MINISTÉRIO DA EDUCAÇÃO

SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA BAIANO - CAMPUS CATU

CURSO TECNÓLOGO EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

TRABALHO DE CONCLUSÃO DE CURSO

Gustavo Jesus da Silva Costa

Orientador: André Luiz Andrade Rezende; Coorientador: Tarsio Ribeiro Cavalcante

EVOLUÇÃO DO M-READER PARA ARQUITETURA CLIENTE-SERVIDOR COM OCR HÍBRIDO: ANÁLISE DE ACURÁCIA E DESEMPENHO EM TECNOLOGIA ASSISTIVA PARA PESSOAS COM DEFICIÊNCIA VISUAL

CATU

2026

Instituto Federal de Educação, Ciência e Tecnologia Baiano – Campus Catu
Setor de Biblioteca

C838 Costa, Gustavo Jesus da Silva
Evolução do M-Reader para arquitetura cliente-servidor com OCR Híbrido: análise de acurácia e desempenho em tecnologia assistiva para pessoas com deficiência visual / Gustavo Jesus da Silva Costa. – 2026.

65 f.:

Orientador(a): Prof. André Luiz Andrade Rezende.
Co-orientador(a): Prof. Tarsio Ribeiro Cavalcante.

TCC (monografia), Instituto Federal de Educação, Ciência e Tecnologia Baiano, Tecnólogo em Análise e Desenvolvimento de Sistemas, Catu, 2026.

1. Tecnologia assistiva. 2. Acessibilidade digital. 3. Transcrição multimodal. I. Rezende, André Luiz Andrade. II. Cavalcante, Tarsio Ribeiro. III. Título.

CDU: 376

RESUMO

Este trabalho avalia a evolução técnica do M-Reader, sistema assistivo de leitura voltado a pessoas com deficiência visual, a partir da migração de um processamento majoritariamente local para uma arquitetura cliente-servidor com OCR híbrido. O problema central é a perda de qualidade na transcrição de documentos fotografados em condições adversas, situação comum para quem depende dessas ferramentas para ler livros e documentos do dia a dia. A investigação é de natureza aplicada, com abordagem quantitativa e delineamento comparativo. Cinco estágios do fluxo de processamento foram confrontados (old, simple, advanced, enhanced e openai) sobre um conjunto de 222 imagens reais capturadas por câmera de Raspberry Pi. O provedor multimodal (GPT-4.1-mini da OpenAI) operou como retorno alternativo orientado por qualidade, sendo acionado apenas quando o motor local indicava baixa confiabilidade. Na comparação pareada entre openai e old (206 imagens com resultado em ambos os métodos), o ganho absoluto de acurácia foi de 5,53 pontos percentuais, com efeito grande confirmado pelo teste de Wilcoxon ($p < 0,001$; $r = 0,62$). A acurácia por palavra subiu de 74,82% para 90,92%, e o ganho concentrou-se nas imagens mais difíceis (+14,06 p.p.), exatamente onde o OCR tradicional mais falha. O custo adicional médio foi de 3,996 segundos por imagem nas condições do benchmark. Ao desacoplar captura e processamento, o sistema ganhou flexibilidade técnica sem abrir mão do hardware de baixo custo, uma base concreta para que mais pessoas com deficiência visual tenham acesso confiável ao texto impresso.

Palavras-chave: Tecnologia Assistiva; OCR Híbrido; Transcrição Multimodal; M-Reader; Acessibilidade Digital.

Abstract

This study presents the technical evolution of the M-Reader assistive reading system for visually impaired users, transitioning from mostly local processing to a client-server architecture with hybrid OCR. The core problem is transcription quality loss when documents are photographed under adverse conditions, a daily obstacle for those who depend on such tools to access printed books, educational materials, and everyday texts. The method is applied, quantitative, and comparatively designed. Five pipeline stages were evaluated (old, simple, advanced, enhanced, and openai) across 222 real-world images captured by a Raspberry Pi camera. The multimodal provider (GPT-4.1-mini) served as a quality-driven fallback, activated only when the local engine signaled low reliability. In paired comparison (206 images with valid results for both openai and old), the absolute accuracy gain was 5.53 percentage points, with large effect confirmed by the Wilcoxon signed-rank test ($p < 0.001$; $r = 0.62$). Word accuracy rose from 74.82% to 90.92%, with gains concentrated in the most difficult images (+14.06 p.p.), precisely where traditional OCR degrades most. Average additional latency was 3.996 seconds per image under benchmark conditions. By decoupling capture from heavy processing, the resulting architecture enables continuous technical improvement without sacrificing viability on low-cost hardware, a concrete foundation for expanding reliable text access to more visually impaired users.

Keywords: Assistive Technology; Hybrid OCR; Multimodal Transcription; M-Reader; Digital Accessibility.

LISTA DE SIGLAS

ABNT - Associação Brasileira de Normas Técnicas

API - Application Programming Interface

CA - Character Accuracy

CER - Character Error Rate

CSV - Comma-Separated Values

HTTP - HyperText Transfer Protocol

IA - Inteligência Artificial

IBGE - Instituto Brasileiro de Geografia e Estatística

IF Baiano - Instituto Federal de Educação, Ciência e Tecnologia Baiano

ISO - International Organization for Standardization

JSON - JavaScript Object Notation

LSTM - Long Short-Term Memory

OCR - Optical Character Recognition (Reconhecimento Óptico de Caracteres)

OEM - OCR Engine Mode

PDCA - Plan-Do-Check-Act

PDF - Portable Document Format

PSM - Page Segmentation Mode

REST - Representational State Transfer

SQLite - Structured Query Language Lite

TA - Tecnologia Assistiva

TCC - Trabalho de Conclusão de Curso

UI - User Interface (Interface de Usuário)

VLM - Vision-Language Model (Modelo Visão-Linguagem)

WA - Word Accuracy

WCAG - Web Content Accessibility Guidelines

WER - Word Error Rate

WSGI - Web Server Gateway Interface

LISTA DE TABELAS

SUMÁRIO

1. Introdução

Para a maioria das pessoas que enxergam, o acesso ao conteúdo de um livro físico é imediato. Entre as pessoas com deficiência visual, a situação varia conforme o grau da perda: quem tem baixa visão muitas vezes consegue ler com o apoio de lupas e outros recursos de ampliação; já para quem é cego ou tem perda visual mais acentuada, a mesma página impressa pode ser uma barreira intransponível, a menos que exista uma tecnologia capaz de fazer a mediação entre o texto impresso e a palavra falada. Em contextos educacionais, no trabalho ou no simples ato de ler uma bula de remédio, essa dependência de ferramentas assistivas é diária, concreta e urgente.

O M-Reader surge nesse contexto: uma solução de baixo custo orientada à acessibilidade, desenvolvida no âmbito do curso de Análise e Desenvolvimento de Sistemas, combinando captura de imagem via Raspberry Pi, processamento textual por OCR (Reconhecimento Óptico de Caracteres) e interface voltada ao uso assistivo. Iniciado como projeto acadêmico (REZENDE _et_ al., 2021a)., o sistema evoluiu ao longo de ciclos incrementais de melhoria, acumulando funcionalidades de digitalização, reconhecimento de texto e leitura acessível por síntese de voz.

A qualidade dessa leitura, porém, depende diretamente das condições da imagem capturada. Iluminação irregular, páginas curvadas, reflexo, foco imperfeito, problemas comuns quando alguém fotografa um livro com uma câmera embarcada em condições reais de uso. Nesses cenários adversos, o OCR local produz transcrições fragmentadas ou ilegíveis, comprometendo a utilidade do sistema precisamente para quem mais precisa dele. A evolução do M-Reader para uma arquitetura cliente-servidor com OCR híbrido representa a resposta técnica a esse problema.

Para testar se essa evolução realmente traz ganhos, o trabalho seguiu uma abordagem experimental comparativa. Foi montado um benchmark usando um conjunto de 222 imagens reais, capturadas pela câmera do Raspberry Pi em condições parecidas com as de uso do dia a dia, muitas delas com problemas de iluminação, foco e curvatura da página. Cada uma dessas imagens passou pelos cinco estágios de processamento do sistema (old, simple, advanced, enhanced e openai), o que permitiu comparar o OCR local com a estratégia híbrida que usa o retorno alternativo multimodal.

Sobre as imagens que tiveram resultado válido nos dois métodos, foram aplicados testes estatísticos, principalmente o teste de Wilcoxon, junto com o cálculo do tamanho de efeito, para verificar se a diferença de acurácia é de fato significativa e não apenas fruto do acaso. O detalhamento da metodologia, do ambiente de teste e da estratégia estatística está descrito no Capítulo 3.

1.1. Problema

Este trabalho investiga se é possível aprimorar o desempenho do M-Reader por meio de uma arquitetura distribuída com seleção de provedores OCR e retorno alternativo (fallback) orientado por qualidade. Partindo das imagens capturadas pelo dispositivo cliente,

analisa-se o efeito dessa estratégia sobre duas variáveis centrais: acurácia da transcrição e tempo de processamento.

A questão de pesquisa que orienta o trabalho é: a adoção de uma arquitetura cliente-servidor com OCR híbrido e retorno alternativo multimodal melhora a acurácia de transcrição do M-Reader, em cenário real de uso assistivo, sem comprometer o tempo de processamento?

1.2. Hipótese

A hipótese levantada é que a arquitetura cliente-servidor, associada à estratégia híbrida de OCR com retorno alternativo para transcrição multimodal, produz ganho de acurácia superior a 3 pontos percentuais em relação ao método Tesseract tradicional, sobretudo em imagens de maior complexidade visual, mantendo latência adicional inferior a 15 segundos por imagem. Assume-se também que o provedor multimodal apresentará menor dispersão de desempenho entre imagens, indicando maior estabilidade em condições adversas de captura.

Os dois limiares não foram definidos de forma arbitrária. O valor de 3 pontos percentuais representa o ganho mínimo para que a mudança compense na prática: acionar um provedor remoto tem custo financeiro por requisição e cria dependência de conectividade, e um ganho abaixo desse patamar dificilmente justificaria essa complexidade extra, além de poder se confundir com flutuações normais de medição. Já o limite de 15 segundos é uma estimativa do tempo máximo de espera que se considera aceitável na leitura assistiva página a página: esperas muito acima disso tornam a leitura de um documento com várias páginas cansativa e desestimulam o uso contínuo do sistema. Vale registrar que esse critério de latência se refere ao ambiente do protocolo de avaliação descrito no Capítulo 3 (servidor em rede local, acesso à API OpenAI via internet) e não se estende automaticamente a cenários de campo com conectividade instável ou servidores geograficamente distantes.

1.3. Justificativa

Segundo o Censo Demográfico 2022, cerca de 7,9 milhões de brasileiros com dois anos ou mais relataram dificuldade funcional para enxergar, a deficiência mais frequente entre todas as categorias levantadas (IBGE, 2025). Por trás desse número há pessoas que encontram barreiras diárias no acesso a livros, documentos e informações impressas. A Lei Brasileira de Inclusão (BRASIL, 2015) reforça que acessibilidade não é benefício opcional, mas direito fundamental. Em escala internacional, a OMS reitera que soluções assistivas precisam combinar efetividade, alcance e viabilidade prática (WHO, 2019). Melhorar o desempenho de uma tecnologia assistiva já existente, portanto, é uma contribuição direta para reduzir barreiras informacionais reais.

Do ponto de vista tecnológico, o Tesseract continua relevante por sua maturidade e baixo custo (SMITH, 2007; TESSERACT OCR, 2024), mas apresenta degradação perceptível em entradas visualmente adversas. A literatura recente mostra avanço consistente de abordagens neurais e multimodais para esse tipo de cenário (LI et al., 2021; BLECHER et al., 2023; POZNANSKI et al., 2025). A adoção dessas abordagens em um

sistema assistivo de baixo custo, porém, não é uma troca simples. Cada requisição a um provedor multimodal remoto tem custo financeiro, o que pesa em um projeto que se propõe a ser acessível. Além disso, o processamento remoto adiciona segundos de espera a cada página, entre o envio da imagem, a inferência e o retorno do resultado, e o sistema passa a depender de conexão com a internet, que nem sempre está disponível ou estável no contexto de uso. O Tesseract, por outro lado, roda localmente, é gratuito e resolve bem os casos simples. Não faria sentido abandoná-lo e pagar pelo provedor remoto em todas as páginas, sendo que só uma parte delas realmente precisa desse reforço.

É desse equilíbrio que nasce tanto a estratégia avaliada quanto a hipótese deste trabalho. Em vez de substituir o OCR tradicional, a pesquisa avalia uma estratégia híbrida: o método convencional atende os casos simples com rapidez e sem custo adicional, e o retorno alternativo multimodal entra em ação apenas quando indicadores de qualidade apontam baixa confiabilidade. Os limiares definidos na hipótese refletem as duas pontas dessa balança: o ganho de acurácia precisa ser grande o suficiente para compensar o custo e a dependência do provedor remoto, daí o mínimo de 3 pontos percentuais, e a espera adicional precisa se manter dentro do tolerável para a leitura página a página, daí o teto de 15 segundos. Essa decisão se alinha à literatura de avaliação em OCR (NEUDECKER et al., 2021) e responde a uma lacuna identificada em revisões sobre IA em tecnologias assistivas: a escassez de estudos que articulem desempenho técnico e impacto operacional em soluções de uso contínuo (TRIYONO et al., 2023).

As contribuições da pesquisa se distribuem em mais de uma frente. No plano social, o trabalho fortalece uma tecnologia assistiva orientada à autonomia de pessoas com deficiência visual. No plano técnico, avalia uma arquitetura escalável com política de acionamento seletivo. Há ainda uma contribuição científica: a análise comparativa é feita sobre artefatos reais do projeto (as imagens capturadas pelo dispositivo, os textos de referência e os registros gerados pelo benchmark), com potencial de orientar ciclos futuros de melhoria. Apesar dos avanços recentes em OCR multimodal, ainda são poucos os trabalhos que avaliam ao mesmo tempo acurácia e tempo de processamento em sistemas assistivos reais e de baixo custo e é justamente essa lacuna que este trabalho busca preencher.

1.4. Objetivo Geral

Avaliar a evolução do M-Reader para uma arquitetura cliente-servidor com OCR híbrido e retorno alternativo orientado por qualidade, mensurando seus impactos sobre a acurácia da transcrição e o tempo de processamento em contexto de tecnologia assistiva.

1.5. Objetivos Específicos

1. Descrever a transição arquitetural do M-Reader de processamento local para arranjo cliente-servidor;

2. Analisar o funcionamento do fluxo operacional atual, com seleção de provedores OCR e acionamento de retorno alternativo por qualidade;
3. Comparar o desempenho dos diferentes estágios de processamento (old, simple, advanced, enhanced e openai) por meio dos resultados do benchmark;
4. Quantificar a diferença absoluta e relativa de acurácia entre o estágio multimodal e o baseline equivalente;

A análise do custo temporal do acionamento remoto e a discussão das limitações da avaliação não entram como objetivos específicos, já que a primeira faz parte do próprio objetivo geral e a segunda é etapa esperada de qualquer trabalho desta natureza. Ambas são desenvolvidas ao longo dos Capítulos 5 e 6.

1.6. Estrutura do trabalho

O Capítulo 2 apresenta a fundamentação teórica, cobrindo tecnologias assistivas, OCR, processamento de imagens, inteligência artificial aplicada a documentos, arquitetura de software e trabalhos correlatos. O Capítulo 3 descreve a metodologia, incluindo o delineamento comparativo, o ambiente de teste e a estratégia estatística. O Capítulo 4 detalha o desenvolvimento e a implementação da arquitetura atual. O Capítulo 5 apresenta e discute os resultados. O Capítulo 6 traz as considerações finais, com síntese dos achados, contribuições, limitações e recomendações para continuidade.

2 - FUNDAMENTAÇÃO TEÓRICA

Este capítulo reúne os fundamentos que sustentam a proposta do trabalho, do contexto humano e social das tecnologias assistivas aos conceitos técnicos de OCR, processamento de imagens, inteligência artificial, arquitetura de software, encerrando com os trabalhos correlatos e as lacunas que este trabalho busca preencher.

2.1 Tecnologias assistivas e inclusão digital

Nos termos da Lei Brasileira de Inclusão, tecnologia assistiva compreende produtos, equipamentos, recursos, metodologias, estratégias, práticas e serviços que visam promover a funcionalidade, a autonomia e a participação social de pessoas com deficiência ou com mobilidade reduzida (BRASIL, 2015). No caso da deficiência visual, esses recursos atuam como ponte entre o conteúdo originalmente visual e formas acessíveis de percepção, áudio e texto estruturado, principalmente. As barreiras enfrentadas por pessoas com deficiência não se resumem a um único tipo: a própria lei cita vários, como as barreiras urbanísticas, as arquitetônicas, as de transporte, as tecnológicas e as barreiras nas comunicações e na informação (BRASIL, 2015). São estas últimas que este trabalho aborda: as barreiras que o M-Reader busca converter em oportunidades concretas de estudo, trabalho e cidadania.

Os dados do Censo Demográfico 2022 evidenciam a dimensão desse desafio no Brasil, reforçando a necessidade de soluções de acessibilidade em escala (IBGE, 2025). O marco legal brasileiro estabelece que acessibilidade não é benefício opcional, mas direito fundamental relacionado à igualdade de oportunidades (BRASIL, 2015). Em perspectiva

internacional, a OMS aponta que barreiras informacionais ainda impactam diretamente a qualidade de vida de milhões de pessoas com perda visual (WHO, 2019).

A primeira condição da inclusão digital, no entanto, ainda é o acesso. Boa parte das soluções assistivas comerciais para leitura, como dispositivos dedicados e softwares proprietários, tem preço elevado para a realidade brasileira, o que exclui justamente a parcela da população que mais precisa delas. É aqui que a ideia de tecnologia social se conecta a este trabalho: soluções de baixo custo, feitas com componentes baratos e fáceis de encontrar, que possam ser reaplicadas para resolver problemas concretos de inclusão. Essa preocupação aparece na própria OMS, que cobra das soluções assistivas tanto o bom funcionamento quanto um custo que permita chegar a quem precisa (WHO, 2019). O M-Reader segue essa lógica ao adotar o Raspberry Pi como plataforma: é o barateamento do hardware que torna o acesso viável, e esse é um diferencial da solução em relação às alternativas comerciais. Não se trata de uma iniciativa isolada: o mesmo grupo de pesquisa já havia explorado essa abordagem no MINDER, um computador pessoal de baixo custo voltado à inclusão sociodigital de pessoas cegas (REZENDE et al., 2021b).

Garantir acesso ao equipamento, porém, é só o primeiro passo da inclusão digital. É preciso assegurar também usabilidade, continuidade de uso e confiabilidade dos sistemas. Revisão abrangente de Yao, Zhou e Hu (2025) sobre sistemas visuais assistivos evidencia que abordagens combinadas, que integram múltiplas técnicas em vez de depender de um único método, tendem a apresentar maior robustez em cenários reais. Uma solução assistiva precisa funcionar de forma consistente inclusive sob variações de iluminação, qualidade de imagem e limitações de conectividade. Essa premissa é central para compreender por que o desempenho técnico do OCR, isoladamente, não basta: é preciso considerar toda a experiência de uso assistivo.

2.1.1 Usabilidade e interação no contexto assistivo

A norma ISO 9241-11 define usabilidade como a medida em que um sistema pode ser usado por usuários específicos para alcançar objetivos com eficácia, eficiência e satisfação em um contexto de uso determinado (ISO, 2018). Quando se trata de tecnologia assistiva, essa definição ganha um peso adicional, porque o contexto de uso envolve usuários que, em geral, não têm uma via alternativa quando o sistema falha. Se uma transcrição sai errada ou o retorno sonoro não acontece, a pessoa com deficiência visual não consegue simplesmente conferir o resultado na tela: a falha interrompe a tarefa por completo. Por isso, requisitos como confiabilidade e previsibilidade de resposta, que em outros sistemas seriam desejáveis, aqui se tornam condição básica de uso.

No campo da interação humano-computador, as recomendações voltadas a pessoas com deficiência visual seguem uma direção parecida. A ISO 9241-171, que trata de acessibilidade de software, orienta que a informação seja oferecida em mais de uma modalidade sensorial, que as ações do usuário recebam retorno imediato e que a operação não dependa exclusivamente da visão (ISO, 2008). Na prática, isso se traduz em interfaces com poucos controles, feedback sonoro consistente para cada ação e tolerância a erros de operação. Essas recomendações se refletem nas decisões de interface do M-Reader, descritas no Capítulo 4, como a operação por comandos simples e o uso da síntese de voz como canal principal de retorno ao usuário.

2.1.2 Normas e diretrizes de acessibilidade

No plano internacional, o conjunto de diretrizes mais difundido é o WCAG 2.1, publicado pelo W3C. Embora tenha sido criado para conteúdo web, seus quatro princípios, perceptível, operável, compreensível e robusto, acabaram virando referência geral para o projeto de sistemas acessíveis (W3C, 2018). O primeiro princípio é o que mais dialoga com este trabalho: a informação precisa estar disponível de forma que o usuário consiga percebê-la, o que, para uma pessoa com deficiência visual diante de um texto impresso, significa convertê-la em áudio ou em outro formato acessível.

No contexto brasileiro, além da Lei Brasileira de Inclusão (BRASIL, 2015), já mencionada, a ABNT publicou a NBR 17060, com diretrizes de acessibilidade para produtos e serviços de tecnologia da informação (ABNT, 2022). Para este trabalho, essas normas importam menos como uma lista de conformidade a ser verificada e mais como fundamento conceitual: elas deixam claro que acessibilidade é um requisito de projeto, e não um ajuste posterior. É essa a lógica que orienta o M-Reader desde sua concepção, uma vez que o sistema existe justamente para tornar perceptível um conteúdo que, na forma impressa, não é.

2.1.3 Critérios de avaliação de tecnologias assistivas

Avaliar uma tecnologia assistiva não se resume a medir seu desempenho técnico. A OMS aponta que soluções assistivas precisam combinar efetividade, alcance e viabilidade prática, ou seja, precisam funcionar bem, chegar a quem precisa e ser sustentáveis em termos de custo (WHO, 2019). Revisões recentes sobre inteligência artificial aplicada à deficiência visual reforçam esse olhar e apontam acurácia, tempo de resposta e robustez em condições reais como os critérios técnicos mais recorrentes de avaliação (TRIYONO et al., 2023; YAO; ZHOU; HU, 2025). Budrionis et al. (2022) observam ainda que boa parte dos estudos da área avalia protótipos apenas em condições controladas, distantes das situações enfrentadas pelo usuário no dia a dia.

É desse quadro que derivam as duas variáveis centrais avaliadas neste trabalho: acurácia da transcrição e tempo de processamento, medidas sobre imagens capturadas em condições reais de uso. A primeira se liga diretamente à efetividade da leitura assistiva; a segunda, à viabilidade prática do sistema, já que tempos de espera longos comprometem a experiência de uso contínuo. São esses critérios que justificam o desenho do benchmark descrito no Capítulo 3.

2.2 Sistema M-Reader: contexto e evolução arquitetural

O M-Reader foi concebido como tecnologia assistiva de baixo custo para leitura de conteúdos impressos, executando sobre Raspberry Pi como plataforma de borda (REZENDE _et_ al., 2021a). Sua evolução recente representa uma mudança arquitetural relevante: a transição de um processamento majoritariamente local para um arranjo cliente-servidor.

Na versão inicial, a lógica de captura e processamento concentrava-se no dispositivo de borda, o que simplificava a implantação, mas limitava a escalabilidade e restringia o uso

de técnicas mais robustas. Por robustas entendem-se aqui as técnicas que continuam funcionando bem mesmo com imagens de baixa qualidade, mas que cobram um preço em capacidade de processamento: é o caso do pré-processamento mais pesado de imagem, com ampliação de resolução, correção de contraste e remoção de ruído, e principalmente dos modelos de IA multimodais usados na transcrição. No hardware limitado do Raspberry Pi, essas etapas se tornavam lentas demais ou simplesmente inviáveis.

Na versão atual, a aplicação cliente permanece responsável por captura e interação acessível, enquanto o servidor OCR assume o processamento intensivo, avaliação de qualidade e orquestração de provedores de transcrição.

Essa separação de responsabilidades traz ganhos concretos de engenharia: facilita a manutenção, permite evolução independente entre interface e *backend*, melhora o potencial de escalabilidade e reduz a pressão computacional no dispositivo de uso. Para um projeto assistivo, esse ponto é estratégico, preserva a fluidez da interação no cliente sem abrir mão da possibilidade de melhorias contínuas no processamento.

2.3 Reconhecimento óptico de caracteres (OCR)

OCR é o conjunto de técnicas que converte texto contido em imagens para formato editável e processável. Historicamente, os sistemas OCR evoluíram de abordagens baseadas em regras e segmentação rígida para métodos estatísticos e, mais recentemente, para modelos neurais e multimodais (SUBRAMANI *et al.*, 2021).

2.3.1 O motor Tesseract

O Tesseract é um dos motores OCR de código aberto mais utilizados no mundo. Originalmente desenvolvido pela HP e posteriormente mantido pelo Google, sua linha clássica de reconhecimento é documentada por Smith (2007). A partir da versão 4, o motor passou a integrar redes neurais recorrentes do tipo LSTM (Long Short-Term Memory, redes que conseguem "lembrar" contexto de caracteres anteriores ao reconhecer os seguintes) como componente central do reconhecimento, o que ampliou sua capacidade em documentos com layout variado (TESSERACT OCR, 2024).

Para o presente trabalho, três parâmetros de configuração do Tesseract são particularmente relevantes. O OEM (OCR Engine Mode) define qual motor de reconhecimento é usado; com o valor 3 (padrão), o Tesseract escolhe automaticamente entre o motor legado e o LSTM, priorizando este quando disponível. O PSM (Page Segmentation Mode) controla como o motor interpreta o layout da página: o modo 3 realiza segmentação automática, adequado para documentos com estrutura mista; o modo 6 trata a entrada como bloco uniforme de texto, útil para imagens pré-cortadas. Por fim, o parâmetro preserve_interword_spaces, quando ativado, mantém o espaçamento entre palavras, melhorando a legibilidade em documentos com layout tabulado ou colunar.

Apesar de sua maturidade e ampla adoção, o Tesseract apresenta degradação de desempenho em entradas com baixa legibilidade, ruído e variações fortes de contraste. Essa limitação explica, em parte, o movimento recente da literatura para arquiteturas mais robustas, capazes de capturar contexto visual e textual de forma conjunta.

2.3.2 Métricas de avaliação de OCR

A avaliação de qualidade em OCR compara o texto extraído com um texto de referência produzido por humanos. A métrica de base é a distância de Levenshtein (LEVENSHTEIN, 1966), um número que representa o mínimo de operações de inserção, remoção e substituição necessárias para transformar a saída do OCR no texto correto. Quanto menor a distância, mais fiel foi a transcrição. A partir dela derivam-se as métricas principais:

- **CER** (Character Error Rate): taxa de erro a nível de caractere, calculada como CER = d(r, h) / |r|, onde d é a distância de Levenshtein entre a referência r e a hipótese h;
- **WER** (Word Error Rate): análogo ao CER, mas comparando palavras inteiras;
- **Character Accuracy** (CA): complementar ao CER, definida como CA = (1 − CER) × 100;
- **Word Accuracy** (WA): complementar ao WER, definida como WA = (1 − WER) × 100;
- **Exact Match**: indicador binário de correspondência exata do texto completo (0 ou 100);
- **Average Accuracy**: métrica agregada adotada neste trabalho, calculada como:

\text{Average Accuracy} = \frac{\text{Character Accuracy} + \text{Word Accuracy} + \text{Exact Match}}{3}

Essa formulação atribui peso igual à precisão por caractere, por palavra e à correspondência exata. Na prática, o Exact Match tende a zero na maioria dos casos reais, qualquer diferença mínima de espaçamento ou pontuação impede a correspondência perfeita, de modo que a Average Accuracy acaba dominada pelos dois primeiros termos, com teto prático em torno de 66% mesmo para transcrições de alta qualidade. Essa característica deve ser considerada ao interpretar os valores absolutos da métrica; o que importa é a diferença entre métodos, não o número em si.

A clareza na definição e aplicação dessas métricas é fundamental para comparações justas, conforme recomendado por Neudecker *et* al. (2021). Para aplicações assistivas, a escolha do mecanismo OCR não deve considerar apenas acurácia em ambientes controlados, mas também estabilidade em condições reais, previsibilidade temporal e custo operacional por inferência.

2.4 Processamento de imagens para OCR

A qualidade da imagem de entrada é determinante para o desempenho do OCR. Técnicas de processamento de imagens são essenciais para aumentar a separabilidade entre texto e fundo e reduzir ambiguidades no reconhecimento (GONZALEZ; WOODS, 2018).

Entre as operações mais comuns de pré-processamento estão:

- **Conversão para escala de cinza**: reduz dimensionalidade da imagem sem perda significativa de informação textual;
- **Filtro bilateral**: suaviza ruído preservando bordas, técnica utilizada no pipeline legado do M-Reader;
- **Binarização adaptativa**: converte a imagem em preto e branco com limiares locais, melhorando contraste em condições de iluminação irregular;
- **Normalização de brilho e contraste**: equaliza a distribuição tonal da imagem;
- **Correção de inclinação e recorte**: alinha o texto ao eixo horizontal para melhorar segmentação;
- **Realce de bordas e nitidez**: destaca contornos de caracteres;
- **Refinamentos morfológicos**: operações de dilatação e erosão para limpar artefatos.

Em documentos reais, a combinação de problemas de captura (sombra, reflexo, foco imperfeito, baixa resolução e ruído) tende a afetar diretamente a qualidade da transcrição final. Por essa razão, abordagens modernas tratam o pré-processamento como parte de um fluxo adaptativo, e não como sequência fixa de filtros. No M-Reader, essa premissa se traduz em múltiplos modos de processamento (simple, advanced, enhanced, hybrid), cada um com diferente nível de intervenção sobre a imagem.

2.5 Inteligência artificial aplicada a documentos

A inteligência artificial, especialmente por meio de aprendizado profundo (deep learning), consolidou-se como recurso central em tarefas de reconhecimento e compreensão de documentos (GOODFELLOW; BENGIO; COURVILLE, 2016). A literatura recente evidencia forte avanço de modelos baseados em Transformer e visão-linguagem para tarefas de OCR e Document AI.

TrOCR (LI *et* al., 2021) demonstra ganhos relevantes de reconhecimento textual com modelos pré-treinados. LayoutLMv3 (HUANG *et* al., 2022) e Donut (KIM *et* al., 2022) ampliam a discussão para inteligência documental, incorporando estrutura visual ao processo de compreensão. Nougat (BLECHER *et* al., 2023) reforça o potencial de modelos especializados para documentos acadêmicos complexos. Mais recentemente, olmOCR (POZNANSKI *et* al., 2025) demonstrou que modelos visão-linguagem de larga escala, sistemas que "leem" imagem e texto de forma integrada, podem processar documentos PDF com ganhos de robustez em cenários heterogêneos.

Uma visão panorâmica dessas abordagens pode ser encontrada em Subramani *et* al. (2021), que sistematizam as principais arquiteturas, conjuntos de dados e desafios abertos da área.

Esses modelos ampliam a capacidade de leitura onde pipelines tradicionais apresentam queda de desempenho, mas introduzem novos desafios: custo de inferência, latência, dependência de infraestrutura e governança de dados. Para sistemas assistivos, isso significa que a decisão técnica não é simplesmente "substituir o OCR tradicional", mas definir estratégia de uso inteligente entre métodos, de acordo com a qualidade da entrada e o objetivo operacional. No M-Reader, essa estratégia se materializa no uso da API da OpenAI (OPENAI, 2026) como provedor de retorno alternativo.

2.6 Arquitetura de software e padrão cliente-servidor

A engenharia de software fornece os princípios que sustentam a organização do M-Reader como sistema modular e evolutivo. Conforme Sommerville (2019), a decomposição de um sistema em componentes com responsabilidades bem definidas é pré-requisito para manutenibilidade, testabilidade e escalabilidade, especialmente em projetos de longa duração com ciclos incrementais de melhoria.

Um dos arranjos mais consolidados para essa decomposição é o padrão cliente-servidor, no qual o sistema se organiza em serviços oferecidos por um servidor e consumidos por clientes por meio da rede (SOMMERVILLE, 2019). A adoção desse padrão em soluções de OCR assistivo equilibra as limitações de borda com a necessidade de algoritmos mais resilientes: o dispositivo local oferece menor latência de rede, porém menor capacidade para modelos complexos, enquanto o processamento remoto ganha poder computacional e flexibilidade de evolução, ao custo de dependência de conectividade e aumento de tempo em certas requisições.

2.6.1 Estilo arquitetural REST

A comunicação entre cliente e servidor segue o estilo arquitetural REST (Representational State Transfer), definido por Fielding (2000). Seus princípios fundamentais, interface uniforme, ausência de estado entre requisições, separação entre representação e recurso, são adequados para o cenário do projeto, no qual o cliente envia imagem, recebe texto processado e consulta histórico por meio de endpoints HTTP padronizados.

A implementação do servidor utiliza o micro framework Flask (FLASK, 2026). Trata-se de uma escolha pragmática, e não de um diferencial da solução: o Flask atende ao que o projeto precisa, expor o pipeline de OCR como serviço HTTP com pouco código, e o padrão de blueprint adotado permite organizar rotas por domínio funcional sem acoplamento rígido entre módulos. A vantagem da solução atual está na arquitetura em si, na separação entre cliente e servidor e na estratégia híbrida de OCR, e não no framework utilizado.

2.6.2 Arquiteturas híbridas com fallback

Nesse cenário, arquiteturas híbridas com retorno alternativo (fallback) por qualidade tornam-se especialmente relevantes: o método tradicional atende casos simples com maior rapidez, enquanto o método multimodal é acionado quando há baixa confiabilidade no resultado inicial. Essa lógica reduz acionamentos desnecessários do método mais custoso e melhora a robustez em casos críticos.

A opção por combinar provedores diferentes tem apoio empírico. Tafti et al. (2016), ao comparar quatro serviços de OCR (Google Docs, Tesseract, ABBYY FineReader e Transym) sobre um mesmo conjunto de imagens, observaram que nenhum motor é superior em todos os cenários: o desempenho varia conforme o tipo de imagem e as condições de

captura. Se nenhum provedor vence sempre, faz mais sentido o sistema saber alternar entre eles conforme o caso do que apostar tudo em um único método.

Além do ganho de acurácia, o fallback também protege a disponibilidade do sistema. Se o provedor remoto estiver fora do ar, por falha de rede ou da própria API, o resultado do OCR local continua existindo e pode ser entregue ao usuário. Para uma tecnologia assistiva, essa degradação controlada importa: é melhor entregar uma leitura imperfeita do que nenhuma leitura, e o usuário não fica refém da conexão com a internet.

2.7 Métricas de avaliação para análise comparativa

A avaliação de desempenho em OCR pode incluir métricas de erro textual (como CER e WER), métricas agregadas de acurácia e métricas temporais. No recorte deste trabalho, os artefatos disponíveis do benchmark permitiram comparar métodos por:

- acurácia média por estágio;
- acurácia de palavras (*word accuracy*);
- ganho absoluto em pontos percentuais;
- ganho relativo percentual;
- tempo médio de execução e diferença temporal entre abordagens.

Essa estratégia é coerente com recomendações de avaliação em OCR que enfatizam clareza metodológica, comparação justa e explicitação de limitações de amostra (NEUDECKER *et* al., 2021). Comparação justa, nesse contexto, significa avaliar todos os métodos nas mesmas condições: mesmo conjunto de imagens, mesmos textos de referência e mesmas métricas. Se cada método fosse testado com imagens diferentes, não daria para saber se a diferença de acurácia vem do método em si ou da dificuldade das imagens que couberam a cada um. É por isso que, no benchmark deste trabalho, os cinco estágios do pipeline são executados sobre o mesmo conjunto de imagens e os resultados comparados par a par, como detalhado no Capítulo 3.

2.8 Trabalhos correlatos integrados à fundamentação

2.8.1 OCR neural e modelos pré-treinados

LI *et* al. (2021) com TrOCR demonstram o avanço dos modelos pré-treinados baseados em Transformer sobre o OCR tradicional: o modelo atingiu taxa de erro de caractere de 2,89% no reconhecimento de manuscritos da base IAM e F1 de 96,58% na leitura de recibos impressos da base SROIE, superando nos dois casos os melhores métodos anteriores, baseados em redes convolucionais. HUANG *et* al. (2022) e KIM *et* al. (2022) ampliam a discussão para inteligência documental, incorporando estrutura visual ao processo de compreensão de documentos. BLECHER *et* al. (2023), com Nougat, reforçam o potencial de modelos especializados para documentos complexos, aqueles em que o conteúdo não se resume a texto corrido: páginas com tabelas, fórmulas matemáticas, múltiplas colunas e figuras misturadas ao texto, comuns em livros didáticos e artigos científicos.

O escopo do M-Reader foi concebido para a leitura de livros e outros conteúdos impressos (REZENDE et al., 2021a), e não há registro no projeto de avaliação com páginas assim. Já a limitação diante desses elementos vem do próprio motor: BLECHER et al. (2023) apontam que motores tradicionais como o Tesseract reconhecem bem caracteres e palavras isoladas, mas, por processarem o texto linha a linha, não capturam a relação entre os elementos, tratando sobrescritos e subscritos de uma fórmula como se fossem texto comum. Com a nova arquitetura, espera-se que o estágio multimodal amplie o tipo de material acessível ao usuário, já que esses modelos interpretam a página como um todo e têm mais condição de preservar a estrutura do conteúdo. Esse ganho, porém, não foi medido aqui: as imagens do benchmark contêm apenas texto corrido em prosa (ver Seção 3.5.2), então a leitura de documentos complexos fica como expectativa apoiada nos correlatos, e não como resultado verificado neste trabalho.

Contribuição para este TCC: esses estudos sustentam a decisão de que o último estágio do pipeline do M-Reader fosse um modelo multimodal, e não mais um motor de OCR tradicional. Os modelos baseados em Transformer leem bem justamente os casos em que o OCR clássico falha, como manuscritos e imagens degradadas. Por isso o M-Reader recorre a um modelo dessa família quando os estágios locais não dão conta. O impacto dessa escolha aparece medido no Capítulo 5: é o estágio multimodal que eleva a acurácia de palavras do sistema de 74,82% para 90,92% nas imagens do benchmark.

2.8.2 Benchmarks e avaliação de OCR

NEUDECKER *et* al. (2021) evidenciam que comparações entre métodos OCR exigem protocolos claros, métricas consistentes e explicitação de contexto experimental. No presente trabalho, o protocolo corresponde ao desenho descrito no Capítulo 3, em que os cinco estágios rodam sobre as mesmas 222 imagens e as saídas são conferidas com os textos de referência, de modo que a avaliação possa ser repetida por outra pessoa. As métricas são as apresentadas na Seção 2.7, calculadas da mesma forma para todos os estágios, senão os números não seriam comparáveis. O contexto experimental, por sua vez, abrange as condições adversas de captura e o fato de o provedor remoto não ter retornado resultado em 16 das 222 imagens, o que obriga a diferenciar a comparação global da comparação pareada.

Contribuição para este TCC: fornece base metodológica para análise crítica dos resultados e para evitar conclusões superdimensionadas.

2.8.3 OCR multimodal em larga escala

POZNANSKI *et* al. (2025) discutem OCR com modelos visão-linguagem em escala ampliada. O olmOCR, apresentado pelos autores, foi treinado e avaliado sobre mais de cem mil documentos PDF de tipos variados, entre artigos acadêmicos, folhetos, documentos jurídicos, tabelas, diagramas e livros digitalizados, incluindo páginas manuscritas, digitalizações de baixa qualidade e layouts de múltiplas colunas. Essa mistura de tipos de documento e de condições de qualidade é o que os autores chamam de cenário heterogêneo, e o resultado que interessa aqui é que o modelo manteve desempenho consistente em toda essa variedade, à frente de ferramentas especializadas na avaliação comparativa conduzida no estudo.

Contribuição para este TCC: reforça a direção arquitetural híbrida, na qual o método multimodal é acionado sob critério de qualidade, e contextualiza os resultados obtidos em relação à literatura.

2.8.4 IA em tecnologias assistivas

TRIYONO *et* al. (2023) apresentam revisão sistemática sobre IA para deficiência visual, destacando potencial de autonomia, mas também desafios práticos de implantação, custo e confiabilidade. BUDRIONIS *et* al. (2022) revisam electronic travel aids. baseados em smartphone e visão computacional, apontando lacunas de integração háptica e de métodos neurais de ponta — perspectiva útil para discutir trade-offs de latência e usabilidade em dispositivos móveis. YAO, ZHOU e HU (2025), em revisão sobre sistemas assistivos visuais, sintetizam linhas de pesquisa que frequentemente combinam sensores, visão e retroalimentação ao usuário, em vez de depender de uma única técnica isolada.

Contribuição para este TCC: essas revisões se refletem em decisões concretas do trabalho. Os desafios de custo e confiabilidade apontados por TRIYONO et al. (2023) são o que motiva manter o OCR local gratuito como caminho padrão e acionar o provedor pago apenas quando necessário. A preocupação de BUDRIONIS et al. (2022) com latência e usabilidade em dispositivos limitados é o motivo de o tempo de processamento ser a segunda variável do benchmark, ao lado da acurácia, e de a hipótese fixar um teto de 15 segundos para o custo temporal. E a observação de YAO, ZHOU e HU (2025), de que os sistemas assistivos raramente se apoiam em uma técnica só, descreve bem o próprio pipeline do M-Reader, que encadeia pré-processamento de imagem, OCR local e modelo multimodal em estágios.

2.8.5 Quadro comparativo dos trabalhos correlatos

Tabela 1 - Quadro comparativo dos trabalhos correlatos

Autor/Ano	**Abordagem**	**Contribuição principal**	**Limitação**	**Relação com M-Reader**
LI *et* al. (2021)	TrOCR (Transformer)	OCR pré-treinado com ganhos em texto impresso	Requer GPU para inferência	Sustenta uso de modelo neural como fallback
HUANG *et* al. (2022)	LayoutLMv3	Compreensão documental multimodal	Foco em extração estruturada, não em leitura corrida	Amplia visão de Document AI

KIM *et* al. (2022)	Donut	OCR sem pipeline explícito	Latência elevada em documentos longos	Reforça potencial de end-to-end
BLECHER *et* al. (2023)	Nougat	OCR para documentos acadêmicos	Restrito a PDFs acadêmicos	Valida modelos especializados
POZNANSKI *et* al. (2025)	olmOCR (VLM)	OCR em larga escala com VLMs	Custo computacional elevado	Referência direta para fallback multimodal
NEUDECKER *et* al. (2021)	Survey de métricas OCR	Protocolo de avaliação justo	Foco em documentos históricos	Base metodológica para benchmark
TRIYONO *et* al. (2023)	Revisão de IA para DV	Mapa de soluções assistivas	Não analisa custo temporal	Contextualiza uso de IA em TA
BUDRIONIS *et* al. (2022)	Revisão de ETAs com smartphone e visão	Lacunas de integração e métodos neurais	Foco em deslocamento, não em OCR de documento	Evidência de trade-off tempo/qualidade em mobilidade
YAO *et* al. (2025)	Revisão de sistemas visuais para DV	Estado da arte em visão assistiva	Escopo amplo, pouca profundidade por sistema	Valida abordagem combinada

Fonte: elaborado pelo autor (2026).

2.8.6 Lacunas identificadas

A integração dos trabalhos correlatos aponta lacunas relevantes:

1. poucos estudos aplicados com foco explícito em soluções assistivas de baixo custo em produção real;
2. escassez de análises que quantifiquem simultaneamente ganho de acurácia e custo temporal em estratégia híbrida;
3. baixa disponibilidade de relatos técnicos que descrevem, com dados de projeto, a transição arquitetural de OCR local para cliente-servidor com retorno alternativo por qualidade.

2.9 Síntese do capítulo

Os fundamentos reunidos neste capítulo mostram que o M-Reader se apoia em três frentes complementares: a demanda social por acessibilidade informacional, o avanço de abordagens neurais e multimodais em OCR, e os princípios de engenharia que sustentam sistemas distribuídos com fallback. A revisão dos trabalhos correlatos também deixou claro onde estão as lacunas da literatura, em particular, a escassez de estudos que quantifiquem juntamente acurácia e custo temporal em sistemas assistivos reais. Os capítulos seguintes tratam do que foi efetivamente medido no benchmark do projeto.

3 - METODOLOGIA

Este capítulo descreve o percurso metodológico adotado para investigar os impactos da evolução do M-Reader para arquitetura cliente-servidor com OCR híbrido. O núcleo da investigação é o delineamento comparativo com medidas repetidas sobre o mesmo acervo de imagens: cada unidade observada (cada imagem) é submetida a vários “tratamentos” algorítmicos (estágios do pipeline), registrando-se acurácia e tempo. O ciclo PDCA, apresentado na seção 3.3, funciona como organização retrospectiva do relato e checklist de rastreabilidade, não substitui esse desenho nem equivale a um experimento prospectivo conduzido exclusivamente sob o formalismo da gestão da qualidade.

3.1 Tipo de pesquisa

A pesquisa é de natureza aplicada, com abordagem quantitativa e delineamento comparativo com medidas repetidas sobre unidades fixas (WAZLAWICK, 2014; GIL, 2017). O caráter aplicado decorre do foco em um sistema assistivo real em operação. A abordagem quantitativa está associada à análise de métricas objetivas extraídas dos artefatos de benchmark, no caso a acurácia média (average accuracy) e a acurácia de palavras (word accuracy), cujo cálculo foi apresentado na seção 2.3.2, e o tempo de execução de cada método em segundos.

Esse delineamento aparece no desenho do benchmark, em que as 222 imagens funcionam como as unidades fixas e todos os métodos previstos são aplicados a cada uma delas. A variável que o estudo manipula é o estágio do pipeline (old, simple, advanced, enhanced, *openai*), e por isso ela é a independente; acurácia e tempo apenas respondem a essa escolha, então entram como variáveis dependentes. Não há alocação aleatória de imagens a um único método, o que caracteriza medidas repetidas, não grupos paralelos. O provedor multimodal não retornou saída em 16 das 222 imagens, o que reduz para 206 o

subconjunto disponível para comparação pareada. A inferência principal sobre ganho de acurácia apoia-se exclusivamente nesse subconjunto pareado.

A classificação tem ainda um componente descritivo, porque parte do trabalho não compara nada, apenas documenta a transição arquitetural do processamento local para o arranjo cliente-servidor com retorno alternativo orientado por qualidade Na prática, documentar essa transição a partir do código existente exige tratar o repositório como fonte primária, o código é o registro mais confiável do que foi de fato implementado, e não raro há diferenças entre a intenção original e a solução efetiva.

3.1.1 Estrutura do delineamento comparativo

O benchmark compara cinco estágios do pipeline de OCR sobre o mesmo conjunto de imagens. A estrutura formal do delineamento é:

- **Variável independente**: método de processamento OCR (old, simple, advanced, enhanced, openai);
- **Variáveis dependentes**: (i) acurácia média (*average accuracy*); (ii) acurácia de palavras (*word accuracy*); (iii) tempo de execução (*execution_time_sec*);
- **Variáveis de controle**: hardware de execução (servidor local para métodos Tesseract, API remota para OpenAI), conjunto fixo de imagens (222), textos de referência inalterados, configuração do Tesseract (PSM 3, OEM 3) e rede local para comunicação cliente-servidor.

A definição das variáveis segue o problema da pesquisa. A independente não foi propriamente escolhida entre alternativas: os cinco métodos são os que existem no histórico do sistema, e o que se quer saber é justamente se essa evolução melhorou a leitura, então é o método que precisa variar. As dependentes vieram do que o M-Reader entrega ao usuário. A acurácia média resume a qualidade geral da transcrição em um único número, o que facilita comparar cinco métodos de uma vez; a acurácia de palavras foi incluída porque o texto extraído chega ao usuário por síntese de voz, e para quem ouve é a palavra errada que compromete o sentido, não um caractere trocado; o tempo de execução entra porque a leitura é interativa, e não adianta muito transcrever melhor se a resposta demora além do tolerável, tanto que a hipótese do trabalho estabelece o teto de 15 segundos.

As variáveis de controle reúnem o que também mexeria na acurácia e no tempo se mudasse no meio do caminho. Se cada método rodasse com outro conjunto de imagens, outra configuração do Tesseract ou outra rede, a diferença medida poderia vir dessas condições, e não do método. Fixá-las é o que permite atribuir ao método de processamento o efeito observado sobre a qualidade da transcrição, embora fatores como variabilidade de latência de rede e estado momentâneo do provedor remoto ainda introduzam ruído não controlável.

Como os tamanhos amostrais diferem entre colunas (222 vs. 206 no OpenAI), adotaram-se duas leituras: global, descritiva, comparando médias com todas as observações disponíveis por método; e pareada, restrita às 206 imagens com resultado tanto em `openai` quanto em `old`. A conclusão principal sobre ganho de acurácia apoia-se

na leitura pareada; o teste de Wilcoxon signed-rank (seção 3.6.1) aplica-se a esses 206 pares. A diferença bruta entre médias globais (63,64% em 206 imagens vs. 57,65% em 222) não substitui essa inferência.

Vale esclarecer o papel do método old nesse delineamento: ele corresponde ao pipeline que operava localmente no Raspberry Pi antes da migração, enquanto os demais estágios integram a arquitetura cliente-servidor atual. Por isso, as diferenças de acurácia observadas referem-se sobretudo ao estágio openai, viabilizado pela nova arquitetura, e não isolam o efeito da mudança de arquitetura desacompanhada do provedor multimodal.

Outro ponto do protocolo merece registro: o estágio openai foi executado em todas as imagens com retorno válido, justamente para permitir a comparação pareada. Em uso real, com a política seletiva, o multimodal nem sempre é acionado. O tempo médio reportado no Capítulo 5 representa, portanto, o custo quando esse estágio é atingido nas condições do benchmark, e não a latência média ponderada do modo automático em produção.

3.2 Desenho metodológico geral

O desenho metodológico foi organizado em cinco macro etapas interdependentes:

1. Mapeamento arquitetural do fluxo anterior e do fluxo atual do M-Reader;
2. Definição das estratégias de comparação entre os estágios do pipeline de OCR;
3. Coleta e consolidação de dados a partir dos artefatos de benchmark;
4. Análise quantitativa das métricas de acurácia e tempo;
5. Interpretação técnica dos resultados à luz das restrições de uso assistivo.

Essas etapas foram executadas em sequência, com retornos pontuais quando a análise pedia nova conferência dos dados. O PDCA, apresentado na seção 3.3, organiza o relato desse percurso em retrospecto, não o guiou em tempo real.

3.3 Ciclo PDCA como ferramenta complementar de organização

Para organizar o relato do processo de trabalho, utilizou-se o ciclo PDCA (Plan-Do-Check-Act) como referência retrospectiva, conforme Werkema (2012). O ciclo vem da gestão da qualidade, área em que qualidade significa o quanto um processo e seus resultados atendem aos requisitos definidos para eles, e o PDCA é o método de planejar, executar, verificar e agir para que esses requisitos continuem sendo atendidos. No caso deste capítulo, o processo é o próprio benchmark, e os requisitos são as regras de conferência e leitura dos dados. É um sentido diferente do critério de qualidade que aciona o retorno alternativo no pipeline, em que qualidade é a confiabilidade estimada da transcrição de cada imagem. As quatro fases descrevem o percurso do benchmark o que foi planejado, executado e verificado sem confundir gestão da qualidade com o núcleo inferencial do delineamento comparativo.

PLAN — Caracterização das 222 imagens; identificação do risco de viés pela diferença entre 222 e 206 retornos; definição das perguntas analíticas (melhor estágio

médio, ganho pareado openai vs. old, custo temporal); artefatos oficiais benchmark_results.json e benchmark_summary.csv.

DO — Execução do benchmark por estágios; registro de métricas por imagem; consolidação em JSON/CSV; rastreio das 16 falhas do provedor remoto.

CHECK — Auditoria de cardinalidade (222 vs. 206); validação cruzada JSON/CSV; separação obrigatória entre leitura global e pareada; verificação conjunta de acurácia e tempo.

ACT — Padronização das regras de leitura na monografia; incorporação das limitações identificadas; registro de melhorias para próximos ciclos.

Na prática, a fase ACT foi executada em paralelo à escrita. Cada decisão sobre como apresentar os resultados — qual média citar, qual comparação priorizar — foi, em si, uma ação de padronização retrospectiva.

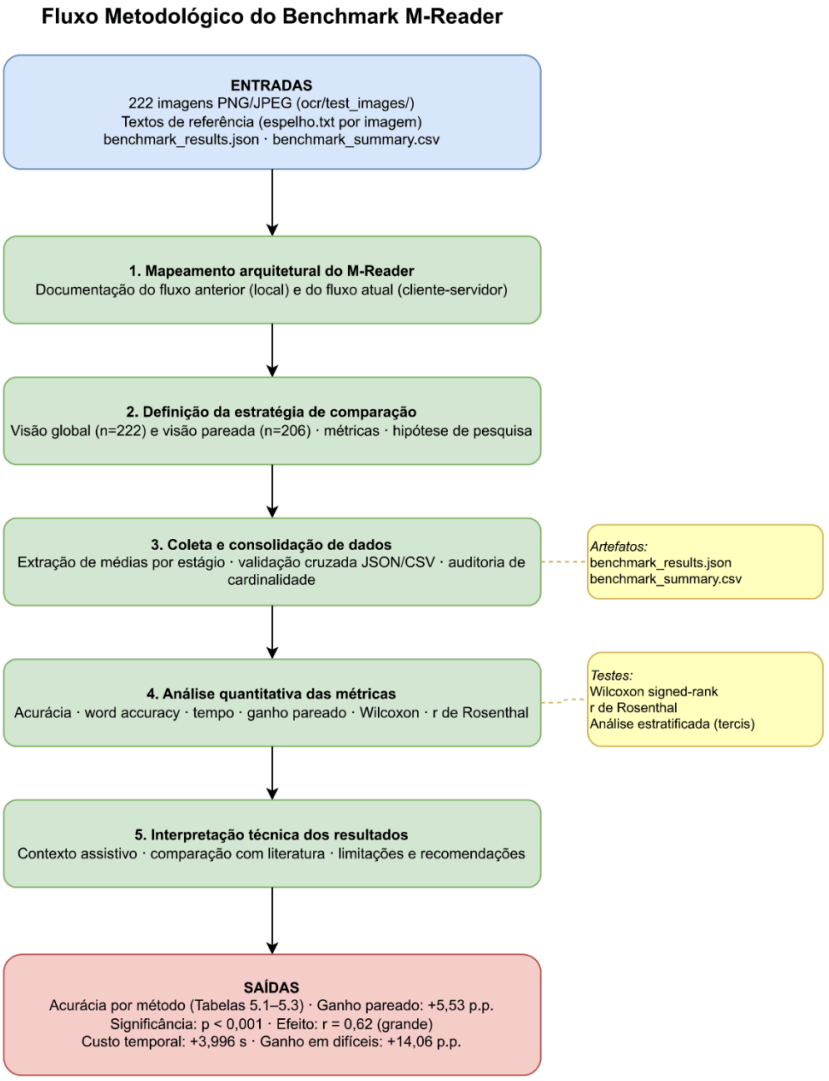
Tabela 2 - Atividades realizadas no ciclo PDCA da metodologia

Etapa	**Atividades executadas**	**Artefatos gerados/atualizados**	**Resultado esperado**
PLAN	Definição de problema, hipóteses, variáveis, critérios de comparação e riscos de viés por amostra desigual	Protocolo metodológico e premissas analíticas registradas no capítulo	Base comparativa consistente para execução e análise
DO	Execução do benchmark por estágios (old, simple, advanced, enhanced, openai) e consolidação das saídas	benchmark_results.json e benchmark_summary.csv	Dados rastreáveis para cálculo de acurácia e tempo
CHECK	Auditoria de cobertura, validação cruzada JSON/CSV, separação entre análise global e	Relatório analítico consolidado para interpretação dos resultados	Redução de viés interpretativo e aumento da validade interna

	pareada e verificação de trade-off		
ACT	Padronização de regras de leitura, incorporação de limitações e definição de ações para próximos ciclos	Ajustes no texto metodológico e recomendações técnicas	Melhoria contínua do método e da evolução do sistema

Fonte: elaborado pelo autor (2026).

3.4 Diagrama do fluxo metodológico



Fonte: elaborado pelo autor (2026)

Fonte: Elaborado pelo autor (2026)

3.5 Ambiente de teste e fontes de dados

Os resultados do Capítulo 5 dependem do contexto em que foram produzidos, ou seja, de qual ambiente executou cada método, sobre quais imagens e a partir de quais arquivos. É esse contexto que esta seção registra, com a infraestrutura de hardware e software na 3.5.1, o conjunto de imagens de teste na 3.5.2 e as fontes de dados do ciclo analisado na 3.5.3.

3.5.1 Infraestrutura de hardware e software

O benchmark foi executado no seguinte ambiente:

Tabela 3 - Ambiente de execução do benchmark

Componente	**Especificação**
Sistema operacional (servidor)	Windows 10
Linguagem	Python 3.11.0
Framework web	Flask 3.1.2
Motor OCR local	Tesseract 5.5.0 (LSTM, OEM 3)
Biblioteca OCR Python	pytesseract 0.3.13
Processamento de imagem	OpenCV 4.12.0, Pillow 11.3.0
Provedor multimodal	OpenAI API (modelo gpt-4.1-mini)
Banco de dados	SQLite 3 (embarcado)
Conversão PDF	pdf2image 1.17.0, Poppler
Cliente (dispositivo de borda)	Raspberry Pi (wxPython)

Fonte: elaborado pelo autor (2026), com base em ocr/requirements.txt e ambiente de execução.

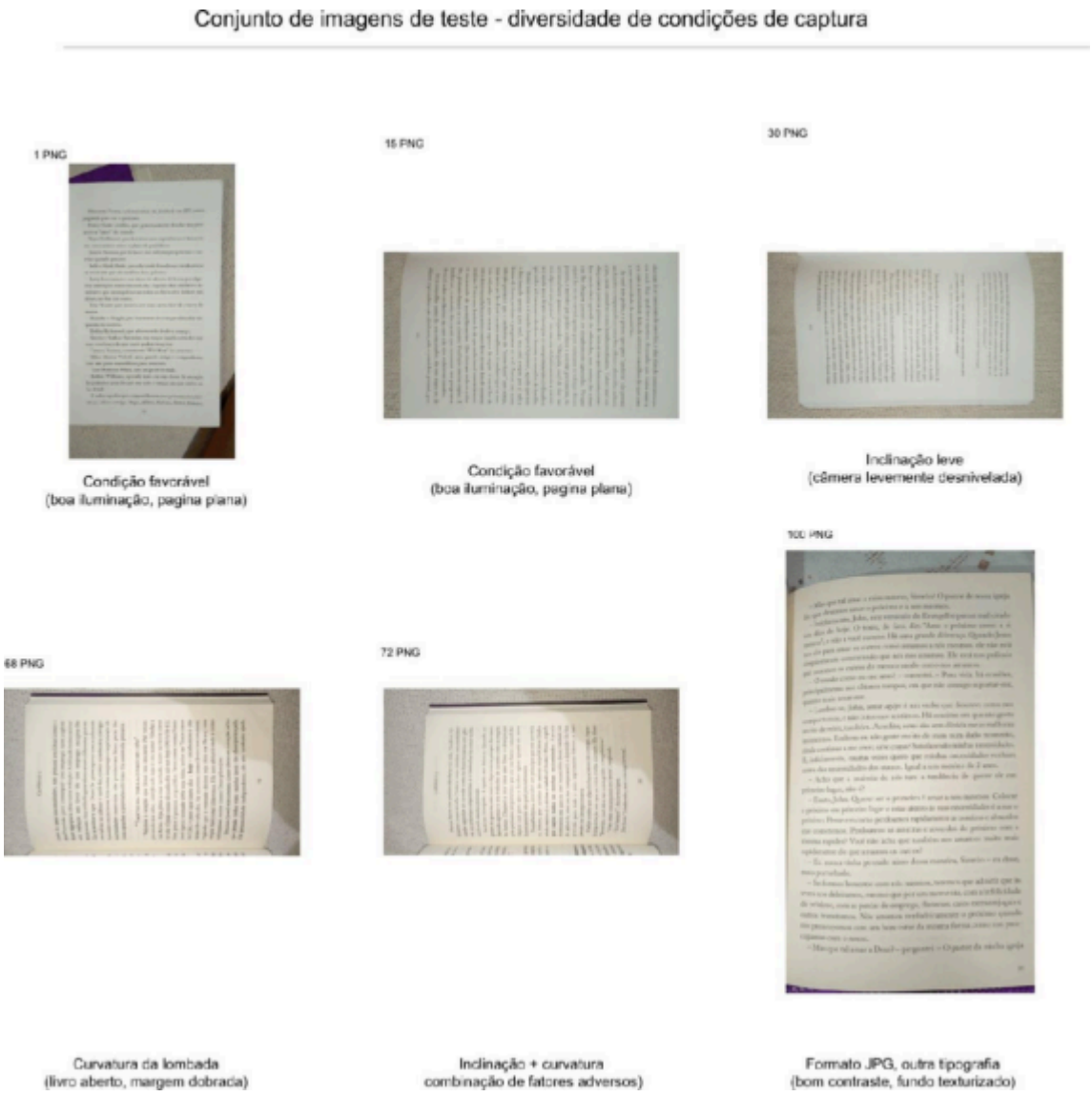
O benchmark foi executado em rede local, com servidor e cliente na mesma infraestrutura, minimizando a variabilidade de latência de rede. Para o provedor OpenAI, o acesso ocorreu via internet, introduzindo dependência de conectividade externa.

3.5.2 Caracterização do conjunto de imagens de teste

O repositório do projeto contém 222 imagens de teste no diretório `ocr/test_images/`, todas utilizadas no ciclo de benchmark analisado. As imagens possuem as seguintes características:

- **Tipo de documento**: páginas de livros impressos em português;
- **Formato**: PNG (imagens 1–75) e JPEG (imagens 76+);
- **Resolução média**: aproximadamente 2 megapixels por imagem;
- **Condição de captura**: fotografias realizadas com câmera do Raspberry Pi em ambiente não controlado, sem tripé fixo, apresentando variações de iluminação, sombra parcial, leve inclinação e foco não uniforme;
- **Conteúdo textual**: texto corrido em prosa, sem tabelas, fórmulas ou elementos gráficos complexos;
- **Texto de referência**: para cada imagem existe um arquivo _espelho.txt com transcrição humana de referência no diretório ocr/texts/.

Protocolo de criação dos textos de referência. Os textos espelho foram produzidos pelo próprio autor do trabalho mediante transcrição manual do conteúdo visível em cada imagem. O processo consistiu em: (i) leitura visual da imagem original; (ii) digitação integral do texto, preservando pontuação, acentuação e estrutura de parágrafos; (iii) revisão comparativa entre texto digitado e imagem. Não foi adotado protocolo de dupla anotação com cálculo de concordância entre anotadores — limitação reconhecida, adotada por restrição de tempo e de recursos do escopo deste TCC, com impacto potencial na estimativa de erro de rotulagem. NEUDECKER *et* al. (2021) reforçam a importância de protocolos e métricas explícitas na avaliação de OCR; não prescrevem anotação única como padrão a seguir.



Fonte: Elaborado pelo autor (2026)

Os textos espelho foram produzidos pelo próprio autor mediante transcrição manual: leitura visual da imagem, digitação integral do texto com preservação de pontuação e acentuação, e revisão comparativa. Não foi adotado protocolo de dupla anotação com cálculo de concordância entre anotadores — limitação reconhecida, decorrente de restrição de tempo e recursos do escopo deste TCC.

Essa caracterização é relevante porque os resultados refletem um cenário de captura real com hardware de baixo custo, diferenciando-se de benchmarks acadêmicos que operam sobre PDFs nativamente digitais ou imagens escaneadas em alta resolução. Um detalhe que só se percebe na transcrição manual: algumas imagens estão em condições tão adversas que mesmo a leitura humana exige esforço. São exatamente essas onde o OCR falha com maior frequência, o que sugere que a heurística de quality_score, ao usar o resultado textual como proxy de legibilidade, está capturando algo genuíno sobre a dificuldade da entrada.

3.5.3 Fontes de dados e ciclo analisado

As análises foram consolidadas a partir de dois artefatos de benchmark compatíveis, produzidos sobre a mesma base de imagens.

- Benchmark local (timestamp 20260307_005504): executou os quatro métodos Tesseract (old, simple, advanced, enhanced) em todas as 222 imagens;
- Benchmark OpenAI-only (timestamp 20260410_235329): executou exclusivamente o provedor multimodal (openai_gpt) nas 222 imagens, obtendo retorno em 206 delas.

A consolidação de duas execuções é justificada porque: o conjunto de imagens permaneceu inalterado; os textos de referência não foram modificados; e a função de cálculo de métricas não sofreu alteração entre as execuções. As 16 imagens sem retorno multimodal caracterizam dados ausentes nessa condição, exigindo leitura global cautelosa e análise pareada restrita às 206 imagens com resultado completo.

Uma nota sobre o tempo do estágio openai: no benchmark, o método foi invocado para todas as 206 imagens com retorno, a fim de obter métricas comparáveis. No uso real com provider=auto, o provedor multimodal aciona-se apenas quando a heurística de qualidade indica baixa confiabilidade do Tesseract. O tempo médio do openai no Capítulo 5 é, portanto, o custo do estágio quando executado, um proxy do pior caso de latência quando o fallback dispara, não uma estimativa do tempo médio do modo automático em campo.

3.6 Estratégia de comparação e métricas

Para garantir validade analítica, a comparação foi segmentada em:

- Visão global: comparação de médias considerando todos os registros disponíveis em cada método (222 para métodos locais, 206 para OpenAI);
- Visão pareada: comparação openai versus old apenas nas 206 imagens com resultados para ambos os métodos.

As métricas utilizadas foram:

- Average_accuracy (conforme definição formal na seção 2.3.2);
- Word_accuracy (acurácia de palavras);
- Ganho absoluto em pontos percentuais;
- Ganho relativo percentual;
- Tempo médio por método (execution_time_sec);
- Diferença temporal entre abordagens.

3.6.1 Teste de significância estatística

Para verificar se a diferença de acurácia entre openai e old é estatisticamente significativa, utilizou-se o teste de Wilcoxon signed-rank (FIELD, 2018; CONOVER, 1999). Esse teste é indicado para comparações pareadas quando não se pode assumir distribuição normal dos dados — que é o caso aqui, dado que a average_accuracy apresenta distribuição assimétrica.

O teste analisa as diferenças entre pares de valores de acurácia, ordenando-as e verificando se há tendência consistente de melhora ou piora. A hipótese alternativa adotada foi unilateral (H_1:acurácia OpenAI > acurácia old). Como medida de tamanho de efeito, foi calculado o coeficiente r de Rosenthal: $r = |z| / \sqrt{n}$, onde z é a aproximação normal da estatística de teste e n é o número de pares. Valores de r acima de 0,5 indicam efeito grande (FIELD, 2018).

3.6.2 Análise estratificada por dificuldade

Para avaliar se o ganho do método multimodal é uniforme ou concentrado em determinados perfis de imagem, os 206 pares foram divididos em três faixas de dificuldade com base na acurácia do método old: tercil inferior (imagens difíceis), tercil central (medianas) e tercil superior (fáceis). Essa estratificação permite testar a hipótese de que o *fallback* multimodal beneficia predominantemente imagens em que o OCR tradicional apresenta maior degradação.

3.7 Procedimento analítico passo a passo

O procedimento de análise seguiu a sequência abaixo:

1. Leitura do bloco summary no JSON;
2. Validação cruzada com o CSV;
3. Extração das médias por estágio;
4. Cálculo de ganho de acurácia em comparação pareada;
5. Cálculo de delta temporal pareado;
6. Interpretação dos resultados no contexto assistivo.

3.8 Critérios de validade e limitações

Para fortalecer a validade interna, os dados foram extraídos de artefatos versionados gerados pelo próprio sistema. Ainda assim, há limitações:

1. Amostra de 222 imagens oferece base sólida para análise, mas não cobre todos os tipos documentais possíveis (manuscritos, formulários, documentos com tabelas);
2. Dados ausentes no método OpenAI para 16 das 222 imagens;
3. Consolidação de duas execuções de benchmark distintas, embora sobre a mesma base e com mesma lógica de métricas;
4. Influência de variáveis de rede no tempo de métodos remotos;
5. Textos de referência produzidos por anotador único, sem protocolo de dupla anotação;
6. Ausência de avaliação longitudinal com usuários finais nesta etapa;
7. Tempos do estágio openai obtidos com o método aplicado a todas as imagens com retorno no benchmark, e não com política seletiva de *fallback* como em produção.

3.9 Considerações éticas e operacionais

Como o sistema processa imagens de documentos, a arquitetura remota requer atenção à privacidade, controle de acesso e minimização de exposição de dados sensíveis.

O trabalho reconhece esse ponto como requisito de continuidade técnica, incluindo recomendações de governança, retenção mínima de dados e monitoramento de uso de provedores. A decisão de armazenar apenas o texto extraído e não a imagem original é, além de uma escolha técnica de eficiência, uma forma de minimizar exposição de conteúdo potencialmente privado: livros fotografados em contexto assistivo podem conter anotações manuscritas, nomes e informações pessoais do usuário.

3.10 Síntese metodológica

A metodologia articula delineamento comparativo com medidas repetidas sobre 222 imagens, leituras global e pareada, teste de Wilcoxon e análise estratificada por dificuldade. O ciclo PDCA aparece como síntese retrospectiva do processo de benchmark, sem confundir gestão da qualidade com o núcleo inferencial. As conclusões sobre acurácia e tempo consideram as limitações da seção 3.8, em especial a diferença entre o benchmark integral do estágio openai e o fallback seletivo em uso real.

4- DESENVOLVIMENTO E IMPLEMENTAÇÃO

Este capítulo descreve a implementação atual do M-Reader com base no código do repositório. A organização segue a ordem em que o trabalho evoluiu: primeiro as tentativas de melhoria do OCR local e o que os dados de benchmark mostraram; em seguida a decisão de separar captura e processamento; por fim a arquitetura cliente-servidor, os contratos entre camadas, a política de fallback e o papel do benchmark integrado ao projeto. Assim, as decisões de engenharia ficam contextualizadas pelos resultados que as motivaram.

4.1 O que foi tentado antes da arquitetura cliente-servidor

4.1.1 Ponto de partida

A versão inicial do M-Reader processava a digitalização de forma majoritariamente local no dispositivo de borda: o Raspberry Pi capturava a imagem, aplicava conversão para escala de cinza e filtro bilateral e enviava o resultado ao Tesseract com idioma português e configuração padrão, sem flags adicionais de segmentação. O texto era então encaminhado à síntese de voz. O desenho era simples e coerente com as restrições do projeto, hardware de baixo custo com software livre, sem dependência da nuvem.

Em condições favoráveis, o fluxo era utilizável. Em condições reais de uso (sombras, livro curvado, reflexo, captura sem tripé) a transcrição degradava de forma perceptível. Um exemplo recorrente na análise do projeto é o trecho esperado "Jane Moore, por mostrar-me uma outra face do câncer de mama": o motor local devolvia esse trecho fragmentado e ilegível para leitura por voz, sem qualquer indicação ao usuário de que a transcrição era pouco confiável. Para uma pessoa que depende do sistema para acessar esse conteúdo, a consequência não é um número de acurácia baixo, é simplesmente não conseguir ler.

4.1.2 Três linhas de melhoria no pré-processamento (simple, advanced, enhanced)

Buscando recuperar qualidade sem abandonar o Tesseract, o projeto experimentou três pipelines adicionais de pré-processamento, todos avaliados no mesmo conjunto de imagens com os mesmos textos de referência:

1. **Método simple** (preprocess_image_simple): mantém a mesma base do legado escala de cinza e bilateralFilter e aplica a normalização de espaços após o OCR. A acurácia média subiu apenas 0,19 p.p. em relação ao old (57,65% -> 57,84%): ganho positivo, mas marginal frente à variedade natural do conjunto.
2. **Método advanced** (preprocess_image com adaptive_mode=True): detecta métricas de qualidade da imagem e aplica operações mais pesadas quando indicado (upscaling, CLAHE, threshold adaptativo e denoising condicionais). A intuição era evitar degradar páginas já legíveis. Na prática, a acurácia média caiu 5,20 p.p. em relação ao old (53,63% versus 57,65%), piorando em 164 das 222 imagens. As combinações de filtros introduziram artefatos que o Tesseract não conseguia compensar.
3. **Método enhanced** (preprocess_image_enhanced): upscaling orientado a equivalente de resolução (~300 DPI), CLAHE, denoising, nitidez, threshold adaptativo e, para imagens cuja maior dimensão excede 3000 px, divisão em setores com sobreposição, OCR por setor e concatenação dos textos. O resultado foi o mais fraco do comparativo local: acurácia média 34,93% (−22,72 p.p. em relação ao old), com piora em 192 imagens. Dois fatores explicam boa parte disso: (i) cada setor é tratado pelo Tesseract como bloco independente, o que parte palavras na fronteira entre recortes; (ii) upscaling agressivo em páginas já ruidosas pode suavizar bordas de caracteres. No Raspberry Pi 4, o pipeline mais pesado também imporia tempos de ordem de dezenas de segundos por página, ordem de grandeza várias vezes superior à do servidor de desenvolvimento, sem compensação em qualidade.

4.1.3 Conclusão dessa fase

Os dados indicaram que, para esse conjunto de imagens e esse motor OCR, aumentar a complexidade do pré-processamento local não elevou de forma confiável o teto de qualidade em dois dos três novos modos, piorou. Talvez o dado mais revelador não seja o desempenho absoluto do enhanced: é o fato de que o método mais elaborado foi também o mais prejudicial. Para imagens capturadas sem controle de iluminação, filtros adicionais têm probabilidade não desprezível de degradar a entrada para o motor OCR em vez de melhorá-la. Esse resultado motivou a mudança de estratégia: reservar um caminho multimodal remoto para quando o texto local for insuficiente, em vez de insistir em intervenções na imagem.

4.2 Decisão arquitetural: separação cliente-servidor e OCR híbrido

4.2.1 Separação de responsabilidades

A resposta estrutural foi desacoplar o dispositivo de uso do núcleo de OCR:

- o cliente (wxPython no Raspberry Pi) permanece responsável pela câmera, pela interface acessível e pela orquestração do fluxo de digitalização;

- o servidor (Flask na pasta ocr/ do repositório) concentra o pipeline Tesseract (com modos configuráveis), a heurística de qualidade, o *fallback* para o provedor OpenAI quando aplicável, e a persistência em SQLite.

Esse desenho reduz o acoplamento entre ciclos de evolução: melhorias no OCR ou novos provedores podem ser implantadas no servidor sem exigir atualização do firmware ou da aplicação na borda a cada ajuste e melhorias de interface podem prosseguir sem alterar o contrato de OCR, desde que o cliente continue a honrar a API.

4.2.2 OCR híbrido com *fallback* por qualidade

O segundo eixo da decisão foi política híbrida: usar Tesseract como caminho rápido e, quando o resultado textual for julgado de baixa qualidade por um escore heurístico (quality_score, com limiar configurável por variável de ambiente, padrão 0,55), acionar o modelo multimodal (gpt-4.1-mini nos artefatos analisados). O multimodal interpreta a página como imagem global, o que contorna em parte as fragilidades do encadeamento "segmentação clássica + caractere a caractere" em fotografias ruins.

Em modo provider=auto, se o fallback remoto falhar, o sistema ainda devolve o texto do Tesseract com estratégia explícita de contingência, preservando disponibilidade para o usuário. Com isso o sistema nunca retorna erro silencioso. Mesmo quando o fallback falha, o usuário recebe texto com qualidade potencialmente inferior, mas presente. Para uma tecnologia assistiva, isso é a diferença entre uma experiência degradada e uma experiência simplesmente interrompida.

4.2.3 Comparativo sintético entre abordagens

Tabela 4 - Comparativo entre arquitetura anterior e arquitetura atual

Critério	**Abordagem anterior**	**Arquitetura atual**
Distribuição de responsabilidades	Predominantemente local	Cliente (UI e captura) + Servidor (OCR e dados)
Flexibilidade para novos provedores	Baixa	Alta, com política provider=auto
Rastreabilidade de execução	Limitada	Jobs e páginas com metadados (effective_provider, processing_strategy, quality_score)

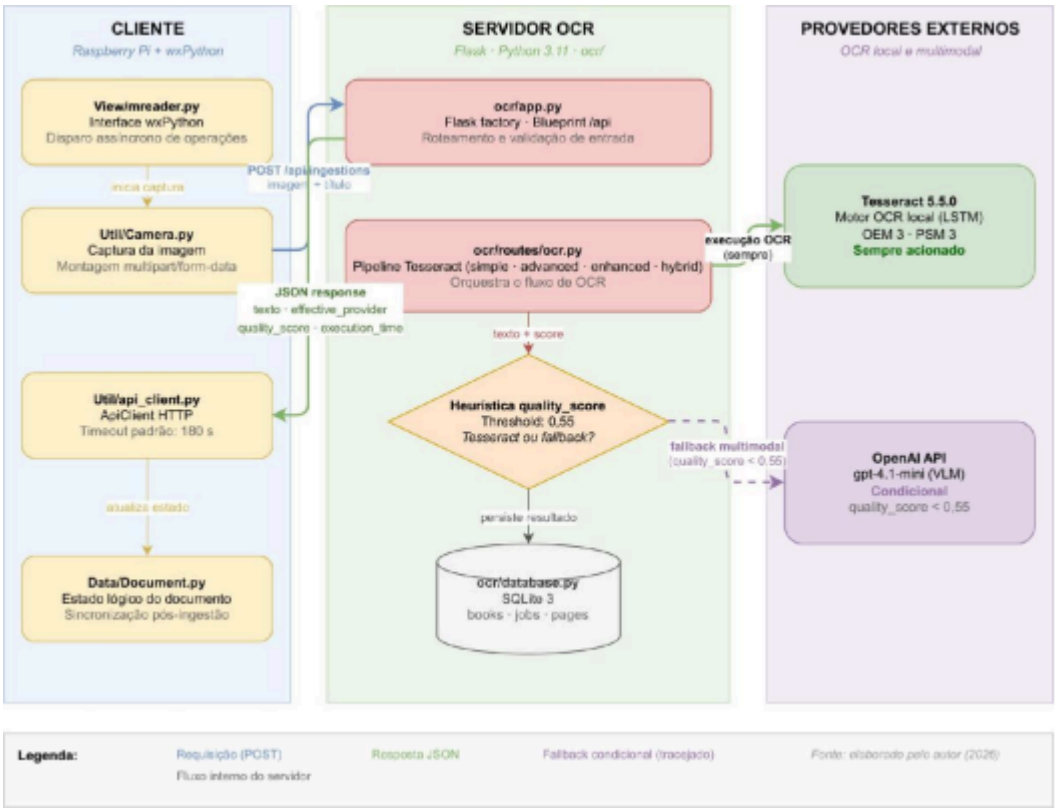
Escalabilidade do pipeline OCR	Restrita ao dispositivo	Modular no servidor
Aderência a benchmark contínuo	Parcial	Forte, com scripts e histórico no repositório

Fonte: elaborado pelo autor (2026), com base no código-fonte do projeto.

4.3 Arquitetura implementada

4.3.1 Visão macro da solução

Em nível macro, o sistema compõe-se de três blocos: cliente desktop, API OCR (Flask) e camada de dados / provedores (Tesseract no servidor e OpenAI via internet).



Fonte: Elaborado pelo Autor (2026)

No fluxo usual de digitalização, o cliente envia a ingestão com provider="auto" e mode="hybrid". O servidor aplica pré-processamento híbrido (preprocess_image_hybrid: base do simple com melhorias seletivas por métricas de qualidade), executa o Tesseract, calcula quality_score e, se o limiar for violado e a chave de API existir, aciona o OpenAI. O resultado é persistido e devolvido ao cliente.

4.3.2 Fluxo ponta a ponta (do clique ao texto final)

O percurso completo pode ser descrito em oito etapas encadeadas; para o usuário, apenas a primeira é explícita.

1. **Interação na interface** — Em View/mreader.py, o usuário informa o título e aciona a digitalização; tarefas de rede e OCR rodam em thread de trabalho para não bloquear a UI (_run_background e rotinas associadas).
2. **Ponte de captura e envio** — Util/Camera.py (imgScan) lê o arquivo capturado e chama ApiClient.process_image() em Util/api_client.py, montando multipart/form-data para POST /api/ingestions (timeout padrão 180 s, configurável por MREADER_API_TIMEOUT_SEC).
3. **Validação e criação de contexto no servidor** — Em ocr/routes/ocr.py, a rota de ingestão valida arquivos (image ou images) e title, cria o job (create_job), associa ou cria o livro (create_or_get_book) e marca execução (mark_job_running).
4. **Pipeline por política** — _process_uploads_with_policy() executa primeiro o fluxo Tesseract com o modo solicitado (no cliente padrão, hybrid). Em seguida _annotate_quality calcula o escore por item e _should_fallback_to_openai decide pelo multimodal.
5. **Estratégias registradas** — Incluem, entre outras, auto_tesseract_accepted, auto_fallback_openai, auto_tesseract_after_openai_failure e auto_tesseract_without_openai (quando não há chave ou o fallback não se aplica).
6. **Persistência** — append_pages() grava texto, provedor efetivo, modo, estratégia e quality_score; mark_job_completed() finaliza tempo e status.
7. **Resposta HTTP** — Retorno com book_id, job_id, effective_provider, processing_strategy, execution_time_sec, páginas e texto consolidado.
8. **Sincronização no cliente** — Data/Document.py aplica o resultado da ingestão e pode sincronizar detalhes com GET /api/books/{id}.

4.3.3 Contratos REST implementados

O serviço expõe, entre outros, os seguintes endpoints:

- POST /api/ingestions e POST /api/ocr — ingestão e processamento OCR;
- GET /api/books, GET /api/books/{book_id} — acervo e detalhe com páginas;
- GET /api/books/{book_id}/text, GET /api/books/{book_id}/pdf — exportação textual e PDF;
- GET /api/books/{book_id}/records, GET /api/jobs/{job_id} — auditoria de execuções;
- DELETE /api/books/{book_id} — remoção de registro;
- GET /api/validate/{title}, GET /api/validate/all — comparação com textos espelho em ocr/texts para validação metodológica.

4.3.4 Especificação funcional da ingestão OCR

Tabela 5 - Parâmetros de entrada relevantes da ingestão

Parâmetro	**Tipo**	**Obrigatório**	**Finalidade**
title	texto	Sim	Identificador lógico do livro/documento
image	arquivo	Sim*	Arquivo único para OCR
images	lista de arquivos	Sim*	Lote de arquivos para OCR
provider	texto	Não	auto, tesseract ou openai
mode	texto	Não	simple, advanced, enhanced, hybrid
dispatch_mode	texto	Não	parallel ou batch (OpenAI)
openai_model	texto	Não	Modelo multimodal
openai_max_concurrency	inteiro	Não	Limite de concorrência (paralelo)
openai_batch_size	inteiro	Não	Tamanho do lote (batch)

Pelo menos um dos campos de arquivo deve estar presente.

Tabela 6 - Campos estratégicos da resposta de ingestão

Campo	**Significado técnico**
book_id	Identificador persistente do documento no backend
job_id	Identificador da execução OCR associada
requested_provider	Provedor solicitado pelo cliente
effective_provider	Provedor efetivamente utilizado após política
processing_strategy	Estratégia final aplicada
execution_time_sec	Tempo total da requisição no *backend*
mode	Modo de pré-processamento utilizado
pages / items	Resultado por página, com texto e metadados
records_url	URL para auditoria completa da execução

4.3.5 Camada de persistência (SQLite)

Implementada em ocr/database.py, com três entidades centrais: books (título e nome normalizado para deduplicação), jobs (status, provedores solicitado/efetivo, estratégia, tempo, erros) e pages (texto por página, quality_score, vínculo ao job). Esse modelo permite reconstruir o histórico de digitalizações e calibrar limiares de fallback em ciclos futuros. A escolha pelo SQLite embarcado, em vez de um banco externo, foi deliberada: mantém o servidor autocontido e implantável sem infraestrutura adicional. Para um projeto assistivo que pode operar em rede local sem acesso à internet, essa simplicidade tem valor prático direto.

4.4 Implementação por componentes

A arquitetura distribui responsabilidades entre quatro módulos principais no cliente e quatro no servidor. No lado do cliente, View/mreader.py é responsável pela interface wxPython e pelo disparo assíncrono de operações; Util/api_client.py encapsula a comunicação HTTP, incluindo tratamento de timeout e erros; Util/Camera.py integra a câmera ao fluxo de ingestão; e Data/Document.py gerencia o estado lógico do documento e a sincronização com o *backend* após cada ingestão.A interface organiza-se em digitalização (inserção de páginas) e consulta (busca por título, texto consolidado, PDF, exclusão). O cliente adota tratamento padronizado de erros de rede, mensagens de status e preservação de seleção na lista após atualização. Os metadados de resposta (effective_provider, execution_time_sec, processing_strategy) permitem, se desejado na evolução da UI, informar ao usuário quando o fallback foi acionado, tornando o sistema mais transparente ao método que produziu o texto recebido

No servidor, ocr/app.py contém a factory Flask e registra o blueprint sob /api; ocr/routes/ocr.py orquestra o pipeline completo, incluindo a política de fallback e os endpoints REST; ocr/database.py centraliza a persistência e o histórico operacional; e providers/openai_ocr.py encapsula o OCR multimodal com suporte a modos de despacho paralelo e em lote, parâmetros configuráveis via variáveis de ambiente e padronização de saída. A rota GET /api/books/{book_id}/pdf gera PDF a partir das páginas persistidas, com seleção de fonte por sistema operacional. O script ocr/benchmark.py completa o conjunto, institucionalizando a comparação por evidência e gerando os artefatos JSON/CSV com histórico de execuções.

4.4.1 Camada cliente

Tabela 7 - Matriz de responsabilidades por componente

Componente	**Responsabilidade principal**	**Responsabilidades secundárias**
View/mreader.py	Interação com usuário	Disparo assíncrono e atualização de estado visual
Util/api_client.py	Comunicação HTTP com *backend*	Timeout e tratamento de erro
Data/Document.py	Estado lógico do documento no cliente	Sincronização com *backend*

Util/Camera.py	Envio à ingestão OCR	Integração com Document
ocr/routes/ocr.py	Orquestração do pipeline de OCR	Política de *fallback* e endpoints
ocr/database.py	Persistência	Histórico operacional
providers/openai_ocr.py	OCR multimodal	Lote, concorrência, padronização de saída
ocr/benchmark.py	Avaliação comparativa	JSON, CSV e histórico de execuções

4.5 Melhorias de interface e operação

A interface organiza-se em **digitalização** (inserção de páginas) e consulta (busca por título, texto consolidado, PDF, exclusão). O cliente adota tratamento padronizado de erros de rede, mensagens de status e preservação de seleção na lista após atualização. Os metadados de resposta (effective_provider, execution_time_sec, processing_strategy) permitem, se desejado na evolução da UI, informar ao usuário quando o fallback foi acionado, tornando o sistema mais transparente ao método que produziu o texto recebido

4.6 Estratégia de benchmark e interpretação no desenvolvimento

4.6.1 Script e artefatos

O benchmark está em ocr/benchmark.py e compara cinco estágios: old_method, new_method_simple, new_method_advanced, new_method_enhanced e openai_gpt. Gera JSON com carimbo de tempo, CSV resumido, cópias “latest” e pasta benchmark_results/named_results/ com transcrições nomeadas por imagem e método, prática que sustenta auditoria qualitativa e reprodutibilidade.

4.6.2 Consolidação analisada neste trabalho

Os números do Capítulo 5 derivam de duas execuções compatíveis: um ciclo com os quatro métodos locais nas 222 imagens e um ciclo OpenAI-only nas mesmas imagens, com subconjunto válido de 206 para análise pareada.

Tabela 8 - Desempenho médio por método no benchmark

Método	Acurácia média (%)	Tempo médio (s)	Imagens
old_method	57,65	3,708	222
new_method_simple	57,84	3,779	222
new_method_advanced	53,63	4,326	222
new_method_enhanced	34,93	8,122	222
openai_gpt	63,64	7,704	206

Fonte: elaborado pelo autor (2026), com base em benchmark_results_20260307_005504.json e benchmark_results_20260410_235329.json.

4.6.3 Leitura para o desenvolvimento do produto

O dado central do ponto de vista de engenharia é a progressão simple → advanced → enhanced: não houve “escada” monotônica de melhoria local; o modo mais rico em operações de imagem foi o de pior acurácia média e maior tempo no servidor de benchmark. Isso sustenta a opção por política adaptativa no servidor (Tesseract com modo hybrid no cliente + auto) em vez de forçar sempre o pipeline mais pesado.

O estágio openai, quando aplicado a 206 imagens com retorno válido mostra acurácia média superior à do old no mesmo subconjunto. O trade-off temporal e o custo por requisição são discutidos no Capítulo 5. Entre as médias globais (63,64% em 206 imagens versus 57,65% do old em 222), a diferença bruta mistura tamanhos de amostra; a inferência principal do trabalho usa o comparativo pareado em 206 imagens (+5,53 p.p. na acurácia média combinada), conforme Capítulo 5. Para o tempo, a diferença média entre openai e old no benchmark foi de +3,996 s por imagem nas condições daquele experimento, interpretável como custo quando o multimodal é efetivamente executado, não como latência média do modo auto em produção.

4.7 Decisões de engenharia, limitações da implementação e síntese

4.7.1 Decisões centrais

1. **Desacoplamento cliente-servidor** — facilita evolução do OCR e concentra histórico em SQLite.
2. ***Fallback* por quality_score** — evita usar sempre o multimodal (latência/custo) nem sempre só o Tesseract (qualidade em páginas difíceis).
3. **Tolerância a falha do provedor externo** — em auto, falha OpenAI ainda devolve Tesseract com estratégia explícita.
4. **Benchmark no repositório** — institucionaliza comparação por evidência.

4.7.2 Limitações técnicas observadas no código

1. Heurística de qualidade ainda **textual/superficial**, sem modelo de confiança treinado no domínio.
2. Dependência de serviços e binários externos (Tesseract, Poppler para PDF, chave OpenAI).
3. Servidor Flask adequado ao ciclo de desenvolvimento; implantação em produção exigiria servidor WSGI/ASGI e demais práticas operacionais.
4. Configuração de URL base no cliente: ApiClient honra variáveis de ambiente, mas há pontos legados com URL fixa que merecem harmonização.

4.7.3 Síntese

A trajetória deste ciclo passou por tentativas frustradas de melhoria local, que os dados do benchmark deixaram claro não serem suficientes, até chegar na separação cliente-servidor com política de fallback, decisão motivada pelos números. O Capítulo 5 apresenta os resultados que essa arquitetura tornou possível medir.

CAPÍTULO 5 - RESULTADOS E DISCUSSÃO

Este capítulo apresenta os resultados obtidos no ciclo de benchmark do M-Reader, organiza a análise por dimensão (acurácia, tempo, qualidade de palavras), inclui comparação com a literatura e discute limitações. Os dados foram consolidados a partir dos artefatos benchmark_results.json e benchmark_summary.csv conforme descrito no módulo 3.

5.1 Visão geral dos dados analisados

O conjunto de teste compreendeu 222 imagens, processadas integralmente pelos quatro métodos locais. O provedor multimodal retornou saída em 206 delas; as 16 restantes não geraram resultado e são investigadas na seção 5.8.6. Por isso, parte das análises é global (222 para métodos locais, 206 para OpenAI) e parte é pareada (206 imagens com resultado em ambos os métodos old e openai), conforme estabelecido no Capítulo 3.

- Parâmetros de execução do benchmark:
- Total de imagens no conjunto de teste: 222
- Imagens processadas por métodos locais: 222 (benchmark 20260307_005504)
- Imagens com saída OpenAI válida: 206 (benchmark 20260410_235329)

- Modo PSM do Tesseract: 3 (segmentação automática)
- Modo de despacho OpenAI: batch (modelo gpt-4.1-mini)

5.2 Acurácia média por método

5.2.1 Resultados globais

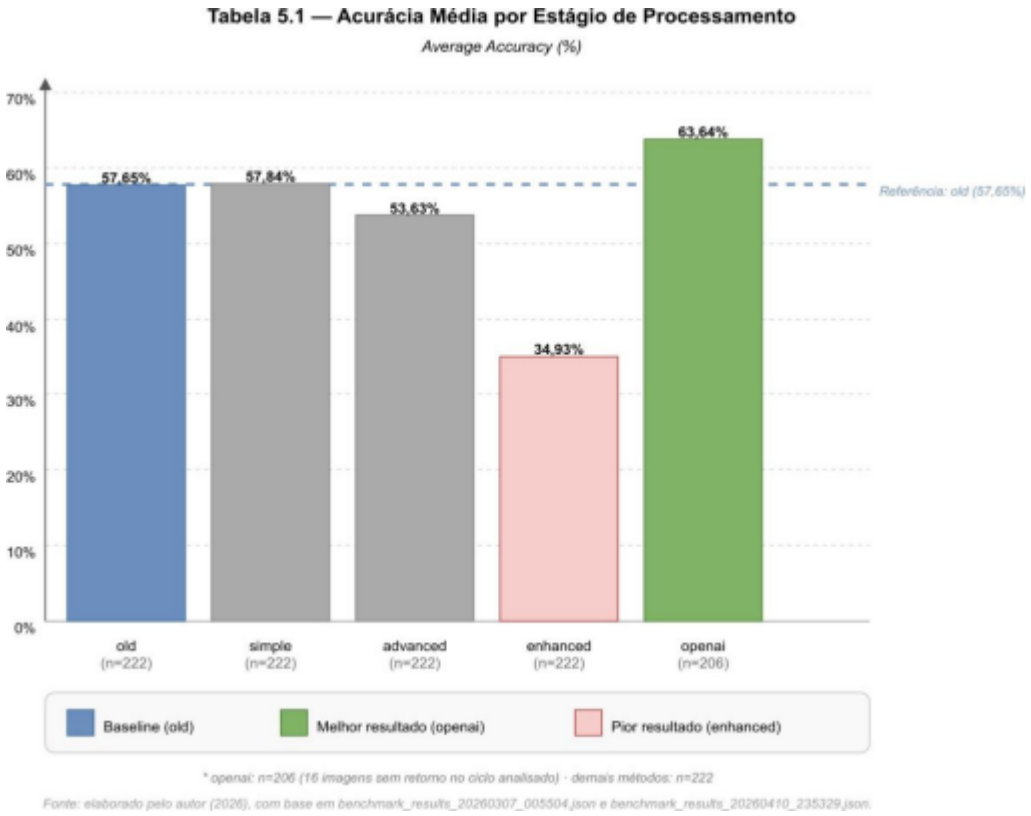
Tabela 9 - Acurácia média por estágio de processamento

Método	**Acurácia média (%)**	**Imagens**
old	57,65	222
simple	57,84	222
advanced	53,63	222
enhanced	34,93	222
openai	63,64	206

Fonte: elaborado pelo autor (2026), com base em benchmark_results_20260307_005504.json e benchmark_results_20260410_235329.json.

Os resultados mostram que a etapa openai apresentou a maior acurácia média no conjunto disponível, seguida por simple e old. O método enhanced teve desempenho inferior no ciclo analisado. A diferença entre o número de imagens por método decorre de 16 casos sem retorno do provedor multimodal, investigados na seção 5.8.6.

É importante observar que os valores absolutos de average accuracy são dominados pela métrica Exact Match, que tende a zero na maioria dos casos reais. Por isso, a interpretação deve priorizar as diferenças entre métodos e as métricas por palavra, mais do que os valores absolutos isolados.



Fonte: elaborado pelo autor (2026).

5.2.2 Dispersão de acurácia por método

Para avaliar a variabilidade dos resultados, além das médias, foram extraídos desvio padrão, valores mínimos e máximos por método.

Tabela 10 - Dispersão de acurácia por método

Método	**Desvio padrão (p.p.)**	**Mínimo (%)**	**Máximo (%)**	**Amplitude (p.p.)**	**n**
old	10,77	0,00	66,10	66,10	222
simple	10,77	0,00	66,10	66,10	222
advanced	10,88	0,00	64,11	64,11	222

enhanced	3,31	0,00	44,03	44,03	222
openai	7,10	0,00	100,00	100,00	206

Fonte: elaborado pelo autor (2026), com base nos artefatos de benchmark consolidados.

A dispersão revela um padrão relevante: o método openai apresenta **menor desvio padrão** (7,10 p.p.) em comparação com o baseline old (10,77 p.p.), indicando maior estabilidade de desempenho entre imagens. Os valores mínimos de 0% em todos os métodos correspondem a casos extremos (imagens muito degradadas ou falhas pontuais de processamento). O valor máximo de 100% no método openai corresponde a casos de correspondência exata com o texto de referência, algo que nenhum método local alcançou no ciclo analisado. A menor variabilidade do OpenAI reforça que o modelo multimodal é mais resiliente a variações de qualidade de captura.

5.2.3 Ganho do método OpenAI em comparação ao baseline

No comparativo pareado openai vs old (206 imagens com resultados para ambos), obteve-se:

- **Média pareada old:** 58,11%
- **Média pareada openai:** 63,64%
- **Ganho absoluto:** +5,53 pontos percentuais
- **Ganho relativo:** +9,51%

A diferença entre a média global do old (57,65%, 222 imagens) e a média pareada (58,11%, 206 imagens) indica que as 16 imagens sem retorno OpenAI tinham acurácia old ligeiramente inferior à média, o que é coerente com a análise de dados faltantes apresentada na seção 5.8.6

5.2.4 Teste de significância estatística

O teste de Wilcoxon signed-rank foi aplicado aos 206 pares de acurácia (openai vs old), com hipótese alternativa unilateral (H_1: acurácia OpenAI > acurácia old). Os resultados foram:

- **Estatística de teste W:** 18.273
- **p-valor (unilateral):** $p < 0{,}001$ ($1{,}68 \times 10^{-21}$)
- **Aproximação normal *z*:** 8,887
- **Tamanho de efeito *r* (Rosenthal):** 0,62

O p-valor extremamente baixo permite rejeitar H_0 com alta confiança: a diferença de acurácia entre os métodos é **estatisticamente significativa**. O tamanho de efeito $r = 0{,}62$

classifica-se como **efeito grande** segundo os critérios de Cohen adaptados por FIELD (2018), indicando que a melhoria não é apenas significativa, mas praticamente relevante.

5.3 Acurácia de palavras (*word accuracy*)

Além da acurácia agregada, o benchmark registrou métricas específicas de reconhecimento de palavras.

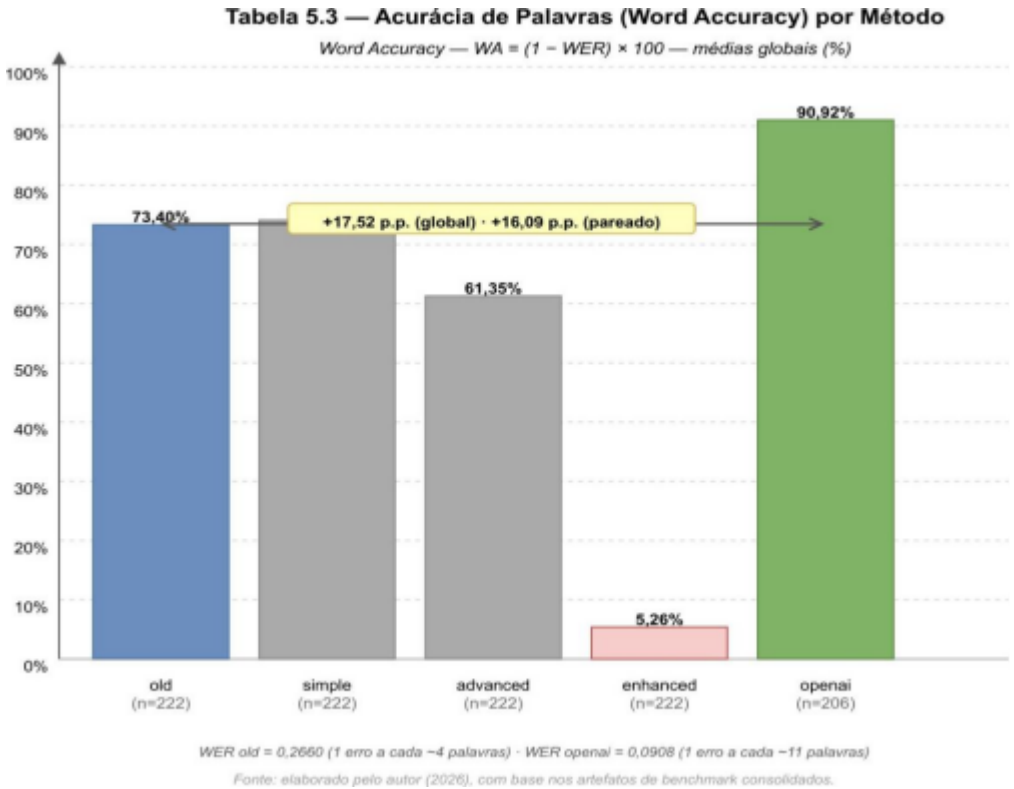
Tabela 11 - Métricas de qualidade de palavras por método (médias globais)

Método	**Word Accuracy (%)**	**Word Error Rate**	**Imagens**
old	73,40	0,2660	222
simple	73,97	0,2603	222
advanced	61,35	0,3865	222
enhanced	5,26	0,9474	222
openai	90,92	0,0908	206

Fonte: elaborado pelo autor (2026), com base nos artefatos de benchmark consolidados.

O método openai apresentou word accuracy de 90,92%, contra 73,40% do baseline old, uma diferença de 17,52 pontos percentuais na visão global. No comparativo pareado (206 imagens), a diferença foi de 16,09 p.p. (old pareado: 74,82%; openai pareado: 90,92%).

Em termos concretos: o Tesseract erra aproximadamente 1 em cada 4 palavras (WER = 0,266), enquanto o OpenAI erra menos de 1 em cada 10 (WER = 0,091). Para um usuário com deficiência visual que recebe o texto por síntese de voz, a diferença entre essas duas taxas é a diferença entre uma frase inteligível e um texto fragmentado que exige esforço constante de interpretação. Um leitor com visão pode corrigir mentalmente erros de OCR; alguém que depende da voz sintetizada não tem essa margem.



Fonte: elaborado pelo autor (2026).

5.4 Análise temporal

5.4.1 Tempo médio por método

Tabela 12 - Tempo médio de execução

Método	**Tempo médio (s)**	**Imagens**
old	3,708	222
simple	3,779	222
advanced	4,326	222
enhanced	8,122	222

openai	7,704	206

5.4.2 Diferença de tempo no comparativo

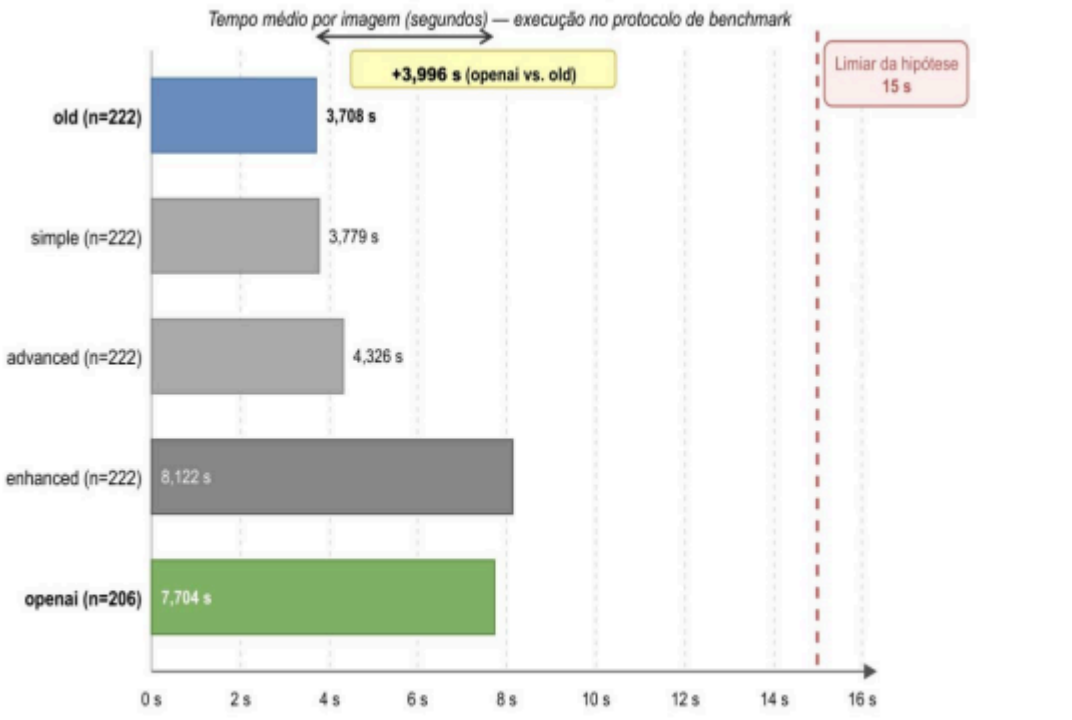
A diferença média de tempo entre os métodos openai e old foi:

- **+3,996 s por imagem** (7,704 s do OpenAI contra 3,708 s do *old*).

Esse valor reflete o custo do estágio multimodal quando executado nas condições do benchmark, servidor em rede local com o cliente, acesso à API via internet. Em produção com fallback seletivo, o multimodal nem sempre executa; o tempo médio real do modo auto depende da frequência com que o quality_score ultrapassa o limiar.

O custo de +3,996 s ficou bem abaixo do limiar de 15 s definido na hipótese. Para contextos assistivos, esse nível de latência é geralmente aceitável: trata-se de segundos a mais de espera em troca de uma transcrição que o usuário conseguirá realmente entender. O enhanced, por comparação, também foi lento (8,122 s) sem qualquer ganho de acurácia.

Tabela 5.4 — Tempo Médio de Execução por Método



* Tempo do openai medido com o método aplicado a todas as 206 imagens com retorno no benchmark — não equivale à latência média do modo auto (fallback seletivo) em produção.

Fonte: elaborado pelo autor (2026), com base nos artefatos de benchmark consolidados.

Fonte: elaborado pelo autor (2026).

5.5 Exemplo qualitativo de transcrição

Para ilustrar o que os números significam na prática, apresenta-se um trecho comparativo da imagem 111 — um dos maiores ganhos individuais do OpenAI sobre o

baseline no ciclo analisado (12,46 p.p.), representativa do perfil de imagens do tercil inferior de dificuldade.

Em trechos simples, ambos os métodos transcrevem corretamente:

Fonte	Texto
Referência (humano)	"Debby Richmond, que adorou tudo desde o começo."
Tesseract (old)	"Debby Richmond, que adorou tudo desde o começo."
OpenAI (gpt-4.1-mini)	"Debby Richmond, que adorou tudo desde o começo."

A diferença aparece em passagens com ruído visual:

Fonte	Texto
Referência (humano)	"Jane Moore, por mostrar-me uma outra face do câncer de mama."
Tesseract (old)	"jane Moorepor mostrar-me'uma outra faãce dorcâncende mama."
OpenAI (gpt-4.1-mini)	"Jane Moore, por mostrar-me uma outra face do câncer de mama."

No trecho do Tesseract há erro de capitalização ("jane"), aglutinação ("Moorepor"), caracteres espúrios ("faãce dorcâncende") e perda de preposição. O OpenAI reconstrói o texto de forma fiel ao original. Esse é o tipo de erro que, em leitura por síntese de voz, não produz apenas ruído auditivo, desfigura o sentido da frase. O nome "Jane Moore" torna-se irreconhecível; a preposição perdida quebra a estrutura sintática.

Este exemplo é ilustrativo do padrão observado nos dados, não uma medida obtida com usuários reais. A confirmação de impacto na experiência assistiva exigiria estudo específico com o público-alvo, lacuna identificada na seção 5.9.

5.6 Análise estratificada por dificuldade

Para verificar se o ganho do método multimodal é uniforme ou concentrado em determinados perfis de imagem, os 206 pares foram divididos em três faixas com base na acurácia do método old.

Tabela 13 - Análise estratificada por tercil de dificuldade

Faixa	**n**	**Média old (%)**	**Média openai (%)**	**Ganho (p.p.)**
Difíceis (tercil inferior)	68	48,00	62,06	+14,06
Medianas (tercil central)	68	61,71	64,41	+2,70
Fáceis (tercil superior)	70	64,44	64,42	−0,02

Fonte: elaborado pelo autor (2026).

O padrão é claro e confirma a premissa da arquitetura de fallback: o ganho do provedor multimodal é fortemente concentrado nas imagens difíceis (+14,06 p.p.), torna-se moderado nas medianas (+2,70 p.p.) e é praticamente inexistente nas fáceis (−0,02 p.p.). Isso tem uma implicação direta de design: o acionamento seletivo do fallback por indicador de qualidade não apenas reduz custos, mas direciona o recurso mais caro para onde ele tem maior impacto — que é exatamente onde o usuário mais precisaria de uma transcrição confiável.

5.7 Verificação formal da hipótese

A hipótese levantada na seção 1.2 estabeleceu dois limiares: (i) ganho de acurácia superior a 3 pontos percentuais em relação ao método tradicional; (ii) latência adicional inferior a 15 segundos por imagem.

Tabela 14 - Confronto entre hipótese e resultados observados

Critério	**Limiar da hipótese**	**Valor observado**	**Verificação**
Ganho de acurácia (pareado)	> 3 p.p.	+5,53 p.p.	Confirmado
Significância estatística	—	$p < 0{,}001$ (Wilcoxon)	Confirmado
Tamanho de efeito	—	$r = 0{,}62$ (grande)	Confirmado
Latência adicional	< 15 s	+3,996 s	Confirmado no protocolo (ambiente da Tabela 3)
Menor dispersão OpenAI	—	$\sigma = 7{,}10$ vs 10,77	Confirmado

Fonte: elaborado pelo autor (2026).

A hipótese é confirmada em todos os critérios formalizados, com base na evidência estatística do benchmark. O ganho de 5,53 p.p. supera o limiar de 3 p.p. com significância alta ($p < 0{,}001$) e efeito grande ($r = 0{,}62$). A latência adicional de 3,996 s situa-se bem abaixo de 15 s nas condições medidas. A menor dispersão do OpenAI alinha-se à expectativa de maior estabilidade do provedor multimodal.

5.8 Discussão técnica dos achados

5.8.1 Sobre a melhoria de acurácia

O ganho observado no método OpenAI confirma a hipótese de que a transcrição multimodal pode recuperar qualidade em entradas onde o OCR tradicional falha por degradação de imagem, ruído ou baixa legibilidade. A análise estratificada (seção 5.6) demonstra que esse ganho não é uniforme: é concentrado nas imagens mais difíceis, onde o OCR tradicional apresenta maior degradação.

Em termos práticos para o M-Reader:

1. O modo auto evita enviar toda imagem ao provedor remoto sem necessidade;
2. O retorno alternativo (*fallback*) é acionado apenas quando a qualidade justifica;
3. A arquitetura preserva o caminho rápido (OCR tradicional) para casos fáceis, onde o ganho do OpenAI é negligível.

5.8.2 Sobre a estabilidade do OpenAI

O método openai apresenta desvio padrão de 7,10 p.p. contra 10,77 p.p. do old, indicando menor variabilidade entre imagens. Isso sugere que o modelo multimodal é mais resiliente a variações de qualidade de captura, característica valiosa para uso assistivo onde as condições de fotografia são imprevisíveis.

5.8.3 Sobre a latência

O aumento de tempo de +3,996 s por imagem é esperado devido ao processamento remoto, serialização da imagem e tempo de resposta do provedor. Esse valor está bem abaixo do limiar de 15 s definido na hipótese e é aceitável na maioria dos cenários assistivos, especialmente se compensado por aumento de legibilidade e menor retrabalho na leitura.

5.8.4 Sobre o impacto arquitetural

A migração para cliente-servidor trouxe dois benefícios estruturais:

- Redução da pressão computacional no dispositivo de borda (Raspberry Pi);
- Maior flexibilidade para evolução do servidor OCR (novos provedores, regras de retorno alternativo e ajustes de qualidade).
- O desacoplamento entre cliente e servidor tem um impacto prático direto: o Raspberry Pi não precisa saber como o processamento é feito, só envia a imagem e aguarda a resposta. Isso significa que qualquer mudança no servidor — trocar um provedor OCR, ajustar o limiar do fallback, corrigir um comportamento — acontece sem que o usuário precise atualizar o dispositivo. Na prática, é o servidor que evolui; o cliente permanece o mesmo.
- A arquitetura também facilita a inclusão de novos provedores OCR. Basta registrar o novo provedor no servidor e ajustar as regras de seleção, sem alterar o cliente. O mesmo vale para tolerância a falhas: se um provedor remoto estiver fora do ar, o sistema pode recuar para o OCR local sem interromper o uso, o que é importante em um contexto assistivo onde a continuidade de acesso importa.
- O fallback seletivo tem ainda um efeito que vai além do ganho de acurácia medido no benchmark. Em produção, o provedor multimodal é acionado apenas quando necessário, o que controla custo por requisição e latência média. O tempo de +3,996 s reportado neste trabalho representa o custo quando o estágio OpenAI é efetivamente ativado, e não a latência média de todas as requisições em modo automático, que tende a ser menor. Esse design garante que o sistema seja rápido na maioria dos casos e preciso quando precisar ser.

5.8.5 Sobre o desempenho do método enhanced

O estágio enhanced apresentou a pior acurácia média (34,93%) e word accuracy de apenas 5,26%. Esse resultado merece atenção específica: o modo enhanced aplica pré-processamento mais agressivo, incluindo segmentação por setores e upscaling, que no conjunto de imagens analisado introduziu artefatos visuais que degradaram a entrada para o Tesseract. A análise do código revela que a função preprocess_image_enhanced aplica redimensionamento com interpolação e filtros morfológicos que, em imagens já de baixa resolução, tendem a borrar bordas de caracteres em vez de realçá-las. O desvio padrão baixo (3,31 p.p.) indica que o desempenho ruim foi consistente, não pontual. Esse achado reforça que pré-processamento mais complexo não é sinônimo de melhor resultado e que a calibração do pipeline precisa ser orientada por dados, não por intuição.

5.8.6 Sobre as 16 imagens sem retorno OpenAI

Das 222 imagens submetidas ao provedor multimodal, 16 não retornaram resultado. Todas apresentaram a mesma causa registrada: "Resposta do OpenAI não retornou esta imagem no lote", indicando falha no processamento em modo *batch*. A análise dos identificadores revela dois padrões:

1. **Imagens com acurácia old muito baixa** (imagens 7, 67, 68, 69, 71, 72, com acurácia old de 33,33%): são imagens de qualidade extremamente baixa que possivelmente desafiaram também o modelo multimodal;
2. **Imagens na faixa intermediária a alta** (imagens 73–79, 190–193, com acurácia old entre 51% e 65%): sugerem falha no protocolo de lote, possivelmente por timeout ou limites de processamento simultâneo.

Do ponto de vista metodológico, a exclusão dessas 16 imagens não introduz viés sistemático, pois incluem tanto imagens difíceis quanto fáceis.

5.8.7 Comparação com a literatura

Os resultados do M-Reader podem ser contextualizados em relação a trabalhos recentes:

- POZNANSKI et al. (2025), com olmOCR, avaliam o processamento de PDFs digitais variados por alinhamento com um modelo de referência e por avaliação humana comparativa, sem reportar acurácia contra transcrições de referência como a usada aqui. Os números, portanto, não são diretamente comparáveis, e o cenário também difere: PDFs nativamente digitais versus fotografias de câmera embarcada em condições não controladas. O que o correlato reforça é a robustez desses modelos diante de documentos variados.
- LI et al. (2021), com TrOCR, atingem taxa de erro de caractere abaixo de 3% em benchmarks controlados, enquanto o M-Reader com OpenAI apresentou erro de palavras de 9,08%. Além de as métricas serem diferentes (erro por caractere versus erro por palavra), o cenário de captura aqui é adverso, o que limita a comparação direta.
- TRIYONO *et* al. (2023) identificam que poucos trabalhos assistivos apresentam métricas quantitativas reais de desempenho em condições operacionais. O presente

trabalho acrescenta números rastreáveis de um sistema em desenvolvimento ativo, com teste de significância estatística sobre o ganho pareado.

A comparação evidencia que os resultados do M-Reader são coerentes com a literatura quando consideradas as restrições de cenário, e que a abordagem híbrida se posiciona como alternativa pragmática entre a precisão de modelos de ponta e as limitações de uso real.

5.9 Limitações dos resultados

Os resultados devem ser interpretados considerando os seguintes fatores:

1. **Cobertura documental:** 222 imagens compõem uma base expressiva para análise comparativa, embora não cubram todos os tipos documentais possíveis (manuscritos, formulários, documentos com tabelas).
2. **Amostra incompleta do método OpenAI:** 16 imagens sem retorno, reduzindo a base pareada para 206. A análise das causas (seção 5.8.6) indica falha no protocolo de lote, sem viés sistemático identificado.
3. **Consolidação de execuções:** os dados combinam dois benchmarks executados em datas distintas, embora sobre a mesma base de imagens e com mesma lógica de métricas.
4. **Dependência de infraestrutura externa:** latência e disponibilidade podem variar por rede e provedor. O custo financeiro por requisição à API OpenAI não foi contabilizado neste ciclo.
5. **Textos de referência:** produzidos por anotador único, sem protocolo de dupla anotação com cálculo de concordância entre anotadores.
6. **Execução única do benchmark:** os resultados de cada método correspondem a uma única execução por imagem, sem repetições para estimar variabilidade intra-método.
7. **Ausência de avaliação com usuários finais:** a análise é exclusivamente quantitativa, sem dados de percepção, usabilidade ou satisfação do público-alvo.
8. **Tempo do openai no benchmark *versus* produção:** o tempo médio do estágio openai foi medido com o método aplicado a todas as imagens com retorno no benchmark, não com a política seletiva de *fallback* do modo auto; latência em campo pode ainda divergir da observada com servidor em rede local (Tabela 3) e internet para a API.

5.10 Síntese do capítulo

Os dados mostram que a estratégia híbrida adotada no M-Reader melhora a qualidade média da transcrição quando comparada ao baseline tradicional em amostra pareada de 206 imagens: +5,53 p.p. em acurácia e +16,09 p.p. em *word accuracy*, com significância estatística confirmada pelo teste de Wilcoxon ($p < 0{,}001$; $r = 0{,}62$). O tempo médio do estágio openai no protocolo foi 3,996 s acima do *old* por imagem (condições da seção 3.5.1). A análise estratificada revelou ganho concentrado nas imagens mais difíceis (+14,06 p.p.), coerente com a ideia de *fallback* seletivo. A dispersão foi menor no OpenAI ($\sigma = 7{,}10$ p.p.) do que no Tesseract ($\sigma = 10{,}77$ p.p.). O exemplo qualitativo ilustra diferença de legibilidade na síntese de voz, **sem** substituir ensaio com usuários (limitação 7). Nos

critérios formalizados na hipótese — **nas condições medidas** — os resultados confirmam ganho de acurácia, latência abaixo de 15 s adicionais e maior estabilidade do multimodal. Esses achados sustentam manter fluxo adaptativo: OCR tradicional nos casos simples e retorno alternativo nos casos críticos, com ressalvas da seção 5.9.

CAPÍTULO 6 - CONSIDERAÇÕES FINAIS

6.1 Síntese dos resultados

Os resultados mostram que a mudança de arquitetura valeu a pena, pelo menos nos cenários mais difíceis. Quando o método multimodal entra em ação, a acurácia sobe de forma relevante e estatisticamente confirmada, e o tempo de resposta se mantém dentro do critério definido na hipótese. No fim deste ciclo, o M-Reader ficou assim: o OCR tradicional atende os casos simples e o fallback multimodal cobre os críticos.

No benchmark consolidado (222 imagens para métodos locais, 206 com retorno OpenAI):

- A etapa openai atingiu **63,64%** de acurácia média e **90,92%** de word accuracy;
- O método old atingiu **57,65%** de acurácia média e **73,40%** de word accuracy;
- No comparativo pareado (openai vs old, 206 imagens), o ganho foi de **5,53 p.p.** em acurácia e **16,09 p.p.** em word accuracy, com significância estatística confirmada pelo teste de Wilcoxon ($p < 0{,}001$; $r = 0{,}62$);
- O tempo médio do estágio openai excedeu o do old em **+3,996 s** por imagem no benchmark (método openai aplicado a todas as 206 imagens com retorno; condições da Tabela 3; não equivale ao tempo médio do modo auto em produção);
- O método openai apresentou desvio padrão de **7,10 p.p.**, contra **10,77 p.p.** do baseline, indicando maior estabilidade;
- A análise estratificada revelou que o ganho é concentrado nas imagens difíceis (**+14,06 p.p.**), tornando-se negligível nas fáceis.

6.2 Atendimento aos objetivos específicos

Cada objetivo específico definido na introdução é retomado a seguir, com a indicação de onde foi atendido no texto.

Objetivo 1 - Descrever a transição arquitetural. Atendido no Capítulo 4: a seção 4.1 situa o ponto de partida local e as tentativas de pré-processamento; as seções 4.2 e 4.3 documentam a decisão de separação e a arquitetura cliente wxPython / servidor Flask. O escopo é descritivo-documental; os ganhos quantitativos referem-se aos estágios de OCR no sistema atual, habilitados pela nova arquitetura.

Objetivo 2 - Caracterizar o fluxo operacional atual. Atendido nas seções 4.2.2, 4.3.2, 4.3.4 e 4.4, que detalham a política híbrida, o fluxo ponta a ponta, os parâmetros de ingestão e a orquestração do pipeline no servidor.

Objetivo 3 - Consolidar os resultados de benchmark. Atendido no Capítulo 5 (seções 5.1 a 5.3), com tabelas de acurácia média, dispersão, word accuracy e tempo por método.

Objetivo 4 - Quantificar a diferença de acurácia. Atendido nas seções 5.2.3 e 5.2.4: ganho pareado de +5,53 p.p. (9,51% relativo) em acurácia e +16,09 p.p. em word accuracy, com significância estatística confirmada ($p < 0,001$).

Além dos objetivos específicos, o objetivo geral previa mensurar também o impacto sobre o tempo de processamento. Essa análise foi realizada na seção 5.4: diferença média de +3,996 s entre *openai* e old quando o estágio executa no protocolo, interpretada como ordem de grandeza do custo quando o multimodal é acionado; a medição direta do fallback seletivo em produção permanece como continuidade. As limitações técnicas e metodológicas da avaliação, por sua vez, estão registradas nas seções 5.9 e 6.4, incluindo a cobertura documental, os dados parciais e a necessidade de validação com usuários.

6.3 Contribuições do trabalho

6.3.1 Contribuições técnicas

1. **Rearquitetura funcional** do M-Reader com desacoplamento entre cliente e processamento OCR, seguindo princípios de engenharia de software (SOMMERVILLE, 2019).
2. **Pipeline adaptativo** com escolha dinâmica de provedor de transcrição via API REST (FIELDING, 2000).
3. **Mecanismo de retorno alternativo (*fallback*) por qualidade**, reduzindo dependência de uma única técnica OCR.
4. **Rastreabilidade por benchmark**, com artefatos em JSON/CSV para auditoria e reanálise.

6.3.2 Impacto para pessoas com deficiência visual

Os números do benchmark descrevem um sistema, mas o motivo de ele existir é acessibilidade: permitir que uma pessoa com deficiência visual leia por conta própria um documento impresso que não está disponível em formato digital.

Nesse contexto, a melhoria de word accuracy de 73,40% para 90,92% muda a experiência de quem recebe o texto por síntese de voz. Uma transcrição como "jane Moorepor mostrar-me'uma outra faãce dorcâncende mama", produzida pelo Tesseract em uma das imagens do conjunto, não serve para leitura por voz; a versão do provedor multimodal recupera o sentido da frase.

O ganho mais expressivo apareceu nas imagens difíceis (+14,06 p.p. no tercil inferior), capturas com livro curvado, iluminação irregular ou foco imperfeito. São os casos em que uma pessoa com deficiência visual, fotografando um documento no dia a dia, tem mais chance de receber um resultado inútil, então é nesses cenários que o fallback multimodal mais ajuda.

A arquitetura também importa do ponto de vista da autonomia. Ao desacoplar o processamento do dispositivo de captura, o sistema permite que o Raspberry Pi permaneça um cliente leve, acessível financeiramente e simples de operar sem exigir hardware sofisticado de quem precisa do dispositivo. As melhorias no servidor ocorrem de forma transparente ao usuário; a interface que a pessoa utiliza não precisa ser atualizada a cada ciclo de evolução do OCR.

Há ainda a questão da escala. O Censo 2022 indica 7,9 milhões de brasileiros com dificuldade funcional para enxergar (IBGE, 2025), e a OMS lembra que uma solução assistiva precisa chegar a quem precisa dela e ser sustentável em termos de custo (WHO, 2019). Um sistema construído sobre hardware de borda barato e software de código aberto, acionando modelos de IA só quando necessário, foi desenhado com esses dois critérios em mente. O que ainda falta para transformar essa viabilidade técnica em contribuição social é a validação com usuários, que os próximos ciclos devem priorizar

6.3.3 Contribuições acadêmicas

1. Relato de caso real de migração de arquitetura em projeto assistivo, com dados quantitativos de desempenho do pipeline por estágio de OCR;
2. Discussão objetiva do compromisso entre acurácia e latência em OCR híbrido, incluindo dispersão e análise estratificada por dificuldade;
3. Evidência empírica contextualizada em relação à literatura (POZNANSKI *et* al., 2025; LI *et* al., 2021; TRIYONO *et* al., 2023), com teste de significância estatística sobre o ganho pareado.

6.3.4 Contribuições para o curso de Análise e Desenvolvimento de Sistemas

O trabalho exercita competências centrais do perfil de ADS: projeto de arquitetura cliente-servidor, desenvolvimento de API REST, modelagem de banco de dados relacional (SQLite com entidades books, jobs, pages), integração com serviços externos, tratamento de concorrência e assincronicidade na interface, e análise quantitativa de desempenho. O percurso também mostra que dá para aplicar essas práticas de engenharia de software a um problema real de acessibilidade, com impacto social direto.

6.4 Limitações

As principais limitações identificadas foram:

1. **Cobertura documental limitada** a páginas de livros impressos em português (222 imagens, 206 com retorno OpenAI), sem inclusão de manuscritos, formulários ou documentos com tabelas.
2. **Resultados incompletos no provedor remoto** em 16 das 222 imagens, embora sem viés sistemático identificado.
3. **Dependência de rede e serviço externo**, com impacto direto em tempo, custo financeiro e previsibilidade.
4. **Consolidação de duas execuções de benchmark**, embora sobre a mesma base de dados e mesma lógica de métricas.

5. **Textos de referência** produzidos por anotador único, sem protocolo de dupla anotação.
6. **Necessidade de validação longitudinal com usuários finais** para mensurar impacto em rotina real de uso assistivo.
7. **Generalização da latência:** tempos medidos com servidor em rede local com o cliente; cenários de internet ou servidor remoto podem alterar o custo real do *fallback*.

6.5 Recomendações para continuidade

Para a próxima etapa do projeto, recomenda-se:

1. ampliar o benchmark com maior diversidade de documentos e condições de captura, incluindo manuscritos, formulários e documentos com tabelas;
2. registrar métricas de qualidade perceptual e usabilidade com usuários finais em cenário real;
3. implementar política de retorno alternativo (*fallback*) com múltiplos níveis (ex.: OCR local, remoto tradicional, multimodal);
4. incorporar monitoramento de custo financeiro, latência e taxa de acionamento do retorno alternativo (*fallback*) em produção;
5. fortalecer governança de dados com políticas de privacidade e retenção compatíveis com uso assistivo;
6. executar benchmark com repetições múltiplas para estimar variabilidade intra-método e adotar protocolo de dupla anotação nos textos de referência;
7. investigar o impacto do M-Reader sobre processos de ensino e aprendizagem de pessoas com deficiência visual, por meio de estudos de campo em ambiente educacional, com foco no acesso a materiais didáticos impressos.

A última recomendação merece um comentário adicional. O M-Reader nasceu em contexto acadêmico e uma de suas aplicações mais diretas é justamente o acesso a materiais educacionais, como livros didáticos, apostilas e textos de apoio. Este trabalho avaliou a dimensão técnica da solução, mas ainda não mediu o efeito que uma leitura mais precisa e mais rápida pode ter sobre a experiência de estudo de um aluno com deficiência visual. Um estudo futuro nessa direção, conduzido em parceria com instituições de ensino e com profissionais do atendimento educacional especializado, permitiria verificar se os ganhos de acurácia observados no benchmark se traduzem em ganho real de aprendizagem e autonomia no ambiente escolar.

6.6 Conclusão final

Há 7,9 milhões de brasileiros com dificuldade funcional para enxergar, e boa parte do material impresso continua inacessível para quem não conta com alguma tecnologia assistiva. Foi para esse problema que o M-Reader foi criado, e é com ele em mente que os resultados deste trabalho devem ser lidos.

Em termos técnicos, este ciclo mostrou que a estratégia híbrida funciona: o Tesseract resolve com rapidez os casos simples e o modelo multimodal entra como reserva nos difíceis, com o servidor cuidando dessa escolha sem que o Raspberry Pi precise saber dos detalhes. A word accuracy de 90,92% do provedor multimodal, contra 73,40% do baseline, significa na prática cerca de 9 palavras corretas a cada 10 transcritas, contra 7 a cada 10 no método anterior. Para quem ouve o texto por síntese de voz, essa diferença muitas vezes decide se a frase lida faz sentido ou não.

O ganho mais expressivo ocorre nas imagens mais difíceis (+14,06 p.p.), que são as mais parecidas com o uso real, com livro mal iluminado, página torta ou câmera sem estabilização. O sistema passa a falhar menos precisamente onde antes mais falhava.

A separação entre cliente e servidor também trouxe um benefício que não aparece nas tabelas. O Raspberry Pi continua barato e simples de operar, e o servidor pode evoluir sem que o usuário precise trocar de aparelho a cada ciclo de evolução do OCR.

O trabalho tem limitações claras, e a principal é que ainda não há dados de pessoas reais usando o sistema. A análise é quantitativa, realizada sobre transcrições automáticas comparadas a textos de referência. Saber que a word accuracy subiu 16 pontos percentuais é importante, mas não é o mesmo que saber que uma pessoa com deficiência visual conseguiu ler um capítulo de livro com mais facilidade. Essa verificação com usuários é a próxima etapa e provavelmente, a mais importante.

O M-Reader segue em construção. Este TCC documenta um ciclo de melhoria, mede o que foi possível medir e aponta o caminho para o que ainda precisa ser feito. A contribuição técnica está consolidada nos dados apresentados. A social depende desses dados virarem, nos próximos ciclos, uma ferramenta mais confiável na mão de quem precisa dela..

REFERÊNCIAS

ABNT. NBR 17060: Acessibilidade — Tecnologia da informação — Diretrizes de acessibilidade para produtos e serviços em tecnologia da informação. Rio de Janeiro: ABNT, 2022.

BLECHER, L.; CAJAL, S.; CUESTA-LAZO, C.; *et* al. Nougat: Neural Optical Understanding for Academic Documents. *arXiv preprint*, arXiv:2308.13418, 2023. Disponível em: https://arxiv.org/abs/2308.13418. Acesso em: 8 abr. 2026.

BRASIL. Lei n. 13.146, de 6 de julho de 2015. Institui a Lei Brasileira de Inclusão da Pessoa com Deficiência (Estatuto da Pessoa com Deficiência). Brasília, DF: Presidência da República, 2015. Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13146.htm. Acesso em: 8 abr. 2026.

BUDRIONIS, A.; PLIKYNAS, D.; DANIUŠIS, P.; INDRULIONIS, A. Smartphone-based computer vision travelling aids for blind and visually impaired individuals: a systematic review. Assistive Technology, v. 34, n. 2, p. 178-194, 2022. Disponível em: <https://www.tandfonline.com/toc/uaty20/34/2>. Acesso em: 12 abr. 2026.

CONOVER, W. J. *Practical Nonparametric Statistics*. 3. ed. New York: Wiley, 1999.

FIELD, A. *Discovering Statistics Using IBM SPSS Statistics*. 5. ed. London: Sage, 2018.

FIELDING, R. T. *Architectural Styles and the Design of Network-based Software Architectures*. 2000. Tese (Doutorado em Ciência da Computação) — University of California, Irvine, 2000.

FLASK. Flask Documentation. [S. l.], 2026. Disponível em: https://flask.palletsprojects.com/. Acesso em: 8 abr. 2026.

FOWLER, M. *Patterns of Enterprise Application Architecture*. Boston: Addison-Wesley, 2002.

GIL, A. C. *Como Elaborar Projetos de Pesquisa*. 6. ed. São Paulo: Atlas, 2017.

GONZALEZ, R. C.; WOODS, R. E. *Digital Image Processing*. 4. ed. New York: Pearson, 2018.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. Cambridge: MIT Press, 2016.

GRINBERG, M. *Flask Web Development*. 2. ed. Sebastopol: O'Reilly, 2018.

HUANG, Y.; CUI, L.; LV, T.; *et* al. LayoutLMv3: Pre-training for Document AI with Unified Text and Image Masking. *arXiv preprint*, arXiv:2204.08387, 2022. Disponível em: https://arxiv.org/abs/2204.08387. Acesso em: 8 abr. 2026.

IBGE. Censo Demográfico 2022: pessoas com deficiência e pessoas diagnosticadas com transtorno do espectro autista — resultados preliminares da amostra. Rio de Janeiro: IBGE, 2025. Disponível em: <https://biblioteca.ibge.gov.br/index.php/biblioteca-catalogo?view=detalhes&id=298009>. Acesso em: 12 abr. 2026.

ISO. ISO 9241-171: Ergonomics of human-system interaction — Part 171: Guidance on software accessibility. Geneva: International Organization for Standardization, 2008.

ISO. ISO 9241-11: Ergonomics of human-system interaction — Part 11: Usability: definitions and concepts. Geneva: International Organization for Standardization, 2018.

KIM, G.; HONG, T.; YIM, M.; *et* al. OCR-free Document Understanding Transformer. In: EUROPEAN CONFERENCE ON COMPUTER VISION, 17., 2022, Tel Aviv. *Proceedings...* Cham: Springer, 2022.

LEVENSHTEIN, V. I. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, v. 10, n. 8, p. 707-710, 1966.

LI, M.; LV, T.; CUI, L.; *et* al. TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models. *arXiv preprint*, arXiv:2109.10282, 2021. Disponível em: https://arxiv.org/abs/2109.10282. Acesso em: 8 abr. 2026.

NEUDECKER, C.; BAIERER, K.; GERBER, M.; CLAUSNER, C.; ANTONACOPOULOS, A.; PLETSCHACHER, S. A survey of OCR evaluation tools and metrics. In: INTERNATIONAL WORKSHOP ON HISTORICAL DOCUMENT IMAGING AND PROCESSING, 6., 2021. *Proceedings...* New York: ACM, 2021. p. 13-18.

OPENCV. OpenCV Documentation. [S. l.], 2026. Disponível em: https://docs.opencv.org/. Acesso em: 8 abr. 2026.

OPENAI. OpenAI API Platform Documentation. [S. l.], 2026. Disponível em: https://platform.openai.com/docs. Acesso em: 8 abr. 2026.

POZNANSKI, J.; RANGAPUR, A.; BORCHARDT, J.; *et* al. olmOCR: Unlocking Trillions of Tokens in PDFs with Vision Language Models. *arXiv preprint*, arXiv:2502.18443, 2025. Disponível em: https://arxiv.org/abs/2502.18443. Acesso em: 8 abr. 2026.

PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de Software: uma abordagem profissional*. 9. ed. Porto Alegre: AMGH, 2021.

PRODANOV, C. C.; FREITAS, E. C. *Metodologia do Trabalho Científico: métodos e técnicas da pesquisa e do trabalho acadêmico*. 2. ed. Novo Hamburgo: Feevale, 2013. Disponível em: https://www.feevale.br/Comum/midias/8807f05a-14d0-4d5b-b1ad-1538f3aef538/E-book%20Metodologia%20do%20Trabalho%20Cientifico.pdf. Acesso em: 8 abr. 2026.

REZENDE, A. L. A.; JESUS, C. P. S.; NOGUEIRA, A. E. S.; et al. Leitor digital autônomo de baixo custo M-Reader: Tecnologia Assistiva como possibilidade de inclusão sociodigital dos sujeitos com deficiência visual. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA

EDUCAÇÃO, 32., 2021. Anais... Porto Alegre: Sociedade Brasileira de Computação, 2021a. p. 552-563. Disponível em: https://sol.sbc.org.br/index.php/sbie/article/view/18086. Acesso em: 15 jul. 2026.

REZENDE, A. L. A.; JESUS, C. P. S.; NOGUEIRA, A. E. S.; et al. Tecnologia Assistiva MINDER: possibilidades de um computador de baixo custo na inclusão sociodigital de sujeitos não videntes. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 32., 2021. Anais... Porto Alegre: Sociedade Brasileira de Computação, 2021b. p. 586-597. Disponível em: https://sol.sbc.org.br/index.php/sbie/article/view/18089. Acesso em: 15 jul. 2026.

SMITH, R. An overview of the Tesseract OCR engine. In: INTERNATIONAL CONFERENCE ON DOCUMENT ANALYSIS AND RECOGNITION, 9., 2007, Curitiba. *Proceedings...* Curitiba: IEEE, 2007. p. 629-633.

SOMMERVILLE, I. *Engenharia de Software*. 10. ed. São Paulo: Pearson, 2019.

SQLITE. SQLite Documentation. [S. l.], 2026. Disponível em: https://www.sqlite.org/docs.html. Acesso em: 8 abr. 2026.

SUBRAMANI, N.; MATTON, A.; GREAVES, M.; LAM, A. A Survey of Deep Learning Approaches for OCR and Document Understanding. *arXiv preprint*, arXiv:2011.13534, 2021. Disponível em: https://arxiv.org/abs/2011.13534. Acesso em: 8 abr. 2026.

SZELISKI, R. *Computer Vision: Algorithms and Applications*. 2. ed. Cham: Springer, 2022.

TAFTI, A. P.; BAGHAIE, A.; ASSEFI, M.; *et* al. OCR as a Service: An Experimental Evaluation of Google Docs OCR, Tesseract, ABBYY FineReader, and Transym. In: INTERNATIONAL SYMPOSIUM ON VISUAL COMPUTING, 12., 2016, Las Vegas. *Proceedings...* Cham: Springer, 2016. p. 735-746.

TESSERACT OCR. Tesseract OCR Documentation. [S. l.], 2024. Disponível em: https://github.com/tesseract-ocr/tesseract. Acesso em: 8 abr. 2026.

TRIYONO, L.; GERNOWO, R.; PRAYITNO; CHOLIL, S. R. A systematic review of artificial intelligence in assistive technology for people with visual impairment. *Kinetik*, v. 8, n. 4, 2023. DOI: 10.22219/kinetik.v8i4.1772.

W3C. Web Content Accessibility Guidelines (WCAG) 2.1. [S. l.], 2018. Disponível em: https://www.w3.org/TR/WCAG21/. Acesso em: 8 abr. 2026.

WAZLAWICK, R. S. *Metodologia de Pesquisa para Ciência da Computação*. 2. ed. Rio de Janeiro: Elsevier, 2014.

WERKEMA, M. C. C. *Métodos PDCA e DMAIC e suas ferramentas analíticas*. Rio de Janeiro: Elsevier, 2012.

WORLD HEALTH ORGANIZATION. *World report on vision*. Geneva: WHO, 2019.

YAO, F.; ZHOU, W.; HU, H. A review of vision-based assistive systems for visually impaired people: technologies, applications, and future directions. *arXiv preprint*, arXiv:2505.14298, 2025. Disponível em: https://arxiv.org/abs/2505.14298. Acesso em: 8 abr. 2026.

Documento Digitalizado Público

TCC - Versão final com ficha catalográfica

Assunto: TCC - Versão final com ficha catalográfica
Assinado por: Andre Rezende
Tipo do Documento: ANEXO
Situação: Finalizado
Nível de Acesso: Público
Tipo do Conferência: Cópia Simples

Documento assinado eletronicamente por:

- **Andre Luiz Andrade Rezende**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 24/07/2026 11:25:13.

Este documento foi armazenado no SUAP em 24/07/2026. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifbaiano.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 1367979
Código de Autenticação: 992e1e77bc