Comparações de LLMs com RAG para o aprendizado de conceitos de agente BDI

Natália Mendes Goes[1], Rafael C. Cardoso[2], Gleifer V. Alves[1], André P. Borges[1]

[1]Universidade Tecnológica Federal do Paraná (UTFPR)
Ponta Grossa, PR, Brasil

[2]University of Aberdeen
Aberdeen, United Kingdom

nataliamendes@alunos.utfpr.edu.br, rafael.cardoso@abdn.ac.uk

{gleifer, apborges}@utfpr.edu.br

Abstract. *This paper presents a comparative analysis of different Large Language Models (LLMs) with Retrieval-Augmented Generation (RAG) to improve human understanding of Belief–Desire–Intention (BDI) agent concepts. The BDI agent model is a software model developed for programming intelligent agents, designed to simulate human-like reasoning and decision-making processes. The RAG integration aims to optimize the output of an LLM, reducing hallucinations and adding value to the user's response, thus facilitating the learning and interpretation of these agents. The study compares the technical characteristics of different LLMs, including an analysis of their size and response time. The results indicate that RAG integration improves the accuracy and clarity of the outputs, while the differences between the analyzed LLMs directly influence system performance and the effectiveness of BDI agent learning.*

Resumo. *Este artigo apresenta uma análise comparativa de diferentes modelos de Large Language Models (LLMs) com Retrieval-Augmented Generation (RAG) para aprimorar a compreensão humana dos conceitos de agentes Belief–Desire–Intention (BDI). O modelo de agente BDI é um modelo de software desenvolvido para programar agentes inteligentes, projetado para simular raciocínio e processos de tomada de decisão semelhantes aos humanos. A integração do RAG visa otimizar a saída de uma LLM, diminuindo as alucinações e agregando valor à resposta ao usuário, facilitando assim a aprendizagem e a interpretação desses agentes. O estudo compara características técnicas de diferentes LLMs, incluindo uma análise de seu tamanho e tempo de resposta. Os resultados indicam que a integração com RAG melhora a precisão e a clareza das saídas , enquanto as diferenças entre os LLMs analisados influenciam diretamente o desempenho do sistema e a efetividade do aprendizado de agentes BDI.*

1. Introdução

A área de estudos de agentes BDI (Crença-Desejo-Intenção) vem crescendo nos últimos anos [Cardoso and Ferrando 2021], atraindo maior interesse. No entanto, compreender seus conceitos ainda é desafiador para iniciantes devido à complexidade do conteúdo.

Para resolver essa questão, surge a necessidade da criação de um sistema que forneça fontes confiáveis e de fácil compreensão.

Os *Large Language Models* (LLMs) [Ma et al. 2023] compõem um tipo de Inteligência Artificial Generativa (IAG) projetado para entender, gerar e manipular a linguagem humana. Esses modelos são treinados em vastas quantidades de dados textuais, utilizam técnicas de *Deep Learning* (DL) e podem processar e analisar diversos conteúdos, como texto, imagens e áudio. A força dos LLMs reside em sua capacidade de vincular tópicos relacionados e fornecer respostas bem informadas sobre temas complexos. Infelizmente, a natureza da tecnologia LLM introduz imprevisibilidade nas respostas geradas e os dados usados para treinar os LLMs são fixos, o que implica que o conhecimento fornecido possui uma data limite. LLMs enfrentam desafios atualmente: fornecem informações falsas quando não possuem uma resposta adequada; oferecem informações desatualizadas ou genéricas quando os usuários solicitam respostas específicas e atualizadas; geram respostas baseadas em fontes não confiáveis. A técnica de *Retrieval-Augmented Generation* (RAG) [Lewis et al. 2020] foi introduzida para mitigar alguns desses problemas.

Existem diferenças significativas entre os modelos de LLMs que afetam o desempenho, a qualidade das respostas, tempo de processamento e a viabilidade de uso local. Neste trabalho, discute-se como o número de parâmetros e a arquitetura do modelo impactam os resultados obtidos de tarefas baseadas em linguagem natural, particularmente no contexto do sistema RAG. RAG é o processo de otimizar a saída de um grande modelo de linguagem, de forma que ele faça referência a uma base de conhecimento confiável fora das suas fontes de dados de treinamento antes de gerar uma resposta.

A contribuição deste trabalho é o desenvolvimento de um sistema baseado em RAG, projetado para minimizar alucinações sobre os conceitos de agentes BDI [Alan et al. 2025]. Para isso, foi montado um banco de dados com o livro do Michel E. Bratman de ”Intention, Plans, and Practical Reason” [Bratman 1987]. A relevância e o reconhecimento universal desse livro na comunidade de agentes inteligentes consolidam sua posição como a referência principal da área para o entendimento do raciocínio BDI. Espera-se que essa camada adicional de recuperação contribua para que as respostas permaneçam precisas e bem fundamentadas.

2. Fundamentos

Nesta seção, inicia-se a explicação de informações fundamentais sobre os LLMs que foram projetados para entender e gerar texto de forma semelhante à linguagem humana, se alinhando ao domínio da inteligência artificial.

2.1. LLMs

Os LLMs [Ma et al. 2023] são modelos de Machine Learning (ML) que empregam algoritmos DL para processar e compreender a linguagem natural. Treinados em vastas quantidades de dados, conseguem executar diversas tarefas de linguagem [Frering et al. 2025], incluindo tradução, resumo e geração de texto coerente e gramaticalmente correto.

No entanto, o uso de LLMs apresenta diversas limitações. Por exemplo, devido à natureza de seu treinamento, esses modelos podem refletir e perpetuar tendências presentes nos dados originais. Adicionalmente, LLMs são propensos a gerar informações falsas

ou enganosas (alucinações), uma vez que sua compreensão do mundo real é limitada aos padrões aprendidos. Frequentemente, também fornecem informações desatualizadas ou genéricas quando solicitadas respostas específicas, um desafio associado ao alto custo de atualização frequente de suas bases de dados de treinamento.

2.2. RAG

O RAG [Lewis et al. 2020] é uma técnica projetada para aprimorar a precisão e a confiabilidade de modelos de IA generativos, utilizando fatos obtidos de fontes externas predeterminadas. Na ausência do RAG, os LLMs dependem exclusivamente da entrada do usuário e de suas informações de treinamento para formular respostas. Com a implementação do RAG, um componente de recuperação de informações é introduzido, que utiliza a consulta do usuário para extrair dados relevantes de uma nova fonte. Tanto a consulta original quanto as informações recuperadas são então fornecidas ao LLM, permitindo respostas adaptadas com base nesse novo conhecimento e em seus dados de treinamento.

Mesmo quando as fontes de dados de treinamento originais de um LLM são adequadas, a manutenção de sua relevância é um desafio constante. O RAG capacita os desenvolvedores a fornecer dados de pesquisa, estatísticas ou notícias mais recentes diretamente aos modelos generativos. Essa abordagem permite, por exemplo, que o modelo entregue as informações recentes aos usuários.

2.3. Agentes BDI

O modelo de agente BDI [Cardoso and Ferrando 2021] é uma estrutura para o desenvolvimento de agentes inteligentes que emulam processos de raciocínio e tomada de decisão humanos. Sua base conceitual advém da teoria filosófica do raciocínio prático de Michael Bratman, articulada em torno de três atitudes principais: crenças, desejos e intenções.

As crenças representam o estado informacional do agente sobre si, o ambiente e outros agentes. Elas compreendem os fatos ou conhecimentos que o agente considera verdadeiros, mesmo que não reflitam com exatidão a realidade. São dinâmicas e se atualizam conforme o agente percebe mudanças no ambiente ou adquire novas informações.

Os desejos constituem o estado motivacional do agente, refletindo suas metas.Podem ser múltiplos e até conflitantes, funcionando como guia para a tomada de decisão, delineando um estado-alvo a ser atingido.

Intenções são os compromissos do agente em agir para realizar seus desejos. Correspondem a planos ou estratégias específicas que dão estabilidade à decisão, ao priorizar ações consistentes com os objetivos escolhidos.

A versatilidade dos agentes BDI é evidente em sua vasta gama de aplicações, que vão desde gerenciamento de tráfego aéreo e saúde eletrônica até automação de atendimento ao cliente [Corchado et al. 2004], robótica [Frering et al. 2025] e sistemas autônomos[Shen and O'Hare 2007]. Além disso, simplifica o design e o desenvolvimento de agentes, ao dissociar a fase de seleção de um plano da implementação das crenças, desejos e intenções do agente.

3. Método

Neste estudo, implementamos um sistema de RAG com o objetivo de diminuir as alucinações e aprimorar a precisão das respostas de LLMs [Alan et al. 2025]. A opção

por RAG, em detrimento do ajuste fino (*Fine-Tuning*), justifica-se pela natureza dos conceitos de agentes BDI. O Fine-Tuning, ao modificar os parâmetros internos de um modelo pré-treinado, molda seu conhecimento e comportamento diretamente, o que seria uma desvantagem dada a possibilidade de mudança nos conceitos BDI.

Assim, optou-se pelo RAG [Lewis et al. 2020] devido ao seu desempenho superior no tratamento tanto do conhecimento pré-existente do modelo quanto da capacidade de incorporar informações inteiramente novas, permitindo a fácil atualização da base de dados. A arquitetura do RAG apresentada na Figura 1 reflete o sistema específico implementado neste trabalho.

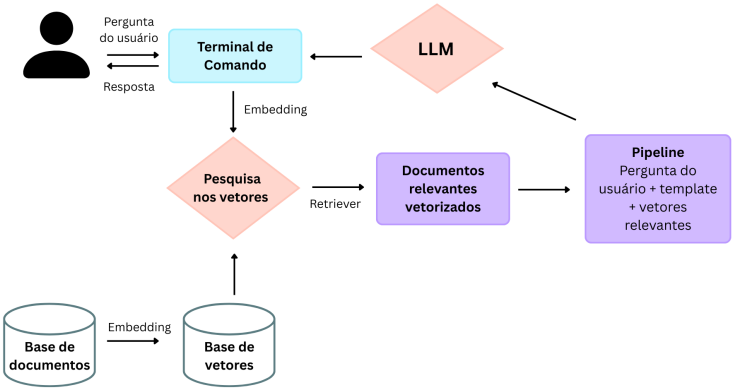

Figura 1. Arquitetura do sistema RAG implementado

3.1. Formando a Base de Dados

A base de documentos foi construída com base no livro "Intention, Plans, and Practical Reason" de Michael E. Bratman. O livro é uma obra seminal na filosofia da ação que revolucionou a compreensão de agente racional e o papel das intenções. Bratman apresenta uma "teoria do planejamento da intenção" que se opõe a abordagens anteriores, as quais frequentemente reduziam a intenção a uma mera combinação de crenças e desejos. Ressalta-se que o livro está disponível apenas em inglês, o que influenciou diretamente as respostas dos LLMs, conforme detalhado na Seção 4.2.

3.2. Criando o Banco de Dados Vetorial

Um banco de dados vetorial armazena e recupera vetores, representações numéricas de dados em um espaço multidimensional. Cada vetor codifica o significado semântico de um texto, e buscas por similaridade são feitas pela proximidade entre vetores. Essa funcionalidade é crucial para aplicações como sistemas RAG, que exigem buscas rápidas e precisas.

Para viabilizar a busca vetorial, o texto é dividido em blocos (chunks) de comprimento fixo a fim de preservar o contexto e a coerência lógica, definimos um tamanho de bloco de 500 caracteres com sobreposição (*overlap*) de 50 caracteres. Para superar os desafios relacionados às limitações de tamanho de bloco e à perda de contexto, foi aplicado o *Recursive Character Text Splitter* usando um método hierárquico com delimitadores

na seguinte ordem:“\n\n”, “\n”, “.”, “ ” e “” essa abordagem prioriza a manutenção da estrutura lógica e semântica do documento original.

Após o fracionamento, cada bloco de texto é transformado em um vetor numérico *embedding* por modelos treinados para capturar a semântica. Para este projeto, o modelo BAAI/bge-base-en-v1.5 foi o escolhido, reconhecido por sua otimização para tarefas de recuperação semântica.

3.3. Pesquisa no Vetor

Para realizar essa busca por informações relevantes, utilizou-se o FAISS (*Facebook AI Similarity Search*), uma biblioteca de código aberto projetada para pesquisa de similaridade e agrupamento de vetores densos. Após a base de conhecimento ser dividida em blocos e convertida em vetores, o FAISS indexa esses vetores em um banco vetorial. Quando o usuário envia uma pergunta, esta também é convertida em vetor e comparada com os vetores armazenados no índice. O FAISS então retorna os blocos mais semelhantes com base em medidas de similaridade, como a distância de cosseno. Esses blocos são utilizados para compor o contexto que será fornecido ao LLM, garantindo que ele tenha acesso às informações mais relevantes para gerar uma resposta precisa.

3.4. Prompt do Pipeline

Um prompt estrategicamente alinhado é crucial para garantir a qualidade satisfatória das respostas geradas por LLMs. Ao projetar um prompt para um LLM, é fundamental incluir o prompt do sistema, o contexto relevante recuperado e a pergunta do usuário. O prompt do sistema, por sua vez, é responsável por definir a formatação da resposta, estabelecer restrições e limitar o acréscimo de informações por parte do modelo.

3.5. Implementação

Neste projeto, utilizou-se o *LangChain*, um *framework* de código aberto projetado para simplificar o desenvolvimento de aplicações baseadas em LLMs. Ele oferece um conjunto de ferramentas e componentes que capacitam os desenvolvedores a conectar LLMs a diversas fontes de dados, facilitando a implementação de arquiteturas RAG. Em nossos experimentos, empregamos diversos modelos de LLM, cujas especificidades serão detalhadas na Seção 4.

A reprodutibilidade dos experimentos é viável em diferentes ambientes computacionais, no entanto, o tempo de execução será diretamente influenciado pelas especificações de hardware de cada máquina.

4. Comparações entre LLMs

Os LLMs são treinados com uma vasta gama de dados e cada um possui características específicas que refletem tanto sua empresa de origem quanto sua intenção de uso. Consequentemente, cada LLM avaliado neste projeto demonstrou resultados distintos, evidenciando as variações inerentes a esses modelos.

4.1. Características Técnicas dos LLMs

Os LLMs possuem características e arquiteturas distintas que impactam diretamente seu desempenho e a qualidade das respostas. Modelos com menos parâmetros, por exemplo, geralmente restringem a capacidade de compreender a linguagem natural.

Este estudo identificou que as limitações dos modelos na tradução e na compreensão de diferentes idiomas são relevantes. Perguntas em inglês resultaram em respostas mais consistentes; em português, entretanto, observou-se maior ocorrência de erros gramaticais e de escrita, indicando desempenho inferior. Tal comportamento decorre do fato de a base de dados estar predominantemente em inglês, o que influencia diretamente o processo de recuperação por similaridade. Como consequência, os trechos são transformados em vetores numéricos distintos, resultando em valores que afetam a precisão das respostas.

Além dos parâmetros, a qualidade e estrutura da resposta são influenciadas, notando-se desvios do tema proposto em alguns casos. O limite de tokens define a extensão que um LLM pode processar ou gerar em uma interação.

O investimento e a natureza da licença (*open source*) impactam diretamente o desempenho. LLMs proprietários (não *open source*) geralmente demonstram maior velocidade e qualidade nas respostas. Em contrapartida, os modelos *open source*, tendem a ter um desempenho inferior. As características técnicas de cada LLM podem ser encontradas na Tabela 1.

Tabela 1. Tabela com informações técnicas sobre as LLMs

Modelo	Parâmetro	Tokens	Open Source	Empresa
Gemini 1.5 Flash	Não divulgado	Até 1M tokens	Não	Google
Gemma 1.1 2B IT	2B	8k tokens	Sim (Licença Aberta)	Google
Phi-2	2.7B	4k tokens	Sim	Microsoft
Qwen3-8B	8B	128k tokens	Sim	Alibaba
Qwen1.5-0.5B	0.5B	2k tokens	Sim	Alibaba
TinyLlama-1.1B-Chat-v1.0	1.1B	2048 tokens	Sim	Comunidade
GPT-3.5-Turbo	Não divulgado	16k tokens	Não	OpenAI

4.2. Tempo de Resposta

O tempo de resposta de um LLM impacta diretamente a experiência do usuário, que geralmente espera uma interação quase imediata. Para o aprendizado de agentes BDI, a latência é particularmente crítica, pois um tempo de espera prolongado pode interferir no processo de aprendizagem.

A otimização dos LLMs em relação ao seu tamanho de dados e arquitetura é um fator determinante do tempo de resposta. Modelos mais estruturados tendem a apresentar melhor aproveitamento e, consequentemente, respostas mais rápidas.

A Tabela 2 apresenta os dados dos modelos avaliados, seus respectivos tempos de resposta e a quantidade de tokens utilizada para gerar as respostas. Para cada modelo, foram formuladas duas perguntas distintas.

Pergunta 1: What is the structure of a BDI agent's plan? Como é a estrutura de um plano de um agente BDI?

Pergunta 2: What is an intention in a BDI agent? O que é uma intenção em um agente BDI?

Tabela 2. Tempo de resposta dos LLMs para as Perguntas 1 e 2

Modelo	Pergunta 1		Pergunta 2	
	Tempo (min)	**Tokens**	**Tempo (min)**	**Tokens**
Gemini 1.5 Flash	**0.3**	**589**	0.47	610
Gemma 1.1 2B IT	0.59	561	0.78	575
Phi-2	2.85	839	1.89	689
Qwen3-8B	72.61	1142	67.74	1192
Qwen1.5-0.5B	0.93	777	0.76	719
TinyLlama-1.1B-Chat-v1.0	2.51	1108	**0.45**	**665**
GPT-3.5-Turbo	**0.25**	**66**	**0.18**	**49**

Os experimentos foram realizados em um sistema equipado com processador AMD Ryzen 7 5700U with Radeon Graphics, 1.80 GHz e 32GB de RAM. O sistema operacional utilizado foi o Windows 11. Para o desenvolvimento e execução, utilizou-se Python 3.13.2, juntamente com as bibliotecas transformers, torch, sentence-transformers, faiss-cpu, langchain, huggingface-hub. O software principal empregado foi Visual Studio Code versão 1.102.1.

4.3. Qualidade das Respostas

A qualidade das respostas geradas pelos LLMs é um fator importante na eficácia do aprendizado de agentes BDI. As variações observadas nas respostas foram analisadas para a pergunta 2 em diferentes LLMs, considerando suas características técnicas e desempenho. Conforme Tabela 1, os modelos utilizados possuem características técnicas diferentes que transparecem nas respostas. Além disso, os critérios avaliados contemplaram correção conceitual, cobertura, clareza, estruturação da explicação e aderência à pergunta.

Qwen1.5-0.5B: Este modelo demonstrou uma resposta com erros significativos de tradução e gramática para o português, além de apresentar uma compreensão conceitual confusa da "intenção" no contexto BDI. A resposta não reflete a definição padrão de intenção de Bratman.

Qwen3-8B: Em contraste, o Qwen3-8B, que possui uma base de dados maior, forneceu uma resposta correta em inglês, alinhada com os conceitos de intenção em agentes BDI. A qualidade do conteúdo gerado é notavelmente superior. Porém, o tempo de resposta foi o mais elevado.

Phi-2: Apresentou uma resposta em inglês compreensível e razoavelmente precisa, embora com uma explicação mais genérica do que uma definição formal.

TinyLlama-1.1B-Chat-v1.0: Este modelo gerou uma resposta concisa e alinhada com a definição de intenção, destacando o papel da intenção como um plano ou objetivo para atingir um propósito específico.

Gemini 1.5 Flash: Forneceu uma resposta em inglês precisa, focando na relação entre ação, desejos e crenças, e na natureza da intenção, além de mencionar sua função em planos futuros.

Gemma-1.1-2B-IT: Produziu uma resposta em inglês muito concisa e correta, focando no aspecto relacional da intenção com desejos e crenças do agente.

GPT-3.5-Turbo: O modelo produziu com precisão a resposta em inglês, alinhando-se com a definição de intenção em agentes BDI presente na literatura e utilizou o menor número de tokens.

5. Conclusão

Este estudo apresentou a aplicação do RAG em conjunto com LLMs para aprimorar a compreensão humana de agentes BDI. Demonstrou-se que a abordagem RAG é eficaz em diminuir as alucinações e aumentar a precisão das respostas geradas pelos LLMs.

A análise comparativa entre os diversos LLMs testados revelou que as características técnicas de cada modelo, como o número de parâmetros, a janela de contexto (tokens) e a natureza (open source *vs* proprietário), influenciam diretamente na qualidade, estrutura e tempo de resposta. Modelos com mais parâmetros, por exemplo, tenderam a apresentar desempenho superior em termos de velocidade e consistência, enquanto modelos menores ou open source exibiram variações e desafios, como erros gramaticais em respostas em português. A implementação do sistema RAG, que utilizou um banco de dados vetorial (FAISS) e blocos de texto do livro de Michael Bratman, mostrou-se fundamental para fornecer contexto relevante e preciso ao LLM, superando as limitações inerentes aos modelos que dependem apenas de seus dados de treinamento.

Em síntese, este trabalho não apenas validou o potencial do uso de LLMs com RAG para facilitar o aprendizado de agentes BDI, como também forneceu evidências relevantes acerca do impacto das características dos LLMs no desempenho global do sistema.

Em trabalhos futuros, planeja-se selecionar um dos modelos de LLM com maior aproveitamento para estender a aplicação do RAG ao ensino do framework **MASPY** [Izidorio et al. 2024], usado para programação de agentes em Python, combinando SMAs BDI com Aprendizado por Reforço. Adicionalmente, considera-se a possibilidade de incorporar agentes de IA no projeto para aprimorar a qualidade das respostas e a interação do sistema.

Agradecimentos: Este trabalho tem financiamento do projeto: 444568/2024-7, CNPq/MCTI/FNDCT Nº 22/2024, *Programa Conhecimento Brasil – Apoio a Projetos em Rede com Pesquisadores Brasileiros no Exterior.*

Referências

Alan, A. Y., Karaarslan, E., and Aydın, Ö. (2025). Improving llm reliability with rag in religious question-answering: Mufassirqas. *Turkish Journal of Engineering*, 9(3):544–559.

Bratman, M. (1987). Intention, plans, and practical reason.

Cardoso, R. C. and Ferrando, A. (2021). A review of agent-based programming for multi-agent systems. *Computers*, 10(2):16.

Corchado, J. M., Pavón, J., Corchado, E. S., and Castillo, L. F. (2004). Development of cbr-bdi agents: a tourist guide application. In *European Conference on Case-based Reasoning*, pages 547–559. Springer.

Frering, L., Steinbauer-Wagner, G., and Holzinger, A. (2025). Integrating belief-desire-intention agents with large language models for reliable human–robot interaction and explainable artificial intelligence. *Engineering Applications of Artificial Intelligence*, 141:109771.

Izidorio, F. M., Mellado, A. L., Borges, A. P., and Alves, G. V. (2024). Agentes bdi e aprendizagem: um mapeamento sistemático e utilização com a biblioteca maspy. In *Workshop-Escola de Sistemas de Agentes, seus Ambientes e Aplicações (WESAAC)*, pages 108–119. SBC.

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.

Ma, X., Fang, G., and Wang, X. (2023). Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720.

Shen, S. and O'Hare, G. M. (2007). Wireless sensor networks, an energy-aware and utility-based bdi agent approach. *International Journal of Sensor Networks*, 2(3-4):235–245.