Revista Brasileira de Informática na Educação – RBIE
Brazilian Journal of Computers in Education
(ISSN online: 2317-6121; print: 1414-5685)
https://sol.sbc.org.br/journals/index.php/rbie

Submission: dd/Mmm/yyyy; 1st round notif.: dd/Mmm/yyyy; New version: dd/Mmm/yyyy; 2nd round notif.: dd/Mmm/yyyy;
Camera ready: dd/Mmm/yyyy; Edition review: dd/Mmm/yyyy; Available online: dd/Mmm/yyyy; Published: dd/Mmm/yyyy;

Avaliando a habilidade do ChatGPT de realizar provas de Dedução Natural em Lógica Proposicional e Lógica de Predicados

Title: Evaluating ChatGPT's ability to perform Natural Deduction proofs in Propositiona and Predicate Logic

Francisco Leonardo Batista Martins
Universidade Federal do Ceará - Campus Quixadá
ORCID: 0009-0009-9174-6601
lmartins@alu.ufc.br
Augusto César Araújo de Oliveira
Universidade Federal do Ceará - Campus Quixadá
ORCID: 0009-0005-3865-6210
augustces@alu.ufc.br
Davi Romero de Vasconcelos
Universidade Federal do Ceará - Campus Quixadá
ORCID: 0000-0002-4857-4156
daviromero@ufc.br
Maria Viviane de Menezes
Universidade Federal do Ceará - Campus Quixadá
ORCID: 0000-0002-7096-4732
vivianemenezes@ufc.br

Resumo

A crescente utilização de agentes conversacionais, comumente conhecidos como chatbots, na educação tem despertado um interesse cada vez maior entre pesquisadores, educadores e instituições de ensino em todo o mundo. Além disso, com a recente popularização dos Grandes Modelos de Linguagem, como o ChatGPT, muitos estudos tem sido conduzidos no sentido de explorar a utilização destas ferramentas para auxiliar no processo de ensino e aprendizagem. Esses sistemas têm a capacidade de compreender e processar enormes quantidades de dados, o que lhes permite, por exemplo, fornecer suporte individualizado aos alunos na resolução de exercícios. Entretanto, é fundamental considerar que esses sistemas geralmente possuem um processo de raciocínio baseado em métodos estatísticos, como algoritmos de aprendizado de máquina, que podem alucinar e resultar em respostas incorretas em tarefas que exigem raciocínio lógico, por exemplo. Este artigo tem como objetivo avaliar a capacidade do agente conversacional ChatGPT na resolução de exercícios de Dedução Natural em Lógica Proposicional e Lógica de Predicados. Para tanto, foram conduzidos experimentos utilizando uma base de dados de exercícios de dedução natural e os resultados confirmam que o processo de raciocínio da ferramenta ainda não se mostra adequado para solucionar este tipo de exercício, tornando essa ferramenta, por ora, inadequada para essa tarefa.

Palavras-chave: *Dedução Natural. Lógica Proposicional. Lògica de Predicados. Raciocínio Lógico. Modelos Grandes de Linguagem. ChatGPT.*

Abstract

The growing use of conversational agents, commonly known as chatbots, in education has sparked increasing interest among researchers, educators, and educational institutions worldwide. Additionally, with the recent popularization of Large Language Models, like ChatGPT, many studies have been conducted to explore the use of these tools to assist in the teaching and learning process. These systems have the ability to understand and process vast amounts of data, allowing them, for example, to provide individualized support to students in solving exercises. However, it is important to consider that these systems mostly have a reasoning process based on statistical methods, such as machine learning algorithms, which can hallucinate resulting in incorrect responses in tasks like logical reasoning. This article aims to evaluate the ability of the conversational agent ChatGPT in solving exercises of Natural Deduction in Propositional and Predicate Logic. In order to do so, we conduct experiments using a database of natural deduction exercises and the results confirm that the statistical reasoning process is still not adequate for perform logical reasoning, making ChatGPT, for now, unsuitable for this task.

Cite as: Martins, F. L. B. & Oliveira, A. C. A. & Vasconcelos, D. R. & Menezes, M. V. (2024). Avaliando a habilidade do ChatGPT de realizar provas de Dedução Natural em Lógica Proposicional e Lógica de Predicados. Revista Brasileira de Informática na Educação, vol, pp-pp. https://doi.org/10.5753/rbie.yyyy.id.

1 Introdução

A utilização de *agentes conversacionais*, também conhecidos como *chatbots*, na educação tem despertado um crescente interesse de pesquisadores, educadores e instituições de ensino em todo o mundo Kasneci et al., 2023; Tlili et al., 2023; Weber et al., 2021. A capacidade desses sistemas em compreender e processar grandes volumes de dados, além de sua habilidade em aprender e adaptar-se a novas informações, oferece oportunidades promissoras para auxiliar no processo de ensino-aprendizagem. Um dos principais benefícios de se utilizar tais ferramentas é a possibilidade de oferecer suporte individualizado aos alunos. Entretanto, é fundamental considerar que esses sistemas geralmente possuem um processo de raciocínio baseado em métodos estatísticos, como algoritmos de aprendizado de máquina, o que pode resultar em *respostas incorretas* em em determinadas situações, especialmente quando lidam com tarefas envolvendo *raciocínio lógico* H. Liu et al., 2023.

A Lógica para Computação é uma disciplina abordada em grande parte dos cursos de graduação na área de Tecnologia da Informação e Comunicação (TICs). O objetivo da lógica na computação é desenvolver linguagens para modelar as situações do mundo e dos sistemas, de modo que possamos analisá-las formalmente, construindo argumentos sobre elas para serem apresentados e *justificados rigorosamente*. A Dedução Natural é um dos conteúdos mais importantes na ementa desta disciplina, sendo utilizada para derivar *conclusões* a partir de sentenças dadas como verdadeiras (as quais são denominadas *premissas*), seguindo regras específicas Huth e Ryan, 2004; Pelletier, 1999.

Neste contexto, este artigo tem como objetivo avaliar a habilidade do agente conversacional ChatGPT, um grande modelo de linguagem (do inglês *Large Language Model* - LLM) desenvolvido pela OpenAI OpenAI, 2021, na resolução de exercícios de Dedução Natural em Lógica Proposicional e Lógica de Predicados. A principal pergunta que pretendemos responder com esse estudo é: "*o ChatGPT é capaz de gerar respostas corretas na tarefa solucionar exercícios de dedução natural em Lógica Proposicional e Lógica de Predicados?*". Para isso, avaliamos o desempenho da ferramenta em um base de dados de exercícios de dedução natural em Lógica Proposicional e Lógica de Predicados, disponibilizadas para alunos de graduação da disciplina de Lógica para Computação ofertada aos alunos da Universidade Federal do Ceará - Campus Quixadá. As interações com o modelo foram realizadas utilizando uma API (do inglês *Application Programming Interface*) disponibilizada pela *OpenAI*, a qual foi acessada por meio de sua biblioteca na linguagem de programação *Python*. Por fim, o desempenho do modelo na resolução dos exercícios foi avaliado verificando se as respostas produzidas estavam corretas. No caso de apresentação de respostas incorretas pelo modelo, avaliamos se os erros eram erros de aplicação de regras lógicas ou apenas erros de escrita da prova.

O restante do artigo está organizado como mostrado a seguir. Na Seção 2, apresentaremos a fundamentação teórica do trabalho, consistindo nos conceitos de Lógica Proposicional, Lógica de Predicados e Redes Neurais Transformacionais. Na Seção 3, apontamos alguns trabalhos relacionados sobre o uso do ChatGPT em tarefas de raciocínio lógico. Na Seção 4, descrevemos a metodologia utilizada para avaliar a ferramenta. Na Seção 5, apresentamos os resultados obtidos. E, na Seção 6 apresentaremos as conclusões e trabalhos futuros.

2 Fundamentação

2.1 Lógica Proposicional

A Lógica Proposicional baseia-se em *proposições* ou *frases declarativas* que se pode argumentar sobre sua veracidade ou falsidade. As frases declarativas são consideradas como *atômicas* (e.g, "*o dólar sobe*", "*os produtos ficam mais caros*") de certa forma que podemos atribuir símbolos distintos $P, Q, R, \cdots$ a cada uma destas frases (e.g, P : "*o dólar sobe*", Q : "*os produtos ficam mais caros*"). Para codificar frases mais complexas usamos os conectivos lógicos: $\neg$ (negação), $\wedge$ (conjunção), $\vee$ (disjunção) e $\rightarrow$ (implicação) Enderton, 1972. Por exemplo: $P \wedge Q$ codifica a frase "O dólar sobe *e* os produtos ficam mais caros"; $P \vee Q$ codifica a frase "O dólar sobe *ou* os produtos ficam mais caros."; $P \rightarrow Q$ representa "*se* O dólar sobe *então* os produtos ficam mais caros." e; $\neg P$: "O dólar não subiu".

O conjunto dos átomos proposicionais, juntamente com os conectivos e os símbolos '(', ')' formam o alfabeto da linguagem da Lógica Proposicional. A sintaxe da Lógica Proposicional pode ser definida por uma gramática na forma de *Backus Naur* (BNF - *Backus Naur Form*) como a seguir Huth e Ryan, 2004:

$$\varphi ::= \bot \mid P \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi)$$

em que $\bot$ e P representam, respectivamente, a contradição e qualquer proposição atômica e cada ocorrência φ a direita de '::=' representa qualquer fórmula já construída.

A Dedução Natural é um sistema dedutivo que permite a construção de uma prova formal, estabelecendo uma conclusão φ a partir de um conjunto de premissas $\Gamma = \{\varphi_1, \varphi_2, \cdots, \varphi_n\}$, denotado por $\Gamma \vdash \varphi$, aplicando-se sucessivamente *regras de demonstração*. O trabalho de Pelletier, 2000 compara o estilo de demonstrações de Dedução Natural de 33 livros-texto, sendo o estilo *Fitch* (caixas) o adotado na maioria deles. Neste estilo, as demonstrações são apresentadas de forma linear e sequencial, na qual cada uma das linhas da prova é numerada, tem uma afirmação e uma justificativa. As justificativas são definidas por serem *premissas* da prova ou pela aplicação de uma das *regras do sistema dedutivo*. As subprovas dentro de uma prova maior têm *caixas* ao redor que servem para delimitar o escopo de hipóteses temporárias. Provas podem ter caixas dentro de caixas, ou pode-se abrir outras caixas depois de fechar outras, obedecendo as regras de demonstração. Uma fórmula só pode ser utilizada em uma prova em um determinado ponto se essa fórmula aconteceu anteriormente e se nenhuma caixa que contenha essa ocorrência da fórmula tenha sido fechada.

Uma demonstração em Dedução Natural é construída a partir da enumeração das premissas e da aplicação das *regras de Dedução Natural*. Em geral, cada conectivo possui uma regra para adicionar e outra regra para eliminar este conectivo. A construção de uma demonstração é um exercício criativo: **não é óbvio quais regras aplicar e em qual ordem, a partir das premissas, para obter a conclusão desejada**. A seguir, serão apresentadas as regras básicas de Dedução Natural em Lógica Proposicional, a saber: $\wedge$-introdução ($\wedge i$), $\wedge$-eliminação ($\wedge e$), $\rightarrow$-introdução ($\rightarrow i$), $\rightarrow$-eliminação ($\rightarrow e$), $\vee$-introdução ($\vee i$), $\vee$-eliminação ($\vee e$), $\bot$-eliminação ($\bot e$), $\neg$-introdução ($\neg i$), $\neg$-eliminação ($\neg e$), raa (redução ao absurdo) e copie.

Premissas O primeiro passo em uma demonstração em Dedução Natural no estilo *Fitch* é enumerar as premissas da prova. A Figura 1 apresenta a estrutura geral da regra das premissas, na

qual $\varphi_1, \varphi_2, \ldots, \varphi_n$ são representadas em uma linha cada, seguindo uma numeração sequencial e como justificativa "premissa".

1.	φ_1	premissa
2.	φ_2	premissa
⋮	⋮	⋮
n.	φ_n	premissa
⋮	⋮	⋮

Figura 1: Enumeração das premissas.

Conjunção Introdução ($\wedge i$) A regra da introdução da conjunção ($\wedge i$) é apresentada na Figura 2 (ou Figura 3), na qual a fórmula $\varphi \wedge \psi$ pode ser concluída em uma linha p se φ e ψ foram demonstradas nas linhas m (ou n) e n (ou m), respectivamente, anteriores a linha p e que não foram descartadas. A Figura 4 exibe a aplicação $\wedge i$ da fórmula $A \wedge B$ na linha 3 a partir das fórmulas A e B, definidas nas linhas 1 e 2, respectivamente, que são anteriores a linha 3.

⋮	⋮	⋮
$m.$	φ	
⋮	⋮	⋮
$n.$	ψ	
⋮	⋮	⋮
$p.$	$\varphi \wedge \psi$	$\wedge$i m,n

Figura 2: Regra $\wedge i$

⋮	⋮	⋮
$m.$	ψ	
⋮	⋮	⋮
$n.$	φ	
⋮	⋮	⋮
$p.$	$\varphi \wedge \psi$	$\wedge$i m,n

Figura 3: Regra $\wedge i$

1.	A	premissa
2.	B	premissa
3.	$A \wedge B$	$\wedge i$ 1,2

Figura 4: $A, B \vdash A \wedge B$

Figura 5: Conjunção Introdução ($\wedge i$)

Conjunção Eliminação ($\wedge e$) A regra da eliminação da conjunção ($\wedge e$) é apresentada na Figura 6 (ou Figura 7), na qual a fórmula φ (ou ψ) pode ser concluída na linha p a partir da eliminação à esquerda (ou à direita) da conjunção da fórmula $\varphi \wedge \psi$ da linha n (anterior a m e não foi descartada). A Figura 8 exibe uma aplicação da regra na qual A é obtida na linha 3 pela eliminação da conjunção à esquerda da fórmula $A \wedge B$ da linha 1.

⋮	⋮	⋮
$m.$	$\varphi \wedge \psi$	
⋮	⋮	⋮
$p.$	φ	$\wedge$e m

Figura 6: Regra $\wedge e$ à esquerda

⋮	⋮	⋮
$m.$	$\varphi \wedge \psi$	
⋮	⋮	⋮
$p.$	ψ	$\wedge$e m

Figura 7: Regra $\wedge e$ à direita

1.	$A \wedge B$	premissa
2.	C	premissa
3.	A	$\wedge e$ 1
4.	$A \wedge C$	$\wedge i$ 3,2

Figura 8: $A \wedge B, C \vdash A \wedge C$

Figura 9: Conjunção Eliminação ($\wedge e$)

Implicação Eliminação (→e) A regra da eliminação da implicação (→e), também conhecida como *Modus Ponens*, é apresentada na Figura 10 (ou Figura 11), na qual a fórmula ψ pode ser concluída na linha p a partir da eliminação da implicação da fórmula $\varphi \rightarrow \psi$ da linha m (ou n) e φ da linha n (ou m), anteriores a p e não descartadas. A Figura 15 exibe uma aplicação da regra na qual a fórmula C é obtida na linha 4 pela eliminação da implicação das fórmulas $B \rightarrow C$ e B das linha 2 e 3, respectivamente.

⋮	⋮	⋮
$m.$	$\varphi \rightarrow \psi$	
⋮	⋮	⋮
$n.$	φ	
⋮	⋮	⋮
$p.$	ψ	→e m,n

Figura 10: Regra $\rightarrow e$

⋮	⋮	⋮
$m.$	φ	
⋮	⋮	⋮
$n.$	$\varphi \rightarrow \psi$	
⋮	⋮	⋮
$p.$	ψ	→e m,n

Figura 11: Regra $\rightarrow e$

1.	$A \wedge B$	premissa
2.	$B \rightarrow C$	premissa
3.	B	$\wedge e$ 1
4.	C	$\rightarrow e$ 2,3

Figura 12: $A \wedge B, B \rightarrow C \vdash C$

Figura 13: Implicação Eliminação ($\rightarrow e$)

Implicação Introdução (→i) A regra da introdução da implicação (→i) constrói condicionais que não ocorrem como premissas. Para construção de um condicional é necessário realizar *raciocínio hipotético*, isto é, supor *temporariamente* que uma dada fórmula é verdadeira. Chamamos esta fórmula de *hipótese*. Assim, utilizamos *caixas de demonstração*, que servem para delimitar o *escopo da hipótese temporária*. Observe na Figura 14 que para provar o condicional $\varphi \rightarrow \psi$ na linha $n+1$, colocamos φ como *hipótese* no topo de uma caixa (linha m), aplicamos um número finito de regras de forma a obter ψ na linha n. Todo o raciocínio para obter ψ depende da veracidade de φ e, por isso, as fórmulas resultante deste raciocínio ficam delimitadas na caixa. Na linha seguinte $(n+1)$ podemos aplicar a regra $\rightarrow i$ para obter $\varphi \rightarrow \psi$, sendo que este condicional não mais depende da hipótese φ. Na justificativa da linha $n+1$ utilizamos o nome da regra seguido das linhas inicial e final da caixa ($\rightarrow i$ *m-n*). A Figura 15 exibe uma aplicação da regra na qual a fórmula $A \rightarrow C$ é obtida na linha 6 a partir da caixa das linhas 3 a 5 em que A é a hipótese e C é a conclusão da caixa.

⋮	⋮	⋮
$m.$	φ	hipótese
⋮	⋮	⋮
$n.$	ψ	
$n+1.$	$\varphi \rightarrow \psi$	→i *m-n*

Figura 14: Regra $\rightarrow i$

1.	$A \rightarrow B$	premissa
2.	$B \rightarrow C$	premissa
3.	A	hipótese
4.	B	$\rightarrow e$ 3,1
5.	C	$\rightarrow e$ 4,2
6.	$A \rightarrow C$	$\rightarrow i$ 3-5

Figura 15: $A \rightarrow B, B \rightarrow C \vdash A \rightarrow C$

Figura 16: Implicação Introdução ($\rightarrow i$)

Demonstrações podem ter caixas dentro de caixas, ou pode-se abrir novas caixas depois de fechar outras. No entanto, existem regras sobre quais fórmulas podem ser utilizadas em que ponto na demonstração. Em geral, só podemos usar uma fórmula em um determinado ponto se esta fórmula ocorre *antes* desse ponto e se nenhuma caixa que contenha a ocorrência desta fórmula tenha sido fechada Huth e Ryan, 2004.

Disjunção Introdução A regra da introdução da disjunção ($\vee i$) é a apresentada na Figura 17 (ou Figura 18), na qual dizemos que temos $\varphi \vee \psi$ em uma linha p se φ (ou ψ) ocorre em uma linha m anterior a p e que não foi descartada. A Figura 19 ilustra a aplicação da introdução da disjunção na linha 3 com a introdução de $A \vee B$ a partir da fórmula A definida na linha 2.

⋮	⋮	⋮
m.	φ	
⋮	⋮	⋮
p.	$\varphi \vee \psi$	$\vee$i m

Figura 17: Regra $\vee i$ à esquerda

⋮	⋮	⋮
m.	ψ	
⋮	⋮	⋮
p.	$\varphi \vee \psi$	$\vee$i m

Figura 18: Regra $\vee i$ à direita

1.	$(A \vee B) \to C$	premissa
2.	A	hipótese
3.	$A \vee B$	$\vee$ i 2
4.	C	$\to$ e 3,1
5.	$A \to C$	$\to$ i 2-4

Figura 19: $(A \vee B) \to C \vdash A \to C$

Figura 20: Disjunção Introdução ($\vee i$)

Disjunção Eliminação ($\vee e$) a regra da disjunção eliminação ($\vee e$) é a apresentada na Figura 21, na qual dizemos que podemos concluir uma fórmula α na linha $p+1$ se eliminarmos a disjunção da fórmula $\varphi \vee \psi$ na linha m e se fizermos a suposição de φ em uma caixa, na linha m, e a suposição de ψ em outra caixa, na linha $n+1$, tal que ambas as caixas tenham como conclusão α, nas linhas n e p, respectivamente, por meio de uma sequência finita de passos (regras). Note que essa regra se assemelha a introdução da implicação no sentido de que fazemos raciocínio hipotético, nas caixas de $(m+1)-n$ e $(n+1)-p$. A Figura 22 ilustra a aplicação da eliminação da disjunção na linha 8, na qual concluímos C, a partir da disjunção de $A \vee B$ na linha 3 e das caixas $4-5$ e $6-7$ onde supomos A na linha 4 e concluímos C na linha 5 e supomos B na linha 6 e concluímos C na linha 7.

Negação Eliminação ($\neg$e) A regra da negação eliminação ($\neg$e) envolve a noção de *contradição*. Contradições são expressões da forma $\varphi \wedge \neg\varphi$ ou $\neg\varphi \wedge \varphi$ onde φ é qualquer fórmula da Lógica Proposicional. A fórmula $\perp$ é utilizada para denotar uma contradição e este fato é expresso na regra $\neg e$. Na Figura 24 (ou Figura 25) temos a fórmula φ na linha m (ou n) e a sua negação $\neg\varphi$ na linha n (ou m) sendo combinadas para aparecimento da contradição $\perp$ na linha p com a aplicação da regra $\neg e$. A Figura 26 exibe uma aplicação da regra na qual a contradição $\perp$ é obtida na linha 3 devido às fórmulas A na linha 1 e $\neg A$ na linha 2.

Negação Introdução ($\neg i$) A regra da negação introdução ($\neg i$) é a apresentada na Figura 28. Esta é uma regra que envolve raciocínio hipotético e contradição. Se tomarmos φ como hipótese (linha

⋮	⋮	⋮
m.	$\varphi \vee \psi$	
m+1.	φ	hipótese
⋮	⋮	⋮
n.	α	
n+1.	ψ	hipótese
⋮	⋮	⋮
p.	α	
p+1.	α	$\vee$e m, (m+1)-n, (n+1)-p

Figura 21: Regra $\vee e$

1.	$A \to C$	premissa
2.	$B \to C$	premissa
3.	$A \vee B$	premissa
4.	A	hipótese
5.	C	$\to$ e 4,1
6.	B	hipótese
7.	C	$\to$ e 6,2
8.	C	$\vee$ e 3, 4-5, 6-7

Figura 22: $A \to C, B \to C, A \vee B \vdash C$

Figura 23: Disjunção Eliminação ($\vee e$)

⋮	⋮	⋮
m.	φ	
⋮	⋮	⋮
n.	$\neg\varphi$	
⋮	⋮	⋮
p.	$\bot$	$\neg$e *m,n*

Figura 24: Regra $\neg e$

⋮	⋮	⋮
m.	$\neg\varphi$	
⋮	⋮	⋮
n.	φ	
⋮	⋮	⋮
p.	$\bot$	$\neg$e *m,n*

Figura 25: Regra $\neg e$

1.	A	premissa
2.	$\neg A$	premissa
3.	$\bot$	$\neg e$ 1,2

Figura 26: $A, \neg A \vdash \bot$

Figura 27: Negação Eliminação ($\neg e$)

m) e, após a aplicação de um número finito de regras, chegarmos a uma contradição $\bot$ na linha *n*, significa que a hipótese não pode ser verdadeira. Desse modo, finalizamos nosso raciocínio hipotético introduzindo a negação na hipótese e obtendo $\neg\varphi$ na linha $n+1$. A Figura 29 mostra um exemplo da aplicação da regra $\neg i$ em para provamos $\neg A$ na linha 6, assumimos A como hipótese no topo da caixa na linha 3 e chegamos a uma contradição no final da caixa na linha 5.

⋮	⋮	⋮
m.	φ	hipótese
⋮	⋮	⋮
n.	$\bot$	
$n+1$.	$\neg\varphi$	$\neg$i m-n

Figura 28: Regra $\neg i$

1.	$A \to B$	premissa
2.	$\neg B$	premissa
3.	A	hipótese
4.	B	$\to e$ 1,3
5.	$\bot$	$\neg e$ 2,4
6.	$\neg A$	$\neg i$ 3-5

Figura 29: $A \to B, \neg B \vdash \neg A$

Figura 30: Negação Introdução ($\neg i$)

Contradição Eliminação ($\perp e$): A regra da contradição eliminação ($\perp e$) é exibida na Figura 31, na qual podemos concluir uma fórmula qualquer ψ na linha n se demonstramos em uma linha m, anterior a n, a contradição. A Figura 32 apresenta a demonstração de B na linha 4 a partir da eliminação da contradição $\perp$ da linha 3.

⋮	⋮	⋮
$m.$	$\perp$	
⋮	⋮	⋮
$n.$	ψ	$\perp$e m

Figura 31: Regra $\perp e$

1.	A	hipótese
2.	$\neg A$	hipótese
3.	$\perp$	$\neg$ e 1,2
4.	B	$\perp$ e 3
5.	$\neg A \rightarrow B$	$\rightarrow$ i 2-4
6.	A$\rightarrow (\neg A \rightarrow B)$	$\rightarrow$ i 1-5

Figura 32: $\vdash A \rightarrow (\neg A \rightarrow B)$

Figura 33: Regra da Eliminação por Contradição ($\perp e$)

Redução ao Absurdo (raa): A regra de redução ao absurdo é apresenta na Figura 34, na qual para provarmos uma fórmula φ em uma linha $n+1$, iremos supor temporariamente a negação da fórmula, $\neg\varphi$, em uma caixa que inicia na linha m e que conclui a contradição, $\perp$, na linha n, após uma sequência de aplicações de regras. A Figura 35 exibe a demonstração de $A \vee \neg A$, também conhecido como terceiro-excluído. Para provarmos $A \vee \neg A$ na linha 8, fazemos a suposição de $\neg(A \vee \neg A)$, na linha 1 (início da caixa) e concluímos a contradição $\perp$, na linha 7 (fim da caixa).

⋮	⋮	⋮
m.	$\neg\varphi$	hipótese
⋮	⋮	⋮
n.	$\perp$	
n+1.	φ	raa m-n

Figura 34: Regra raa

1.	$\neg(A \vee \neg A)$	hipótese
2.	$\neg A$	hipótese
3.	A$\vee\neg A$	$\vee$ i 2
4.	$\perp$	$\neg$ e 3,1
5.	A	raa 2-4
6.	A$\vee\neg A$	$\vee$ i 5
7.	$\perp$	$\neg$ e 6,1
8.	A$\vee\neg A$	raa 1-7

Figura 35: $\vdash A \vee \neg A$

Figura 36: Redução ao Absurdo (raa)

Copie: A regra copie, apresentada na Figura 37, é necessária, no estilo de Fitch, para permitir concluir uma caixa com uma fórmula que já apareceu anteriormente na demonstração. Por exemplo, para demonstrar $A \rightarrow B$, veja Figura 38, na linha 10, é preciso que a caixa que justifica a

introdução da implicação inicie com A, linha 8, e termine com B, linha 9. Ocorre que a justificativa de B já havia sido realizada e, portanto, a justificativa da linha 9 é a cópia da fórmula da linha 7.

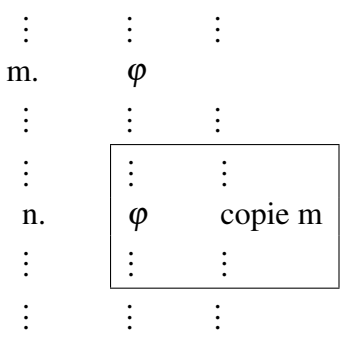


Figura 37: Regra copie

1.	$\neg A \vee B$	premissa
2.	$\neg A$	hipótese
3.	A	hipótese
4.	$\perp$	$\neg$ e 2,3
5.	B	$\perp$ e 4
6.	A$\rightarrow B$	$\rightarrow$ i 3-5
7.	B	hipótese
8.	A	hipótese
9.	B	copie 7
10.	A$\rightarrow B$	$\rightarrow$ i 8-9
11.	A$\rightarrow B$	$\vee$ e 1, 2-6, 7-10

Figura 38: $\neg A \vee B \vdash A \rightarrow B$

Figura 39: Regra Copie

2.2 Lógica de Predicados

A Lógica de Predicados surgiu da necessidade de representar aspectos mais ricos da linguagem natural do que apenas as frases declarativas da Lógica Proposicional. Esta lógica foi desenvolvida de forma independente por Gottlob Frege e Charles Sanders Peirce Ferreirós, 2001; Hammer, 1998. As fórmulas da lógica de predicados são compostas por predicados, constantes, variáveis, símbolos funcionais, quantificadores e conectivos.

Um *predicado* é um símbolo que representa uma propriedade ou uma relação entre objetos do mundo. Por exemplo, na fórmula $Estudante(maria)$ o predicado *Estudante* expressa a propriedade de que um símbolo constante *maria* é uma estudante. Já na fórmula *Professor(joão, maria)* o predicado *Professor* relaciona as constantes *joão* e *maria*, representando que *joão* é professor de *maria*. Essas relações também podem ser estabelecidas para variáveis de um certo domínio. Além disso, a Lógica de Predicados inclui *quantificadores* com os quais é possível representar a noção de "*existe*" (*existe um x que é estudante*) e "*para todo*" (*todos os x são estudantes*). Também é possível representar *símbolos funcionais* que são associados a objetos do mundo. Por exemplo, *professor(maria)*[1] é uma função que devolve a constante *joão*.

Devido ao seu maior poder de expressividade, a linguagem da Lógica de Predicados é mais complexa do que a da Lógica Proposicional. Nesta lógica há dois tipos de expressões: os termos e as fórmulas. Termos são expressões que relacionam-se a objetos de um domínio: indivíduos como

[1]Representamos funções com letras minúsculas para diferenciar dos predicados que são representados com letras maiúsculas.

joão e *maria* são exemplos; variáveis como x e; símbolos funcionais que também referem-se a objetos, como exemplo *professor(maria)* refere-se ao objeto *joão*. Já as fórmulas são expressões que podem ser relacionadas a um valor verdade: verdadeiro ou falso.

O conjunto dos termos da linguagem da Lógica de Predicados pode ser definido por uma gramática na forma BNF Huth e Ryan, 2004:

$$t ::= x \mid c \mid f(t, \cdots, t),$$

em que x é uma variável, c é uma constante e f é um símbolo funcional com aridade $n > 0$.

Já as fórmulas da Lógica de Predicados podem ser definidas em BNF, como a seguir Huth e Ryan, 2004:

$$\varphi ::= \bot \mid P(t_1, t_2, \cdots, t_n), \mid (\neg\varphi) \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi) \mid (\forall x\ \varphi) \mid (\exists x\ \varphi),$$

em que $\bot$ e P representam, respectivamente, a contradição e um símbolo predicado n-ário com $n \geq 0$, $t_1, \cdots t_n$ são termos e x é uma variável.

Nas fórmulas da Lógica de Predicados, as variáveis podem ocorrer *livres* ou *ligadas* (presas). Considere a seguinte fórmula φ:

$$(\exists x(P(x) \wedge Q(x)) \vee R(x, y).$$

Dizemos que a variável x é *livre* em φ se ela não é capturada por um quantificador ($\forall x$ ou $\exists x$) em φ, isto é, se a variável x não está no escopo de um quantificador $\forall x$ ou $\exists x$. Caso contrário, dizemos que a variável é ligada (presa) na fórmula φ. Na fórmula anterior, as duas primeiras ocorrências da variável x são *ligadas* (estão no escopo do quantificador $\exists x$) e a terceira ocorrência desta variável é *livre* (não está no escopo do quantificador $\exists x$). Já a única ocorrência da variável y é *livre*.

Definimos φ_t^x como a expressão obtida da fórmula φ pela substituição da variável x, sempre que ela ocorrer livre em φ, pelo termo t. Por exemplo, $(H(x) \rightarrow \forall x M(x))_y^x = (H(y) \rightarrow \forall x M(x))$. Dizemos que o termo t é substituível para x em φ se não existe uma variável y em t tal que ela é capturada por um ($\forall y$ ou $\exists y$) quantificador de φ_t^x. Por exemplo, o termo z é substituível para y em $\forall x P(x, y)$. Por outro lado, o termo x não é substituível para y em $\forall x P(x, y)$.

As demonstrações de Dedução Natural para a Lógica de Predicados são semelhantes às demonstrações para a Lógica Proposicional com a adição de novas regras para tratar dos *quantificadores*. Deste modo, qualquer regra de Dedução Natural para a Lógica Proposicional pode ser aplicada também para as fórmulas da Lógica de Predicados. A seguir, serão descritas as regras para introdução e eliminação dos quantificadores universal e existencial.

Eliminação do Universal ($\forall e$): A regra da eliminação do universal ($\forall e$) é apresentada na Figura 40, na qual a fórmula φ_t^x pode ser concluída em uma linha p se $\forall x \varphi$ foi demonstrada na linha m, desde que o termo t seja substituível pela variável x em φ. A Figura 41 exibe a aplicação $\forall e$ da fórmula $H(s) \rightarrow M(s)$ na linha 3 a partir da fórmula $\forall x(H(x) \rightarrow M(x))$, definida na linha 1.

⋮	⋮	⋮
m.	$\forall x\varphi$	
⋮	⋮	⋮
p.	φ_t^x	$\forall$e m

t é substituível para x em φ

Figura 40: Regra Universal Eliminação

1.	$\forall x(H(x) \rightarrow M(x))$	premissa
2.	H(s)	premissa
3.	H(s) $\rightarrow M(s)$	$\forall e$ 1
4.	M(s)	$\rightarrow e$ 2,3

Figura 41: $\forall x(H(x) \rightarrow M(x)), H(s) \vdash M(s)$

Figura 42: Regra Universal Eliminação ($\forall e$)

Introdução do Universal ($\forall i$): A regra da introdução do universal ($\forall i$) é apresentada na Figura 43, na qual para provarmos $\forall x\varphi$, na linha $n+1$, iremos supor que para uma variável a qualquer (arbitrária) em uma caixa que inicia na linha m e que conclui φ_a^x, na linha n. Dizemos que a variável a é qualquer se ela é uma variável nova na linha m, ou seja, a não ocorre como variável livre de qualquer fórmula que aconteça anteriormente a linha m que não esteja em uma caixa fechada. Na Figura 44, ilustramos a introdução do universal na linha 7 para provar $\forall xM(x)$ a partir da escolha de a, no início da caixa, na linha 3, e demonstramos $M(a)$ ao final da caixa, na linha 6. Note que a variável a escolhida não é uma variável livre das fórmulas das linhas 1 e 2.

article graphicx subcaption

⋮	⋮	⋮
m.	a	
⋮	⋮	⋮
n.	φ_a^x	
n+1.	$\forall x\varphi$	$\forall$i m-n

a é uma variável nova

Figura 43: Universal Introdução ($\forall i$)

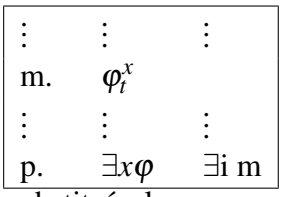
1.	$\forall x(H(x) \rightarrow M(x))$	premissa
2.	$\forall xH(x)$	premissa
3.	a	
4.	H(a)$\rightarrow M(a)$	$\forall e$ 1
5.	H(a)	$\forall e$ 2
6.	M(a)	$\rightarrow e$ 5,4
7.	$\forall xM(x)$	$\forall i$ 3-6

Figura 44: $\forall x(H(x) \rightarrow M(x)), \forall xH(x) \vdash \forall xM(x)$

Figura 45: Regra Universal Introdução ($\forall i$)

Introdução do Existencial ($\exists i$): A regra da introdução do existencial ($\exists i$) é apresentada na Figura 46, na qual a fórmula $\exists x\varphi$ pode ser concluída em uma linha p se φ_t^x foi demonstrada na linha m, desde que o termo t seja substituível para a variável x em φ. A Figura 47 exibe a aplicação $\exists i$ da fórmula $\exists xP(x)$ na linha 3 a partir da fórmula $P(a)$, definida na linha 1.

Eliminação do Existencial ($\exists e$): A regra da eliminação do existencial ($\exists e$) é apresentada na Figura 49, na qual para provarmos uma fórmula α, na linha $p+1$, iremos eliminar o existencial da fórmula $\exists x\varphi$, na linha m, supondo a fórmula φ_a^x para alguma variável a em uma caixa que inicia na linha n e que concluiu com uma fórmula α ao final da caixa, na linha p, desde que a não ocorra



t é substituível para x em φ

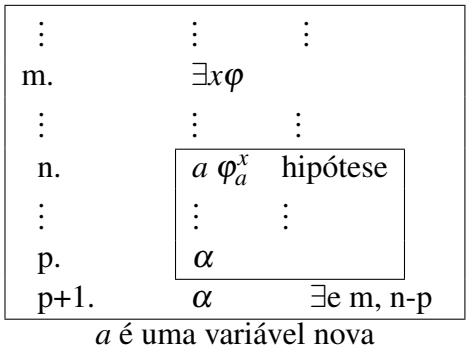
Figura 46: Existencial ($\exists i$)

1.	P(a)	premissa
2.	$\exists xP(x) \rightarrow B$	premissa
3.	$\exists xP(x)$	$\exists$ i 1
4.	B	$\rightarrow$ e 3,2

Figura 47: $P(a), \exists xP(x) \rightarrow B \vdash B$

Figura 48: Regra Existencial Introdução ($\exists i$)

na conclusão α. Nesta regra sabemos que a fórmula φ vale para algum elemento. Entretanto, não podemos assumir nenhuma propriedade específica para esta variável. Assim, a variável a deve ser uma variável nova na linha n, ou seja, a não ocorre como variável livre de qualquer fórmula que aconteça anteriormente a linha n que não esteja em uma caixa fechada e nem pode estar na conclusão α da regra. Na Figura 50, ilustramos a eliminação do existencial para concluir $\perp$ na linha 9, a partir da fórmula $\exists xH(x)$, na linha 3, e pela caixa que inicia na linha 4 com a suposição da fórmula $H(a)$ com a (nova) variável a e que termina a caixa na linha 8 com a conclusão $\perp$.



a é uma variável nova
a não ocorre em α

Figura 49: Existencial Eliminação ($\exists e$)

1.	$\forall x(H(x) \rightarrow M(x))$		premissa
2.	$\forall x\neg M(x)$		premissa
3.	$\exists xH(x)$		hipótese
4.	a	H(a)	hipótese
5.		H(a)$\rightarrow M(a)$	$\forall e$ 1
6.		M(a)	$\rightarrow e$ 4,5
7.		$\neg M(a)$	$\forall e$ 2
8.		$\perp$	$\neg e$ 6,7
9.	$\perp$		$\exists e$ 3, 4-8
10.	$\neg\exists xH(x)$		$\neg i$ 3-9

Figura 50: $\forall x(H(x) \rightarrow M(x)), \forall x\neg M(x) \vdash \neg\exists xH(x)$

Figura 51: Regra Existencial Eliminação ($\exists e$)

2.3 Redes Neurais Transformacionais

Aprendizado de Máquina é uma subárea da Inteligência Artificial que estuda modelos e algoritmos capazes de aprender a partir de dados. As *Redes Neurais Artificiais* (RNAs) fundamentam-se na estrutura e funcionamento do cérebro humano, consistindo em um conjunto interconectado de unidades de processamento chamadas neurônios artificiais, que são organizados em camadas. Cada neurônio recebe entradas, aplica uma função de ativação e gera uma saída. As informações são passadas por meio das camadas até que a saída seja gerada. As RNAs são usadas em uma

enorme variedade de aplicações, como reconhecimento de padrões, classificação, regressão, visão computacional e processamento de linguagem natural Russell, 2010. A Figura 52 ilustra a arquitetura básica de uma RNA com uma camada de entrada, duas camadas ocultas e uma camada de saída.

A arquitetura da rede neural está intimamente relacionada ao tipo de problema que se pretende resolver: certas arquiteturas são mais adequadas para certos tipos de dados e tarefas. Como exemplos de arquiteturas de redes neurais temos: *Multilayer Perceptrons* (MLPs), CNNs (Redes Neurais Convolucionais) e RNNs (Redes Neurais Recorrentes).

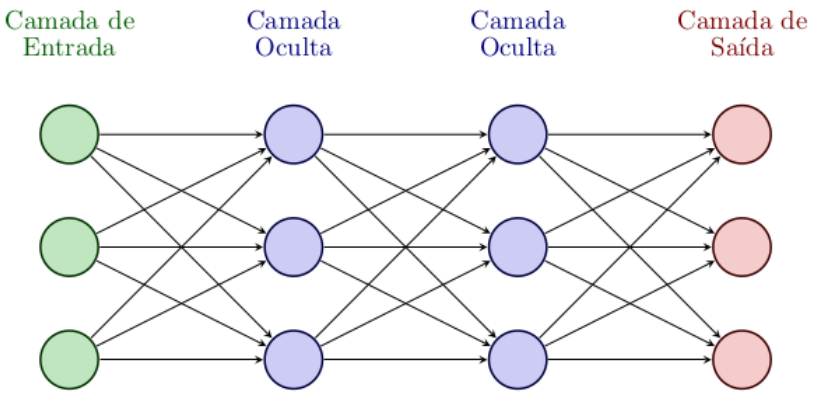


Figura 52: Arquitetura de uma RNA com múltiplas camadas.

As Redes Neurais Transformacionais (ou simplesmente *Transformers*) Vaswani et al., 2017 são uma arquitetura de redes neurais que têm desempenhado um papel fundamental no Processamento de Linguagem Natural (PLN). Utilizam mecanismos de *atenção* para permitir que cada elemento de entrada se comunique diretamente com todos os outros elementos da rede, o que fornece a capacidade de capturar relações de longo alcance nas sequências e lidar com contextos complexos.

O ChatGPT OpenAI, 2021 é um chatbot desenvolvido pela OpenAI que utiliza a rede neural transformacional *Generative Pre-trained Transformer* (GPT-3) Brown et al., 2020. Ele foi treinado com bilhões de palavras e trechos de texto de várias fontes na internet, permitindo que a ferramenta aprendesse a gramática, o estilo e o contexto das diferentes formas de comunicação humana. Uma das principais características do GPT-3 é a sua capacidade de gerar texto de forma autônoma. Ao receber uma pergunta ou uma instrução, o modelo processa a informação, consulta seu conhecimento prévio e gera uma resposta coerente e relevante. Ele pode incorporar o contexto da conversa anterior para fornecer respostas mais personalizadas e contextualmente apropriadas.

No entanto, é importante ressaltar que o GPT-3 é um modelo de linguagem estatístico e não possui compreensão profunda sobre certos tópicos. Embora seja capaz de produzir respostas incrivelmente impressionantes em determinadas situações, ele também pode gerar respostas incorretas em outras. A confiabilidade das respostas do ChatGPT depende da qualidade e da precisão dos dados com os quais o modelo foi treinado.

3 Trabalhos Relacionados

Em Yang et al., 2024 é apresentado o LeanDojo[2] que consiste em um conjunto de ferramentas, modelos e *benchmarks* de código aberto para prova de teoremas usando o assistente de provas *Lean* (De Moura et al., 2015). Também é apresentado o *ReProver*, um provador baseado em *LLM*, com baixa demanda de recursos computacionais e que necessita de pouco tempo de treinamento (alguns dias de GPUs). Os autores apresentam o conceito de Provadores de Teorema Iterativos (do inglês ITP - *Interactive Theorem Proving*), nos quais as provas são construídas por especialistas humanos interagindo com ferramentas tais como Coq (Coq, 1996), Isabelle (Nipkow et al., 2002) e Lean (De Moura et al., 2015). Os autores afirmam que os algoritmos de aprendizado de máquina podem automatizar essa interação, abrindo novas perspectivas para área de provadores automáticos de teorema (Yang e Deng, 2019): o modelo pode aprender a interagir com os assistentes de prova, a partir dos dados provenientes das provas escritas pelos especialistas humanos. Os autores evidenciam, porém, que a área de prova formal de teoremas impõe desafios para os algoritmos de aprendizado de máquina uma vez que há meios de se checar formalmente e automaticamente a correção das provas e, assim, não há espaço para que o modelo alucine. No entanto, acoplar LLMs com assistentes de provas pode melhorar a capacidade dos LLMs de realizar provas com raciocínio de vários passos (Schick et al., 2024). Todavia, os autores afirmam que a pesquisa de utilizar LLMs para prova de teoremas tem enfrentado desafios pelos seguintes fatos: (i) não há LLMs-baseados em provadores que sejam código aberto (First et al., 2023; Han et al., 2022; Jiang et al., 2021, 2022; Lample et al., 2022; Polu e Sutskever, 2020; Polu et al., 2022; Wang et al., 2023), não sendo possível reproduzir o treinamento e iterações com os assistentes de prova; (ii) eles usam dados pré-treinados privados e; (iii) necessitam de complexos recursos computacionais que podem chegar a milhares de dias de uso de GPUs.

O LeanDojo extrai dados do assistente de provas Lean e permite a interação com o ambiente de provas. Ela contém anotações de premissas nas provas, fornecendo dados para a tarefa de seleção de promissas, que é um gargalo na área de provadores de teorema. Usando estes dados, os autores construíram um novo *benchmark* com 98.734 teoremas e o usaram para treinar e avaliar o modelo do ReProver. Segundo o autor o treinamento levou apenas cinco dias em uma única GPU. O ReProver foi capaz de provar $51,2\%$ dos teoremas no *benchmark* do LeanDojo, superando um baseline que utiliza o GPT-4 $(29,0\%)$. Além disso, o *ReProver* foi capaz de provar 65 teoremas que atualmente não possuem provas no *Lean*, demonstrando sua capacidade de contribuir para a expansão das bibliotecas matemáticas existentes.

As semelhanças do trabalho aqui proposto e do trabalho de Yang e Deng, 2019 é que ambos estão avaliando o uso de LLMs na tarefa de *provas de teoremas*. No entanto, o trabalho de Yang e Deng, 2019 utiliza provas matemáticas no geral enquanto esse trabalho foca na resolução de um tipo específico de prova que é a dedução natural. Os autores do trabalho Yang e Deng, 2019 também constroem um novo provador de teoremas, treinado com os dados disponíveis no Lean, e, pelo mostrado nos experimentos, esta nova ferramenta melhorou o desempenho de generalização de provas de vários passos.

O trabalho de Ando et al., 2023 investiga se os LLMs atuais apresentam vieses de raciocínio lógico semelhantes a vieses apresentados por seres humanos. A avaliação é efetuada em raciocínio

[2]https://leandojo.org/

sobre silogismos, que é uma forma de inferência bem estudada em Lógica, consistindo em quatro tipos de sentenças categóricas como mostrado na Tabela 1.

Tabela 1: Quatro Tipos de Sentenças Categóricas. Tradução Livre de Ando et al., 2023.

Tipo	Forma	Descrição
A	Todo S é P	Afirmação Universal
E	Nenhum S é P	Negação Universal
I	Algum S é P	Afirmação Particular
O	Algum S não é P	Negação Particular

Cada silogismo possui duas premissas (P1 e P2) e uma conclusão (C). Assim, podemos identificar um silogismo de acordo com o tipo das premissas que o formam. O Silogismo 1 apresentado a seguir é do tipo **EIO**.

$$\begin{aligned} P1 &: \text{Nenhum B é C. (tipo } \mathbf{E}\text{)} \\ P2 &: \text{Algum A é B. (tipo } \mathbf{I}\text{)} \\ \hline C &: \text{Algum A não é C. (tipo } \mathbf{O}\text{)} \end{aligned} \tag{1}$$

Os vieses examinados foram de três tipos: *viés de crença*, *erros de conversão* e *efeitos de atmosfera*. Os *vieses de crença* ocorrem quando há dificuldade de determinar se uma inferência é válida quando a mesma possui uma *frase que é contrária ao senso comum*. No Silogismo 2, apresentado a seguir, podemos observar que *mesmo que o silogismo seja logicamente válido* o fato da conclusão ser contra o senso comum *pode levar algumas pessoas a julga-lá como contraditória*.

$$\begin{aligned} P1 &: \text{Alguns animais são seres humanos.} \\ P2 &: \text{Todos os animais são tomates.} \\ \hline C &: \text{Alguns humanos são tomates.} \end{aligned} \tag{2}$$

Os *erros de conversão* são erros causados pela interpretação incorreta de termos. Podemos observar no Silogismo 3 uma conclusão considerada neutra, que não podemos afirmar ou negar logicamente sua veracidade, entretanto as pessoas podem ter a tendência de julga-lá como válida ao considerar que a sentença "*Todos A são B*" é equivalente a "*Todos B são A*".

$$\begin{aligned} P1 &: \text{Todos B são A.} \\ P2 &: \text{Todos B são C.} \\ \hline C &: \text{Todos A são C.} \end{aligned} \tag{3}$$

Os *efeitos atmosféricos* podem ser interpretados a partir de dois princípios. O princípio da qualidade, no qual se uma ou ambas as premissas forem negativas (tipo E ou tipo O), a conclusão deve ser negativa; caso contrário, é positivo (tipo A ou tipo I). E, o princípio da quantidade: se uma ou ambas as premissas forem particulares (tipo I ou tipo O), então a conclusão será particular; caso contrário, é universal (tipo A ou tipo E). Podemos observar no Silogismo 4 uma conclusão considerada neutra, isto é, não se pode afirmar ou negar logicamente a sua veracidade.

$$\begin{array}{l} P1 : \text{Todos os animais são seres vivos.} \\ \underline{P2 : \text{Alguns humanos não são seres vivos.}} \\ C : \text{Nenhum dos animais é humano.} \end{array} \quad (4)$$

Para avaliação se os LLMs cometiam tais vieses descritos anteriormente, os autores criaram uma base de dados chamada *NeuBaroco*, tendo como base o dataset *Baroco* Shikishima et al., 2009, porém com questões adicionadas e rótulos atribuídos manualmente. Para cada inferência foi atribuído um rótulo de implicação, contradição ou neutro, sendo 375 inferências no total: 122 rotulados como implicação; 71 rotulados como contradição e; 182 como neutras. Foram avaliados os modelos RoBERTa Y. Liu et al., 2019 e BART Lewis et al., 2020 e GPT-3.5 (GPT-3.5-turbo[3]) e os resultados mostram que os modelos apresentam muitas falhas causadas por vieses semelhantes aos seres humanos.

As semelhanças entre o trabalho de Ando et al., 2023 e o proposto nesse artigo residem na avaliação do LLM GPT-3.5 em tarefas relacionadas ao raciocínio lógico. Ambos os estudos empregam inferências ligadas à lógica de predicados, chegando à conclusão de que os LLMs ainda não atingiram um desempenho satisfatório nesse tipo de tarefa. No entanto, enquanto os autores de Ando et al., 2023 utilizam textos em linguagem natural como entrada, esse artigo utiliza fórmulas nas linguagens da lógica proposicional e de predicados.

Em Saparov et al., 2023 foi avaliada a capacidade de raciocínio dedutivo geral de LLMs, medindo quão bem eles são capazes de aprender com provas simples e generalizar esse aprendizado para realizar provas mais complexas. A Tabela 2 mostra os tipos de generalização pretendidas pelos autores.

De modo geral, cada exemplo de treinamento é um exemplo de demonstração de sequência de raciocínio (do inglês, *CoT - Chain of Thoughts*), uma sequência lógica de ideias ou conceitos interconectados que formam um raciocínio ou argumento coerente, e cada exemplo de teste é um exemplo de prova que o modelo deve produzir. A complexidade das provas foram descritas em três grupos: (i) quantidade de regras de dedução envolvidas; (ii) profundidade da prova, i.e., o comprimento de uma cadeia sequencial de passos de uma prova e; (iii) largura da prova, i.e., o número de premissas de cada passo de prova.

Para medir a capacidade geral de raciocínio dedutivo dos LLMs, os autores: determinaram se os modelos aprenderam um conjunto completo de regras de dedução natural; avaliaram se os modelos conseguem raciocinar sobre provas mais longas do que as fornecidas como exemplos (generalização em profundidade e largura) e; avaliaram se os modelos são capazes de utilizar múltiplas regras de dedução diferentes em uma única prova (generalização composicional).

O conjunto de dados utilizado foi o PRONTOQA-OOD [4] no qual cada exemplo contém um conjunto de premissas, uma consulta (o fato alvo a ser provado ou reprovado) e uma cadeia de pensamento contendo a prova da consulta. Para avaliar o raciocínio usando diferentes regras de dedução, foram geradas provas com regras de dedução para todos os conectivos na lógica proposicional (conjunção, disjunção, implicação e negação). Para evitar que os LLMs explorem o conhecimento do pré-treinamento para resolver os problemas sem raciocínio, os autores utilizaram nomes fictícios para todos os conceitos (por exemplo, "*wumpus*" no lugar de "gato"). O conjunto

[3] https://platform.openai.com/docs/model-index-for-researchers

[4] https://github.com/asaparov/prontoqa

Tabela 2: Tipos de generalização avaliadas por Saparov et al., 2023.

Exemplo de treino	Tipo de teste	Exemplo de teste
"Alex é um cachorro. Todos os cães são mamíferos. Alex é um mamífero."	Teste sobre regras de dedução não vistas em treinamento	"Alex não é um mamífero. Todos os cães são mamíferos. Suponha que Alex seja um cachorro. Alex é um mamífero. Isso contradiz que Alex não é um mamífero. Alex não é um cachorro."
"Alex é um cachorro. Todos os cães são mamíferos. Alex é um mamífero."	Teste em provas mais profundas	"Alex é um cachorro. Todos os cães são mamíferos. Alex é um mamífero. Todos os mamíferos são vertebrados. Alex é um vertebrado."
"Alex é um cachorro. Alex é macio. Alex é um cachorro e macio".	Teste em provas mais amplas	"Alex é um cachorro. Alex é macio. Alex é gentil. Alex é um cachorro, macio e gentil."
"Alex é um cachorro. Todos os cães são mamíferos. Alex é um mamífero. Fae é um gato. Fae é agradável. Fae é agradável e um gato".	Teste em provas de composição	"Alex é um cachorro. Todos os cães são mamíferos. Alex é um mamífero. Alex não é mau. Alex é um mamífero e não é mau."

Fonte: Tradução Livre de Saparov et al., 2023.

de dados utilizado exige que os LLMs gerem provas completas, no lugar de respostas como verdadeiro e falso. Os experimentos foram realizados utilizando quatro LLMs, *FLAN-T5* Chung et al., 2024, *LLaMA* Touvron et al., 2023, *GPT-3.5* [5] e *PaLM* Chowdhery et al., 2023, de tamanhos e objetivos de treinamento variados. Para os experimentos, foram utilizadas oito demonstrações, sendo comparados os desempenhos em dois ambientes. No primeiro ambiente, os exemplos de contexto e os exemplos de teste vêm da são semelhantes aos utilizados nos treinamentos (chamados de exemplos de mesma distribuição). No segundo ambiente, o exemplo de teste possui regra de dedução específica, mas os exemplos no contexto são distribuídos uniformemente por todas as outras regras de dedução - exemplos chamados de fora de distribuição (do inglês, OOD - *Out of Distribution*). A Figura 53 apresenta um exemplo de questão do conjunto de dados contendo as premissas, o comando da prova (que deve ser provado ou rejeitado), a resposta fornecida pelo modelo (resposta prevista) e, logo após, a resposta esperada.

De modo geral, não foi observada uma forte correlação entre tamanho do modelo e desempenho. Além disso, a maioria dos modelos provou ser capaz de usar cada regra de dedução razoavelmente bem, com o GPT-3.5 apresentando o melhor desempenho. Os modelos foram capazes de usar quatro regras de dedução natural, apesar de não ser mostrado um exemplo no contexto com essas regras: introdução de disjunção, eliminação de implicações, conjunção introdução e conjunção eliminação. Por fim, os modelos foram capaz de gerar provas composicionais até certo ponto e tiveram dificuldade de gerar provas mais longas, além de requerem demonstração explícita para produzir sub provas hipotéticas, principalmente em provas por casos e contradição.

[5] https://platform.openai.com/docs/model-index-for-researchers

Premissas: Lempuses são gorpuses. Rompuses são wumpuses. Tumpuses são wumpuses. Dumpuses são gorpuses. Todo zumpus é um wumpus. Todo brimpus é um gorpus. Yumpuses são gorpuses. Lempuses são wumpuses. Rompuses são gorpuses. Tumpuses são gorpuses. Max é um tumpus ou um rompus ou um lempus. Max é um zumpus, um tumpus, ou um rompus.
Prove: Max é um gorpus.
Resposta prevista: Max é um tumpus ou um rompus ou um lempus. Max é um tumpus. Tumpuses são wumpuses. Max é um wumpus. Rompuses são gorpuses. Max é um gorpus. Max é um gorpus.
Resposta esperada: Suponha que Max é um tumpus. Tumpuses são gorpuses. Max é um gorpus. Suponha que Max é um rompus. Rompuses são gorpuses. Max é um gorpus. Suponha que Max é um lempus. Lempuses são gorpuses. Max é um gorpus. Visto que Max é um tumpus ou um rompus ou um lempus, Max é um gorpus.

Figura 53: Exemplo de uma prova incorreta gerada pelo GPT-3.5. As premissas utilizadas na prova estão escritas em azul e os passos inválidos em vermelho.

Assim como o nosso trabalho, os autores em Saparov et al., 2023 buscam avaliar o desempenho de LLMs, incluindo o GPT 3.5, na tarefa de realizar provas complexas de dedução natural. Entretanto no conjunto de dados utilizado em Saparov et al., 2023, as questões são descritas em linguagem natural (que poderia ser traduzida para a linguagem da lógica de predicados) e o conjunto de dados que usamos neste trabalho utiliza as questões descritas como fórmulas nas linguagens das lógicas proposicional e de predicados. O estudo de Saparov et al., 2023 tem conclusões semelhantes ao apresentado neste artigo sobre a dificuldade dos LLMs de aplicar múltiplas regras de dedução natural para obtenção dos vários passos das provas pretendidas. No entanto, os autores em Saparov et al., 2023 realizam um treinamento para que os LLMs aprendam sobre as regras de dedução natural para só depois avaliar se eles são capazes de aplicá-las em diferentes contextos de provas. Neste trabalho, nós não treinamos o LLM utilizado, pois o objetivo era avaliar sua capacidade genuína de escrever provas de dedução natural.

4 Metodologia

Para avaliar o desempenho do modelo GPT-3.5-turbo[6] na tarefa de construir provas de exercícios de Dedução Natural foram utilizados um conjunto de exercícios aplicados durante a disciplina de Lógica para Computação para alunos de graduação em cursos da área de Tecnologia da Informação da Universidade Federal do Ceará - Campus Quixadá, a saber: 41 exercícios de lógica proposicional e 20 exercícios de lógica de predicados[7].

Os exercícios são textos em LaTeX (e.g, `$A \rightarrow B, \lnot B \vdash \lnot A$`) correspondentes a teoremas com premissas e conclusão:

- $A \rightarrow B, \neg B \vdash \neg A$ (com premissas $A \rightarrow B$ e $\neg B$ e conclusão $\neg A$).

[6] https://platform.openai.com/docs/models
[7] Exercícios disponíveis em https://github.com/leonardomartins777/ChatGPT_Naturalded

Para a comunicação com o modelo *GPT-3.5-turbo* foi utilizada a API (do inglês, *Application Programming Interface*) disponibilizada pela *OpenAI*[8], que foi acessada por meio de sua biblioteca na linguagem de programação *Python*. O modelo recebe uma conversa, que consiste em uma lista de mensagens como entrada e retorna uma mensagem gerada como saída. A seguir, mostramos exemplos de exercícios fornecidos ao modelo em lógica proposicional e em lógica de predicados.

4.1 Exemplo de Exercício de Lógica Proposicional Fornecido ao Modelo

Inicialmente, um exercício exemplo de lógica proposicional foi submetido à ferramenta cujo enunciado é um texto com um teorema em LaTeX (conforme mostrado na Figura 54) e a saída é um código LaTeX (conforme mostrado na Figura 55), no formato adotado pelo livro texto "*Lógica em Ciência da Computação: Modelagem e Raciocínio sobre sistemas*" Huth e Ryan, 2004, o qual utiliza o pacote `logicproof`[9] de escrita de provas de dedução natural no estilo *Fitch*. Em seguida, novos exercícios são submetidos ao modelo para que ele construa a sua solução, como o exemplo apresentado na Figura 56.

"escreva a prova `$A \rightarrow B, B \rightarrow C \vdash A \rightarrow C$` usando o sistema de dedução natural no estilo fitch e o pacote latex logicproof."

Figura 54: Pergunta exemplo fornecida ao modelo.

```
\begin{logicproof}{6}
    A \rightarrow B & premissa \\
    B \rightarrow C & premissa \\
    \begin{subproof}
        A & hipótese \\
        B & $\rightarrow e$ 3,1 \\
        C & $\rightarrow e$ 4,2
    \end{subproof}
    A \rightarrow C & $\rightarrow i$ 3-5
\end{logicproof}
```

Figura 55: Resposta exemplo fornecida ao modelo.

"escreva a prova `\vdash A \rightarrow (B \rightarrow A)` usando o sistema de dedução natural no estilo fitch e o pacote latex logicproof"

Figura 56: Pergunta fornecida para que o modelo construa sua demonstração.

[8] https://platform.openai.com/docs/api-reference
[9] https://ctan.org/pkg/logicproof

4.2 Exemplo de Exercício de Lógica Predicados Fornecido ao Modelo

Assim como em logica proposicional, um exercício exemplo de lógica de predicados foi submetido à ferramenta com o enunciado sendo um texto com o seguinte teorema em LaTeX: $\forall x(H(x) \rightarrow M(x)), \forall x \neg M(x) \vdash \neg \exists x H(x)$ (conforme mostrado na Figura 57) e a saída um código LaTeX (conforme mostrado na Figura 58). Depois os novos exercícios são submetidos ao modelo para que ele construa a sua solução, como é possível observar na Figura 59.

"escreva a prova `$ \forall x (H(x) \rightarrow M(x)),\forall x \lnot M(x) \vdash \lnot \exists x H(x)$`
usando o sistema de dedução natural no estilo fitch e o pacote latex logicproof."

Figura 57: Pergunta exemplo fornecida ao modelo.

```
\begin{logicproof}{6}
    \forall x (H(x)\rightarrow M(x)) & premissa \\
    \forall x \lnot M(x) & premissa \\
    \begin{subproof}
        \exists x H(x) & hipótese \\
        \begin{subproof}
            \llap{a\quad}H(a) & hipótese \\
            H(a)\rightarrow M(a) & \forall e 1 \\
             M(a) & \rightarrow e 4, 5 \\
             \lnot M(a) & \forall e 2 \\
             \bot & \lnot e 6, 7
        \end{subproof}
        \bot & ∃e 3,4-8
    \end{subproof}
    \lnot \exists x H(x) & \lnot i 3-9
\end{logicproof}
```

Figura 58: Resposta exemplo fornecida ao modelo.

"escreva a prova `\forall x P(x) \vdash \lnot \exists x \lnot P(x)`
usando o sistema de dedução natural no estilo fitch e o pacote latex logicproof"

Figura 59: Pergunta fornecida para que o modelo construa sua demonstração.

5 Análise Experimental

Avaliamos se nas respostas fornecidas *as regras de Dedução Natural eram aplicadas corretamente* e se as provas foram escritas de acordo com a sintaxe do pacote `logicproof`.

5.1 Resultados para os Exercícios de Lógica Proposicional

A Figura 60 mostra um exemplo de demonstração realizada corretamente pelo ChatGPT para o enunciado $\vdash (A \to (B \to C)) \to ((A \to B) \to (A \to C))$. Nesta resposta, o modelo escreveu um código LaTeX, o qual foi compilado e corresponde a demonstração mostrada na figura. Veja em mais detalhes que o modelo: fez a suposição correta das hipóteses nas linhas 1, 2 e 3; aplicou corretamente a regra da implicação eliminação ($\to e$) e referenciou corretamente as linhas com a fórmula contendo o condicional e a veracidade do antecedente nas linhas 4, 5 e 6; aplicou corretamente a regra da implicação introdução ($\to i$) e referenciou corretamente os intervalos das caixas nas linhas 7, 8 e 9.

1.	$A \to (B \to C)$	hipótese
2.	$A \to B$	hipótese
3.	A	hipótese
4.	B	$\to e$ 3, 2
5.	$B \to C$	$\to e$ 1, 3
6.	C	$\to e$ 4, 5
7.	$A \to C$	$\to i$ 3-6
8.	$(A \to B) \to (A \to C)$	$\to i$ 2-7
9.	$(A \to (B \to C)) \to ((A \to B) \to (A \to C))$	$\to i$ 1-8

Figura 60: Prova correta para $\vdash (A \to (B \to C)) \to ((A \to B) \to (A \to C))$, resultado da compilação do código LaTeX escrito pelo modelo.

No entanto, a maior parte das respostas obtidas continham algum tipo de erro, os quais foram classificados em: *erros lógicos*, quando ocorrem erros na aplicação das regras de Dedução Natural e; *erros de referências nas regras*, quando as respostas, apesar das regras serem utilizadas adequadamente, apresentavam erros nas referências durante sua aplicação.

Nos *erros lógicos* avaliamos todos os erros decorrentes do mau uso das regras de Dedução Natural e incoerências lógicas em geral, que são os erros mais críticos para o objetivo do experimento. A partir dos resultados, podemos observar que a maioria das respostas apresentam erros lógicos, um total de 31 (75,61%). Os erros de escrita de prova foram bem frequentes nestes experimentos. Destacamos aqui as questões que tiveram exclusivamente erros de referência de linhas, no qual são feitas referências incorretas às linhas durante a aplicação de determinadas regras. Esses erros totalizaram apenas 04 questões (9,76%). Dessa forma, podemos observar uma grande imprecisão do modelo na tarefa de escrever as demonstrações, totalizando 35 questões erradas (85,37%). Em algumas respostas produzidas pelo modelo, há vários erros dessa categoria.

A Figura 61 ilustra uma resposta incorreta produzida pelo modelo na qual há erros de escrita de prova (referência de linhas) e também erros lógicos. Nesta demonstração, o modelo:

- referenciou incorretamente nas linhas 3 e 4, as linhas 1, 1, uma vez que a aplicação da regra $\vee e$ deveria referenciar apenas a linha 2;

- referenciou incorretamente na linha 6, a linha 2 na aplicação da regra $\wedge e$, uma vez que deveria ter mencionado a linha 5 (o mesmo ocorre na linha 8);
- referenciou incorretamente na linha 7, as linhas $3,2$, uma vez que a aplicação da regra $\vee i$ deveria referenciar apenas a linha 6;
- referenciou incorretamente na linha 9, as linhas $3,4$, uma vez que a aplicação da regra $\vee i$ deveria referenciar apenas a linha 8 e;
- **cometeu um erro lógico** na linha 10 no uso da regra $\wedge i$, a qual não poderia ser aplicada nesse contexto (O correto seria utilizar a regra $\vee e$ para obter $A \vee C$. No entanto, não seria possível obter $(A \vee B)$).

A Figura 62 apresenta uma prova correta na qual é usada a regra $\vee e$ na linha 12 para se obter o resultado esperado. É interessante observar que nas linhas 5 e 11 existem adições de passos (que fazem uso da regra $\wedge i$) que tornam o uso da regra $\vee e$ na linha 12 viável. Além disso, todas as referencias correspondentes as da Figura 61 foram corrigidas.

1.	$A \vee (B \wedge C)$	premissa
2.	A	hipótese
3.	$A \vee B$	$\vee i$ 1, 1
4.	$A \vee C$	$\vee i$ 1, 1
5.	$B \wedge C$	hipótese
6.	B	$\wedge e_1$ 2
7.	$A \vee B$	$\vee i$ 3, 2
8.	C	$\wedge e_2$ 2
9.	$A \vee C$	$\vee i$ 3, 4
10.	$(A \vee B) \wedge (A \vee C)$	$\wedge i$ 2-6

Figura 61: Demonstração incorreta realizada pelo modelo.

1.	$A \vee (B \wedge C)$	premissa
2.	A	hipótese
3.	$A \vee B$	$\vee i$ 2
4.	$A \vee C$	$\vee i$ 2
5.	$(A \vee B) \wedge (A \vee C)$	$\wedge i$ 3,4
6.	$B \wedge C$	hipótese
7.	B	$\wedge e$ 6
8.	$A \vee B$	$\vee i$ 7
9.	C	$\wedge e$ 6
10.	$A \vee C$	$\vee i$ 9
11.	$(A \vee B) \wedge (A \vee C)$	$\wedge i$ 8,10
12.	$(A \vee B) \wedge (A \vee C)$	$\vee e$ 1, 2-5, 6-11

Figura 62: Exemplo de demonstração correta.

Figura 63: Exemplos de demonstrações para $A \vee (B \wedge C) \vdash (A \vee B) \wedge (A \vee C)$, resultado da compilação do código LaTeX escrito pelo modelo. Erros destacados em vermelho e correções em azul.

A Tabela 3 resume o desempenho do modelo ChatGPT na tarefa de resolução de exercícios de dedução natural em lógica proposicional, onde ele teve apenas 06 acertos completos num total de 41 exercícios (14,63%). Além disso, podemos observar os tipos de erros mais comuns apresentados pelo modelo, sendo que para cada resposta mais de um erro pode ter ocorrido.

Tabela 3: Desempenho apresentado pelo modelo na tarefa de dedução natural

Modelo	Acertos	Erros Apenas de Referência	Erros Lógicos
ChatGPT	6	4	31
Total (%)	14,63%	9,76%	75,61%

5.2 Resultados para os Exercícios em Lógica de Predicados

O desempenho do modelo na resolução de exercícios de lógica de predicados foi tão ruim quanto os apresentados para a lógica proposicional. Na Figura 64, apresentamos um exemplo de resposta correta produzida pelo modelo na solução do seguinte enunciado: $\forall x P(x) \vdash \neg\exists\neg P(x)$. Podemos observar na resposta a declaração correta da premissa (linha 1), a suposição correta de hipóteses (linhas 2 e 3) e aplicação correta das regras eliminação do quantificador universal ($\forall e$), negação eliminação ($\neg e$), eliminação do quantificador existencial ($\exists e$) e negação introdução ($\neg i$) nas linhas 4, 5, 6 e 7 respectivamente. Além disso, as duas caixas (subprovas) foram bem construídas e os intervalos das linhas nas referências foram descritos corretamente.

1.		$\forall x P(x)$	premissa
2.		$\exists x \neg P(x)$	hipótese
3.	a	$\neg P(a)$	hipótese
4.		$P(a)$	$\forall e$ 1
5.		$\bot$	$\neg e$ 3, 4
6.		$\bot$	$\exists e$ 2, 3-5
7.		$\neg\exists x \neg P(x)$	$\neg i$ 2-6

Figura 64: Prova correta para $\forall x P(x) \vdash \neg\exists x \neg P(x)$, resultado da compilação do código LaTeX escrito pelo modelo. Erros destacados em vermelho e correções em azul.

No entanto. na Figura 65, podemos observar um exemplo onde o modelo respondeu incorretamente o exercício $\exists x(P \to Q(x)) \vdash P \to \exists x Q(x)$. A prova possui os seguintes **erros lógicos**:

- na linha 3 a variável b é declarada e não utilizada durante a prova;
- na linha 4 a regra da implicação eliminação ($\to e$) foi utilizada de forma incorreta, pois as linhas referenciadas não correspondem aos requisitos de uso da regra aplicada;
- na linha 6 a regra da implicação introdução ($\to i$) foi utilizada de forma incorreta, pois, o intervalo referenciado não possui os requisitos de uso da regra aplicada e;
- na linha 6 a caixa externa é fechada sem concluir nada;

Como **erros de referência** podemos observar na linha 5 a referência da linha na regra da introdução do existencial ($\exists i$) que deveria referenciar a linha 4 e na linha 6 a referência das linhas na aplicação da regra de implicação introdução ($\to i$) deveria ser as *3-5* e não *2-4*.

A Figura 66 apresenta um exemplo de demostração correta para o exercício $\exists x(P \rightarrow Q(x)) \vdash P \rightarrow \exists x Q(x)$: a variável a é declarada de forma correta e utilizada em sequência durante o resto da prova, as referências nas linhas 4 e 5 foram corrigidas, no caso da linha 6 foi utilizada a regra de $\exists e$ para se chegar na fórmula $\exists x Q(x)$, e na linha 7 se aplicar a regra $\rightarrow i$ para se chegar na conclusão desejada.

1.		$\exists x(P \rightarrow Q(x))$	premissa
2.	a	P	hipótese
3.	b	$P \rightarrow Q(a)$	hipótese
4.		$Q(a)$	$\rightarrow e$ 1, 2
5.		$\exists x Q(x)$	$\exists i$ 3
6.		$P \rightarrow \exists x Q(x)$	$\rightarrow i$ 2-4
7.			

Figura 65: Exemplo de demonstração incorreta realizada pelo modelo.

1.		$\exists x(P \rightarrow Q(x))$	premissa
2.		P	hipótese
3.	a	$P \rightarrow Q(a)$	hipótese
4.		$Q(a)$	$\rightarrow e$ 2, 3
5.		$\exists x Q(x)$	$\exists i$ 4
6.		$\exists x Q(x)$	$\exists e$ 1,3-5
7.		$P \rightarrow \exists x Q(x)$	$\rightarrow i$ 2-6

Figura 66: Exemplo de demonstração correta.

Figura 67: Exemplos de demonstrações para $\exists x(P \rightarrow Q(x)) \vdash (P \rightarrow \exists x Q(x))$, resultado da compilação do código LaTeX escrito pelo modelo. Erros destacados em vermelho e correções em azul.

A Tabela 4 apresenta o desempenho do modelo ChatGPT nas 20 questões de Dedução Natural em Lógica de Predicados. Diferentemente dos resultados dos experimentos obtidos nas questões de lógica proposicional, não verificamos aqui a ocorrência de erros apenas de referências de linhas, pois as demonstrações que apresentavam erros de referência também possuíam erros lógicos. Entretanto, nem todas as demostrações que possuíam erros lógicos também apresentavam erros de referências.

Tabela 4: Desempenho apresentado pelo modelo na tarefa de dedução natural

Modelo	Acertos	Erros de Referência	Erros Lógicos
ChatGPT	3	13	17
Total (%)	15,0%	65,0%	85,0%

6 Conclusão e Trabalhos Futuros

Neste artigo, avaliamos o desempenho do ChatGPT na tarefa de construir provas em dedução natural na lógica proposicional e lógica de predicados. Após a análise dos resultados, fica evidente que o modelo apresenta limitações significativas nesse domínio. Dos 41 exercícios de lógica proposicional fornecidos, o modelo acertou apenas 06 exercícios (14,63% do total) e; dos 20 exercícios de lógica de predicados, o modelo acertou a resolução de apenas 3 exercícios (15% do total).

Várias são as razões possíveis para o desempenho insatisfatório do modelo. Em primeiro lugar, a construção de provas de dedução natural tanto em lógica proposicional como em lógica de predicados requer mais que um conhecimento sobre como aplicar as regras de inferência: é necessário que o modelo seja capaz de generalizar esse conhecimento específico das regras e saber que regra aplicar e em qual momento. O ChatGPT pareceu ter dificuldades tanto em selecionar uma regra apropriada como também em aplicar corretamente essas regras. Além disso, é importante considerar que o modelo foi treinado em uma ampla variedade de textos e não recebeu treinamento específico em lógica proposicional e lógica de predicados.

Como trabalhos futuros, pretendemos: treinar o modelo para verificar se ele é capaz de aprender as regras de dedução natural em lógica proposicional e de predicados e avaliar se há melhora no desempenho; utilizar bancos de questões públicos como conjunto de dados e; verificar o desempenho do modelo na tarefa de resolução de exercícios de outras temáticas envolvendo lógica proposicional e lógica de predicados tais como resolução de exercícios de argumentação em linguagem natural e exercícios de *tableaux* analíticos.

Referências

Ando, R., Morishita, T., Abe, H., Mineshima, K., & Okada, M. (2023, junho). Evaluating Large Language Models with NeuBAROCO: Syllogistic Reasoning Ability and Human-like Biases. Em S. Chatzikyriakidis & V. de Paiva (Ed.), *Proceedings of the 4th Natural Logic Meets Machine Learning Workshop* (pp. 1–11). Association for Computational Linguistics. https://aclanthology.org/2023.naloma-1.1

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, *33*, 1877–1901.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., … Fiedel, N. (2023). PaLM: Scaling Language Modeling with Pathways. *Journal of Machine Learning Research*, *24*(240), 1–113. http://jmlr.org/papers/v24/22-1144.html

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S. S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., … Wei, J. (2024). Scaling Instruction-Finetuned Language Models. *Journal of Machine Learning Research*, *25*(70), 1–53. http://jmlr.org/papers/v25/23-0870.html

Coq, P. (1996). The Coq Proof Assistant-Reference Manual. *INRIA Rocquencourt and ENS Lyon, version*, *5*.

De Moura, L., Kong, S., Avigad, J., Van Doorn, F., & von Raumer, J. (2015). The Lean theorem prover (system description). *Automated Deduction-CADE-25: 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings 25*, 378–388.

Enderton, H. B. (1972). *A mathematical introduction to logic*. Academic Press.

Ferreirós, J. (2001). The road to modern logic—an interpretation. *Bulletin of Symbolic Logic*, *7*(4), 441–484.

First, E., Rabe, M., Ringer, T., & Brun, Y. (2023). Baldur: Whole-proof generation and repair with large language models. *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 1229–1241.

Hammer, E. M. (1998). Semantics for existential graphs. *Journal of Philosophical Logic*, *27*, 489–503.

Han, J. M., Rute, J., Wu, Y., Ayers, E. W., & Polu, S. (2022). PROOF ARTIFACT CO-TRAINING FOR THEOREM PROVING WITH LANGUAGE MODELS. *International Conference on Learning Representations*.

Huth, M., & Ryan, M. (2004). *Logic in Computer Science: Modelling and Reasoning about Systems (2nd Ed.)* Cambridge University Press. https://doi.org/10.1017/CBO9780511810275

Jiang, A. Q., Li, W., Han, J. M., & Wu, Y. (2021). Lisa: Language models of isabelle proofs. *6th Conference on Artificial Intelligence and Theorem Proving*, 378–392.

Jiang, A. Q., Li, W., Tworkowski, S., Czechowski, K., Odrzygóźdź, T., Miłoś, P., Wu, Y., & Jamnik, M. (2022). Thor: Wielding hammers to integrate language models and automated theorem provers. *Advances in Neural Information Processing Systems*, *35*, 8360–8373.

Kasneci, E., Seßler, K., Küchemann, S., Bannert, M., Dementieva, D., Fischer, F., Gasser, U., Groh, G., Günnemann, S., Hüllermeier, E., et al. (2023). ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and Individual Differences*, *103*, 102274.

Lample, G., Lacroix, T., Lachaux, M.-A., Rodriguez, A., Hayat, A., Lavril, T., Ebner, G., & Martinet, X. (2022). Hypertree proof search for neural theorem proving. *Advances in neural information processing systems*, *35*, 26337–26349.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2020, julho). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. Em D. Jurafsky, J. Chai, N. Schluter & J. Tetreault (Ed.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 7871–7880). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.703

Liu, H., Ning, R., Teng, Z., Liu, J., Zhou, Q., & Zhang, Y. (2023). Evaluating the logical reasoning ability of chatgpt and gpt-4. *arXiv preprint arXiv:2304.03439*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Nipkow, T., Wenzel, M., & Paulson, L. C. (2002). *Isabelle/HOL: a proof assistant for higher-order logic*. Springer.

OpenAI. (2021). ChatGPT [Acesso em: 13 de junho de 2023.].

Pelletier, F. J. (1999). A brief history of natural deduction. *History and Philosophy of Logic*, *20*(1), 1–31.

Pelletier, F. J. (2000). A history of natural deduction and elementary logic textbooks. *Logical consequence: Rival approaches*, *1*, 105–138.

Polu, S., Han, J. M., Zheng, K., Baksys, M., Babuschkin, I., & Sutskever, I. (2022). Formal mathematics statement curriculum learning. *arXiv preprint arXiv:2202.01344*.

Polu, S., & Sutskever, I. (2020). Generative language modeling for automated theorem proving. *arXiv preprint arXiv:2009.03393*.

Russell, S. J. (2010). *Artificial intelligence a modern approach*. Pearson Education, Inc.

Saparov, A., Pang, R. Y., Padmakumar, V., Joshi, N., Kazemi, M., Kim, N., & He, H. (2023). Testing the General Deductive Reasoning Capacity of Large Language Models Using OOD Examples. Em A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt & S. Levine (Ed.), *Advances in Neural Information Processing Systems* (pp. 3083–3105, Vol. 36). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2023/file/09425891e393e64b0535194a81ba15b7-Paper-Conference.pdf

Schick, T., Dwivedi-Yu, J., Dessı, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., & Scialom, T. (2024). Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, *36*.

Shikishima, C., Hiraishi, K., Yamagata, S., Sugimoto, Y., Takemura, R., Ozaki, K., Okada, M., Toda, T., & Ando, J. (2009). Is g an entity? A Japanese twin study using syllogisms and intelligence tests. *Intelligence*, *37*(3), 256–267.

Tlili, A., Shehata, B., Adarkwah, M. A., Bozkurt, A., Hickey, D. T., Huang, R., & Agyemang, B. (2023). What if the devil is my guardian angel: ChatGPT as a case study of using chatbots in education. *Smart Learning Environments*, *10*(1), 15.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., & Lample, G. (2023). LLaMA: Open and Efficient Foundation Language Models.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

Wang, H., Yuan, Y., Liu, Z., Shen, J., Yin, Y., Xiong, J., Xie, E., Shi, H., Li, Y., Li, L., et al. (2023). Dt-solver: Automated theorem proving with dynamic-tree sampling guided by proof-level value function. *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 12632–12646.

Weber, F., Wambsganss, T., Rüttimann, D., & Söllner, M. (2021). Pedagogical agents for interactive learning: A taxonomy of conversational agents in education. *Forty-Second International Conference on Information Systems. Austin, Texas*, 1–17.

Yang, K., & Deng, J. (2019). Learning to prove theorems via interacting with proof assistants. *International Conference on Machine Learning*, 6984–6994.

Yang, K., Swope, A., Gu, A., Chalamala, R., Song, P., Yu, S., Godil, S., Prenger, R. J., & Anandkumar, A. (2024). Leandojo: Theorem proving with retrieval-augmented language models. *Advances in Neural Information Processing Systems*, *36*.