Inteligência Artificial Generativa como Auxílio para Melhoria das Estimativas de Esforço de Projetos de Software em Métodos Ágeis

Bárbara Mattioly Andrade[1], **Laura Enísia Rodrigues Melo**[2]

[1]Instituto de Ciências Exatas e Informática
Pontifícia Universidade de Minas Gerais (PUC Minas)
Belo Horizonte – MG – Brasil

{barbara.andrade, laura.enisia}@sga.pucminas.br

Abstract. *Effort estimation in agile software projects is hindered by the subjectivity of methods such as Planning Poker, which can compromise planning accuracy. This study investigates how Artificial Intelligence (AI) can support effort prediction using public datasets with effort estimates. Natural language processing techniques (Gemini API) and statistical models were applied to identify patterns between task characteristics and actual effort. The evaluation metrics include mean absolute error, mean relative deviation, correlation, and dispersion. Results show that AI is more consistent for lower-complexity tasks, with average errors between 1.83 and 2.77 points, although its accuracy decreases as complexity increases.*

Resumo. *A estimativa de esforço em projetos ágeis de software é prejudicada pela subjetividade de métodos como Planning Poker, que podem comprometer a precisão do planejamento. Desse modo, esse estudo investiga como a Inteligência Artificial (IA) pode auxiliar a previsão de esforço, utilizando datasets públicos com estimativas de esforço.Foram aplicadas técnicas de processamento de linguagem natural (API do Gemini) e modelos estatísticos para identificar padrões entre características das tarefas e o esforço real. As métricas avaliadas incluem erro médio absoluto, desvio médio relativo, correlação e dispersão. Os resultados mostram que a IA apresenta maior consistentência em tarefas de menor complexidade, com erro médio entre 1,83 e 2,77 pontos, embora sua precisão diminui conforme a complexidade aumenta.*

Bacharelado em Engenharia de Software - PUC Minas
Trabalho de Conclusão de Curso (TCC)

Orientador de conteúdo (TCC I): João Pedro Oliveira Batisteli - joaobatisteli@pucminas.br
Orientador de conteúdo (TCC I): Joana Souza - joanasouza@pucminas.br
Orientador de conteúdo (TCC I): Leonardo Vilela - leonardocardoso@pucminas.br
Orientador acadêmico (TCC I): Cleiton Tavares - cleitontavares@pucminas.br
Orientador do TCC II: João Pedro Oliveira Batisteli - joaobatisteli@pucminas.br

Belo Horizonte, 23 de Novembro de 2025.

1. Introdução

O gerenciamento de projetos de software desempenhou um papel fundamental na organização, planejamento, execução e controle das atividades necessárias para o desenvolvimento de um sistema. Um aspecto importante é a estimativa de esforço, responsável por prever o trabalho necessário em termos de tempo e recursos, impactando prazos, custos e previsibilidade [Aziz et al. 2024]. Com o Manifesto Ágil [Manifesto para Desenvolvimento Ágil de Software 2001], metodologias ágeis como o Scrum tornaram-se populares por sua capacidade de adaptação e entrega incremental [Hoda et al. 2018]. Nessas metodologias, a estimativa é realizada de maneira iterativa, por meio de técnicas como *Planning Poker*, *T-Shirt Sizing* e o método dos três pontos [Alhamed and Storer 2021]. Contudo, métodos como o *Planning Poker*, por exemplo, podem introduzir subjetividade e variações nas estimativas, resultando em processos suscetíveis a erros e comprometendo a precisão do planejamento [Terlapu et al. 2024]. Portanto, foi essencial buscar estratégias que aumentassem a confiabilidade dessas avaliações.

Estudos anteriores destacaram a dificuldade de padronizar e alcançar consenso na realização de estimativas de esforço, fortemente influenciadas pela experiência dos desenvolvedores, pela complexidade das tarefas e pelo volume de demandas [Alhamed and Storer 2021]. Há também trabalhos que exploram o uso do ChatGPT para extrair informações de modelos conceituais com o intuito de apoiar processos de estimativa [Arman et al. 2023]. No entanto, esses estudos não investigaram o uso de um *prompt* base padronizado aplicado diretamente a *Large Language Models* (LLMs) para apoiar a geração de estimativas de forma consistente, precisa e replicável. Assim, permanece o **desafio de avaliar a precisão presente nas estimativas em processos ágeis utilizando ferramentas baseadas em LLMs, o que pode comprometer o cumprimento dos prazos planejados.**

Com os avanços da tecnologia, a Inteligência Artificial (IA) consolidou-se como uma oportunidade de tornar o processo de estimativas mais preciso e automatizado. Algoritmos de aprendizado de máquina foram capazes de analisar dados de projetos e identificar padrões que auxiliaram na previsão e na estimativa de esforço [Sarro et al. 2022]. Nesse contexto, a IA atuou como uma ferramenta de suporte e como facilitadora para que equipes ágeis tomassem decisões mais precisas e consolidadas. Dessa forma, a adoção de técnicas baseadas em IA permitiu que as estimativas se tornassem mais precisas e adaptáveis, proporcionando maior confiança para as equipes tomarem decisões ao longo do ciclo de vida do software [Fu and Tantithamthavorn 2023].

Diante desse cenário, o objetivo geral deste trabalho foi **explorar como o uso de IA pôde auxiliar na previsão da quantidade de esforço necessária para concluir histórias de usuários ou tarefas em projetos de software que aplicaram metodologia ágil**. Para atingir esse objetivo, foram desenvolvidos os seguintes objetivos específicos: I) Realizar o tratamento e limpeza de dois conjuntos de dados reais de histórias de usuário provenientes de projetos ágeis e identificar atributos importantes para a tarefa de previsão de esforço; II) Explorar e avaliar o uso de modelo de linguagem baseado em IA, com foco no modelo Gemini, na geração de estimativas de esforço. III) Avaliar em quais condições o modelo de IA apresenta melhor desempenho na estimativa de esforço e identificar os cenários em que sua precisão diminui, de modo a compreender os limites e as condições de uso mais adequadas.

Os resultados desta pesquisa demonstraram que a IA produziu estimativas mais próximas do valor real nas tarefas de menor complexidade, apresentando erro médio de 1,83 em um dos *datasets* utilizados na execução da pesquisa e 2,77 no outro, considerando histórias de até 5 *story points*. Esse desempenho contrastou com o observado em tarefas mais complexas, nas quais o erro aumentou de forma significativa nas faixas de 8 a 13 *story points*. Além disso, verificou-se que o tamanho da descrição textual exerceu influência aproximadamente nula na precisão das estimativas, com correlações próximas de zero. Esses resultados evidenciaram que a precisão do modelo varia conforme a complexidade e a diversidade das tarefas. Desse modo, quanto maior a complexidade e a incerteza da tarefa, maior a variação entre o estimado e o realizado.

Além desta seção introdutória, este trabalho foi organizado da seguinte forma. A Seção 2 apresentou a fundamentação teórica do estudo. A Seção 3 contextualizou a área e discutiu os trabalhos relacionados. A Seção 4 descreveu os materiais e métodos utilizados na pesquisa. A Seção 5 expôs os resultados obtidos a partir da análise dos dados coletados. A Seção 6 evidenciou as ameaças à validade. Por fim, a Seção 7 apresentou as conclusões e as sugestões para trabalhos futuros.

2. Fundamentação Teórica

Nesta seção, foram descritos os principais tópicos que contextualizaram e fundamentaram a pesquisa. Na subseção 1, foi analisado o processo de estimativas de esforço no contexto do Scrum, abordando conceitos como *story points*, esforço ideal e *velocity* e na subseção 2, foram apresentados os conceitos relacionados a engenharia de *prompt*.

2.1. Estimativas de esforço no Scrum

Estimativas de esforço correspondem aos processos de prever o quanto de trabalho será necessário para concluir uma tarefa. Essa prática representa um grande desafio para equipes de desenvolvimento ágil, dada a dificuldade em prever com precisão o tempo necessário para concluir uma tarefa. Segundo Shore e Warden (2007), uma abordagem eficaz consiste em estimar o esforço ideal, o tempo necessário para completar uma tarefa em condições ideais, sem interrupções utilizando unidades chamadas de *story points*. Essas estimativas não refletem diretamente o tempo em calendário, mas sim a complexidade relativa das tarefas, o que permite que as equipes mantenham consistência mesmo diante de variações externas.

Com base nesse modelo, Shore e Warden (2007) apresentaram o conceito de *velocity*, utilizado pelas equipes para representar a média de *story points* concluídos nas iterações anteriores, para planejar com mais realismo o volume de trabalho a ser assumido. Essa abordagem permitiu identificar a capacidade produtiva da equipe ao longo do tempo, funcionando como um parâmetro para limitar a quantidade de tarefas selecionadas para o *sprint*. No entanto, a eficácia desse processo depende diretamente da precisão das estimativas anteriores, reforçando a importância de métodos que reduzam a subjetividade e aumentem a previsibilidade nas decisões de planejamento.

As práticas de estimativa de esforço, como o uso de *story points* e *velocity*, são fundamentais para o funcionamento do Scrum, uma abordagem ágil que organiza o trabalho em ciclos iterativos. Sommerville (2015) apresenta o Scrum como uma metodologia que enfatiza o gerenciamento iterativo do desenvolvimento de software, podendo ser complementada por práticas técnicas de outras metodologias como o *Extreme Programming*

(XP). Um dos fundamentos do Scrum é a realização de planejamentos em dois níveis, o planejamento do projeto e o planejamento da iteração, que possui uma perspectiva de curto prazo e foca na definição do próximo incremento. Esse planejamento é baseado na priorização do *product backlog* e na seleção das tarefas mais relevantes para o próximo *sprint*, as quais devem ser estimadas pela equipe de desenvolvimento para garantir a entrega dentro do prazo e dos recursos disponíveis.

2.2. Engenharia de *Prompt*

Engenharia de *prompt* consiste na formulação estratégica de instruções claras, específicas e estruturadas com o objetivo de orientar o comportamento dos LLMs e gerar respostas mais assertivas, relevantes e contextualizadas. Segundo Liu et.al (2023), essa nova forma de interação com a IA, denominada *Pre-train, Prompt, and Predict*, representa uma mudança importante no campo do processamento de linguagem natural. Em vez de ajustar os modelos com grandes volumes de dados como nas abordagens tradicionais, os *prompts* permitem que o modelo gere respostas úteis sem precisar ser reconfigurado. Com isso, torna-se possível aplicar os LLMs em contextos como estimativas de esforço em projetos ágeis, oferecendo maior consistência e replicabilidade.

Pequenas variações no texto do *prompt* podem causar mudanças significativas no desempenho do modelo, tornando o processo de construção e elaboração uma tarefa sensível. De acordo com Liu et al. (2023), mesmo profissionais experientes podem ter dificuldade em formular *prompts* ideais de forma manual, especialmente em tarefas complexas. Segundo a OpenAI (2024), devido à natureza não determinística dos modelos de linguagem, a formulação de *prompts* eficazes exige uma abordagem iterativa, ressaltando que instruções bem definidas, com estrutura e objetivos claros e precisos, são fundamentais para orientar corretamente o modelo e obter respostas consistentes, relevantes e alinhadas à tarefa proposta.

3. Trabalhos Relacionados

O uso de técnicas de aprendizado de máquina para melhorar a qualidade das estimativas de esforço em projetos de software é um tema que tem sido objeto de vários estudos na literatura acadêmica. *Catak et.al* (2024) propuseram um modelo de estimativa baseado em Processamento de Linguagem Natural (PLN), com o objetivo de aumentar a precisão e reduzir incertezas em ambientes ágeis. O estudo aplicou técnicas de pré-processamento e rotulagem para preparar os dados, além de utilizar *clustering* semissupervisionado, usando *K-means* para filtrar ruídos. Os resultados indicaram que a filtragem de ruídos melhorou significativamente o desempenho do modelo, atingindo 96,8% de precisão. A principal contribuição desse trabalho consistiu em demonstrar que o ruído nos dados impacta diretamente a performance preditiva dos modelos. Esse estudo foi utilizado como base para a etapa de preparação e filtragem de dados da presente pesquisa, embora o foco se diferenciasse ao extrair uma base de dados própria de projetos do *GitLab*, ferramenta amplamente utilizada em contextos ágeis, e ao aplicar pontos de referência para a realização das estimativas

De maneira complementar, *Fu. M e Tantithamthavorn. C* (2023) apresentaram o *GPT2SP*, uma abordagem baseada em *Transformers* com o modelo pré-treinado *GPT-2*, que teve como objetivo melhorar a precisão e qualidade das estimativas de pontos de

história. O modelo foi treinado com dados de mais de 23 mil tarefas de 16 projetos de código aberto. Os resultados indicaram que o *GPT2SP* superou métodos convencionais com uma precisão 34%-57% superior em estimativas dentro de projetos. Este artigo se relacionou ao presente estudo ao explorar o uso de técnicas de inteligência artificial para aprimorar estimativas em contextos ágeis. No entanto, diferiu-se metodologicamente ao propor um novo modelo preditivo baseado principalmente no *GPT-2*, enquanto este trabalho adotou a integração de métodos de estimativa ágeis, com diferentes técnicas de IA.

Arman et al. (2023) propuseram uma abordagem para estimativa de esforço em projetos de software, buscando superar limitações de métodos tradicionais, como subjetividade, complexidade e coleta manual de dados. O estudo utiliza modelos conceituais como diagramas *Business Process Model and Notation* (BPMN) para modelar o sistema, convertendo-os em trios semânticos a fim de identificar micro-serviços e estimar separadamente o esforço para o desenvolvimento de cada um. Na etapa final, aplicou-se engenharia de *prompt* juntamente ao *ChatGPT* para estimar, o número de linhas de código e as horas de desenvolvimento necessárias para cada micro-serviço. Diferentemente deste estudo, que realizou a estimativa de esforço a partir da decomposição do sistema em microserviços com base em modelos conceituais, o presente trabalho utilizou abordagens de IA e engenharia de *prompt* para estimativas.

Alhamed et. al (2021) exploraram a aplicação da técnica *Planning Poker* em larga escala por meio do método *Crowd Planning Poker* (CPP), com o objetivo de viabilizar estimativas de esforço em projetos de software sem depender exclusivamente da presença de especialistas. A pesquisa envolveu mais de 5.000 *crowd workers* e 39 tarefas de software o que demonstrou que, as estimativas da multidão obtiveram um erro médio relativo menor do que o de especialistas (239,83% contra 342%). Além disso, em 30 rodadas foi atingido um nível de consenso elevado, indicando confiabilidade na abordagem. Este estudo compartilhou com o presente trabalho, melhorar a qualidade das estimativas de esforço de forma escalável. Todavia, enquanto o presente trabalho adotou uma abordagem híbrida baseada em IA para aprimorar estimativas, Alhamed and Storer (2021) focaram na computação humana distribuída como alternativa escalável para contextos com limitações de especialistas.

Sarro et al. (2022) , apresentaram a abordagem de *Learning From Mistakes (LFM)* (Aprendizado com Erros), que teve como objetivo aprimorar a estimativa de esforços em projetos de software a partir do aprendizado de erros cometidos em estimativas anteriores. O estudo analisou 402 projetos e focou nas previsões incorretas realizadas por especialistas humanos, classificando os erros quanto ao tipo (baixo/alto), a gravidade (baixa, média ou alta) e magnitude (diferença entre a estimativa e o valor do esforço real). Para isso, os autores aplicaram técnicas de aprendizado de máquina, para identificar padrões recorrentes nos erros de estimativas cometidos. Como resultado principal, a metodologia LFM atingiu acurácia de 86% na identificação correta das características dos erros observados. Porém, diferentemente do presente trabalho, que propôs um modelo capaz de estimar o esforço de software com base nas características dos projetos, o trabalho de *Sarro et al.* (2022) teve como propósito prever os erros humanos nas estimativas, com o intuito de melhorar as estimativas futuras.

4. Materiais e Métodos

A pesquisa é classificada como quantitativa, de natureza exploratória e descritiva, uma vez que envolve a coleta e o tratamento de dados numéricos derivados das *User Stories (US)* e dos resultados da análise da IA. O caráter exploratório decorre da investigação sobre o potencial da ferramenta de IA para apoiar e aprimorar o processo de estimativa de esforço, enquanto o caráter descritivo refere-se à sistematização e a análise das relações entre os padrões identificados nas estimativas geradas com os atributos das US. A metodologia adotada neste estudo foi organizada em fases sequenciais conforme ilustrada na Figura 1 e detalhada na subseção 4.1.

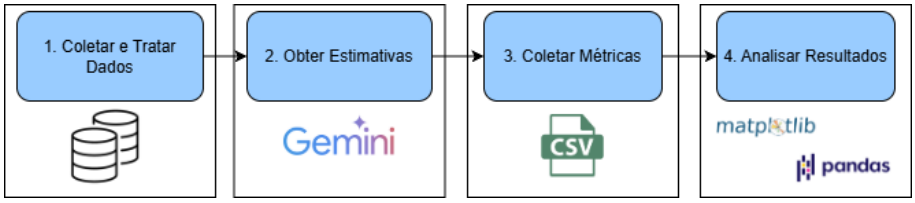

Figura 1. Diagrama do processo metodológico

4.1. Descrição das etapas

4.1.1. Coletar e Tratar Dados

Foram utilizados dois *datasets* distintos. O primeiro foi o NeoDataset, composto por 34 projetos de desenvolvimento de software provenientes de repositórios do GitLab, com o total de 20.474 histórias de usuário com estimativas em *story points*. O segundo *dataset* foi obtido a partir de um estudo publicado por Choetkiertikul et al. (2019), que contém 16 projetos e 23.213 histórias de usuário. Inicialmente, ambos os conjuntos passaram por um processo de extração, filtragem e limpeza.

No NeoDataset, foram removidos itens do tipo *bug* e consideradas apenas histórias com estado *closed* e que possuíam estimativa registrada. Para garantir que as descrições fossem suficientemente detalhadas, considerou-se apenas conteúdos com tamanho igual ou maior à mediana de cada projeto, após a remoção de marcações HTML. Essa escolha justifica-se pela grande variação observada na extensão das descrições, prevenindo que textos curtos comprometessem a análise. A Figura 2 apresenta a mediana do tamanho das descrições por projeto, o que destaca as diferenças entre os conjuntos analisados. Além disso, foram selecionados apenas projetos com mais de 100 histórias de usuário, de modo a garantir um volume representativo para extração de padrões. Os registros filtrados foram exportados para um arquivo CSV, enquanto os registros inconsistentes foram armazenadas separadamente para controle de erros.

No segundo conjunto de dados, obtido de Choetkiertikul et al. (2019), foram selecionados apenas 7 dos 16 projetos originais, pois somente esses utilizavam a escala de Fibonacci como referência de estimativa de esforço. Esse critério foi adotado visando compatibilidade com o comportamento observado no modelo de IA. Após a filtragem, os projetos selecionados foram consolidados em arquivos CSV e preparados para as etapas subsequentes. Registros inconsistentes foram igualmente armazenados para análises adicionais.

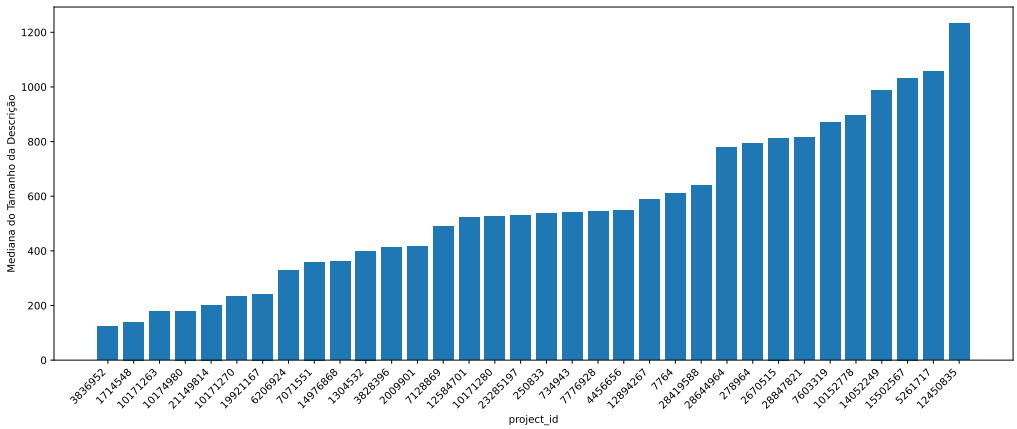

Figura 2. Mediana do tamanho das descrições, por projeto em NeoDataset

4.1.2. Obter Estimativas

Nesta etapa, foi estabelecido um conjunto de pontos de referência utilizados como parâmetro para guiar o modelo durante o processo de estimativa. Esses pontos forneceram uma base padronizada, como por exemplo, considerar que um ponto de referência, no contexto de *backend*, equivale à criação de um *endpoint* simples. Em seguida, foi elaborado um *prompt* disponível no (Apêndice A), estruturado com foco em clareza e alinhamento às boas práticas de Engenharia de *Prompt*. Para isso, foi utilizado o modelo *Gemini*, no qual foram submetidas, individualmente, as descrições das histórias de usuário. Para cada entrada, o modelo retornou o valor estimado em *story points* juntamente com uma breve descrição com a justificativa estimada, considerando os pontos de referência fornecidos.

4.1.3. Coletar Métricas

Após a execução das estimativas, todos os resultados gerados pela IA foram coletados e armazenados em arquivos no formato CSV. As variáveis registradas incluíram o identificador da tarefa, a estimativa em pontos atribuída pelo modelo Gemini, o tipo de tarefa identificado e uma breve justificativa para a estimativa gerada. Essa estrutura possibilitou a organização das evidências necessárias para as análises subsequentes, garantindo a rastreabilidade dos resultados e a preparação adequada dos dados para a aplicação das técnicas estatísticas.

Para garantir a consistência das análises, os dados passaram por etapas de tratamento que incluíram a remoção de tarefas cuja estimativa real era igual a zero e a detecção de *outliers* com base no método *Interquartile Range* (IQR). Em seguida, foram calculadas métricas gerais, como o erro médio absoluto, desvio médio relativo e a correlação entre o tamanho das descrições textuais e o erro absoluto. Adicionalmente, as tarefas foram agrupadas por faixas de *story points* e por tamanho da descrição, possibilitando análises segmentadas e a construção de visualizações gráficas que evidenciaram padrões e tendências nos dados.

4.1.4. Analisar Resultados

Por fim, foi conduzida uma análise quantitativa por meio da aplicação de técnicas estatísticas sobre os dados tratados. Além disso, foram elaboradas visualizações gráficas utilizando recursos da biblioteca *Pandas*, com o objetivo de avaliar a relação entre os valores reais e os valores estimados pela IA. Essas análises consideraram a segmentação dos dados por nível de detalhamento da descrição e faixa de *story points*, permitindo identificar padrões de comportamento do modelo e fatores que influenciam a precisão das estimativas geradas.

Para complementar a análise quantitativa, também foi realizada uma análise qualitativa de casos individuais, contemplando tarefas com estimativas próximas ao valor real e tarefas com grandes divergências. Essa abordagem possibilitou avaliar como o modelo se comporta diante de diferentes níveis de complexidade e detalhamento das descrições, além de evidenciar padrões que ajudam a interpretar os resultados e a identificar limitações do modelo.

4.2. Ferramentas e Métodos Utilizados

Para processar e tratar os dados provenientes das US, foi utilizada a linguagem de programação *Python* (versão 3.13), pela sua ampla adoção em ciência de dados e pela vasta disponibilidade de bibliotecas para análise e manipulação, como *pandas*. Para gerar as estimativas de esforço, empregou-se a API (do inglês *Application Programming Interfaces*) do *Gemini 2.5 Flash* selecionado por sua capacidade de lidar com grandes volumes de texto, além de ser uma alternativa economicamente viável, em comparação a outros modelos.

A estratégia de amostragem adotada nesta pesquisa foi a amostragem por conveniência, na qual utilizou-se 2 bases de dados. A primeira possuiu informações extraídos de um conjunto público de repositórios do *GitLab*, o qual disponibiliza um volume expressivo de dados relevantes, como descrição de US e seus respectivos *Story Points*, no qual atendeu aos requisitos necessários do experimento. A segunda base, reuniu dados de projetos ágeis, no qual possuía estimativas realizadas com base na escala de Fibonacci. A escolha dessas fontes deve-se pela disponibilidade e pertinência dos dados para o estudo realizado.

Para a análise desses dados, foi utilizada a regressão linear, com o objetivo de modelar por meio de uma equação a relação entre o esforço estimado e as variáveis extraídas das histórias de usuário. Essa técnica permite identificar padrões e realizar previsões mais precisas ao representar, por meio de uma equação, a relação entre a variável dependente (esforço estimado) e variáveis independentes (atributos das US).

A técnica de mineração de dados adotada foi o processo de *Extract, Transform, Load (ETL)*. Essa técnica consiste em três principais etapas: I) extração dos dados a partir da base de dados, II) transformação, que consiste na limpeza e seleção das informações necessárias, e III) carregamento, no qual os dados tratados são armazenados em um arquivo *CSV* para utilização nas etapas subsequentes. Esse método foi escolhido pela sua eficácia em garantir a qualidade dos dados antes do carregamento, o que contribuiu para otimizar o uso de recursos computacionais e assegurar a confiabilidade das análises.

4.3. Métricas de Avaliação

Para avaliar a eficácia do modelo de estimativa de esforço baseado em inteligência artificial neste trabalho, foi utilizada a métrica Erro Médio Absoluto (MAE, do inglês *Mean Absolute Error*). Essa métrica mensurou o quão próximas (em média) as estimativas geradas pelo modelo estão dos valores reais observados, levantado através do tempo em que uma tarefa demorou para ser concluída e sua respectiva conversão em pontos. Quanto menor o valor de MAE, melhor o desempenho do modelo de estimativa, uma vez que ele representa a média dos erros absolutos entre previsões e valores reais, no qual reduziu o impacto de valores extremos (*outliers*).

A fórmula do MAE é definida pela Equação 1.

$$MAE = \frac{1}{n} \sum_{i=1}^{n} \left| Y_i - \hat{Y}_i \right| \tag{1}$$

onde n corresponde ao número total de tarefas analisadas, Y_i a estimativa real da história, $\hat{Y}_i$ ao valor estimado pelo modelo de IA para a mesma tarefa e $\left| Y_i - \hat{Y}_i \right|$ ao erro absoluto da previsão.

Foram empregadas métricas complementares para aprofundar a análise do desempenho do modelo. O Desvio Médio Relativo (RMD, do inglês *Relative Mean Deviation*) foi utilizado para identificar possíveis vieses sistemáticos nas estimativas, permitindo avaliar tendências de superestimação ou subestimação do esforço em *story points*. Adicionalmente, foi analisada a correlação entre o comprimento das descrições textuais das tarefas e o erro absoluto das estimativas, com o objetivo de investigar se o nível de detalhamento da descrição influencia a acurácia do modelo.

5. Resultados

Nesta Seção são apresentados os resultados obtidos por meio da execução da metodologia descrita na Seção 4.4, compreendendo a caracterização dos dados, apresentação e análise dos resultados obtidos e discussão.

5.1. Caracterização dos *Dados*

O primeiro conjunto de dados analisado contém informações provenientes de tarefas de projetos de software, totalizando 14.443 registros. Desses, 8 registros apresentavam estimativa original igual a zero, sendo todos removidos do conjunto final. Após o pré-processamento e a remoção de *outliers*, o total de registros utilizados para análise foi de 12.831 tarefas. A detecção de *outliers* foi realizada pelo método IQR, no qual utilizou-se a regra de *Tukey* (1,5 × IQR). Foram considerados outliers valores de *story points* reais fora do intervalo [-2,5; 9,5] e estimativas geradas pela IA fora do intervalo [0; 8], resultaram na remoção de 1.463 registros.

O segundo conjunto de dados é composto apenas por projetos que são indicados por utilizar a escala de *Fibonacci* para estimativas, o que totaliza 3.816 registros. O processo de detecção e remoção de *outliers* seguiu o mesmo procedimento aplicado ao primeiro conjunto. Foram considerados como *outliers* valores de *story points* originais fora do intervalo [-4,50; 15,50] e estimativas geradas pela IA fora do intervalo [0; 8].

Após a filtragem, 107 registros foram removidos, portanto 3.709 tarefas foram utilizadas na análise final.

As Figuras **??** e **??** apresentam a distribuição das tarefas por faixa de *story points* originais nos dois *datasets* antes da remoção de *outliers*, no qual evidenciou um padrão comum de concentração nas faixas inferiores. No primeiro conjunto, 10.024 tarefas se situavam entre 0–3 pontos, o que indica forte assimetria à esquerda. De forma semelhante, o segundo conjunto, apresentou 3.499 tarefas nas faixas entre 0 e 8 pontos, enquanto não foram encontrados valores com estimativa acima de 34 pontos. Em ambos os casos, foi observado que a maior parte das histórias avaliadas possuía complexidade relativamente baixa, independente da abordagem de estimativa adotada.

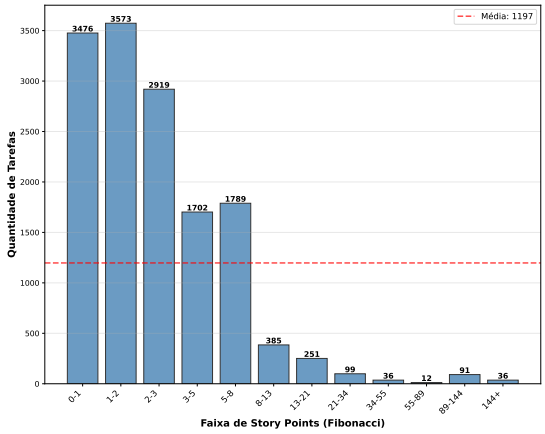

Figura 3. Distribuição de Tarefas por Faixa de *Story Points* Originais *NeoDataset*

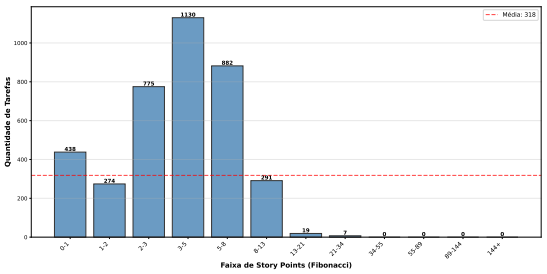

Figura 4. Distribuição de Tarefas por Faixa de *Story Points* Originais *Dataset* II

A Tabela 1 apresenta a distribuição das tarefas por quartis para ambos os conjuntos de dados. No primeiro *dataset*, observou-se que a escala utilizada pelas equipes não seguia estritamente a sequência de *Fibonacci* tradicionalmente empregada em métodos de estimativa ágil, como o *Planning Poker*. No segundo *dataset*, embora os projetos declarassem utilizar *Fibonacci*, foram identificadas 2 estimativas que não obedeciam à sequência e que, por isso, foram removidas da análise final. Essa inconsistência entre as escalas de estimativas presentes nos dados originais e a escala utilizada pela IA nas previsões pode ter contribuído para o aumento dos erros observados nas etapas seguintes, uma vez que diferenças estruturais na representação de esforço tendem a impactar diretamente o comportamento dos modelos.

Tabela 1. Distribuição de tarefas por quartis em ambos os *datasets*

***Dataset* 1**			
Quartil	Faixa Início	Faixa Fim	Quantidade
Q1	1	1	3335
Q2	1	2	3504
Q3	2	4	3192
Q4	4	9	2800

***Dataset* 2**			
Quartil	Faixa Início	Faixa Fim	Quantidade
Q1	1.0	3.0	1461
Q2	3.0	5.0	1109
Q3	5.0	8.0	866
Q4	8.0	13.0	273

5.2. Análise dos Resultados das Estimativas de Esforço

A avaliação de desempenho do modelo de IA foi realizada por meio de métricas estatísticas aplicadas às estimativas geradas, comparadas com os valores reais de *story points*. O modelo apresentou um MAE de aproximadamente 1,83 no primeiro *dataset*, indicando que, em média, a estimativa gerada pela IA difere do valor real em cerca de 1,83 pontos. Embora esse valor parecesse baixo, ao considerar a natureza da escala de Fibonacci utilizada pela IA, cada incremento representou um salto de complexidade. Portanto, o erro apesar de numericamente baixo, foi significativo no contexto da escala de estimativas ágeis. Já no *dataset* II, o MAE observado foi de aproximadamente 2,77, valor superior ao registrado no primeiro, o que indica uma discrepância ainda maior entre os valores previstos pela IA e os valores reais. Essa diferença reforçou que, mesmo em projetos que utilizam a escala de Fibonacci, a acurácia permaneceu limitada.

O Desvio Médio Relativo (RMD, do inglês *Relative Mean Deviation*) no primeiro conjunto foi de 1,01, ou seja, o erro médio correspondeu a 101% do valor real estimado. Esse resultado indicou que o modelo tendia a errar em média o mesmo valor que estava estimado, revelando desempenho limitado e uma tendência à superestimação das estimativas. Já no segundo *dataset*, o RMD obtido foi de aproximadamente -1,20, valor numericamente superior e com sinal negativo, indicando uma tendência sistemática de subestimação por parte da IA. Esse comportamento pode ser explicado pelas características do próprio conjunto de dados, que apresentou maior concentração de tarefas em faixas intermediárias e superiores de *story points*, além da adoção explícita da escala de Fibonacci. Como o *prompt* restringiu as previsões a valores discretos dessa sequência, o modelo tendeu a gerar estimativas mais conservadoras, especialmente em tarefas de maior complexidade, resultando em valores inferiores aos reais e em um RMD negativo.

As Figuras 5 e 6 apresentaram o gráfico de dispersão entre os valores reais e as estimativas produzidas pela IA em ambos os conjuntos. Observou-se que os pontos não se alinhavam à diagonal de 45° (linha ideal onde estimativa e valor real seriam iguais), o que forma um padrão de grade disperso. Isso indicou que, para um mesmo valor real, a IA tendeu a gerar diversas previsões distintas, o que reforçou a baixa precisão observada nas métricas.

A análise por faixas de *story points* reais apresentada nas Figuras 7 e 8 mostrou que o erro médio possuiu menor variação até a faixa de 5 pontos, aumentando de forma significativa em tarefas mais complexas. Observou-se que, no primeiro *dataset*, o MAE cresceu de aproximadamente 1,84 nas faixas menores para 4,14 na faixa de 8 a 13 pontos, com comportamento similar no segundo *dataset*, com aumento de 2,60 para 8,11 nessa mesma faixa. Esse comportamento reforçou que tarefas de maior complexidade foram mais suscetíveis a erros de estimativa, indicando a importância de avaliar a quebra dessas tarefas em unidades menores para melhorar a acurácia das previsões.

Outro aspecto relevante observado nos experimentos é em relação ao comportamento do modelo referente as restrições impostas pelo *prompt*. Embora os resultados não foram plenamente satisfatórios em termos de acurácia, o modelo atendeu rigorosamente às instruções fornecidas, produzindo exclusivamente valores de acordo com a sequência de Fibonacci. Ainda que o *prompt* solicitasse explicitamente o uso do *Planning Poker*, o modelo inferiu corretamente que essa técnica tradicional utiliza a sequência de Fibonacci para representar níveis crescentes de complexidade.

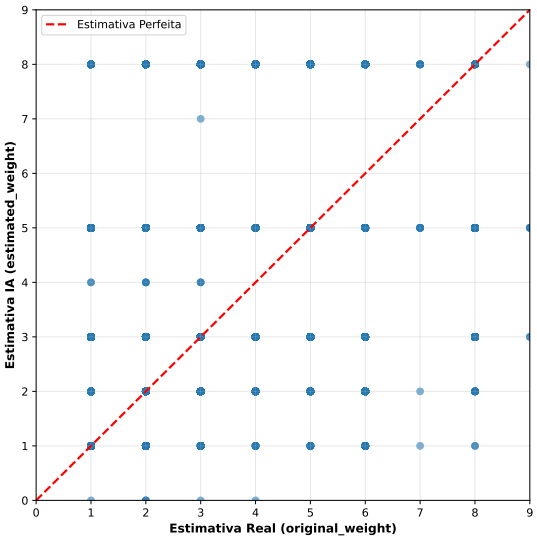

Figura 5. Representação da dispersão entre os valores reais e as estimativas geradas pela IA Neodataset

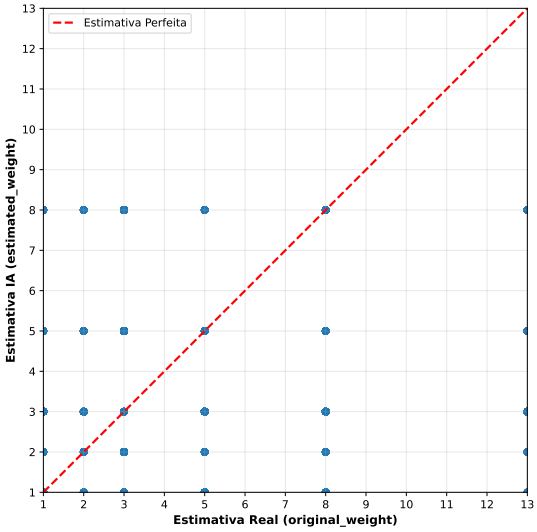

Figura 6. Representação da dispersão entre os valores reais e as estimativas geradas pela IA *Dataset* II

Esse comportamento evidenciou que, apesar das limitações na capacidade preditiva, a IA demonstrou sensibilidade ao contexto fornecido e aderência às regras implícitas associadas ao método de estimativa. Essa observação reforçou a importância do *design* do *prompt* como elemento central no direcionamento do comportamento do modelo. Ajustes mais refinados nas instruções, como explicitação de critérios de granularidade, contextualização técnica das tarefas ou fornecimento de exemplos estruturados podem ter potencializado a melhoria da qualidade das estimativas, apontando caminhos promissores para a evolução deste experimento em estudos futuros.

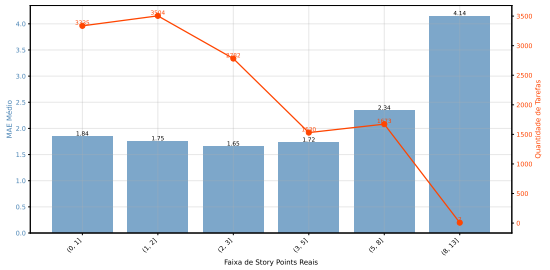

Figura 7. MAE e Quantidade de Tarefas por Faixa de *Story Points* NeoDataset

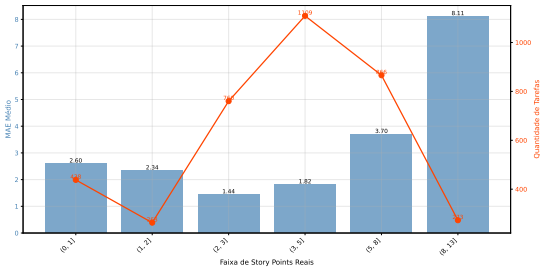

Figura 8. MAE e Quantidade de Tarefas por Faixa de *Story Points Dataset* II

As Figuras 9 e 10 mostraram a influência do tamanho da descrição textual das tarefas sobre o erro das estimativas. No primeiro *dataset*, a correlação entre o comprimento da descrição e o erro absoluto foi de 0,102, enquanto no segundo *dataset* o coeficiente calculado foi de –0,0044. Ambos os valores indicaram uma relação muito fraca (praticamente nula) entre as variáveis, sugerindo ausência de dependência linear entre o tamanho do texto e a precisão das estimativas geradas pelo modelo. Desse modo, com base nas

métricas analisadas, não foi possível afirmar que o tamanho da descrição influencia de forma significativa a assertividade das estimativas produzidas pela IA. Descrições mais longas não necessariamente fornecem informações adicionais para melhorar a acurácia das previsões, e descrições curtas também não pareceram prejudicar o desempenho de forma sistemática. Uma possível explicação é que o modelo pode estar capturando padrões mais semânticos do que estruturais, priorizando o conteúdo informativo e não a extensão do texto.

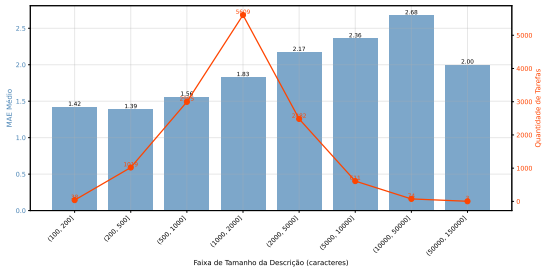

Figura 9. Representação do MAE e Quantidade de Tarefas por Tamanho de Descrição Neodataset

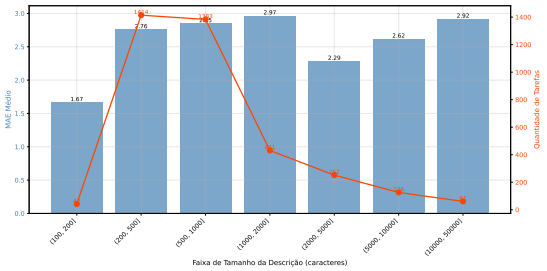

Figura 10. Representação do MAE e Quantidade de Tarefas por Tamanho de Descrição *Dataset* II

5.3. Discussão dos Resultados

Os resultados obtidos mostraram que a IA apresentou desempenho limitado na tarefa de estimar esforço com base apenas nas descrições textuais, sem considerar fatores como o contexto do projeto, o padrão histórico de estimativas do time, o domínio funcional da aplicação ou dependências técnicas entre tarefas. O aumento do erro em histórias de maior complexidade reforçou que tarefas grandes tenderam a concentrar múltiplos requisitos, cenários pouco refinados e baixa clareza semântica. Esse tipo de descrição dificultou o reconhecimento de padrões pelo modelo e pelo *prompt* utilizado, que dependeram diretamente da clareza e do nível de detalhamento das informações fornecidas.

As análises por faixas de *story points* reforçaram esse comportamento. No primeiro *dataset*, o erro médio cresceu de aproximadamente 1,84 nas tarefas de baixa complexidade para 4,14 na faixa de 8 a 13 pontos. No segundo *dataset*, o aumento foi ainda mais acentuado, passando de 2,60 para 8,11. Esse padrão evidenciou que tarefas mais complexas, com múltiplos requisitos e menor clareza, tornaram o texto mais difícil de interpretar pelo modelo. Assim, a baixa acurácia observada não decorreu apenas do tamanho das histórias, mas também da natureza do *dataset* e do nível de refinamento técnico, fator que reduziu a capacidade do modelo e do *prompt* de reconhecer padrões relevantes para a estimativa.

Além das métricas quantitativas, foram examinados exemplos específicos de tarefas com estimativas precisas e com grandes divergências, apresentados no Apêndice B. As estimativas corretas ocorreram principalmente em tarefas pequenas e bem delimitadas, enquanto as maiores discrepâncias sucederam em histórias complexas com múltiplas regras de negócio ou dependências técnicas implícitas. Esses casos indicaram que o baixo detalhamento técnico e o contexto limitaram a capacidade do modelo e do *prompt* de gerar estimativas precisas.

Na implementação adotada, cada tarefa foi enviada ao modelo de forma independente, sem compartilhamento de histórico entre estimativas, de modo que a IA não ajustou suas respostas com base em previsões anteriores nem aprendeu com padrões presentes no conjunto de dados, produzindo estimativas exclusivamente a partir do *prompt* e dos parâmetros fornecidos. Essa ausência de memória e de contexto contribuiu para a dispersão observada nas Figuras 5 e 6, nas quais os pontos não se aproximaram da diagonal ideal, evidenciando a dificuldade do modelo em se aproximar dos valores reais, especialmente em tarefas mais complexas. Nesse cenário, técnicas de *few-shot prompting* podem representar uma alternativa promissora ao fornecer exemplos representativos que auxiliem na calibração das respostas e na redução de ambiguidades durante o processo de estimativa.

6. Ameaças à validade

A realização deste estudo enfrentou ameaças à validade que podem impactar tanto a interpretação quanto a generalização dos resultados obtidos. Em relação à validade interna, a principal ameaça está relacionada à ausência de um contexto detalhado sobre os projetos analisados. Como o modelo não possuía acesso ao ambiente real de desenvolvimento nem ao contexto de regras de negócio do projeto, as limitações técnicas podem refletir em interpretações incompletas das tarefas, o que possivelmente contribui para estimativas subestimadas de esforço.

Quanto à validade externa, destaca-se a falta de padronização nas escalas de estimativa utilizadas pelas equipes dos projetos analisados. Embora o *Dataset II*, indicasse explicitamente o uso da sequência Fibonacci, foram identificadas histórias de usuário que não seguiam esse padrão, evidenciando inconsistências internas no próprio conjunto de dados. Desse modo, não foi possível determinar com precisão qual método foi efetivamente adotado e aplicado em todos os casos. Além disso, mesmo com o *prompt* baseado no método *Panning Poker*, que também baseia na sequência de Fibonacci, observou-se que a IA em algumas situações, gerou valores discrepantes em relação ao esperado. Essa divergência indica que a combinação de diferentes escalas de estimativas e possíveis restrições na interpretação do modelo podem ter contribuído para o aumento de erros detectados.

Em relação à validade de construção, destaca-se a natureza indeterminada dos modelos de linguagem, uma vez que pequenas variações na temperatura, no formato do *prompt* ou na ordem de processamento podem gerar resultados distintos entre execuções, dificultando a replicação rigorosa do experimento. Além disso, a escolha do modelo Gemini 2.5 Flash constitui outra ameaça, visto que foi projetado para fornecer respostas rápidas, não necessariamente para realizar análises profundas ou raciocínios elaborados, o que pode levar à simplificação excessiva do conteúdo das tarefas e à falha na captura de nuances semânticas relevantes para a estimativa de esforço em cada *user story*. Soma-se a isso a discrepância de idioma entre os artefatos analisados e as instruções fornecidas ao modelo, já que as descrições das tarefas estavam em inglês enquanto o *prompt* foi elaborado em português, podendo introduzir ruídos no processamento das informações, especialmente em descrições técnicas, e impactar negativamente a precisão das estimativas geradas.

7. Conclusão

O trabalho investigou como técnicas de Inteligência Artificial podem contribuir para o processo de estimativas de esforço em projetos de software que adotam metodologias ágeis. Para isso, foram coletados, filtrados e analisados dois conjuntos de dados reais de repositórios do GitLab e de um *dataset* acadêmico. Em seguida, elaborou-se um prompt com um conjunto padronizado de pontos de referência, que possibilitou ao modelo Gemini 2.5-flash criar estimativas com *story points* e uma breve justificativa. Por fim, foram aplicadas métricas estatísticas e análises gráficas para mensurar e comparar o desempenho da IA em relação aos valores reais ja estimados em cada história de usuário.

Os resultados demonstraram que, embora a IA tenha apresentado potencial para auxiliar o processo de estimativas, seu desempenho enfrentoulimitações relevantes. A análise dos dois *datasets* revelou que o modelo Gemini 2.5 Flash apresentou valores de MAE significativos, especialmente em tarefas mais complexas, além de variações expressivas entre superestimação e subestimação. Além disso, observou-se uma relação muito fraca entre o tamanho das descrições das tarefas e a exatidão das estimativas, no qual indica que descrições mais extensas não garantem, por si só, melhor desempenho do modelo. A dispersão observada nos gráficos também evidenciou que a IA não conseguiu estabelecer um padrão confiável entre os valores reais e estimados nas condições avaliadas. Contudo, essas limitações não demonstram que a IA seja ineficaz, mas ressaltam que sua aplicação em estimativas de esforço ainda demanda ajustes, calibração adequada e o desenvolvimento de abordagens mais robustas, apontando para um cenário promissor de aprimoramento futuro.

Como trabalhos futuros, recomenda-se explorar outros modelos de linguagem e modelos mais robustos, como *GPT*, *Gemini Pro* e *DeepSeek* afim de realizar uma análise comparativa. Sugere-se também ampliar a variedade de *datasets* utilizados, para explorar diferentes contextos e domínios para aumentar a diversidade dos cenários e melhorar a capacidade de generalização com a IA. Além disso, investigações que integrem a IA a outras técnicas de estimativas, como métodos de aprendizado supervisionado, podem contribuir para aumentar a precisão dos resultados. Ademais, seria importante aprofundar a análise sobre o impacto da Engenharia de *Prompt* por meio de testes que explorem variações estruturais. Nesse contexto, destaca-se a investigação do uso de técnicas de *few-shot prompting*, incorporando exemplos previamente estimados como forma de fornecer maior contexto ao modelo. Por fim, recomenda-se explorar o uso da IA considerando o contexto completo das tarefas, de modo a melhorar a acurácia e a aplicabilidade das estimativas.

Pacote De Replicação

O pacote de replicação encontra-se disponível em https://github.com/ICEI-PUC-Minas-PPLES-TI/plf-es-2025-1-tcci-0393100-pes-barbara-mattioly-e-laura-melo

Referências

(2024). Text generation and prompting. Acessado em: 29 de junho de 2025.

Alhamed, M. and Storer, T. (2021). Playing planning poker in crowds: Human computation of software effort estimates. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 1–12.

Arman, A., Di Reto, E., Mecella, M., and Santucci, G. (2023). An approach for software development effort estimation using chatgpt. In *2023 IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, pages 1–7.

Aziz, R., Afzal, V., Rana, Z. A., and Gilani, S. (2024). Improving software effort estimation by adjusting for over-commitment. In *2024 International Conference on Frontiers of Information Technology (FIT)*, pages 1–6.

Catak, T., Durdu, P. O., and Ilhan Omurca, S. (2024). Enhancing agile effort estimation: An nlp approach for software requirements analysis. In *2024 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pages 1–8.

Choetkiertikul, M., Dam, H. K., Tran, T., Pham, T., Ghose, A., and Menzies, T. (2019). A deep learning model for estimating story points. *IEEE Transactions on Software Engineering*, 45(7):637–656.

Fu, M. and Tantithamthavorn, C. (2023). Gpt2sp: A transformer-based agile story point estimation approach. *IEEE Transactions on Software Engineering*, 49(2):611–625.

Hoda, R., Salleh, N., and Grundy, J. (2018). The rise and evolution of agile software development. *IEEE Software*, 35:58–63.

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., and Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, 55(9).

Manifesto para Desenvolvimento Ágil de Software (2001). Manifesto para desenvolvimento Ágil de software. Acessado em: 18 de maio de 2025.

Sarro, F., Moussa, R., Petrozziello, A., and Harman, M. (2022). Learning from mistakes: Machine learning enhanced human expert effort estimates. *IEEE Transactions on Software Engineering*, 48(6):1868–1879.

Shore, J. and Warden, S. (2007). *The Art of Agile Development*. O'Reilly Media.

Silva Neo, G. d., Silva Neo, A. V. B., Galvao Filho, K. J. A., Moura, J. A. B., and Junior, O. d. G. F. (2024). Neodataset-um conjunto de dados com user stories e story points. *Revista dos Mestrados Profissionais*, 13(2):194–210.

Sommerville, I. (2015). *Software Engineering*. Pearson, 10 edition.

Terlapu, P. V., Raju, K. K., Kiran Kumar, G., Jagadeeswara Rao, G., Kavitha, K., and Samreen, S. (2024). Improved software effort estimation through machine learning: Challenges, applications, and feature importance analysis. *IEEE Access*, 12:138663–138701.

Apêndice

A. Exemplo de Prompt Utilizado

> ***Prompt:*** *Sou um especialista em metodologias ágeis (Scrum, Kanban) e trabalho como agilista em projetos de software. Tenho conhecimento em arquitetura de software, APIs REST, bancos de dados, UI/UX e integrações externas. Além disso, tenho capacidade de comparar tarefas com pontos de referência pré-definidos para manter consistência nas estimativas. Utilizo o modelo Planning Poker para estimar esforço em cada Story Point.*
>
> *Para estimar a complexidade de cards (histórias/tarefas) em story points você deve considerar como referência:*
>
> - *1 ponto (backend): Criação de um endpoint simples sem regras de negócios complexas*
> - *1 ponto (frontend): Criação de um modal simples sem muitas validações*
>
> *Título: {title}*
>
> *Descrição: {description}*
>
> *Tipo: {issue_type}*
>
> *Com base nesses pontos de referência, forneça:*
>
> - *1 Primeira linha: apenas o número do story point*
> - *2 Segunda linha: o tipo Backend — Frontend — Fullstack*
> - *3 Terceira linha: breve justificativa (máx. 3 linhas, 40 palavras e sem tópicos)*

B. Exemplos de Estimativas

B.1. Exemplos de Estimativas com Grande Divergência (Real vs estimada pelo Gemini)

Tabela 2. Exemplo de estimativa divergente - NeoDataset

Real	Gemini	Título	Descrição	Análise Gemini
1	8	Do not show ”This video is still transcoding”if the video plays	This video is still transcoding. You will receive a notification when it is complete. The video should not actually post to subs until the minimum transcodes in done, right? Also, it should not show up after the first version is done.	A complexidade do backend reside na alteração de regras de negócio para publicação de vídeos com base no status de transcoding. No frontend, há lógica condicional para exibir mensagens, interagindo com o player e o status do backend.

Tabela 3. Exemplo de estimativa divergente — Dataset II

Real	Gemini	Título	Descrição	Análise Gemini
13	1	Use pre-configured web servers to launch Mobile Web preview	The Mobile Web browser preview launch configuration uses some separate settings for determining which browser to launch. Instead, we should have a drop down respecting the current browser settings in preferences: * Have a radio button for Global settings or Specific choose:. * If user selects to specifically choose a different option than the global settings, have a drop down with the following options: ** Internal browser ** System browser ** Browser 1 ** Browser 2 * Have a link to open the preference page for users to manage the settings. If any launch configuration fails because a browser setting has been removed, fall back to an external browser using the system default	O card envolve configurar um script de lançamento, utilizando web servers já pre-configurados, para permitir a visualização da aplicação em dispositivos móveis. A complexidade é baixa, focando na orquestração de ferramentas existentes para um propósito específico de preview.

B.2. Exemplos de Estimativas sem Erro (Real vs Estimada pelo *Gemini*)

Tabela 4. Exemplo de estimativa correta — NeoDataset

Real	Gemini	Título	Descrição	Análise Gemini
3	3	Email: Automate welcome post for 1 day after registration	Summary Send users a welcome post the day after they sign up for Minds. Conditions - User has signed up for an account - User has verified their email Subject Welcome to Minds, @username Preheader You've taken the first step. Here's what's next. Copy Thanks for joining Minds! Here are a few things you can do next to establish yourself on Minds, to speak freely, earn crypto, and elevate discourse with us. - Minds is better with more connections. Discover interesting people and subscribe to their posts to keep your newsfeed alive. - Check out trending topics to see what the community is talking about. - Introduce yourself with a post, and let the community know what brought you to Minds. Use the hashtag NewFren to meet other newbies and our community welcome party. - Get notified of replies to your posts, track your crypto rewards, and post on the go with the Minds mobile app, for Android and iOS. Welcome aboard, The Minds Team	A tarefa envolve um processo agendado, complexa filtragem de usuários por tempo e status, montagem de email dinâmico e integração com serviço externo. Isso a torna mais complexa que um item de 1 ponto.

Tabela 5. Exemplo de estimativa correta — Dataset II

Real	Gemini	Título	Descrição	Análise Gemini
1	1	Hide user email address at bottom right hand corner of IDE	Note: discussed with Ingo today by email. Currently, TiStudio shows the user's email address in the bottom right hand corner of the IDE. This may not be good for users wishing to publish public demo or training videos, as it reveals their email address to everyone. Consider either replacing this with their devcenter display name (...), or replacing it with a "logged in"or "logged out"indicator.	Ação frontend para ocultar ou mascarar um dado existente na interface do usuário do IDE. Não implica novas regras complexas ou alterações no backend.