VIII Congresso Nacional de Inovação e Tecnologia - INOVA 2025
06 e 07 de novembro de 2025
São Bento do Sul / Santa Catarina - Brasil
ISSN 2526-3145

Comparativo entre as técnicas RAG e ajuste fino para a injeção de conhecimento específico em LLM

Comparison between RAG and fine-tuning techniques for specific knowledge injection in LLM

João Paulo Siqueira [*], Leandro Correa Pykosz [†]

RESUMO

O surgimento do ChatGPT popularizou o uso dos assistentes virtuais. Ele foi desenvolvido usando o GPT, um tipo de inteligência artificial (IA) conhecido como Large Language Model (LLM) ou grande modelo de linguagem. Os LLMs são treinados com enormes quantidades de textos, possuindo domínio em diversas áreas. Contudo, seu conhecimento se limita aos dados usados no seu treinamento e ao período em que foram coletados. Para usar um LLM como um assistente virtual personalizado, é necessário que ele aprenda certos conhecimentos específicos. Este trabalho busca comparar a técnica de geração aumentada de recuperação (RAG) com o treinamento via ajuste fino do modelo usando low-rank adaptation (LoRA) para o desenvolvimento de um assistente virtual personalizado sobre uma instituição de ensino. Para isso, textos coletados sobre a instituição foram utilizados na criação do conjunto de dados usados no treinamento e como repositório de dados para o RAG. Perguntas sobre a instituição foram feitas para ambas as técnicas, e suas respostas avaliadas por voluntários, a qual escolheram às cegas a melhor resposta. A técnica RAG demostrou taxa de escolha de 42,11%, superior ao resultado do ajuste fino, com apenas 15,79%. Trabalhos futuros podem ser feitos para melhorar os resultados.

PALAVRAS-CHAVE: Grandes modelos de linguagens; Geração aumentada de recuperação; Ajuste fino.

ABSTRACT

The emergence of ChatGPT popularized the use of virtual assistants. It was developed using GPT, a type of artificial intelligence (AI) known as a Large Language Model (LLM). LLMs are trained with enormous amounts of text, possessing expertise in various fields. However, their knowledge is limited to the data used in their training and to the period in which it was collected. To use an LLM as a personalized virtual assistant, it must acquire certain specific knowledge. This work aims to compare the Retrieval-Augmented Generation (RAG) technique with model fine-tuning using Low-Rank Adaptation (LoRA) for the development of a personalized virtual assistant focused on an educational institution. For this purpose, texts collected about the institution were used to create the dataset for training and as a data repository for RAG. Questions about the institution were posed to both techniques, and their answers were evaluated by volunteers, who blindly chose the best response. The RAG technique achieved a selection rate of 42.11%, higher than the fine-tuning result, which reached only 15.79%. Future work can be carried out to improve these results.

KEYWORDS: *Large language models; Retrieval-augmented generation; Fine-tuning.*

1 INTRODUÇÃO

Apenas dois meses após o seu lançamento, o ChatGPT alcançou a marca de 100 milhões de usuários, essa taxa de crescimento é sem precedentes para um aplicativo de usuário final (Milmo, 2023). O ChatGPT é um modelo de linguagem que usa a arquitetura GPT (*Generative Pre-trained Transformer*) da OpenAI (Ahmed, 2024). Essa arquitetura faz parte de um subconjunto da inteligência artificial generativa conhecido como *Large Language Model* (LLM) ou grande modelo de linguagem (Chen, 2024). Desde então, outros modelos LLM vêm sendo lançados, como a coleção de modelos LLaMA (*Large Language Model Meta AI*) da empresa *Meta* (Touvron, 2023a), ou o modelo Mistral 7B da empresa

* Universidade do Estado de Santa Catarina, São Bento do Sul, Santa Catarina, Brasil. ✉ joaopaulo1094@gmail.com
† Universidade do Estado de Santa Catarina, São Bento do Sul, Santa Catarina, Brasil. ✉ leandro.pykosz@udesc.br

Mistral AI lançado sob a licença Apache 2.0 (Jiang, 2023). Os modelos de código aberto podem ser customizados e até usados para fins comerciais, tornando o seu uso cada vez mais popular (Singh; Anurag, 2024).

Os LLM são capazes de processar linguagem natural e são projetados para compreender, gerar e traduzir linguagem em um nível que se assemelha à compreensão da linguagem humana (Singh; Anurag, 2024). Para que os modelos LLMs possam aprender os padrões, gramática e a semântica da linguagem humana, são necessárias grandes quantidades de dados textuais, onde são usados textos coletados da internet, livros, jornais e outras fontes (Chen, 2024). Para comparação, foram utilizados 45 terabytes de dados para treinar o modelo base do ChatGPT (Singh; Anurag, 2024).

Devido a enorme quantidade de dados, o treinamento necessita de extensos recursos computacionais, portanto as unidades de processamento gráfico (GPUs) são amplamente utilizadas (Chen, 2024). Os modelos da família Llama 2 foram treinados em 3.311.616 horas de GPU, usando uma combinação de dados disponíveis publicamente (Touvron, 2023b).

A partir dos dados, é realizado o pré-treinamento para criar os *fundation models*, ou modelos bases, ou modelos pré-treinados, que servem como base para um maior ajuste fino ou adaptação a uma ampla gama de tarefas posteriores (Singh; Anurag, 2024). O ajuste fino, ou *fine-tuning*, é o processo de utilizar um conjunto de dados para treinar ainda mais o modelo base em uma tarefa específica, é um aprendizado supervisionado permitindo especializar o modelo em tarefas relevantes de acordo com a necessidade (Chen, 2024).

Porém, o conhecimento dos LLMs não é ilimitado, embora eles sejam treinados com enormes quantidades de dados, ainda há informações que eles não têm como saber, como dados privados ou dados em tempo real (Bryant; Mukherjee, 2023). O conhecimento dos modelos também é limitado pelo *knowledge cutoff*, ou limite de conhecimento, que é a data limite dos dados usados no pré-treinados ou no ajuste fino (Fregly; Barth; Eigenbrode, 2023). Ao gerar uma resposta sobre uma informação que o modelo não tem acesso, os LLM geram alucinações, onde a resposta é factualmente incorreta, gerando algo sem sentido ou improvável (Bryant; Mukherjee, 2023). Além disso, devido ao limite de conhecimento, o modelo pode retornar uma resposta desatualizada em relação aos dados atuais (Fregly; Barth; Eigenbrode, 2023).

Todavia, é possível incorporar novos conhecimentos de domínio específico em LLMs por meio de técnicas como RAG ou pelo ajuste fino dos parâmetros do modelo (Gupta, 2024). Entretanto, fazer o ajuste fino de todos os parâmetros de um modelo é uma tarefa intensiva em recursos (Auffarth, 2023). Treinar o GPT-3 com 175 bilhões de parâmetro exigiria 1.2 terabytes de memória de vídeo, com LoRA essa quantidade é reduzida para 350 gigabytes (Hu, 2021). LoRA é uma técnica que permite reduzir o número de parâmetros treinados mantendo a qualidade comparada ao ajuste fino (Auffarth, 2023). Outra forma de fazer um LLM responder questões sobre conhecimento específico, é por meio da “Geração Aumentada de Recuperação”, em inglês Retrieval-Augmented Generation (RAG) (Lewis, 2020).

Este trabalho tem como objetivo comparar as duas técnicas de injeção de conhecimentos em LLMs, o treinamento via ajuste fino dos parâmetros usando LoRA e a técnica RAG. Com o propósito de descobrir qual a melhor delas para futuramente desenvolver um assistente virtual com conhecimento específico sobre uma instituição de ensino.

2 REFERENCIAL TEÓRICO

2.1 EMBEDDINGS

Um *embedding* é uma representação numérica do conteúdo em uma forma que a máquina possa compreender e processar, ele é um vetor que contêm o conteúdo semântico enquanto descarta detalhes irrelevantes (Auffarth, 2023). “Um *embedding* pega um conteúdo, como uma palavra, sentença, frase ou imagem, e o mapeia em um espaço vetorial multidimensional” Auffarth (2023).

Ao representar objetos de dados como vetores numéricos, podemos realizar operações matemáticas sobre eles e medir sua similaridade, ou calcular a distância entre dois embeddings (Auffarth, 2023).

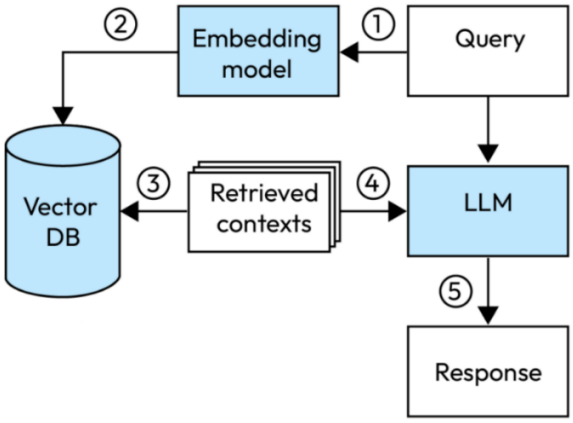
Um banco de dados vetorial pode ser utilizado para salvar os embeddings, esse tipo de banco de dados é capaz de realizar pesquisa vetoriais, retornando os vetores que são mais similares a partir de um vetor de entrada (Auffarth, 2023).

2.2 RAG

Auffarth (2023) define RAG como sendo “uma técnica que aprimora a geração de texto recuperando e incorporando conhecimento externo. Isto fundamenta o resultado em informações factuais, em vez de confiar apenas no conhecimento que está codificado nos parâmetros do modelo de linguagem”. É um método que fornece aos LLMs acesso ao conhecimento externo, melhorando sua precisão e proficiência em domínios específicos (Auffarth, 2023). A recuperação de contexto relevante é feita usando algoritmos de busca semântica, que envolve a indexação de documentos em vetores de *embeddings*, permitindo consultas de similaridade por meio de pesquisa aproximada do vizinho mais próximo (Auffarth, 2023).

Quando um usuário faz uma pergunta no método RAG, ela é convertida em um *embedding*, que é utilizado como consulta em um banco de dados vetorial, que por sua vez retorna os documentos próximo para serem incluídos para o LLM gerar a resposta (Singh; Anurag, 2024). Esse fluxo de trabalho para a geração da resposta do RAG pode ser visto na Figura 1.

Figura 1 – Fluxo de trabalho para geração da resposta no método RAG.



Fonte: Singh; Anurag (2024).

2.3 AJUSTE FINO

Fregly, Barthe Eigenbrode (2023) afirmam que o ajuste fino é um tipo de aprendizado de máquina supervisionado (SFT) que melhora o modelo comparando continuamente a saída do modelo para uma determinada entrada com o rótulo da verdade básica. O modelo primeiro faz uma previsão usando a entrada. Em seguida, compara a previsão com a saída para calcular o valor da perda, que é usado para propagar de volta pela rede neural, atualiza os parâmetros do modelo, para melhorar a previsão no futuro (Fregly; Barthe; Eigenbrode, 2023).

> Os Parameter-Efficient Fine-Tuning (PEFT) ou ajuste fino com eficiência de parâmetros são métodos que permitem o uso de pequenos pontos de verificação para cada tarefa, tornando os modelos mais portáveis. Este pequeno conjunto de pesos treinados pode ser adicionado ao LLM, permitindo que o mesmo modelo seja usado para múltiplas tarefas sem substituir o modelo inteiro. (Auffarth, 2023).

Auffarth (2023) define Low-Rank Adaptation (LoRA) como “um tipo de PEFT, onde os pesos do modelo pré-treinado são congelados. Ele introduz matrizes de decomposição de classificação treináveis em cada camada da arquitetura do Transformer para reduzir o número de parâmetros”. Esse método de treinamento possui qualidade comparável com o ajuste fino, ao mesmo tempo em que possui menos parâmetros treináveis e maior rendimento do treinamento (Auffarth, 2023).

3 METODOLOGIA

Textos sobre a instituição de ensino foram coletados da Internet a partir de fontes como o *site* oficial e páginas da Wikipédia. Vieram a ser coletados 9 textos, somando pouco menos do que seis mil palavras. Os arquivos foram salvos no formato texto, para serem usados na criação do conjunto de dados, e depois convertidos para o formato *markdown*, para serem usados nas etapas do RAG.

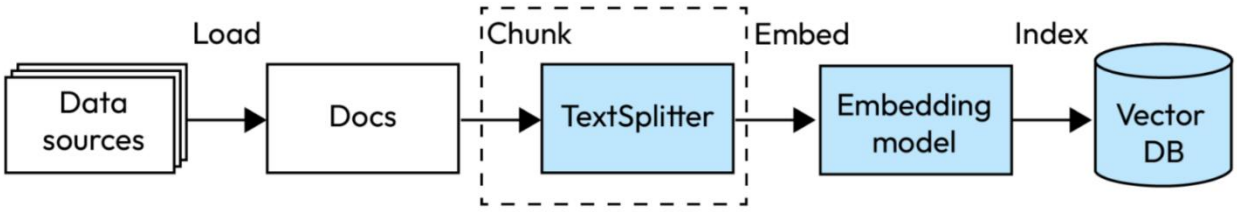
O processo de criação do conjunto de dados consistiu em utilizar um modelo de maior capacidade para processar os textos e criar os pares de perguntas e respostas, para essa tarefa foi escolhido o modelo Mixtral 8x7B Instruct v0.1 por ser disponibilizado sob

licença Apache 2.0 e ter melhor desempenho do que o modelo Mistral 7B Instruct v0.2 (Jiang, 2024). Processo semelhante é descrito por Gupta (2024) ao utilizar o modelo GPT para gerar perguntas e respostas com base nos documentos utilizados. Foi instruindo o modelo Mixtral a gerar as perguntas e suas respectivas respostas com base no texto de entrada e responder no formato JSON (*JavaScript Object Notation*), para facilitar a manipulação dos dados. Cada texto de entrada foi informado na íntegra, para que o modelo pudesse ter o contexto geral de seu conteúdo. Mas a geração dos pares de perguntas e resposta aconteceu em etapas, onde era informado qual trecho do texto o modelo deveria gerar as perguntas e respostas. Na sequência, era informado os demais trechos até que todo o texto tenha sido processado pelo modelo. As perguntas e respostas foram conferidas manualmente com as informações contidas no texto original e salvadas em um arquivo JSON, no total foram criados 359 pares de perguntas e respostas.

O conjunto de dados foi utilizado para treinar o modelo Mistral 7B Instruct v0.2, as perguntas e respostas foram formatadas no mesmo padrão de conversa "*chat template*" usados pelo modelo e usadas como entrada e saída para o treinamento. O treinamento foi feito via SFT utilizando LoRA, no total, foram feitos 1100 *steps*, equivalente a 3.06 *epochs*.

O método RAG é dividido em duas etapas, a etapa de processamento dos documentos, que pode ser visto na Figura 2 e a etapa da geração aumentada da resposta pode ser visto na Figura 1. Para o processamento dos documentos, os textos originais foram convertidos manualmente para o formato *markdown*, processo necessário, pois foi escolhido utilizar a função *MarkdownHeaderTextSplitter* da biblioteca *langchain* para fazer a divisão dos documentos nos pedaços de textos conhecido como *chunks*. Essa função divide os arquivos utilizando os cabeçalhos do documento *markdown*. Em seguida, caso um *chunk* passasse do limite de 512 tokens de tamanho, ele era dividido de forma recursiva usando a função *SemanticChunker* até que seu tamanho ficasse menor do que o limite. Foi escolhido o modelo de *embedding* mxbai-embed-large-v1 (Li, 2023), pois ele foi disponibilizado sob licença Apache 2.0. O modelo mxbai-embed-large-v1 é responsável por converter os *chunks* de textos em seus respectivos *embeddings*, ou seja, em uma representação numérica vetorial que captura o conteúdo semântico do texto (Auffarth, 2023). Os *embeddings* dos chunks são salvos no banco de dados vetorial Chroma DB, para serem consultados na etapa da geração aumentada. Na etapa de geração, assim que uma pergunta é feita, ela é convertida em um *embedding* e então utilizada para consultar no banco vetorial, os três *chunks* de texto mais similares, que então são incluídos como documentos de contexto para que o modelo possa responder com base neles.

Figura 2 – Etapa de processamento dos documentos do método RAG.



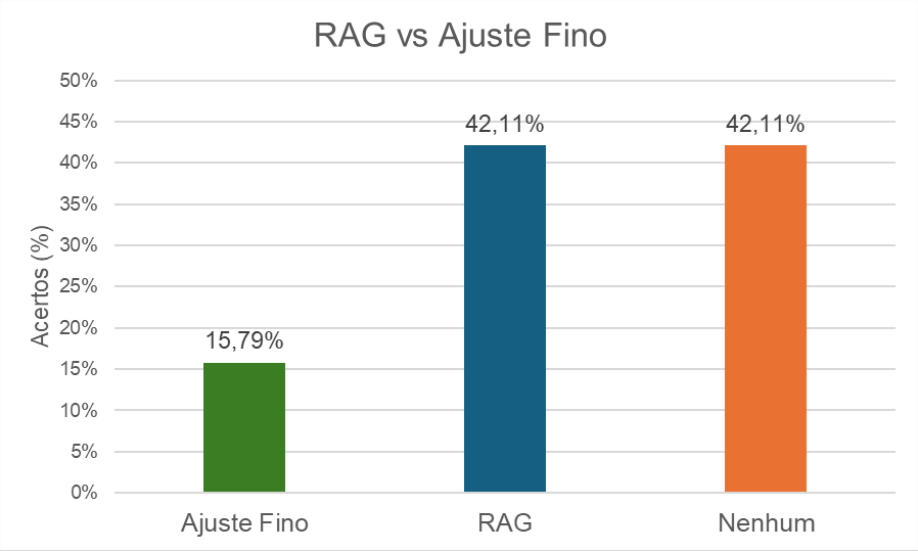
Fonte: Singh; Anurag (2024).

Para avaliar o desempenho da técnica RAG e ajuste fino, foi elaborado 19 perguntas com base nos textos originais, a lista de perguntas foi então submetida para que ambas as técnicas gerassem as suas respostas. Esse estudo usou três voluntários para avaliar as respostas, eles receberam as perguntas, as duas respostas geradas e os textos originais. Os voluntários avaliaram as técnicas sem saberem qual técnica gerou cada resposta. A avaliação consistiu em escolher entre uma das duas respostas ou nenhuma delas. Eles deveriam escolher a melhor opção dentre dessas três escolhas, e poderiam utilizar o texto original para poderem fazer a avaliação das respostas.

4 RESULTADOS E DISCUSSÕES

Ao contabilizar as escolhas dos voluntários via somatória simples, calculamos a porcentagem de acerto de cada uma das alternativas. Conforme pode ser observado na Figura 3, o resultado do treinamento por ajuste fino teve o pior desempenho, com apenas 15,79% de acertos, mostrando que essa técnica não foi eficaz em adicionar conhecimento específico ao LLM. Conforme demostrado por Ovadia (2023), é possível utilizar paráfrases para aumentar o conjunto de dados e melhorar o desempenho do ajuste fino. Isso não foi abordado neste trabalho, mas é um caminho para ser abordado em trabalhos futuros para se tentar melhorar os resultados do ajuste fino. O método RAG teve melhor desempenho com 42,11%, sendo mais eficiente do que o ajuste fino para adicionar conhecimentos externos ao modelo. Resultado semelhante é encontrado por Ovadia (2023), onde a técnica RAG teve desempenho consistente melhor do que apenas o ajuste fino. A porcentagem das respostas recusadas, onde nenhuma das duas respostas geradas pelas duas abordagens foi escolhida, também ficou em 42,11%, indicando que os voluntários recusaram as respostas de ambas as técnicas.

Figura 3 – Porcentagem de acertos das alternativas.



Fonte: autoria própria (2024).

Algumas respostas do método RAG foram geradas em inglês, isso pode ser resolvido alterando o prompt de sistema, para instruir o modelo à sempre responder no idioma português.

Alguns textos usados no trabalho continham informações desatualizadas, o que pode ter influenciado o desempenho de ambas as abordagens, esses textos poderiam passar por uma revisão para serem atualizados.

5 CONSIDERAÇÕES FINAIS

Este trabalho utilizou apenas um LLM, o Mistral 7B Instruct v0.2, em futuros trabalhos podem ser interessante comparar o desempenho de outros LLM. Além disso, foi utilizado apenas 9 textos, para futuras pesquisas, pode ser interessante utilizar a técnica de paráfrase para aumentar o conjunto de dados e tentar melhorar o desempenho do ajuste fino. Aliás, foi utilizado apenas 3 voluntários e apenas 19 questões para avaliar o desempenho das técnicas, métodos de avaliação mais robustos devem ser aplicados para se ter resultados mais fundamentados. Outros métodos mais eficientes de treinamento podem ser usados, e métodos mais avançados de RAG, com outros modelos de *embedding* poder ser testados. Para completar, conforme descrito por (Ovadia, 2023), combinar as técnicas de RAG e ajuste fino podem melhorar o desempenho geral, isso não foi feito neste trabalho, mas abre espaço para os próximos trabalhos.

Conforme demostrado pelos resultados, o método RAG se mostrou mais promissor do que o treinamento por via ajuste fino, entretanto a também alta porcentagem de respostas recusadas, indica que mais estudos de devem ser feitos para que se tenha uma conclusão mais embasada. Mesmos com esses resultados preliminares, é possível o uso do método RAG para o desenvolver um assistente virtual com foco em responder questões específicas, deste que as respostas do assistente virtual sejam consideradas com cautela sobre possíveis respostas inverídicas.

Agradecimentos

Agradeço a oportunidade de estudar em uma instituição de ensino de qualidade e gratuito. Agradeço ao programa de auxílio financeiro que permitiu a minha permanência no curso. Agradeço aos professores, família e amigos que apoiaram e ajudaram durante todo esse tempo. E em especial ao meu professor orientador aos voluntários que participaram do trabalho.

Disponibilidade de código

Como se trada de um estudo preliminar, o código fonte não está finalizado, ele ainda passará por diversas mudanças. Por esse motivo, não disponibilizarei os códigos fontes.

Conflito de interesse

Não há conflito de interesse.

REFERÊNCIAS

MILMO, Dan. ChatGPT reaches 100 million users two months after launch. **The Guardian**, 02 fev. 2023. Disponível em: https://www.theguardian.com/technology/2023/feb/02/chatgpt-100-million-users-open-ai-fastest-growing-app. Acesso em: 12 jun. 2024.

AHMED AL SHAMSI. **How to Use ChatGPT**. [s.l.] Ahmed Al Shamsi, 2024.

CHEN, J. **Demystifying Large Language Models**. [s.l.] James Chen, 2024.

TOUVRON, Hugo et al. Llama: Open and efficient foundation language models. **arXiv preprint arXiv:2302.13971**, 2023a.

JIANG, Albert Q. et al. Mistral 7B. **arXiv preprint arXiv:2310.06825**, 2023.

SINGH, P.; ANURAG KARUPARTI. **Generative AI for Cloud Solutions**. [s.l.] Packt Publishing Ltd, 2024.

TOUVRON, Hugo et al. Llama 2: Open foundation and fine-tuned chat models. **arXiv preprint arXiv:2307.09288**, 2023b.

BRYANT, J.; MUKHERJEE, A. **The Future of Finance with ChatGPT and Power BI**. [s.l.] Packt Publishing Ltd, 2023.

FREGLY, C.; BARTH, A.; EIGENBRODE, S. **Generative AI on AWS**. [s.l.] "O'Reilly Media, Inc.", 2023.

GUPTA, Aman et al. RAG vs Fine-tuning: Pipelines, Tradeoffs, and a Case Study on Agriculture. **arXiv preprint arXiv:2401.08406**, 2024.

AUFFARTH, B. **Generative AI with LangChain**. [s.l.] Packt Publishing Ltd, 2023.

HU, Edward J. et al. Lora: Low-rank adaptation of large language models. **arXiv preprint arXiv:2106.09685**, 2021.

LEWIS, Patrick et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. **Advances in Neural Information Processing Systems**, v. 33, p. 9459-9474, 2020.

JIANG, Albert Q. et al. Mixtral of experts. **arXiv preprint arXiv:2401.04088**, 2024.

OVADIA, Oded et al. Fine-tuning or retrieval? comparing knowledge injection in llms. **arXiv preprint arXiv:2312.05934**, 2023.

LI, Xianming; LI, Jing. Angle-optimized text embeddings. **arXiv preprint arXiv:2309.12871**, 2023.