UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

BRUNO ALVES DE OLIVEIRA

UM ESTUDO DOS PADRÕES DE PROMPT USADOS EM MODELOS GRANDES DE LINGUAGENS DURANTE A REALIZAÇÃO DE TAREFAS DE PROGRAMAÇÃO E COMPREENSÃO DE CÓDIGO

CAMPO MOURÃO

2025

BRUNO ALVES DE OLIVEIRA

UM ESTUDO DOS PADRÕES DE PROMPT USADOS EM MODELOS GRANDES DE LINGUAGENS DURANTE A REALIZAÇÃO DE TAREFAS DE PROGRAMAÇÃO E COMPREENSÃO DE CÓDIGO

A study of prompt patterns used in Large Language Models during programming and code comprehension tasks

Dissertação de Mestrado apresentado como requisito para obtenção do título de Mestrado em Ciência da Computação do Programa de Pós-Graduação em Ciência da Computação da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Igor Scaliante Wiese

CAMPO MOURÃO

2025

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Campo Mourão

BRUNO ALVES DE OLIVEIRA

UM ESTUDO DOS PADRÕES DE PROMPT USADOS EM MODELOS GRANDES DE LINGUAGENS DURANTE A REALIZAÇÃO DE TAREFAS DE PROGRAMAÇÃO E COMPREENSÃO DE CÓDIGO

Trabalho de pesquisa de mestrado apresentado como requisito para obtenção do título de Mestre Em Ciência Da Computação da Universidade Tecnológica Federal do Paraná (UTFPR). Área de concentração: Metodologia E Técnicas Da Computação.

Data de aprovação: 16 de Dezembro de 2025

Dr. Igor Scaliante Wiese, Doutorado - Universidade Tecnológica Federal do Paraná

Dr. Ivanilton Polato, Doutorado - Universidade Tecnológica Federal do Paraná

Dra. Mairieli Santos Wessel, Doutorado - Radboud University Nijmegen

Documento gerado pelo Sistema Acadêmico da UTFPR a partir dos dados da Ata de Defesa em 16/12/2025.

À Daniele Lemos Marinho, minha esposa e companheira de vida.

AGRADECIMENTOS

Agradeço ao meu orientador, Dr. Igor Scaliante Wiese, pela orientação segura, pela paciência e pelo compartilhamento de seu vasto conhecimento, que foram fundamentais para o amadurecimento desta pesquisa.

Aos membros da banca examinadora, Prof. Dr. Ivanilton Polato e Profa. Dra. Mairieli Santos Wessel, pelas valiosas contribuições, críticas e sugestões que certamente enriqueceram a versão final deste trabalho.

De forma especial, agradeço à minha esposa, Daniele Lemos Marinho, pelo companheirismo incondicional e pela compreensão nos dias de profunda dedicação; sua presença foi o meu maior incentivo. À minha mãe, por ser o alicerce de toda a minha trajetória, pelo apoio constante e por sempre acreditar no valor da minha educação. Sem o apoio de vocês, esta conquista não seria possível.

RESUMO

A popularização dos Modelos de Linguagem de Grande Escala (LLMs) tem transformado a engenharia de software, permitindo que programadores de diversos níveis obtenham suporte em tarefas de compreensão e implementação de código. O objetivo principal desta pesquisa foi compreender como diferentes perfis de desenvolvedores interagem com essas ferramentas, investigando a influência da experiência e do estilo cognitivo (mensurado pelo método GenderMag) na construção de prompts e nas estratégias de resolução de problemas. A metodologia consistiu em um estudo empírico com estudantes e profissionais de diferentes nacionalidades, totalizando 27 participantes. Inicialmente, os participantes responderam a questionários sociodemográficos e de perfil cognitivo para classificação nas personas GenderMag (Abi e Tim). Em seguida, realizaram seis tarefas de desenvolvimento e correção de bugs em uma cópia do projeto TodoMVC, com tempo limite de 40 minutos, utilizando o Copilot livremente. A análise dos dados foi mista, avaliando aspectos quantitativos (métricas linguísticas e contagem de interações) e qualitativos (classificação dos tipos de prompt). Os resultados indicaram que o esforço linguístico na elaboração dos prompts é adaptativo: tarefas mais complexas resultam em maior verbosidade, embora a linguagem permaneça acessível, com nível escolar estimado entre a 8ª e a 9ª série. Qualitativamente, a categoria de prompt predominante foi a "Pergunta Aberta", sugerindo que os usuários utilizam a LLM principalmente como um tutor para compreender o contexto do código, e não apenas como gerador de soluções. A análise comparativa revelou uma associação estatisticamente significativa entre o perfil e o tipo de interação: usuários com perfil TIM tendem a ser mais diretos e concisos, enquanto usuários ABI são mais verbosos. Além disso, a experiência atuou como fator moderador, onde novatos do perfil Abi recorreram a prompts excessivamente longos para compensar a falta de vocabulário técnico, enquanto profissionais experientes desse mesmo perfil superaram a aversão ao risco esperada, adotando instruções mais objetivas. Conclui-se que a interação com assistentes de programação baseados em Inteligência Artificial não se apresenta de maneira uniforme, sendo significativamente influenciada pela combinação entre o estilo cognitivo e o nível de experiência dos usuários. O estudo aponta a necessidade de ferramentas de LLM mais inclusivas, capazes de auxiliar na síntese de intenção para usuários verbosos, e destaca a importância de os desenvolvedores aprimorarem seu vocabulário técnico para uma comunicação mais eficiente com os agentes inteligentes.

Palavras-chaves: LLM, IA, desenvolvedores de software, Copilot, classificação de prompts, GenderMag

ABSTRACT

The popularization of Large Language Models (LLMs) has transformed software engineering, enabling programmers of varying proficiency levels to obtain support in code comprehension and implementation tasks. The primary objective of this research was to understand how different developer profiles interact with these tools, investigating the influence of experience and cognitive style (measured by the GenderMag method) on prompt construction and problem-solving strategies. The methodology consisted of an empirical study involving 27 participants, comprising both students and professionals from various nationalities. Initially, participants completed sociodemographic and cognitive profile questionnaires to be classified into GenderMag personas (Abi and Tim). Subsequently, they performed six development and debugging tasks on a fork of the TodoMVC project within a 40-minute time limit, utilizing GitHub Copilot freely. A mixed-methods data analysis was employed, evaluating quantitative aspects (linguistic metrics and interaction counts) and qualitative aspects (prompt type classification). The results indicated that linguistic effort in prompt formulation is adaptive: more complex tasks result in greater verbosity, although the language remains accessible, with an estimated reading level between the 8th and 9th grades. Qualitatively, the predominant prompt category was the "Open Question,"suggesting that users utilize the LLM primarily as a tutor to understand code context rather than merely as a solution generator. Comparative analysis revealed a statistically significant association between the user profile and the type of interaction: TIM profile users tend to be more direct and concise, whereas ABI users are more verbose. Furthermore, experience acted as a moderating factor: ABI novices resorted to excessively long prompts to compensate for a lack of technical vocabulary, while experienced professionals of the same profile overcame expected risk aversion, adopting more objective instructions. It is concluded that interaction with AI-based programming assistants is not uniform, being significantly influenced by the combination of cognitive style and the users' experience level. This study highlights the need for more inclusive LLM tools capable of assisting in intent synthesis for verbose users and emphasizes the importance of developers enhancing their technical vocabulary for more efficient communication with intelligent agents.

Palavras-chaves: LLM, IA, software developers, Copilot, prompt classification

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

A forma como os programadores estão resolvendo tarefas no dia a dia está mudando rapidamente com a popularidade dos modelos grandes de linguagem (LLMs). Programadores de diferentes níveis de conhecimento e experiência têm tido bons resultados nas soluções de tarefas de compreensão de código, soluções de *bugs*, tarefas de autocompletar códigos, depuração e melhoria de segurança de software com a ajuda de LLMs. É amplamente reconhecido que programadores com maior conhecimento e experiência tendem a obter resultados superiores ao utilizar essas ferramentas. (PINTO et al., 2023; NAM et al., 2024).

A Inteligência artificial generativa tem sido usada para gerar respostas textuais que imitam uma conversa humana, incluindo a capacidade de responder a perguntas subsequentes com base no contexto anterior, como visto no trabalho de Tayeb et al. (2024). Na engenharia de software, diversos trabalhos têm discutido os desafios relacionados ao uso de LLMs para apoiar tarefas desenvolvidas por engenheiros de software, como por exemplo, o trabalho de Oliveira (2024), focado em testes, qualidade do código gerado e legibilidade de código por Borg et al. (2025), segurança e correção de bugs, mencionado por Macháček et al. (2025), entre outras atividades.

Neste contexto, sabe-se que o desenvolvimento de "*prompts*" é considerado um elemento fundamental na eficácia dos resultados ao utilizar modelos de linguagem. Trabalhos como o de Mondal et al. (2024) mostram que falhas no design de prompts levam a múltiplas interações para chegar a uma solução desejada. Além disso, o estudo de White et al. (2023) reforça a importância de documentar e adotar padrões reutilizáveis, sugerindo que a experiência em criar prompts pode influenciar diretamente a qualidade e a eficiência das respostas obtidas. Considerando esse impacto, é esperado que desenvolvedores experientes e novatos interajam de maneira diferente com os LLMs durante a realização de uma tarefa e que desenvolvedores experientes encontrem mais facilidade neste ajuste de prompts para alcançar soluções mais ágeis e precisas graças ao seu maior entendimento tanto do problema quanto do comportamento do modelo. (WU et al., 2024; CHAMPA et al., 2024).

Apesar dos esforços realizados pelos pesquisadores, existem poucos estudos voltados para entender o impacto do perfil cognitivo dos indivíduos e da sua experiência durante a interação e o uso de inteligência artificial generativa na realização de tarefas de desenvolvimento de software. Nesse sentido, a análise do estilo cognitivo combinada com a experiência no uso de LLMs pode fundamentar a personalização de futuras interfaces de programação. Compreender essas facetas permite que as ferramentas auxiliem o desenvolvedor para uma interação mais eficaz e adaptada ao seu perfil individual, otimizando o fluxo de trabalho.

Assim, o objetivo desta pesquisa é entender como diferentes perfis de pessoas desenvolvedoras utilizam modelos grandes de linguagem durante a realização de tarefas de programação e compreensão de código, procurando entender se a experiência e o perfil cognitivo impactam na escrita do *prompt* que é usado para interagir com assistentes de IA, por exemplo o *GitHub Copilot*.

Para realizar esta análise, três questões de pesquisa são investigadas: **QP1** - Como varia o nível de sofisticação linguística e o esforço de elaboração dos *prompts* durante a interação com LLMs em tarefas de programação?; **QP2** - De que maneira os participantes estruturam suas solicitações à LLM: via questionamento aberto, verificação de hipóteses, instruções diretas ou por comandos implícitos ?; **QP3** - Como se comparam os *prompts* de pessoas com diferentes perfis?.

Em resumo conseguimos identificar que a sofisticação linguística (**QP1**), permanece inalterada, acessível e apenas ligeiramente ajustada ao contexto técnico. Métricas de escolaridade, como *Flesch-Kincaid* e *Gunning Fog*, sugerem um nível apropriado para estudantes da 8ª à 9ª série, evidenciando que a estrutura das frases permanece simples, mesmo em tarefas de programação. A maioria dos prompts é de fácil leitura, apesar de a clareza diminuir nas tarefas que exigem manipulação de código (2-A e 2-C). Essa diminuição não é resultado de maior complexidade linguística, mas da inclusão de trechos de código e jargões técnicos, que impactam as métricas de legibilidade.

Concluímos também que a categoria de prompt (**QP2**) mais empregada foi a de Pergunta Aberta, presente em quase todas as atividades. Isso sugere que os participantes utilizaram o LLM principalmente como suporte para investigar e entender a base de código, e não somente para a geração automática de código. A predominância desse tipo de prompt indica que, na ausência de familiaridade com o código, a principal abordagem é compreender o contexto.

Em relação à comparação entre perfis (**QP3**), a análise estatística indicou uma associação significativa entre o grupo do participante e o tipo de prompt empregado (p=0,0309). Isso confirma que a interação é influenciada pela combinação de estilo cognitivo e experiência. Notou-se que usuários TIM costumam elaborar *prompts* mais diretos e concisos (com uma média de 13 palavras), enquanto usuários ABI tendem a ser mais verbosos (com uma média de 20 palavras). A experiência desempenhou um papel moderador significativo: o grupo ABI - Experiente superou a aversão ao risco prevista para sua persona, empregando mais instruções diretas do que o modelo estatístico antecipava. Por outro lado, os novatos do perfil ABI adotaram uma estratégia compensatória, criando *prompts* excessivamente longos e complexos — classificados no agrupamento "Instrução + Implícito"— para tentar compensar a falta de vocabulário técnico específico. Por fim, o perfil TIM - Experiente se mostrou o mais equilibrado, variando entre investigação e instrução de acordo com a necessidade.

O restante do trabalho é organizado da seguinte forma: o capítulo 2 mostra a fundamentação teórica. O capítulo 3 traz os trabalhos relacionados. O capítulo 4 apresenta os passos da metodologia empregada para coleta e análise dos dados desta pesquisa. Resultados encontrados estão no capítulo 5. Discussões sobre o estudo são abordados no capítulo 6. Por fim o capítulo 7 dedica-se à apresentação das conclusões e das possíveis implicações.

2 REFERENCIAL TEÓRICO

2.1 Modelos de linguagem

Os Modelos de Linguagem de Grande Escala (LLMs) são sistemas de aprendizado de máquina criados para interpretar e gerar texto de forma semelhante à linguagem humana. Para que esses modelos possam responder com precisão a perguntas e executar tarefas específicas, eles precisam ser treinados em grandes volumes de dados textuais.

O desenvolvimento desses modelos tem avançado significativamente ao longo do tempo, com grandes marcos transformando a maneira como os agentes de inteligência artificial entendem e produzem linguagem natural. O início desses progressos está ligado às redes neurais e ao desenvolvimento de representações vetoriais contínuas para palavras, chamadas de word embeddings. Trabalhos como o Word2Vec (MIKOLOV et al., 2013) e GloVe (PENNINGTON et al., 2014) desempenharam um papel fundamental ao criar e otimizar essas representações, propondo soluções que melhoraram o desempenho dos modelos.

Posteriormente, um avanço crucial foi alcançado com a introdução da arquitetura Transformer, proposta por Vaswani et al. (2023). Os transformadores revolucionaram o campo ao utilizarem operações matemáticas para identificar relações de contexto em sequências de entrada, resultando em saídas baseadas nessas conexões contextuais. Esse tipo de arquitetura permitiu o treinamento de modelos maiores e mais eficientes com grandes volumes de dados, usando menos recursos computacionais.

Seguindo essa linha de evolução, surgiram os modelos pré-treinados, como o BERT de Devlin et al. (2019) e o GPT de Yenduri et al. (2023), que são inicialmente treinados com uma vasta quantidade de dados e, posteriormente, ajustados para tarefas específicas por meio de técnicas como *Fine-tuning*. Até o presente, esses modelos têm se destacado como fundamentais para o avanço das inteligências artificiais.

As pesquisas no campo continuam a evoluir, e novos modelos estão constantemente sendo desenvolvidos, trazendo melhorias no aprendizado e na geração de linguagem. Um exemplo significativo é o GPT-3 de Brown et al. (2020), com seus impressionantes 175 bilhões de parâmetros. Ao mesmo tempo, há esforços contínuos para otimizar as várias etapas envolvidas no desenvolvimento dos LLMs.

Empresas como OpenAI, Microsoft e Google têm liderado o desenvolvimento de modelos mais eficientes, treinando-os com grandes volumes de dados e disponibilizando-os para o público. Ferramentas como o ChatGPT da OpenAI e o BERT da Google foram pioneiras ao oferecer modelos pré-treinados com interfaces simples, facilitando o uso de LLMs em diversas aplicações e áreas de conhecimento.

O modelo BERT, por exemplo, fundamenta-se no trabalho de Vaswani et al. (2023). Ele destaca-se pela capacidade de análise bidirecional, a qual permite uma compreensão mais profunda das relações contextuais do texto, superando modelos anteriores limitados a uma análise unidirecional.

O GitHub Copilot utiliza uma combinação de modelos de linguagem da série GPT do OpenAI. Este modelo evoluiu a partir da arquitetura *Transformer* e é amplamente conhecido pela sua capacidade de gerar texto a partir de vastos conjuntos de dados. Lançado em novembro de 2022, o ChatGPT é uma ferramenta de IA que permite interações simples através de uma interface de chat, acessível via navegador web. Inicialmente baseado no GPT-3, o modelo foi aprimorado, e sua versão mais recente, o GPT-4o , está disponível atualmente para assinantes. A interação com o Copilot começa com a inserção de um texto ou consulta, denominado **prompt**. Este *prompt* serve como ponto de partida para o modelo gerar uma resposta. O processo segue as etapas abaixo:

- **Entrada de Dados:** O usuário fornece o prompt, que pode ser uma pergunta, uma instrução ou qualquer outra solicitação textual.
- **Processamento:** O Copilot, utilizando os conhecimentos adquiridos durante seu treinamento, analisa o prompt e prevê a sequência de palavras mais adequadas para continuar ou responder à solicitação, utilizando o contexto fornecido.
- **Geração de Resposta:** O modelo gera a resposta baseada em um processo de inferência probabilística, onde tenta prever as palavras ou frases que têm maior probabilidade de serem relevantes e coerentes.
- **Manutenção de Contexto:** Durante uma sessão de diálogo, o Copilot é capaz de reter o contexto das interações anteriores para gerar respostas consistentes ao longo da conversa. Isso permite que ele mantenha a coerência em trocas de mensagens mais longas.

Ao longo da evolução desses modelos, várias técnicas de treinamento foram desenvolvidas para aumentar a qualidade das respostas. Métodos como Fine-tuning, Few-shot Learning, Zero-shot Learning e RAG foram criados para aprimorar o treinamento e permitir que os LLMs forneçam respostas mais precisas e alinhadas ao contexto.

2.2 Retrieval-Augmented Generation

A técnica foca em processo de geração das respostas pelo modelo e não no treinamento. Combina métodos de recuperação de informações para aplicá-las a um contexto enviado juntamente ao *input* de um modelo pré-treinado. Essa técnica emprega este contexto para conduzir análises e formular respostas mais precisas às solicitações.

2.3 Few-shot learning

Utilizando uma amostra pequena para realizar o treinamento, o modelo é levado a aprender extraindo o máximo de informações e classificações de uma quantidade restrita de dados. Essa técnica simplifica os processos de treinamento tradicionais que necessitam de uma carga expressiva de dados.

2.4 Fine-tuning

O ajuste fino (*fine-tuning*) é executado em arquiteturas pré-treinadas, uma técnica comumente utilizada em situações de alta especificidade ou quando há insuficiência de métodos tradicionais. O processo inclui um treinamento supervisionado extra, com a inserção de dados rotulados, com o objetivo de classificar e utilizar em tarefas futuras. Essa abordagem possibilita a exploração de domínios externos ao treinamento inicial, melhorando a precisão das respostas. Ao ser combinada com o *Few-shot Learning*, a técnica reduz a demanda por grandes quantidades de dados, possibilitando que o modelo atenda a pedidos em áreas ainda não exploradas com um número menor de exemplos.

2.5 Zero-shot learning

Essa abordagem possibilita que o modelo identifique e classifique elementos, conceitos ou ações que não foram incluídos em seu treinamento original. Dessa forma, ele não fica limitado aos dados e classes previamente rotulados, mas utiliza descrições ou representações semânticas para reconhecer classes desconhecidas.

2.6 Prompt

White et al. (2023) definem *prompt* como um conjunto de instruções fornecidas a um LLM para programá-lo, permitindo a personalização, o aprimoramento ou o refinamento de suas habilidades. O *prompt* pode impactar as interações subsequentes com o LLM e o material produzido, definindo normas e orientações específicas para um diálogo. Por exemplo, um prompt estabelece o cenário da interação e indica ao LLM quais informações são relevantes, além de especificar a forma e o conteúdo que se deseja obter como resultado. Por exemplo, um prompt pode definir que o LLM só deve produzir código que adote um estilo de programação ou paradigma específico. Similarmente, é possível solicitar que o LLM realce determinadas palavras-chave ou frases em um documento produzido e forneça informações extras relacionadas a esses termos. Ao estabelecer essas orientações, os prompts simplificam o processo.

2.7 Engenharia de prompt

A implementação de modelos generativos pré-treinados, como o GPT-4, desencadeou um fenômeno denominado engenharia de prompt (*prompt engineering*), no qual os usuários de modelos escrevem e revisam repetidamente *prompts* na tentativa de realizar uma tarefa. (LIANG et al., 2024) .

A engenharia de prompts é o meio pelo qual os LLMs são programados por meio de prompts. Um bom exemplo do potencial da engenharia de prompt e mostrado no trabalho de (WHITE et al., 2023).

- **Prompt**: "*De agora em diante, gostaria que você me fizesse perguntas para implantar uma aplicação Python na AWS. Quando tiver informações suficientes para realizar a implantação, crie um script em Python para automatizá-la.*"

Este prompt permite que o ChatGPT inicie questionamentos ao usuário acerca da sua aplicação de software. O ChatGPT seguirá guiando o processo de questões até reunir dados suficientes para criar um script em Python que automatize a implementação. Este exemplo mostra a capacidade de programação dos prompts, ultrapassando o padrão de prompts como "crie um método que realize..."ou "responda a esta pergunta do quiz".

2.8 Padrões de prompts

No âmbito das interações entre programadores e LLM, Ekin (2023) define que os padrões de *prompts* constituem estruturas essenciais que direcionam o fluxo e a efetividade dessas interações. Estes padrões funcionam como modelos para elaborar consultas de maneira a maximizar a relevância e eficácia das respostas produzidas pelo modelo (WU et al., 2024).

- **Entrada Semântica:** Está principalmente relacionada ao padrão de criação de linguagem, no qual os utilizadores criam *prompts* como "Sempre que eu disser A, faça B"para direcionar os LLM. Apesar de auxiliar na interpretação contextual, existe a possibilidade de ambiguidade semântica e uma possível interpretação incorreta das instruções pelos LLMs.
- **Saída Personalizada:** Esta categoria define o resultado dos LLMs, definindo tipos, formatos e estruturas. Este padrão orienta o modelo a seguir as etapas propostas, minimizando solicitações repetitivas do usuário. O modelo é orientado a assumir um papel ou personagem particular para solucionar problemas.
- **Identificação de Erros:** Os LLMs abordam erros em suas saídas, abrangendo dois padrões principais. O padrão "lista de verificação de fatos"é usado para mitigar a tendência dos LLMs de gerar saídas imprecisas ou fabricadas, orientando-o a validar dados quanto à autenticidade. Alternativamente, para questões de dedução teórica sem verificação factual, os *prompts* podem solicitar explicações detalhadas dos LLMs sobre suas conclusões, permitindo que os usuários avaliem possíveis erros no raciocínio. Essa abordagem é classificada como "padrão de reflexão".
- **Melhoria de Prompt:** Esta categoria visa melhorar a qualidade do prompt, aprimorando assim a qualidade das saídas dos LLMs. Este padrão incentiva os LLMs a gerar prompts melhores e mais abrangentes, o que pode melhorar o desempenho das tarefas dos LLMs.
- **Interação**: Esta categoria melhora a interatividade com os LLMs. O padrão interação invertida permite que os LLMs façam perguntas durante a resolução de problemas. O padrão de jogo estabelece um ambiente de jogo para as reações dos LLMs. Em tarefas repetitivas, o padrão de geração infinita possibilita que os LLMs prossigam até serem interrompidos, diminuindo a demanda por interrupções repetidas.
- **Controle de Contexto:** Esta categoria trata do problema dos LLMs incluírem detalhes irrelevantes ou enfatizarem aspectos indesejados nas respostas. O padrão gerenciamento

de contexto direciona os LLMs a focar em áreas específicas, usando diretivas como “considere apenas”, “ignore”, etc., para refinar o escopo de sua resposta.

2.9 GenderMag

Trabalhos como de Santos et al. (2023) mostram que diferenças individuais nos estilos cognitivos, referidos como facetas, estão associadas a gêneros. O método *GenderMag* encapsula essas facetas em personas — Abi, Pat e Tim representam extremos opostos dos valores das facetas. A persona Abi está alinhada a valores de facetas que as mulheres tendem a preferir, enquanto Tim incorpora valores de facetas tipicamente preferidos por homens. A persona Pat inclui uma combinação desses valores de facetas.

Ao todo o GenderMag possui cinco facetas: (i) **Motivação**: Usuários Abi são motivados pelo que a tecnologia lhes permite alcançar, enquanto usuários Tim tendem a ser motivados pelo prazer de utilizar a própria tecnologia. (BURNETT et al., 2011) ; (ii) **Estilos de processamento de informação**: Abis processam novas informações de forma abrangente — reunindo informações bastante completas antes de prosseguir — enquanto Tims utilizam processamento seletivo de informações, seguindo a primeira informação promissora e voltando atrás, se necessário (RIEDL et al., 2010; MEYERS-LEVY; LOKEN, 2015); (iii) **Autoeficácia em computação**: Relaciona-se à confiança de uma pessoa em ter sucesso em uma tarefa específica, o que influencia o uso de estratégias cognitivas, persistência e formas de lidar com obstáculos. Abis apresentam uma autoeficácia em computação mais baixa em comparação com seus pares; (iv) **Aversão ao risco**: Abis são mais avessos a riscos ao experimentar novos recursos, em comparação com Tims (DOHMEN et al., 2011; CHARNESS; GNEEZY, 2012), o que afeta suas decisões sobre quais conjuntos de funcionalidades utilizar; e (v) **Aprendizado: por Processo vs. por Experimentação**: Abis preferem um aprendizado orientado por processos, enquanto Tims gostam de experimentar de forma lúdica ("brincar") com recursos de software novos para eles (BECKWITH et al., 2006; CAO et al., 2010; BURNETT et al., 2010).

3 TRABALHOS RELACIONADOS

Em se tratando de interação com LLMs, o design de *prompts* tem um impacto crucial na eficácia dos resultados. A seção abaixo mostra resultados obtidos em trabalhos que buscaram entender melhor a interação do desenvolvedor com a LLMs.

Tarefas repetitivas ou com soluções aproximadas podem ser resolvidas utilizando o mesmo design de *prompt*. O artigo de White et al. (2023) mostra a importância de se adotar e documentar padrões de *prompts*. Em seu trabalho, os autores apresentam um catálogo de padrões de *prompt*, sendo aplicados para resolver problemas comuns no domínio da interação conversacional com LLMs e da geração de resultados para automação de tarefas de *software*. Esses padrões de *prompt* são análogos aos padrões de *software* e têm o objetivo de fornecer soluções reutilizáveis para problemas enfrentados pelos usuários ao interagir com LLMs.

Na pesquisa de Mondal et al. (2024), identificou-se que *prompts* com alguma falha de design (especificações ausentes, respostas verbosas, contexto ausente) geram múltiplas interações para se chegar ao resultado desejado. Ao identificar essas lacunas no design e resolvê-las, foi possível chegar à mesma resposta com apenas uma interação. Desta forma, fica evidente que a qualidade dos *prompts* fornecidos a agentes de inteligencia artificial é fundamental para que se tenha resultados com precisão. Os autores Wu et al. (2024) analisaram interações de programadores do repositório DevGPT, apresentado por Xiao et al. (2024), com o objetivo central de identificar padrões de *prompts* que levam à resolução eficaz de problemas. Na análise dos dados, este trabalho observou que os programadores muitas vezes fazem muitas perguntas e vão aperfeiçoando os pedidos para ir melhorando as respostas do ChatGPT. Outro dado importante é que em todas as conversas analisadas pelo trabalho, o padrão de personalização de saída foi o mais utilizado. O estudo também observou que quando o assunto é desenvolvimento de *software*, as questões predominantes incluem a geração de código, otimização e análise de dados.

O trabalho de Liang et al. (2024), traz contribuições interessantes sobre como algumas formas de *prompts* são programas e que o desenvolvimento de *prompt* é um fenômeno distinto dentro da programação, denominando esse fenômeno como *prompt programming*. Neste trabalho, os pesquisados realizaram um estudo qualitativo envolvendo 20 desenvolvedores em uma variedade de contextos, modelos, domínios e níveis de complexidade de *prompts*. Neste estudo, os autores mostram 14 observações sobre o *prompt programming*. Por exemplo, em vez de construir modelos mentais de código, os programadores de *prompts* desenvolvem modelos mentais do comportamento do modelo de linguagem (*FM*, do inglês *Foundation Model*) em relação ao *prompt* e às suas qualidades únicas por meio da interação com o modelo. Enfatiza também que programadores de *prompts* que já desenvolveram dezenas de *prompts*, cada um com várias iterações, ainda enfrentam dificuldades para criar modelos mentais confiáveis. Isso contribui para um processo de desenvolvimento rápido e não sistemático.

Outro trabalho importante a ser mencionado é a pesquisa de Champa et al. (2024), onde foram analisados 2.865 conversas reais de desenvolvedores com o ChatGPT contidas no conjunto de dados do DevGPT, Xiao et al. (2024). Este estudo qualitativo mostra que, entre as 12 categorias de tarefas analisadas, a gestão da qualidade do código e a resolução de problemas em *commits* são os tipos de tarefas em que os desenvolvedores mais buscam auxílio do ChatGPT, especialmente ao lidar com código em Python. Em todos os tipos de tarefas, a assistência para *scripts* de *shell* é a mais procurada no ChatGPT. Evidencia também que as conversas/colaborações entre desenvolvedores e o ChatGPT são menos eficientes ao lidar com tarefas relacionadas à configuração do ambiente de desenvolvimento, geração de documentação e gestão de qualidade de código.

Em um estudo empírico recente, feito por Grewal et al. (2024), é visto que, em média, 54% das linhas geradas pelo ChatGPT são encontradas em projetos do *GitHub* e que essas linhas de código permanecem, em média, 89 dias inalteradas após serem adicionadas. Outras descobertas citadas nos trabalhos são: 18% dos trechos de código alterados em um *commit* subsequente foram excluídos, enquanto os outros 82% foram modificados. Alterações de impacto mínimo na funcionalidade representam cerca de 36%, reorganização de código 33% e alterações de nomes de variáveis, classificadas pelos autores como refinamento de nomes, ficam em 13%. Destaca-se ainda que as modificações em códigos gerados pelo ChatGPT, realizadas em *commits* subsequentes, tendem a ocorrer em um intervalo inferior a 24 horas após a submissão inicial. Esse estudo analisou 3.044 trechos de códigos publicados no dataset DevGPT (XIAO et al., 2024).

Em resumo, os trabalhos citados discutem diversas abordagens e desafios no que envolvem a criação de *prompts*, e como ocorre essa interação entre desenvolvedor e modelos de linguagem. Entretanto, ainda são necessários estudos mais aprofundados para compreender melhor como desenvolvedores formulam seus *prompts* e verificar se é possível comparar adequadamente aqueles produzidos por perfis distintos.

4 MÉTODO DE PESQUISA

Este estudo visa investigar a interação de desenvolvedores com Grandes Modelos de Linguagem (LLMs) durante a resolução de tarefas de programação. O foco central consiste em analisar se a experiência técnica e o estilo cognitivo — avaliados pelas facetas do GenderMag conforme abordado por Santos et al. (2023) — influenciam a elaboração de *prompts* em assistentes de codificação baseados em IA, exemplificados pelo *GitHub Copilot*.

Para tanto, a metodologia a seguir descreve o cenário do estudo, as tarefas de programação e de compreensão de código, os procedimentos de coleta de dados das interações com o *Copilot* e as técnicas de análise adotadas.

4.1 Recrutamento e coleta de dados do perfil dos desenvolvedores

Para a realização deste trabalho, é importante coletar dados sobre a experiência e o perfil dos participantes.

No total, 27 participantes, que trabalham na indústria ou são estudantes de diferentes anos de graduação, participaram do estudo. A estratégia para o recrutamento dos participantes foi realizada por meio de (i) apresentações curtas em aulas, (ii) grupos de mensagens de programas de estudo, (iii) contatos pessoais e profissionais, (iv) divulgações em redes sociais, como o linkedin, Facebook e Twitter.

Todos os participantes foram voluntários e assinaram um termo de consentimento antes das sessões (apêndice A. O único pré-requisito é ter experiência básica em programação (por exemplo, ter feito um curso de programação ou programar como hobby).

Antes da realização do experimento, cada participante respondeu a dois questionários. O primeiro instrumento consistiu em um questionário sociodemográfico para coleta de dados pessoais. Por exemplo, neste questionário, perguntamos aos participantes: *Qual é o seu nome? Qual é o seu endereço de email? Escolaridade? Atualmente, sua graduação na área de programação encontra-se em?Como você avalia sua proficiência em programação geral:programação em JavaScript? Desenvolvimento web?; React? Visual Studio Code?.* O questionário completo pode ser conferido no **apêndice B**. O segundo teve como objetivo mapear o estilo cognitivo dos participantes, utilizando a escala GenderMag, Burnett et al. (2010). Esse método utiliza 5 características de diferença de gênero, chamadas de facetas. Estes dados foram obtidos por meio do questionário sobre estilos cognitivos, disponível no **apêndice C**.

Essas 5 facetas foram encapsuladas em personas, mencionadas no trabalho de Vorvoreanu et al. (2019): Timothy / Timara, Patricia / Patrick e Abigail / Abishek, resumidas nesse trabalho para Tim e Abi:

- **Abi**: Prefere interações que forneçam explicações detalhadas e promovam um estilo exploratório, como "Explique a lógica deste código em detalhes."
- **Tim:** Prefere que as interações sejam mais rápidas e objetivas, como "Encontre o erro neste trecho de código."

Para determinar o perfil correspondente à persona (Abi ou Tim), cada participante respondeu o questionário de aspectos do GenderMag, Hamid et al. (2024). O questionário de aspectos do GenderMag, (Figura 4.1), validado por Aguiar et al. (2011), é um instrumento baseado na escala *Likert* que coleta os valores específicos de cada respondente para os cinco aspectos: *Autoeficácia*; *Motivação*; *Estilo de aprendizagem*; *Processamento de Informação* e *Atitude em relação ao risco*. Este questionário também oferece uma avaliação dos resultados de diversidade em uma resolução mais alta do que as medidas demográficas padrão (HAMID et al., 2024).

Após os participantes responderem ao questionário de aspectos, pontuamos as respostas usando a mesma chave de pontuação do trabalho de Hamid et al. (2024), ilustrado na Figura 4.2. Abaixo, mostramos, com mais detalhes, como a pontuação será utilizada.

- **Passo 1 (Complemento):** Converter as respostas em pontuações numéricas de 1 (Discordo completamente) a 9 (Concordo completamente). Para algumas perguntas, valores mais próximos de 9 indicam maior similaridade com Tim. Porém, o contrário é verdadeiro para as perguntas 2 e para as de 9 a 13. Para essas perguntas, inverta as respostas dos participantes para o complemento em base 10 (ou seja, converta “9” para “1”, “8” para “2” e assim por diante).
- **Passo 2 (Soma de cada aspecto):** Para cada participante, serão somados os resultados do Passo 1 de cada aspecto. Este passo gera 5 pontuações por participante, uma para cada aspecto.
- **Passo 3 (Calcular medianas dos aspectos):** Para identificar o valor mediano do grupo de pares (assumindo que o grupo de pares corresponde aos participantes recrutados para o estudo), calculamos a mediana da soma das pontuações de todos os participantes do mesmo grupo de pares, para cada aspecto.
- **Passo 4 (Marcar a pontuação de cada participante para cada aspecto):** À direita da mediana (acima) é considerado "Tim", caso contrário,"Abi".

Ao final, cada participante terá uma marcação de 5 elementos, cada um representando um aspecto.

4.2 Tarefas

Foi utilizado o fork do projeto todoMvc[1] como base do código. Este projeto utiliza a linguagem de programação JavaScript, mais especificamente *React* com *JSX*. Modificamos o projeto para torná-lo mais simples e inserimos alguns defeitos, como erros de importação nas classes e implementações incompletas de funcionalidades. A Figura 4.3 ilustra a aplicação utilizada no experimento.

Ao todo, cada participante deveria realizar 6 tarefas. A cada tarefa, o grau de complexidade aumenta. Abaixo estão detalhados as tarefas e os erros incluídos no projeto.

[1] (http://todomvc.com)

Autoeficácia

1. Sou capaz de usar <insira o tipo de tecnologia> quando…
- a. Eu tenho apenas a ajuda embutida para me auxiliar.
- b. Já vi outra pessoa usando antes de tentar por conta própria.
- c. Não há ninguém por perto para me ajudar, caso eu precise.
- d. Alguém me ajudou a começar.
- e . Alguém me mostrou como fazer primeiro.
- f. Já usei uma tecnologia semelhante para realizar a mesma tarefa.
- g. Nunca usei nada parecido antes.

2. Não me sinto confiante em minha capacidade de usar e aprender <insira o tipo de tecnologia>. Tenho outras habilidades.

Motivação

3. Dedico tempo para explorar <insira o tipo de tecnologia> que não é essencial para o meu trabalho.

4. Um dos motivos pelos quais gasto tempo e dinheiro em <insira o tipo de tecnologia> é porque isso é uma maneira de me destacar entre os colegas.

5. É divertido experimentar novas <insira o tipo de tecnologia> que ainda não estão disponíveis para todos, como participar de programas beta para testar tecnologias inacabadas.

Estilo de aprendizagem

6. Gosto de descobrir recursos e funcionalidades menos conhecidos da <insira o tipo de tecnologia> que uso.

7. Exploro áreas de <insira o tipo de tecnologia> antes que seja necessário usá-las.

8. Nunca fico satisfeito com as configurações padrão da minha <insira o tipo de tecnologia>; eu as personalizo.

Processamento de Informação

9. Quero fazer tudo certo na primeira vez, então, antes de decidir como agir, reúno o máximo de informações possível.

10. Sempre faço pesquisas extensivas e comparações antes de realizar compras importantes.

11. Quando preciso tomar uma decisão, é importante para mim reunir detalhes relevantes antes de decidir, para garantir que estamos seguindo na direção certa.

Atitude em relação ao risco

12. Evito botões ou seções "avançados" em <insira o tipo de tecnologia>.

13. Evito atividades que sejam perigosas ou arriscadas.

14. Apesar dos riscos, utilizo funcionalidades em <insira o tipo de tecnologia> que ainda não foram comprovadas como eficazes.

Concordo completamente			Nem Concordo, Nem discordo				Discordo completamente	
1	2	3	4	5	6	7	8	9

Figura 4.1. O questionário de aspectos. (Parte superior) (Parte inferior) Todas as perguntas utilizam uma escala Likert de 9 pontos.

1a, 1b, 1c, 1d, 1e, 1f, 1g 2	Autoeficácia (Tim) Baixa Autoeficácia (Abi)
3, 4, 5	Motivações: Tecnologia pelo próprio valor (Tim)
6, 7, 8	Aprendizado: Explorador (Tim)
9, 10, 11	Processamento de Informação Abrangente (Abi)
12, 13 14	Aversão a risco (Abi) Não aversão a risco (Tim)

Figura 4.2. Chave do questionário de aspectos. Quanto mais fortemente o participante concordar com uma pergunta, mais próximo estará o valor do seu aspecto do extremo (nome da persona) indicado.

Figura 4.3. Aplicação todoMvc

4.2.1 Tarefa 1 - A - Alterar a cor

Quando não há afazeres, aparece uma caixa vermelha com a mensagem "Nenhum afazer aqui ainda". Altere a cor dessa caixa para azul (#007bff).

4.2.2 Tarefa 1 - B - Alterar o placeholder

O texto do 'placeholder' no campo de entrada atualmente exibe a pergunta "O que precisa ser feito?". Altere o texto para "Digite um a fazer aqui".

4.2.3 Tarefa 1 - C - Ordenar afazeres

Atualmente, os afazeres são exibidos na ordem em que foram adicionados. Modifique o aplicativo para que os afazeres sejam ordenados alfabeticamente.

4.2.4 Tarefa 2 - A - Salvar dados

Atualmente, todos os afazeres desaparecem quando a página é recarregada. Modifique a aplicação para que os afazeres permaneçam mesmo após um recarregamento. Certifique-se de que o estado de cada afazer (se está concluído ou ativo) também seja mantido. Uma tentativa de implementar essa funcionalidade foi feita, mas, no momento, ela não está funcionando corretamente. Corrija ou reimplemente a funcionalidade.

4.2.5 Tarefa 2 - B - Salvar rascunhos

Se um usuário recarregar a página enquanto estiver digitando um novo afazer, o texto no campo de entrada desaparece. Modifique a aplicação para que o campo de entrada salve automaticamente rascunhos a cada poucos segundos e os restaure ao recarregar a página.

4.2.6 Tarefa 2 - C - Paginação

Atualmente, a aplicação de Lista de Afazeres exibe todas as tarefas em uma única lista. Nesta tarefa, você implementará a paginação para limitar o número de afazeres exibidos por página, permitindo que os usuários naveguem facilmente entre as páginas.

4.2.7 Erros adicionados no projeto

- No arquivo main.jsx adicionado o erro de importação na linha 6;
- No arquivo reducer.js adicionado erro de nome de atributo no *switch*, linha 44, causando o não carregamento das tarefas salvas;
- No arquivo paginationControls.jsx, linha 13, deletada a condição do if que trata da função de retroceder à página;
- No arquivo input.js foi adicionada uma verificação para que o afazer só seja adicionada na lista se o seu nome contiver mais de 2 caracteres, linha 18.

4.3 Ambiente do experimento

A sessão do experimento consistiu em uma videochamada, na qual foi disponibilizado um ambiente controlado para a resolução das tarefas. Para cada participante foi marcada uma reunião no Microsoft Teams. Antes de cada sessão, foi configurado um ambiente de desenvolvimento na plataforma *CodeSpace*, na qual é disponibilizado um editor de código, o VSCode 4.4, muito conhecido na comunidade de desenvolvimento. O protocolo de configuração do ambiente seguiu os seguintes passos:

- Crie um codespace:
- Acesse (https://github.com/codespaces).
- Clique em ’New codespace’, selecione ‘iuryholiveira/copilot-study-todo‘;
- Escolha a branch ‘main‘ para sessões em inglês ou ‘main-br‘ para sessões;
- Entre no codespace;
- Execute o comando ‘code –install-extension telemetry.vsix‘;
- Abra o workspace ‘session.code-workspace‘ e verifique se a mensagem "Telemetry data is being logged."aparece;
- Instale as extensões recomendadas;
- Copie o link do Codespace e saia dele.

4.3.1 O Experimento

Antes do início de cada sessão, realizou-se a configuração do ambiente de desenvolvimento. Durante o processo, seguiu-se um protocolo de conduta, visando mitigar possíveis vieses no estudo. O protocolo compreende as seguintes etapas:

- Verificar se o participante concluiu a pesquisa sobre estilos cognitivos. Caso não tenha feito, enviar o link para o participante e pedir para concluir;
- Fornecer ao participante o link do Codespace e os detalhes de login do GitHub;
- O participante deve usar uma janela anônima do Google Chrome ou uma janela privada do Firefox;
- Pedir ao participante para desligar a câmera, caso esteja ligada;
- Gravar e transcrever a sessão pelo Microsoft Teams;
- Peça ao participante para (re)compartilhar a tela e ativar o microfone (compartilhamento de tela e microfone são desativados quando a gravação começa);
- Utilizar o texto de introdução da seção do experimento.

  > "*Olá, obrigado por participar deste estudo. Meu nome é …, sou um estudante de Mestrado/Doutorado da … Neste estudo, queremos entender como as pessoas usam o GitHub Copilot. Não apenas o que você usa e como, mas também o que você pensa sobre o Copilot. Para isso, precisamos que você diga em voz alta tudo o que está fazendo e pensando. Isso pode parecer um pouco estranho no início, mas é muito útil para nós! Talvez ajude a imaginar que sou um programador iniciante e que você está tentando me ensinar mostrando seu processo de pensamento. Você irá trabalhar em algumas tarefas. São três tarefas práticas para se familiarizar com a base de código e com o uso do Copilot e três tarefas reais. Cada tarefa tem vários requisitos, cada um com etapas de verificação para avaliar sua solução. Tente garantir que todas as etapas de verificação sejam concluídas antes de avançar. Você não precisa completar todas as tarefas — após 40 minutos, pediremos que pare de programar e preencha um questionário de pós-estudo. Você não é obrigado a usar o Copilot; utilize-o da forma que desejar.*"

- Enviar ao participante o PDF da primeira tarefa. Somente envie a próxima quando a atual for concluída;
- Na primeira tarefa, certifique-se de que o participante leu atentamente as instruções do Copilot. Confirme que ele sabe como abrir a janela de chat e como os comandos funcionam;
- Se o participante pular um requisito ou uma etapa de verificação, alertá-lo e pedir que complete antes de seguir;
- Advertir o participante a não utilizar a ferramenta chat em linha (*inline chat*) do VSCode.
- Após 40 minutos, pedir ao participante para parar;
- Pedir ao participante para salvar todas as conversas que teve com o Copilot;
- Enviar ao participante o link para a pesquisa pós-estudo;

4.4 Teste piloto

Antes da aplicação do experimento, foram realizados três testes-piloto. Os voluntários são graduados em ciências da computação com 2 a 5 anos de experiência em programação. No primeiro teste-piloto,

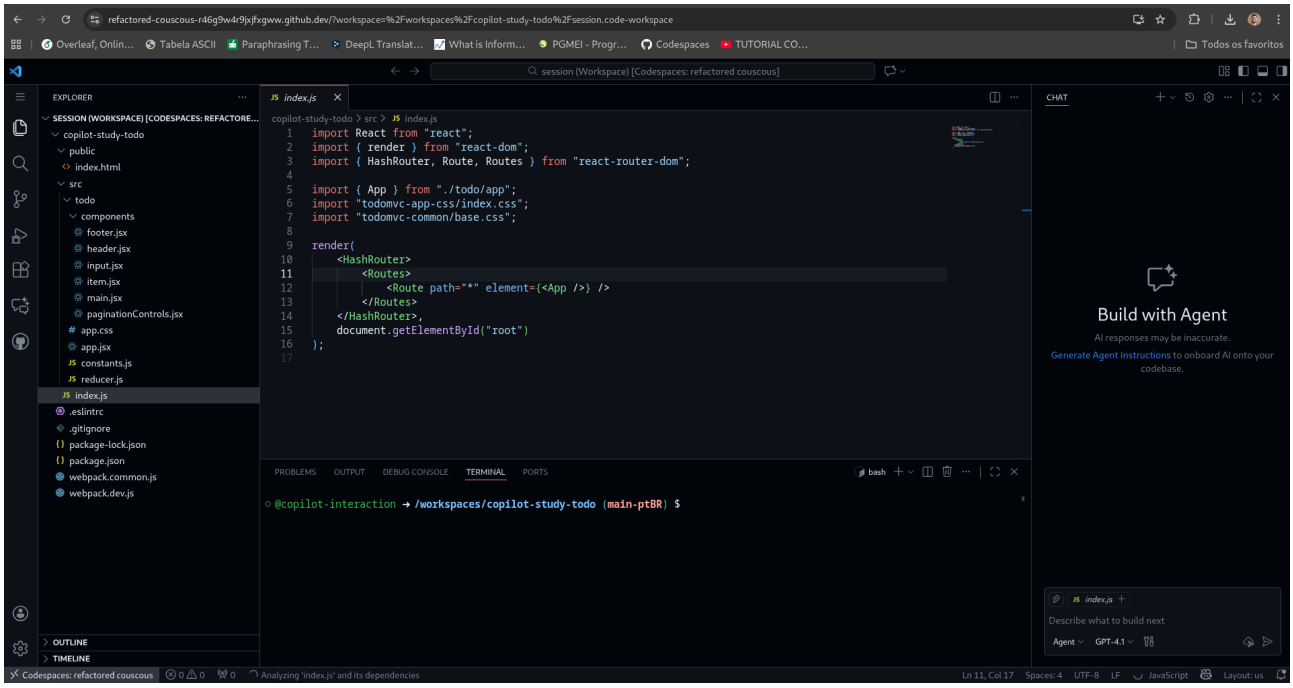

Figura 4.4. Ambiente de desenvolvimento

constatou-se que as tarefas não podiam ser executadas no ambiente de desenvolvimento. O Copilot conseguiu verificar as tarefas antes que o participante as mencionasse. Comprometendo a interação do participante com a LLM. Assim, foi criado um documento com a descrição e especificação das tarefas disponíveis em links externos ao ambiente de desenvolvimento. Isso ajudou a forçar o participante a ler a tarefa e montar o *prompt* para a Copiloto. Outro aspecto importante do teste piloto foi a validação de que seria possível responder a todas as tarefas em 40 minutos. Dois dos três participantes conseguiram concluir todas as tarefas.

4.5 Análise dos dados

Para guiar a análise dos dados, dividimos-a em três questões de pesquisa. A primeira questão **QP1 - Como varia o nível de sofisticação linguística e o esforço de elaboração dos prompts durante a interação com LLMs em tarefas de programação?** Para analisar como os participantes escreveram os prompts usados na interação com a LLM, extraímos as seguintes métricas 4.1: (i) Quantidade de palavras; (ii) Número de sentenças; (iii) Facilidade de leitura (Flesch Reading Ease); (iv) Nível Escolar Flesch-Kincaid (Flesch-Kincaid Grade Leve); (v) Gunning Fog Index; (vi) SMOG Index; (vii) Coleman-Liau Index; (viii) Automated Readability Index; (ix) Contagem de palavras difícies (Dale-Chall Readability Score). Para calcular as métricas, utilizamos a biblioteca Python textstat. (ALBRECHT et al., 2020).

Para responder à segunda questão de pesquisa, **QP2 - De que maneira os participantes estruturam suas solicitações à LLM: via questionamento aberto, verificação de hipóteses, instruções diretas ou por comandos implícitos ?**, realizamos a análise de aspectos *qualitativos*

Tabela 4.1. Métricas de Texto e Legibilidade

Métrica	**Definição**	**Explicação e Interpretação**
(i) Quantidade de palavras	Contagem total de *tokens* (palavras) presentes no texto.	Medida básica. Base para quase todos os cálculos de legibilidade. Textos muito longos podem cansar, muito curtos podem ser superficiais.
(ii) Número de sentenças	Contagem total de frases gramaticais (delimitadas por pontuação como ., ?, !).	Indica a segmentação. Muitas sentenças, com poucas palavras, sugerem frases curtas. Poucas sentenças em um texto longo indicam complexidade sintática.
(iii) Flesch Reading Ease	Índice de 0 a 100 que estima a facilidade de leitura (com base em sílabas e no comprimento da frase).	Quanto maior, mais fácil. 90-100 (Muito fácil); 60-70 (Padrão); 0-30 (Muito difícil/Acadêmico).
(iv) Flesch-Kincaid Grade Level	Conversão do Flesch Reading Ease para o sistema de séries escolares dos EUA.	Indica anos de estudo necessários. Ex: 8.0 significa que um aluno da 8ª série conseguiria ler.
(v) Gunning Fog Index	Estima anos de educação formal necessários para entender o texto na primeira leitura.	Penaliza o uso de “palavras complexas” (3 ou mais sílabas).
(vi) SMOG Index	Estima anos de educação para 100% de compreensão (baseado em polissílabas).	Considerado o “padrão-ouro” para textos de saúde. Mais rigoroso que o Flesch-Kincaid.
(vii) Coleman-Liau Index	Índice de nível escolar baseado em caracteres, não em sílabas.	Usa a média de letras por 100 palavras. Computacionalmente eficiente e preciso.
(viii) Automated Readability Index (ARI)	Índice de nível escolar baseado em caracteres (similar ao índice de Coleman-Liau).	Ajusta os pesos de caracteres por palavra e palavras por frase para aproximar-se da idade de leitura.
(ix) Dale-Chall Readability Score	Métrica baseada na familiaridade do vocabulário (lista de 3.000 palavras comuns).	Calcula a proporção de palavras fora da lista de vocabulário comum. Avalia se o léxico é adequado ao público-alvo.

dos prompts. Agrupamos os prompts de acordo como as perguntas foram formuladas (RICHARDS; WESSEL, 2024).

1. **Pergunta Aberta (Open Question):** caracteriza-se por solicitações que buscam explicações, localizações ou "como fazer". Normalmente iniciadas por pronomes interrogativos como "How", "Where"e "What", essas perguntas sugerem que o usuário considera o LLM como um consultor ou fonte de conhecimento. Ex: "How do I change the color of the message?".

2. **Instrução (Instruction):** Comandos imperativos são aqueles em que o usuário atribui uma ação direta ao modelo, aguardando um resultado imediato (como a geração ou a refatoração de código), sem necessidade de solicitar uma explicação do procedimento. Ex: "Sort this list."ou "Fix the code below.".
3. **Hipótese (Hypothesis):** Interações em que o usuário apresenta uma solução ou compreensão prévia e pede a LLM que a valide ou corrija algo. Ex: "Is the error caused by the missing variable X?".
4. **Implícito (Implicit):** Prompts que não apresentam uma pergunta ou instrução explícita, mas apenas trechos de código ou contexto, aguardando que a LLM deduza a necessidade de ajuda (normalmente "complete"ou "explique").

Para a realização desse agrupamento, dois pesquisadores (autor e orientador) conduziram uma análise manual e individual de cada prompt.

Para a questão de pesquisa três, **QP3 - Como se comparam os prompts de pessoas com diferentes perfis?** Nesta questão, realizamos a comparação dos resultados obtidos em QP1 e QP2, considerando os diferentes perfis dos estilos cognitivos, comparando TIM e ABI, experientes e novatos. A comparação considera os resultados obtidos nas duas primeiras questões de pesquisa, com o objetivo de verificar se há diferenças estatisticamente significativas no estilo de escrita dos prompts. As métricas que utilizamos na análise são:

Contagem de palavras	X	Persona
Nível de escolaridade	X	Persona
Facilidade de leitura	X	Persona
Facilidade de leitura	X	Nível de experiência - Medido em anos. 2 (1 a 2 anos), 3 (3 a 5 anos), 4 (5 a 10 anos), 5 (acima de 10 anos).
Palavras por Prompt	X	Persona + experiência
Clusterização (Métricas + classificação)		

Tabela 4.2. Métricas para analise da QP3

Levando em conta a natureza das métricas coletadas, como a contagem de palavras e os índices de legibilidade, decidiu-se usar testes não paramétricos. O teste U de *Mann-Whitney* foi utilizado para comparar as médias e distribuições entre dois grupos independentes, como os perfis ABI e TIM. Quando os dados não seguem uma distribuição normal, esse teste é apropriado para determinar se existem diferenças estatisticamente significativas. Além disso, calculou-se o tamanho do efeito (d de *Cohen*) para medir a magnitude da diferença observada entre os grupos, o que possibilitou avaliar a importância prática dos resultados estatísticos.

O algoritmo de aprendizado de máquina não supervisionado *K-Means* foi empregado para investigar padrões de comportamento que poderiam não ser claros apenas com a separação de personas. Por fim, para verificar a existência de associação entre as variáveis qualitativas —

especificamente entre os grupos combinados (Estilo Cognitivo + Experiência) e a categoria de *prompt* utilizada (Pergunta Aberta, Instrução, Hipótese ou Implícito) — foi aplicado o teste *Chi-Square*.

4.6 Características dos participantes

Com base nos dados levantados através do questionário sociodemográfico, a amostra final é composta por um total de 27 indivíduos de diversas nacionalidades.

No que diz respeito à distribuição por gênero, verifica-se uma predominância do gênero masculino, com $81,5\%$ da amostra ($n = 22$), enquanto o gênero feminino corresponde a $14,8\%$ do total ($n = 4$). Um participante ($3,7\%$) optou por não informar o seu gênero.

No que diz respeito à origem geográfica, a amostra apresenta uma diversidade internacional significativa, incluindo participantes de dez países distintos. Com 13 participantes, o Brasil é o país mais representado, seguido pela Holanda, que conta com 5. A distribuição abrange também a Romênia ($n = 2$) e participações individuais ($n = 1$) de países como Rússia, Reino Unido, Índia, Japão, China, Estônia e Itália.

Em relação à situação profissional, a amostra apresenta dados diversificados. O grupo mais significativo é formado por pessoas que trabalham em tempo integral ($n = 11$; $40,7\%$), seguido por estudantes ($n = 9$; $33,3\%$). Há também participantes que trabalham meio período ($n = 4$; $14,8\%$), profissionais autônomos ($n = 2$; $7,4\%$) e um participante que está desempregado ($3,7\%$).

Adicionalmente, analisou-se o idioma empregado pelos participantes durante a interação com o LLM, por meio da elaboração dos prompts. Observou-se que 15 participantes utilizaram o português (Brasil), correspondendo a $55,6\%$ da amostra, enquanto os demais 12 participantes ($44,4\%$) conduziram suas interações em inglês. Com o objetivo de assegurar a normalização dos dados e a uniformidade na análise, optou-se por traduzir para o inglês todas as interações originalmente realizadas em português.

5 RESULTADOS

Este capítulo detalha os resultados da análise de 257 *prompts* extraídos das interações de 27 desenvolvedores com o GitHub Copilot no projeto TodoMvc. O estudo concentra-se no efeito do estilo cognitivo (personas GenderMag Abi e Tim) e da experiência técnica na criação dessas instruções, com o objetivo de entender como esses fatores influenciam a engenharia de *prompts*.

5.1 QP1 - Como variam o nível de sofisticação linguística e o esforço de elaboração de prompts durante a interação com LLMs em tarefas de programação?

Para responder a esta questão de pesquisa, foram analisados 257 *prompts* coletados no nosso estudo. De maneira geral, os *prompts* nas 6 tarefas apresentaram, em média, **19,80 palavras** por prompt. A tarefa 2 - C apresentou a maior média, com 52,62 palavras, resultado esperado, já que esta tarefa pode ser considerada a mais complexa do estudo.

A Figura 5.1 mostra o histograma, indicando que a maioria dos prompts é curta (entre 8 e 23 palavras), mas há uma variação de *prompts* muito extensos (até 163 palavras). Esta variação é o efeito de alguns participantes que utilizaram o enunciado da tarefa como *prompt*, por exemplo, o participante **P3**, na tarefa 2 - C.

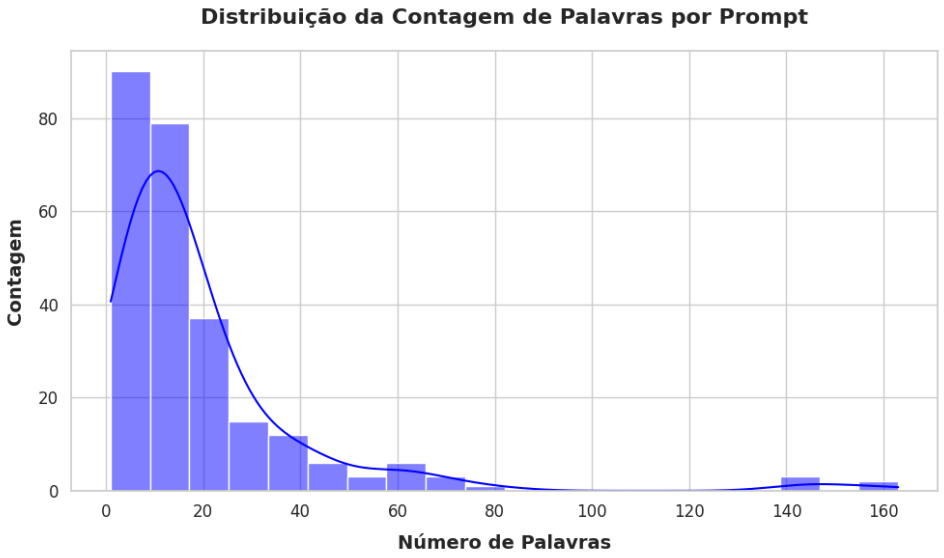

Figura 5.1. Histograma, distribuição de palavras

A Figura 5.2 apresenta o boxplot da variação na extensão dos *prompts* (medida em número de palavras) submetidos pelos participantes a cada uma das seis tarefas analisadas (1-A a 2-C). O eixo horizontal representa as tarefas, enquanto o eixo vertical quantifica a verbosidade das interações. As

tarefas introdutórias, especificamente as Tarefas 1-A e 1-B, apresentam as menores medianas e uma dispersão interquartil reduzida, indicando que a maioria dos participantes utilizou prompts curtos e diretos, com pouca variação entre eles.

Ao abordar as tarefas subsequentes, especialmente nas séries 2-A, 2-B e 2-C, observa-se um aumento da mediana e maior dispersão dos dados, incluindo a presença de valores isolados (*outliers*), o que indica que, diante da maior complexidade das tarefas, os participantes tendem a adotar prompts com descrições mais longas.

Os resultados deste estudo indicam que a complexidade da tarefa sustenta a hipótese de que a interação com o LLM é adaptativa: a carga cognitiva e o esforço de formulação do prompt, considerando o número de palavras, aumentam proporcionalmente à complexidade percebida da tarefa.

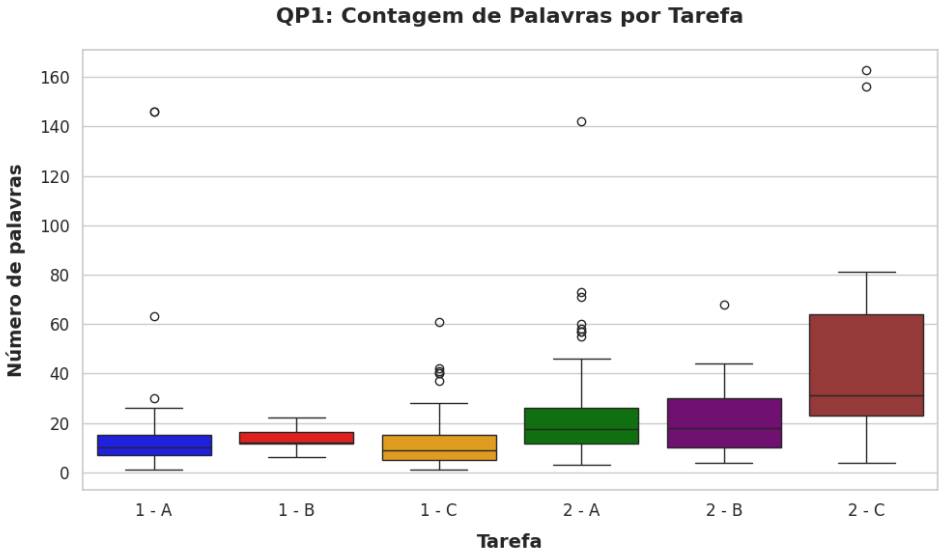

Figura 5.2. Quantidade de palavras por tarefa

Para aprofundar a análise, consideramos um conjunto de métricas que medem a complexidade dos prompts escritos. Para isso, utilizamos a métrica que determina o nível de escolaridade (*Grade Levels*) do *prompt*. Como estas métricas podem variar na interpretação, usamos 5 algoritmos distintos para verificar se há consenso quanto à complexidade do *prompt*. Como as métricas `gunning_fog` e `smog_index` usam fórmulas diferentes (algumas pesam mais as sílabas complexas, outras as frases longas), colocá-las lado a lado ajuda a ver se o texto é consistentemente complexo. O gráfico 5.3 mostra que o nível de leitura dos prompts construídos pelos participantes é do nível 8ª-9ª série dos Estados Unidos.

A análise visual dos boxplots sugere que a linguagem empregada na interação humano-LLM durante as tarefas apresenta menor complexidade sintática. Essa simplicidade na linguagem indica que os participantes priorizam a eficácia na comunicação e a rapidez, em detrimento da formalidade gramatical. Embora a tarefa seja técnica e introduza jargões específicos (capturados por métricas

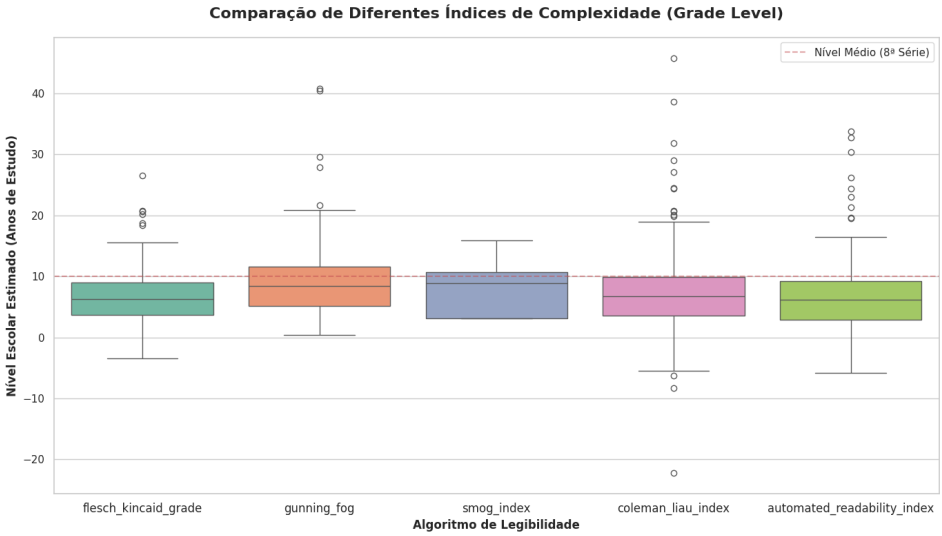

Figura 5.3. Nível de Escolaridade

sensíveis a polissílabos, como Gunning Fog ou SMOG), a maioria das frases permanece direta e compreensível.

A Figura 5.4 mostra a variação do índice de facilidade de leitura de Flesch (FRE) ao longo das seis tarefas. Nesta escala, textos considerados de fácil leitura e acessibilidade apresentam valores mais altos (entre 60 e 70), enquanto valores mais baixos indicam maior densidade acadêmica ou técnica.

A distribuição dos dados mostra que, de forma consistente em todas as tarefas, a facilidade de leitura permanece majoritariamente alta, geralmente situando-se na faixa de "Leitura Fácil"a "Muito Fácil"(pontuações entre 70 e 90+). No entanto, nota-se uma leve queda no desempenho em tarefas que requerem a manipulação direta de trechos de código, como as Tarefas 2-A e 2-C. Em situações específicas como essas, a inclusão de trechos de código ou de termos técnicos no *prompt* geralmente reduz temporariamente a facilidade de uso.

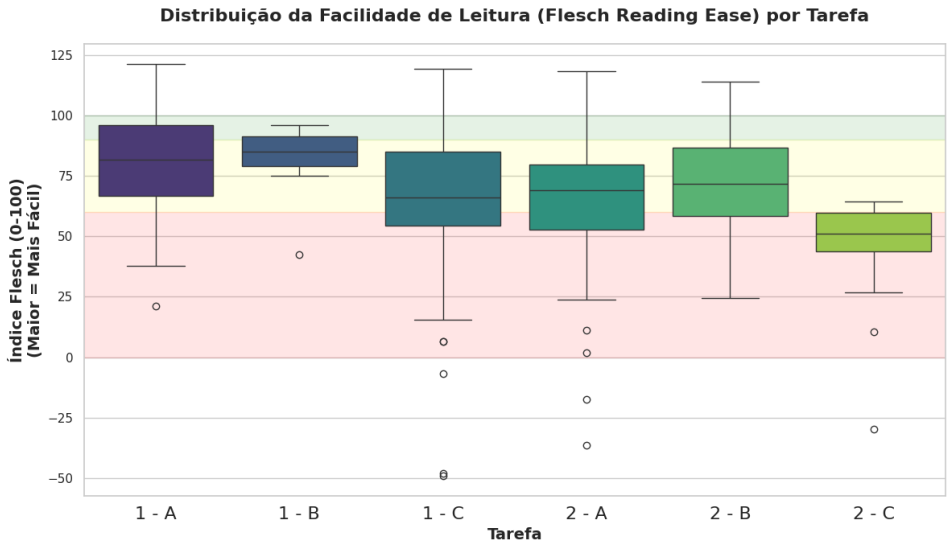

Figura 5.4. Distribuição da facilidade de leitura

Em resumo o **esforço na elaboração** dos *prompts*, medido pelo número de palavras, aumenta à medida que a complexidade da tarefa aumenta. Em desafios introdutórios (1- A, 1-B) os participantes utilizaram *prompts* curtos e diretos. A mediana de palavras é baixa e a variação é pequena, indicando que a comunicação é simples. Por outro lado, para tarefas com desafios maiores (2 - A, 2 - C), observa-se um aumento significativo no tamanho das interações. O boxplot 5.2 confirma que a Tarefa 2 - C apresenta a maior dispersão e valores mais elevados (incluindo *outliers*), o que reforça a ideia de que tarefas mais complexas demandam descrições mais elaboradas e maior esforço cognitivo. No entanto, é fundamental observar que esse aumento no volume textual não implica necessariamente em maior complexidade sintática. A sofisticação linguística permanece **estável e acessível**, com uma leve adaptação técnica. As métricas de escolaridade, como *Flesch-Kincaid* e *Gunning Fog*, indicam que a linguagem está em um nível adequado para os alunos da 8ª à 9ª série. Isso sugere que, mesmo em um contexto de programação, a estrutura sintática das frases é básica. A facilidade de leitura (*Flesch*) está presente na maioria dos *prompts*. Contudo, nota-se uma diminuição da clareza de leitura nas atividades que requerem a manipulação de código (2-A e 2-C). Isso não acontece porque a linguagem se torna "complexa", e sim porque a inclusão de trechos de código e de jargões técnicos compromete as métricas de legibilidade.

5.2 QP2 - De que maneira os participantes estruturam suas solicitações à LLM: por meio de questionamento aberto, verificação de hipóteses, instruções diretas ou comandos implícitos ?

A segunda pergunta de pesquisa (QP2) explora a natureza qualitativa das interações. Para abordar essa questão, 257 *prompts* foram classificados por dois pesquisadores (autor e orientador). Assim, cada *prompt* foi lido e classificado conjuntamente, com discussões até chegar ao consenso.

O gráfico 5.5 mostra a distribuição de frequência dos padrões ao longo das tarefas, indicando a incidência de cada tipo de *prompt* nas interações dos 27 participantes.

A análise dos dados revela uma grande desigualdade nas estratégias de interação adotadas pelos participantes. É possível perceber a predominância da categoria **"Pergunta Aberta"**, indicando que, ao lidar com as tarefas propostas, os participantes, por não terem conhecimento da base de código, possivelmente recorrem a essa estratégia de prompts para explorar e compreender o código, em vez de se restringir à geração automática de código. Os participantes empregaram o LLM principalmente como um instrumento de navegação e assistência.

A segunda estratégia mais usada foi a realização de instruções diretas, indicando comandos imperativos empregados somente quando o usuário já tinha um entendimento claro do que deveria

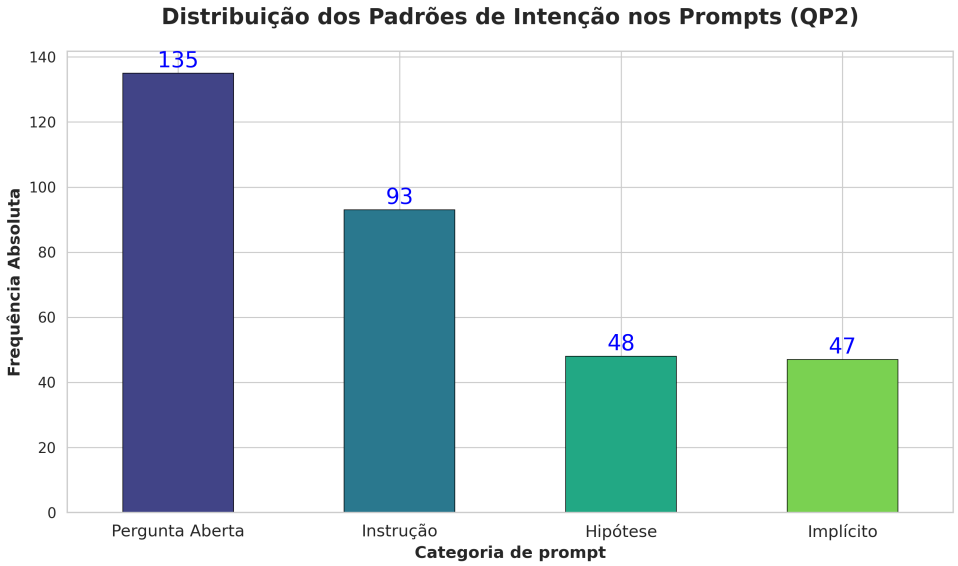

Figura 5.5. Distribuição dos padrões de prompts

ser feito, com o objetivo de auxiliar na implementação e, possivelmente, impactar a produtividade da tarefa.

Por fim, é possível observar baixa utilização dos padrões de hipótese e implícito, sugerindo que os participantes pouco utilizaram o Copiloto para confirmar uma possível solução que haviam pensado. Ao contrário disso, os participantes delegaram a construção da solução desde o início.

A Figura 5.6 apresenta um gráfico de barras empilhado que mostra o número de tipos de *prompts* por tarefa. O eixo vertical representa o total de *prompts*, permitindo comparar o volume total de interações e a proporção de cada tipo de *prompt* em relação à tarefa específica.

Os resultados indicam que a "Pergunta Aberta"(em azul) prevalece em quase todas as tarefas. Isso reforça a conclusão de que, independentemente do tipo específico do problema — seja a simples tarefa de trocar o cor de um elemento de uma página web (1-A) ou a correção de um defeito (2-A), a principal estratégia dos participantes foi questionar o Copiloto para promover a compreensão, em vez de simplesmente ordenar ações. Nota-se que a categoria "Instrução"(verde-escuro) foi o segundo padrão mais utilizado pelos participantes; entretanto, foi o tipo de *prompt* mais utilizado na tarefa de maior complexidade **2 - C**. Entre as tarefas simples e as mais complexas, o padrão **"Hipótese"** foi o menos utilizado pelos participantes.

Uma análise mais granular dos dados identificou casos com múltiplas classificações, apresentados na Tabela 5.1. Cerca de 64 dos 257 prompts (24,9%) possuem dois ou mais padrões. A existência de prompts híbridos sugere que os desenvolvedores adotam estratégias para garantir que o LLM compreenda o contexto da tarefa. A combinação mais frequente: **Pergunta aberta + Instrução**. Ao criar um prompt que inclui uma pergunta ("Como faço X?") sobre um comando ("Gere o código para X"), o desenvolvedor mostra que seu objetivo é duplo: procura compreender o conceito (aprender) ao mesmo tempo em que busca uma solução prática (automatizar).

Embora a categoria "Hipótese"tenha sido pouco usada isoladamente, encontramos casos em que os participantes realizaram combinações como Pergunta + Hipótese (n=11) e Hipótese + Instrução

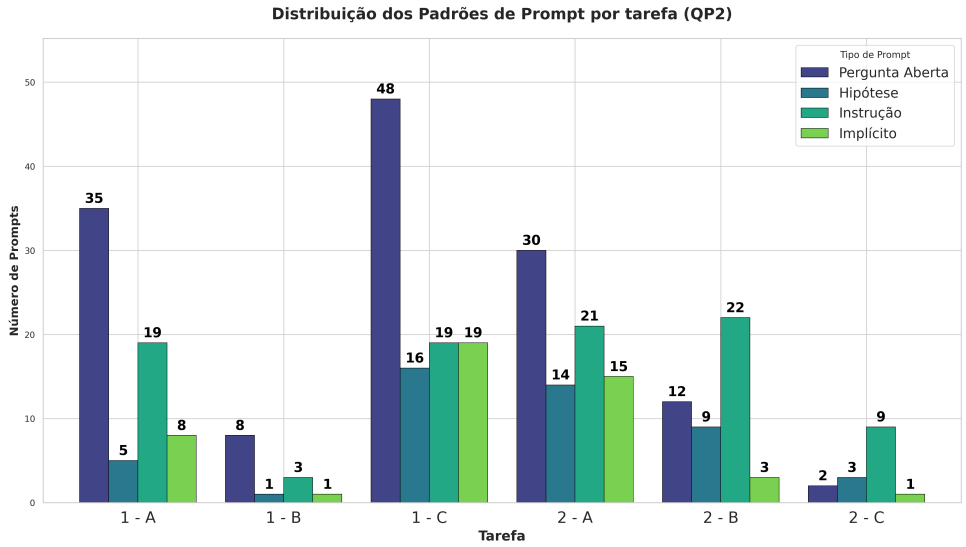

Figura 5.6. Distribuição dos Padrões de prompt por tarefa

(n=10), o que sugere que os participantes geralmente empregam suas hipóteses não apenas para validação, mas também como contexto para direcionar a resposta da LLM. Outro fato é a recorrência da combinação **Pergunta + Implícito**. Este dado reflete o fluxo de trabalho típico de manutenção de código: o desenvolvedor fornece um fragmento de código ou um *log* de erro (Implícito) e acopla uma dúvida explícita (Pergunta).

Tabela 5.1. Prompts multi padrões

Combinação	**Total**
Pergunta + Instrução	18
Pergunta + Implícito	12
Pergunta + Hipótese	11
Hipótese + Instrução	10
Hipótese + Implícito	7
Instrução + Implícito	3
Pergunta + Instrução + Implícito	2
Hipótese + Instrução + Implícito	1

Por fim, os dados evidenciam que a categoria de *prompt* mais empregada é a **Pergunta Aberta**, que predomina em quase todas as tarefas. Essa abordagem sugere que os participantes usaram o LLM principalmente como uma ferramenta de navegação e de suporte para explorar e entender a base de código, em vez de limitá-lo à geração automática de código. A elevada frequência de perguntas abertas corrobora a conclusão de que, na ausência de conhecimento prévio do código, a principal abordagem é compreender o contexto.

5.3 QP3 - Como se comparam os prompts de pessoas com diferentes perfis

A terceira pergunta de pesquisa (QP3) analisa, de forma qualitativa, os *prompts* dos participantes com estilos cognitivos distintos.

Para responder a esta questão de pesquisa, analisamos as métricas quantitativas de duas formas. Primeiro, vamos comparar as métricas de Facilidade de Leitura, o Nível de Escolaridade da estrutura textual estimada e o número de palavras, considerando apenas o estilo cognitivo (TIM e ABI), sem levar em conta a experiência dos participantes. Na segunda análise, apresentaremos uma comparação entre o estilo cognitivo e a experiência, formando quatro grupos: TIM - Experiente, ABI Experiente, TIM - Não Experiente e ABI - Não Experiente; neste caso, a experiência é representada por 4 ou mais anos de atuação no desenvolvimento de software. A tabela 5.2 apresenta a distribuição do número de participantes e de *prompts* realizadas durante as seis tarefas executadas pelos participantes.

Tabela 5.2. Distribuição do número de participantes e prompts

Grupo	Participantes	IDs dos Participantes	Total Prompts
ABI (Exp.)	8	P04, P05, P06, P12, P16, P26, P27, P28	74
ABI (Não-Exp.)	7	P07, P15, P17, P18, P19, P20, P22	76
TIM (Exp.)	7	P01, P03, P08, P10, P11, P13, P14	64
TIM (Não-Exp.)	5	P02, P09, P21, P23, P24	43

A Figura 5.7 mostra que os participantes do TIM apresentam pontuação média maior (74,6) do que os do ABI (66,8). Embora o teste U de *Mann-Whitney* apresente *p-value* de 0.35, indicando a ausência de diferença estatisticamente significativa na distribuição dos valores de scores da métrica de facilidade de leitura, acredita-se que essa ausência de poder estatístico provavelmente está relacionada à quantidade de participantes em cada grupo. No entanto, observa-se que a diferença na média entre os dois grupos é evidenciada pela presença de *outliers* no grupo ABI.

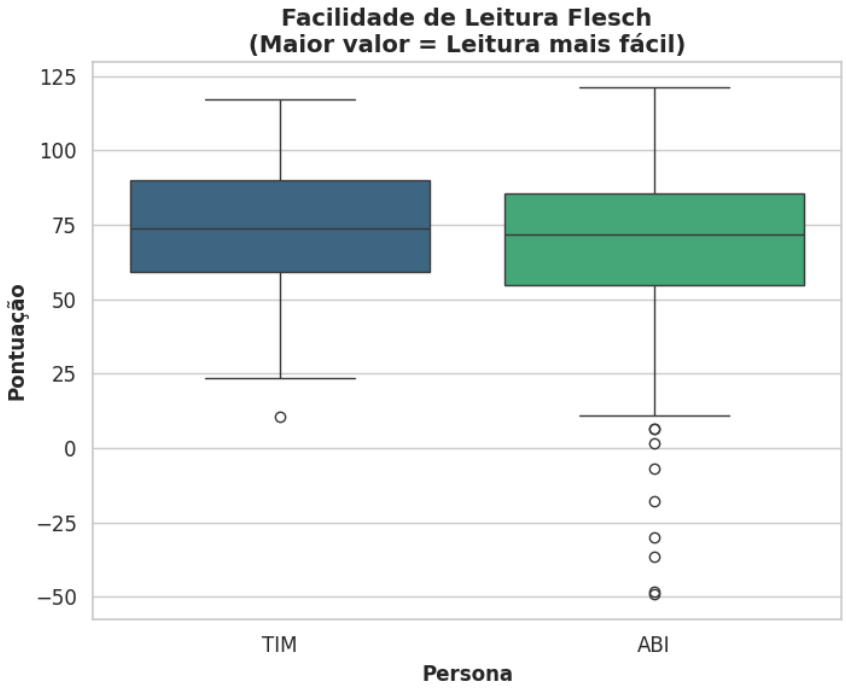

Figura 5.7. Facilidade de leitura

Da mesma forma, na Figura 5.8, a persona ABI apresenta um nível de escolaridade (*Flesch-Kincaid Grade*) exigido ligeiramente maior (6,6) do que o TIM (5,8), sugerindo que ABI utiliza uma linguagem um pouco mais complexa ou técnica, conforme a Figura 5.8, mas não há diferença estatística entre as personas nesta métrica, *Mann-Whitney U Test* (*p-value* = 0.64239)

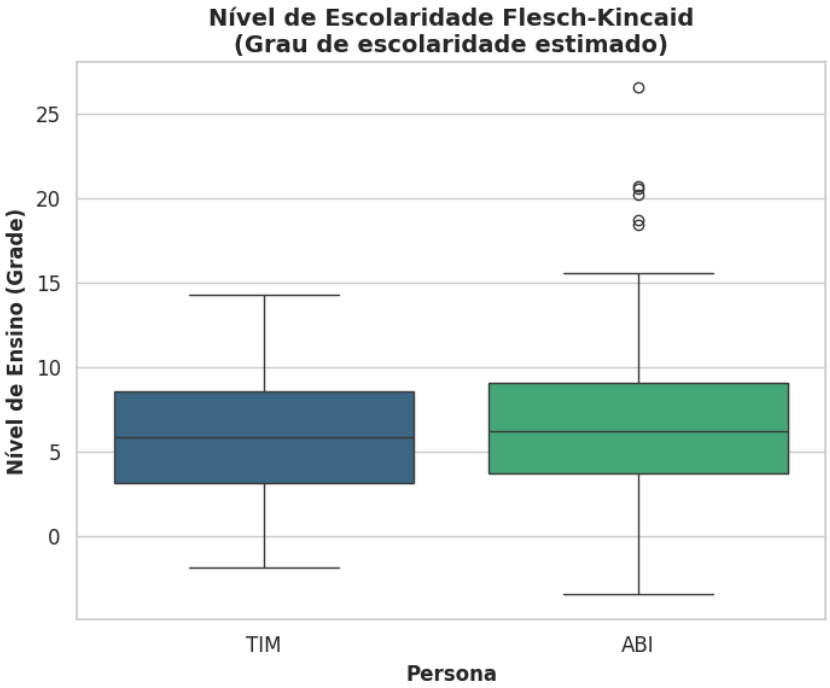

Figura 5.8. Flesch-Kincaid Grade

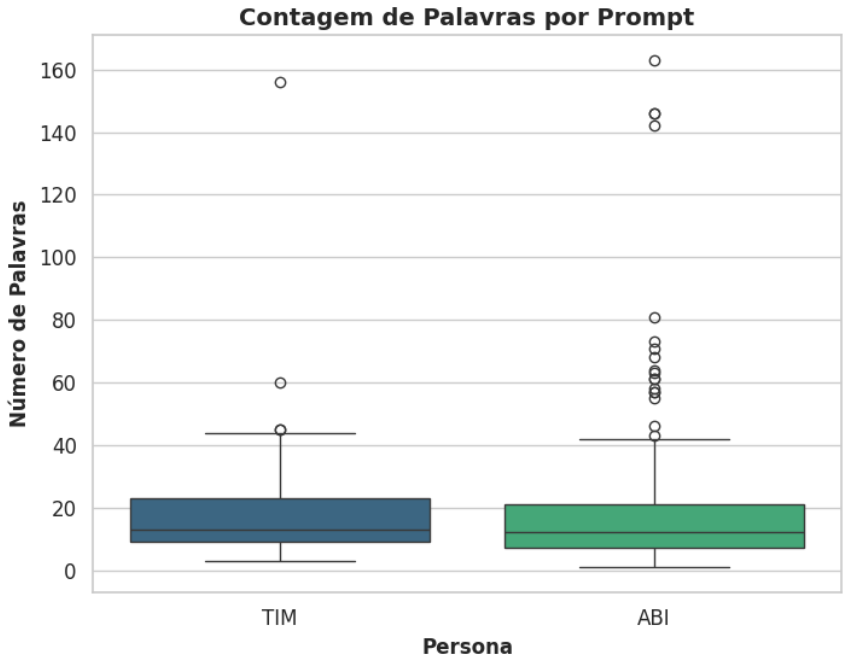

Figura 5.9. Contagem de palavras por prompt

Para aprofundar o entendimento do padrão de escrita dos *prompts* das personas, analisamos o tamanho dos *prompts*, medido em número de palavras, na Figura 5.9. A persona ABI tende a ser um pouco mais verbosa (média de 21,93 palavras) do que a TIM (média de 12 palavras), embora o teste estatístico de *Mann-Whitney U Test* (*p-value* = 0,38958) indique que não há diferença estatisticamente significativa (0,05); o teste de tamanho de efeito indica uma pequena diferença estatisticamente significativa (cohens d = 0,303). De forma mais detalhada, observamos que a diferença existe nos grupos de ABI - Experiente vs TIM - Não Experiente (cohens d= 0,26), conforme apresentamos na Figura 5.10

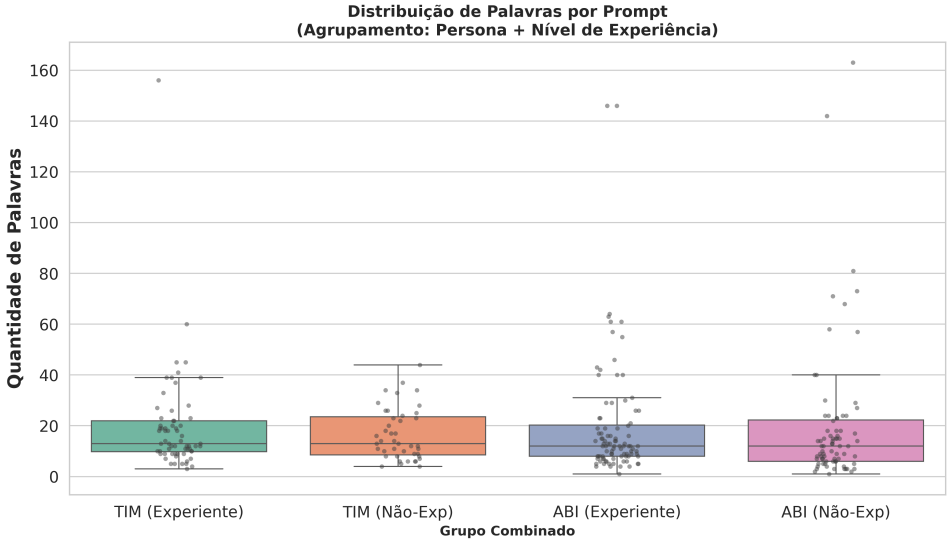

Figura 5.10. Distribuição de palavras por prompt (persona + experiência)

Como não observamos diferença estatisticamente significativa, realizamos uma análise para verificar se os participantes apresentaram comportamentos distintos em alguma das 6 tarefas. A média de palavras empregadas nos *prompts* por tarefa difere entre as personas ABI e TIM e é apresentada na Tabela 5.3.

Tabela 5.3. Métricas de interação por tarefa

Grupos de Participantes	Tarefas	Média de prompts por tarefa	Média de palavras por tarefa	Nº participantes que concluíram a tarefa
ABI - Experiente	1-A	3.00	94.17	8
ABI - Experiente	1-B	1.50	18.00	8
ABI - Experiente	1-C	2.88	40.00	8
ABI - Experiente	2-A	2.00	37.25	7
ABI - Experiente	2-B	2.50	44.25	4
ABI - Experiente	2-C	2.00	83.00	3
ABI - Não Experiente	1-A	3.4	35.50	7
ABI - Não Experiente	1-B	–	–	7
ABI - Não Experiente	1-C	4.43	43.71	5
ABI - Não Experiente	2-A	3.6	121.4	3
ABI - Não Experiente	2-B	2.0	66.33	2
ABI - Não Experiente	2-C	1.33	90.33	2
TIM - Experiente	1-A	2.00	21.60	7
TIM - Experiente	1-B	1.50	20.00	7
TIM - Experiente	1-C	2.00	30.00	7
TIM - Experiente	2-A	3.83	82.50	5
TIM - Experiente	2-B	3.25	63.00	3
TIM - Experiente	2-C	1.00	156.00	1
TIM - Não Experiente	1-A	2.00	18.25	5
TIM - Não Experiente	1-B	1.67	24.33	5
TIM - Não Experiente	1-C	2.20	31.20	4
TIM - Não Experiente	2-A	1.75	34.00	3
TIM - Não Experiente	2-B	2.67	61.33	2
TIM - Não Experiente	2-C	2.00	45.50	2

Ao analisar a relação entre a realização da tarefa e os grupos e as médias de *prompts* e de palavras, observamos que, nas tarefas 1-A e 1-B, todos os grupos obtiveram 100% de sucesso ao completá-la. O grupo ABI - Experiente usou 94 palavras de média, enquanto os demais grupos usaram entre 14 e 35 palavras.

Na tarefa 1-C, observa-se que o grupo ABI utilizou mais *prompts* do que o grupo TIM. O Grupo ABI - Não Experiente precisou de média de 4,43 *prompts* (mais do que o dobro dos TIMs), consequentemente, também apresentou o maior número médio de palavras. Interessante observar que o grupo ABI - Não Experiente teve a menor taxa de conclusão dos participantes (71,43%), enquanto o TIM - Não Experiente teve (80%) e os grupos experientes de ABI e TIM concluíram com 100% de conclusão.

Nas tarefas da série 2, observa-se que o percentual de sucesso diminui em todos os grupos. Por exemplo, na tarefa 2-A, ABI - Experiente e TIM - Experiente têm as melhores taxas de conclusão, 87,5% e 71,4%. Os grupos não experientes obtiveram 42,85% (ABI), 60% (TIM). Interessante observar que o número de *prompts* da ABI - Não Experiente foi o maior entre os grupos e que a média de palavras foi a maior. No oposto, os indivíduos do grupo ABI - Experiente apresentaram a menor quantidade de *prompts*, em comparação com os do grupo TIM - Experiente.

Em relação à tarefa 2-B o grupo TIM - Experiente obteve a maior média de iterações (3,25 prompts) e palavras (63) para atingir apenas 42,9% de sucesso. Em contraste, o grupo ABI - Não Experiente apresentou a menor média de *prompts* (2,0) e o menor sucesso (25,57%).

A tarefa 2-C apresentou a maior discrepância entre os grupos. O grupo TIM - Experiente obteve apenas 14,3% utilizando a média de 1,00 *prompts* , com o maior volume textual (156 palavras). Já os grupos ABI (Não Experiente e Experiente) apresentaram desempenho entre 37,5% à 28,5% e idêntico entre si, realizando o dobro de interações (2,00 *prompts*) com textos longos (83-90 palavras), o que demonstra que, nesta tarefa complexa, a capacidade de iterar (conversar e ajustar) foi importante para melhorar a taxa de conclusão. O grupo TIM - Não Experiente obteve 40% de sucesso com a menor quantidade média de *prompts* (2,0) e de palavras (45,50).

A análise mais granular indicou diferenças de interação com o copiloto, o que motivou uma análise de agrupamento dos participantes para entender se essas diferenças poderiam ser agrupadas em grupos com algum padrão de interação comum. Para isso, agrupamos os *prompts* dos participantes utilizando o algoritmo *K-Means*, com base nas métricas analisadas. Essa análise permitiu a identificação de quatro Agrupamentos (0 a 3), conforme apresentado na Figura 5.11. Ressalta-se que estes agrupamentos agruparam *prompts* de perfis cognitivos e experiências distintas; no entanto, nossa hipótese é que o agrupamento conseguiria agrupar majoritariamente perfis em um único grupo.

Ao fazer a análise dos padrões de comportamento, a segmentação demonstrou a existência de 4 grandes grupos:

1. **Pergunta Aberta**: O agrupamento 1, com 125 ocorrências, apresenta um comportamento em que os *prompts* são majoritariamente formulados como perguntas. Neste grupo, a legibilidade é elevada (82,1) e a concisão sugere que, na maioria das vezes, os participantes usam o Copiloto como uma ferramenta de consulta rápida e de baixo esforço cognitivo. Das 125 ocorrências, verificamos que 37 *prompts* pertencem ao grupo ABI - Experiente; 34 ABI - Não Experiente; 37 TIM - Experiente e 17 TIM - Não Experiente
2. **Instrução**: O Agrupamento 2, com 77 ocorrências, representa a transição da incerteza à ação. A redução da facilidade de leitura para um nível médio (68,6) indica a necessidade de empregar terminologia técnica para fornecer instruções claras sobre como modificar o código. Das 77 ocorrências, 27 *prompts* estão no grupo ABI - Experiente; 12 ABI - Não Experiente; 18 TIM - Experiente e 20 TIM - Não Experiente
3. **Implícito**: O Agrupamento 0, com 47 ocorrências, isola uma ação específica de“debug”. A legibilidade diminui significativamente (40,1) não devido a uma má redação, mas à alta concentração de jargão técnico nos trechos de código colados. Este agrupamento mostra que,

quando ocorrem erros, os usuários optam por manter a descrição em linguagem natural e por recorrer à evidência direta (o código). Das 47 ocorrências analisadas, observou-se que 7 *prompts* pertencem ao grupo ABI – Experiente, 26 ao grupo ABI – Não Experiente, 8 ao grupo TIM – Experiente e 6 ao grupo TIM – Não Experiente.

4. **Instrução + implícito:** A característica que define o agrupamento 3, com 8 ocorrências, é a extensão significativa dos *prompts*. Em comparação com a média geral de 20 palavras, este grupo apresenta uma média de **118 palavras**, o que caracteriza textos mais longos. A classificação evidencia uma combinação de **Instrução** com **Implícito** (código colado), indicando que o participante busca, simultaneamente, fornecer diretrizes e inserir código. A baixa legibilidade (pontuação *Flesch* 28,4) indica um texto difícil, denso e com prováveis problemas na estrutura sintática. Das 8 ocorrências, 3 *prompts* pertencem ao grupo cognitivo ABI - Experiente; 4 ao grupo ABI - Não Experiente; e 1, ao grupo TIM - Experiente.

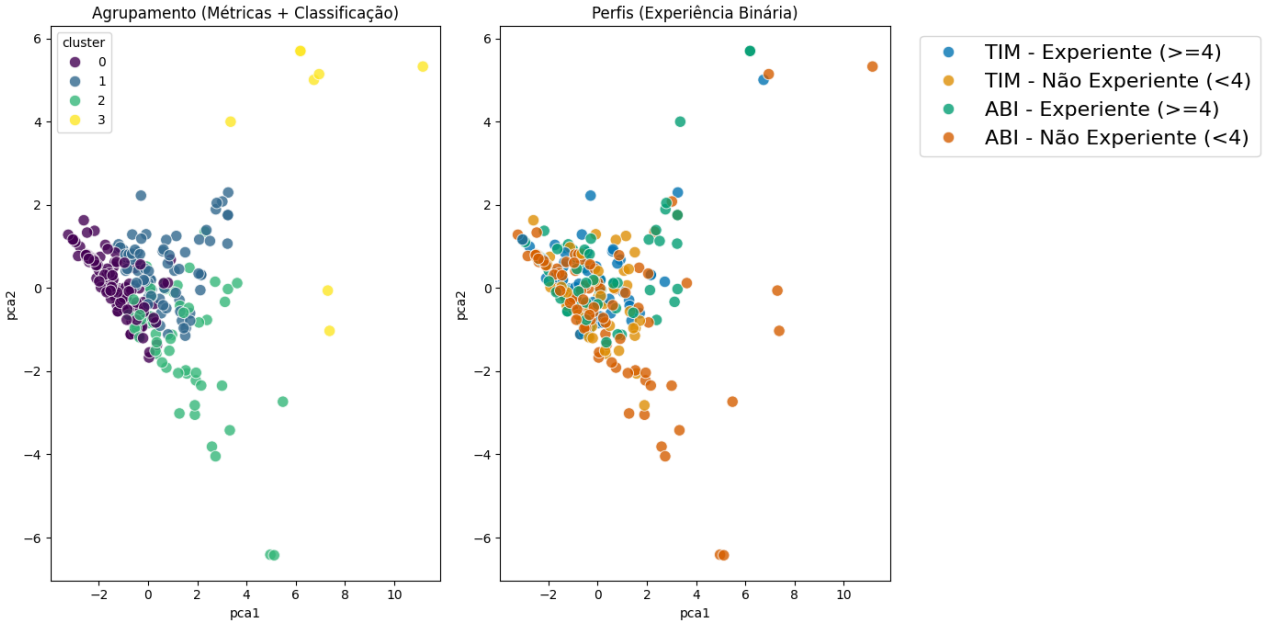

Figura 5.11. Agrupamento: Métricas + Classificação

Por fim, para concluir a análise da QP3, verificou-se a associação entre o estilo cognitivo e o tipo de *prompt* dos participantes do estudo. Para esta análise, construímos a Tabela 5.4 e utilizamos o teste *chi-square* que é um teste estatístico não paramétrico utilizado para verificar se existe uma associação significativa entre duas variáveis qualitativas (categóricas). Ele compara as frequências observadas em uma tabela de contingência às frequências esperadas, caso as variáveis fossem independentes.

Tabela 5.4. Tipo de *prompts* por persona

Grupo (Persona + Experiência)	Pergunta aberta	Hipótese	Instrução	Implícito
ABI - Experiente	36	13	32	11
ABI - Não Experiente	40	15	17	23
TIM - Experiente	37	11	23	9
TIM - Não Experiente	22	9	21	4

O teste revelou uma associação estatisticamente significativa entre o Grupo e o Tipo de *prompt* ($X^2 = 17,35$; GL = 9; p = 0.0380). Este resultado (p < 0,05) permite rejeitar a hipótese nula, confirmando que o padrão de escolha das interações não é aleatório, e sim dependente do perfil do usuário, concluindo que embora quantitativamente as métricas não indiquem uma diferença estatística relevante no comportamento global das personas, é sim possível verificar as personas tem tipos de *prompts* que são mais usados como estratégia para resolução de tarefas.

O grupo ABI - Experiente utilizou mais instruções diretas do que o esperado estatisticamente (32 observadas vs. 26 esperadas). O comportamento alinha-se à literatura sobre usuários experientes, que tendem a formular comandos claros e orientados à ação, em vez de perguntas exploratórias. Curiosamente, este resultado indica que os participantes ABI foram menos avessos ao risco — comportamento esperado de uma pessoa com este estilo cognitivo.

O grupo ABI - Não Experiente apresentou comportamento oposto. O uso de instruções foi abaixo do esperado (17 observados vs. 27,35 esperados) e o de perguntas abertas, acima do esperado (40 observados vs. 39,71 esperados), o que corresponde à potencial falta de experiência e ao estilo cognitivo da persona ABI.

O grupo TIM - Não Experiente não seguiu o padrão esperado para novatos. Eles utilizaram mais instruções diretas do que o esperado (21 observadas vs. 16,12 esperadas). Entretanto, este é um comportamento esperado entre pessoas com o estilo cognitivo TIM. Os participantes do grupo TIM - Experiente apresentaram um perfil equilibrado quanto aos tipos de *prompt*, ora investigativo, ora instrucional, ora baseado em hipótese, o que o torna o perfil mais equilibrado quando comparado aos demais perfis cognitivos.

Em resumo, a análise comparativa dos perfis demonstra que a interação com LLMs pode ser influenciada pela combinação entre o estilo cognitivo (GenderMag) e o nível de experiência do desenvolvedor. Os testes estatísticos confirmaram uma associação significativa entre o grupo do participante e o tipo de *prompt* utilizado. Foi demonstrado que a experiência desempenha um papel moderador, capaz de modificar os comportamentos previstos pelas personas. Por exemplo, o grupo ABI - Experiente contrastou a expectativa de aversão ao risco prevista para esse perfil ao empregar mais instruções diretas do que o modelo estatístico previa. No grupo ABI - Não Experiente, a ausência de conhecimento técnico intensificou os traços exploratórios da persona, levando a uma inclinação por perguntas abertas e à criação de *prompts* excessivamente longos e de difícil compreensão (observados no Agrupamento 3). Esses *prompts* foram empregados para compensar a falta de um vocabulário técnico preciso. Em contrapartida, o perfil TIM - Não Experiente preservou a tendência comportamental de sua persona, dando prioridade a orientações diretas, mesmo sem a experiência necessária. Por último, o grupo TIM - Experiente se destacou por apresentar o comportamento mais equilibrado, alternando entre investigação, instrução e formulação de hipóteses de acordo com a demanda. Esses dados indicam que a eficácia e a natureza da interação com o Copiloto não são aleatórias, mas provavelmente moldadas intrinsecamente pelo perfil do usuário.

6 DISCUSSÕES DOS RESULTADOS

Neste capítulo, discutimos os resultados previamente apresentados e interpretamos como os padrões de interação com Modelos de Linguagem de Grande Escala (LLMs) variam conforme a complexidade da tarefa e o perfil do desenvolvedor. A análise é organizada em torno dos três eixos centrais do estudo: o esforço cognitivo e linguístico (QP1), as estratégias de formulação de *prompts* (QP2) e o impacto dos estilos cognitivos e da experiência (QP3).

6.1 Esforço linguístico e cognitivo na interação (QP1)

Os resultados sugerem que a interação entre desenvolvedores e LLMs não é fixa, e sim adaptável à complexidade da tarefa. Notou-se uma relação direta entre a complexidade da tarefa e a verbosidade dos *prompts*. Nas tarefas iniciais (1-A e 1-B), os participantes adotaram uma abordagem econômica, utilizando *prompts* breves e objetivos. Contudo, à medida que a complexidade cresceu — particularmente nas tarefas 2-A (Persistência) e 2-C (Paginação) — observou-se um aumento considerável na contagem de palavras e na dispersão das informações.

Isso indica que, quando enfrentam desafios lógicos mais complexos, os desenvolvedores sentem a necessidade de fornecer mais contexto para obter uma resposta precisa, o que aumenta o esforço de elaboração. No entanto, é evidente que, embora os *prompts* tenham crescido em tamanho, a sofisticação linguística permaneceu acessível (nível escolar entre a 8ª e a 9ª série). Isso mostra que os programadores, independentemente da experiência, preferem uma comunicação prática e eficiente em vez de se preocupar com formalidades gramaticais. A diminuição da facilidade de leitura (Flesch) em tarefas complexas não indica uma linguagem mais sofisticada, e sim a necessidade de incluir trechos de código e termos técnicos.

6.2 Participantes usaram o copiloto como um tutor (QP2)

A análise qualitativa dos padrões de *prompts* mostrou que a categoria “Pergunta Aberta” foi muito mais prevalente do que as categorias “Instruções” diretas e “Hipóteses”. Esse resultado é essencial para compreender o modelo mental do usuário em relação ao uso do copiloto. Assim, os participantes usam o Copiloto principalmente como um tutor ou guia para entender a base de código desconhecida (projeto *TodoMVC*) e não apenas como um gerador automático de código.

Outro fato importante a ser observado é a pouca utilização isolada da categoria "Hipótese". Isso pode indicar que os desenvolvedores preferem confiar o raciocínio inicial ao copiloto, em vez de formular suas próprias suposições para validação. No entanto, a presença de *prompts* híbridos (por exemplo, Pergunta + Instrução ou Pergunta + Implícito) sugere uma tática de aprimoramento: o usuário procura, inicialmente, compreender o contexto ("Como faço X?") para, na mesma interação, solicitar a execução ("...e crie o código para isso"). A frequência da combinação "Pergunta +

Implícito"indica um processo comum de manutenção e depuração, no qual o código (implícito) atua como validação, enquanto a pergunta natural orienta o copiloto na interpretação desse código.

6.3 A Influência do estilo cognitivo e da experiência (QP3)

Os dados confirmam estatisticamente (p= 0.0380 - teste de chi-quadrado) que a interação com o copiloto é influenciada pela combinação entre o estilo cognitivo e a experiência. A experiência desempenha um papel significativo como fator moderador: o grupo ABI - Experiente superou a aversão ao risco esperada da persona, ao utilizar mais instruções diretas do que o modelo previa. Em contrapartida, o grupo ABI - Não Experiente recorreu a uma estratégia compensatória, gerando *prompts* longos e densos (padrão "Instrução + Implícito") para suprir a falta de vocabulário técnico preciso.

Quanto ao perfil da TIM, a tendência à objetividade se manteve. O grupo TIM - Não Experiente priorizou instruções diretas, comportamento típico de seu estilo cognitivo, mesmo sendo novato. Já o TIM - Experiente mostrou-se o perfil mais equilibrado no uso dos tipos de *prompt*. Contudo, em tarefas de alta complexidade (2-C), a brevidade excessiva do TIM - Experiente resultou em menor sucesso (14,3%) quando comparada à abordagem mais iterativa e verbosa dos grupos ABI (37,5%), demonstrando que a capacidade de "conversar"com o modelo pode de fato ter alguma influência para o sucesso de desafios maiores.

7 CONCLUSÃO

Esta pesquisa teve como objetivo principal compreender como diferentes perfis de desenvolvedores interagem com Modelos de Linguagem de Grande Escala (LLMs), especificamente o *GitHub Copilot*, ao realizar tarefas de compreensão e implementação de código. O estudo buscou preencher uma lacuna na literatura ao investigar não apenas os padrões técnicos dos *prompts*, mas também como fatores humanos — especificamente a experiência e o estilo cognitivo (mensurados pelo método GenderMag) — influenciam a engenharia de *prompts* na resolução de tarefas de programação.

Por meio de um estudo empírico com 27 participantes, analisando dados quantitativos e qualitativos das interações em um projeto personalizado *TodoMVC*, foi possível responder às três questões de pesquisa propostas.

Os resultados mostraram que a interação com Copiloto é adaptativa e está afetada pela complexidade da tarefa. Notou-se que o trabalho de criação do *prompt*, medido pelo número de palavras, cresce proporcionalmente à complexidade do desafio. No entanto, a sofisticação linguística continuou sendo acessível, com um nível escolar estimado entre a 8ª e a 9ª série. Isso sugere que os programadores valorizam uma comunicação direta e a combinação entre linguagem natural e sintaxe de código, em vez de seguir formalismos gramaticais de forma geral.

Quanto à estruturação dos *prompts*, identificou-se a predominância da categoria "Pergunta Aberta"em quase todas as tarefas. Esse dado revela que os participantes utilizaram o Copiloto, especialmente, como tutor para a navegação e a compreensão da base de código desconhecida, e não apenas como um gerador passivo de código.

A análise estatística confirmou uma correlação significativa entre o perfil do usuário (que combina o estilo cognitivo GenderMag e a experiência) e o tipo de *prompt* empregado. Enquanto os perfis TIM mantiveram a tendência à objetividade e às instruções diretas, os perfis ABI inexperientes recorreram a *prompts* extensos e complexos devido à ausência de vocabulário técnico. Em contrapartida, a experiência atuou como fator moderador, levando o grupo ABI - Experiente a superar a aversão ao risco e a adotar instruções mais diretas.

As seções a seguir exploram as implicações dos resultados, considerando especificamente os diferentes interessados no ecossistema de engenharia de software.

7.1 Implicações para os desenvolvedores

Os resultados deste estudo sugerem que a eficiência no uso de assistentes de inteligência artificial não é automática, mas sim uma habilidade que evolui com a experiência técnica. A ocorrência de *prompts* híbridos, como “Pergunta + Implícito” (12 ocorrências) e "Instrução + Implícito", indica que a inclusão de trechos de código é uma prática comum na depuração. No entanto, o Agrupamento 3 revelou que desenvolvedores inexperientes (especialmente do perfil ABI) tendem a criar *prompts* excessivamente

longos (média de 118 palavras) e de baixa legibilidade (Flesch 28,4) ao tentarem fornecer contexto. Isso implica que os desenvolvedores devem buscar um equilíbrio, aprendendo a sintetizar o contexto em vez de depender da colagem extensa de código, o que resulta em textos densos e mal estruturados.

Dado que a categoria "Pergunta Aberta"foi a mais prevalente (135 ocorrências) em quase todas as tarefas, os desenvolvedores devem considerar as LLMs como instrumentos para navegar e aprender a partir da base de código, e não apenas para gerar sintaxe. Essa abordagem foi a principal estratégia para lidar com a falta de conhecimento sobre o projeto.

7.2 Implicações para pesquisa

A confirmação estatística de que o estilo de escrita e a escolha dos *prompts* dependem do perfil do usuário ($X^2 = 17,35$) implica que pesquisas futuras não podem tratar "desenvolvedores"de forma única ao analisar os resultados de seus estudos. A distinção entre as personas ABI e TIM, moderada pela experiência, revelou-se uma variável importante, na qual os perfis TIM tendem a ser mais concisos (média de 13 palavras) e os ABI, mais verbosos (média de 20 palavras). Há risco de um estudo adicionar viés ao analisar os desenvolvedores sem considerar experiência, estilo cognitivo ou outros fatores.

O grupo TIM - Experiente apresentou a menor taxa de sucesso (14,3%) na tarefa mais complexa (2-C) usando *prompts* extremamente curtos (1 *prompt*). Em contrapartida, os grupos ABI alcançaram maior sucesso (42,85%) com maior iteração. Isso indica que novos estudos podem explorar esta lacuna e investigar como os padrões de *prompts* podem impactar diretamente o resultado da solução da tarefa.

7.3 Implicações para o desenvolvimento de ferramentas LLM

A predominância significativa de “Perguntas Abertas” (125 ocorrências no Agrupamento 1) sugere que as ferramentas devem ser desenvolvidas principalmente como interfaces de consulta e esclarecimento. A interface não deve apenas simplificar a geração de código ("Instrução"), mas também oferecer suporte sólido ao fluxo de “consultor” ou “tutor” que os usuários tendem a seguir.

A identificação do Agrupamento 3, definido por *prompts* longos, complexos e majoritariamente utilizados por perfis "Não Experientes", indica que as ferramentas de LLM necessitam de recursos para ajudar usuários iniciantes na organização do contexto. A ferramenta seria capaz de detectar quando um usuário cola grandes trechos de código (padrão implícito excessivo) e propor métodos mais eficientes para referenciar arquivos ou funções.

8 AMEAÇAS A VALIDADE

Todo estudo empírico tem limitações intrínsecas que podem influenciar a interpretação e a generalização de seus resultados. Neste trabalho, foram identificadas principais ameaças que precisam ser levadas em conta. A primeira delas refere-se ao tamanho e à representatividade da amostra, considerando que o estudo contou apenas com 27 participantes voluntários. Apesar de o grupo ser composto por estudantes e profissionais da indústria de diferentes nacionalidades e grau de experiência, a amostra é estatisticamente reduzida. Isso significa que os padrões comportamentais identificados, como as diferenças entre os perfis ABI e TIM, podem não refletir a totalidade dos desenvolvedores de software.

Outro aspecto a ser considerado é a especificidade da tarefa e da tecnologia empregada, já que as atividades foram limitadas a um projeto específico (*TodoMVC*) desenvolvido com a biblioteca *React* e a linguagem *JavaScript*. Em outros paradigmas de programação, como linguagens fortemente tipadas ou desenvolvimento *backend*, os padrões de interação e a verbosidade dos *prompts* podem variar, já que a sintaxe do código ou a necessidade de contexto podem demandar estratégias de *prompt* diferentes.

Por fim, destaca-se a ameaça relacionada à adequação das métricas de legibilidade empregadas para responder à primeira questão de pesquisa (QP1). Índices como *Flesch-Kincaid* e *Gunning Fog* foram desenvolvidos originalmente para prosa convencional e seu uso em *prompts* que mesclam linguagem natural com trechos de código (nomes de variáveis e sintaxe de funções) pode gerar resultados distorcidos. Nesses casos, os algoritmos podem interpretar o código como "palavras complexas"ou "polissílabas", elevando artificialmente a estimativa do nível de escolaridade sem que isso reflita uma real sofisticação linguística do participante. Outro fator adicional à validade dos resultados refere-se ao processo de normalização dos dados linguísticos. Como a amostra foi composta por participantes de diversas nacionalidades, observou-se que 55,6% das interações (n=15) foram originalmente realizadas em português, enquanto 44,4% (n=12) ocorreram em inglês. Para garantir a uniformidade na análise e a aplicação das métricas de legibilidade (que possuem parâmetros específicos para a língua inglesa), todos os *prompts* redigidos em português foram traduzidos para o inglês. Apesar do esforço para manter a fidelidade semântica, o processo de tradução pode ter alterado sutilmente a estrutura sintática original e o número de sílabas, o que representa uma variável que pode influenciar os índices de legibilidade calculados, como o *Flesch-Kincaid* e o *Gunning Fog*.

REFERÊNCIAS

AGUIAR, Bernardo; CORREIA, Walter; CAMPOS, Fábio. Uso da escala likert na análise de jogos. *Salvador: SBC-Proceedings of SBGames Anais*, v. 7, n. 2, 2011.

ALBRECHT, Jens; RAMACHANDRAN, Sidharth; WINKLER, Christian. *Blueprints for text analytics using Python*. [S.l.]: "O'Reilly Media, Inc.", 2020.

BECKWITH, Laura; KISSINGER, Cory; BURNETT, Margaret; WIEDENBECK, Susan; LAWRANCE, Joseph; BLACKWELL, Alan; COOK, Curtis. Tinkering and gender in end-user programmers' debugging. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, 2006. (CHI '06), p. 231–240. ISBN 1595933727. Disponível em: <https://doi.org/10.1145/1124772.1124808>.

BORG, Markus; HEWETT, Dave; HAGATULAH, Nadim; COUDERC, Noric; SöDERBERG, Emma; GRAHAM, Donald; KINI, Uttam; FARLEY, Dave. *Echoes of AI: Investigating the Downstream Effects of AI Assistants on Software Maintainability*. 2025. Disponível em: <https://arxiv.org/abs/2507.00788>.

BROWN, Tom B.; MANN, Benjamin; RYDER, Nick; SUBBIAH, Melanie; KAPLAN, Jared; DHARIWAL, Prafulla; NEELAKANTAN, Arvind; SHYAM, Pranav; SASTRY, Girish; ASKELL, Amanda; AGARWAL, Sandhini; HERBERT-VOSS, Ariel; KRUEGER, Gretchen; HENIGHAN, Tom; CHILD, Rewon; RAMESH, Aditya; ZIEGLER, Daniel M.; WU, Jeffrey; WINTER, Clemens; HESSE, Christopher; CHEN, Mark; SIGLER, Eric; LITWIN, Mateusz; GRAY, Scott; CHESS, Benjamin; CLARK, Jack; BERNER, Christopher; MCCANDLISH, Sam; RADFORD, Alec; SUTSKEVER, Ilya; AMODEI, Dario. *Language Models are Few-Shot Learners*. 2020. Disponível em: <https://arxiv.org/abs/2005.14165>.

BURNETT, Margaret; FLEMING, Scott D.; IQBAL, Shamsi; VENOLIA, Gina; RAJARAM, Vidya; FAROOQ, Umer; GRIGOREANU, Valentina; CZERWINSKI, Mary. Gender differences and programming environments: across programming populations. In: *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*. New York, NY, USA: Association for Computing Machinery, 2010. (ESEM '10). ISBN 9781450300391. Disponível em: <https://doi.org/10.1145/1852786.1852824>.

BURNETT, Margaret M.; BECKWITH, Laura; WIEDENBECK, Susan; FLEMING, Scott D.; CAO, Jill; PARK, Thomas H.; GRIGOREANU, Valentina; RECTOR, Kyle. Gender pluralism in problem-solving software. Elsevier Science Inc., USA, v. 23, n. 5, p. 450–460, set. 2011. ISSN 0953-5438. Disponível em: <https://doi.org/10.1016/j.intcom.2011.06.004>.

CAO, Jill; RECTOR, Kyle; PARK, Thomas H.; FLEMING, Scott D.; BURNETT, Margaret; WIEDENBECK, Susan. A debugging perspective on end-user mashup programming. In: *2010 IEEE Symposium on Visual Languages and Human-Centric Computing*. [S.l.: s.n.], 2010. p. 149–156.

CHAMPA, Arifa I.; RABBI, Md Fazle; NACHUMA, Costain; ZIBRAN, Minhaz F. Chatgpt in action: Analyzing its use in software development. In: *2024 IEEE/ACM 21st International Conference on Mining Software Repositories (MSR)*. [S.l.: s.n.], 2024. p. 182–186.

CHARNESS, Gary; GNEEZY, Uri. Strong evidence for gender differences in risk taking. *Journal of Economic Behavior Organization*, v. 83, n. 1, p. 50–58, 2012. ISSN 0167-2681. Gender Differences in Risk Aversion and Competition. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167268111001521>.

DEVLIN, Jacob; CHANG, Ming-Wei; LEE, Kenton; TOUTANOVA, Kristina. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. Disponível em: <https://arxiv.org/abs/1810.04805>.

DOHMEN, Thomas; FALK, Armin; HUFFMAN, David; SUNDE, Uwe; SCHUPP, Jürgen; WAGNER, Gert G. Individual risk attitudes: Measurement, determinants, and behavioral consequences. *Journal of the European Economic Association*, v. 9, n. 3, p. 522–550, 06 2011. ISSN 1542-4766. Disponível em: <https://doi.org/10.1111/j.1542-4774.2011.01015.x>.

EKIN, Sabit. *Prompt Engineering For ChatGPT: A Quick Guide To Techniques, Tips, And Best Practices*. 2023.

GREWAL, Balreet; LU, Wentao; NADI, Sarah; BEZEMER, Cor-Paul. Analyzing developer use of chatgpt generated code in open source github projects. In: *Proceedings of the 21st International Conference on Mining Software Repositories*. New York, NY, USA: Association for Computing Machinery, 2024. (MSR '24), p. 157–161. ISBN 9798400705878. Disponível em: <https://doi.org/10.1145/3643991.3645072>.

HAMID, Md; CHATTERJEE, Amreeta; GUIZANI, Mariam; ANDERSON, Andrew; MOUSSAOUI, Fatima; YANG, Sarah; ESCOBAR, Isaac; SARMA, Anita; BURNETT, Margaret. How to measure diversity actionably in technology. In: _____. [S.l.: s.n.], 2024. p. 469–485. ISBN 978-1-4842-9650-9.

LIANG, Jenny T.; LIN, Melissa; RAO, Nikitha; MYERS, Brad A. *Prompts Are Programs Too! Understanding How Developers Build Software Containing Prompts*. 2024. Disponível em: <https://arxiv.org/abs/2409.12447>.

MACHáčEK, Roman; GRISHINA, Anastasiia; HORT, Max; MOONEN, Leon. *The Impact of Fine-tuning Large Language Models on Automated Program Repair*. 2025. Disponível em: <https://arxiv.org/abs/2507.19909>.

MEYERS-LEVY, Joan; LOKEN, Barbara. Revisiting gender differences: What we know and what lies ahead. *Journal of Consumer Psychology*, v. 25, n. 1, p. 129–149, 2015. ISSN 1057-7408. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1057740814000588>.

MIKOLOV, Tomas; CHEN, Kai; CORRADO, Greg; DEAN, Jeffrey. *Efficient Estimation of Word Representations in Vector Space*. 2013. Disponível em: <https://arxiv.org/abs/1301.3781>.

MONDAL, Saikat; BAPPON, Suborno Deb; ROY, Chanchal K. *Enhancing User Interaction in ChatGPT: Characterizing and Consolidating Multiple Prompts for Issue Resolution*. 2024. Disponível em: <https://arxiv.org/abs/2402.04568>.

NAM, Daye; MACVEAN, Andrew; HELLENDOORN, Vincent; VASILESCU, Bogdan; MYERS, Brad. *Using an LLM to Help With Code Understanding*. 2024. Disponível em: <https://arxiv.org/abs/2307.08177>.

OLIVEIRA, Caio Monteiro de. Utilização de chat bots baseados em llms para automação de testes de software. 2024.

PENNINGTON, Jeffrey; SOCHER, Richard; MANNING, Christopher D. Glove: Global vectors for word representation. In: *Conference on Empirical Methods in Natural Language Processing*. [s.n.], 2014. Disponível em: <https://api.semanticscholar.org/CorpusID:1957433>.

PINTO, Gustavo; SOUZA, Cleidson de; ROCHA, Thayssa; STEINMACHER, Igor; SOUZA, Alberto de; MONTEIRO, Edward. *Developer Experiences with a Contextualized AI Coding Assistant: Usability, Expectations, and Outcomes*. 2023. Disponível em: <https://arxiv.org/abs/2311.18452>.

RICHARDS, Jonan; WESSEL, Mairieli. *What You Need is What You Get: Theory of Mind for an LLM-Based Code Understanding Assistant*. 2024. Disponível em: <https://arxiv.org/abs/2408.04477>.

RIEDL, René; HUBERT, Marco; KENNING, Peter. Are there neural gender differences in online trust? an fmri study on the perceived trustworthiness of ebay offers. *MIS Q.*, Society for Information Management and The Management Information Systems Research Center, USA, v. 34, n. 2, p. 397–428, jun. 2010. ISSN 0276-7783.

SANTOS, Italo; PIMENTEL, João Felipe; WIESE, Igor; STEINMACHER, Igor; SARMA, Anita; GEROSA, Marco A. Designing for cognitive diversity: Improving the github experience for newcomers. In: *2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*. IEEE, 2023. p. 1–12. Disponível em: <http://dx.doi.org/10.1109/ICSE-SEIS58686.2023.00007>.

TAYEB, Ahmad; ALAHMADI, Mohammad D.; TAJIK, Elham; HAIDUC, Sonia. *Investigating Developers' Preferences for Learning and Issue Resolution Resources in the ChatGPT Era*. 2024. Disponível em: <https://arxiv.org/abs/2410.08411>.

VASWANI, Ashish; SHAZEER, Noam; PARMAR, Niki; USZKOREIT, Jakob; JONES, Llion; GOMEZ, Aidan N.; KAISER, Lukasz; POLOSUKHIN, Illia. *Attention Is All You Need*. 2023. Disponível em: <https://arxiv.org/abs/1706.03762>.

VORVOREANU, Mihaela; ZHANG, Lingyi; HUANG, Yun-Han; HILDERBRAND, Claudia; STEINE-HANSON, Zoe; BURNETT, Margaret. From gender biases to gender-inclusive design: An empirical investigation. In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, 2019. (CHI '19), p. 1–14. ISBN 9781450359702. Disponível em: <https://doi.org/10.1145/3290605.3300283>.

WHITE, Jules; FU, Quchen; HAYS, Sam; SANDBORN, Michael; OLEA, Carlos; GILBERT, Henry; ELNASHAR, Ashraf; SPENCER-SMITH, Jesse; SCHMIDT, Douglas C. *A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT*. 2023. Disponível em: <https://arxiv.org/abs/2302.11382>.

WU, Liangxuan; ZHAO, Yanjie; HOU, Xinyi; LIU, Tianming; WANG, Haoyu. Chatgpt chats decoded: Uncovering prompt patterns for superior solutions in software development lifecycle. In: *2024 IEEE/ACM 21st International Conference on Mining Software Repositories (MSR)*. [S.l.: s.n.], 2024. p. 142–146.

XIAO, Tao; TREUDE, Christoph; HATA, Hideaki; MATSUMOTO, Kenichi. Devgpt: Studying developer-chatgpt conversations. In: *Proceedings of the 21st International Conference on Mining Software Repositories*. ACM, 2024. (MSR '24), p. 227–230. Disponível em: <http://dx.doi.org/10.1145/3643991.3648400>.

YENDURI, Gokul; M, Ramalingam; G, Chemmalar Selvi; Y, Supriya; SRIVASTAVA, Gautam; MADDIKUNTA, Praveen Kumar Reddy; G, Deepti Raj; JHAVERI, Rutvij H; B, Prabadevi; WANG, Weizheng; VASILAKOS, Athanasios V.; GADEKALLU, Thippa Reddy. *Generative Pre-trained Transformer: A Comprehensive Review on Enabling Technologies, Potential Applications, Emerging Challenges, and Future Directions*. 2023. Disponível em: <https://arxiv.org/abs/2305.10435>.

APÊNDICES

APÊNDICE A. TERMO DE CONSENTIMENTO

INFORMAÇÕES SOBRE A PESQUISA

para participação em pesquisa científica: Diversidade Cognitiva nas Interações de Desenvolvedores(as) com Assistentes Conversacionais Baseados em Modelos de Linguagem de Grande Escala (LLM – Large Language Models)

Introdução

Você foi convidado(a) a participar deste estudo porque é um(a) programador(a) profissional ou iniciante. Antes de decidir participar, pedimos que leia as seguintes informações para entender sobre o projeto de pesquisa, o que esperamos de você e como lidamos com o processamento dos seus dados pessoais. Com base nessas informações, você pode indicar se aceita participar deste projeto de pesquisa e o processamento dos seus dados pessoais.

Objetivo da pesquisa

O objetivo deste projeto é investigar as interações e preferências de desenvolvedores de software em relação a assistentes de programação conversacionais baseados em Modelos de Linguagem de Grande Escala (LLM – Large Language Models). Caracterizaremos como vários aspectos da cognição dos usuários (por exemplo, estilo de processamento de informações e motivação) impactam essas interações e preferências. O assistente de programação utilizado neste estudo é o GitHub Copilot.

O que é esperado de você?

Você participará de um projeto de pesquisa no qual coletaremos informações por meio de um questionário pré-estudo e de uma sessão experimental. O questionário pré-estudo é usado para identificação de perfil e seleção de participantes. A sessão consiste em tarefas, onde os participantes corrigirão bugs e implementarão novos recursos em uma base de código existente. Você realizará essas tarefas no GitHub Codespace, um ambiente de desenvolvimento integrado (IDE – Integrated Development Environment) baseado no Visual Studio Code. Você pode interagir livremente com a função de chat do GitHub Copilot para auxiliá-lo(a) na conclusão das tarefas. Durante as tarefas, será solicitado que narre seu processo de pensamento e suas percepções sobre as tarefas e a interação com o Copilot. Após as tarefas, será solicitado que você preencha um questionário para avaliar sua interação com o Copilot.

Duração: A participação no estudo levará entre 60 e 70 minutos.
Compensação: Não há compensação financeira para a participação neste estudo.

Riscos potenciais e inconvenientes

Sua participação nesta pesquisa não envolve riscos físicos, legais ou econômicos. No entanto, suas atividades durante a sessão serão processadas pelo GitHub, um serviço terceirizado (veja "O que acontecerá com meus dados?"). Embora o GitHub seja um serviço confiável, não podemos evitar que você envie informações sensíveis acidentalmente. Você não precisa

responder a perguntas que não deseja responder. Sua participação é voluntária. Você pode encerrar sua participação a qualquer momento, sem precisar justificar sua decisão.

Retirada do consentimento e detalhes de contato

Você pode encerrar sua participação na pesquisa a qualquer momento ou retirar seu consentimento para o uso dos seus dados sem precisar dar qualquer explicação. Se optar por sair da pesquisa, os dados já fornecidos não serão utilizados e serão removidos. Caso tenha dúvidas, reclamações ou deseje encerrar sua participação, entre em contato com Bruno Alves De Oliveira pelo email bruno.1990@alunos.utfpr.edu.br.

Quais dados são coletados e processados?

Para conduzir a pesquisa, precisamos coletar e processar dados pessoais. Dados pessoais são informações que podem identificá-lo(a) diretamente (como nome e e-mail) ou indiretamente (como seu nível de experiência). Abaixo estão os tipos de dados coletados e suas finalidades:

Categoria	Dados pessoais	Finalidade
Dados de contato	Nome, email	Para contato e fins administrativos apenas.
Dados demográficos	Idade, gênero, país de residência	Para análise demográfica da amostra de participantes.
Experiência	Função no trabalho e anos de experiência, anos de experiência em programação, familiaridade com assistentes baseados em LLM	Para análise demográfica da amostra de participantes e para explorar como perfis de experiência podem se relacionar com resultados específicos do estudo.
Estilo cognitivo	Motivação, estilo de processamento de informações, autoeficácia em computação, aversão ao risco, experimentação	Para explorar como estilos cognitivos podem se relacionar com resultados específicos do estudo.
Dados da sessão	Gravações de áudio e tela, interações no chat, código escrito, preferências e percepções sobre a sessão	Para caracterizar sua interação com o Copilot. As gravações de áudio e tela serão deletadas após a transcrição.

O que acontecerá com meus dados?

As informações que você fornecer nos questionários e na sessão serão tratadas com máximo cuidado e acessíveis somente por pesquisadores autorizados da Universidade Tecnológica Federal do Paraná (UTFPR), Brasil. Os dados anonimizados serão compartilhados com pesquisadores autorizados da Radboud University, Países Baixos.

Gravações de áudio e tela serão deletadas após a transcrição. Todos os demais dados coletados serão anonimizados e disponibilizados na forma de uma base de dados da pesquisa. Os dados passarão por nova etapa de anonimização a fim de remover qualquer informação que possa identificar os participantes. No entanto, não é possível realizar a anonimização dos dados antes de enviar a atividade, realizada durante a sessão, para o GitHub.

Foram tomadas precauções para evitar que alguns dos serviços (de terceiros) usados durante este estudo registrem dados do seu navegador, como cookies e endereço IP. Como isso não pode ser evitado para os serviços do GitHub, recomenda-se o uso de uma janela de navegação anônima.

Dados pessoais: Para proteger sua privacidade, seus dados pessoais serão armazenados de forma pseudonimizada, ou seja, separados dos dados da pesquisa e protegidos por um arquivo-chave criptografado, que também é protegido por senha. Apenas pesquisadores autorizados da Universidade Tecnológica Federal do Paraná terão acesso a esse arquivo. Nenhuma outra parte envolvida na pesquisa receberá quaisquer dados que possam ser rastreados até você.

Período de retenção e armazenamento: Os dados pessoais coletados por meio da pesquisa serão guardados em instalações de armazenamento da Universidade Tecnológica Federal do Paraná. O link entre seus dados pessoais e seus dados de pesquisa será mantido por no máximo 1 mês após a conclusão da pesquisa. Você pode solicitar a exclusão dos seus dados até 1 mês após sua participação enviando um e-mail para bruno.1990@alunos.utfpr.edu.br. Após esse prazo, os dados não poderão mais ser excluídos pois já terão sido usados na análise ou armazenados apenas de forma anônima. Isso significa que não saberemos mais quais dados pertencem a você.

Os dados anonimizados serão mantidos por pelo menos 10 anos após a conclusão do estudo. Esses dados anonimizados serão disponibilizados publicamente na forma de uma base de dados da pesquisa. Não podemos garantir que terceiros usarão o conjunto de dados anonimizados apenas para fins científicos.

Você tem alguma dúvida sobre a pesquisa?

Você pode entrar em contato com o pesquisador responsável, Bruno Alves em bruno1990@alunos.utfpr.edu.br se tiver alguma dúvida.

TERMO DE CONSENTIMENTO

para participação em pesquisa científica: Diversidade Cognitiva nas Interações de Desenvolvedores(as) com Assistentes Conversacionais Baseados em Modelos de Linguagem de Grande Escala (LLM – Large Language Models)

Eu confirmo que:

• Fui informado(a) satisfatoriamente sobre a pesquisa por meio deste documento;

• Li as informações e tive a oportunidade de fazer perguntas. Minhas dúvidas foram esclarecidas;

• Tive tempo suficiente para pensar antes de decidir participar;

• Estou participando voluntariamente.

Eu entendo que:

• Tenho o direito de retirar meu consentimento a qualquer momento, sem precisar justificar minha decisão e sem sofrer quaisquer consequências adversas, entrando em contato com Bruno Alves de Oliveira pelo e-mail bruno.1990@alunos.utfpr.edu.br.

• Tenho o direito de solicitar a exclusão dos meus dados de pesquisa até 1 mês após minha participação;

• Os seguintes dados pessoais serão coletados, utilizados e armazenados para este estudo: dados de contato, dados demográficos, perfil de experiência, perfil cognitivo e dados da sessão, incluindo gravações de áudio e tela;

• Tenho o direito de retirar meu consentimento para o (posterior) processamento dos meus dados pessoais; meus dados serão tratados de acordo com as regulamentações europeias de proteção de dados;

Eu concordo que:

• O formulário de consentimento assinado, contendo meus dados pessoais, será armazenado de forma segura por 10 anos;

• Os dados de pesquisa anonimizados obtidos no contexto deste estudo serão utilizados pelos pesquisadores para fins científicos e estarão disponíveis publicamente por pelo menos 10 anos, sem garantia de que terceiros os utilizarão exclusivamente para fins científicos;

• As autoridades de supervisão poderão inspecionar meus dados pessoais e de pesquisa para auditoria do estudo.

Eu concordo em participar do estudo.

Nome do participante:
Assinatura:
Data:

Nome do pesquisador:
Assinatura:
Data:

APÊNDICE B. QUESTIONÁRIO SOCIODEMOGRÁFICO

Seção A: Informações de contato

Por favor, insira suas informações de contato para que possamos entrar em contato e agendar a sessão. Observe que, conforme mencionado no termo de consentimento informado, essas informações são apenas para fins administrativos e serão descartadas após a conclusão do estudo.

A1. Qual é o seu nome?

A2. Qual é o seu endereço de email?

Seção B: Dados demográficos

B1. Qual é a sua idade?

- ☐ 18-24
- ☐ 25-34
- ☐ 35-44
- ☐ 45-54
- ☐ 55-64
- ☐ 65+

B2. Como você se identifica (gênero)?

- ☐ Feminino
- ☐ Masculino
- ☐ Não binário
- ☐ Prefiro não dizer
- ☐ Prefiro me autodescrever:

Prefiro me autodescrever:

B3. Qual é seu país de origem?

- ☐ Afeganistão

Uganda ☐

Ucrânia ☐

Emirados Árabes Unidos ☐

Reino Unido ☐

Estados Unidos ☐

Uruguai ☐

Uzbequistão ☐

Vanuatu ☐

Cidade do Vaticano ☐

Venezuela ☐

Vietnã ☐

Wallis e Futuna ☐

Saara Ocidental ☐

Iêmen ☐

Zâmbia ☐

Zimbábue ☐

Seção C: Escolaridade e ocupação

C1. O que melhor descreve a sua ocupação atual?

Estudante ☐

Empregado(a), tempo integral ☐

Empregado(a), meio período ☐

Trabalhador(a) independente, freelancer ☐

Não empregado(a), mas a procura ☐

Não empregado(a) e não estou a procura. ☐

Aposentado(a) ☐

C2. Atuamente, o que melhor descreve seu foco de estudo em programação?

- [ ] Ciência da Computação
- [ ] Engenharia de Software
- [ ] Tecnologia da Informação
- [ ] Engenharia da Computação
- [ ] Outras áreas relacionadas a desenvolvimento de software

C3. Atualmente, sua gradução na área de programção encontra-se em ?

- [ ] Graduando(a), no primeiro ano
- [ ] Graduando(a), não no primeiro ano
- [ ] Graduado(a)

C4. O que melhor descreve o seu trabalho atual (aquele que você realiza na maior parte do tempo)?

- [ ] Outros, não relacionados ao desenvolvimento de software
- [ ] Pesquisador(a) acadêmico
- [ ] Blockchain
- [ ] Engenheiro(a) de infraestrutura em nuvem
- [ ] Engenheiro(a) de dados
- [ ] Analista de dados
- [ ] Cientista de dados ou especialista em aprendizado de máquina
- [ ] Administrador(a) de banco de dados
- [ ] Designer
- [ ] Especialista DevOps
- [ ] Developer Advocate
- [ ] Developer Experience
- [ ] Desenvolvedor(a), AI
- [ ] Desenvolvedor(a), QA ou teste
- [ ] Desenvolvedor(a), back-end
- [ ] Desenvolvedor(a), desktop ou aplicativos empresarial
- [ ] Desenvolvedor(a), aplicativos embarcados
- [ ] Desenvolvedor(a), front-end

- [ ] Desenvolvedor(a), full-stack
- [ ] Desenvolvedor(a), jogos, gŕaficos
- [ ] Desenvolvedor(a), mobile
- [ ] Professor(a)
- [ ] engenheiro(a), site
- [ ] Engenheiro(a) de software
- [ ] Engenheiro(a) Hardware
- [ ] Marketing
- [ ] Gerente de produtos
- [ ] Gerente de projetos
- [ ] Cargo de Pesquisa e Desenvolvimento (Research & Development)
- [ ] Cientista
- [ ] Profissional em segurança
- [ ] Executivo(a) Senior (C-Suite, VP, etc.)
- [ ] Administrador(a) de sistemas

C5. O que melhor descreve seu último trabalho?

- [ ] Outros, não relacionados ao desenvolvimento de software
- [ ] Pesquisador(a) academico
- [ ] Blockchain
- [ ] Engenheiro(a) de infraestrutura, nuvem
- [ ] Engenheiro(a) de dados
- [ ] Analista de dados
- [ ] Cientista de dados ou especialista em aprendizado de máquina
- [ ] Administrador(a) de banco de dados
- [ ] Designer
- [ ] Especialista DevOps
- [ ] Developer Advocate
- [ ] Developer Experience
- [ ] Desenvolvedor(a), AI
- [ ] Desenvolvedor(a), QA or test

- [ ] Desenvolvedor(a), back-end
- [ ] Desenvolvedor(a), desktop or aplicações empresariais
- [ ] Desenvolvedor(a), aplicativos embarcados
- [ ] Desenvolvedor(a), front-end
- [ ] Desenvolvedor(a), full-stack
- [ ] Desenvolvedor(a), jogos
- [ ] Desenvolvedor(a), mobile
- [ ] Professor(a)
- [ ] Engenheiro(a), site
- [ ] Engenheiro(a), administrador
- [ ] Engenheiro(a), Hardware
- [ ] Marketing
- [ ] Gerente de produto
- [ ] Gerente de projeto
- [ ] Cargo de Pesquisa e Desenvolvimento (Research & Development)
- [ ] Cientista
- [ ] Profissional em segurança
- [ ] Executivo(a) Senior (C-Suite, VP, etc.)
- [ ] Administrador(a) de sistema

C6. Você é graduado(a) em algumas área relacionada ao desenvolvimento de software, como Ciência da Computação, Engenharia de Software, Tecnologia da Informação ou Engenharia da Computação?

- [ ] Não
- [ ] Sim, graduação
- [ ] Sim, pós-graduação

Seção D: Experiência

D1. Como você avaliaria sua proficiência nas seguintes habilidades e ferramentas?

	Sem experiência	Iniciante	Intermediário	Avançado	Especialista
Programação em geral	☐	☐	☐	☐	☐

	Sem experiência	Iniciante	Intermediário	Avançado	Especialista
Programação em JavaScript	☐	☐	☐	☐	☐
Desenvolvimento web	☐	☐	☐	☐	☐
React (biblioteca de interface do usuário)	☐	☐	☐	☐	☐
Visual Studio Code (editor de código)	☐	☐	☐	☐	☐

D2. Há quantos anos você programa ...

	Nenhum	<1 ano	1-2 anos	3-5 anos	5-10 anos	10+ anos
No total, incluindo contextos profissionais e não profissionais (educação e programação pessoal)?	☐	☐	☐	☐	☐	☐
Profissionalmente (como parte do seu trabalho), NÃO incluindo contextos não profissionais?	☐	☐	☐	☐	☐	☐
Em JavaScript, incluindo contextos profissionais e não profissionais?	☐	☐	☐	☐	☐	☐
No desenvolvimento web, incluindo contextos profissionais e não profissionais?	☐	☐	☐	☐	☐	☐
Em React, incluindo contextos profissionais e não profissionais?	☐	☐	☐	☐	☐	☐
Em Visual Studio Code, incluindo contextos profissionais e não profissionais?	☐	☐	☐	☐	☐	☐

D3. Com que frequência você utiliza funcionalidades de chat de assistentes baseados em IA, incluindo assistentes de programação (como o GitHub Copilot) e assistentes de uso geral (como o ChatGPT), para tarefas relacionadas à programação, como depuração, geração de código e aprendizado de conceitos gerais de programação?

- ☐ Nunca
- ☐ Raramente
- ☐ Às vezes
- ☐ Frequentemente
- ☐ Quase sempre

D4. Você pode explicar brevemente quais assistentes você utiliza e para quais tipos de tarefas de programação?

APÊNDICE C. ESTILO COGNITIVO

Seção A: Estilos cognitivos

Por favor, indique até que ponto você concorda com as seguintes afirmações sobre o uso de ferramentas de suporte à programação.

Essas ferramentas auxiliam as pessoas desenvolvedoras automatizando tarefas, detectando erros, melhorando a qualidade do código e aumentando a produtividade. Exemplos incluem:

Ambientes de desenvolvimento integrados (Integrated development environments - IDEs) como o Visual Studio, IntelliJ IDEA e Eclipse Ferramentas de depuração como Chrome DevTools e WinDbg Interfaces gráficas de controle de versão como GitHub Desktop e Sourcetree Ferramentas de análise de código como SonarQube e o inspetor integrado do PyCharm Assistentes baseados em IA, como GitHub Copilot e ChatGPT

A1. Sou capaz de utilizar ferramentas de apoio à programação quando...

	1 - Discordo totalmente	2	3	4	5 - Nem concordo nem discordo	6	7	8	9 - Concordo completamente
... Tenho apenas a ajuda incorporada do assistente.	☐	☐	☐	☐	☐	☐	☐	☐	☐
... Já vi alguém utilizar antes de o experimentar.	☐	☐	☐	☐	☐	☐	☐	☐	☐
... Ninguém está por perto para ajudar se eu precisar.	☐	☐	☐	☐	☐	☐	☐	☐	☐
... Alguém me ajudou a começar.	☐	☐	☐	☐	☐	☐	☐	☐	☐
... Alguém me mostrou como fazer primeiro.	☐	☐	☐	☐	☐	☐	☐	☐	☐
... Já usei tecnologia semelhante antes, para fazer a mesma tarefa.	☐	☐	☐	☐	☐	☐	☐	☐	☐
... Nunca usei nada parecido antes.	☐	☐	☐	☐	☐	☐	☐	☐	☐

A2.

	1 - Discordo completamente	2	3	4	5 - Nem concordo nem discordo	6	7	8	9 - Concordo completamente
Eu não estou confiante na minha capacidade de usar e aprender ferramentas de programação. Tenho outros pontos fortes.	☐	☐	☐	☐	☐	☐	☐	☐	☐
Eu reservo tempo para explorar ferramentas de programação que não são essenciais para o meu trabalho.	☐	☐	☐	☐	☐	☐	☐	☐	☐
Um dos motivos pelos quais invisto tempo e dinheiro em ferramentas de programação é porque isso me ajuda a parecer bem aos olhos dos meus colegas.	☐	☐	☐	☐	☐	☐	☐	☐	☐

	1 - Discordo completamente	2	3	4	5 - Nem concordo nem discordo	6	7	8	9 - Concordo completamente
É divertido experimentar novas ferramentas de programação que ainda não estão disponíveis para todos, como participar de programas beta para testar ferramentas de programação ainda em desenvolvimento.	☐	☐	☐	☐	☐	☐	☐	☐	☐
Eu gosto de descobrir os recursos e capacidades menos conhecidos das ferramentas de programação que utilizo.	☐	☐	☐	☐	☐	☐	☐	☐	☐
Eu exploro áreas das ferramentas de programação antes de precisar usá-las.	☐	☐	☐	☐	☐	☐	☐	☐	☐
Eu nunca fico satisfeito com as configurações padrão das minhas ferramentas de programação.	☐	☐	☐	☐	☐	☐	☐	☐	☐
Eu quero fazer as coisas certas logo na primeira vez, então, antes de decidir como agir, reúno o máximo de informações possível.	☐	☐	☐	☐	☐	☐	☐	☐	☐
Sempre faço uma pesquisa extensa e comparo opções antes de fazer compras importantes	☐	☐	☐	☐	☐	☐	☐	☐	☐
Quando é preciso tomar uma decisão, é importante para mim reunir detalhes relevantes antes de decidir, para ter certeza da direção que estamos seguindo.	☐	☐	☐	☐	☐	☐	☐	☐	☐
Eu evito botões ou seções 'avançadas' em ferramentas de programação.	☐	☐	☐	☐	☐	☐	☐	☐	☐
Eu evito atividades que são perigosas ou arriscadas.	☐	☐	☐	☐	☐	☐	☐	☐	☐
Apesar dos riscos, eu uso recursos em ferramentas de programação que ainda não foram comprovadamente funcionais.	☐	☐	☐	☐	☐	☐	☐	☐	☐