

|  |  |
| --- | --- |
| Radix - IEL  Desafio |  |
| Bolsista Doutor [Ciência de Dados, Engenharia da Computação, Engenharia de Software] RADIX  Por Gilvan Wanderley de Farias Neto |  |

|  |  |
| --- | --- |
| INTRODUÇÃO 1.1 Contextualização  O estudo e o tratamento de grandes volumes de documentos financeiros, tais como relatórios, artigos e documentação específica do setor, têm sido uma atividade cada vez mais difícil devido ao aumento exponencial dos dados disponíveis. A exatidão, rapidez e eficiência na extração de informações fundamentais destes documentos são indispensáveis para tomadas de decisão sólidas no setor financeiro [1].  Para este fim, os modelos de linguagem de grande porte (Large Language Models - LLM) têm mostrado uma capacidade significativa para entender e gerar texto com alta precisão. Contudo, a aplicação direta desses modelos não é sempre a melhor alternativa, especialmente quando se trata de dados extremamente específicos e técnicos como os documentos financeiros [2].  É nesse contexto, que o conceito de Retrieval Augmented Generation (RAG) aparece como uma solução promissora. O RAG combina a capacidade de recuperação de informações relevantes de grandes conjuntos de dados com a geração de respostas contextualmente corretas e embasadas por LLMs. Essa abordagem permite acessar informações específicas de forma eficiente e também gerar respostas de alta qualidade e corretamente contextualizadas com base nas informações recuperadas [2].  1.2 Objetivos  Este projeto visa desenvolver um sistema de Retrieval Augmented Generation (RAG) aplicado a documentos financeiros, que aumente a eficiência e precisão da extração e geração de informações importantes a partir de grandes volumes de dados financeiros. Este sistema deve ser capaz de encontrar informação especifica em documentos financeiros e depois dar respostas contextualmente concisas, claramente informativas. FUNDAMENTOS TEÓRICOS 2.1 Conceitos de Retrieval Augmented Generation (RAG)  Retrieval Augmented Generation (RAG) é uma técnica que combina recuperação e geração de textos para melhorar sistemas de Processamento de Linguagem Natural (NLP). Esta abordagem tem sido especialmente útil em cenários onde os dados são vastos e variados, como documentos financeiros.  A técnica RAG consiste em duas grandes partes: o retriever e o generator. O retriever é responsável pela busca de informação relevante a partir de um grande corpus, enquanto o gerador usa essas informações reconstruídas para produzir respostas sejam precisas e coerentes em termos contextuais. A abordagem completa da técnica RAG é visualizada na Figura 1 abaixo, extraída do artigo “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks” [3].  A diagram of a computer program  Description automatically generated with medium confidence  Figura 1. Arquitetura de um sistema Retrieval Augmented Generation (RAG) [3].  O processo tem início com a entrada de uma consulta (query) *x*, que pode ser uma pergunta ou uma afirmação que precisa ser averiguada. Essa consulta é processada pelo *Query Encoder*, que transforma *x* em um vetor de consulta *q(x)*. Esse vetor é uma representação densa e de alta dimensionalidade da consulta, que captura suas características semânticas.  O vetor de consulta *q(x)* é utilizado pelo retriever *Pn* para procurar informações relevantes em um conjunto de documentos. O retriever faz essa busca usando MIPS (Maximum Inner Product Search), que é uma técnica que encontra documentos *d(z)* cujos vetores têm maior similaridade com o vetor da consulta *q(x)*. Como resultado, o retriever recupera os documentos *z1*, *x2*, …, *zk* que são considerados os mais relevantes para a consulta.  Os documentos recuperados são passados para o Generator *Pθ*, responsável por produzir a resposta final *y*. O Generator marginaliza sobre os documentos recuperados z para gerar respostas que são informadas pelo conteúdo de tais documentos. Esse processo garante que a resposta seja precisa e contextualizada de acordo com as informações mais relevantes encontradas pelo Retriever.  Segundo a Figura, o sistema tem uma capadidade end-to-end backpropagation. Em outras palavras, o Query Encoder e o Generator podem ser treinados conjuntamente para otimizar o desempenho do sistema. Desta forma, os parâmetros de ambos os componentes são ajustados de forma coordenada, melhorando a precisão tanto na recuperação quanto na geração de informações.  2.2 Aplicações de RAG em Documentos Financeiros  As técnicas de RAG proporcionam inúmeras aplicações no setor financeiro. Uma delas é na análise minuciosa de balanços e demonstrações contábeis, onde o sistema consegue recuperar seções específicas desses documentos complexos e sintetizar suas informações de forma automatizada, facilitando a compreensão rápida pelos analistas.  Na ajuda à tomada de decisão sobre investimentos, o RAG pode fornecer respostas contextuais precisas acerca do desempenho comparativo de empresas ao longo dos tempos, assim como entre setores distintos, agregando valor às análises.  A extração inteligente de notícias também é benéfica, onde matérias relevantes são localizadas e resumidas de maneira consolidada para que os profissionais das finanças se mantenham atualizados de forma eficaz sobre os acontecimentos do mercado.  No tocante à observância das leis e regulamentos, os sistemas RAG auxiliam na interpretação abrangente de documentos regulatórios, elucidando requisitos de aderência por meio de resumos precisos acerca de novas normativas.  No âmbito acadêmico e das pesquisas financeiras, o RAG facilita o acesso ágil a artigos significativos, sintetizando descobertas principais e auxiliando na revisão da literatura especializada acerca de tendências mercadológicas.  Por fim, no atendimento ao cliente, o RAG consegue responder consultas sobre produtos e políticas financeiras de forma contextualizada e precisa, contribuindo para a agilidade e qualidade dos serviços prestados.  2.3 Visão Geral sobre Large Language Model (LLMs)  Large Language Models (LLMs) são sistemas de machine learning treinados em conjuntos maciços de texto para compreender e gerar linguagem natural de maneira precisa. Estes modelos empregam arquiteturas sofisticadas de redes neurais, como *Transformers*, para capturar padrões complexos e contextos presentes na linguagem. Essa arquitetura foi introduzida pelo trabalho "*Attention is All You Need*" [4]. Essas redes neurais empregam mecanismos de atenção para processar simultaneamente as palavras de um texto, ao contrário de métodos sequenciais como RNNs (Redes Neurais Recorrentes). Tal abordagem permite que os *Transformers* lidem com longas sequências de texto de forma mais eficiente, capturando dependências contextuais de maneira mais eficaz.  O treinamento de LLMs envolve a exposição do modelo a uma vasta quantidade de textos para que ele aprenda as estatísticas da linguagem. Duas abordagens de treinamento são muito utilizadas: Autoregressive Language Modeling e Masked Language Modeling. Na primeira abordagem, o modelo é treinado para precer a próxima palavra em uma sequência de texto. Modelos como o GPT-3 usam esse tipo de treinamento. Na segunda abordagem, os modelos são treinados para preencher lacunas de textos. Modelos como o BERT, usam esse método.  A capacidade dos LLMs frequentemente é medida pela quantidade de parâmetros que eles contêm. Por exemplo, o GPT-3 (Generative Pre-trained Transformer 3) possui centenas de bilhões de parâmetros, o que permite capturar nuances linguísticas e contextuais de maneira detalhada. Essa enorme capacidade torna os LLMs altamente escaláveis para diferentes tarefas de NLP, desde a geração de texto até a compreensão da linguagem. Arquitetura do Sistema A Figura 1 apresentada ilustra um fluxo de processamento de documentos para busca semântica e integração com um modelo de linguagem grande (LLM) usando Streamlit.    Figura 1. Arquitetura do sistema  A primeira etapa é o carregamento dos documentos para o sistema. Em seguida, temos a etapa de Chunk dos Documentos, onde o documento carregado é dividido em pedaços menores, chamados de chunks. Posteriormente, cada chunk do documento é transformado em um vetor de alta dimensão, conhecido como embedding. Esses embeddings são representações numéricas que capturam o conteúdo semântico dos chunks.  Os embeddings gerados são armazenados em memória, mas poderiam ser armazenados em um banco de dados de embeddings. Este banco de dados serve como um repositório que pode ser consultado para realizar buscas semânticas.  O Streamlit é utilizado para criar uma interface de usuário interativa. Os usuários podem interagir com o sistema através desta interface, fazendo consultas e recebendo resultados em tempo real.  Quando um usuário faz uma consulta, o sistema utiliza a busca semântica para encontrar os embeddings no banco de dados que mais se assemelham ao contexto da consulta. Os embeddings relevantes encontrados na busca semântica são usados para recuperar os chunks de texto correspondentes do documento original. Esse contexto é essencial para fornecer respostas precisas e relevantes.  Os chunks de texto recuperados são então fornecidos ao modelo de linguagem grande (LLM). O LLM usa esse contexto para gerar respostas. Finalmente, as respostas geradas pelo LLM são apresentadas ao usuário através da interface do Streamlit, completando o ciclo de processamento.  Tanto para a LLM (“gpt-3.5-turbo”) quanto para o Embedding (“text-embedding-3-large”) foram utilizadas as configurações padrão por questão de praticidade. Em um cenário com mais tempo, seria possível fazer uma avaliação mais criteriosa desses métodos. Avaliação do RAG Para a avaliação das respostas do RAG desenvolvido, foram utilizados os resultados financeiros publicados pela RD Saúde (Raia Drogasil) nos quatro trimestres de 2023. Esses relatórios foram obtidos diretamente do site de relacionamento com os investidores, e os PDFs utilizados estão salvos na pasta docs.  A primeira pergunta avalia a capacidade de síntese do sistema, pois requer uma resposta que envolve todos os PDFs utilizados para trazer contexto. Como podemos ver na Figura 2, o modelo apresentou os 10 principais pontos que ele classificou como importantes. Comprando as informações e valores apresentados pelo sistema com as informações dos relatórios podemos observar coerência do modelo, Figura 3.  Interface gráfica do usuário, Texto, Aplicativo  Descrição gerada automaticamente  Figura 2. Primeira pergunta.  Texto  Descrição gerada automaticamente  Figura 3. Trecho do relatório do 4T23.  O modelo foi capaz de identificar quais foram os principais desafios para o ano de 2023, trazendo a seguinte frase:  *“Os principais desafios enfrentados no ano de 2023 mencionados no texto são: acelerar a digitalização da relação com o consumidor, aumentar a participação dos clientes digitais na venda total, personalizar a experiência do consumidor, acelerar os prazos de entrega, elevar o Net Promoter Score (NPS) dos serviços de entrega e dos aplicativos, e aumentar a produtividade dos squads.”*  Por fim, conforme mostrado na Figura abaixo, o sistema ainda foi capaz de fornecer recomendações para melhoria dos resultados no ano seguinte.    Figura 4. Recomendações para o próximo ano. Referências 1. Michael Werner, Michael Wiese, Annalouise Maas, Embedding process mining into financial statement audits, International Journal of Accounting Information Systems, Volume 41, 2021, 100514, ISSN 1467-0895, https://doi.org/10.1016/j.accinf.2021.100514. (https://www.sciencedirect.com/science/article/pii/S1467089521000166)  2. by I. de Zarzà, J. de Curtò 1,2,3ORCID,Gemma Roig 1,4ORCID andCarlos T. Calafate 2ORCID Optimized Financial Planning: Integrating Individual and Cooperative Budgeting Models with LLM Recommendations AI  3. Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS '20). Curran Associates Inc., Red Hook, NY, USA, Article 793, 9459–9474.  4. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 6000–6010. |  |