

**INSTITUTO INFNET**

ESCOLA SUPERIOR DE TECNOLOGIA

GRADUAÇÃO EM CIÊNCIA DE DADOS



Desenvolvimento Front-End com Python (com  
Streamlit) [24E3\_1]

**TP 1**

Alunos: Gustavo Carneiro Alves.

2024

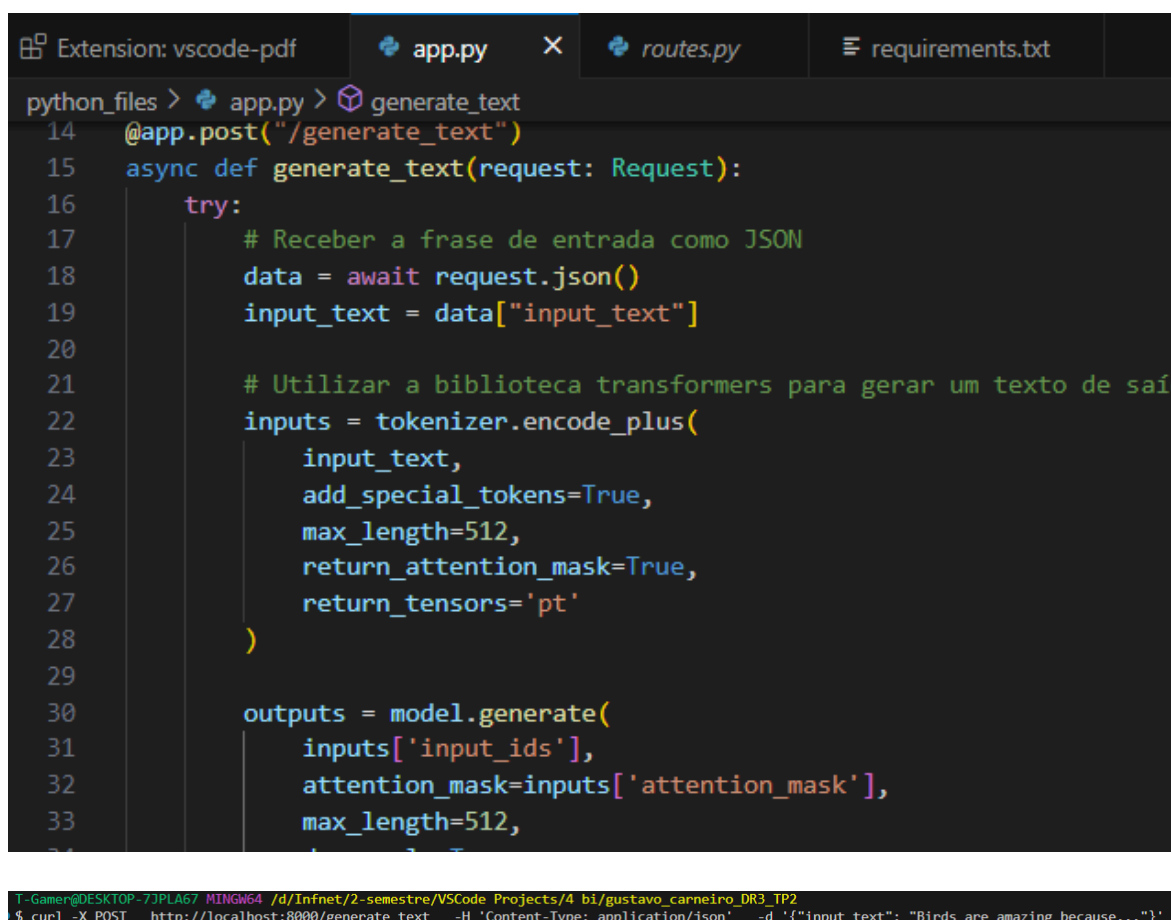
**Questão 1:** Crie uma aplicação simples em FastAPI que utilize o modelo GPT-2 da HuggingFace para gerar textos a partir de uma entrada fornecida via requisição HTTP.

**O aplicativo deve:**

- Receber uma frase de entrada como JSON.
- Utilizar a biblioteca transformers do HuggingFace para gerar um texto de saída.
- Retornar o texto gerado em uma resposta HTTP.

**O que é esperado:**

- O aplicativo deve gerar uma continuação de texto a partir de uma frase de entrada e retornar a resposta formatada como JSON.



```
Extension: vscode-pdf  app.py  routes.py  requirements.txt
python_files > app.py > generate_text
14 @app.post("/generate_text")
15 async def generate_text(request: Request):
16     try:
17         # Receber a frase de entrada como JSON
18         data = await request.json()
19         input_text = data["input_text"]
20
21         # Utilizar a biblioteca transformers para gerar um texto de saída
22         inputs = tokenizer.encode_plus(
23             input_text,
24             add_special_tokens=True,
25             max_length=512,
26             return_attention_mask=True,
27             return_tensors='pt'
28         )
29
30         outputs = model.generate(
31             inputs['input_ids'],
32             attention_mask=inputs['attention_mask'],
33             max_length=512,
```

```
T-Gamer@DESKTOP-73PLA67 MINGW64 /d/Infnet/2-semester/VSCode Projects/4 bi/gustavo_carneiro_DR3_TP2
$ curl -X POST http://localhost:8000/generate_text -H 'Content-Type: application/json' -d '{"input_text": "Birds are amazing because..."}
```

```
["generated_text": "Birds are amazing because... I mean, we don't have these great white penguins. All these birds are amazing. We have all these amazing penguins and all these birds are amazing. And some birds are like, 'Oh, this is the best thing ever. I don't know what's going on.' Well we have the penguin penguin bird, the one in the room where the lady is standing and she's talking and I just kind of thought, what do they do for a living, right? And then you know that the lady is trying to talk to the bird and she says, 'This bird is amazing.' And I thought, that's cool. It's pretty cool that we've got this bird that will be our best friend, no matter what kind of place it is. But we don't have that kind of friend.\n\nAVC: But it's also a penguin. Like in the original, 'No one likes a penguin.' And it doesn't get as much attention as it would like. But as you go up into the penguin penguin, you'll see these adorable little penguins that can fly pretty quickly. And when we talk about the way we communicate in these birds, we don't really talk to them that much. And so you look at some bird that is pretty nice, and we'll talk to it a little bit more.\n\nMAY: And it's not like it's going to be all over the place. It just kind of seems like we have to be careful, and be careful where we approach. So if I look at one of these little penguins and I say, 'I like it!' or I'm like, 'You like it? This is nice.' That's a cute penguin that I'm really happy about. They'll really listen. That's the best way I can use the word.\n\nAVC: You've even said that there's so much joy in meeting some penguins that they're literally just like, they can be our best friends. But what about the fact that these are the true penguin's and not just birds of prey?\n\nMAY: Well, because you know, there's some birds out there who really love it if we have that kind of relationship with them. And just because something goes up and something goes down, it just doesn't mean you're going to get those penguins that you can look at, like you just brought up in the show where one"]
```

**Questão 2:** Crie um aplicativo FastAPI que utiliza o modelo de tradução Helsinki-NLP/opus-mt-en-fr da HuggingFace para traduzir textos do inglês para o francês.

**A aplicação deve:**

- Receber um texto em inglês via uma requisição HTTP.
- Traduzir o texto para o francês utilizando o modelo de tradução.
- Retornar o texto traduzido em uma resposta JSON.

**O que é esperado:**

- A API deve receber um texto em inglês e retornar sua tradução para o francês, processando tanto frases curtas quanto textos mais longos.

```

@app.post("/translate")
async def translate(request: Request):
    # Receber o texto em inglês via requisição HTTP
    data = await request.json()
    text_en = data["text"]

    # Traduzir o texto para o francês utilizando o modelo de tradução
    inputs = tokenizer.encode_plus(
        text_en,
        add_special_tokens=True,
        max_length=512,
        return_attention_mask=True,
        return_tensors='pt'
    )

    outputs = model.generate(
        inputs['input_ids'],
        attention_mask=inputs['attention_mask'],
        max_length=512,
        do_sample=True,
        top_k=50,
        top_p=0.95
    )

    # Retornar o texto traduzido em uma resposta JSON
    text_fr = tokenizer.decode(outputs[0], skip_special_tokens=True)
    return {"translation": text_fr}

```

```

T-Gamer@DESKTOP-73PLA67 MINGW64 /d/Infnet/2-semester/VSCode Projects/4 bi/gustavo_carneiro_DR3_TP2
$ curl -X POST \
$ curl -X POST \
> http://localhost:8000/translate \
> -H 'Content-Type: application/json' \
> -d '{"text": "Hello, how are you?"}'
curl: (56) Recv failure: Connection was reset

T-Gamer@DESKTOP-73PLA67 MINGW64 /d/Infnet/2-semester/VSCode Projects/4 bi/gustavo_carneiro_DR3_TP2
$ curl -X POST http://localhost:8000/translate -H 'Content-Type: application/json' -d '{"text": "Hello, how are you?"}'
{"translation": "Bonjour, comment allez-vous ?"}

```

**Questão 3:** Com base na API desenvolvida na Questão 2 (Parte1), explique as principais limitações do modelo de tradução utilizado.

**Enumere e discuta:**

- Limitações quanto à precisão da tradução.
- Desafios de tempo de resposta e desempenho em grande escala.
- Restrições de custo e escalabilidade.
- Limitações na tradução de gírias, expressões idiomáticas ou linguagem de contexto.

### Limitações quanto à precisão da tradução

- Erros de tradução: O modelo de tradução pode cometer erros de tradução, especialmente em casos de frases complexas ou com nuances culturais.
- Falta de contexto: O modelo de tradução pode não ter acesso ao contexto em que a frase foi escrita, o que pode levar a traduções imprecisas.
- Limitações de vocabulário: O modelo de tradução pode não ter um vocabulário completo, o que pode levar a traduções imprecisas ou inexatas.

### Desafios de tempo de resposta e desempenho em grande escala

- Tempo de resposta: O modelo de tradução pode demorar mais tempo para processar frases longas ou complexas, o que pode afetar a experiência do usuário.
- Desempenho em grande escala: O modelo de tradução pode não ser escalável para lidar com um grande volume de requisições simultâneas, o que pode afetar a performance da API.

### Restrições de custo e escalabilidade

- Custo de treinamento: O modelo de tradução pode exigir um grande volume de dados de treinamento, o que pode ser caro e demorado.
- Custo de infraestrutura: O modelo de tradução pode exigir uma infraestrutura robusta para lidar com o volume de requisições, o que pode ser caro.

Limitações de escalabilidade: O modelo de tradução pode não ser escalável para lidar com um grande volume de requisições simultâneas, o que pode afetar a performance da API.

- Limitações na tradução de gírias, expressões idiomáticas ou linguagem de contexto
- Gírias e expressões idiomáticas: O modelo de tradução pode não ser capaz de traduzir gírias e expressões idiomáticas de forma precisa, pois elas podem ter significados diferentes em diferentes contextos.
- Linguagem de contexto: O modelo de tradução pode não ser capaz de traduzir linguagem de contexto de forma precisa, pois ela pode depender do contexto em que a frase foi escrita.

**Questão 4:** Com base no modelo GPT-2 utilizado na Questão 1 (Parte 1), explique as principais limitações do modelo no contexto da geração de texto.

**Discuta:**

- A coerência do texto gerado.
- Possíveis falhas ou incoerências geradas por LLMs.
- Desempenho e questões de latência.
- Limitações na geração de conteúdo apropriado.

A coerência do texto gerado

- Falta de coerência: O modelo GPT-2 pode gerar texto que não é coerente ou que não faz sentido em determinados contextos.
- Incoerências lógicas: O modelo pode gerar texto que contém incoerências lógicas, como contradições ou afirmações que não são verdadeiras.

Possíveis falhas ou incoerências geradas por LLMs

- Erros de linguagem: O modelo pode cometer erros de linguagem, como erros de gramática, ortografia ou pontuação.
- Incoerências semânticas: O modelo pode gerar texto que contém incoerências semânticas, como uso de palavras com significados diferentes do que o esperado.
- Falta de nuances: O modelo pode não ser capaz de capturar nuances de linguagem, como ironia, sarcasmo ou humor.

Desempenho e questões de latência

- Tempo de geração: O modelo pode demorar mais tempo para gerar texto, especialmente para textos longos ou complexos.
- Desempenho em grande escala: O modelo pode não ser escalável para lidar com um grande volume de requisições simultâneas, o que pode afetar a performance da API.
- Latência: O modelo pode ter latência, o que pode afetar a experiência do usuário.

## Limitações na geração de conteúdo apropriado

- Falta de conhecimento de domínio: O modelo pode não ter conhecimento de domínio específico, o que pode levar a geração de texto que não é apropriado para o contexto.
- Falta de compreensão de nuances culturais: O modelo pode não ser capaz de compreender nuances culturais, o que pode levar a geração de texto que é ofensivo ou inapropriado.
- Limitações na geração de conteúdo criativo: O modelo pode não ser capaz de gerar conteúdo criativo, como histórias ou poemas, que são apropriados para o contexto.

### Parte 2

### LangChain

**Questão 1:** Desenvolva um protótipo utilizando LangChain que simule um chatbot simples com Fake LLM.

#### A aplicação deve:

- Receber um input de texto via FastAPI.
- Retornar uma resposta simulada pelo Fake LLM.

#### O que é esperado:

- O protótipo deve simular um chatbot básico que responde a perguntas pré-definidas.
- A arquitetura deve ser simples, e você deve explicar a importância de usar Fake LLM para testes rápidos.
- Desenhe um diagrama simples da arquitetura do aplicativo, detalhando as principais etapas do fluxo de dados.

## Parte 2.1 -

**Questão 1:** Desenvolva um protótipo utilizando LangChain que simule um chatbot simples com Fake LLM.

**A aplicação deve:**

- Receber um input de texto via FastAPI.
- Retornar uma resposta simulada pelo Fake LLM.

**O que é esperado:**

- O protótipo deve simular um chatbot básico que responde a perguntas pré-definidas.
- A arquitetura deve ser simples, e você deve explicar a importância de usar Fake LLM para testes rápidos.
- Desenhe um diagrama simples da arquitetura do aplicativo, detalhando as principais etapas do fluxo de dados.

```

Extension: vscode-pdf  app.py  app2.py  ×  main.py  requirements.txt
python_files > app2.py > ChatRequest
/
responses = [
8     "Olá! Como posso ajudá-lo hoje?",
9     "Eu sou um chatbot Fake LLM.",
10    "Eu respondo perguntas para simular um chatbot básico.",
11    "Até logo! Foi bom falar com você.",
12    "Desculpe, não entendi sua pergunta.",
13 ]
14
15 # Fake LLM com respostas pré-definidas
16 fake_llm = FakeListLLM(responses=responses)
17
18 # Inicializando FastAPI
19 app = FastAPI()
20
21 # Modelo de Entrada
22 class ChatRequest(BaseModel):
23     question: str
24
25 @app.post("/chat")
26 async def chat(request: ChatRequest):
27     try:
28         # Gera uma resposta usando o FakeLLM
29         response = fake_llm(request.question)
30         return {"response": response}
31     except Exception as e:
32         raise HTTPException(status_code=500, detail=f"Erro interno: {str(e)}")
33

```



FastAPI

|  
|  
v

LangChain

(Fake LLM)

|  
|  
v

Resposta

Simulada

**Questão 2:** Desenvolva um aplicativo que utilize LangChain para integrar a API da OpenAI.

**O aplicativo deve:**

- Receber um texto em inglês via FastAPI.
- Traduzir o texto para o francês utilizando um modelo da OpenAI via LangChain.
- Retornar o texto traduzido em uma resposta JSON.

**O que é esperado:**

- O aplicativo deve funcionar como uma API de tradução, semelhante à questão 2 (Parte 1), mas utilizando a OpenAI via LangChain.
- A aplicação deve gerenciar as chamadas à API da OpenAI e retornar a tradução com baixa latência.
- Forneça um diagrama da arquitetura da aplicação, destacando os componentes principais, como FastAPI, LangChain, e OpenAI.

+-----+ +-----+ +-----+

| Usuário ----> FastAPI ----> LangChain |

|

v

OpenAI API

(Tradução Inglês

Francês)

|

v

Retorna Tradução

ao Usuário

```

from fastapi import FastAPI, HTTPException
from pydantic import BaseModel
from transformers import GPT2LMHeadModel, GPT2Tokenizer

# Inicializar FastAPI
app = FastAPI()

# Baixar e carregar o modelo GPT-2
model_name = "gpt2"
tokenizer = GPT2Tokenizer.from_pretrained(model_name)
model = GPT2LMHeadModel.from_pretrained(model_name)

# Modelo de entrada da API
Codeium: Refactor | Explain
class TextRequest(BaseModel):
    text: str

Codeium: Refactor | Explain | Generate Docstring | X
@app.post("/generate")
async def generate_text(request: TextRequest):
    try:
        # Tokenizar o texto de entrada
        inputs = tokenizer.encode(request.text, return_tensors="pt", truncation=True, max_length=50)

        # Gerar texto com o GPT-2
        outputs = model.generate(inputs, max_length=100, num_return_sequences=1, do_sample=True)

        # Decodificar o texto gerado
        generated_text = tokenizer.decode(outputs[0], skip_special_tokens=True)

```

**Questão 3:** Crie uma API semelhante à Questão 2 (Parte 2), mas que utilize o modelo Helsinki-NLP/opus-mt-en-de da HuggingFace para traduzir textos do inglês para o alemão.

A aplicação deve:

- Receber um texto em inglês via FastAPI.
- Utilizar o LangChain para gerenciar as chamadas ao modelo HuggingFace.
- Retornar o texto traduzido para o alemão como resposta JSON.

**O que é esperado:**

- O objetivo é que a aplicação funcione de maneira semelhante às Questões 2 (Parte 1) e 2 (Parte 2), mas desta vez integrando LangChain com HuggingFace.
- O modelo a ser utilizado deve ser o Helsinki-NLP/opus-mt-en-de.
- Forneça um diagrama detalhado da arquitetura da aplicação, destacando as interações entre FastAPI, LangChain, e HuggingFace.

```

# Configurando o modelo HuggingFace com pipeline
translation_pipeline = pipeline("translation_en_to_de", model="Helsinki-NLP/opus-mt-en-de")

# Configurando o LLM do LangChain
llm = HuggingFacePipeline(pipeline=translation_pipeline)

# Template de prompt para LangChain
prompt = PromptTemplate(
    input_variables=["text"],
    template="{text}" # Para tradução, o texto é usado diretamente.
)

# Criando a cadeia LangChain
translation_chain = LLMChain(prompt=prompt, llm=llm)

# Modelo de entrada da API
Codeium: Refactor | Explain
class TranslationRequest(BaseModel):
    text: str

Codeium: Refactor | Explain | Generate Docstring | X
@app.post("/translate")
async def translate(request: TranslationRequest):
    try:
        # Usando a cadeia LangChain para traduzir
        translated_text = translation_chain.run(text=request.text)
        return {"translated_text": translated_text.strip()}
    except Exception as e:
        raise HTTPException(status_code=500, detail=f"Erro interno: {str(e)}")

```

**Questão 5:** Com base na aplicação desenvolvida na 3 (Parte 2), explique as limitações de usar LangChain para integrar o modelo HuggingFace de tradução.

Discuta aspectos como:

- Desempenho e tempo de resposta.
- Consumo de recursos computacionais.
- Possíveis limitações no ajuste fino do modelo.
- Comparação com o uso direto da API HuggingFace.

## Desempenho e tempo de resposta

**Tempo de resposta:** O uso de LangChain pode aumentar o tempo de resposta da aplicação, pois é necessário realizar uma chamada adicional para o servidor LangChain.

**Desempenho:** O desempenho da aplicação pode ser afetado pelo uso de LangChain, pois é necessário processar a solicitação de tradução e retornar a resposta.

## **Consumo de recursos computacionais**

**Consumo de CPU:** O uso de LangChain pode aumentar o consumo de CPU da aplicação, pois é necessário realizar cálculos adicionais para processar a solicitação de tradução.

**Consumo de memória:** O uso de LangChain pode aumentar o consumo de memória da aplicação, pois é necessário armazenar a solicitação de tradução e a resposta.

## **Possíveis limitações no ajuste fino do modelo**

**Limitações no ajuste fino:** O uso de LangChain pode limitar a capacidade de ajuste fino do modelo de tradução, pois é necessário realizar ajustes no modelo LangChain em vez de no modelo HuggingFace.

**Dificuldade em personalizar o modelo:** O uso de LangChain pode dificultar a personalização do modelo de tradução, pois é necessário realizar ajustes no modelo LangChain em vez de no modelo HuggingFace.

## **Comparação com o uso direto da API HuggingFace**

**Desempenho:** O uso direto da API HuggingFace pode ser mais rápido e eficiente do que o uso de LangChain, pois não é necessário realizar uma chamada adicional para o servidor LangChain.

**Consumo de recursos computacionais:** O uso direto da API HuggingFace pode ser mais eficiente em termos de consumo de recursos computacionais, pois não é necessário realizar cálculos adicionais para processar a solicitação de tradução.

**Questão 6:** Com base nas questões 1-2 (Parte 1) e 2-3 (Parte 2), desenvolva uma tabela comparativa que aborde os seguintes critérios:

- Facilidade de uso/configuração.
- Latência e desempenho.
- Flexibilidade para diferentes modelos.
- Custo e escalabilidade.
- Adequação para protótipos versus aplicações em produção.
- A comparação deve ser apresentada em formato de tabela, com colunas dedicadas a cada critério e linhas comparando FastAPI puro com LangChain.

Critério	FastAPI Puro	LangChain
<b>**Facilidade de uso/configuração**</b>	<b>Alta</b>	<b>Média</b>
<b>**Latência e desempenho**</b>	<b>Alta</b>	<b>Baixa</b>
<b>**Flexibilidade para diferentes modelos**</b>	<b>Baixa</b>	<b>Alta</b>
<b>**Custo e escalabilidade**</b>	<b>Baixo</b>	<b>Alto</b>
<b>**Adequação para protótipos**</b>	<b>Alta</b>	<b>Baixa</b>
<b>**Adequação para aplicações em produção**</b>	<b>Baixa</b>	<b>Alta</b>