

INFNET

Desenvolvimento de Data-Driven Apps com Python [24E4_3] - TP2

Aluno: Rodrigo Moreira Avila

Repositório GIT: <https://github.com/r-moreira/infnet-data-driven-apps-tp2-p1>

Questões teóricas

Parte 1

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.

A API desenvolvida na Questão 2 (Parte1) possui limitações.

1. **Limitações quanto à precisão da tradução:** O modelo pode cometer erros assim como qualquer outro modelo de LLM, no caso do modelo em questão, ele pode ter dificuldade ao traduzir frases complexas ou ambíguas.
2. **Desafios de tempo de resposta e desempenho em grande escala:** O modelo pode ter um tempo de resposta maior em grande escala, pois o uso de recursos usado pelo modelo é grande, ao se comparar com uma consulta em banco de dados, por exemplo.
3. **Restrições de custo e escalabilidade:** O modelo pode ter um custo alto para ser utilizado em grande escala, devido a quantidade de recursos usados, e consequentemente, afetando a escalabilidade do modelo.
4. **Limitações na tradução de gírias, expressões idiomáticas ou linguagem de contexto:** O modelo pode ter dificuldade em traduzir gírias, expressões idiomáticas ou linguagem de contexto, pois o modelo nem sempre consegue entender o contexto da frase, e assim, pode traduzir de forma errada.

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: O modelo GPT-2 se comparado a outros modelos, ele tem menos precisão, e nem sempre consegue gerar um texto coerente, ele acaba alucinando bastante, principalmente em textos mais longos.

Possíveis falhas ou incoerências geradas por LLMs: Falhas ou incoerências são comuns, pois ele não tem acesso a uma base de dados factual, e assim, ele pode gerar informações erradas, tudo depende do treinamento do modelo e da quantidade de dados que ele foi treinado.

Desempenho e questões de latência: O modelo GPT-2 pode ter um desempenho ruim em grande escala, pois o uso de recursos usado pelo modelo é grande, ao se comparar com uma consulta em banco de dados, por exemplo, inclusive ele é muito mais pesado que o modelo de tradução da Questão 2 (Parte1).

Limitações na geração de conteúdo apropriado: O modelo GPT-2 pode acabar gerando conteúdo inadequado ou ofensivo, já que ele não tem uma compreensão real do que está sendo gerado, ele apenas tenta gerar um texto coerente baseado no treinamento que ele recebeu. Além disso, ele pode apresentar vieses e falta de contexto atualizado, o GPT2 é um modelo de 2019.

```
In [1]: !jupyter nbconvert --to webpdf rodrigo_avila_DR3_TP2_p1.ipynb
```

```
[NbConvertApp] Converting notebook rodrigo_avila_DR3_TP2_p1.ipynb to webpdf
[NbConvertApp] Building PDF
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 171716 bytes to rodrigo_avila_DR3_TP2_p1.pdf
```

INFNET

Desenvolvimento de Data-Driven Apps com Python [24E4_3] - TP2

Aluno: Rodrigo Moreira Avila

Repositório GIT: <https://github.com/r-moreira/infnet-data-driven-apps-tp2-p2>

Questões teóricas

Parte 2

Questão 4: Com base na implementação da Questão 2 (Parte 2), explique as principais limitações de utilizar LangChain para integrar a API da OpenAI.

Discuta os seguintes aspectos:

- Latência de resposta.
- Limites de uso da API da OpenAI.
- Desafios de escalabilidade e custo.
- Qualidade das traduções geradas em comparação com outros modelos.

1. **Latência de resposta:** O modelo da Open AI possui uma latência baixa, pois roda na infraestrutura da Open AI que já é otimizada para processamento de modelos de linguagem.
2. **Limites de uso da API da OpenAI:** A OpenAI possui limites de uso da API, que podem ser consultados em <https://beta.openai.com/docs/api-reference/usage-limits>. A API da OpenAI é cobrada por uso, e o custo pode ser um limitante para a utilização em larga escala.
3. **Desafios de escalabilidade e custo:** Por ser uma API paga, a OpenAI possui uma infraestrutura robusta e escalável, o que facilita a utilização em larga escala, porém o custo pode ser um limitante para a utilização.
4. **Qualidade das traduções geradas em comparação com outros modelos:** A qualidade das traduções geradas pelo modelo da OpenAI é muito boa, me parece ser um dos melhores disponíveis no mercado.

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.

A performance pode variar dependendo do tamanho do modelo e da infraestrutura. Modelos maiores podem exigir mais recursos computacionais. O ajuste fino do modelo pode ser limitado, pois o modelo é pré-treinado e não é possível ajustar os pesos do modelo.

O uso direto da API HuggingFace pode ser mais eficiente, pois a API é otimizada para o modelo e possui uma infraestrutura robusta e escalável.

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	HuggingFace e Open AI (diretamente)	LangChain
Facilidade de uso/configuração	Requer configuração manual de modelos e pipelines. Pode ser mais complexo para iniciantes.	Abstrai grande parte da configuração, facilitando o uso inicial.
Latência e desempenho	Depende da implementação e otimização manual. Pode ser mais rápido se bem otimizado.	Pode introduzir alguma latência adicional devido à abstração, mas ainda eficiente.
Flexibilidade para diferentes modelos	Alta flexibilidade, permite uso de qualquer modelo disponível no HuggingFace.	Flexível, mas pode ser limitado às integrações suportadas pela biblioteca.
Custo e escalabilidade	Custo depende do uso de infraestrutura própria ou	Pode ser mais caro devido à abstração e serviços

Critério	HuggingFace e Open AI (diretamente)	LangChain
	serviços de terceiros. Escalabilidade manual.	adicionais. Escalabilidade facilitada.
Adequação para protótipos versus aplicações em produção	Bom para protótipos e produção, mas requer mais esforço para escalar e manter.	Excelente para protótipos rápidos e aplicações em produção com menos esforço de manutenção.

```
In [ ]: !jupyter nbconvert --to webpdf rodrigo_avila_DR3_TP2_p2.ipynb
```