Julgamento Automatizado de Agentes Conversacionais: Usando *LLM-as-a-Judge* para avaliar um software baseado em Inteligência Artificial para o Ensino de Computação

Danilo Guimarães Souza Azevedo[1], **Melques Santos Paiva**[1], **Ana Kessilly Chiachio Cerqueira**[1], **Crescencio Lima**[1], **Djan Almeida Santos**[1], **Luis Paulo da Silva Carvalho**[1]

[1]Instituto Federal de Educação, Ciência e Tecnologia da Bahia (IFBA)
Vitória da Conquista - BA

{dan.azevedo82, melque1703}@gmail.com

kessilly_chiachio@hotmail.com

{crescencio, djan.santos, luiscarvalho}@ifba.edu.br

Resumo. *A integração de Large Language Models (LLMs) em contextos educacionais exige validação rigorosa para evitar alucinações e garantir precisão. Este trabalho apresenta uma solução de avaliação automatizada baseada na técnica LLM-as-a-Judge, aplicada a um agente conversacional para o ensino de Ciência da Computação. O estudo descreve os fundamentos teóricos, a arquitetura da solução, a construção dos prompts e um fluxo de julgamento. Para validação, o consenso das IAs foi comparado à análise de um especialista humano. Os resultados indicam julgamentos coerentes e próximos aos do especialista, evidenciando o potencial da abordagem para apoiar a validação contínua de respostas tutoriais. Por fim, discutem-se limitações, desafios e perspectivas futuras.*

Abstract. *The integration of Large Language Models (LLMs) into educational settings requires rigorous validation to prevent hallucinations and ensure content accuracy. This study presents an automated evaluation approach based on the LLM-as-a-Judge technique, applied to a conversational agent for Computer Science education. The paper outlines the theoretical foundations, an architecture, prompt design, and evaluation workflow. To validate the approach, AI consensus was compared with a human expert's assessment. Results indicate judgments consistent with those of the specialist, highlighting the method's potential to support continuous validation of AI-generated tutoring responses. Finally, limitations, challenges, and future research directions are discussed.*

1. Introdução

LLM-as-a-Judge, ou "avaliação por LLM"(*LLM evaluation*), é uma técnica emergente que utiliza Grandes Modelos de Linguagem, ou, em inglês, *Large Language Models* (LLMs), como avaliadores automatizados para determinar a qualidade, relevância e eficácia de textos ou respostas geradas. Esta abordagem é proposta como uma alternativa escalável e eficiente à avaliação humana, que é frequentemente lenta e custosa

[Zheng et al. 2023]. O conceito baseia-se na capacidade dos LLMs de simular o raciocínio humano e aplicar critérios complexos de avaliação [Gu et al. 2025].

A utilização da técnica *LLM-as-a-Judge* oferece uma série de vantagens cruciais, especialmente em cenários que exigem avaliação de larga escala, velocidade e um alto grau de alinhamento com o julgamento humano. O principal benefício é a escalabilidade e a eficiência econômica, funcionando como uma alternativa promissora aos métodos tradicionais de avaliação humana [Zheng et al. 2023]. A utilização de *LLM-as-a-Judge* é significativamente mais barata e rápida, permitindo iterações mais velozes e a criação rápida de *benchmarks* escaláveis [Chiang and yi Lee 2023].

Diante de tais vantagens, este trabalho avalia a aplicação da técnica *LLM-as-a-Judge* na criação de uma solução *web* automatizada de avaliação (ou julgamento) de pares de pergunta-resposta advindos de softwares conversacionais. Inicialmente foi desenvolvido para tratar dos pares de pergunta-resposta originados de um Sistema de Informação baseado em Inteligência Artificial (IA), o TutorIA.

Atualmente, o TutorIA conta com um módulo simples de julgamento baseado em uma única IA. Neste trabalho, expandimos esta função para que ela: (i) forneça julgamento baseado no consenso de mais de uma IA, aumentando a confiabilidade do julgamento, mediante utilização de métricas encontradas em literatura correlata e, (ii) através de uma Interface de Programação de Aplicações (API) de serviços, se torne uma aplicação *web* independente e disponibilize julgamentos para qualquer solução de IA que necessite deste tipo de avaliação.

Também apresentamos o resultado de um estudo conduzido para comparar o julgamento realizado pelas IAs com um humano-especialista. O propósito da comparação é atuar como um balizador de acurácia das respostas providas por um agente computacional didático. A contribuição deste trabalho é a aplicação da técnica de *LLM-as-a-Judge* através de uma aplicação *web* de uso simplificado para julgamento automatizado de perguntas e respostas de outras IAs.

2. Trabalhos Correlatos

Buscamos compreender o estado da arte da técnica *LLM-as-a-Judge* como base para a construção do nosso fluxo de julgamento (Seção 4.2) e do software proposto. Para obtenção dos trabalhos citados abaixo como referência, foram utilizados buscadores da internet como Google Acadêmico[1], buscando por termos como "*LLM-as-a-judge*", "julgamento de IA", dentre outros.

Como estudo de caso, utilizou-se o TutorIA, agente conversacional para tutoria em Ciência da Computação que emprega Geração Aumentada por Recuperação (RAG) e um módulo simples de julgamento com uma única IA avaliadora (Figura 1). Neste trabalho, o TutorIA é tratado apenas como aplicação experimental para geração dos pares pergunta–resposta avaliados.

O artigo "A Survey on LLM-as-a-Judge"[Gu et al. 2025] define formalmente a técnica como um processo de avaliação contextual e destaca a meta-avaliação para medir a confiabilidade dos juízes LLM em relação a humanos, além de discutir métricas de alinhamento e vieses que afetam o julgamento.

[1] `https://scholar.google.com.br/`

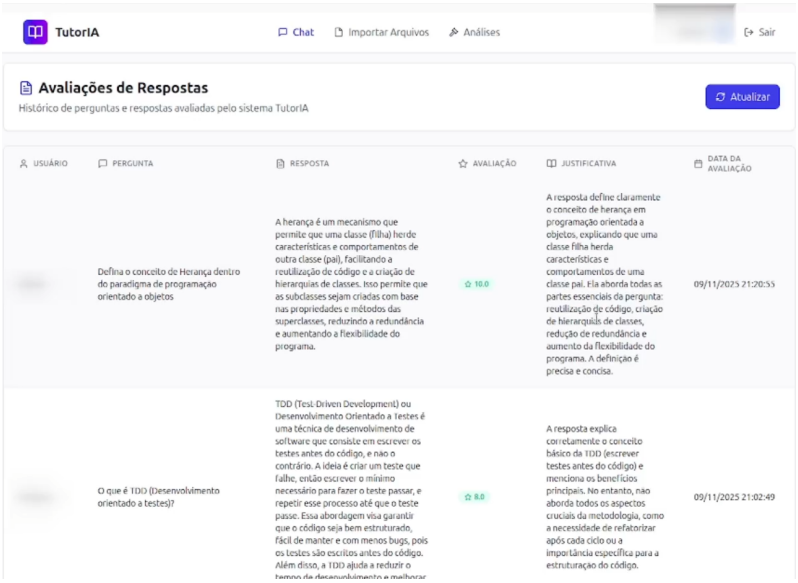

Figura 1. Tela do módulo de julgamento de respostas do TutorIA

Em [Zheng et al. 2023], demonstra-se que modelos como o GPT-4 podem atingir concordância com preferências humanas comparável ao consenso entre avaliadores humanos, além de identificar e mitigar vieses como o *position bias*, reforçando o uso de v*benchmarks* automatizados e avaliações pareadas.

O estudo [Li et al. 2024] sistematiza aplicações do *LLM-as-a-Judge*, incluindo avaliação de respostas e aprimoramento de modelos, destacando a Avaliação Colaborativa Humano-LLM e categorizando métricas de meta-avaliação para medir alinhamento com julgamentos humanos.

Já [Bencke et al. 2024] analisa a confiabilidade da técnica em português no domínio de Recuperação de Informação, avaliando não apenas correlação com humanos, mas também estabilidade. Embora haja variação em pontuações absolutas, o ranqueamento relativo apresentou correlação perfeita com julgamentos humanos.

Apesar dessas bases teóricas e empíricas, nosso trabalho diferencia-se ao priorizar a acessibilidade e a reprodutibilidade de um fluxo de julgamento (Seção 4.2).

3. Fundamentação Teórica e Prática

Encontrar um equilíbrio entre avaliações abrangentes e escaláveis é um desafio persistente. Até o surgimento das LLMs, avaliações conduzidas por humanos-especialistas era o padrão. No entanto, essas abordagens são custosas, difíceis de escalar e suscetíveis à inconsistência. Por outro lado, métodos de avaliação objetiva, como métricas automáticas, oferecem grande escalabilidade e consistência [Gu et al. 2025]. Neste contexto, surge a técnica *LLM-as-a-Judge*.

Além dos ganhos em eficiência, a técnica *LLM-as-a-Judge* proporciona melhorias significativas na qualidade e na interpretabilidade do processo de avaliação. Modelos avançados como o GPT-4[2] demonstraram uma correlação substancial com os julgamentos humanos, atingindo mais de 80% de concordância, o mesmo nível de concordância

[2]https://openai.com/pt-BR/index/gpt-4-research/

observado entre os próprios avaliadores humanos. Isso atesta sua capacidade de replicar a inferência humana [Dubois et al. 2024].

Essa eficiência permite que pesquisadores e organizações avaliem conjuntos de dados maiores com mais frequência e reduzam a necessidade de envolvimento humano [Zheng et al. 2023], o que pode, por exemplo, minimizar a exposição humana a conteúdo inapropriado (como material violento, sexual ou enviesado) [Chiang and yi Lee 2023].

3.1. Critérios e Métricas de Julgamento

LLM-as-a-Judge pressupõe uma avaliação multifacetada e alinhada à percepção humana, sendo essencial para tarefas de Geração de Linguagem Natural. O sucesso do *LLM-as-a-Judge* depende de que o modelo avalie os itens contra critérios de avaliação claros [Gu et al. 2025]. Os critérios de avaliação, Correção, Clareza, Didática, Completude, Relevância e Utilidade, se encaixam nas categorias amplas de avaliação de modelos LLM: Qualidade Linguística, Precisão de Conteúdo e Métricas Específicas de Tarefa e Utilidade [Li et al. 2024, Assis et al. 2025]. A adoção de múltiplos critérios é crucial, pois uma avaliação abrangente não pode se basear em uma única métrica, mas deve analisar dimensões separadas.

Os critérios de Relevância (*Relevance*), Correção (*Correctness*), e Completude (*Completeness*) representam a base da avaliação de conteúdo. A Correção (*Accuracy* ou *Factual Accuracy*), por exemplo, foca na exatidão e na ausência de informações incorretas. A Relevância avalia se a saída está relacionada à pergunta do usuário, enquanto a Completude (*Completeness* ou *Level of Detail*) garante que todos os aspectos da solicitação foram abordados satisfatoriamente. O critério de Clareza (*Clarity*), por sua vez, alinha-se à dimensão de Qualidade Linguística (*Linguistic Quality*), que engloba a fluência e a gramaticalidade do texto, aspectos fundamentais para a clareza e legibilidade da resposta [Li et al. 2024, Assis et al. 2025].

Adicionalmente, os critérios de Didática (*Pedagogical Value*) e Utilidade (*Helpfulness*) ajustam o avaliador para o domínio específico da aplicação. A Utilidade é uma dimensão central na avaliação de agentes conversacionais, avaliando o quão útil a resposta é para o usuário [Wang et al. 2023]. Já a Didática, que busca o uso de analogias e exemplos práticos, é uma métrica personalizada, frequentemente exigida no domínio da Educação, onde o foco da avaliação pode ser o valor inspirador e educacional das respostas [Li et al. 2024].

Ao decompor o julgamento em critérios discretos e explícitos, os LLMs podem avaliar cada aspecto de forma independente [Chiang and yi Lee 2023], o que permite traduzir julgamentos qualitativos em avaliações estruturadas, possibilitando o cálculo de métricas quantitativas de desempenho essenciais para a tomada de decisão em sistemas RAG [Albuquerque et al. 2024].

4. Desenvolvendo o Julgador

A Figura 2 ilustra a arquitetura de software do Julgador, evidenciando que se trata de uma aplicação *web* estruturada em dois módulos principais: um *front-end* executado no navegador do pesquisador e um *back-end* responsável pelo processamento das avaliações. Essa separação permite que a interface de uso permaneça leve e responsiva, enquanto as operações mais custosas, como orquestração das chamadas aos modelos de linguagem,

sejam tratadas no servidor. A existência de um *back-end* também garante um acesso direto por outros softwares através de rotas de um serviço *web* [Senduk et al. 2023].

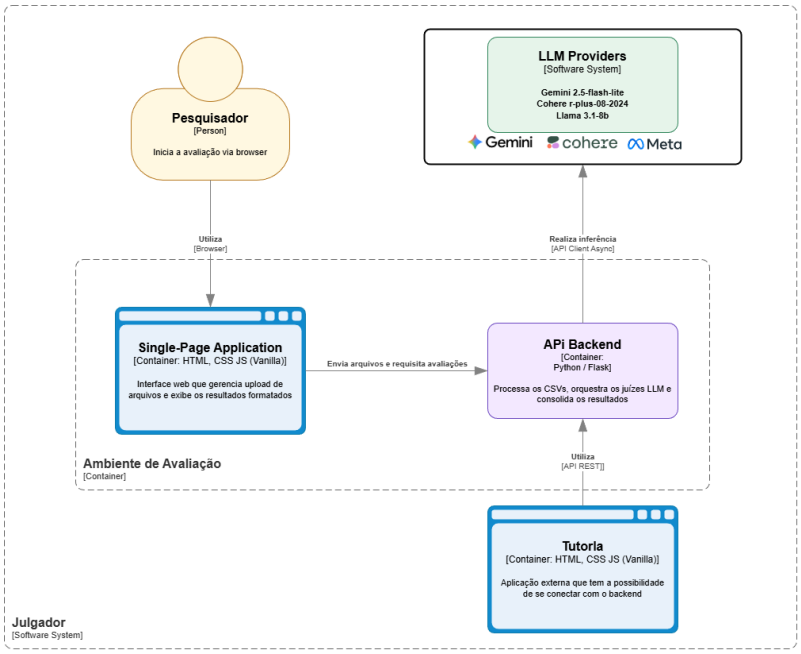

Figura 2. Arquitetura Geral do Julgador

Um Pesquisador pode iniciar o processo de avaliação diretamente via navegador. Esse usuário interage com o módulo de *front-end*, implementado como uma Single-Page Application (SPA) construída com HTML (*HyperText Markup Language*), CSS e JavaScript. A função principal desse módulo é gerenciar o *upload* dos arquivos CSV contendo os pares pergunta–resposta, além de exibir ao usuário os resultados consolidados gerados pelo sistema.

As requisições de julgamento de pares de pergunta-resposta são enviadas ao *API Backend*. Esse módulo é responsável por realizar todo o processamento central do Julgador: (i) leitura e validação dos arquivos CSV, (ii) preparação das requisições, (iii) envio assíncrono das entradas aos diferentes modelos de linguagem e (iv) consolidação dos resultados retornados por cada modelo. Dessa forma, o *back-end* funciona como um orquestrador dos "juízes LLM". Ou seja, a principal atividade do *back-end* é parametrizar via *prompt* os modelos de IA descritos na Tabela 1 para que atuem como julgadores.

Tabela 1. Modelos de IA Julgadores

Modelo	Versão
Gemini	gemini-2.5-flash-lite [3]
Cohere	command-r-plus-08-2024 [4]
Llamma	llama-3.1-8b-instant [5]

Tais modelos foram escolhidos levando em consideração alguns critérios principais, como acessibilidade via API gratuita, diversidade arquitetural e equilíbrio entre desempenho e custo computacional.

Para realizar as inferências, a *API Backend* se comunica com os *LLM Providers*, representados na arquitetura como um sistema externo de software. Esses provedores executam as avaliações automáticas conforme solicitado e retornam para o *back-end* os julgamentos produzidos. A interação entre esses componentes forma um fluxo:

1. o pesquisador ou o TutorIA submete os dados;
2. a SPA envia as requisições ao *back-end* ;
3. o *back-end* processa e coordena as inferências;
4. os provedores de LLM realizam o julgamento;
5. os resultados consolidados são devolvidos à interface para visualização.

Essa distribuição modular favorece escalabilidade, isolamento de responsabilidades e facilidade de manutenção do sistema [Parnas 1972, van Bekkum et al. 2021].

4.1. Prompt de contextualização

Posterior à escolha dos modelos, foi realizada a parametrização dos mesmos através de um *prompt* de contextualização. *Prompts* permitem contextualizar modelos de IA para que respondam de forma mais precisa e alinhada ao objetivo da aplicação. Ao incluir no *prompt* informações relevantes (papel esperado do modelo, restrições, exemplos e formato de saída) é possível direcionar o comportamento da IA sem retreiná-la [Son et al. 2025].

Todos modelos descritos na Tabela 1 foram configurados com o mesmo *prompt*, que se encontra replicado a seguir:

Listing 1. Prompt utilizado

```
Voce e um Juiz Imparcial de IA, um especialista em avaliar a qualidade de chatbots
    didaticos projetados para ensinar ciencia da computacao a estudantes. Sua tarefa e
    avaliar a qualidade da resposta de um chatbot para uma determinada pergunta.
**Contexto:** O chatbot avaliado tem como objetivo ser um tutor virtual. As respostas
    devem ser precisas, faceis de entender, pedagogicamente uteis e completas.
**Pergunta do Usuario:** {question}
**Resposta do Chatbot:** {answer}
**Criterios de Avaliacao:** Avalie a resposta do chatbot nos seguintes aspectos,
    atribuindo uma nota de 1 a 5 para cada um (1=Muito Ruim, 5=Excelente):
1. **Correcao (Correctness):** A informacao tecnica apresentada esta correta e livre de
    erros?
2. **Clareza (Clarity):** A linguagem e simples, direta e facil para um estudante
    entender? Evita jargoes desnecessarios?
3. **Didatica (Pedagogical Value):** A resposta utiliza analogias, exemplos praticos ou
    uma estrutura que facilite o aprendizado?
4. **Completude (Completeness):** A resposta aborda todos os pontos da pergunta do
    usuario de forma satisfatoria?
5. **Relevancia (Relevance):** A resposta e relevante para a pergunta do usuario?
6. **Utilidade (Helpfulness):** A resposta e genuinamente util para um estudante?
Forneca: * Nota para cada aspecto (1 a 5) * Justificativa geral * Pontos fortes * Pontos
     fracos * Nota final (1 a 5)
Evite vieses como o de verbosidade (nao favoreca respostas apenas por serem longas) e
    seja o mais objetivo possivel.
**Formato de Saida OBRIGATORIO:**
Sua avaliacao DEVE ser um objeto JSON valido, sem nenhum texto ou formatacao adicional
    antes ou depois dele. Use o seguinte formato:
(
'scores': ( 'correcao' (nota de 1 a 5), 'clareza' (nota de 1 a 5), 'didatica' (nota de 1
     a 5), 'completude' (nota de 1 a 5), 'relevancia' (nota de 1 a 5), 'utilidade' (nota
     de 1 a 5) ), 'justificativa': '(Sua analise detalhada explicando as notas
    atribuidas, com no maximo 100 palavras.)', 'veredito-final': '('Aprovado' ou '
    Reprovado', com base na qualidade geral da resposta.)', 'pontos-fortes': '(Sua
    analise detalhada explicando os pontos fortes da resposta, com no maximo 100
    palavras.)', 'pontos-fracos': '(Sua analise detalhada explicando os pontos fracos da
     resposta, com no maximo 100 palavras.)', 'score-final': (nota de 1 a 5)
)
```

4.2. Fluxo de Execução para Julgamento

Durante a realização de um ciclo de julgamento os passos ilustrados na Figura 3 são executados.

O processo se inicia com o envio de pares de perguntas e respostas a serem avaliados. Em seguida, o sistema valida os dados para garantir que cada entrada esteja no formato esperado. Após essa etapa, é iniciado o módulo de envio assíncrono das requisições aos modelos avaliadores, permitindo que múltiplos pares sejam processados simultaneamente. Nesta fase, os LLMs, llama, Cohere e Gemini, recebem os mesmos dados de entrada, possibilitando uma avaliação paralela e independente. Conforme as respostas são retornadas, ocorre a etapa de coleta dos resultados, na qual todas as avaliações são registradas. Com base nesses retornos, o sistema também calcula estatísticas descritivas (distribuição de notas, dispersão e eventuais divergências entre modelos), que servem para contextualizar a qualidade e a consistência das avaliações automáticas.

Após a coleta, o fluxo segue para a verificação de uma possível avaliação humana complementar. Caso o usuário opte por incluir julgamentos manuais, o sistema realiza o cálculo percentual da concordância entre avaliadores humanos e automáticos. Se não houver avaliação humana, essa etapa é ignorada. Em seguida, todas as avaliações disponíveis, automáticas e/ou humanas, passam por um processo de agregação e consolidação, no qual são combinadas de forma estruturada para produzir uma interpretação unificada dos resultados.

Com a consolidação concluída, o sistema gera um resumo analítico contendo os principais achados do processo, destacando convergências, divergências e evidências relevantes para a decisão final. Por fim, é produzido um veredito que representa a conclusão do ciclo de julgamento e é apresentado ao usuário como resultado final do processo.

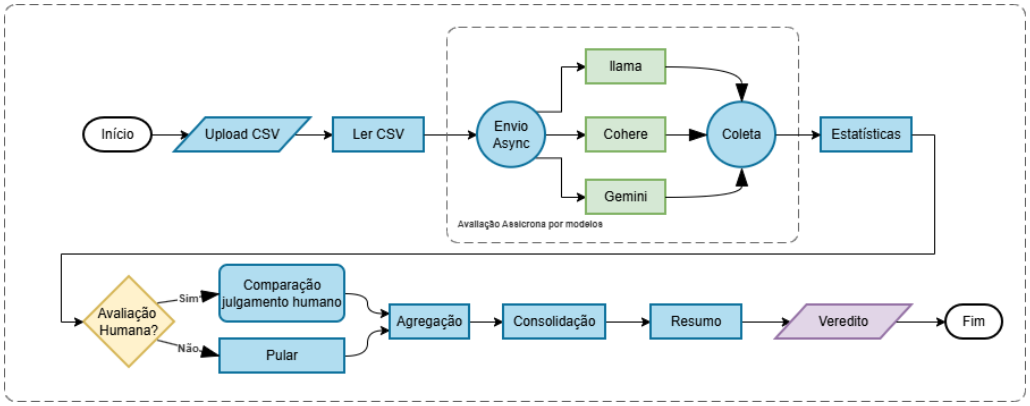

Figura 3. Fluxo de Julgamento

4.3. Exemplo de Resultado de Julgamento

Na Figura 4 é possível observar a tela inicial do Julgador de IA (ou Julgador, simplesmente), onde é possível inserir um arquivo CSV gerado a partir do TutorIA para avaliação das perguntas e respostas pelos juízes.

O arquivo CSV deve obedecer o formato exibido na Tabela 2. É possível ver exemplos de algumas perguntas, na coluna ”Pergunta”, e respostas, na coluna ”Resposta”,

Figura 4. Tela inicial do Julgador de IA

que foram utilizados como entrada para julgamento. Na coluna ”Avaliação Humana”, há um parecer de um agente humano, indicando se aprova ou reprova a resposta gerada pela IA que está sendo julgada.

A partir da entrada do CSV o Julgador aciona o fluxo de processamento exibido na Figura 3, exibindo, no final, o resultado da avaliação, o que se encontra exemplificado na Figura 5. É possível observar que a avaliação pelo juízes automatizados obteve 100% de aprovação.

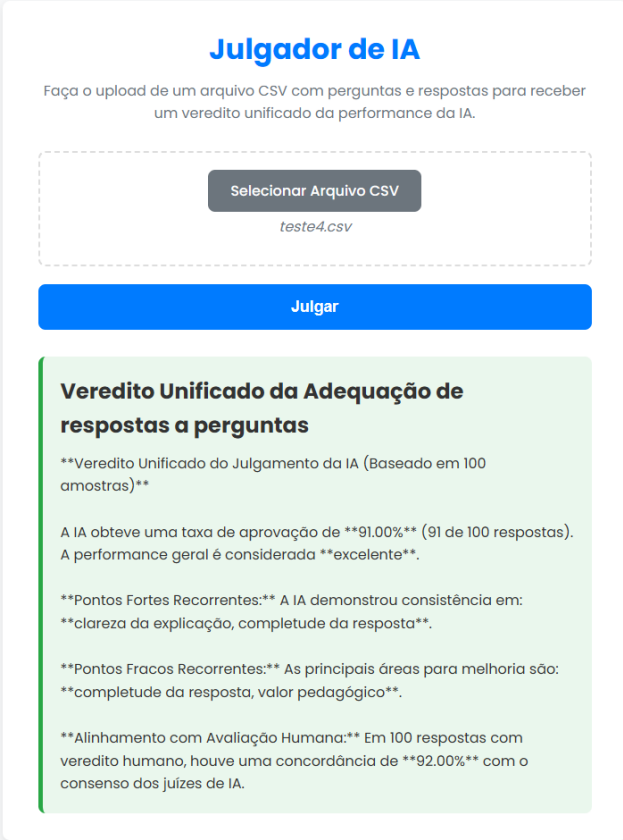

Figura 5. Resultado do julgamento de IA com avaliação humana

A comparação entre julgamento automatizado e avaliação humana foi realizada por meio de concordância percentual simples. Para cada par pergunta–resposta, considerou-se o veredito final do consenso entre os três modelos LLM e o veredito do

Tabela 2. Exemplo de entrada de perguntas e respostas para avaliação

Pergunta	Resposta	Avaliação Humana
Explique a diferença entre o paradigma de programação imperativo e o declarativo	O paradigma de programação imperativo se concentra na descrição das ações que o programa deve realizar, especificando os passos para alcançar um resultado. Já o paradigma de programação declarativo se concentra na descrição do que o programa deve produzir, sem preocupações com como isso será feito. Exemplo: em linguagens imperativas, você escreveria ”faça isso”e ”faça aquilo”, enquanto em linguagens declarativas, você escreveria ”o resultado deve ser isso”.	Aprovado
Explique por que a composição é, em muitos casos, preferível à herança no paradigma orientado a objetos.	A composição é preferível à herança quando se deseja criar uma relação entre classes onde uma classe tem comportamentos ou características adicionais, mas não necessariamente compartilha a mesma hierarquia de classes que a outra. Isso permite maior flexibilidade e reutilização de código, pois as classes podem ser compostas de outras classes sem estar vinculadas por uma relação de herança.	Aprovado
Como o processo de chunking de documentos influencia a qualidade da recuperação em um sistema RAG?	O processo de chunking de documentos divide os textos originais em pequenos trechos (chucks) para que possam ser processados e armazenados mais eficientemente. Isso pode influenciar a qualidade da recuperação em um sistema RAG, pois: * Pequenos chucks podem facilitar a recuperação de informações relevantes; * Grande número de chucks pode aumentar a precisão da recuperação. No entanto, se os chucks forem muito pequenos, pode haver perda de contexto e significado.	Aprovado

avaliador humano. A concordância foi calculada pela razão entre o número de decisões coincidentes e o total de casos avaliados. No conjunto analisado, houve concordância em 92 dos 100 casos, resultando em 92,0% de alinhamento.

A quantidade dos pares pergunta–resposta foi definida com base na disponibilidade de dados oriundos da fase atual de desenvolvimento do sistema utilizado como estudo de caso. Ressalta-se que o número não foi determinado com base em critérios estatísticos ou métricas prévias, configurando uma amostra exploratória. Como consequência, os resultados devem ser interpretados como evidência preliminar de alinhamento entre julgadores LLM e avaliador humano, não permitindo generalizações estatísticas amplas. Estudos futuros deverão ampliar significativamente o volume de dados avaliados. Disponibilizamos um vídeo demonstrativo da ferramenta através do link:

`https://zenodo.org/records/18738901`

5. Ameaças à Validade

A seguir, identificamos as ameaças à validade do estudo.

Uma ameaça à **Validade de Construção** reside no *prompt* utilizado para instruir os LLMs julgadores (Tabela 1), que pode não abranger todas as dimensões de qualidade das respostas, gerando avaliações parciais ou enviesadas. Além disso, possíveis vieses oriundos dos dados de treinamento dos modelos podem influenciar os julgamentos.

Quanto à **Validade Interna**, há risco associado à variabilidade das respostas dos LLMs entre execuções, mesmo com o mesmo *prompt*, fenômeno já documentado [Wang and Wang 2025]. Soma-se a isso a seleção de pares pergunta–resposta do TutorIA, ainda não avaliados por usuários finais, o que pode introduzir correlações artificiais.

A **Validade Externa** pode ser limitada pela possível falta de generalização dos resultados para outros modelos, diferentes tamanhos de contexto (*tokens*) ou outros domínios além do ensino de Ciência da Computação. Também não foram utilizados modelos previamente especializados na área.

Por fim, a **Validade de Conclusão** é ameaçada pelo fato de os julgamentos refletirem exclusivamente os critérios definidos no *prompt*, podendo divergir da percepção humana de qualidade. Assim, as inferências devem ser interpretadas com cautela, considerando as limitações dos modelos e dos critérios adotados.

6. Conclusão

Este trabalho apresentou o desenvolvimento e a validação de uma ferramenta *web* capaz de julgar automaticamente respostas de agentes conversacionais, aplicando a técnica *LLM-as-a-Judge*. Através de uma arquitetura modular que orquestra múltiplos modelos de linguagem (Gemini, Cohere e Llama) para avaliar critérios qualitativos, tais como, por exemplo, correção, didática e clareza, foi possível demonstrar que o sistema é capaz de fornecer julgamentos consistentes e escaláveis, com boa proximidade a um especialista humano, destacando o potencial da solução para apoiar processos de validação de respostas em sistemas educacionais baseados em IA.

Com base nas ameaças à validade identificadas, alguns caminhos podem ser explorados para aprimorar a robustez metodológica e ampliar o alcance dos achados. Entre as principais limitações, destacam-se: (i) tamanho reduzido da amostra; (ii) utilização de apenas um avaliador humano; (iii) ausência de múltiplas rodadas para medir estabilidade das respostas dos LLMs; e (iv) uso exclusivo de modelos gratuitos. Tais fatores podem influenciar a robustez dos resultados e limitam a generalização das conclusões.

Outra linha de investigação é a análise e detecção de vieses dos modelos, aplicando técnicas como *bias probing*, que é um método que pode ser utilizado para detectar e quantificar vieses em modelos de IA, especialmente em LLMs [Zhao et al. 2023]. Recomendamos também a execução de múltiplas rodadas de julgamento para o mesmo conjunto de pares, permitindo mensurar a estabilidade das respostas.

Compartilhamos o código-fonte da solução visando o reuso em trabalhos futuros ou a replicação dos resultados apresentados, por meio do link:

`https://zenodo.org/records/18738727`

Referências

Albuquerque, A., Wensing, I., Filho, N. J., and Dorneles, C. (2024). Avaliação de aplicações de geração aumentada de recuperação por meio de feedback implícito. In *Anais Estendidos do XXXIX Simpósio Brasileiro de Bancos de Dados*, pages 253–259, Porto Alegre, RS, Brasil. SBC.

Assis, G., Freitas, C., and Paes, A. (2025). Exploring brazil's llm fauna: Investigating the generative performance of large language models in portuguese. *Journal of the Brazilian Computer Society*, 31(1):940–972.

Bencke, L., Paula, F., dos Santos, B., and Moreira, V. P. (2024). Can we trust llms as relevance judges? In *Anais do XXXIX Simpósio Brasileiro de Bancos de Dados*, pages 600–612, Porto Alegre, RS, Brasil. SBC.

Chiang, C.-H. and yi Lee, H. (2023). Can large language models be an alternative to human evaluations?

Dubois, Y., Li, X., Taori, R., Zhang, T., Gulrajani, I., Ba, J., Guestrin, C., Liang, P., and Hashimoto, T. B. (2024). Alpacafarm: A simulation framework for methods that learn from human feedback.

Gu, J., Jiang, X., Shi, Z., Tan, H., Zhai, X., Xu, C., Li, W., Shen, Y., Ma, S., Liu, H., Wang, S., Zhang, K., Wang, Y., Gao, W., Ni, L., and Guo, J. (2025). A survey on llm-as-a-judge.

Li, H., Dong, Q., Chen, J., Su, H., Zhou, Y., Ai, Q., Ye, Z., and Liu, Y. (2024). Llms-as-judges: A comprehensive survey on llm-based evaluation methods.

Parnas, D. L. (1972). On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058.

Senduk, F., Najoan, X., and Sompie, S. (2023). Development of microservices architecture with restful api gateway using backend-for-frontend pattern in higher education academic portal. *J. Tek. Inform*, 18(1):315–324.

Son, M., Won, Y.-J., and Lee, S. (2025). Optimizing large language models: A deep dive into effective prompt engineering techniques. *Applied Sciences (2076-3417)*, 15(3).

van Bekkum, M., de Boer, M., van Harmelen, F., Meyer-Vitali, A., and Teije, A. t. (2021). Modular design patterns for hybrid learning and reasoning systems: a taxonomy, patterns and use cases. *Applied Intelligence*, 51(9):6528–6546.

Wang, J. J. and Wang, V. X. (2025). Assessing consistency and reproducibility in the outputs of large language models: Evidence across diverse finance and accounting tasks. *arXiv preprint*. arXiv:2503.16974.

Wang, P., Li, L., Chen, L., Cai, Z., Zhu, D., Lin, B., Cao, Y., Liu, Q., Liu, T., and Sui, Z. (2023). Large language models are not fair evaluators.

Zhao, J., Fang, M., Pan, S., Yin, W., and Pechenizkiy, M. (2023). Gptbias: A comprehensive framework for evaluating bias in large language models. *arXiv preprint arXiv:2312.06315*.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., and Stoica, I. (2023). Judging llm-as-a-judge with mt-bench and chatbot arena.

Minibiografia dos autores

Danilo Guimarães Souza Azevedo (IFBA): Graduado em Sistemas de Informação pelo Instituto Federal da Bahia (IFBA), Campus Vitória da Conquista-BA. Especialista em Desenvolvimento Web pelo IFBA. Atua profissionalmente como especialista em sistemas no mercado de varejo.

Melques Santos Paiva (IFBA): Desenvolvedor de Software com mais de 7 anos de experiência em desenvolvimento web, com foco em back-end utilizando PHP e Go. É graduado em Engenharia de Computação pela Faculdade Independente do Nordeste e atualmente cursa pós-graduação em Desenvolvimento Web pelo IFBA. Atualmente, é Senior Software Developer na DB1 Global Software, onde lidera o desenvolvimento e manutenção de integrações com gateways de pagamento de alta escala. Sua atuação envolve sistemas que processam milhões de transações diárias, além de melhorias de performance e monitoramento.

Ana Kessilly Chiachio Cerqueira (IFBA): Especialista em Desenvolvimento Web pelo Instituto Federal de Educação, Ciência e Tecnologia da Bahia (IFBA, 2025) e graduada em Ciência da Computação pelo Centro Universitário Estácio Ribeirão Preto (2023). Possui trajetória profissional de três anos no desenvolvimento de software, com ênfase em arquitetura de sistemas web. Sua atuação técnica concentra-se na implementação de soluções escaláveis, com foco em pesquisa e desenvolvimento voltados à integração de Inteligência Artificial em Softwares como Serviço (SaaS).

Crescêncio Lima (IFBA): Possui graduação em Ciência da Computação pela Universidade Estadual do Sudoeste da Bahia (2005), pós graduação em Engenharia de Software pela Faculdade Boa Viagem (2008), mestrado em Ciência da Computação pela Universidade Federal de Pernambuco (2011) e doutorado pelo programa de pós-graduação da Universidade Federal da Bahia (PGCOMP-UFBA). Atua como professor colaborador do Programa de Pós-Graduação em Engenharia de Sistemas e Produtos (PPGESP). Nos anos de 2019-2023 foi coordenador do Curso de Pós-graduação Lato Sensu em Desenvolvimento Web (PGDW) do IFBA campus Vitória da Conquista e atualmente é vice-coordenador do PGDW.

Djan Almeida Santos (IFBA): Professor do Instituto Federal da Bahia, campus Vitória da Conquista. Doutor em Ciência da Computação pela Universidade Federal da Bahia (2023), Mestre em Ciências Ambientais com Área de Concentração em Meio Ambiente e Desenvolvimento e linha de pesquisa em modelagem computacional pela Universidade Estadual do Sudoeste da Bahia (2013), especialista em Administração de Sistemas de Informação pela Universidade Federal de Lavras - MG, especialista em Desenvolvimento de Sistemas Web pela Faculdade de Tecnologia e Ciência (FTC) e Bacharel em Ciências da Computação pela Universidade Estadual de Santa Cruz (2003). Atuou na área da Ciência da Computação com ênfase em Engenharia de Software, compreensão de programas, sistemas configuráveis, modelagem matemática e simulação computacional.

Luis Paulo da Silva Carvalho (IFBA): Graduado em Informática pela Universidade Católica do Salvador (2002), Mestrado em Sistemas e Computação pela UNIFACS (2012) e Doutorado em Ciência da Computação pela UFBA (2020). Atualmente é professor do IFBA em Vitória da Conquista. Atuou por 16 anos na Indústria de Software como Programador, Analista, Arquiteto de Sistema e Líder Técnico. Tem experiência na área de

Ciência da Computação, com ênfase em Arquitetura de Sistemas de Computação, atuando principalmente nos temas: dispositivos móveis, telecomunicação, automação, sistemas embarcados, sistemas distribuídos, inteligência artificial, bancos de dados, sistemas web, visualização de software, arquitetura orientada a serviços, padrões de projeto, análise e modelagem de sistemas, IHM, Inteligência Artificial e simulação computacional.