UTILIZAÇÃO DE FINE-TUNING PARA MELHORAR O DESEMPENHO DE CUSTOMIZAÇÃO DE MODELOS DE IA(LLM) UTILIZANDO QLoRA

CASARA, Lucas Patrick [1]
HEIDEMANN, André Nicolas [2]
ALVES, Luiz Carlos Machado [3]
WOLFF, Victor Alexandre [4]
PETRI, Marcelo [5]

RESUMO

Este artigo apresenta uma avaliação do desempenho de adaptadores QLoRA para modelos de linguagem grandes (LLMs) quantizados. Os adaptadores QLoRA são uma técnica recente que permite melhorar o desempenho de LLMs quantizados sem afetar significativamente sua eficiência. O artigo foi realizado em um conjunto de dados de texto e código e os resultados mostraram que os adaptadores QLoRA podem melhorar significativamente o desempenho de LLMs quantizados em uma variedade de tarefas, incluindo geração de texto, tradução e resposta a perguntas.

Palavras-chave: Adaptadores; QLoRA; LLMs quantizados; Geração de texto; Experts; MoE; Dataset; StableBeluga; GPTQ.

[1]Graduanda(o) do Curso de Engenharia da Computação do Centro Universitário UNISOCIESC, lucas.ghostpack@gmail.com; [2]Graduanda(o) do Curso de Engenharia da Computação do Centro Universitário UNISOCIESC, dev.andreheidemann@gmail.com; [3]Graduanda(o) do Curso de Engenharia da Computação do Centro Universitário UNISOCIESC, luiz@assessoritec.com.br; [3]Graduanda(o) do Curso de Engenharia da Computação do Centro Universitário UNISOCIESC, victorwolff@alerok.net; [5]Marcelo Petri Orientador, Centro Universitário UNISOCIESC, marcelo.petri@unisociesc.com.br;

Joinville – SC, 28 de Novembro de 2023.

ABSTRACT

This article presents an evaluation of the performance of QLoRA adapters for quantized large language models (LLMs). QLoRA adapters are a recent technique that allows to improve the performance of quantized LLMs without significantly affecting their efficiency. The paper was carried out on a text and code dataset and the results showed that QLoRA adapters can significantly improve the performance of quantized LLMs in a variety of tasks, including text generation, translation and question answering.

Keywords: Adapters; QLoRA; Quantized LLMs; Text generation; Experts; MoE; Dataset; StableBeluga; GPTQ.

1 INTRODUÇÃO

As evoluções tecnológicas na área de *deep learning*[1] estão cada vez mais presentes em nosso cotidiano, tanto na geração de imagens a partir de uma descrição textual, quanto em serviços de atendimento ao cliente. Com essa evolução, a demanda por modelos de linguagem altamente customizados e de baixo custo aumenta exponencialmente. Afinal, cada vez mais empresas desejam usufruir dos benefícios que essa tecnologia oferece. Porém, o principal problema enfrentado por aqueles que desejam utilizar dessas tecnologias é o custo e complexidade da operação, que demanda muito poder computacional e geralmente tem um custo de manutenção muito elevado. A solução encontrada por muitos é terceirizar essas operações, fazendo com que todo o gerenciamento fique a cargo de uma entidade especializada. Nesses casos, o *fine-tuning*[2] é uma ótima alternativa para a empresa prestadora de serviços tanto para remover comportamentos indesejados, quanto para adicionar novos. Porém, fazer fine-tuning de grandes modelos de linguagem natural também demanda muito poder computacional atrelado a altos custos operacionais. (HUANG, C.; ZHANG, Z.; MAO, B.; et al, 2023, p. 799-819)

Empregar novas ferramentas de otimização torna viável reduzir drasticamente a quantidade de recursos computacionais necessários para a realização do *fine-tuning*, sem que haja alteração significativa da performance.

No artigo publicado no QLORA, foi demonstrado pelos pesquisadores, pela primeira vez, a possibilidade de realizar o *fine-tuning* de um modelo quantizado de 4

[1] Aprendizado profundo
[2] Ajuste fino

bits sem comprometer o desempenho, mesmo com o uso significativo de recursos. O objetivo do projeto é obter um ganho de desempenho considerável através de métodos de *fine-tuning*, ao mesmo tempo em que se reduz o custo. (Dettmers, Tim; Pagnoni, Artidoro; Holtzman, Ari et al., 2023, p. 1).

2 REFERENCIAL TEÓRICO

A Inteligência Artificial (IA) teve seu início após o término da Segunda Guerra Mundial, com o termo sendo definido apenas em 1956 por John McCarthy, que descreveu a IA como a ciência e engenharia de criar máquinas inteligentes.

Atualmente, é comum categorizar os tipos de IA em algumas categorias sendo as principais desse artigo: IAE[3] e IAG[4].

A IAE refere-se a sistemas de inteligência artificial “projetados para desempenhar tarefas específicas e limitadas” (Kurzweil, 2006, p. 560). Esses sistemas são capazes de realizar suas tarefas de forma eficiente e consistente, mas não possuem a capacidade de compreender ou aprender além do escopo da tarefa designada.

A IAG é uma forma de inteligência artificial que seria capaz de realizar qualquer tarefa cognitiva que um ser humano pudesse realizar, e poderia realizar essas tarefas com a mesma facilidade e flexibilidade de um ser humano” (Goertzel, 2014, p. 3).

Os modelos de ML[5] geralmente são compostos por um conjunto de regras, procedimentos ou sofisticadas “funções de transferência” que podem ser usadas para descobrir padrões de dados interessantes ou antecipar comportamentos.(DUA, S.; DU, X.,2016, p. 22-66)

Com esse contexto histórico, chega-se ao presente, onde se trabalha para criar modelos públicos e alcançar IAEs personalizadas para melhor atender a mais cenários e talvez no futuro chegar a IAG. Neste artigo, o foco é demonstrar como isso pode ser alcançado sem um custo operacional elevado e trazer novas funcionalidades para os modelos.

---

[3] Inteligência Artificial Estreita
[4] Inteligência Artificial Geral
[5] Machine Learning

2.1 LLaMa2 - META

Large Language Model Meta AI 2 (LLaMa2) é uma família de modelos de linguagem de última geração de acesso aberto lançados pela Meta. Esses modelos foram treinados em 40% mais *tokens*[6] (unidades básicas de texto) e têm um comprimento de contexto muito mais longo (4k *tokens*) em comparação com os modelos LLaMa1. Além disso, eles usam a atenção de consulta agrupada para uma inferência rápida do modelo de 70B. Os modelos LLaMa2-Chat foram otimizados para aplicações de diálogo usando *Reinforcement Learning from Human Feedback* (RLHF) e apresentam um desempenho melhor do que a maioria dos modelos abertos em uma ampla gama de *benchmarks* de utilidade e segurança. (Touvron, Hugo; Martin, Louis; Stone, Kevin et al. 2023)

2.2 STANDARD FINE-TUNING

O *fine-tuning* padrão, também conhecido como "*Standard Fine-tuning*", é uma técnica comumente usada no treinamento de modelos de aprendizado profundo. Essa técnica envolve a utilização de um modelo pré-treinado, que já aprendeu representações úteis de um grande conjunto de dados, e o ajuste desse modelo em um conjunto de dados menor e mais específico para uma tarefa particular.

Para usar os modelos de maneira eficaz, incluímos instruções e, às vezes, vários exemplos em um *prompt*. Usar demonstrações para mostrar como realizar uma tarefa é frequentemente chamado de "aprendizado de poucos exemplos" ou "*few-shot learning*".

Uma vez que um modelo tenha sido ajustado, não será necessário fornecer tantos exemplos no *prompt* (instruções de contexto). Isso economiza custos e permite solicitações de menor latência. Em um alto nível, o *fine-tuning* envolve as seguintes etapas: Preparar e carregar os dados de treinamento. Treinar um novo modelo ajustado. Avaliar os resultados e voltar ao passo 1, se necessário. Usar o modelo ajustado.(Dettmers, Tim; Pagnoni, Artidoro; Holtzman, Ari et al., 2023, p 4-7.)

[6] Agrupamento de caracteres representando a unidade fundamental de um texto

2.3 LoRA

O *Low Rank Adapters* (LoRA) Adaptadores de baixa classificação é uma abordagem que permite que um modelo pré-treinado seja compartilhado entre várias tarefas, reduzindo a necessidade de treinar modelos separados para cada tarefa.

Normalmente, a maioria dos modelos de fine tuning possibilita o aprendizado de uma parte dos parâmetros que já foram treinados, enquanto isso: "LoRA dá um passo adiante e não requer que a atualização do gradiente acumulado nas matrizes de peso tenha posto completo durante a adaptação. Isso significa que, ao aplicar LoRA em todas as matrizes de peso e treinar todos os Padrões de Regras (em inglês: "Bias", também pode ser entendido como limitadores)."(HU, Edward J.; SHEN, Yelong; WALLIS, Phillip; et al., 2021, p 4.)

O "*fine-tuning*" do LoRA comparado a um modelo ajustado por construção não apresenta latência adicional durante a inferência como demonstrado abaixo "Garante que não introduzimos qualquer latência adicional durante a inferência em comparação com um modelo ajustado por construção."(HU, Edward J.; SHEN, Yelong; WALLIS, Phillip; et al., 2021, p 4.)

"Os Benefícios Práticos mais significativos advém da redução no uso de memória e armazenamento. Para um grande Transformer treinado com "*AdamW*"[7], reduzimos o uso de VRAM em até 2/3 se o valor de r for muito menor que o valor de "dmodel" (dimensões do modelo), então o uso de VRAM[8] (memória utilizada pela GPU[9])", e LoRA possui algumas limitações, Mesmo que seja possível realizar o *merge*[10] entre modelos LoRA para eliminar a latência adicional de inferência não é possível mesclar os pesos ("*weights*", que nesse contexto é a o que o modelo aprendeu no seu treinamento) e escolher dinamicamente os módulos LoRA para as amostras em lote em que a latência não é crítica. (HU, Edward J.; SHEN, Yelong; WALLIS, Phillip; et al., 2021, p 5.)

LoRA possibilita a criação de diversos modelos personalizados que podem ser facilmente alternados em máquinas que armazenam os pesos pré-treinados na VRAM. "Também observamos um aumento de 25% na velocidade durante o treinamento no GPT-3 175B em comparação com o *fine-tuning* completo, pois não

---

[7] Optimizador paginado com o argumento *AdamW*
[8] *Video Random Access Memory*
[9] *Graphics Processing Unit*
[10] Combinar informações, dados ou estruturas.

precisamos calcular o gradiente para a grande maioria dos parâmetros.” (HU, Edward J.; SHEN, Yelong; WALLIS, Phillip; et al., 2021, p 5.)

O “*fine-tuning*” de modelos de linguagem enormes exige *hardware* com alta capacidade de processamento e sendo assim elevando o custo para treiná-lo.
“Propomos o LoRA, uma estratégia eficiente de adaptação que não introduz latência de inferência nem reduz o comprimento da sequência de entrada, ao mesmo tempo que mantém alta qualidade do modelo.” (HU, Edward J.; SHEN, Yelong; WALLIS, Phillip; et al., 2021, p 12.)

Os trabalhos futuros podem ser combinados com outros métodos eficientes de adaptação, apresentando um potencial de melhoria ortogonal, prevendo a execução por meio de maneiras mais eficientes de operar o modelo.

2.4 QLoRA

O “*Quantized Low Rank Adapters*” (QLoRA) é uma abordagem eficiente para o *fine-tuning* de LLMs[11] que reduz significativamente o uso de memória enquanto mantém o desempenho do *fine-tuning* completo de 16 *bits*. Ele consegue isso ao propagar gradientes através de um modelo de linguagem pré-treinado congelado e quantizado em 4 *bits* para Adaptadores de Baixa Classificação (LoRA). (DETTMERS, Tim; PAGNONI, Artidoro; HOLTZMAN, Ari; et al. 2023)

O QLoRA utiliza duas técnicas principais para alcançar a quantização de *fine-tuning* de alta fidelidade de 4 *bits* - Quantização “*NormalFloat*” de 4 *bits* (NF4) e Quantização Dupla. Além disso, ele introduz Otimizadores Paginados para prevenir picos de memória durante o “*checkpointing*” de gradientes que causam erros de falta de memória que tradicionalmente tornam o *fine-tuning* em uma única máquina difícil para modelos grandes. (DETTMERS, Tim; et al. 2023)

No contexto, o termo “*checkpoint*” é frequentemente empregado para se referir a um estado salvo do modelo durante o processo de treinamento. A prática de criar checkpoints apresenta uma série de benefícios. Primeiramente, permite a recuperação de falhas. Caso o treinamento seja interrompido por algum motivo, como uma falha de energia ou um erro de sistema, é possível retomar o treinamento a partir do último checkpoint, evitando a necessidade de começar do zero. Em segundo lugar, os checkpoints podem ser utilizados como ponto de partida para o

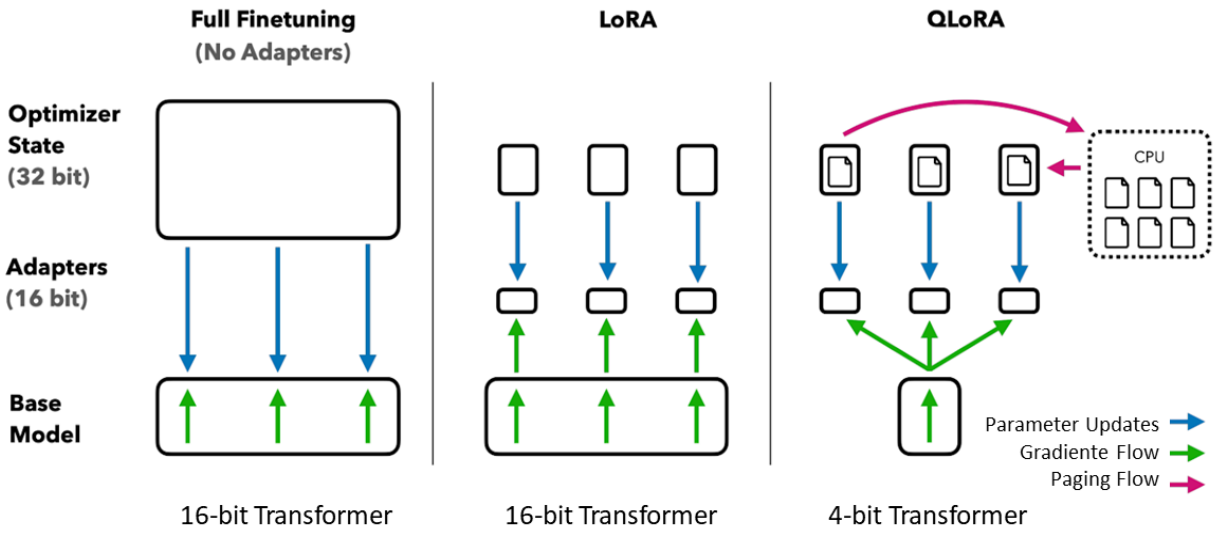
[11] *Large Language Models*

fine-tuning do modelo em um novo conjunto de dados. Por fim, os checkpoints permitem a inspeção do modelo para entender como os parâmetros estão mudando ao longo do tempo. Portanto, a criação de *checkpoints* é uma prática valiosa que contribui para a eficiência e a eficácia do treinamento de modelos de aprendizado de máquina. (DETTMERS, Tim; et al. 2023)

A Quantização "*NormalFloat*" de 4 *bits* (NF4) é uma técnica baseada na Quantização de Quantil, que é um tipo de dados teoricamente ótimo que garante que cada "*bin*" de quantização (conjunto de informações) tenha um número igual de valores atribuídos a partir do *tensor*[12] de entrada. A quantização de quantil funciona estimando o quantil do *tensor* de entrada através da função de distribuição cumulativa empírica. (DETTMERS, Tim; et al. 2023)

Na Figura 1 temos: Métodos diferentes de *fine-tuning* e seus requisitos de memória. QLoRA melhora em relação ao LoRA ao quantizar o modelo do transformador para precisão de 4 *bits* e utilizar otimizadores paginados para lidar com picos de memória.

Figura 1: *Fine-tunning* - QLoRA



Fonte: DETTMERS, Tim; PAGNONI, Artidoro; HOLTZMAN, Ari; et al. (2023).

A Quantização Dupla ou "*Double Quantization*" (DQ) é um processo que quantiza as constantes de quantização para economias adicionais de memória. Embora um pequeno tamanho de bloco seja necessário para uma quantização precisa de 4 bits, ele também tem uma sobrecarga de memória considerável. Por

[12] Um *tensor* é uma matriz n-dimensional que pode ser utilizado tanto na GPU quanto na CPU

exemplo, usando constantes de 32 *bits* e um tamanho de bloco de 64 para W, as constantes de quantização adicionam 32/64 = 0,5 *bits* por parâmetro em média. A Quantização Dupla ajuda a reduzir o uso de memória das constantes de quantização. (DETTMERS, Tim; et al. 2023)

2.5 MODELOS GPTQ

Os Modelos GPTQ são uma série de modelos de linguagem generativa pré-treinados e ajustados finos, variando em escala de 7 bilhões a 70 bilhões de parâmetros. Eles são otimizados para casos de uso de diálogo.

O GPTQ é uma técnica de quantização pós-treinamento que foi projetada para comprimir Modelos de Linguagem, incluindo LLaMa. Ao reduzir o número de *bits* ("*Binary Digit*" em inglês, "dígito binário") necessários para armazenar cada peso no modelo de 32 *bits* para 3-4 *bits*, o GPTQ minimiza significativamente o custo de memória do modelo. Isso permite operações eficientes em recursos de hardware mais baixos, como uma única GPU, para modelos LLaMa2 de 7B ou 13B.

Especificamente, o GPTQ pode quantizar modelos GPT com 175 bilhões de parâmetros em aproximadamente quatro horas de GPU, reduzindo a largura de *bits* para 3 ou 4 *bits* por peso, com degradação de precisão negligenciável em relação à linha de base não compactada. Isso permite que, pela primeira vez, seja possível executar um modelo de 175 bilhões de parâmetros dentro de uma única GPU.

Em termos de desempenho, o GPTQ mais do que duplica os ganhos de compressão em relação aos métodos de quantização *one-shot* propostos anteriormente, preservando a precisão. Além disso, o GPTQ ainda pode fornecer precisão razoável no regime de quantização extrema, no qual os pesos são quantizados para níveis de quantização de 2 *bits* ou até ternários. (Frantar, E.; Ashkboos, S.; Hoefler, T. et al. 2023.)

2.6 DATASET

“Dataset”, ou conjunto de dados, é uma coleção de dados que é tratada como uma única unidade por um computador. Isso significa que um *dataset* contém muitos pedaços separados de dados, mas pode ser usado para treinar um algoritmo com o

objetivo de encontrar padrões previsíveis dentro do conjunto de dados como um todo.

Os dados são um componente essencial de qualquer modelo de IA, e basicamente, uma das principais razões para o aumento da popularidade do aprendizado de máquina observado atualmente. Devido à disponibilidade de dados, algoritmos de ML escaláveis se tornaram viáveis como produtos reais que podem agregar valor a um negócio, em vez de serem um subproduto de seus principais processos.

Para criar um *dataset* eficaz, é importante entender como gerar dados para aprendizado de máquina e quais dados são necessários. A qualidade e o tamanho do *dataset* desempenham um papel crucial na determinação da precisão e desempenho do modelo. Em geral, quanto mais dados o modelo tiver acesso, melhor ele irá se apresentar. (DETTMERS, Tim; PAGNONI, Artidoro; HOLTZMAN, Ari; et al. 2023)

Um exemplo de *dataset* é o ORCA, composto por uma extensa coleção de dados FLAN[13] aumentados, sendo um tesouro para todos envolvidos no Processamento de Linguagem Natural (NLP). Este *dataset* é uma parte crucial na criação de modelos de IA, pois fornece uma grande quantidade de dados que podem ser usados para treinar e aprimorar os modelos de IA.(Orca ,2023)

2.6.1 COLAPSO DO MODELO

"O Colapso do Modelo, ou "*Model Collapse*", é um processo degenerativo que afeta gerações de modelos gerativos aprendidos. Nesse processo, os dados gerados acabam poluindo o conjunto de treinamento da próxima geração de modelos. Ao serem treinados em dados poluídos, esses modelos acabam percebendo erroneamente a realidade. Existem dois casos especiais: o colapso do modelo precoce e o colapso do modelo tardio. No colapso do modelo precoce, o modelo começa a perder informações sobre as extremidades da distribuição; no colapso do modelo tardio, o modelo confunde diferentes modos das distribuições originais e converge para uma distribuição que tem pouca semelhança com a original, muitas vezes com variância muito pequena. (Shumailov, Shumaylov at al., 2023, p 2-4).

[13] *Fine-tuned LAnguage Net*

É importante notar que esse processo é diferente do processo de esquecimento catastrófico, pois estamos considerando vários modelos ao longo do tempo. Nossos modelos não esquecem os dados aprendidos anteriormente, mas começam a interpretar erroneamente o que acreditam ser real, reforçando suas próprias crenças. (Shumailov, Shumaylov at al., 2023, p 2-4).

Esse processo ocorre devido a duas fontes específicas de erro que se acumulam ao longo das gerações e causam desvio do modelo original. Dessas, uma fonte de erro desempenha um papel primário, e na ausência dela, o processo não ocorreria além da primeira geração.” (Shumailov, Shumaylov at al., 2023, p 2-4).

2.7 STABLE BELUGA

O Stable Beluga é um modelo de linguagem autorregressivo que foi feito um *fine-tuning* no modelo LLaMa2. Ele é usado para gerar respostas de texto a partir de *prompts*. Este modelo foi desenvolvido pela Stability AI e é otimizado para uso com a biblioteca Transformers da HuggingFace.

O Stable Beluga foi treinado em um conjunto de dados interno no estilo ORCA. O treinamento foi realizado através de *fine-tuning* supervisionado em *datasets* mencionados, treinado em precisão mista (BF16) e otimizado com “*paged_AdamW_32bit”*.

Em termos de desempenho, o Stable Beluga demonstrou habilidades excepcionais de raciocínio aparente em vários *benchmarks*. Apesar de ser treinado em uma fração dos dados usados pelo artigo original do ORCA, os modelos do mesmo foram capazes de exibir um desempenho notável em diversos *benchmarks*. Isso valida a abordagem da Stability AI para *datasets* gerados sinteticamente. (stability.ai, 2023)

Para esse artigo, foi utilizada a versão de 7 Bilhões de parâmetros devido à necessidade do menor consumo de memória.

2.8 Estudo SkunkworksAI e MoE

De acordo com um estudo recente de SkunkworksAI, os dados preliminares sugerem que “Modelos MoE[14] superam modelos densos em termos de FLOPs[15] de inferência e também apresentam desempenho superior em relação ao seu nível de complexidade: *Switch-Base* (modelo que utiliza múltiplos Adaptors QLoRA) alcança menor perplexidade no C4 do que o T5-Large (estilos de máquinas que a Amazon oferece para alugar) usando 30% dos FLOPs e 10 vezes os parâmetros, enquanto Switch-Large é competitivo com o T5-XXL com 6% dos FLOPs e 2,5 vezes os parâmetros.”. (dados em desenvolvimento).

O MoE é uma das possibilidades com o QLoRA devido à sua capacidade de reduzir o custo operacional em todas as etapas, gerando uma redução de FLOPs. Isso possibilita a criação de uma “Arquitetura mista de especialistas ativados para escalar a capacidade do modelo, ao mesmo tempo em que incorre em custos de treinamento substancialmente menores em comparação com as variantes densas. “(DU et al., 2022).

2.9 PRINCIPAIS EMPRESAS COM HARDWARE FOCANDO EM IA

Serão mencionadas algumas empresas importantes para o artigo e para a área de IA: NVIDIA é uma das líderes na produção de *hardware e Software* para IA. (NVIDIA, 2023).

Intel: A Intel está colaborando com a Hugging Face para construir aceleração de *hardware* e *software* de última geração para treinar, ajustar e prever com *transformers*. A plataforma Intel® Geti™ é a nova plataforma de *software* da Intel para a construção de modelos de visão computacional em uma fração do tempo e com menos dados.(INTEL, 2023).

Apple: A Apple está trabalhando em sua própria tecnologia de IA. Anunciou os *chips* M3, M3 Pro e M3 Max, três processadores com tecnologias inovadoras que oferecem um aumento dramático no desempenho e liberam novos recursos para o Mac.(APPLE, 2023).

---

[14] *Mixture of Experts*

[15] *Floating-point operations per second* - Unidade de medida para calcular a capacidade de um computador

Google Colab: ou “*Colaboratory*”, é uma ferramenta que permite escrever e executar Python no navegador, com acesso a GPUs e compartilhamento fácil. Você pode usar o Colab para ciência de dados, aprendizado de máquina e outros projetos interativos, e importar dados da sua conta do Google Drive, GitHub ou outras fontes. O Colab é um produto do Google Research, a área de pesquisas científicas do Google.(GOOGLE, 2023).

3 METODOLOGIA

Neste tópico, será abordada a utilização de modelos de inteligência artificial para a criação de dados. Esses dados serão utilizados para o treinamento e criação de adaptadores QLoRA, com o objetivo de descobrir mais cenários em que é possível utilizar, por exemplo, chatbots ou sistemas empresariais. As fontes consultadas incluem artigos e pesquisas, como o apresentado nos tópicos **2.3 e 2.4 sobre LoRA e QLoRA**, entre outras referências.

3.1 CARACTERIZAÇÃO DA PESQUISA

1. **Objetivo da pesquisa**: A pesquisa tem como objetivo avaliar a eficácia da customização de modelos utilizando QLoRA.
2. **Tipo de pesquisa**: Trata-se de uma pesquisa aplicada e de desenvolvimento, pois o estudo visa resolver um problema específico, que é a customização de modelos utilizando QLoRA, através do desenvolvimento e aplicação de adaptadores QLoRA.
3. **Métodos de pesquisa**: Os métodos de pesquisa envolvem o desenvolvimento e treinamento de modelos de IA utilizando conjuntos de dados gerados, e a aplicação dos modelos para gerar novas informações sobre a QLoRA e sua possível utilização.
4. **Fontes de pesquisa**: As fontes de pesquisa utilizadas foram artigos sobre o tema e conteúdos em vídeo e fóruns de pesquisadores.

5. **População e abrangência do estudo**: O estudo é aplicável a todos os usuários e pesquisadores da QLoRA interessados em obter novas informações sobre a tecnologia.
6. **Características de interesse e as respectivas variáveis associadas**: Características de interesse são a eficácia dos modelos de IA no desenvolvimento de customização e personificação de chatbots e aplicações. As variáveis associadas são determinadas com base nos objetivos específicos do estudo.

3.2 AMBIENTE DA PESQUISA

O ambiente da pesquisa é predominantemente digital, uma vez que o estudo envolve a análise de modelos de Inteligência Artificial (IA) e dados. A pesquisa é conduzida utilizando a Tecnologia QLoRA, que introduz um novo método de treinamento e uso de modelos em sistemas computacionais.

A pesquisa é realizada em um ambiente de aprendizado de máquina, onde os modelos de IA são treinados e testados para avaliar seu desempenho. O ambiente de pesquisa também engloba a comunidade de usuários e pesquisadores da QLoRA, que desempenham um papel importante no processo de pesquisa, fornecendo *feedback* e *insights* valiosos.

3.3 GERAÇÃO DE DADOS

Para a geração dos dados iniciais, foi utilizado o modelo Stable Beluga 7B GPTQ. Com o auxílio de um *prompt* específico que se encontra no GitHub em "*https://github.com/LucasCrossDimitri/QLoRA-Adaptor-TCC*". Foram geradas 18256 linhas de interação entre o atendente e o usuário da empresa "CrossDimitriSolutions".

A metodologia em questão está sendo utilizada apenas como exemplo para este artigo. Normalmente, para melhorar o desempenho tanto do Adaptor QLoRA (como é chamada a versão menor de um modelo completo) quanto para outros estilos de treino, para treinar um modelo de IA, é necessário captar dados reais e não gerados como demonstrado no artigo: "*THE CURSE OF RECURSION: TRAINING ON GENERATED DATA MAKES MODELS FORGET*" consta:

“Descobrimos que aprender a partir de dados produzidos por outros modelos causa o colapso do modelo - um processo degenerativo pelo qual, ao longo do tempo, os modelos esquecem a verdadeira distribuição de dados subjacente, mesmo na ausência de uma mudança na distribuição ao longo do tempo. Damos exemplos de colapso do modelo para Modelos de Mistura Gaussiana (GMMs), Autoencoders Variacionais (VAE) e Modelos de Linguagem Grande (LLMs). Mostramos que, com o tempo, começamos a perder informações sobre a verdadeira distribuição, que começa primeiro com o desaparecimento das extremidades, e ao longo das gerações os comportamentos aprendidos começam a convergir para uma estimativa de ponto com variância muito pequena. Além disso, mostramos que esse processo é inevitável, mesmo para casos com condições quase ideais para aprendizado a longo prazo, ou seja, sem erro de estimativa de função.” (Shumailov, Shumaylov at al., 2023, p 2).

3.3.1 Pré-processamento de Dados: Após a geração dos dados, foi realizado um tratamento para unificar as interações dos “*User*:” e “*Assistant*:” em uma única interação e transformar o *dataset* no modelo do Stable Beluga, utilizando os marcadores "*### Comment*:", "*### REPLY*:", "*### END*.". Foi realizada uma checagem de dados e finalizações, adicionando um cabeçalho ao *dataset* para garantir que todo o *dataset* segue o mesmo padrão, o que é crucial para o treinamento.

Além disso, foi necessário realizar um tratamento para verificar a ocorrência de linhas de interação que não continham pergunta e resposta. Poucas foram as linhas em que a resposta não estava correta. Posteriormente, procedeu-se à limpeza de algumas linhas que, embora estivessem de acordo com o padrão do modelo e tivessem passado por todo o filtro pré-estabelecido, acumularam erros e agregaram várias interações dentro da linha, mesmo sem uma ter relação com a outra. Isso exigiu a exclusão manual de mais 3 linhas. Como é comum no treinamento de um modelo, a coleta de dados, seu ajuste e filtro são os processos que mais consomem tempo.

3.3.2 Transformação e *Upload* do *Dataset*: Com o *dataset* pronto e padronizado, foi realizada sua transformação de *txt* para *parquet*[16] utilizando o código apresentado no

[16] Formato de arquivo de código aberto feito para armazenar grandes *datasets* de maneira eficiente

GitHub do projeto em "https://github.com/LucasCrossDimitri/QLoRA-Adaptor-TCC". O formato *parquet* é ideal para conjuntos de dados grandes, pois é um formato de arquivo colunar que é suportado pela maioria dos sistemas de processamento de dados em larga escala. O *dataset* foi finalizado com um total de 3747 interações entre o usuário e o assistente, formatado para a formatação *parquet* e então carregado para o Hugging Face em "https://huggingface.co/datasets/CrossDimitri/CrossDimitriSolutionsText".

3.3.3 Treino do Adaptor QLoRA: O ambiente para o *fine-tuning* foi configurado no Google Colab. Isso envolveu a definição de parâmetros e teste da melhor configuração dentro dos limites do Colab, utilizando apenas a versão gratuita.

Para o treinamento, foram utilizadas uma variedade de bibliotecas. A seguir, serão comentadas as principais e feita uma breve explicação de cada uma:

Transformers oferece arquiteturas de propósito geral para Compreensão da Linguagem Natural (NLU) e Geração de Linguagem Natural (NLG), fornecendo APIs(Interface de Programação de Aplicações) e ferramentas para facilitar o *download* e o treinamento de modelos pré-treinados de última geração. A utilização de modelos pré-treinados pode reduzir seus custos computacionais e sua pegada de carbono.

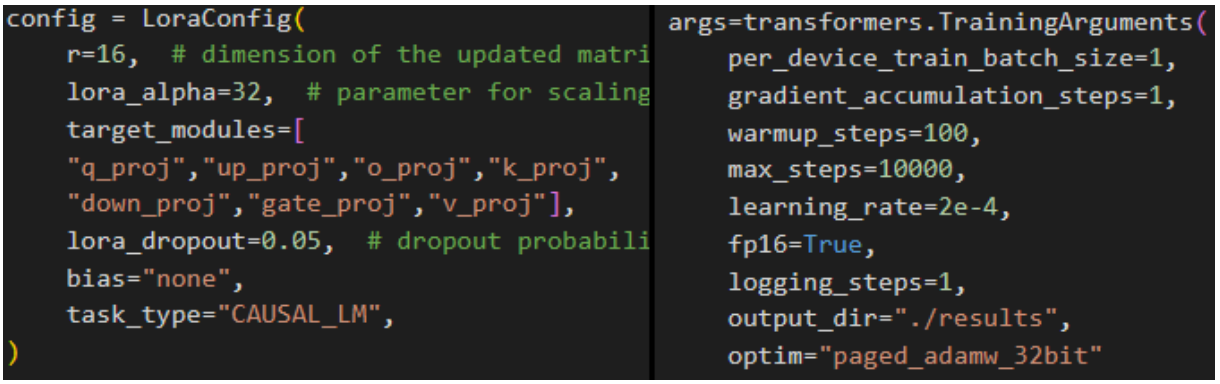
"Bitsandbytes" é uma biblioteca de quantização que inclui suporte para quantização de 4 *bits* e 8 *bits*. A quantização reduz o tamanho do seu modelo em comparação com sua versão de precisão total nativa, tornando mais fácil acomodar modelos grandes em GPUs com memória limitada.

"Accelerate" é uma biblioteca que possibilita a execução do mesmo código PyTorch em qualquer configuração distribuída. Em suma, proporciona um treinamento e inferência em escala de maneira simplificada, eficiente e adaptável.

PEFT, ou "*Fine-Tuning* Eficiente em Parâmetros" (do inglês, *Parameter-Efficient Fine-Tuning*), é uma biblioteca para adaptar de maneira eficiente modelos de linguagem pré-treinados (PLMs) para várias aplicações *downstream* sem a necessidade de ajustar todos os parâmetros do modelo. Os métodos PEFT ajustam apenas um pequeno número de parâmetros (extras) do modelo, diminuindo significativamente os custos computacionais e de armazenamento, pois o *fine-tuning* de PLMs em larga escala é proibitivamente caro. As técnicas PEFT de última geração alcançam um desempenho comparável ao do *fine-tuning* completo.

“AutoGPTQ ” colaborou com a biblioteca Optimum para fornecer uma API simples que aplica a quantização GPTQ em modelos de linguagem. Com a quantização GPTQ, você pode quantizar seu modelo de linguagem favorito para 8, 4, 3 ou até 2 *bits*. Isso ocorre sem uma grande queda de desempenho e com uma velocidade de inferência mais rápida.

Figura 2: Exemplo das configurações de treino para o Adaptador QLoRA



```
config = LoraConfig(
    r=16,  # dimension of the updated matri
    lora_alpha=32,  # parameter for scaling
    target_modules=[
    "q_proj","up_proj","o_proj","k_proj",
    "down_proj","gate_proj","v_proj"],
    lora_dropout=0.05,  # dropout probabili
    bias="none",
    task_type="CAUSAL_LM",
)
```

```
args=transformers.TrainingArguments(
    per_device_train_batch_size=1,
    gradient_accumulation_steps=1,
    warmup_steps=100,
    max_steps=10000,
    learning_rate=2e-4,
    fp16=True,
    logging_steps=1,
    output_dir="./results",
    optim="paged_adamw_32bit"
```

Fonte: O Autor (2023)

Conforme ilustrado na Figura 2, a configuração do treinamento QLoRA define o que será treinado no modelo. É possível passar como parâmetro a dimensão da matriz do modelo e sua escalabilidade, além de escolher seus módulos como se fossem pedaços de aprendizagem. Também é possível definir a qualidade do erro que é aceitável e se o modelo deve seguir algum padrão de regras. Além disso, é necessário informar o tipo do modelo que está sendo treinado.

À direita da imagem, os parâmetros de treinamento do Transformers são definidos. Estes parâmetros incluem o tamanho do lote de treinamento por dispositivo, o número de etapas para acumular os gradientes antes de realizar uma atualização dos parâmetros, o número de etapas de aquecimento durante o treinamento, o número de etapas de treinamento durante o processo de treinamento variou entre 1.000, 2.000, 4.000, 8.000 e 10.000 durante as avaliações, a taxa de aprendizado para o otimizador, o uso de precisão mista (FP16) durante o treinamento, a frequência de registro, o diretório onde os resultados do treinamento serão salvos e o otimizador usado para o treinamento.

Para mais detalhes sobre o código, criado por Lucas, está disponível no GitHub em “https://github.com/LucasCrossDimitri/QLoRA-Adaptor-TCC”.

3.3.4 Fine-tuning com Stable Beluga 7B GPTQ: Após o treinamento do modelo QLoRA e a realização de testes, observou-se que alguns modelos não foram suficientemente treinados para produzir resultados satisfatórios. O modelo treinado com 10.000 passos mostrou-se suficiente para produzir resultados satisfatórios e foi utilizado como um adaptador com o Stable Beluga 7B GPTQ.

Conforme representado nos exemplos que se encontram no GitHub em "*https://github.com/LucasCrossDimitri/QLoRA-Adaptor-TCC*". tem-se um modelo que aprendeu o que é a CrossDimitriSolutions e como são as respostas da empresa, proporcionando a personalização necessária para uma boa utilização de um chatbot empresarial. Com um conjunto de dados maior e um treinamento mais extenso, pode-se chegar a respostas quase perfeitas, passando-se por um funcionário.

Com a implementação e o tratamento da saída de dados do adaptador QLoRA, é possível alcançar um resultado ainda melhor.

3.3.5 *Benchmarks*: Conforme ilustrado na tabela 1, atualmente o Stable Beluga 7B não está entre os melhores modelos na classificação de 7 bilhões de parâmetros. No entanto, é importante ressaltar que alguns modelos nesse *ranking* são projetados especificamente para obter pontuações altas nos *benchmarks* específicos que são realizados, e seu desempenho para uso prático pode não corresponder à sua pontuação nos *rankings*.

Tabela 1: *Rank* de modelos 7 Bilhões - Top 10 e métricas do Stable Beluga-7B

Type	Model	Average	ARC	HellaSwag	MMLU	DROP
◆	fblgit/juanako-7b-UNA	59.91	68.17	85.34	62.47	38.74
◆	Intel/neural-chat-7b-v3-1	59.06	66.21	83.64	62.37	43.84
◆	Intel/neural-chat-7b-v3-1	59.04	66.3	83.6	62.44	44
◆	Intel/neural-chat-7b-v3-1	59.01	65.7	83.54	62.12	43.54
◆	Weyaxi/OpenHermes-2.5-ne	58.6	66.55	84.47	63.34	32.66
○	chargoddard/loyal-piano-m7	58.42	66.72	85.03	64.43	27.92
◆	Gryphe/MythoMist-7b	58.26	65.87	83.55	62.32	37.82
◆	teknium/OpenHermes-2.5-M	57.85	64.93	84.18	63.64	35.79
◆	teknium/OpenHermes-2.5-M	57.82	64.93	84.3	63.82	35.99
...	...	...	...	...	...	...
◆	circulus/Llama-2-7b-orca-v1	48.17	56.31	79.14	52.71	15.81
◆	stabilityai/StableBeluga-7B	48.17	56.31	79.14	52.71	15.81

Fonte: Open LLM Leaderboard (nov. 2023)

Atualmente, o Stable Beluga 7B tem uma média de 48.17, conforme demonstrado na tabela 1. Com a liberação semanal de novos modelos, muitos estão apresentando desempenho na vida real semelhante ou até superior ao do Stable Beluga 7B. No entanto, esses modelos ainda precisam passar por um teste humano que garanta que eles se desempenham bem para a função de chatbot.

No artigo “QLoRA: Efficient Finetuning of Quantized LLMs”, é mencionado que, “seguindo uma prática comum, o *benchmark* MMLU (Massively Multitask Language Understanding) é utilizado para medir o desempenho em uma variedade de tarefas de compreensão de linguagem. Este é um *benchmark* de múltipla escolha que abrange 57 tarefas, incluindo matemática elementar, história dos EUA, ciência da computação, direito, entre outros. A precisão do *5-shot* é verificada.“ (DETTMERS at al., 2023, p 8).

Focando no Benchmark do Stable beluga 2 na sua versão 70B, versão completa, temos no site da stability.ai (https://stability.ai), na seção *"news - Meet Stable Beluga 1 and Stable Beluga 2"* em 22 de novembro de 2023.
Falando que: *“As of July 27th, 2023, Stable Beluga 2 is the very best model (#1) on the leaderboard, and Stable Beluga 1 is #4:”*

Com o lançamento de novos modelos, o modelo Stable Beluga 2 70B não se encontra mais no topo do “Open LLM Leaderboard”, conforme observado no *leaderboard* da HuggingFace em (https://huggingface.co), na seção “spaces/HuggingFaceH4/open_llm_leaderboard” em 22 de novembro de 2023.

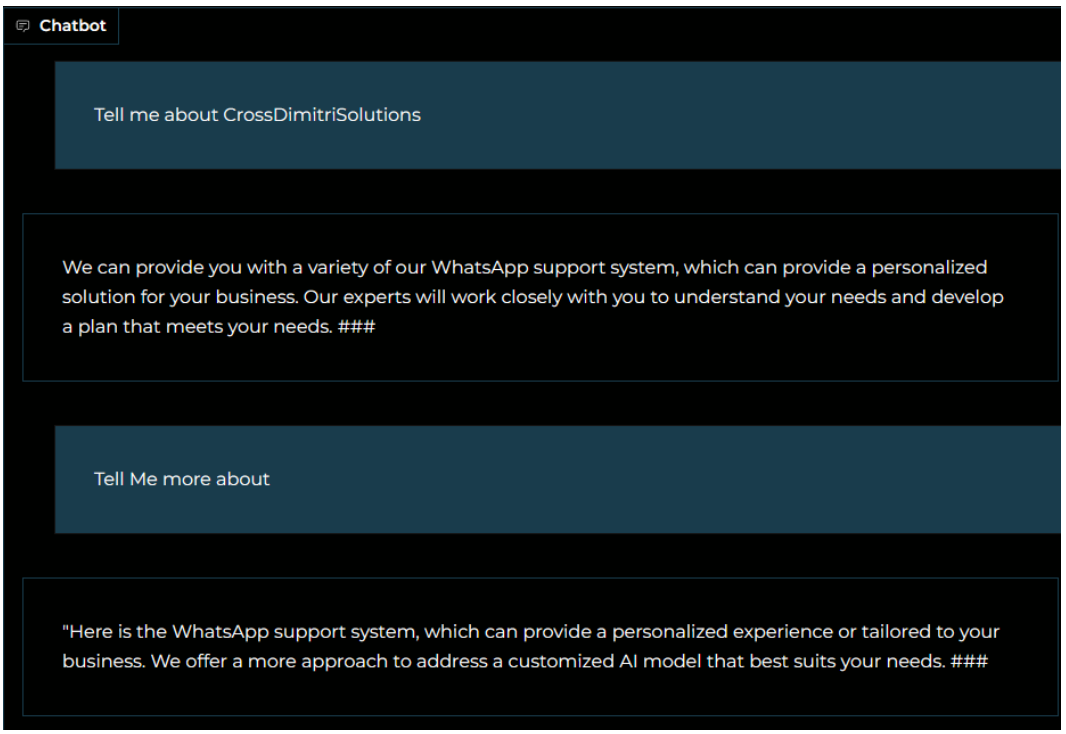
Porém ele ainda é um ótimo modelo que entrega uma performance comparada ao ChatGPT4 em vários *benchmarks*.

4 RESULTADOS E DISCUSSÃO

Neste artigo, são apresentados os resultados de um estudo sobre o desempenho de adaptadores QLoRA. Os resultados indicam uma customização aceitável dentro das limitações de *hardware* propostas durante o desenvolvimento do projeto. Os Exemplos estão disponível no GitHub em “*https://github.com/LucasCrossDimitri/QLoRA-Adaptor-TCC*”, é possível observar o modelo sem QLoRA respondendo sem compreender o contexto, gerando

informações potencialmente imprecisas sobre a empresa e rodando o adaptador QLoRA com o modelo na sua versão treinada de 10000 *steps*.

Figura 3: Exemplo do modelo rodando com Adaptador QLoRA



Fonte: O Autor (2023)

O resultado ainda não é consistente. Normalmente, ocorre entre 10 e 15 interações, começando com a frase "*Tell me about CrossDimitriSolutions*" como representado na figura 3. Isso ocorre porque o treinamento foi realizado com apenas 10.000 steps e o conjunto de dados é menor do que o normalmente usado. Além disso, conforme explicado no tópico 3.3.1, os dados foram gerados com o modelo, o que acarreta em um adicional de perda sendo transportado pelo conjunto de dados de volta ao modelo por meio do adaptador.

Com base nas informações apresentadas sobre MoE e o Estudo do SkunkworksAI no tópico 2.8, discute-se que o caso de uso em questão representa apenas a utilização inicial do adaptador QLoRA. Este estudo demonstra que, com custos reduzidos e *hardware* atual, é possível planejar pequenos modelos de arquitetura que se beneficiam do uso de menos *hardware* e alcançam resultados satisfatórios.

O grupo de estudo focou na utilização de GPU comerciais de até R$ 5.000,00 e no uso da versão gratuita do Colab para treinar o modelo entre 10.000 e 20.000 passos. Esta versão tem a limitação de uso de 12 horas por dia, e esse tempo médio vai reduzindo dependendo do que está sendo processado. Nos últimos dias de treino, o Colab já estava sem acesso à opção de GPU por uso extensivo de testes para treinar os modelos.

Com o comparativo apresentado, verifica-se a satisfação com o resultado encontrado. Antecipa-se que, com o avanço desse nicho de mercado que vem crescendo nos últimos anos, grandes empresas do segmento de *hardware* já estão produzindo equipamentos mais acessíveis com capacidade de executar esses modelos. Além disso, algumas delas podem vir a entregar novos modelos de processadores e GPUs especializadas em executar modelos de IA.

Atualmente, a Apple, uma empresa conhecida por sua inovação no setor de processadores de IA na área de processadores especializados em rodar modelos de IA para produtos comerciais, oferece os processadores M2 e M3. Esses processadores entregam uma performance aceitável entre 30 e 60 *tokens/s* para rodar modelos de 13B e 7B, conforme demonstrado por usuários neste fórum: arstechnica em “the-m3-max-pro-performance-comparison-thread”. (Ars Technica, 2023)

Por outro lado, sistemas com placas de vídeo de custo elevado conseguem rodar modelos entre 60-120 *tokens/s*. Claro que, com uma placa de vídeo dedicada, obtém-se o benefício de treinar modelos com um desempenho melhor.

Conforme discutido na seção 3.3.3, os treinamentos foram realizados utilizando a plataforma Colab, que apresenta algumas limitações. Nos testes de configurações, foi alcançado o limite de 10.000 treinos. No entanto, acredita-se que uma conta com poucas horas de uso possa realizar um treino de 20.000. É importante ressaltar que o tamanho do conjunto de dados influencia significativamente o tempo de treinamento. Considerando que o conjunto de dados utilizado é pequeno, são necessários muitos passos para começar a obter um resultado.

Normalmente, com conjuntos de dados maiores, resultados aceitáveis são alcançados com 3.000 a 4.000 passos. Além disso, a utilização de um conjunto de dados naturalmente criado com conversas reais tende a apresentar um desempenho muito melhor, pois é baseado em interações reais entre pessoas e abrange mais

casos de conversas que talvez nunca ocorreriam em um conjunto de dados criado. No caso deste estudo, houve dificuldades em encontrar a melhor configuração para as limitações que o Colab impõe.

Apesar dos desafios, foi possível obter resultados com 10.000 passos. Infelizmente, o tempo restante não permitiu a realização de mais testes com o projeto.

CONSIDERAÇÕES FINAIS

Os resultados do estudo sobre o desempenho de adaptadores QLoRA indicam uma customização aceitável dentro das limitações de *hardware* propostas durante o desenvolvimento do projeto.

Os resultados preliminares do estudo de SkunkworksAI sugerem que modelos MoE superam modelos densos em termos de FLOPs de inferência e também apresentam desempenho superior em relação ao seu nível de complexidade. Isso sugere que o adaptador QLoRA pode ser usado para criar modelos MoE mais eficientes e com melhor desempenho.

O estudo do SkunkworksAI também demonstrou que é possível planejar pequenos modelos de arquitetura que se beneficiam do uso de menos *hardware* e alcançam resultados satisfatórios. Em nosso estudo, o modelo foi treinado com 10.000 passos e um conjunto de dados pequeno e baixo custo. Mesmo com essas limitações, o modelo foi capaz de gerar respostas relevantes para as consultas.

O QLoRA, com sua capacidade de reduzir o custo operacional e aumentar a eficiência, representa um avanço significativo para o futuro da inteligência artificial. Ele tem o potencial de transformar a maneira como os modelos de IA são projetados e implementados. A possibilidade de planejar modelos menores que utilizam menos *hardware*, mas ainda assim alcançam resultados satisfatórios, abre um novo horizonte de possibilidades. No entanto, é necessário realizar mais pesquisas para explorar plenamente esse potencial.

Ao implementar este projeto, é recomendado verificar questões de segurança, pois os modelos podem expor dados de treinamento quando ocorre um colapso de modelo. Isso pode ocasionar um vazamento de dados acidental. Por conta disso, é

de suma importância que exista um controle minucioso dos dados coletados para a criação de um *dataset*.

AGRADECIMENTOS

Queremos expressar nossos sinceros agradecimentos ao professor Marcelo Petri, nosso orientador, cuja orientação e contribuições foram fundamentais para a elaboração deste artigo. Seus *insights* e *feedbacks* valiosos foram essenciais para o desenvolvimento do trabalho.

Agradecemos também à faculdade Unisociesc por disponibilizar os recursos acadêmicos essenciais para a realização deste estudo.

Às nossas famílias, que sempre nos proporcionaram o apoio necessário para continuar nossos estudos e explorar novas ideias, expressamos nossa gratidão.

Aos colegas de classe e amigos que conhecemos ao longo do curso de Engenharia da Computação, agradecemos por serem uma constante fonte de inspiração e parceria.

A todos mencionados acima, reconhecemos que o desenvolvimento deste trabalho não teria sido possível sem o apoio dedicado de cada um de vocês.

REFERÊNCIAS

APPLE. **Apple unveils M3, M3 Pro, and M3 Max — the most advanced chips ever for a personal computer**. Disponível em: https://www.apple.com/newsroom/2023/10/apple-unveils-m3-m3-pro-and-m3-max-the-most-advanced-chips-for-a-personal-computer/. Acesso em: 24 nov. 2023.

ARSTECHNICA. **M3 Max Pro Performance Comparison Thread.** Disponível em: https://arstechnica.com/civis/threads/the-m3-max-pro-performance-comparison-thread.1496840/. Acesso em: 24 nov. 2023.

BOSTROM, N. **Superintelligence: paths, dangers, strategies. Oxford: Oxford University Press**, 2014. 390 p. ISBN 978-0-19-967811-2.

DETTMERS, Tim; PAGNONI, Artidoro; HOLTZMAN, Ari; ZETTLEMOYER, Luke. **QLoRA: Efficient Finetuning of Quantized LLMs**. 2023. Disponível em: https://arxiv.org/abs/2305.14314. Acesso em: 14 nov. 2023

DU, Nan; HUANG, Yanping; DAI, Andrew M.; et al. **GLaM: Efficient Scaling of Language Models with Mixture-of-Experts**. 2022. Disponível em: https://arxiv.org/pdf/2112.06905.pdf. Acesso em: 24 nov. 2023.

DUA, Sumeet; DU, Xian. **Data mining and machine learning in cybersecurity. Auerbach Publications**, 2016. ISBN 978-1439839423.

Frantar, E.; Ashkboos, S.; Hoefler, T.; Alistarh, D. **GPTQ: ACCURATE POST-TRAINING QUANTIZATION FOR GENERATIVE PRE-TRAINED TRANSFORMERS.** 2023. Disponível em: https://arxiv.org/pdf/2210.17323.pdf. Acesso em: 20 nov. 2023.

GOERTZEL, B. **Artificial General Intelligence: Concept, State of the Art, and Future Prospects. Journal of Artificial General Intelligence**, v. 5, n. 1, 2014.

GOOGLE. **Google Colab**. Disponível em:https://research.google.com/colaboratory/intl/pt-BR/faq.html. Acesso em: 24 nov. 2023.

HENDRYCKS, D.; BURNS, C.; BASART, S.; ZOU, A.; MAZEIKA, M.; SONG, D.; STEINHARDT, J. **Medindo a compreensão da linguagem multitarefa massiva. In: INTERNATIONAL CONFERENCE ON LEARNING REPRESENTATIONS**, 2020. Acesso em: 22 nov. 2023.

HUANG, C.; ZHANG, Z.; MAO, B.; YAO, X. **An Overview of Artificial Intelligence Ethics. IEEE Transactions on Artificial Intelligence**, v. 4, n. 4, p. 799-819, ago. 2023. DOI: 10.1109/TAI.2022.3194503.

HU, Edward J.; SHEN, Yelong; WALLIS, Phillip; et al. **LoRA: Low-Rank Adaptation of Large Language Models**. 2021. Disponível em: https://arxiv.org/pdf/2106.09685.pdf. Acesso em: 24 nov. 2023.

HUGGING FACE. **Accelerate Documentation**. Disponível em: https://huggingface.co/docs/accelerate/index. Acesso em: 20 nov. 2023.

HUGGING FACE. **Auto-GPTQ Documentation**. Disponível em: https://huggingface.co/docs/optimum/main/en/llm_quantization/usage_guides/quantization#requirements. Acesso em: 20 nov. 2023.

HUGGING FACE. **bitsandbytes Documentation**. Disponível em: https://huggingface.co/docs/transformers/v4.35.2/en/perf_infer_gpu_one#bitsandbytes. Acesso em: 20 nov. 2023.

HUGGING FACE. **PEFT Documentation**. Disponível em: https://huggingface.co/docs/peft/index. Acesso em: 20 nov. 2023.

HUGGING FACE. **Summary of the tokenizers.** Disponível em: https://huggingface.co/docs/transformers/tokenizer_summary. Acesso em: 18 nov. 2023.

HUGGING FACE. **Transformers Documentation**. Disponível em: https://huggingface.co/docs/transformers/index. Acesso em: 20 nov. 2023.

HUGGING FACE SPACES. **Open LLM Leaderboard**. Disponível em: https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard. Acesso em: 22 nov. 2023.

INTEL. **Intel® Geti™**. Disponível em: https://geti.intel.com/. Acesso em: 24 nov. 2023.

KURZWEIL, R. **A singularidade está próxima: quando os humanos transcendem a biologia**. São Paulo: Aleph, 2006. 560 p. ISBN 85-7657-032-8.

MERRIAM-WEBSTER. **Definition of DATASET**. Disponível em: https://www.merriam-webster.com/dictionary/dataset. Acesso em: 21 nov. 2023.

META. **LLaMa**. Disponível em: https://ai.meta.com/llama/. Acesso em: 18 nov. 2023.

NVIDIA. **Microsoft Build: NVIDIA AI Powers New Generation of Windows 11 PCs, Workstations**. Disponível em: Driving Innovation for Windows PCs in Generative AI Era | NVIDIA Blog . Acesso em: 24 nov. 2023.

ORCA. **Orca: Progressive Learning from Complex Explanation Traces of GPT-4**. Disponível em:https://arxiv.org/pdf/2306.02707.pdf. Acesso em: 20 nov. 2023.

PROMPT ENGINEERING. **Master Prompt Engineering**. Disponível em: https://promptengineering.org/master-prompt-engineering-ai-prompt/. Acesso em: 21 nov. 2023.

SHUMAILOV, Ilia; SHUMAYLOV, Zakhar; ZHAO, Yiren; GAL, Yarin; PAPERNOT, Nicolas; ANDERSON, Ross. **The Curse of Recursion: Training on Generated Data Makes Models Forget**. https://arxiv.org/pdf/2305.17493.pdf. Acesso em: 21 nov. 2023.

SKUNKWORKS AI. **Hydra-MoE: A new class of Open-Source Mixture of Experts.** Disponível em: https://github.com/SkunkworksAI/hydra-moe. Acesso em: 24 nov. 2023.

STABILITY.AI. **Stable Beluga: Large Instruction Fine-Tuned Models**. Disponível em:https://stability.ai/news/stable-beluga-large-instruction-fine-tuned-models. Acesso em: 20 nov. 2023.

ZHONG, Wanjun; CUI, Ruixiang; GUO, Yiduo; LIANG, Yaobo; LU, Shuai; WANG, YANLIN, Saied, Amin; CHEN, Weizhu; DUAN. Nan. **AGIEval: A Human-Centric Benchmark for Evaluating Foundation Models**. 2023. Disponível em: https://arxiv.org/pdf/2304.06364.pdf. Acesso em: 22 nov. 2023.