UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Automação de Mensagens de *Commits* Utilizando Modelos de *Large Language Models*

Felipe Moreira Ferreira

Monografia - MBA em Inteligência Artificial e Big Data

ICMC USP SÃO CARLOS

Felipe Moreira Ferreira

Automação de Mensagens de *Commits* Utilizando Modelos de *Large Language Models*

Monografia apresentada ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Área de concentração: Inteligência Artificial

Orientador: Prof. Dr. Diego Raphael Amancio

Versão original

São Carlos

2025

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

F383a

Ferreira, Felipe Moreira
Automação de Mensagens de Commits Utilizando Modelos de Large Language Models / Felipe Moreira Ferreira; orientador Diego Raphael Amancio. -- São Carlos, 2025.
40 p.

Trabalho de conclusão de curso (MBA em Inteligência Artificial e Big Data) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2025.

1. Inteligência Artificial. 2. Processamento de Linguagem Natural. 3. Modelos de Linguagem. 4. Automação de Software. 5. Mensagens de Commit. I. Amancio, Diego Raphael, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Felipe Moreira Ferreira

Commit Message Automation Using *Large Language Models*

Monograph presented to the Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, as part of the requirements for obtaining the title of Specialist in Artificial Intelligence and Big Data.

Concentration area: Artificial Intelligence

Advisor: Prof. Dr. Diego Raphael Amancio

Original version

São Carlos

2025

Felipe Moreira Ferreira

Automação de Mensagens de *Commits* Utilizando Modelos de *Large Language Models*

Monografia apresentada ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Data de defesa: 08 de novembro de 2025

Comissão Julgadora:

Prof. Dr. Diego Raphael Amancio
Orientador

Professora
Dra. Heloisa de Arruda Camargo

São Carlos
2025

AGRADECIMENTOS

Agradeço, primeiramente, à minha família, pelo apoio incondicional, por me ouvir falar sobre temas fora de suas áreas de conhecimento, pelas inúmeras revisões, pela compreensão nos momentos de dedicação intensa e por todo o incentivo ao longo desta jornada acadêmica. Sua presença e paciência foram essenciais para minha jornada até agora.

Expresso também minha gratidão aos meus colegas de trabalho, que colaboraram de forma ativa durante as etapas de testes e validação da ferramenta, contribuindo com tempo, experiência e percepções práticas que enriqueceram significativamente os resultados desta pesquisa.

Por fim, agradeço a todos que, de alguma forma, ofereceram suporte, incentivo e confiança ao longo do desenvolvimento deste estudo. Registro meu sincero reconhecimento, agradecimento e apreço a todos.

"Artificial intelligence is not a substitute for human intelligence;
it is a tool to amplify human creativity and ingenuity."
Fei-Fei Li

RESUMO

FERREIRA, F. M. **Automação de Mensagens de *Commits* Utilizando Modelos de *Large Language Models***. 2025. 40 p. Monografia (MBA em Inteligência Artificial e Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2025.

A qualidade das mensagens de commit é essencial para a manutenção, colaboração e rastreabilidade em projetos de software. No entanto, a escrita manual dessas mensagens costuma ser negligenciada, resultando em descrições incompletas e inconsistentes. Este trabalho apresenta o desenvolvimento de uma ferramenta que automatiza e revisa mensagens de commit utilizando Modelos de Linguagem de Grande Escala (LLMs) integrados a Redes Adversariais Generativas (GANs). A abordagem proposta permite gerar mensagens concisas e informativas com base nas alterações realizadas no código-fonte, além de avaliar a clareza e a completude de mensagens existentes. A metodologia envolveu a implementação de uma arquitetura híbrida com GANs para aprimorar a qualidade textual e a integração da solução a sistemas de controle de versão, possibilitando sugestões automáticas durante o processo de commit. Os resultados quantitativos, avaliados pelas métricas BLEU, ROUGE e METEOR, indicaram alta correspondência lexical e semântica entre as mensagens geradas e as utilizadas por desenvolvedores. A avaliação qualitativa, conduzida com especialistas, apontou percepções positivas quanto à clareza, utilidade e aderência das mensagens, especialmente em commits complexos. Conclui-se que a integração entre LLM e GAN representa uma abordagem viável e inovadora para aprimorar a comunicação técnica e a qualidade documental em repositórios de código, contribuindo para a produtividade e consistência das equipes de desenvolvimento.

Palavras-chave: mensagens de commit. modelos de linguagem. redes adversariais generativas. processamento de linguagem natural. automação de software.

ABSTRACT

FERREIRA, F. M. ***Commit* Message Automation Using *Large Language Models***. 2025. 40 p. Monograph (MBA in Artificial Intelligence and Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2025.

The quality of commit messages plays a key role in software maintenance, collaboration, and project traceability. However, developers often neglect this task, resulting in vague or inconsistent documentation of code changes, which directly impacts maintenance efficiency and team productivity. This study presents the development and evaluation of a tool that automates and reviews commit messages using Large Language Models (LLMs) integrated with Generative Adversarial Networks (GANs), aiming to optimize the time spent on documentation. The proposed system generates concise and informative messages based on source code modifications and assesses the clarity and completeness of existing ones. The methodology involved fine-tuning an LLM with software development data, applying a GAN architecture to refine message generation, and integrating the solution with version control systems for real-time commit assistance. Quantitative evaluations using BLEU, ROUGE, and METEOR demonstrated high lexical and semantic correspondence between generated and real messages. Complementary human evaluations validated the usability, generation accuracy, and the effectiveness of the system's review functionality, highlighting the tool's usefulness and adherence, particularly for complex commits. The results validate the technical feasibility and practical applicability of combining LLMs and GANs to elevate documentation quality and boost developer productivity. Future work includes improving contextual adaptation across programming languages and refining the model's sensitivity to smaller-scale commits.

Keywords: Commit Messages, Generative Adversarial Networks, Large Language Models, Natural Language Processing, Software Engineering Automation.

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

LLM	Large Language Models
RAG	Retrieval-Augmented Generation
GAN	Generative Adversarial Network
API	Application Programming Interface
PLN	Processamento de Linguagem Natural
BLEU	Bilingual Evaluation Understudy
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
METEOR	Metric for Evaluation of Translation with Explicit ORdering

SUMÁRIO

1 INTRODUÇÃO

A qualidade das mensagens de *commit* em projetos de software é crucial para a manutenção, colaboração e entendimento do histórico de mudanças. Mensagens de *commit* claras e explicativas auxiliam os desenvolvedores a entenderem o raciocínio por trás das alterações realizadas, facilitando a depuração de problemas e a continuidade dos projetos ao longo do tempo. Contudo, em muitos casos, as mensagens de *commit* são vagas, incompletas ou mal estruturadas, o que dificulta a comunicação entre os membros da equipe de desenvolvimento. A proposta deste projeto consiste no desenvolvimento de um sistema que utilize *Large Language Models* (LLM) para automatizar a criação de mensagens de *commit* concisas e informativas, com base nas alterações realizadas em um repositório de código. Além disso, a plataforma será capaz de avaliar a qualidade das mensagens de *commit* existentes, fornecendo pontos de melhoria sobre a clareza e completude das informações.

1.1 Objetivo Geral

Desenvolver uma ferramenta capaz de auxiliar os desenvolvedores a manterem um histórico de código mais organizado e compreensível, economizando tempo e esforço na criação de mensagens de *commit*, visando melhorar a qualidade e consistência das mensagens de *commit*, facilitando a compreensão e manutenção do código. A ferramenta irá utilizar uma abordagem baseada em *Generative Adversarial Network* (GAN) para gerar sugestões de mensagens de *commit* personalizadas e de alta qualidade, com base nas alterações feitas no código. Além disso, a ferramenta irá revisar as mensagens escritas pelo desenvolvedor, avaliando a completude e clareza das mesmas com base nas mudanças, utilizando um modelo de LLM refinado para detectar padrões e inconformidades nas mensagens de *commit*. A integração da GAN com o LLM permitirá que a ferramenta aprenda a gerar mensagens de *commit* que sejam não apenas coerentes e claras, mas também consistentes com as convenções e padrões de *commit* existentes no projeto.

1.2 Objetivos Específicos

1. Treinar ou refinar um LLM a fim de torná-lo capaz de processar alterações de código-fonte e ser capaz de gerar e avaliar mensagens de *commit*, utilizando conceitos baseados em GAN, visando melhorar a eficiência e produtividade dos desenvolvedores.

2. Desenvolver um modelo de GAN para gerar sugestões de mensagens de *commit* personalizadas e de alta qualidade, com base nas alterações feitas no código, utilizando

um LLM como gerador de texto, garantindo que as mensagens de *commit* sejam consistentes e padronizadas.

3. Definir um modelo padrão de mensagem de *commit* que sirva de guia para a geração automática, contemplando boas práticas de desenvolvimento (e.g., estilo imperativo, concisão, clareza), e utilizar técnicas de *Retrieval-Augmented Generation* (RAG) para adaptar esse modelo às necessidades específicas do projeto, garantindo a flexibilidade e escalabilidade da plataforma.

4. Integrar a plataforma com sistemas de controle de versão, permitindo que os desenvolvedores recebam sugestões automáticas durante o processo de *commit* e possam revisar suas mensagens de *commit* eficazmente, utilizando o módulo gerador da GAN baseado em LLM para gerar sugestões de mensagens de *commit* personalizadas e relevantes.

5. Desenvolver uma funcionalidade de revisão de mensagens de *commit*, onde o sistema será capaz de analisar uma mensagem já existente, revisando-a e sugerindo pontos de melhoria, considerando sua completude e relevância, baseando-se nas mudanças realizadas no código, utilizando o módulo detector da GAN baseado em LLM para detectar padrões e inconformidades nas mensagens de *commit*, e garantindo que as mensagens de *commit* sejam de alta qualidade e relevantes.

1.3 Justificativa

O controle de versão é uma prática fundamental no desenvolvimento de software, e as mensagens de *commit* desempenham um papel crucial na documentação das alterações realizadas no código. No entanto, a criação de mensagens de *commit* concisas e informativas não é uma tarefa trivial e consome tempo; não obstante, a falta de padronização e de clareza nas mensagens de *commit* pode gerar retrabalho e dificuldade na manutenção dos projetos a longo prazo.

Esta monografia propõe uma solução capaz de auxiliar os desenvolvedores neste problema, utilizando LLM e GAN para automatizar a geração de mensagens de *commit* que seguem padrões de boas práticas, economizando tempo dos desenvolvedores, garantindo maior consistência no repositório de código e, a longo prazo, melhorando a qualidade da documentação e facilitando a colaboração entre desenvolvedores. Além disso, a ferramenta também será capaz de avaliar a qualidade das mensagens de *commit* existentes, fornecendo pontos de melhoria sobre a clareza e completude das informações, o que pode ajudar a melhorar a qualidade do código e reduzir o tempo de depuração.

A integração da GAN com o LLM permite que a ferramenta aprenda a gerar mensagens de *commit* que sejam não apenas coerentes e claras, mas também consistentes com as convenções e padrões de *commit* existentes no projeto, e desta forma tem o

potencial de melhorar significativamente a qualidade e consistência das mensagens de *commit*, facilitando a manutenção e colaboração nos projetos de software.

2 DESENVOLVIMENTO

2.1 Referencial Teórico

2.1.1 Mensagens de *Commit* e Boas Práticas no Controle de Versão

Em projetos de desenvolvimento de software, o uso de sistemas de controle de versão, onde ferramentas *Git* são as mais utilizadas, é uma prática consolidada para o gerenciamento do histórico de alterações no código-fonte. Dentre as diversas entradas registradas nesses sistemas de controle, as mensagens de *commit* desempenham o papel essencial de auxiliar a documentação das modificações realizadas, permitindo os desenvolvedores associarem mensagens explicando não só oque fora alterado mas também podendo incluir as motivações por trás das modificações. Uma mensagem de *commit* bem redigida pode facilitar o rastreamento histórico da introdução de potenciais erros, auxiliar a contextualização e assim acelerar a revisão de código além de simplificar processos de manutenção e evolução de sistemas.

Apesar da importância reconhecida, a escrita de mensagens de *commit* ainda é, muitas vezes, negligenciada por desenvolvedores, resultando em registros vagos, genéricos ou mesmo ausentes. Estudos demonstram que essa prática impacta negativamente a qualidade do repositório de código (Spinellis, 2005), dificultando a compreensão do histórico do projeto e gerando retrabalho durante atividades como depuração, auditoria e refatoração.

Nesse contexto, algumas diretrizes são propostas para orientar a produção de mensagens de *commit* claras, concisas e úteis. Dentre as principais recomendações, destaca-se o uso de linguagem imperativa no título da mensagem de *commit*, a separação entre o título e o corpo da mensagem, a inclusão de justificativas e implicações das mudanças, bem como a consistência de estilo ao longo do repositório (Chacon; Straub, 2014). Além disso, padrões como o *Conventional Commits* (Conventional Commits, 2023) propõem uma estrutura formalizada que facilita a automação de processos como versionamento semântico, geração de *changelogs* e a integração contínua.

A aderência a essas boas práticas contribui significativamente para a legibilidade e manutenção dos projetos. Entretanto, a aplicação consistente dessas diretrizes depende do engajamento individual dos desenvolvedores e do tempo disponível para redigir mensagens informativas, o que nem sempre é viável em ambientes de alta produtividade. Por essa razão, a automação da geração e avaliação de mensagens de *commit* surge como uma alternativa promissora, capaz de reduzir o esforço manual e aumentar a padronização, sem comprometer a qualidade da documentação do código.

2.1.2 LLMs no Domínio de Software

Os Modelos de Linguagem de Grande Escala (LLM) desempenham um papel crescente em diversas tarefas no domínio de software, como geração de código, explicação de trechos de programas, revisão de código e automação de tarefas repetitivas. Esses modelos são treinados sobre vastos conjuntos de dados textuais e possuem capacidade de compreender, gerar e manipular linguagem natural e linguagem de programação com elevado grau de sofisticação (Brown *et al.*, 2020).

A aplicação de LLM em tarefas específicas do ciclo de vida do software tem gerado avanços significativos na produtividade e na qualidade do desenvolvimento. Modelos como o *Codex*, derivado do *GPT-3*, foram projetados para entender instruções em linguagem natural e transformá-las em código funcional em diversas linguagens de programação (Chen *et al.*, 2021). Isso permite, por exemplo, que desenvolvedores solicitem funcionalidades por meio de descrições textuais e recebam trechos de código que atendem a tais demandas, reduzindo o tempo necessário para tarefas rotineiras e facilitando o *onboarding* de novos membros em equipes técnicas.

Além da geração de código, os LLM também são empregados em tarefas como explicação de *commits*, documentação automatizada, identificação de potenciais *bugs* e até mesmo análise de conformidade com padrões de codificação. De acordo com estudos (Ahmad *et al.*, 2021), esses modelos conseguem identificar e prever alterações em código-fonte com base em históricos de *commit*, sugerindo modificações e justificativas plausíveis que se alinham com práticas reais de desenvolvimento.

Entretanto, o uso de LLM no contexto do desenvolvimento de software exige cuidados específicos. O vocabulário técnico, a estrutura sintática dos códigos em diversas linguagens de programação e o contexto de execução são elementos que diferenciam esse domínio do uso geral da linguagem natural. Por essa razão, técnicas como *fine-tuning* em dados específicos de repositórios, ou a utilização de abordagens como RAG, são utilizadas para adaptar os modelos ao vocabulário e padrões característicos do desenvolvimento de software (Lewis *et al.*, 2021).

No contexto desta monografia, os LLM são empregados tanto na geração automática de mensagens de *commit* com base em alterações de código, quanto na avaliação de mensagens já existentes. A capacidade desses modelos de correlacionar linguagem natural com estruturas de código os torna adequados para tarefas que exigem não somente compreensão textual, mas também interpretação semântica do que foi modificado no repositório, promovendo maior padronização e qualidade no histórico de versões dos projetos.

2.1.3 Redes Adversariais Generativas e sua Aplicação no Processamento de Linguagem Natural

As Redes Adversariais Generativas (GAN) (Goodfellow *et al.*, 2014), são uma classe de modelos de aprendizado profundo compostos por dois componentes principais: o gerador e o discriminador. Esses dois modelos são treinados simultaneamente em um processo competitivo, no qual o gerador tenta produzir amostras realistas a partir de ruído, enquanto o discriminador tenta distinguir entre amostras reais e sintéticas. Essa arquitetura inovadora tem sido amplamente explorada em tarefas de geração de imagens, mas também tem demonstrado potencial crescente em aplicações de *Processamento de Linguagem Natural* (PLN).

No domínio de PLN, as GAN são empregadas para melhorar a qualidade e a diversidade dos textos gerados, detectar falsificações, mitigar vieses e, mais recentemente, refinar saídas produzidas por LLM. Sua principal vantagem nesse contexto reside na capacidade de aprender padrões linguísticos complexos e gerar textos mais próximos de exemplos reais, tanto em estilo quanto em coerência contextual (Rosa; Papa, 2021).

A combinação entre LLM e GAN, nesta monografia, cria uma arquitetura híbrida capaz de aliar a fluência linguística com o rigor técnico, elevando o nível de automação e qualidade no controle de versões de software. Essa abordagem se alinha a tendências recentes em sistemas de PLN que buscam equilibrar geração e validação em ambientes supervisionados por aprendizado profundo (Subramanian *et al.*, 2018). Enquanto o LLM é responsável por propor uma primeira versão da mensagem com base nas alterações detectadas no código, o discriminador da GAN avalia essa mensagem à luz de exemplos de boas práticas, aprendendo a rejeitar sugestões inconsistentes ou de baixa qualidade. Esse ciclo iterativo busca não somente automatizar a escrita de mensagens, mas também garantir que elas atendam a padrões de clareza, concisão e relevância técnica esperados em ambientes profissionais de desenvolvimento.

2.1.4 Avaliação de Qualidade de Texto Gerado Automaticamente

A avaliação da qualidade de textos gerados automaticamente é um desafio recorrente em tarefas de PLN, especialmente quando se busca garantir que o conteúdo gerado seja compreensível, relevante e útil para o usuário final. No contexto deste trabalho, em que mensagens de *commit* são geradas e revisadas por meio de modelos baseados em LLM e GAN, torna-se essencial adotar métricas capazes de mensurar aspectos como clareza, completude e aderência às práticas esperadas de documentação de código.

Entre as abordagens mais utilizadas, destacam-se as métricas automáticas baseadas em similaridade lexical. O *Bilingual Evaluation Understudy* (BLEU) (Papineni *et al.*, 2002) avalia a sobreposição de n-gramas entre o texto gerado e um ou mais textos de referência, sendo amplamente utilizado na avaliação de sistemas de tradução automática. Outras

métricas populares incluem o *Recall-Oriented Understudy for Gisting Evaluation* (ROUGE) (Lin, 2004), que mede a sobreposição de unidades como n-gramas e palavras-chave, e o *Metric for Evaluation of Translation with Explicit ORdering* (METEOR) (Banerjee; Lavie, 2005), que incorpora fatores como sinônimos, *stemming* e alinhamento semântico para capturar similaridades mais sutis.

Apesar de sua popularidade, essas métricas apresentam limitações, principalmente por dependerem fortemente de referências pré-definidas e não capturarem nuances semânticas ou a fluidez do texto. Estas limitações tornam-se problemáticas em problemas onde as respostas não são determinísticas, como a geração de mensagens de *commit* onde vários resultados podem ser classificados como corretos e são igualmente válidos dado o contexto inicial. Não obstante, essas métricas tendem a penalizar variações linguísticas criativas, mesmo quando o conteúdo permanece fiel ao objetivo estabelecido.

Por essas razões, avaliações humanas ainda são amplamente utilizadas como complemento às métricas automáticas. Nessas avaliações, critérios como clareza, completude, relevância e coerência contextual são métricas a serem consideradas para assim melhor validar a qualidade das mensagens geradas. A combinação dessas abordagens, automática e humana, permite uma análise mais abrangente do desempenho do sistema proposto e de seu potencial de adoção prática em ambientes de desenvolvimento reais.

2.2 Metodologia

A metodologia deste trabalho foi estruturada visando desenvolver e avaliar uma ferramenta baseada em LLM e GAN, com a finalidade de automatizar e revisar mensagens de *commit*, auxiliando assim as equipes de desenvolvimento em projetos de software. Para alcançar esse propósito, as atividades foram organizadas em cinco etapas principais: coleta e preparação dos dados, treinamento e refinamento dos modelos, desenvolvimento da plataforma, avaliação dos resultados e validação da ferramenta em cenários reais. Essa estrutura metodológica busca garantir, desde as fases iniciais, tanto a robustez técnica quanto a aplicabilidade prática da solução proposta, assegurando que as mensagens de *commit* geradas sejam contextualizadas, claras e aderentes às boas práticas de desenvolvimento. A seguir, detalham-se os processos adotados em cada fase da metodologia.

2.2.1 Coleta e Preparação dos Dados

A primeira etapa consiste na **coleta e preparação dos dados**, com foco em garantir diversidade e representatividade nos exemplos utilizados para treinamento. Foram extraídos dados de repositórios públicos, contendo tanto as alterações realizadas no código quanto as respectivas mensagens de *commit*. Além disso, serão coletados *commits* de projetos específicos, visando contemplar contextos variados e mais próximos da realidade de uso da ferramenta. Antes do uso, os dados serão submetidos a um processo

de pré-processamento, que incluirá a remoção de ruídos, duplicidades e inconsistências, assegurando a qualidade e consistência das informações utilizadas na etapa seguinte.

2.2.2 Treinamento e *Fine-Tuning* dos Modelos

Na sequência, é realizado o **treinamento e *fine-tuning* dos modelos**. Um LLM pré-treinado foi ajustado por meio de *fine-tuning* com dados do domínio de desenvolvimento de software, incorporando técnicas de RAG para melhor contextualização técnica das alterações no código. Simultaneamente, é utilizado um modelo de GAN, onde o LLM atuará como gerador e, em um módulo a parte, como discriminador avaliando a qualidade das mensagens produzidas, promovendo um ciclo de melhoria contínua. Essa abordagem visa produzir sugestões de mensagens de *commit* personalizadas, coerentes e de alta qualidade.

2.2.3 Desenvolvimento da Plataforma

A terceira etapa contempla o **desenvolvimento da plataforma**, que incluiu a implementação de uma *Application Programming Interface* (API) para integração com sistemas de controle de versão, como o *Git*. Também foi desenvolvida uma interface gráfica que permita aos desenvolvedores visualizar as sugestões geradas, além de receber *feedbacks* sobre a clareza e completude das mensagens. Um módulo adicional é responsável por revisar mensagens já formuladas, identificando possíveis melhorias com base nas mudanças realizadas no código e nas boas práticas previamente definidas.

2.2.4 Avaliação dos Resultados

Para garantir a eficácia da solução, a quarta etapa envolverá a **avaliação dos resultados**. Foram aplicadas métricas automáticas de similaridade, como BLEU, ROUGE e *METEOR*, para comparar as mensagens geradas com mensagens reais de *commits*. Além disso, foi realizada uma avaliação qualitativa com desenvolvedores experientes, um grupo composto por Engenheiros e Cientistas da Computação com a média de experiência no ambiente corporativo superior a quinze anos e titulações acadêmicas chegando aos níveis de Mestrado e Doutorado; Este grupo julgou a utilidade, clareza e aderência das sugestões providas pela plataforma. Foi ainda observadas métricas de aceitação das sugestões e o nível de satisfação dos usuários com a ferramenta.

2.2.5 Validação da Ferramenta

Por fim, a etapa de **validação da ferramenta** foi conduzida em contextos reais de desenvolvimento, utilizando dados e fluxos de trabalho reais de projetos ativos. O objetivo era verificar a aplicabilidade e desempenho da solução em situações práticas, bem como identificar possíveis ajustes necessários para sua adoção em ambientes de produção. A partir desses testes, esperou-se comprovar a viabilidade técnica da abordagem proposta e

seu impacto na melhoria da documentação e comunicação entre os membros da equipe de desenvolvimento.

3 AVALIAÇÃO EXPERIMENTAL

3.1 Ambientação e Procedimentos de Avaliação

A avaliação experimental da ferramenta foi conduzida em um ambiente corporativo real, de modo a garantir um contexto prático e representativo das rotinas de desenvolvimento de software. Essa escolha permitiu analisar o comportamento da solução proposta em situações autênticas de versionamento, envolvendo equipes ativas em projetos com prazos e demandas reais. O experimento contou com a participação dos especialistas previamente descritos, que atuaram tanto no desenvolvimento dos seus respectivos projetos quanto na validação qualitativa das mensagens de *commit* geradas.

Marcando o início do período de avaliação da plataforma, foi realizada uma apresentação introdutória sobre as funcionalidades gerais da ferramenta, com o objetivo de familiarizar os participantes com sua interface e modo de uso. Entretanto, para não interferir nos resultados e evitar vieses cognitivos relacionados à forma de avaliação, os métodos de aferição de eficácia e os critérios de desempenho adotados foram apresentados apenas após o término do período experimental. Essa abordagem buscou preservar a espontaneidade do uso e garantir que as percepções dos desenvolvedores refletissem a utilidade prática da ferramenta, e não expectativas prévias sobre o desempenho do modelo.

Os testes foram realizados ao longo de seis semanas consecutivas, correspondendo a três ciclos de desenvolvimento ágil (*sprints*), o que possibilitou observar o uso da ferramenta em diferentes fases e iterações do fluxo de trabalho. Durante esse período, foram analisados projetos que utilizavam predominantemente as linguagens *Java* e a *framework Vue2*, garantindo diversidade entre código *backend* e *frontend*, e, consequentemente, maior robustez na avaliação da capacidade da ferramenta de compreender contextos distintos de modificação de código.

Com o intuito de conservar a confidencialidade corporativa dos dados proprietários envolvidos no processo de desenvolvimento, optou-se por utilizar uma instância local de LLM, especificamente o Phi-3, desenvolvido pela *Microsoft* (Abdin *et al.*, 2024). A integração do modelo com a plataforma desenvolvida seguiu o modelo descrito previamente, que recebia automaticamente as alterações de código realizadas pelos desenvolvedores e gerava, para cada *commit*, uma sugestão de mensagem de *commit*. O desenvolvedor tinha a opção de aceitar integralmente a sugestão, refiná-la manualmente ou apenas descartá-la. Todas as mensagens geradas e as versões finais submetidas pelos desenvolvedores foram armazenadas em um banco de dados para posterior análise quantitativa e qualitativa.

Essa configuração experimental permitiu coletar dados abrangentes sobre o comportamento da ferramenta em um ambiente real, possibilitando a mensuração de métricas

de desempenho e a observação de aspectos subjetivos, como a clareza e a utilidade das mensagens de *commit* geradas. As próximas subseções apresentam as métricas utilizadas e a análise dos resultados obtidos a partir dessa coleta.

3.2 Métricas de Desempenho

A avaliação de desempenho da ferramenta proposta foi estruturada de modo a combinar métricas automáticas de similaridade textual com avaliações qualitativas conduzidas por desenvolvedores especialistas, permitindo uma análise abrangente da eficácia, usabilidade e relevância das mensagens de *commit* geradas. Essa abordagem mista foi escolhida por refletir tanto o aspecto objetivo da correspondência linguística entre as mensagens geradas e as submetidas pelos desenvolvedores, quanto a percepção humana sobre a clareza, completude e aderência das sugestões à realidade do desenvolvimento de software.

Para a avaliação quantitativa, foram utilizadas três métricas amplamente reconhecidas na literatura de PLN: BLEU, ROUGE e METEOR. A métrica BLEU foi empregada para mensurar a similaridade n-gramática entre as mensagens de *commit* geradas automaticamente e aquelas efetivamente submetidas pelos desenvolvedores, conforme metodologia proposta por (Papineni *et al.*, 2002). Já a métrica ROUGE avaliou o nível de sobreposição de palavras e frases relevantes, priorizando aspectos de cobertura lexical e informacional (Lin, 2004). Por fim, a métrica METEOR foi aplicada para capturar relações semânticas mais sutis, considerando sinônimos, radicais e alinhamento semântico entre as mensagens (Banerjee; Lavie, 2005). A união destas três métricas visavam fornecer uma visão equilibrada entre precisão lexical e coerência semântica, permitindo identificar o quanto as mensagens geradas se aproximavam, em conteúdo e estilo, das mensagens reais utilizadas pelos desenvolvedores.

Complementando as análises quantitativas, foi conduzida uma avaliação qualitativa com o grupo de profissionais especialistas, composto por Engenheiros e Cientistas da Computação com média de experiência corporativa superior a quinze anos, e com titulações acadêmicas que chegavam ao nível de Mestrado e Doutorado. Esse grupo realizou uma avaliação subjetiva da ferramenta sob quatro dimensões principais: usabilidade, utilidade, coesão e praticidade. Essas avaliações foram registradas e posteriormente analisadas estatisticamente, permitindo a identificação de padrões e preferências no uso da ferramenta. A partir dessas dimensões, buscou-se compreender não apenas o desempenho técnico da ferramenta, mas também visando medir o nível de contentamento geral dos usuários quanto à integração da ferramenta ao fluxo de *commits*, a redução do tempo gasto na redação de mensagens e sua aplicabilidade real no ciclo de vida de desenvolvimento de software

Por fim, foram observadas métricas de aceitação das sugestões, mensurando o percentual de mensagens sugeridas que foram aceitas integralmente, editadas ou descartadas

pelos desenvolvedores. Essa métrica permitiu avaliar o grau de confiança e utilidade percebida das sugestões providas pela plataforma.

A combinação de métricas quantitativas e qualitativas forneceu uma visão abrangente do desempenho da solução, possibilitando analisar tanto a qualidade linguística das mensagens geradas quanto a percepção prática dos usuários sobre sua utilidade em contextos reais de desenvolvimento.

3.3 Resultados Coletados

Os resultados obtidos com a fase de avaliação experimental permitiram analisar, de forma quantitativa e qualitativa, o desempenho da ferramenta de automação e revisão de mensagens de *commit* em um cenário corporativo real. Durante o período de avaliação 184 modificações de código foram registradas pela ferramenta, permeando seis projetos de componentes distintos porem interligados no ecossistema da organização. Assim como descrito anteriormente, durante esta fase, foram coletados informações relacionadas as sugestões de mensagens e as mensagens reais submetidas, a fim de serem posteriormente analisadas através das métricas propostas anteriormente.

Tabela 1 – Correlação entre métricas de avaliação automática de texto

Métrica	**Valor obtido**	**Faixa de referência**	**Interpretação geral**
BLEU	0.72	0.6 – 0.8	Muito boa correspondência lexical e estrutural
ROUGE	0.68	0.5 – 0.7	Boa cobertura do conteúdo de referência
METEOR	0.65	0.5 – 0.7	Boa correspondência linguística e semântica

Os resultados quantitativos obtidos, indicados pela Tabela 1, demonstram um nível de correlação alto entre as mensagens geradas automaticamente e aquelas efetivamente utilizadas pelos desenvolvedores. Os valores de BLEU (0,72), ROUGE (0,68) e METEOR (0,65) evidenciam que as mensagens produzidas pelo modelo apresentam elevada similaridade lexical, estrutural e semântica em relação às mensagens reais. Em termos práticos, isso sugere que a plataforma proposta foi capaz de gerar mensagens semanticamente próximas às aceitas pelos desenvolvedores, mantendo consistência terminológica e coerência contextual em relação às modificações no código.

A métrica de aceitação das sugestões apresentou resultados relevantes, com 27% das mensagens aceitas integralmente, 62% parcialmente aceitas e 11% descartadas, indicando um bom nível de confiança e utilidade percebida pelos usuários. Observou-se ainda que a taxa de edição parcial foi significativamente superior à de rejeição, o que reforça a validade estrutural e semântica das sugestões geradas, demonstrando que, mesmo

quando ajustes manuais foram necessários, o modelo forneceu bases consistentes, contextualmente adequadas e alinhadas ao estilo real das mensagens de *commit* produzidas pelos desenvolvedores.

Além dos resultados obtidos através das métricas quantitativas, a avaliação humana revelou percepções positivas quanto à utilidade e clareza das mensagens geradas. Os avaliadores atribuíram notas médias de 7,8 para utilidade, 8,4 para clareza e 7,4 para aderência (considerando uma escala de 1 a 10), além da percepção geral onde ela foi especialmente eficiente para *commits* de maior complexidade, que tendem a envolver a edição de múltiplos arquivos e demandam maior esforço cognitivo para a elaboração de descrições textuais adequadas. As métricas qualitativas de usabilidade e praticidade também receberam avaliações favoráveis, refletindo o impacto positivo da integração fluida da ferramenta ao fluxo usual de trabalho dos desenvolvedores e sua contribuição para a redução do tempo e da carga manual associada à escrita de mensagens de *commit*.

3.4 Discussão dos Resultados

Ao realizar uma análise dos resultados obtidos nesta etapa de avaliação experimental, estes demonstraram que a plataforma proposta atingiu um nível satisfatório de desempenho, agregando qualidade linguística, relevância semântica e usabilidade prática na geração e revisão de mensagens de *commit*.

A combinação de métricas quantitativas (BLEU, ROUGE e METEOR) com avaliações qualitativas conduzidas por especialistas demonstrou que a integração entre LLM e GAN foi eficaz para compreender o contexto das modificações de código e produzir descrições textuais com estrutura e vocabulário próximos aos produzidos pelos desenvolvedores. A avaliação humana reforçou esses resultados ao apontar que as mensagens geradas apresentaram clareza e coerência adequadas à complexidade das modificações analisadas, além de reduzirem o tempo gasto na redação das mensagens finais.

Além da qualidade textual, a taxa de aceitação completas ou parciais das sugestões indicou que a ferramenta possui alta utilidade prática, sendo incorporada de maneira natural ao fluxo de *commits* durante os ciclos de desenvolvimento. Os participantes relataram que a solução foi especialmente benéfica para *commits* de maior escopo, em que o resumo das mudanças exige maior esforço cognitivo. Em contrapartida, observou-se que em alterações menores a ferramenta, por vezes, gerou mensagens genéricas ou redundantes, o que sugere a necessidade de aprimoramento do modelo no tratamento de contextos com baixo volume de alterações.

4 CONCLUSÃO

O presente trabalho apresentou o desenvolvimento e a avaliação de uma ferramenta voltada à automação e revisão de mensagens de *commit* em projetos de software, fundamentada na integração entre LLM e GAN. A proposta teve como objetivo principal melhorar a clareza, consistência e padronização das mensagens de commit, ao mesmo tempo em que buscou reduzir o esforço cognitivo e o tempo despendido pelos desenvolvedores na documentação de alterações de código.

Ao longo da pesquisa, foi possível identificar que o uso combinado de LLM e GAN oferece um potencial significativo para geração textual contextualizada, especialmente quando aplicada ao domínio do desenvolvimento de software. A ferramenta implementada demonstrou capacidade de compreender as modificações no código-fonte e produzir descrições coerentes e relevantes, atendendo aos princípios de boas práticas de versionamento. Além disso, o uso de métricas quantitativas (BLEU, ROUGE e METEOR) em conjunto com avaliações humanas evidenciou alto grau de similaridade entre as mensagens geradas e as efetivamente utilizadas, validando a eficiência da abordagem proposta.

A aplicação da solução em ambiente corporativo real, envolvendo desenvolvedores experientes, confirmou sua viabilidade prática e impacto positivo sobre a produtividade e a qualidade da documentação de código. Observou-se que a ferramenta se integrou de forma fluida ao ciclo de desenvolvimento, sendo percebida como útil e intuitiva pelos profissionais participantes. Ainda assim, verificou-se que em contextos de *commits* mais simples ou com menor densidade de alteração, o modelo tende a gerar mensagens genéricas ou redundantes, sinalizando a necessidade de aprimoramento da sensibilidade contextual do sistema.

Como perspectivas de continuidade, recomenda-se o aperfeiçoamento dos mecanismos de contextualização e adaptação do modelo, permitindo que a ferramenta reconheça padrões específicos de diferentes linguagens de programação, *frameworks* e convenções de projeto.

Em síntese, os resultados obtidos demonstram que a ferramenta desenvolvida atinge plenamente os inicialmente objetivos propostos, apresentando solidez técnica, aplicabilidade prática e relevância científica. A integração entre LLM e GAN mostrou-se uma abordagem viável e inovadora para aprimorar a comunicação técnica em ambientes de desenvolvimento colaborativo, reduzindo o esforço inicial e aumentando a consistência documental dos repositórios de código. Ao unir fundamentos de processamento de linguagem natural e aprendizado profundo à prática cotidiana de versionamento, este trabalho abre caminho para novas soluções baseadas em intenligência artificial que ampliem a produtividade, a rastreabilidade e a qualidade da engenharia de código, promovendo um ecossistema mais

eficiente e colaborativo.

REFERÊNCIAS

ABDIN, M. I. *et al.* **Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone**. [*S.l.*], 2024. Disponível em: https://www.microsoft.com/en-us/research/publication/phi-3-technical-report-a-highly-capable-language-model-locally-on-your-phone/.

AHMAD, W. U. *et al.* **Unified Pre-training for Program Understanding and Generation**. 2021. Disponível em: https://arxiv.org/abs/2103.06333.

BANERJEE, S.; LAVIE, A. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. *In*: GOLDSTEIN, J. *et al.* (ed.). **Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization**. Ann Arbor, Michigan: Association for Computational Linguistics, 2005. p. 65–72. Disponível em: https://aclanthology.org/W05-0909/.

BROWN, T. B. *et al.* **Language Models are Few-Shot Learners**. 2020. Disponível em: https://arxiv.org/abs/2005.14165.

CHACON, S.; STRAUB, B. **Pro Git**. 2. ed. Apress, 2014. Disponível em: https://github.com/progit/progit2/releases/download/2.1.447/progit.pdf. Acesso em: 21 Abr. 2025.

CHEN, M. *et al.* **Evaluating Large Language Models Trained on Code**. 2021. Disponível em: https://arxiv.org/abs/2107.03374.

Conventional Commits. **Conventional Commits Specification**. 2023. Disponível em: https://www.conventionalcommits.org/en/v1.0.0/. Acesso em: 21 Abr. 2025.

GOODFELLOW, I. J. *et al.* Generative adversarial nets. *In*: GHAHRAMANI, Z. *et al.* (ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2014. v. 27. Disponível em: https://proceedings.neurips.cc/paper_files/paper/2014/file/f033ed80deb0234979a61f95710dbe25-Paper.pdf.

LEWIS, P. *et al.* **Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks**. 2021. Disponível em: https://arxiv.org/abs/2005.11401.

LIN, C.-Y. ROUGE: A package for automatic evaluation of summaries. *In*: **Text Summarization Branches Out**. Barcelona, Spain: Association for Computational Linguistics, 2004. p. 74–81. Disponível em: https://aclanthology.org/W04-1013/.

PAPINENI, K. *et al.* Bleu: a method for automatic evaluation of machine translation. *In*: ISABELLE, P.; CHARNIAK, E.; LIN, D. (ed.). **Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics**. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, 2002. p. 311–318. Disponível em: https://aclanthology.org/P02-1040/.

ROSA, G. H. de; PAPA, J. P. A survey on text generation using generative adversarial networks. **Pattern Recognition**, Elsevier BV, v. 119, p. 108098, nov. 2021. ISSN 0031-3203. Disponível em: http://dx.doi.org/10.1016/j.patcog.2021.108098.

SPINELLIS, D. Version control systems. **IEEE Software**, v. 22, n. 5, p. 108–109, 2005.

SUBRAMANIAN, S. *et al.* Towards text generation with adversarially learned neural outlines. *In*: BENGIO, S. *et al.* (ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2018. v. 31. Disponível em: https://proceedings.neurips.cc/paper_files/paper/2018/file/aaaccd2766ec67aecbe26459bb828d81-Paper.pdf.