*Curso*: **IA para Devs**

*Turma*: **Pós Tech - 2IADT**

*Aluno*: **Ricardo de Moura Meneses**

**Tech Challenge - Fase 03**

**Preparando o Dataset**

Seguindo as orientações contidas na documentação do Tech Challenger (disponível em [FIAP](https://on.fiap.com.br/mod/assign/view.php?id=436348&c=12019)), realizei o download da base de referência para o exercício.

De acordo com o professor Carlos, no vídeo de apresentação do Tech Challenge realizado no dia 08/10/2024, a base de dados era muito grande. Considerando essa premissa, decidi reduzir a base de treinamento da LLM para 200.000 registros, seguindo recomendações relacionadas ao tempo necessário para treinamento, dado o poder de processamento limitado do meu computador.

Desenvolvi uma rotina para ler a base original e criar uma versão reduzida. Durante o desenvolvimento, identifiquei a necessidade de eliminar registros com conteúdo em branco. Adicionei, ainda, limites mínimos (10 palavras) e máximos (200 palavras) para os textos, baseando-me em um artigo que mencionava melhorias no tempo de treinamento e inferência com essa abordagem.

Com base nesse tratamento, foram desenvolvidos os seguintes arquivos para o exercício:

* 1-Filtrar e preparar os dados.ipynb
* 2-Foundation Model.ipynb
* 3-Fine-tuning.ipynb
* 4-TesteNoResultadoDoFine-Tuning.ipynb
* 5-TesteNoGPT2.ipynb
* 99-PerplexidadeDoModelo.ipynb

O conteúdo de cada arquivo será abordado no momento pertinente ao longo do documento. O conteúdo do arquivo 1-Filtrar dados já foi tratado neste item.

**Pesquisa direta no LLM escolhido para o treinamento**

O arquivo 2-Foundation Model.ipynb foi desenvolvido para que buscas fossem realizadas diretamente no modelo GPT-2 pré-treinado. As perguntas realizadas ao modelo retornaram respostas aleatórias e desconexas, sem relação com o contexto usado nos questionamentos.

Esse resultado não foi surpreendente, pois o GPT-2 pré-treinado não possui conhecimento específico sobre os dados utilizados no exercício. O modelo escolhido foi o GPT-2 por ser uma forma prática de ambientação com a estrutura que pretendo utilizar profissionalmente ao trabalhar com APIs do GPT nas próximas semanas.

**O Fine-tuning**

O código para o Fine-tuning foi desenvolvido e testado gradualmente. Durante o processo, encontrei e corrigi diversos erros, como problemas de sintaxe, uso incorreto de funções e parâmetros, entre outros. Embora esses erros tenham ocorrido em todos os arquivos gerados, no Fine-tuning a quantidade foi significativamente maior. Não nego que foi uma experiência divertida e desafiadora.

A maior surpresa ocorreu quando executei o código corretamente pela primeira vez no Jupyter Notebook no meu computador: o tempo estimado para finalizar o processo de treinamento era de 96 horas. Inicialmente, não percebi essa estimativa; foi somente após alguns segundos analisando as mensagens na tela que identifiquei a previsão.

Diante desse cenário, tomei as seguintes decisões, todas feitas rapidamente e com certo pânico:

* Redução da base de treinamento para 100.000 registros → tempo estimado: 7h30.
* Redução da base de treinamento para 50.000 registros → tempo estimado: 45 minutos.

Outros parâmetros da função TrainingArguments foram alterados, mas o único que apresentou impacto prático foi num\_train\_epochs, que reduzi de 3 para 1 ao longo das tentativas.

Considerei o tempo de 45 minutos aceitável e deixei o processamento seguir. Infelizmente, faltando menos de um minuto para o término, meu computador entrou em modo de espera, suspendendo o treinamento. Durante a execução, o consumo de CPU permaneceu em 100%, o que me fez evitar interações para não interferir no processamento.

Foi neste momento que me lembrei do Google Colab. Ao migrar o código e os dados para o Colab, obtive os seguintes resultados:

* 50.000 registros → tempo de processamento: 12 minutos.
* 100.000 registros → tempo de processamento: 56 minutos.
* 200.000 registros → tempo de processamento: 1h45 estimado, tempo real 49min.

Após o primeiro teste, aumentei o parâmetro num\_train\_epochs para 2 e dobrei o per\_device\_train\_batch\_size, que havia começado em 4. Para os 200.000 registros, aumentei o per\_device\_train\_batch\_size para 16. Embora tenha considerado aumentar o gradient\_accumulation\_steps, optei por não fazê-lo, temendo aumentar excessivamente o tamanho efetivo do lote. Também cogitei diminuir o learning\_rate, mas descartei a ideia para evitar prolongar ainda mais o tempo de processamento.

**Testes no resultado obtido com o treinamento**

Após o treinamento, os arquivos foram armazenados na pasta “./gpt2-finetuned”. Realizei o download dos arquivos e refiz os mesmos testes executados na fase do Foundation Model.

Os resultados foram insatisfatórios: o modelo apresentou as mesmas respostas para todas as perguntas. Esse comportamento parece estar relacionado à falta de contexto causada pela base reduzida.

Esse teste foi importante para validar a qualidade dos dados e o comportamento do modelo. Pretendo realizar novas análises e refinamentos futuros para explorar possíveis melhorias e usos.

**Testes no GPT-2 após o treinamento**

Os testes realizados no modelo GPT-2 após o treinamento apresentaram respostas satisfatórias, considerando que foi um primeiro ajuste sem um fine-tuning avançado.

Novos testes serão conduzidos em treinamentos futuros, com ajustes mais precisos nos parâmetros.

Perplexidade do Modelo

Para verificar a qualidade do treinamento, calculei a perplexidade do modelo. O valor obtido foi aproximadamente 20,09, o que indica um ajuste de modelo razoável.

**Conclusão**

Como uma primeira tentativa de uso de LLMs com um dataset desconhecido, considero que o aprendizado obtido foi significativo. A conclusão do exercício, mesmo com suas limitações, proporcionou um aprendizado prático valioso.

Reconheço que os resultados não são ideais para uso em um contexto profissional, pois o trabalho ainda demanda melhorias significativas. Entretanto, do ponto de vista acadêmico e prático, considero que o resultado foi excelente e cumpriu seus objetivos.