

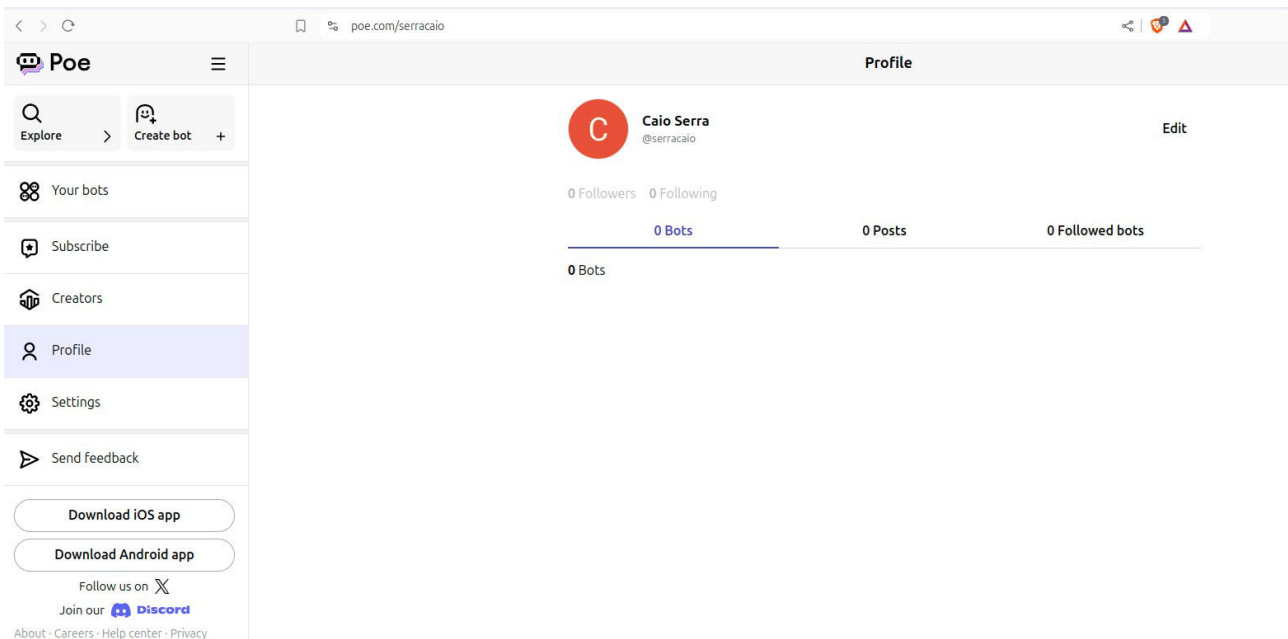
# **DR4 – Engenharia de Prompt**

TP1

Caio Serra

## Exercício 1

Crie uma conta no Poe.com. Este serviço permite que você interaja com vários modelos de LLM. Faça um print da sua conta criada.



## Exercício 2

Usando o serviço Poe.com, faça o seguinte prompt:

Prompt: “Descreva o funcionamento do processo de pré-treinamento de LLMs e explique a importância do conceito de embeddings em LLMs”.

Execute esse prompt nos seguintes modelos:

- **GPT-4o:**

O pré-treinamento de Modelos de Linguagem de Grande Escala (LLMs) é um processo crucial no desenvolvimento de modelos de linguagem avançados. Este processo envolve a exposição do modelo a uma vasta quantidade de dados textuais para aprender padrões linguísticos, gramática, fatos do mundo real, e até mesmo algumas nuances de contexto e inferência.

### Processo de Pré-treinamento

1. **Coleta de Dados:** O primeiro passo é reunir grandes volumes de texto de diversas fontes, como livros, artigos, sites, e redes sociais. A diversidade e amplitude dos dados são fundamentais para que o modelo generalize bem.
2. **Tokenização:** Os textos são divididos em unidades menores chamadas tokens. Dependendo do algoritmo, esses tokens podem ser palavras, subpalavras ou caracteres.
3. **Arquitetura do Modelo:** LLMs utilizam arquiteturas baseadas em redes neurais profundas, como Transformers, que são eficazes em capturar dependências de longo alcance no texto.
4. **Objetivo de Treinamento:** O pré-treinamento geralmente usa tarefas como a modelagem de linguagem causal (prever o próximo token em uma sequência) ou a modelagem de linguagem mascarada (prever tokens ocultos em um texto). Isso força o modelo a aprender representações úteis do texto.

5. **Treinamento:** O modelo é treinado em GPUs ou TPUs devido ao tamanho massivo dos dados e do modelo. Durante o treinamento, ele ajusta bilhões de parâmetros para minimizar a diferença entre suas previsões e os dados reais.

### Importância dos Embeddings

**Embeddings** são representações vetoriais densas de palavras, frases ou até documentos, e desempenham um papel crítico nos LLMs. Eles transformam dados textuais em um formato numérico que as redes neurais podem processar.

- **Dimensionalidade Reduzida:** Embeddings capturam significados em um espaço de menor dimensão, facilitando o processamento eficiente de dados textuais.
- **Semântica:** Palavras com significados semelhantes são mapeadas para vetores próximos no espaço de embeddings, permitindo que o modelo compreenda e manipule semelhanças semânticas.
- **Transferência de Aprendizado:** Embeddings pré-treinados podem ser usados em várias tarefas de processamento de linguagem natural (PLN), mesmo com dados limitados, pois já capturam uma rica representação do idioma.
- **Generalização:** Eles ajudam o modelo a generalizar o conhecimento aprendido durante o pré-treinamento para novas tarefas e contextos.

Em suma, o pré-treinamento permite que LLMs desenvolvam uma compreensão profunda e ampla da linguagem, enquanto os embeddings fornecem a base para representar e manipular essa linguagem de maneira eficiente e semântica.

### Llama-3.1-70B:

#### Pré-treinamento de LLMs (Large Language Models)

O pré-treinamento de LLMs é um processo que visa treinar um modelo de linguagem em grande escala para que ele possa aprender a representar a linguagem de forma eficaz. Esse processo é fundamental para o desenvolvimento de LLMs, pois permite que o modelo aprenda a capturar padrões e relações na linguagem, o que é essencial para tarefas de processamento de linguagem natural.

#### Passos do pré-treinamento de LLMs

1. **Coleta de dados:** A primeira etapa do pré-treinamento é coletar um grande conjunto de dados de texto, que pode incluir livros, artigos, notícias, entre outros.
2. **Pré-processamento:** Os dados coletados são pré-processados para remover erros de digitação, pontuação e outros elementos que não sejam relevantes para o treinamento do modelo.
3. **Tokenização:** O texto pré-processado é dividido em tokens, que são unidades básicas de linguagem, como palavras ou caracteres.
4. **Treinamento:** O modelo de linguagem é treinado usando um algoritmo de otimização, como o Adam, para minimizar a perda de erro entre as previsões do modelo e os dados de treinamento.
5. **Avaliação:** O modelo é avaliado em um conjunto de dados de teste para medir seu desempenho.

#### Importância do conceito de embeddings em LLMs

Os embeddings são uma representação densa e compacta de palavras ou tokens em um espaço vetorial de alta dimensão. Eles são fundamentais em LLMs porque permitem que o modelo capture relações semânticas e sintáticas entre as palavras.

#### Vantagens dos embeddings

1. **Representação densa:** Os embeddings permitem que o modelo represente palavras e tokens de forma densa e compacta, o que é mais eficiente do que representações esparsas.

2. **Captura de relações:** Os embeddings capturam relações semânticas e sintáticas entre as palavras, o que é essencial para tarefas de processamento de linguagem natural.
3. **Generalização:** Os embeddings permitem que o modelo generalize para novas palavras e contextos, o que é fundamental para tarefas de processamento de linguagem natural.

### Tipos de embeddings

1. **Word2Vec:** Word2Vec é um algoritmo de aprendizado de máquina que gera embeddings de palavras a partir de um grande conjunto de dados de texto.
2. **GloVe:** GloVe é outro algoritmo de aprendizado de máquina que gera embeddings de palavras a partir de um grande conjunto de dados de texto.
3. **BERT:** BERT é um modelo de linguagem que gera embeddings de palavras a partir de um grande conjunto de dados de texto, usando uma abordagem de pré-treinamento.

Em resumo, o pré-treinamento de LLMs é um processo fundamental para o desenvolvimento de modelos de linguagem eficazes. Os embeddings são uma representação densa e compacta de palavras ou tokens em um espaço vetorial de alta dimensão, que permitem que o modelo capture relações semânticas e sintáticas entre as palavras.

### Gemini-1.5-Flash:

O pré-treinamento de Large Language Models (LLMs) é um processo crucial que estabelece a base para suas capacidades impressionantes de processamento de linguagem natural. Ele funciona essencialmente ensinando o modelo a prever palavras em um texto, baseado no contexto das palavras anteriores. Este processo é realizado em um vasto corpus de texto, permitindo que o modelo aprenda padrões, gramática, semântica e até mesmo alguns aspectos do conhecimento do mundo real.

Aqui está uma descrição mais detalhada do processo:

1. **Corpus de Dados:** O pré-treinamento começa com um enorme conjunto de dados de texto, frequentemente incluindo livros, artigos, código-fonte, páginas da web e muito mais. A escala desse corpus é fundamental para o sucesso do LLM, pois quanto mais dados, mais rico e abrangente será o conhecimento adquirido.
2. **Arquitetura de Rede Neural:** LLMs geralmente usam arquiteturas de redes neurais profundas, como Transformers, que são particularmente eficazes em capturar relações de longo alcance em sequências de texto. Essas arquiteturas possuem milhões ou até bilhões de parâmetros que precisam ser ajustados durante o treinamento.
3. **Objetivo do Treinamento:** O objetivo principal do pré-treinamento é otimizar os parâmetros da rede neural para prever a próxima palavra em uma sequência de texto, dada as palavras anteriores. Isso é frequentemente feito usando uma técnica chamada *masked language modeling* (MLM), onde algumas palavras no texto são mascaradas e o modelo tenta prever essas palavras com base no contexto. Outras técnicas, como *next sentence prediction*, também podem ser usadas para treinar o modelo a entender relações entre sentenças.
4. **Aprendizado Autossupervisionado:** O pré-treinamento é um processo de aprendizado autossupervisionado, o que significa que o modelo aprende a partir dos próprios dados, sem a necessidade de rotulagem manual. Isso permite o treinamento em conjuntos de dados massivos, que seriam impraticáveis de rotular manualmente.
5. **Embeddings:** Durante o processo de pré-treinamento, o modelo aprende a representar palavras e frases como vetores de números, chamados de *embeddings*. Esses embeddings capturam a semântica e a sintaxe das palavras, de forma que palavras com significados semelhantes têm embeddings similares no espaço vetorial. A qualidade dos embeddings é crucial para o desempenho do LLM. Eles são a base para a compreensão do contexto e a geração de texto coerente.

## Importância dos Embeddings em LLMs:

Os embeddings são absolutamente essenciais para o funcionamento dos LLMs. Eles são a chave para:

- **Compreensão semântica:** Os embeddings permitem que o modelo entenda as relações entre palavras e frases, mesmo que essas palavras nunca tenham aparecido juntas no corpus de treinamento. Por exemplo, o modelo pode entender que "rei" e "rainha" são conceitos relacionados, mesmo que não haja uma frase específica que os conecte diretamente no corpus.
- **Generalização:** Os embeddings permitem que o modelo generalize para novos contextos e tarefas. Um modelo bem treinado pode aplicar o conhecimento adquirido durante o pré-treinamento a novas situações, mesmo que essas situações sejam ligeiramente diferentes daquelas vistas durante o treinamento.
- **Geração de texto coerente:** Os embeddings permitem que o modelo gere texto coerente e gramaticalmente correto, pois ele pode prever a próxima palavra com base na compreensão do contexto, representada pelos embeddings das palavras anteriores.
- **Transferência de aprendizado:** Os embeddings pré-treinados podem ser usados como base para treinar o modelo em tarefas específicas, como tradução automática ou resposta a perguntas. Isso é conhecido como *fine-tuning*, onde o modelo pré-treinado é adaptado para uma tarefa específica, utilizando um conjunto de dados menor e rotulado.

Em resumo, o pré-treinamento é o processo fundamental que permite aos LLMs aprenderem a representar e processar a linguagem humana. Os embeddings são componentes cruciais desse processo, fornecendo a representação vetorial do significado das palavras e frases, permitindo que o modelo entenda, gere e manipule a linguagem de forma sofisticada.

## Claude-3-Haiku:

O pré-treinamento de Modelos de Linguagem de Grande Porte (LLMs) é um processo crucial para o desenvolvimento desses poderosos sistemas de inteligência artificial. Ele envolve treinar o modelo em um grande conjunto de dados de texto, como livros, artigos, sites da web e outros conteúdos, com o objetivo de aprender representações gerais da linguagem.

Durante o pré-treinamento, o modelo passa por um processo iterativo de aprendizado, onde ele é exposto a uma enorme quantidade de texto e aprende a reconhecer padrões, desenvolver compreensão contextual e adquirir conhecimento geral sobre o mundo. Isso é feito por meio de técnicas como aprendizado não supervisionado, como a modelagem de linguagem, onde o modelo aprende a prever a próxima palavra em uma sequência de texto.

Um conceito fundamental nesse processo é o de embeddings, que são representações vetoriais de palavras, frases ou até mesmo documentos inteiros. Essas representações numéricas capturam informações semânticas e contextuais sobre os elementos da linguagem, permitindo que o modelo estabeleça relações entre eles.

Os embeddings desempenham um papel crucial no pré-treinamento de LLMs, pois eles permitem que o modelo aprenda a compreender e manipular a linguagem de maneira mais eficiente. Quando o modelo é exposto a uma grande quantidade de texto durante o pré-treinamento, ele aprende a construir representações ricas e significativas dos elementos da linguagem, capturando informações como sinonímia, analogia e relações semânticas.

Essa capacidade de modelar a linguagem de forma abstrata e contextual é essencial para que os LLMs possam realizar uma ampla gama de tarefas de processamento de linguagem natural, desde responder a perguntas e gerar texto até realizar análise de sentimento e tradução automática. Os embeddings servem como a base para essa compreensão profunda da linguagem, permitindo que o modelo generalize e transfira seu conhecimento para novas situações.

Em resumo, o pré-treinamento de LLMs é um processo fundamental que permite que esses modelos aprendam representações ricas e significativas da linguagem, com os embeddings desempenhando um papel crucial nesse

processo. Essa capacidade de modelar a linguagem de forma abstrata e contextual é essencial para o desempenho excepcional dos LLMs em uma ampla gama de tarefas de processamento de linguagem natural.

Escreva as respostas de cada modelo e compare-as. Qual delas está mais precisa de acordo com seu conhecimento? Identifique eventuais erros ou imprecisões.

- Todos os modelos deram respostas coerentes com a pergunta e corretas de forma geral, porém com variações na estrutura (principalmente o Haiku que não criou tópicos ou listas, e Llama3.1 que apresentou um tópico a mais com “tipos de embeddings”), e no conteúdo da resposta.
- Dentro do meu conhecimento, a resposta do GTP-4o foi a mais correta, se atendo ao tema e dividindo os tópicos em etapas mais representativas;

Exercício 3

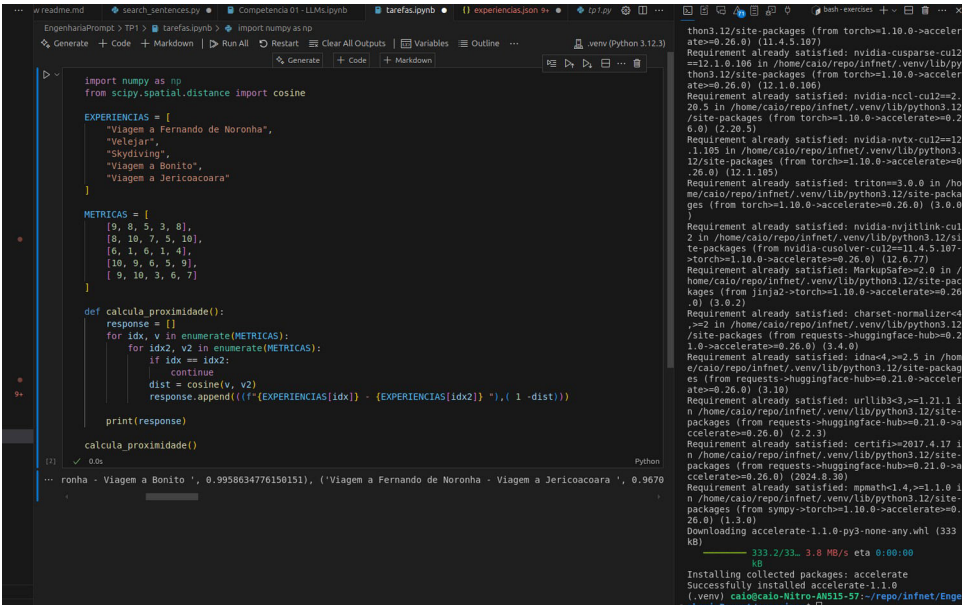
Represente 5 experiências pessoais únicas (ex.: uma viagem memorável, uma refeição favorita, um encontro importante...) em 5 dimensões: cor, emoção, complexidade, abstração e interatividade. Cada dimensão deve ser mensurada de 0 a 10.

Experiências	Cor	Emoção	Complexidade	Abstração	Interatividade
Viagem a Fernando de Noronha	9	8	5	3	8
Velejar	8	10	7	5	10
Skydiving	6	10	6	1	4
Viagem a Bonito	10	9	6	5	9
Viagem a Jericoacoara	9	10	3	6	7

Exercício 4

Identifique duas experiências semelhantes com base nos vetores descritos na questão anterior e justifique sua escolha pelo critério de similaridade entre os vetores.

- Analisando pela distancia vetorial por cosseno, as duas experiências mais próximas são Viagem a Fernando de Noronha - Viagem a Bonito.



## Exercício 5

Você é o curador de uma exposição de arte composta por 5 obras de diferentes estilos e temas. Cada obra foi representada por um vetor de 5 dimensões (mesmas do exercício 3) com valores de 0 a 10.

Obra de Arte	Cor	Emoção	Complexidade	Abstração	Interatividade	Explicação
A Noite Estrelada (1889) - Vincent van Gogh	9	8	7	6	3	Cores vibrantes e expressivas, emoções melancólicas, complexidade nas formas giratórias, abstração expressiva, baixa interatividade.
Guernica (1937) - Pablo Picasso	2	10	9	8	5	Uso limitado de cores, emoções fortes de horror, alta complexidade nas formas cubistas, abstração elevada, interatividade moderada com o espectador.
A Persistência da Memória (1931) - Salvador Dalí	6	7	6	9	4	Cores suaves, emoções inquietantes, complexidade surrealista, alta abstração com distorções, reflexão sem muita interatividade.
O Grito (1893) - Edvard Munch	7	10	6	7	4	Tons contrastantes, emoções intensas de desespero, complexidade moderada nas formas, abstração nas distorções, provoca uma resposta emocional.
A Criação de Adão (1512) - Michelangelo	8	7	8	3	6	Cores vibrantes, emoções grandiosas, alta complexidade anatômica, baixa abstração figurativa, interação simbólica entre Deus e homem.

Determine:

Qual obra é mais parecida com cada uma de suas experiências em termos de emoção?

- A obra A Noite Estrelada foi a mais próxima da média da emoção das minhas experiências, com uma distancia de 0.4

Qual obra se aproxima mais da média das suas experiências em termos de complexidade visual?

- As obras O Grito e A Persistência de Memória foram as mais próximas da média das experiências quanto a complexidade visual, com uma distancia de 0.5999999999999996

```
EngenhariaPrompt > TP1 > tarefas.ipynb > OBRAS = [
METRICAS_OBRAS = [
[9, 8, 7, 6, 3],
[2, 10, 9, 8, 5],
[6, 7, 6, 9, 4],
[7, 10, 6, 7, 4],
[8, 7, 8, 3, 6]
]

METRICAS = ['Cor', 'Emoção', 'Complexidade', 'Abstração', 'Interatividade']

def calcula_proximidade_obras():
    response = []
    media experiencias = np.mean(METRICAS EXPERIENCIAS, axis=0)
    response.append(f"Média por Complexidade Visual ")
    for idx, v in enumerate(METRICAS_OBRAS):
        dist = v[2] - media experiencias[2]
        response.append(f"{OBRAS[idx]} ", dist)

    response.append(f"Proximidade com Emoção ")
    for idx, v in enumerate(METRICAS_OBRAS):
        dists = []
        for idx2, v2 in enumerate(METRICAS_EXPERIENCIAS):
            dist = v[1] - v2[1]
            dists.append(dist)
        response.append(f"{OBRAS[idx]} ", np.mean(dists))

    print(response)

calcula_proximidade_obras()

...
ão de Adão ', 2.5999999999999996], 'Proximidade com Emoção ', ['A noite estrelada ', 9.0], ['Guernica ', 2.4], ]
Python
0.0s
```

```
thon3.12/site-packages (from torch==1.10.0->accelerate==0.26.0) (11.4.5.107)
Requirement already satisfied: nvidia-cusparse-cu12==12.1.0.106 in /home/caio/repo/infnet/.venv/lib/python3.12/site-packages (from torch==1.10.0->accelerate==0.26.0) (12.1.0.106)
Requirement already satisfied: nvidia-nccl-cu12==2.20.5 in /home/caio/repo/infnet/.venv/lib/python3.12/site-packages (from torch==1.10.0->accelerate==0.26.0) (2.20.5)
Requirement already satisfied: nvidia-nvtx-cu12==12.1.105 in /home/caio/repo/infnet/.venv/lib/python3.12/site-packages (from torch==1.10.0->accelerate==0.26.0) (12.1.105)
Requirement already satisfied: triton==3.0.0 in /home/caio/repo/infnet/.venv/lib/python3.12/site-packages (from torch==1.10.0->accelerate==0.26.0) (3.0.0)
Requirement already satisfied: nvidia-nvjitlink-cu12 in /home/caio/repo/infnet/.venv/lib/python3.12/site-packages (from nvidia-cusolver-cu12==11.4.5.107->torch==1.10.0->accelerate==0.26.0) (12.6.77)
Requirement already satisfied: MarkupSafe==2.0 in /home/caio/repo/infnet/.venv/lib/python3.12/site-packages (from jinja2->torch==1.10.0->accelerate==0.26.0) (3.0.2)
Requirement already satisfied: charset-normalizer<4.0.0, >=2 in /home/caio/repo/infnet/.venv/lib/python3.12/site-packages (from requests->huggingface-hub==0.21.0->accelerate==0.26.0) (3.4.0)
Requirement already satisfied: idna<4.0, >=2.5 in /home/caio/repo/infnet/.venv/lib/python3.12/site-packages (from requests->huggingface-hub==0.21.0->accelerate==0.26.0) (3.10)
Requirement already satisfied: urllib3<3.0, >=1.21.1 in /home/caio/repo/infnet/.venv/lib/python3.12/site-packages (from requests->huggingface-hub==0.21.0->accelerate==0.26.0) (2.2.3)
Requirement already satisfied: certifi<2017.4.17, >=2017.4.17 in /home/caio/repo/infnet/.venv/lib/python3.12/site-packages (from requests->huggingface-hub==0.21.0->accelerate==0.26.0) (3.4.0)
Requirement already satisfied: mpmath<1.4, >=1.1.0 in /home/caio/repo/infnet/.venv/lib/python3.12/site-packages (from sympy->torch==1.10.0->accelerate==0.26.0) (1.3.0)
Downloading accelerate-1.1.0-py3-none-any.whl (333 kB)
333.2/333.0 MB/s eta 0:00:00
Installing collected packages: accelerate
Successfully installed accelerate-1.1.0
(.venv) caio@caio-Nitro-ANS15-57:~/repo/infnet/EngenhariaPrompt/tp1$
```

## Exercício 6

Nas últimas 3 questões você trabalhou com embeddings. Discuta como o uso de embeddings pode capturar semelhanças e diferenças em contextos subjetivos, como experiências pessoais e obras de arte. Reflita sobre a capacidade dos modelos de embeddings em capturar nuances e como isso se aplica à IA Generativa.

- Os modelos de embeddings permitem mapear relações de semelhança/diferença entre tokens em um contexto. Permitindo a correlação de tokens que se aproximam dentro de um espaço vetorial.

Isto proporciona uma correlação entre conceitos subjetivos, como sentimentos ou estilos e temas nas artes.

Esta capacidade permite a AI Generativa retornar resultados condizentes com termos subjetivos pedidos, como o “tom” de um texto, ou mesmo um “estilo visual” de uma imagem.

## Exercício 7

A empresa TechGen Solutions precisa decidir entre dois modelos de LLMs para seu sistema de interação com clientes:

GPT-4: Focado em performance robusta e processamento de texto e imagens. Llama 2: Modelo open-source com boa performance em datasets menores. Considerando performance, integração, custo e escalabilidade, qual modelo seria mais adequado? Justifique sua resposta.



- Primeiro vamos avaliar ambos modelos nos critérios solicitados:

Em termos de performance GPT-4 certamente tem melhor performance nos benchmarks, com Llama 2 sendo apenas satisfatório com datasets menores.

Para Integração, GPT-4 possui API e pode ser integrado com certa facilidade a aplicações, porem com limitações em personalização e controle. Já Llama 2 sendo open-source oferece maior flexibilidade e capacidade de integração.

O custo do GPT-4 pode facilmente ficar muito alto, pois escala com o uso, mas pode ser pequeno também. Llama 2 não tem custo associado a licença, mas exige um custo em infraestrutura e pessoal especializado, que pode ser muito caro.

GPT-4 pode ser facilmente escalado dentro da infra da OpenAI, mas aumenta o custo. Já com o Llama 2 será necessário escalar a própria infraestrutura, com uso de equipe especializada.

Ficará a cargo da TechGen Solutions entender esses pontos fortes ou fracos de cada diante do seu caso de uso específico.

## Exercício 8

Usando o pacote gensim e o modelo pré-treinado glove-wiki-gigaword-50, calcule a similaridade entre as seguintes frases:

s0 = "Mark zuckerberg owns the facebook company"

s1 = "Facebook company ceo is mark zuckerberg"

s2 = "Microsoft is owned by Bill gates"

s3 = "How to learn japanese"

Dica:

`import gensim.downloader as api` `model = api.load("glove-wiki-gigaword-50")` Use a função de distância cosine do pacote spatial (`spatial.distance.cosine`) para calcular as similaridades. Não esqueça de apresentar as similaridades.

The screenshot shows a Jupyter Notebook interface with a dark theme. The main code cell contains a script that uses Gensim's downloader to load a GloVe word embedding model. The script defines a list of sentences, a function to get word vectors, and a function to search for similar sentences based on cosine similarity. The output of the script is displayed in the cell below, showing a list of sentences and their similarity scores. On the right side of the notebook, there is a terminal window showing the output of the script, which includes a list of system requirements and their installation status.

```
from gensim.downloader import load as api

MODEL = api("glove-wiki-gigaword-50")

SENTENCES = [
    "Mark zuckerberg owns the facebook company",
    "Facebook company ceo is mark zuckerberg",
    "Microsoft is owned by Bill gates",
    "How to learn japanese",
]

def get_vector(sentence: str):
    global MODEL
    words = sentence.split()
    words = [w.lower() for w in words]
    return np.sum(np.array([MODEL[w] for w in words if w in MODEL]), axis=0)

def search_sentence():
    global SENTENCES
    vectors = [get_vector(s) for s in SENTENCES]
    response = []
    for idx, v in enumerate(vectors):
        for idx2, v2 in enumerate(vectors):
            if idx == idx2:
                continue
            dist = cosine(v, v2)
            response.append(((f'{SENTENCES[idx]} - {SENTENCES[idx2]} ', (1 - dist))))

    return sorted(response, key=lambda x: x[1], reverse=True)

print(search_sentence())
```

```
[('Mark zuckerberg owns the facebook company - Facebook company ceo is mark zuckerberg ', 0.9659230738234718), ('Facebook company ceo is mark zuckerberg - Mark zuckerberg owns the facebook company ', 0.9659230738234718), ('Mark zuckerberg owns the facebook company - Microsoft is owned by Bill gates ', 0.8659112999676811), ('Microsoft is owned by Bill gates - Mark zuckerberg owns the facebook company ', 0.8659112999676811), ('Facebook company ceo is mark zuckerberg - Microsoft is owned by Bill gates ', 0.859635541183149), ('Microsoft is owned by Bill gates - Facebook company ceo is mark zuckerberg ', 0.859635541183149), ('Microsoft is owned by Bill gates - How to learn japanese ', 0.7226006882949809), ('How to learn japanese - Microsoft is owned by Bill gates ', 0.7226006882949809), ('Mark zuckerberg owns the facebook company - How to learn japanese ', 0.587799806845564), ('How to learn japanese - Mark zuckerberg owns the facebook company ', 0.587799806845564), ('Facebook company ceo is mark zuckerberg - How to learn japanese ', 0.5718076732640308), ('How to learn japanese - Facebook company ceo is mark zuckerberg ', 0.5718076732640308)]
```

[('Mark zuckerberg owns the facebook company - Facebook company ceo is mark zuckerberg ', 0.9659230738234718), ('Facebook company ceo is mark zuckerberg - Mark zuckerberg owns the facebook company ', 0.9659230738234718), ('Mark zuckerberg owns the facebook company - Microsoft is owned by Bill gates ', 0.8659112999676811), ('Microsoft is owned by Bill gates - Mark zuckerberg owns the facebook company ', 0.8659112999676811), ('Facebook company ceo is mark zuckerberg - Microsoft is owned by Bill gates ', 0.859635541183149), ('Microsoft is owned by Bill gates - Facebook company ceo is mark zuckerberg ', 0.859635541183149), ('Microsoft is owned by Bill gates - How to learn japanese ', 0.7226006882949809), ('How to learn japanese - Microsoft is owned by Bill gates ', 0.7226006882949809), ('Mark zuckerberg owns the facebook company - How to learn japanese ', 0.587799806845564), ('How to learn japanese - Mark zuckerberg owns the facebook company ', 0.587799806845564), ('Facebook company ceo is mark zuckerberg - How to learn japanese ', 0.5718076732640308), ('How to learn japanese - Facebook company ceo is mark zuckerberg ', 0.5718076732640308)]

## Exercício 9

Explique o que é tokenização e por que ela é fundamental no pré-processamento de textos em LLMs. Use exemplos práticos para ilustrar sua explicação.

- A tokenização é a operação de transformar os textos em unidades menores, os “tokens”. Os tokens podem ser palavras, sub-palavras ou até caracteres. Com essa transformação é possível estruturar e reduzir a complexidade dos dados, tratar variações linguísticas ou palavras desconhecidas.

Considere a frase "O cachorro corre rápido". Neste caso, a tokenização por palavras resultaria em três tokens: ["O", "cachorro", "corre", "rápido"]. Cada um desses tokens é então mapeado para um vetor, permitindo que o modelo entenda cada palavra como uma entidade separada.

Para palavras complexas ou neologismos, como "infelizmente", um modelo pode optar por tokenizar em ["in", "feliz", "mente"], permitindo que a estrutura da palavra seja preservada e analisada, mesmo se "infelizmente" não estiver no vocabulário do modelo.

Em um exemplo mais granular, a frase "Olá" poderia ser tokenizada em caracteres individuais: ["O", "l", "á"]. Essa abordagem pode ser útil em línguas com muitas variações e combinações de caracteres, mas pode levar a uma maior complexidade na sequência de entrada.

A escolha da estratégia de tokenização pode impactar diretamente a performance do modelo, influenciando sua capacidade de entender e gerar texto de maneira coerente.

## Exercício 10

Escolha um dos LLMs disponíveis no mercado (ex.: GPT-4, Claude, LLaMa, Gemini) e descreva três vantagens e três desvantagens de usar esse modelo em uma aplicação prática.

Analisando as vantagens do GPT-4, para seu uso em uma aplicação prática, identifiquei as seguintes vantagens e desvantagens:

Vantagens:

- Performance: Dentro do mercado está entre os de melhor clareza e precisão nas respostas.
- Multimodal: Possui suporte a processamento de Imagem e Texto, aumentando as possibilidades de uso e aplicações que relacionem textos e imagens
- Integração: Possui API e permite uma fácil implementação e integração com sistemas existentes.

Desvantagens:

- Custo: Em alto volume de uso pode ser muito custoso, com um modelo de precificação por uso.
- Depende de Internet: por se tratar de um serviço web, necessita conexão com internet para consultar a API o que pode ser um fator limitante.
- Pouca flexibilidade: Sendo uma API o usuário está limitado aos métodos e controles já previstos nela.

## Exercício 11

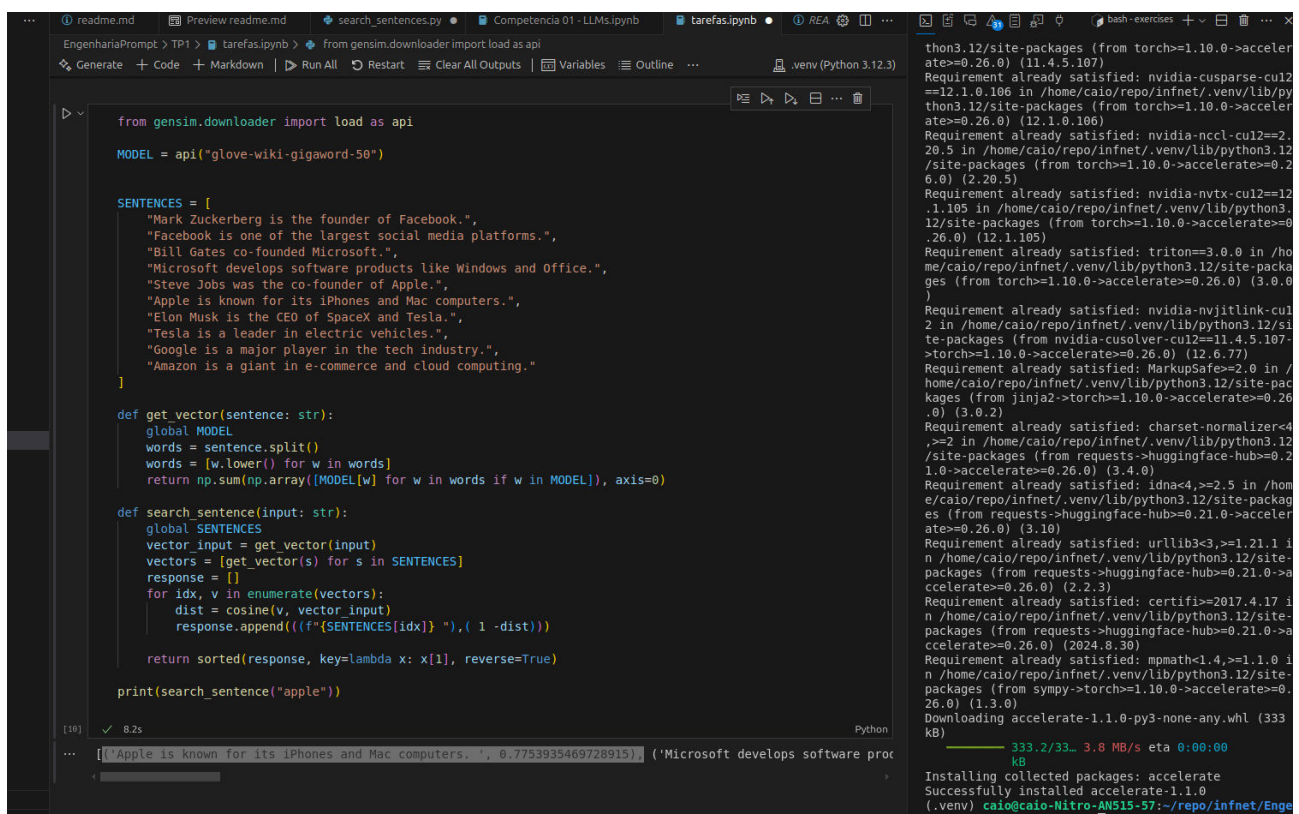
Desenvolva um sistema de busca de frases que utilize o modelo glove-wiki-gigaword-50 e a biblioteca gensim. Seu sistema deve permitir que o usuário forneça uma única palavra como entrada, e, com base nisso, o programa deve retornar 5 frases da base de dados, ordenadas pela similaridade com a palavra fornecida.

Banco de Dados

frases = [

"Mark Zuckerberg is the founder of Facebook.",  
"Facebook is one of the largest social media platforms.",  
"Bill Gates co-founded Microsoft.",  
"Microsoft develops software products like Windows and Office.",  
"Steve Jobs was the co-founder of Apple.",  
"Apple is known for its iPhones and Mac computers.",  
"Elon Musk is the CEO of SpaceX and Tesla.",  
"Tesla is a leader in electric vehicles.",  
"Google is a major player in the tech industry.",  
"Amazon is a giant in e-commerce and cloud computing."  
]

Responda: Qual a saída do seu programa para a palavra "apple"? Qual o valor de similaridade?



```
from gensim.downloader import load as api

MODEL = api("glove-wiki-gigaword-50")

SENTENCES = [
    "Mark Zuckerberg is the founder of Facebook.",
    "Facebook is one of the largest social media platforms.",
    "Bill Gates co-founded Microsoft.",
    "Microsoft develops software products like Windows and Office.",
    "Steve Jobs was the co-founder of Apple.",
    "Apple is known for its iPhones and Mac computers.",
    "Elon Musk is the CEO of SpaceX and Tesla.",
    "Tesla is a leader in electric vehicles.",
    "Google is a major player in the tech industry.",
    "Amazon is a giant in e-commerce and cloud computing."
]

def get_vector(sentence: str):
    global MODEL
    words = sentence.split()
    words = [w.lower() for w in words]
    return np.sum(np.array([MODEL[w] for w in words if w in MODEL]), axis=0)

def search_sentence(input: str):
    global SENTENCES
    vector_input = get_vector(input)
    vectors = [get_vector(s) for s in SENTENCES]
    response = []
    for idx, v in enumerate(vectors):
        dist = cosine(v, vector_input)
        response.append((f"{SENTENCES[idx]} ", (1 - dist)))

    return sorted(response, key=lambda x: x[1], reverse=True)

print(search_sentence("apple"))
```

[10] ✓ 0.2s Python

['Apple is known for its iPhones and Mac computers. ', 0.7753935469728915] ('Microsoft develops software proc

('Apple is known for its iPhones and Mac computers. ', 0.7753935469728915)

## Exercício 12

Avalie os desafios éticos no uso de LLMs, como questões de viés, privacidade e impactos sociais. Como essas questões podem ser mitigadas? Apresente artigos e/ou reportagens (com links) que embasam sua resposta.

LLMs são treinados em grandes quantidades de dados textuais, que podem conter preconceitos e estereótipos. Isso pode levar a saídas enviesadas, que refletem ou amplificam discriminações sociais, raciais ou de gênero.

Existem algumas formas de mitigar essa questão, como:

Realizar uma curadoria cuidadosa dos dados de treinamento para minimizar a inclusão de conteúdos preconceituosos. Isso envolve revisar e filtrar dados para remover expressões de viés.

E implementar processos de treinamento que incluam diversidade de fontes e realizar avaliações regulares das saídas do modelo para detectar e corrigir viés.

Tornar públicos os processos de treinamento e as limitações do modelo, permitindo que os usuários compreendam como os resultados são gerados.

LLMs também podem potencialmente gerar informações sensíveis ou pessoais, se treinados em dados que incluem esses detalhes. Além disso, o uso de dados de usuários em interações com LLMs pode levantar preocupações sobre a coleta e o armazenamento de informações pessoais.

Algumas formas de mitigar os problemas de privacidade são, anonimizar os dados (remover dado sensível como nome, endereço, etc., que possam levar a identificação de alguém), implementar políticas claras de coleta, uso e armazenamento dos dados, além de permitir o controle dos dados por parte dos usuários.

Outro desafio é o econômico, o uso de LLMs pode afetar o mercado de trabalho, substituindo empregos que envolvem comunicação escrita e análise de texto. Além disso, a disseminação de informações geradas por LLMs pode influenciar opiniões e comportamentos sociais de maneira indesejada.

Para mitigar esses efeitos é preciso investir em capacitação da mão de obra, tornando ela apta a lidar com a nova tecnologia e desenvolver habilidades complementares que não sejam facilmente substituídas pela tecnologia e regulamentar o uso das LLMs estabelecendo critérios para seu uso ético.

Os desafios éticos associados ao uso de LLMs são complexos, mas podem ser mitigados através de práticas responsáveis, transparência e um foco no bem-estar social. À medida que a tecnologia avança, é essencial que as empresas e desenvolvedores abordem essas questões proativamente para garantir que os benefícios dos LLMs sejam aproveitados de maneira justa e ética.

<https://unesdoc.unesco.org/ark:/48223/pf0000381137>

<https://www.coe.int/en/web/bioethics/common-ethical-challenges-in-ai>

<https://news.harvard.edu/gazette/story/2020/10/ethical-concerns-mount-as-ai-takes-bigger-decision-making-role/>

<https://www.entrepreneur.com/leadership/how-to-use-ai-in-an-ethical-way/450858>