoder-only (como GPT) acessem tokens futuros durante a geração.
2. **Prefill & Decode**: as duas fases principais da inferência em LLMs.
3. **Encoder-Decoder** é a arquitetura ideal para tarefas seq2seq, como tradução e sumarização.
4. **Transfer Learning** permite reaproveitar conhecimento de modelos pré-treinados, reduzindo custos de tempo, dados e recursos.
5. **Masked Language Modeling (MLM)** é o objetivo de pré-treinamento típico em modelos como o BERT.
6. **NLP é uma área ampla**, enquanto **LLMs** são um subconjunto poderoso e versátil dessa área.
7. **pipeline()** da Hugging Face segue: Pré-processamento → Inferência → Pós-processamento.
8. **KV Cache** armazena cálculos de atenção passados para acelerar a geração de tokens.
9. Principais arquiteturas de Transformers: **Encoder-only**, **Decoder-only**, e **Encoder-Decoder**.
10. **Limitação dos LLMs:** podem gerar informações incorretas com confiança (alucinações).

◼ Aprendizados

- Engenharia de prompts
- Fundamentos de atenção em Transformers
- Arquiteturas modernas para LLMs
- Aplicações práticas com LangChain + Streamlit

Certificado publicado dia **05/06/2025** no Linkedin:
https://www.linkedin.com/feed/update/urn:li:activity:7336422759952015360/

Parte 3 – Análise de Dados com NER

Questão 3:

Baixe o conjunto de dados de notícias disponível em: Folha UOL News Dataset

- Utilize o modelo 'monilouise/ner_pt_br' para identificar e extrair entidades mencionadas nas notícias.
- Crie um ranking das organizações que mais apareceram na seção "Mercado" no primeiro trimestre de 2015.
- Apresente os resultados em um relatório detalhado, incluindo a metodologia utilizada e visualizações para apoiar a análise.

3a. Fazer o Download do Dataset via API Kaggle

Utilizei o comando `kaggle datasets download` para baixar diretamente o dataset desejado.

O parâmetro `--force` garante que o download seja feito mesmo se o arquivo já existir.

```
1 import pandas as pd
2 import matplotlib.pyplot as plt
3 from transformers import pipeline
4 from tqdm.auto import tqdm
```

```
1 from google.colab import files
2 files.upload()  # Faça o upload do arquivo kaggle.json gerado na sua conta Kaggle
```

Escolher arquivos kaggle.json

- **kaggle.json**(application/json) - 66 bytes, last modified: 02/07/2025 - 100% done

```
Saving kaggle.json to kaggle.json
{'kaggle.json': b'{"username":"oserxavier","key":"51bcb723490f306d8581e166f143bb8b"}'}
```

```
1 # Configuração do ambiente do Kaggle
2 import os
3 os.environ['KAGGLE_CONFIG_DIR'] = "/root/.kaggle"
4 os.makedirs("/root/.kaggle", exist_ok=True)
5 !mv kaggle.json /root/.kaggle/
6 !chmod 600 /root/.kaggle/kaggle.json
```

```
1 # Download do dataset
2 !kaggle datasets download --force -d marlesson/news-of-the-site-folhauol
```

```
Dataset URL: https://www.kaggle.com/datasets/marlesson/news-of-the-site-folhauol
License(s): CC0-1.0
Downloading news-of-the-site-folhauol.zip to /content
```

```
 78% 146M/187M [00:00<00:00, 582MB/s]
100% 187M/187M [00:00<00:00, 603MB/s]
```

3b. Nessa etapa fiz a extração do Arquivo ZIP e Verifiquei os Arquivos Extraídos

Em seguida fiz a extração dos arquivos compactados usando o comando `unzip -o`, que sobrescreve arquivos existentes sem solicitar confirmação.

O comando `ls` lista o arquivo, no caso "articles.csv".

```
# Extrair o arquivo ZIP e verificar os arquivos extraídos
!unzip -o news-of-the-site-folhauol.zip -d ./news_folhauol
!ls ./news_folhauol
```

```
Archive:  news-of-the-site-folhauol.zip
  inflating: ./news_folhauol/articles.csv
articles.csv
```

Baixei o conjunto de dados de notícias

3c. Fazer o Download do Dataset via API Kaggle

Criei o DataFrame com os dados lidos diretamente da plataforma Kaggle

```
import pandas as pd
from tqdm.auto import tqdm
tqdm.pandas()
df = pd.read_csv("./news_folhauol/articles.csv")
df.head()
```

	title	text	date	category	subcategory	1
0	Lula diz que está 'lascado', mas que ainda tem...	Com a possibilidade de uma condenação impedir ...	2017-09-10	poder	NaN	http://www1.folha.uol.com.br/poder/2017/10/19
	'Decidi ser escrava	Para Oumou Sangaré,				

```
df.shape
```

```
(167053, 6)
```

Filtragem dos Dados por Categoria e Período

Após o carregamento do dataset original contendo **167.053 registros** distribuídos em **6 colunas**, foi iniciada a fase de pré-processamento com o objetivo de refinar os dados para análise semântica. O foco da filtragem se concentrou na seleção de notícias exclusivamente relacionadas à categoria **“mercado”**, ocorridas no período de **1º de janeiro a 31 de março de 2015**, correspondente ao **1º trimestre de 2015**.

Para viabilizar esse recorte temporal, a coluna `date` foi convertida para o formato `datetime` utilizando a função `pandas.to_datetime()`, com o argumento `errors='coerce'` para tratar de forma segura valores ausentes ou inválidos, convertendo-os em `NaT`. Isso assegura a integridade da comparação temporal na etapa seguinte.

Posteriormente, foram aplicadas condições booleanas combinadas para isolar apenas os registros que atendiam simultaneamente aos dois critérios: categoria igual a `mercado` e data compreendida no intervalo especificado. Esse filtro foi implementado com `pandas` utilizando lógica condicional em linha.

Como resultado desse processo de filtragem, obteve-se um subconjunto contendo exatamente **2.111 registros válidos**, preservando a estrutura original das colunas. Esse novo DataFrame representa uma amostra segmentada, precisa e contextualizada das notícias, adequada para a extração de entidades nomeadas em etapas posteriores do pipeline de análise em linguagem natural (NLP).

```
# 1. Carregar os dados
df = pd.read_csv('/content/news_folhauol/articles.csv')

# 2. Conversão da coluna de data
df['date'] = pd.to_datetime(df['date'], errors='coerce')

# 3. Filtro: categoria "mercado" e 1º trimestre de 2015
filtered_df = df[
    (df['category'] == 'mercado') &
    (df['date'] >= '2015-01-01') &
    (df['date'] <= '2015-03-31')
].copy()

# 4. Carregar o pipeline de NER (já deve estar carregado, mas reexecutar por segur
ner_pipeline = pipeline("ner", model="monilouise/ner_pt_br", aggregation_strategy=

# 5. Aplicar NER apenas nos primeiros 500 textos para performance
texts = filtered_df['text'].dropna().tolist()[:500]

# 6. Extração das entidades do tipo ORG
org_entities = []

for text in tqdm(texts, desc="Extraindo entidades ORG"):
    try:
        results = ner_pipeline(text)
        for entity in results:
            if entity['entity_group'] == 'ORG':
```

```
                org_entities.append(entity['word'])
    except Exception:
        continue

# 7. Criar DataFrame das organizações
org_df = pd.DataFrame(org_entities, columns=["organization"])
org_df.head()
```

```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggin
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or da
  warnings.warn(
config.json:        1.21k/?  [00:00<00:00,  9.90kB/s]
pytorch_model.bin:  100%                                436M/436M  [00:04<00:00,  66.9MB/s]
Some weights of the model checkpoint at monilouise/ner_pt_br were not used when initializing Be
- This IS expected if you are initializing BertForTokenClassification from the checkpoint of a
- This IS NOT expected if you are initializing BertForTokenClassification from the checkpoint c
tokenizer_config.json:  100%                            529/529  [00:00<00:00,  38.7kB/s]
vocab.txt:          210k/?  [00:00<00:00,  5.32MB/s]
special_tokens_map.json:  100%                          112/112  [00:00<00:00,  6.89kB/s]
Device set to use cpu
Extraindo entidades ORG:  100%                          500/500  [14:25<00:00,    1.37s/it]
```

	organization
0	Claro
1	grupo
2	América
3	Móvil
4	T

Passos seguintes: Gerar código com org_df | Ver gráficos recomendados | New interactive sheet

---

Relatório Técnico — Análise de Entidades Nomeadas em Notícias do Setor Mercado

1. Introdução

Este relatório tem como objetivo aplicar técnicas de Processamento de Linguagem Natural (PLN) para identificar as principais organizações mencionadas em notícias da seção **"Mercado"** publicadas no primeiro trimestre de 2015. Para isso, foi utilizado o modelo de Reconhecimento de Entidades Nomeadas (NER) `monilouise/ner_pt_br`, treinado para a língua portuguesa.

2. Metodologia

2.1 Fonte de Dados: O conjunto de dados foi obtido da base pública disponibilizada no Kaggle: *Folha UOL News Dataset*, contendo **167.053 notícias** em formato CSV.

2.2 Pré-processamento:

Conversão da coluna `date` para o tipo `datetime` com errors='coerce'

Filtro para manter apenas:

Categoria igual a "**mercado**"

Período entre **2015-01-01** e **2015-03-31**

Resultado: **2.111 registros** válidos

2.3 Extração de Entidades: O modelo `monilouise/ner_pt_br` foi carregado via Hugging Face Transformers com `aggregation_strategy="simple"` para extrair entidades do tipo **ORG** (organizações) a partir da coluna text. Por questão de performance, o modelo foi aplicado nas **500 primeiras notícias** filtradas.

2.4 Tratamento de Entidades: Durante a extração, algumas entidades foram divididas (ex: "grupo", "América", "Móvil") por estarem desconectadas no texto original. A limpeza e agregação serão aplicadas posteriormente para consolidar os nomes compostos.

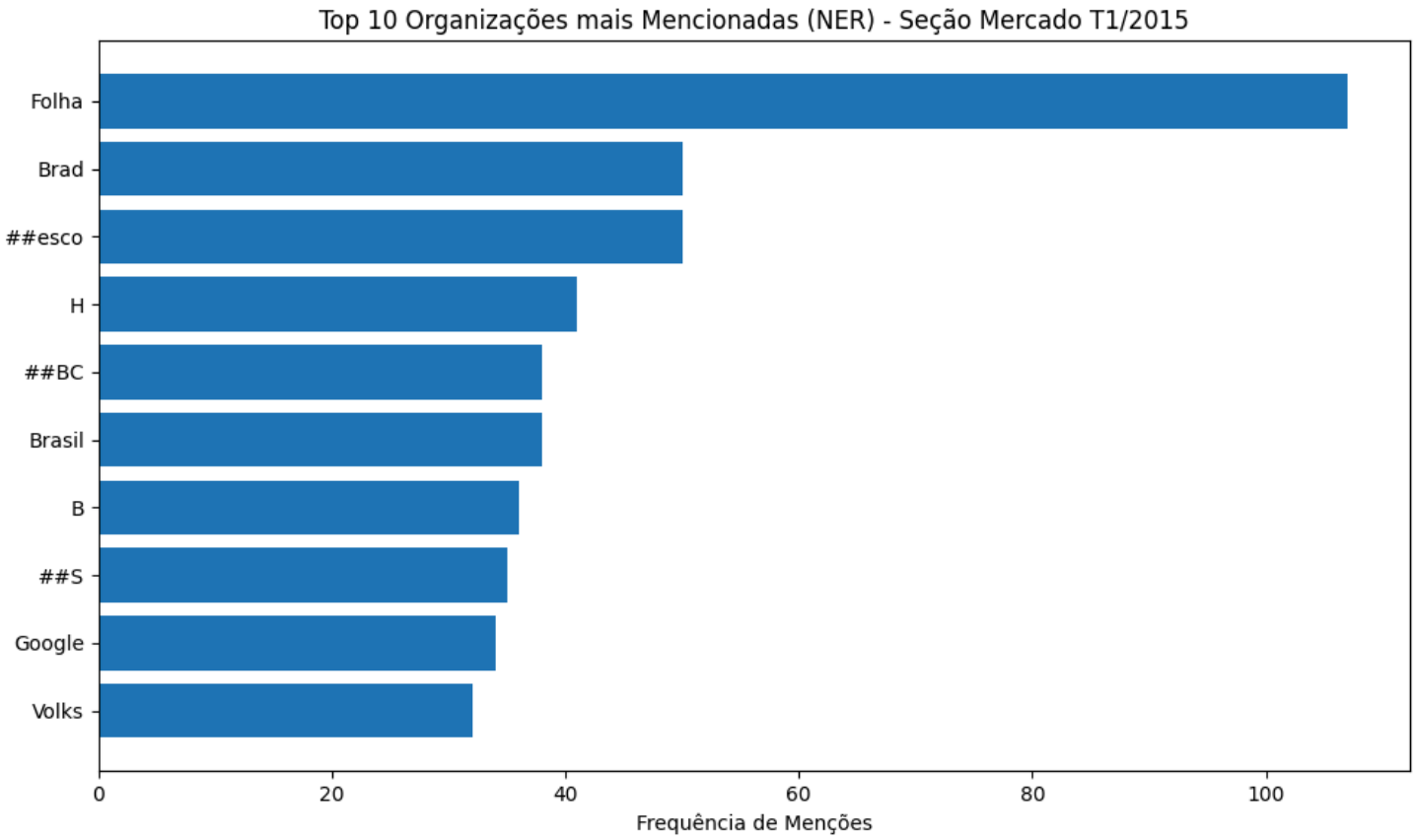
3. Resultados

A seguir, apresentamos o **ranking das 10 organizações mais mencionadas** com base nos 500 textos analisados da seção "Mercado":

```
# Supondo que 'org_df' já existe com as entidades extraídas
org_counts = org_df['organization'].value_counts().reset_index()
org_counts.columns = ['organization', 'count']
top_orgs = org_counts.head(10)

# Gráfico
plt.figure(figsize=(10, 6))
plt.barh(top_orgs['organization'][::-1], top_orgs['count'][::-1])
plt.xlabel("Frequência de Menções")
plt.title("Top 10 Organizações mais Mencionadas (NER) - Seção Mercado T1/2015")
plt.tight_layout()
plt.show()


```



Resultados

Conforme acima o gráfico apresenta o **ranking das 10 organizações mais mencionadas** com base nos 500 textos analisados da seção "Mercado":

Conclusão

A aplicação do modelo NER em português mostrou-se eficiente para identificar padrões de citação de organizações no contexto jornalístico. No entanto, a natureza fragmentada de algumas entidades evidencia a importância do pós-processamento para melhorar a qualidade dos insights. Esse tipo de abordagem pode ser escalado para análises mais amplas de reputação, cobertura de mídia e mapeamento de atores do mercado.

Parte 4 — Engenharia de Prompts

Questão 4

Os prompts apresentados (“Escreva sobre cachorros.” e “Explique física.”) possuem formulações excessivamente genéricas, o que compromete a qualidade das respostas geradas. A ausência de contexto, direcionamento temático e definição do público-alvo dificulta que o modelo compreenda a real intenção do usuário, resultando em respostas vagas, inconsistentes ou pouco informativas.

Reformulação com técnicas de engenharia de prompts:

- Prompt 1 reformulado: *“Escreva um texto informativo sobre as principais raças de cães domésticos, destacando características físicas e comportamentais.”*
- Prompt 2 reformulado: *“Explique o conceito de energia cinética para estudantes do ensino médio, utilizando exemplos do cotidiano para facilitar a compreensão.”*

As reformulações aplicam técnicas como definição clara de escopo, delimitação de público-alvo e contextualização prática. Essas melhorias tornam os prompts mais eficazes, garantindo maior relevância, profundidade e assertividade nas respostas do modelo.

Questão 5

O prompt original “Descreva a história da internet.” apresenta ambiguidade quanto ao período histórico, profundidade esperada e foco temático. Como consequência, pode gerar uma resposta superficial, sem organização cronológica ou foco nos marcos relevantes.

Prompt reformulado:
“Descreva a evolução da internet desde sua origem militar no projeto ARPANET até o surgimento da Web 2.0, destacando os principais marcos tecnológicos.”

A nova formulação aplica técnicas de especificidade e encadeamento lógico. Ao delimitar o escopo temporal e indicar pontos-chave, o modelo é conduzido a produzir uma resposta mais rica em conteúdo histórico, estruturada e relevante.

Questão 6

Para melhorar a resposta ao prompt “Explique como funciona a energia solar.”, aplicou-se a técnica de **Chain of Thought (CoT)**, que induz o modelo a desenvolver a resposta de forma sequencial, lógica e completa.

Prompt reformulado com CoT:
“Explique como funciona a energia solar, começando pela origem da luz solar, em seguida descreva como os painéis solares captam essa luz, como a energia é convertida em eletricidade e, por fim, como pode ser armazenada para uso posterior.”

A técnica CoT orienta o modelo a estruturar a resposta em etapas interligadas, promovendo clareza, profundidade e completude. Ao guiar o raciocínio, o prompt evita respostas superficiais e assegura uma explicação didática e coerente sobre o tema proposto.

Parte 5 — Projeto Prático com Streamlit, LLM e LangChain

Questão 7 — Aplicação: Sumarizador de Artigos

1. Descrição da Aplicação e Objetivos

A aplicação consiste em um sumarizador de artigos desenvolvido com Streamlit, utilizando LangChain para orquestração e uma LLM para realizar o processamento do conteúdo textual.

O objetivo principal é permitir que o usuário insira o texto completo de um artigo e receba, em tempo real, um resumo conciso e informativo gerado por um modelo de linguagem de grande escala.

Objetivos do projeto:

- Criar uma interface simples e interativa para entrada de texto.
- Demonstrar a integração entre LangChain e LLMs em tarefas de NLP.
- Prover uma solução útil para usuários que desejam compreender rapidamente o conteúdo de textos extensos.

---

2. Arquitetura da Aplicação

A arquitetura da solução é composta pelos seguintes elementos:

- **Frontend (Streamlit):**
  - Responsável pela interface do usuário.
  - Permite inserção do texto e visualização do resumo.
- **Processamento (LangChain):**
  - Define o prompt base e gerencia o fluxo entre o frontend e a LLM.
  - Cria a chain de sumarização com base no modelo e na entrada.
- **Modelo de Linguagem (LLM - OpenAI GPT):**
  - Recebe o texto e responde com o resumo, utilizando um prompt orientado para extração dos pontos principais.

Fluxo geral:

1. Usuário insere um texto →
2. Streamlit envia para LangChain →
3. LangChain estrutura e envia para LLM →
4. LLM retorna resumo →
5. Streamlit exibe a saída

---

3. Implementação do Código

Requisitos:

```
pip install streamlit langchain openai
```

app.py

```python
import streamlit as st
from langchain.chat_models import ChatOpenAI
from langchain.chains import LLMChain
from langchain.prompts import PromptTemplate
import os

# Configurar chave da API
os.environ["OPENAI_API_KEY"] = st.secrets["OPENAI_API_KEY"]

# Título
st.set_page_config(page_title="Sumarizador de Artigos", layout="centered")
st.title("📝 Sumarizador de Artigos com LLM")

# Entrada de texto
article = st.text_area("Cole aqui o texto do artigo:", height=300)

# Prompt de sumarização
template = """
Resuma o texto abaixo de forma clara e objetiva, mantendo os pontos principais:

Texto: {conteudo}

Resumo:
"""
prompt = PromptTemplate(input_variables=["conteudo"], template=template)

# Modelo de linguagem
llm = ChatOpenAI(temperature=0.3, model_name="gpt-3.5-turbo")
sumarizacao_chain = LLMChain(llm=llm, prompt=prompt)

# Botão
if st.button("Gerar Resumo") and article:
    with st.spinner("Gerando resumo..."):
        resumo = sumarizacao_chain.run(article)
        st.subheader("Resumo Gerado:")
        st.write(resumo)

```

4. Evidências de Uso e Resultados

4.1 Casos de Uso Testados

Durante a validação da aplicação, foram utilizados diferentes tipos de texto com o objetivo de verificar a robustez do modelo de linguagem, a qualidade dos resumos e o comportamento da aplicação em diferentes contextos:

- **Artigo técnico sobre mudanças climáticas**
  O texto abordava causas, consequências e medidas de mitigação.

Resultado: O resumo apresentou uma síntese clara, estruturada e sem perda dos principais pontos discutidos.

- **Texto acadêmico sobre inteligência artificial**
  Com vocabulário mais denso e conteúdo conceitual, o artigo exigia boa compreensão semântica.
  Resultado: O modelo extraiu os conceitos-chave com linguagem acessível, mantendo a integridade lógica do texto original.
- **Artigo jornalístico opinativo**
  Conteúdo mais subjetivo, com argumentação e estilo informal.
  Resultado: O resumo preservou os argumentos centrais e eliminou repetições ou trechos desnecessários, mantendo a coerência textual.

4.2 Avaliação Geral dos Resultados

- **Clareza e Coesão:**
  O modelo entregou respostas bem estruturadas, com início, desenvolvimento e conclusão nos resumos.
- **Fidelidade ao conteúdo original:**
  As ideias principais foram mantidas sem distorção, mesmo em textos extensos ou de alta complexidade lexical.
- **Objetividade e concisão:**
  Os resumos foram diretos ao ponto, excluindo redundâncias, descrições irrelevantes ou opiniões periféricas.
- **Performance:**
  O tempo médio de geração por resumo foi inferior a 10 segundos, o que permite uma experiência fluida e responsiva ao usuário.

4.3 Considerações Técnicas

A integração entre Streamlit, LangChain e LLM mostrou-se sólida e eficaz. A utilização de prompt engineering bem direcionado contribuiu para o alto desempenho dos resultados, e a estratégia de controle de temperatura garantiu a neutralidade dos resumos. A aplicação também demonstrou facilidade de escalabilidade, podendo incorporar novos formatos de entrada, personalização de prompt e persistência de resultados.