Sistema de Perguntas e Respostas (QA System) com LLM e RAG: Estudo de Caso Aplicado à Educação Financeira

Bolívar Francisco Braga[1], **Andrws Aires Vieira**[1]

[1]Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS)
Campus Ibirubá - RS - Brasil

msumib32@gmail.com, andrws.vieira@ibiruba.ifrs.edu.br

Abstract. *Financial education is fundamental for asset protection and wealth growth, but the complexity of its concepts often alienates novice audiences. Taking this into consideration, this study proposes the development of a Question and Answering (QA) system focused on democratizing financial knowledge, utilizing Large Language Models (LLMs) integrated into a Retrieval-Augmented Generation (RAG) architecture. The infrastructure was built entirely with open-source tools, supported by rigorous data extraction and preparation processes (ETL). The quantitative validation of the proposal was conducted via the automated RAGAS framework. The results demonstrated the system's effectiveness in mitigating hallucinations, achieving high precision in information retrieval (with a Context Recall metric above 0.99 for all models), and ensuring that the generated responses were strictly faithful to reliable sources.*

Resumo. *A educação financeira é fundamental para a proteção e o crescimento patrimonial, mas a complexidade de seus conceitos frequentemente afasta o público iniciante. Levando isso em consideração, este trabalho propõe o desenvolvimento de um sistema de Perguntas e Respostas (QA) focado na democratização do conhecimento financeiro, utilizando Modelos de Linguagem de Grande Escala (LLMs) integrados a uma arquitetura de Geração Aumentada por Recuperação (RAG). A infraestrutura foi construída inteiramente com ferramentas de código aberto, apoiada por processos rigorosos de extração e preparação de dados (ETL). A validação quantitativa da proposta foi conduzida via framework automatizado RAGAS. Os resultados demonstraram a eficácia do sistema em mitigar alucinações, alcançando alta precisão na recuperação da informação (com métrica Context Recall superior a 0,99 para todos os modelos) e garantindo que as respostas geradas fossem estritamente fiéis às fontes confiáveis.*

1. Introdução

Investimentos são uma das formas de combater a inflação e promover lucro para os investidores [Luintel and Paudyal 2006]. No entanto, apesar dos benefícios, muitas pessoas enfrentam dificuldades para entender e utilizar as ferramentas e técnicas disponíveis no mercado financeiro. Essa barreira de conhecimento acaba limitando o acesso da população a oportunidades de crescimento patrimonial, perpetuando desigualdades econômicas e reduzindo a inclusão financeira [Lusardi and Mitchell 2007].

Dessa forma, criar ferramentas que simplifiquem e tornem mais intuitivo o aprendizado sobre investimentos, especialmente para quem está começando, é crucial

[Hilgert et al. 2003]. Uma alternativa promissora é o uso de Modelos de Linguagem de Grande Escala (*Large Language Model - LLM*) em conjunto com sistemas de Geração Aumentada por Recuperação (*Retrieval-Augmented Generation - RAG*), capazes de fornecer respostas precisas e personalizadas com base em fontes confiáveis. Essas tecnologias podem transformar a maneira como as pessoas interagem com informações financeiras, simplificando conceitos complexos e oferecendo orientação prática [Lewis et al. 2020].

Trabalhos recentes têm explorado a integração entre LLMs e a arquitetura RAG para a construção de sistemas em domínios específicos. Chaudhary et al. (2024) desenvolveram um chatbot baseado na família LLaMA para suporte técnico em telecomunicações [Chaudhary et al. 2024]. De forma similar, Barbosa (2024) propôs um sistema contextualizado para assistência acadêmica universitária, alcançando 90,4% de acurácia global [Barbosa 2024]. No âmbito da educação financeira, Ramjattan et al. (2021) criaram um agente conversacional avaliado positivamente por 82% dos usuários em testes empíricos [Ramjattan et al. 2021].

Diante disso, este trabalho propõe o desenvolvimento e a avaliação de um sistema de Perguntas e Respostas (QA) baseado em LLMs, integrado à arquitetura RAG, com foco em educação financeira. O objetivo é oferecer respostas precisas, contextualizadas e acessíveis a usuários iniciantes, contribuindo para a democratização do conhecimento financeiro.

2. Materiais e Métodos

Este estudo se concentra no desenvolvimento de um sistema de perguntas e respostas, utilizando LLMs em conjunto com a arquitetura RAG, para fornecer respostas precisas sobre o mercado de investimentos. Para a execução deste projeto, foram empregadas as tecnologias a seguir.

2.1. Infraestrutura e Pipeline RAG

A orquestração do *pipeline* foi construída com o **Haystack** [deepset 2025], um *framework* modular que gerencia o fluxo desde o processamento de documentos (*DocumentSplitter*) até a geração final do texto (*PromptBuilder* e *Generator*). Para a interface com o usuário, utilizou-se o **Streamlit** [Streamlit 2026], provendo uma aplicação *web* fluida e interativa.

A etapa de recuperação de informação baseou-se no **Milvus** [Wang et al. 2021], um banco vetorial otimizado para buscas rápidas. A vetorização dos textos foi realizada pelo **all-MiniLM-L6-v2** [Wang et al. 2020], que gera *embeddings* de 384 dimensões. Para refinar a busca e garantir maior precisão, adotou-se o modelo *cross-encoder* **BAAI/bge-reranker-base** [Xiao et al. 2023], que reordena os documentos recuperados com base na interação semântica com a pergunta.

2.2. Modelos de Linguagem e Execução Local

Para garantir privacidade e eliminar custos com APIs, o **Ollama** [Ollama 2025] foi utilizado para o gerenciamento e execução local dos LLMs. Os seguintes modelos, de pesos abertos, foram avaliados: **Qwen 3** [Yang et al. 2025] que equilibra raciocínio e baixa latência; **DeepSeek-R1** [Guo et al. 2025] , que integra processamento de cadeia de raciocínio (CoT) nativo, útil para lógica financeira; e **LLaMA 3.1** [Sam 2024], focado em estabilidade.

2.3. Avaliação Automática (RAGAS)

A validação do sistema utilizou o *framework* **RAGAS** [Es et al. 2024], que opera sob o paradigma *LLM-as-a-Judge* para dispensar avaliação humana manual. O desempenho foi medido em duas frentes: **(i)Métricas de Recuperação:** *Context Recall* (avalia a proporção da informação relevante efetivamente recuperada ante a referência) e *Context Precision* (penaliza documentos irrelevantes no topo da busca); **(ii)Métricas de Geração:** *Faithfulness* (fidelidade ao contexto, principal indicador contra alucinações), *Factual Correctness* (exatidão factual perante o *ground truth*), *Semantic Similarity* (proximidade de significado via distância geométrica de *embeddings*) e *Answer Relevancy* (diretividade e completude da resposta à pergunta original).

3. Metodologia

3.1. Extração e preparação dos dados

O corpus deste trabalho foi construído sob uma abordagem bilíngue. A primeira fonte baseia-se no *dataset* público **"Investopedia Terms"**, disponibilizado no repositório Hugging Face [infCapital 2023]. Este conjunto de dados contém uma vasta coleção de termos financeiros, definições e conceitos fundamentais do mercado global. Para garantir que o sistema operasse com terminologias e conceitos alinhados à realidade do mercado financeiro brasileiro, foi escolhido o portal "Dicionário Financeiro". Essa fonte foi selecionada por oferecer explicações didáticas e localizadas em Português do Brasil [7Graus 2026].

O processo de preparação dos dados, conhecido como ETL (*Extract, Transform, Load*), foi automatizado por meio de scripts [1] desenvolvidos na linguagem Python com a biblioteca Pandas e um script[2] de *Web Scraping* utilizando as bibliotecas requests para comunicação HTTP e BeautifulSoup para o *parsing* do HTML. Iniciou-se a conversão do formato Parquet para CSV e na sequência o arquivo CSV foi transformado em arquivos de texto plano (.txt) individuais. O algoritmo mapeou os links via *sitemap.xml* e executou a limpeza dos dados, isolando a *tag articleBody*. Para preservar as tabelas do site, o algoritmo também transformou as tabelas HTML para a linguagem Markdown, por fim, cada artigo foi salvo como um arquivo de texto plano (.txt) limpo e pronto para indexação.

Para realizar a validação, foi construído um *golden dataset*[3] atuando como o padrão-ouro (ground truth) contendo os pares de perguntas e respostas corretas esperadas para avaliar o desempenho do sistema. Os dados foram organizados em arquivo CSV[4], resultando em 120 pares validados.

3.2. Pipeline de indexação e armazenamento vetorial

A indexação viabiliza a recuperação de documentos pelo *pipeline* RAG. Os documentos passam pelo *DocumentSplitter*, são vetorizados e armazenados no banco vetorial Milvus. Como estratégia de *chunking* realizou-se a segmentação dos textos via *DocumentSplitter*. Foi aplicado a técnica de janela deslizante (*sliding window*) em blocos de 250 palavras. Estabeleceu-se uma sobreposição (*overlap*) de 30 palavras entre *chunks* consecutivos.

[1]https://github.com/initdream/fined/tree/main/tools/parquet

[2]https://github.com/initdream/fined/blob/main/tools/scraper/dicionario.py

[3]Um golden dataset é um conjunto de dados de referência de alta qualidade e precisão, construído e validado com curadoria humana.

[4]https://github.com/initdream/fined/blob/main/eval/questions.csv

3.3. Seleção e configuração das LLMs

Optou-se pela utilização de modelos de pesos abertos executados localmente através do Ollama. Essa abordagem elimina custos de API e garante que os dados não sejam enviados a terceiros. Foram selecionados três modelos: Llama 3.1, DeepSeek-R1 e Qwen3.

3.4. Pipeline RAG

O módulo RAG é orquestrado pelo *framework* Haystack. A Figura 1 apresenta a arquitetura do sistema.

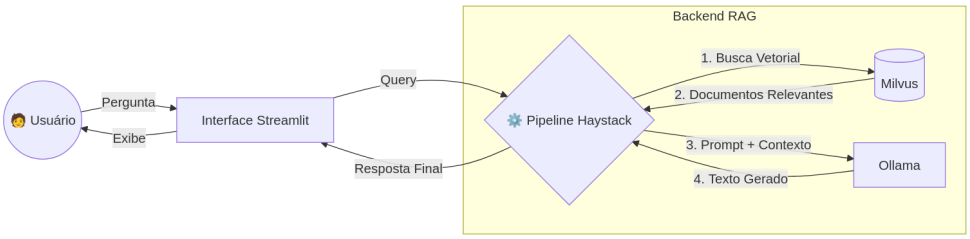

Figura 1. Arquitetura do sistema (Fonte: Autores)

O processo inicia com o *Text Embedder*, que traduz o *prompt* para vetor. O vetor é processado pelo *Retriever*, integrado ao Milvus, e extrai 20 documentos candidatos. Os dados passam pelo módulo *Reranker* (*Cross-Encoder*), que analisa o grau de correlação, reordenando-os e descartando documentos de baixa similaridade. O *Prompt Builder* age como integrador, combinando o *prompt*, histórico do *chat* e documentos recuperados. O template resultante é submetido ao LLM executado pelo Ollama, que retorna a resposta final.

3.4.1. *Prompt engineering*

Todos os modelos foram configurados com um *System Prompt*. O *template* foi dividido em: *system message* e *user message*. O prompt utilizado está disponível no GitHub[5].

3.5. Interface do Usuário

A interface foi construída utilizando Streamlit, apresentando campo de entrada de texto e histórico de conversa, operando em conjunto com o backend do Haystack.

3.6. *Pipeline* de validação

O *pipeline* de testes separa a geração da validação. A primeira etapa itera sobre o *golden dataset* e submete cada questão ao *pipeline* RAG. O histórico do chat é limpo a cada iteração. O algoritmo captura a resposta gerada e os blocos de contexto recuperados, salvando em arquivo JSON.

A segunda etapa é responsável pelo cálculo das métricas. O módulo consome o arquivo JSON e instancia o RAGAS. Para atuar como juiz, foi configurado o LLM Sabiá-3 [Abonizio et al. 2024] e o all-MiniLM-L6-v2 como modelo local de *embeddings*. O script executa o processamento das seis métricas e exporta um arquivo CSV.

[5]https://github.com/initdream/fined/blob/a445ec5382c1f99f7e3537b8c21a4a00e04a1401/chat_pipeline.py#L44

4. Resultados e discussões

Para avaliar o *pipeline* RAG e comparar os três LLMs, os resultados extraídos pelo *framework* RAGAS foram compilados. A Tabela 1 apresenta as médias obtidas, evidenciando desempenhos complementares: o modelo Qwen 3 destacou-se nas métricas de fidelidade (*Faithfulness*) e relevância (A*nswer Relevancy*), enquanto o DeepSeek-R1 obteve o melhor desempenho em exatidão factual (*Factual Correctness*).

Métrica	**Llama 3.1 (8B)**	**DeepSeek-R1**	**Qwen 3 (14B)**
Factual Correctness (F1)	0.449	**0.506**	0.464
LLM Context Recall	0.991	0.991	0.991
LLM Context Precision	0.936	**0.945**	0.944
Faithfulness	0.982	0.942	**0.992**
Semantic Similarity	0.857	0.834	**0.868**
Answer Relevancy	0.847	0.832	**0.875**

Tabela 1. Médias do desempenho dos modelos nas métricas do RAGAS

4.1. Discussão sobre o *Retrieval*

Conforme os dados na Tabela 1, a métrica LLM Context Recall atingiu média acima de 0.99 independentemente do LLM, o que indica que o módulo recuperou com sucesso as informações necessárias, confirmando a eficácia da arquitetura de duas etapas.

A métrica Context Precision manteve-se alta, com média de 0,941, o que confirma que o Reranker está classificando e filtrando os documentos corretamente. Essa estabilidade estabeleceu uma base sólida para a etapa de geração de texto.

4.2. Discussão sobre o *Generation*

A métrica *Faithfulness* apresentou valores próximos a 1 entre todos os modelos. Esse resultado demonstra que *prompt engineering* juntamente com instruções restritas contribuem para mitigar a alucinação. Os modelos basearam suas respostas no contexto fornecido, recusando-se a inventar dados.

Observou-se uma discrepância na métrica *Factual Correctness*, onde o DeepSeek-R1 obteve o melhor resultado (0,506). Apesar do Context Recall alto indicar que os modelos receberam o documento correto, o DeepSeek apresentou menor Faithfulness, possivelmente em função de sua arquitetura baseada em Cadeia de Raciocínio (Chain-of-Thought – CoT), que introduz passos lógicos intermediários na resposta. Essa discrepância se deve à maneira pela qual o modo F1 considera o *overlap* de *tokens*. Se a resposta de referência for curta e o LLM gerar uma explicação longa, a métrica reduz a pontuação, mesmo que a resposta seja fiel ao contexto. Em contrapartida, os valores altos obtidos pela métrica *Semantic Similarity* (média: 0,853) confirmam esta hipótese, indicando que o sentido geral das respostas estava correto, ainda que a estrutura divergisse da referência. Em relação à *Answer Relevancy*, o modelo Qwen apresentou a maior consistência.

Em uma análise qualitativa, foi observado um caso atípico em que *Factual Correctness* atingiu zero, enquanto *Faithfulness* e *Context Recall* obtiveram 1. Esse exemplo indica uma limitação do método F1-Score: o Retriever entregou o documento correto e o LLM utilizou os fatos, porém, a resposta gerada divergiu da referência. Esse exemplo

obteve 0,78 na métrica Semantic Similarity, indicando que a resposta não estava distante da referência. Esse comportamento ressalta a importância de avaliar sistemas RAG utilizando múltiplas métricas complementares.

5. Considerações finais

Este estudo propôs e validou um sistema de perguntas e respostas baseado em LLMs, integrado à arquitetura RAG, com foco em educação financeira. A proposta foi implementada com ferramentas de código aberto, permitindo execução local, controle dos dados e independência de serviços proprietários. Os resultados obtidos demonstram que arquiteturas RAG baseadas em ferramentas abertas constituem uma alternativa viável para aplicações especializadas de perguntas e respostas.

As métricas de *Context Recall* e *Context Precision* evidenciam que o *pipeline* de duas etapas (*Bi-Encoder* e *Cross-Encoder*) foi capaz de recuperar conteúdos relevantes, fornecendo uma base sólida para a geração das respostas. A avaliação automatizada via RAGAS confirmou que as respostas geradas ancoraram-se estritamente no *corpus* recuperado, com resultados quase perfeitos na métrica de fidelidade (*Faithfulness*), demonstrando que o sistema conseguiu mitigar significativamente o problema de alucinação. Em suma, a pesquisa atesta que a integração de LLMs locais com RAG constitui uma ferramenta computacional robusta para traduzir a complexidade do mercado financeiro.

Como trabalhos futuros, propõe-se a ampliação do *corpus* com dados dinâmicos do mercado financeiro. Reconhecendo as limitações inerentes às métricas automatizadas baseadas em similaridade, sugere-se adotar estratégias de avaliação híbridas (avaliadores humanos e automáticos) para uma análise mais robusta e precisa. Outra possibilidade é a exploração de técnicas avançadas de *prompt engineering* e *fine-tuning*, bem como a adaptação do sistema para outros domínios. Por fim, melhorias na interface podem contribuir para tornar a ferramenta mais acessível.

Declaração de Uso de IA Generativa

Ferramentas de IA generativa, especificamente o Gemini 3.1 Pro, foram utilizadas durante a elaboração deste trabalho para fins de revisão linguística e resumo.

Origem do Trabalho

Este trabalho tem origem no Trabalho de Conclusão de Curso (TCC) do curso de Ciência da Computação do IFRS – Campus Ibirubá.

Referências

7Graus (2026). Dicionário financeiro. `https://www.dicionariofinanceiro.com/`. [Accessed 16-03-2026].

Abonizio, H., Almeida, T. S., Laitz, T., Junior, R. M., Bonás, G. K., Nogueira, R., and Pires, R. (2024). Sabi\'a-3 technical report. *arXiv preprint arXiv:2410.12049*.

Barbosa, L. S. (2024). Um chatbot especializado para o contexto da universidade federal de ouro preto.

Chaudhary, D., Vadlamani, S. L., Thomas, D., Nejati, S., and Sabetzadeh, M. (2024). Developing a llama-based chatbot for ci/cd question answering: A case study at ericsson.

deepset (2025). Haystack: The production-ready open source ai framework. `https://haystack.deepset.ai`. Accessed on: 2025-06-18.

Es, S., James, J., Anke, L. E., and Schockaert, S. (2024). Ragas: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th conference of the european chapter of the association for computational linguistics: system demonstrations*, pages 150–158.

Guo, D., Yang, D., Zhang, H., Song, J., Wang, P., Zhu, Q., Xu, R., Zhang, R., Ma, S., Bi, X., et al. (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hilgert, M. A., Hogarth, J. M., and Beverly, S. G. (2003). Household financial management: The connection between knowledge and behavior. *Fed. Res. Bull.*, 89:309.

infCapital (2023). infcapital/investopedia_terms_en · datasets at hugging face huggingface.co. `https://huggingface.co/datasets/infCapital/investopedia_terms_en`. [Accessed 10-03-2026].

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Luintel, K. B. and Paudyal, K. (2006). Are common stocks a hedge against inflation? *Journal of Financial Research*, 29(1):1–19.

Lusardi, A. and Mitchell, O. S. (2007). Financial literacy and retirement planning: New evidence from the rand american life panel. Technical report, CFS working paper.

Ollama (2025). Ollama: Get up and running with large language models. `https://ollama.com`. Accessed on: 2025-06-18.

Ramjattan, R., Hosein, P., and Henry, N. (2021). Using chatbot technologies to help individuals make sound personalized financial decisions. In *2021 IEEE International Humanitarian Technology Conference (IHTC)*, pages 1–4.

Sam, K. (2024). Llama 3.1: An in-depth analysis of the next-generation large language model. *Available at SSRN 6139407*.

Streamlit (2026). Streamlit - a faster way to build and share data apps. `https://streamlit.io`. [Accessed 28-03-2026].

Wang, J., Yi, X., Guo, R., Jin, H., Xu, P., Li, S., Wang, X., Guo, X., Li, C., Xu, X., et al. (2021). Milvus: A purpose-built vector data management system. In *Proceedings of the 2021 international conference on management of data*, pages 2614–2627.

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., and Zhou, M. (2020). Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33:5776–5788.

Xiao, S., Liu, Z., Zhang, P., and Muennighoff, N. (2023). C-pack: Packaged resources to advance general chinese embedding.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al. (2025). Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.