Benchmark de Ferramentas de OCR para Digitalização de Registros Imobiliários: Uma Análise Software-Hardware para Definição de Arquitetura

Elton Sarmanho[1], Carlos Portela[1,2], Edinaldo Henriques[2], Gleyciane Freitas[2], Vinicius Almeida[3], Greg Barreto[4]

[1]Faculdade de Sistemas de Informação (FASI)
Universidade Federal do Pará (UFPA) - Cametá - PA

[2]Programa de Pós-Graduação em Computação Aplicada (PPCA)
Universidade Federal do Pará (UFPA) - Tucuruí - PA

[3]Cartório de Registro de Imóveis de Correntina - Correntina - BA

[4]Cartório de Registro de Imóveis GB - Luís Eduardo Magalhães - BA

{eltonss,csp}@ufpa.br,{edinaldohenriques02,
gleycianefreitas04}@gmail.com,{vinialmeida,greg_barreto}@hotmail.com

Abstract. *This paper presents a benchmark of Optical Character Recognition (OCR) tools in the context of digitizing land registry documents. We compare proprietary (AWS Textract, Google Vision, Gemini) and open-source tools (Tesseract, EasyOCR, PaddleOCR) across on-premises CPU/GPU deployments and cloud APIs. The results indicate higher accuracy for proprietary solutions and higher speed for GPU-accelerated open-source options, supporting hybrid architectures. We discuss operational impacts related to energy use and data traffic across the different architectures. The evidence gathered informs architecture and deployment decisions in real-world document processing services.*

Resumo. *Este artigo apresenta um benchmark de ferramentas de Reconhecimento Óptico de Caracteres (OCR) no contexto de digitalização de registros imobiliários. Foram comparadas ferramentas proprietárias (AWS Textract, Google Vision, Gemini) e open source (Tesseract, EasyOCR, PaddleOCR) em implantações CPU/GPU on premise e APIs em nuvem. Os resultados indicam maior acurácia nas soluções proprietárias e maior velocidade nas alternativas open source aceleradas em GPU, apoiando arquiteturas híbridas. Discutiu-se os impactos operacionais relacionados ao uso de energia e tráfego de dados nas diferentes arquiteturas. As evidências obtidas orientam decisões de arquitetura e implantação em cenários reais de serviços documentais.*

1. Introdução

A digitalização automatizada de documentos registrais é crucial para setores jurídico, cartorial e imobiliário no Brasil [Filho et al. 2019]. Esses documentos são heterogêneos e resultam de um histórico normativo desde a Lei de Registros Públicos (Lei nº 6.015) [Brasil 1973], combinando texto datilografado e impresso, assinaturas e elementos georreferenciados, o que impõe desafios ao Reconhecimento Óptico de Caracteres (OCR).

OCR é uma tecnologia para converter imagens escaneadas em texto estruturado, por meio de um *pipeline* que envolve pré-processamento, segmentação, extração, classificação e pós-processamento [Vijayarani and Sakila 2015]. Apesar de avanços, baixa resolução e layouts complexos diminuem a precisão em cenários reais [Lima and Silva 2025]. Destacam-se soluções proprietárias dispostas como APIs e alternativas *open source* com maior flexibilidade [Kanagarathinam and Sekar 2019, Silva et al. 2021, Lima and Silva 2025]. A decisão, porém, não é apenas de software, pois a infraestrutura (CPU vs. GPU) de hardware, latência fim-a-fim e custos operacionais influenciam o desenho do *pipeline* e a escalabilidade.

Este trabalho investiga ferramentas de OCR para digitalização de matrículas de imóveis, comparando opções proprietárias e *open source* em implantações locais (CPU/GPU) e em nuvem, considerando acurácia e eficiência a fim de apoiar decisões de arquitetura e implantação em ambientes reais. Neste contexto, definiu-se a Questão de Pesquisa **(QP)**: *Qual a diferença de desempenho entre ferramentas de OCR proprietárias e open source na extração de informações de documentos de matrícula de imóveis?*

2. Revisão da Literatura

A tecnologia de OCR evoluiu e se disseminou, viabilizando digitalização em larga escala e automação de fluxos de trabalho [Vijayarani and Sakila 2015, Hegghammer 2021, Tafti et al. 2016]. Estudos de *benchmark* iniciais avaliaram ferramentas e apontaram limitações em símbolos/equações [Vijayarani and Sakila 2015, Vijayarani and Sakila 2016]. Avaliações focadas em domínios específicos evidenciaram a necessidade de pré-processamento dedicado mesmo em soluções comerciais [Silva et al. 2021]. Com *datasets* amplos e ruído controlado, comparativos recentes mostraram superioridade de serviços em nuvem sobre Tesseract, além de persistência de lacunas por idioma [Hegghammer 2021]. Resultados com múltiplas ferramentas também destacaram queda de desempenho em imagens de baixa qualidade [Tafti et al. 2016]. Em cenários com texto e equações, o PaddleOCR se destacou em acurácia, embora Tesseract e i2OCR fossem mais rápidos [Francis and Sangeetha 2025].

Paralelamente, abordagens baseadas em detecção MSER e morfologia elevaram a acurácia em domínios restritos [Kanagarathinam and Sekar 2019], enquanto super-resolução com GANs (BSRGAN) aprimorou a legibilidade para OCR, superando filtros clássicos [Lima and Silva 2025]. No nível arquitetural, combinações CNN+LSTM (Calamari) superaram modelos LSTM rasos (OCRopus), e técnicas como *data augmentation* e *confidence voting* reduziram erros com poucos dados [Wick et al. 2018]. Mais recentemente, LLMs multimodais, como Gemini 1.5 Pro, demonstraram ganhos de precisão e tempo em manuscritos frente a OCR tradicional [Gupta et al. 2024].

Apesar do progresso, há lacunas em *benchmarks* modernos que integrem comparação de ferramentas proprietárias e *open source* com diferenciação de *hardware* (CPU/GPU) aplicada a domínios documentais legais complexos, como matrículas imobiliárias brasileiras, que é caracterizada por alta variabilidade de *layout* e qualidade [Hegghammer 2021, Kanagarathinam and Sekar 2019, Gupta et al. 2024]. Este trabalho contribui preenchendo essa lacuna com um *benchmark* sistemático que avalia impacto de aceleração por hardware, discutindo implicações para escolha de ferramentas e desenho de *pipelines* em contextos reais.

3. Metodologia

A abordagem experimental é de natureza quantitativa, focada na medição objetiva de métricas de acurácia e velocidade, e validada por meio de análises e testes estatísticos para garantir a coerência das conclusões [Hegghammer 2021, Tafti et al. 2016]. Foram selecionadas três soluções proprietárias (**AWS Textract** [Amazon Web Services 2025, Francis and Sangeetha 2025], **Cloud Vision API** [Google 2025], **Google Gemini** [Google DeepMind 2023, Gemma 2024]) e três *open source* (**Tesseract** [Smith 2007], **EasyOCR** [JaidedAI 2025, Chaitra et al. 2023], **PaddleOCR** [Cui et al. 2025]), conforme mostra a Tabela 1.

Tabela 1. Características das ferramentas de OCR avaliadas no estudo

Ferramenta	Mantenedor	Arquitetura	Instalação	Custo
AWS Textract	Amazon Web Services	Rede Neural Profunda	Servidor (API)	$1.50 por 1000 páginas
Cloud Vision API	Google Cloud	Rede Neural Profunda	Servidor (API)	$1.50 por 1000 páginas
Google Gemini[a]	Google	Transformer (LLM Multimodal)	Servidor (API)	$7,00 por 1000 páginas
Tesseract	Tesseract OCR Project	LSTM (Long Short-Term Memory)	Local	Gratuito
EasyOCR	Jaided AI	CRNN (CNN + RNN/LSTM + CTC)	Local (Python)	Gratuito
PaddleOCR	Baidu (PaddlePaddle)	Baseada em CRNN (e.g., SVTR)	Local (Python)	Gratuito

[a] Este custo para o Google Gemini é uma estimativa baseada no consumo de tokens para um documento de matrícula de imóvel com texto denso. O preço real da API Gemini é calculado por milhão de tokens (e não por página), com a imagem de entrada custando aproximadamente $3.50 a $7.00 por 1 milhão de tokens para o modelo Gemini 1.5 Pro, dependendo do tamanho do contexto utilizado.

3.1. Configuração Experimental

Ajustes foram definidos após análise exploratória para robustez a *layouts* heterogêneos. A Tabela 2 resume a *grid* e escolhas finais.

Para os modelos locais, o processamento foi executado em ambiente Ubuntu 22.04 LTS, utilizando aceleração por hardware (CUDA 11.8) nas versões especificadas como GPU. As ferramentas proprietárias (AWS Textract, Google Vision e Gemini 1.5 Pro) foram acessadas via SDKs oficiais (Agosto/2025), operando sob arquitetura Serverless em seus endpoints estáveis.

O conjunto de documentos é composto por 31 arquivos únicos de matrículas de imóveis, totalizando 219 páginas em formato PDF, conforme Tabela 3.

Devido à natureza sensível dos dados contidos nas matrículas de imóveis e em

Tabela 2. Grade de Parâmetros e Configurações Experimentais dos Motores Open-Source

Motor (Versão)	Parâmetro	Valor Selecionado	Justificativa do Ajuste
Tesseract	OEM (*Engine*)	3 (LSTM)	Melhor desempenho em textos densos e ruidosos.
(v5.3.0)	PSM (*Segment.*)	3 (Auto)	Testado contra PSM 1 (OSD) e PSM 6 (Bloco único). O PSM 3 foi superior na detecção de layouts mistos.
EasyOCR	Decoder	*Greedy*	Validado contra *Beam Search*. A diferença em acurácia não justificou o custo computacional.
(v1.7.1)	*Beam Width*	5	Equilíbrio entre precisão e latência observado no estudo piloto.
PaddleOCR	Modelo	PP-OCRv3	Selecionado pela eficiência em CPU/GPU em comparação ao modelo v4 para textos em português.
(v2.7.0)	*Det. Thresh.*	0,3	Limiar de detecção de caixas de texto ajustado para capturar caracteres menores e carimbos.

Tabela 3. Parâmetros e escala do experimento de *benchmarking*

Parâmetro	Valor
Nº de Documentos Únicos	31
Nº de Páginas Únicas no *Corpus*	219
Nº de Métodos OCR Testados	8
Nº de Iterações por Combinação	3
Total de Testes Executados	**744**
Total de Páginas Processadas	**5.235**

conformidade com o termo de confidencialidade que rege a pesquisa, Figura 1 apresenta um exemplo do documento oficial com desfoque com a devida autorização do cartório.

Para garantir a validade interna do experimento e a precisão das métricas de avaliação (WER e CER), a construção do texto de referência (*Ground Truth*) seguiu um protocolo rigoroso de curadoria manual. O processo de transcrição abrangeu a totalidade das 219 páginas do corpus e foi executado em duas etapas de verificação humana:

1. **Transcrição Manual:** Duas pessoas realizaram a transcrição integral dos documentos, com a diretriz estrita de preservar fielmente o conteúdo visual. Isso incluiu a manutenção de erros de digitação originais do documento (*sic*), abreviações arcaicas, formatações específicas de moeda e inconsistências de layout presentes nos registros físicos.
2. **Validação Cruzada:** O material transcrito foi submetido à revisão de dois outros pesquisadores distintos (validação por pares). Esta etapa consistiu em uma verificação caractere a caractere para assegurar que o texto digital correspondesse

Figura 1. Páginas de um Documento de Matrícula de Imóvel

exatamente à imagem de entrada utilizada no teste.

Ao garantir um gabarito com fidelidade absoluta ao documento físico, assegura-se que qualquer divergência apontada pelas métricas de distância (Levenshtein) seja estritamente atribuível a falhas de reconhecimento das ferramentas de OCR, e não a inconsistências na base de dados de referência.

3.2. Delineamento, métodos e métricas

O estudo seguiu um delineamento fatorial completo, com 3 iterações independentes para cada combinação de documento e método de OCR. Foram realizadas 3 iterações para mitigar eventuais oscilações de *cold-start* das APIs e variabilidade de escalonamento de processos na CPU local, reportando-se a média aritmética. Com um *corpus* de 31 documentos e 8 implementações de OCR distintas, conforme a Tabela 4, o experimento totalizou 744 execuções independentes (31 documentos $\times$ 8 métodos $\times$ 3 iterações). Considerando o volume de páginas, um total de 5.235 páginas foi processado, garantindo que cada página única do *corpus* fosse submetida a 24 ciclos de processamento, o que confere robustez estatística aos resultados.

Tabela 4. Implementações de OCR avaliadas no estudo.

Ferramenta Base	Implementações Testadas
AWS Textract	API de Serviço Comercial
Cloud Vision API	API de Serviço Comercial
Google Gemini	API de Modelo Multimodal
Tesseract	Sequencial (CPU)
Tesseract	Paralela (CPU)
EasyOCR	CPU
EasyOCR	Acelerada por GPU
PaddleOCR	CPU
PaddleOCR	Acelerada por GPU

3.3. Ambiente experimental

Todos os testes descritos neste estudo foram conduzidos em um único ambiente computacional para garantir a consistência, a comparabilidade e a reprodutibilidade dos resultados. A máquina utilizada para a execução do *benchmark* foi um notebook Dell G15 equipado com um processador Intel® Core™ i7-11800H de 11ª geração (8 núcleos, 16 threads @ 2.30GHz), 16 GB de memória RAM e uma unidade de processamento gráfico (GPU) NVIDIA GeForce RTX 3060 Laptop com 6 GB de VRAM dedicada.

O armazenamento principal consiste em uma unidade de estado sólido (SSD NVMe) de 476 GB, garantindo alta velocidade de leitura e escrita dos dados. O ambiente de software foi baseado no sistema operacional Ubuntu (kernel 6.14.0). As implementações de código aberto foram executadas utilizando a linguagem de programação Python e suas respectivas bibliotecas, enquanto as ferramentas proprietárias foram acessadas via suas APIs oficiais.

4. Resultados e Discussão

Esta seção apresenta os resultados da avaliação comparativa de desempenho das oito implementações de OCR. A análise é dividida em três vertentes principais: acurácia, eficiência e validação estatística. Por fim, uma discussão geral sintetiza os achados e os contextualiza sob a perspectiva de pessoas, processos e tecnologias.

4.1. Análise de Acurácia

A acurácia dos modelos foi a principal dimensão avaliada, utilizando as métricas de Similaridade Levenshtein, Taxa de Erro de Palavras (WER) e Taxa de Erro de Caracteres (CER). Os resultados consolidados, apresentados na Tabela 5, revelam uma hierarquia de desempenho entre as ferramentas, em que os métodos estão dispostos em ordem decrescente de performance geral, permitindo uma visualização direta da classificação do melhor para o pior resultado com base nas métricas mensuradas.

Tabela 5. Resultados consolidados do *benchmark* de OCR

Método OCR	Similaridade Média	WER Médio	CER Médio	Velocidade (Chars/s)
AWS Textract	0.8631	0.2737	0.2624	420.5
Google Gemini	0.8570	0.3223	0.2592	155.8
Cloud Vision API	0.8467	0.3302	0.2826	448.3
Easyocr (GPU)	0.8319	0.4663	0.3084	226.5
Easyocr (CPU)	0.8314	0.4690	0.3089	47.0
Paddleocr (CPU)	0.8199	0.4522	0.3113	720.3
Paddleocr (GPU)	0.8199	0.4525	0.3116	2538.6
Pytesseract (Sequencial)	0.7588	0.9609	0.4131	1298.6
Pytesseract (Paralelizado)	0.7588	0.9609	0.4131	2303.6

As ferramentas proprietárias baseadas em nuvem dominaram as primeiras posições. O **AWS Textract** alcançou o melhor desempenho geral, com a maior similaridade média (0.8631) e o menor WER (0.2737), estabelecendo-se como a solução mais precisa para a tarefa. Logo em seguida, o **Google Gemini** se destacou ao obter a menor

taxa de erro a nível de caractere (CER de 0.2592), demonstrando a força dos modelos de linguagem multimodais na interpretação de caracteres complexos. O **Cloud Vision API** também apresentou resultados robustos, consolidando a superioridade dos modelos comerciais. Este resultado alinha-se com as conclusões de [Hegghammer 2021], que também observou um desempenho superior de ferramentas de OCR baseadas em servidor (Amazon e Google) em comparação com alternativas de código aberto.

Entre as ferramentas de código aberto, o **EasyOCR** (tanto em CPU quanto em GPU) apresentou a maior acurácia, superando o **PaddleOCR** e o **Pytesseract** nas métricas de qualidade. O Pytesseract, em suas duas variantes, obteve os maiores índices de erro, indicando menor adequação para documentos complexos como as matrículas de imóveis. As Figuras 2a, 2b e 2c ilustram visualmente a distribuição desses resultados de acurácia, evidenciando a hierarquia de desempenho observada.

4.2. Análise de Eficiência

A análise de eficiência focou na vazão efetiva (caracteres por segundo), conforme Tabela 5. Os resultados evidenciam as características distintas das arquiteturas avaliadas. O **PaddleOCR** em GPU atingiu a maior velocidade (2538.6 chars/s), beneficiando-se do processamento local acelerado sem o *overhead* de comunicação de rede.

As ferramentas proprietárias (AWS Textract e Cloud Vision API) apresentaram velocidades intermediárias (aprox. 420-450 chars/s). É crucial notar que, para estas ferramentas, a métrica inclui a latência de rede (upload/download). Embora não utilizem os recursos da máquina local, sua velocidade é limitada pela largura de banda e pelo tempo de resposta do servidor remoto. O Google Gemini, sendo um modelo multimodal massivo (LLM), apresentou a menor velocidade entre as proprietárias (155.8 chars/s), um custo esperado dada a complexidade computacional da inferência de modelos Generativos em comparação a OCRs tradicionais.

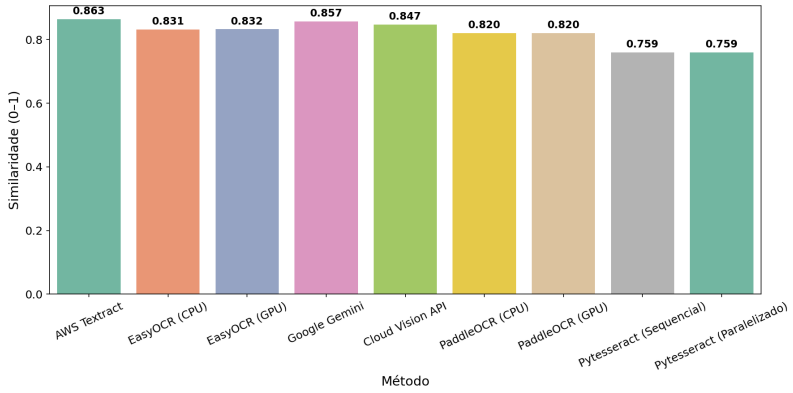
O impacto da aceleração local por GPU foi expressivo: o **PaddleOCR** em GPU foi aproximadamente 3.5 vezes mais rápido que sua versão em CPU, e o **EasyOCR** cerca de 4.8 vezes. Isso confirma que, para cenários onde a latência de rede é um gargalo ou a conectividade é instável, soluções locais com hardware dedicado oferecem uma vantagem de vazão superior, ainda que com o compromisso de menor acurácia observado na seção anterior.

4.3. Análise Estatística

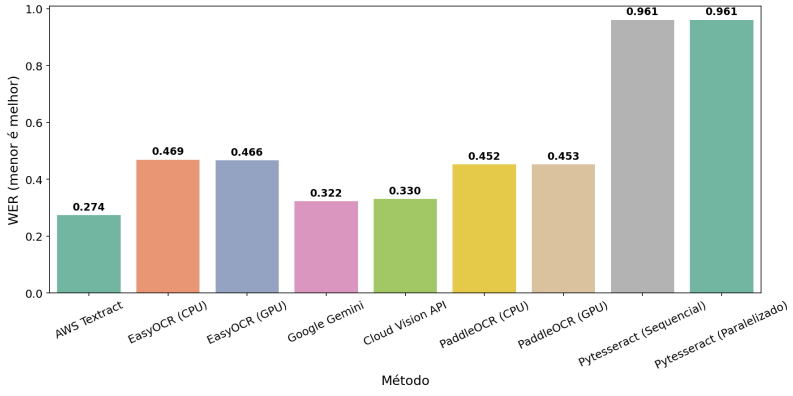
Para validar as diferenças de desempenho observadas, aplicou-se o teste não-paramétrico de Friedman para amostras pareadas, considerando que todos os modelos processaram o mesmo conjunto de documentos válidos. Essa abordagem isola a variabilidade intrínseca à complexidade de cada documento, focando nas diferenças entre as ferramentas.

O teste de Friedman indicou uma diferença estatisticamente significativa entre os modelos ($\chi^2 = 85.0394$, $p < 0.001$). Rejeitada a hipótese nula global, procedeu-se à análise post-hoc de Nemenyi para comparações par-a-par. Os resultados, detalhados na matriz de p-valores (Tabela 6), confirmaram que:

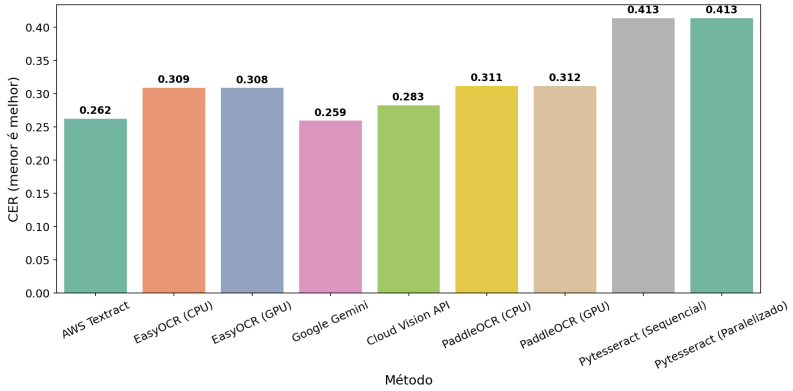
- **Superioridade das Ferramentas Proprietárias:** O **Google Gemini** e o **AWS Textract** formam o grupo de elite, superando estatisticamente todas as soluções *open-source* (EasyOCR, PaddleOCR e Pytesseract) com $p < 0.001$.



(a) Gráfico da Similaridade Levenshtein média por método OCR.



(b) Gráfico do Word Error Rate (WER) médio por método OCR.



(c) Gráfico do Character Error Rate (CER) médio por método OCR.

Figura 2. Conjunto de gráficos com as métricas de acurácia.

- **Empate Técnico no Topo:** Não houve diferença estatística significativa entre o **Google Gemini** e o **AWS Textract** ($p = 0.8998$).
- **Diferença entre Gerações Google:** O **Google Gemini** superou estatisticamente o **Google Vision** ($p = 0.0016$), evidenciando a evolução dos modelos baseados em LLM sobre as APIs de OCR tradicionais da mesma empresa.
- **Equivalência entre Modelos Locais:** Não foram encontradas diferenças estatisticamente significativas ($p > 0.05$) entre **EasyOCR**, **PaddleOCR** e **Pytesseract**, sugerindo que, em termos de acurácia bruta (CER), eles performam de maneira similar neste *corpus*, diferenciando-se apenas pela velocidade de processamento.

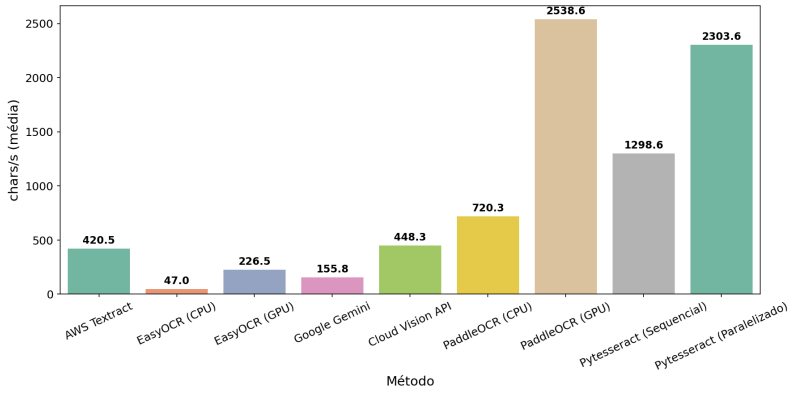


Figura 3. Gráfico da Velocidade média (caracteres/segundo) por método OCR

Tabela 6. Matriz de P-Valores do Teste Post-Hoc de Nemenyi (Diferenças significativas em negrito, $p < 0.05$)

Modelo	AWS	EasyOCR	Gemini	Vision	Paddle	Tess.
AWS Textract	-	$<$ **0.001**	0.900	0.056	$<$ **0.001**	$<$ **0.001**
EasyOCR	$<$ **0.001**	-	$<$ **0.001**	0.127	1.000	1.000
Google Gemini	0.900	$<$ **0.001**	-	**0.002**	$<$ **0.001**	$<$ **0.001**
Google Vision	0.056	0.127	**0.002**	-	0.098	0.090
PaddleOCR	$<$ **0.001**	1.000	$<$ **0.001**	0.098	-	1.000
Pytesseract	$<$ **0.001**	1.000	$<$ **0.001**	0.090	1.000	-

Em suma, os resultados respondem à questão de pesquisa (**QP**): as ferramentas proprietárias, de fato, superaram as de código aberto em acurácia de forma estatisticamente significativa. Enquanto estudos como o de [Silva et al. 2021] mostraram que ferramentas comerciais podem falhar em tarefas mais simples sem pré-processamento, porém o benchmark desta pesquisa demonstra que, para documentos complexos e heterogêneos, os modelos avançados da AWS e Google Gemini oferecem um desempenho robusto. Portanto, a seleção da ferramenta mais adequada depende do aspecto mais relevante dentro do contexto: para **máxima acurácia**, AWS Textract e Google Gemini são as escolhas indicadas, enquanto para **máxima velocidade**, em que uma margem de erro é aceitável, PaddleOCR em GPU é a solução mais eficiente.

4.4. Discussão e Contribuições

A análise comparativa realizada neste estudo não apenas responde à questão de pesquisa, como também explicita implicações práticas para o desenho e a implantação de soluções de OCR em cenários reais, onde decisões de software e hardware (infraestrutura) estão intrinsecamente acopladas. Em primeiro lugar, os resultados mostram que a escolha do motor de OCR impacta diretamente a experiência do usuário e a confiabilidade do dado extraído. Em atividades cartoriais e jurídicas, pequenas falhas de reconhecimento geram retrabalho, atrasos e incertezas na validação. Os achados indicam que soluções proprietárias tendem a maior acurácia, reduzindo correções manuais e aumentando a confiança no texto. Por outro lado, alternativas *open source* aceleradas em GPU oferecem tempos de processamento significativamente menores, o que é decisivo para

triagem, digitalização em lote e cenários de PD/capacitação.

Em segundo lugar, evidenciamos que o desempenho não é função apenas do algoritmo, mas do ambiente de execução e do *pipeline* como um todo. Executar localmente em CPU/GPU ou delegar a uma API em nuvem implica diferenças materiais de latência ponta-a-ponta, custo operacional, disponibilidade de hardware e dependência de conectividade. Consequentemente, o planejamento deve ser holístico: pré-processamento, inferência, pós-processamento, validação humana e políticas de reprocessamento precisam ser considerados conjuntamente, refletindo a operação real e requisitos de auditoria e rastreabilidade.

Por fim, os resultados sustentam a adoção de arquiteturas híbridas. Estratégias que combinam motores rápidos para alto volume com modelos mais precisos em documentos críticos ou em etapas de validação equilibram qualidade, custo e tempo de resposta. Esse arranjo favorece escalabilidade operacional, permite otimizar alocação de recursos (CPU, GPU, nuvem) e viabiliza políticas de roteamento por criticidade, qualidade de imagem ou sinais de incerteza do próprio modelo.

4.5. Limitações do Trabalho

Uma limitação central reside na assimetria de medição de eficiência entre soluções locais e serviços em nuvem. Para APIs (AWS Textract, Google Vision, Gemini), o tempo reportado inclui a latência de rede além da inferência. Já nos métodos locais (EasyOCR, PaddleOCR, Tesseract), mede-se o tempo de processamento no hardware disponível. Essa decisão metodológica é deliberada para refletir o indicador relevante para o usuário, que é a latência ponta-a-ponta, que inclui transmissão, processamento e retorno. No contexto brasileiro, onde a conectividade pode variar substancialmente, capturar esse componente é fundamental para decisões de adoção.

Portanto, a eventual desvantagem temporal das APIs deve ser interpretada como um custo operacional realista, e não como viés experimental. Adicionalmente, reconhecemos que diferentes políticas de *throttling* dos provedores, variações transitórias de carga e limites de taxa podem afetar medições de latência em nuvem. Para mitigar esse risco, optou-se por múltiplas iterações e aleatorização da ordem de execução, ainda que algum ruído residual possa permanecer.

5. Conclusão

Este trabalho conduziu um *benchmark* experimental para comparar ferramentas de OCR proprietárias e de código aberto na extração de dados de matrículas imobiliárias brasileiras, quantificando acurácia e eficiência com validação estatística rigorosa. Ao final, respondeu-se de forma objetiva à questão de pesquisa: ferramentas proprietárias (e.g., AWS Textract, Google Gemini) apresentam acurácia superior (em consonância com [Hegghammer 2021]), enquanto alternativas *open source* com aceleração por GPU (em especial PaddleOCR) alcançam maiores velocidades, explicitando o balanço entre custo computacional e precisão.

As implicações práticas abrangem a dimensão operacional, a medida que o desenho de fluxos de extração se tornam mais confiáveis e eficientes, reduzindo retrabalho por meio de políticas de validação humana informadas por métricas (e.g., *confidence*,

WER/CER) e por roteamento seletivo. Além disso, apoia-se o processo de definição arquitetural, pois o estudo observou que adoção de arranjos híbridos conciliam rapidez para alto volume com precisão para casos críticos, auxiliando a escolha entre CPU, GPU e nuvem segundo requisitos de latência ponta-a-ponta, disponibilidade e custo.

Entre as limitações, destaca-se a ausência de *fine-tuning* específico por domínio, que poderia reduzir lacunas de acurácia, e o foco em um tipo documental, ainda que complexo (matrículas), o que recomenda cautela ao generalizar para outros contextos. Como trabalhos futuros, propõe-se: (a) extensão a outros documentos legais/administrativos com diferentes padrões de layout e ruído; (b) investigação de *fine-tuning* e de pós-correção híbrida (por exemplo, combinar a velocidade do PaddleOCR com a acurácia do Gemini em etapas de validação); (c) análises de custo-benefício e sustentabilidade operacional; e (d) avaliação de novos modelos e *toolchains*, incluindo Calamari [Wick et al. 2020], além de estratégias baseadas em incerteza para roteamento dinâmico.

Em síntese, fornecemos evidências reprodutíveis para apoiar decisões de arquitetura e implantação de OCR em cenários reais de serviços documentais, contribuindo tanto para a prática (cartórios, órgãos públicos, TI) quanto para a pesquisa aplicada em engenharia de software e processamento de imagens. O código-fonte encontra-se em repositório público *[omitido para revisão]*.

6. Agradecimentos

Os autores agradecem ao Cartório de Registro de Imóveis de Correntina pela disponibilização das matrículas de imóveis em consonância com a Lei Geral de Proteção de Dados Pessoais (LGPD). Conforme o Código de Conduta da SBC, destaca-se que foi utilizada a IA Generativa do Gemini Flash 2.5 na revisão do conteúdo deste artigo. Os autores revisaram criticamente todo o conteúdo gerado e assumem total responsabilidade pela versão final do trabalho, incluindo sua precisão, originalidade e as conclusões apresentadas.

Referências

Amazon Web Services (2025). *Amazon Textract Documentation*. AWS. https://aws.amazon.com/textract/, Setembro.

Brasil (1973). Lei nº 6.015, de 31 de dezembro de 1973. https://www.planalto.gov.br/ccivil_03/leis/l6015compilada.htm, Setembro.

Chaitra, M., M J, D., Gopalakrishna, M., Md, S., and Aditya, C. (2023). *Text Detection and Recognition from the Scene Images Using RCNN and EasyOCR*, pages 75–85.

Cui, C., Sun, T., Lin, M., Gao, T., Zhang, Y., Liu, J., Wang, X., Zhang, Z., Zhou, C., Liu, H., Zhang, Y., Lv, W., Huang, K., Zhang, Y., Zhang, J., Zhang, J., Liu, Y., Yu, D., and Ma, Y. (2025). Paddleocr 3.0 technical report. https://arxiv.org/abs/2507.05595.

Filho, E. S. L., Falavigna, J. O. B., and de Lima, A. S. (2019). Hemeroteca digital. *Anais Estendidos do XV Simpósio Brasileiro de Sistemas de Informação*, pages 132–133. https://sol.sbc.org.br/index.php/sbsi_estendido/article/view/7454.

Francis, S. and Sangeetha, M. (2025). A comparison study on optical character recognition models in mathematical equations and in any language. *Results in Control and Optimization*, 18:100532.

Gemma (2024). Gemma: Open models based on gemini research and technology.

Google (2025). *Cloud Vision API Documentation*. Google Cloud. https://cloud.google.com/vision, Setembro.

Google DeepMind (2023). *Gemini: Multimodal AI Models*. https://deepmind.google/technologies/gemini/, Setembro.

Gupta, U., Kumar, A., Gupta, A., Raj, G., and Agrawal, A. P. (2024). Advances in handwritten character recognition: A comparison of ocr and large language model-based approaches. In *2024 International Conference on Emerging Technologies and Innovation for Sustainability (EmergIN)*, pages 192–198.

Hegghammer, T. (2021). Ocr with tesseract, amazon textract, and google document ai: a benchmarking experiment. *Journal of Computational Social Science*, 5:861 – 882.

JaidedAI (2025). GitHub - JaidedAI/EasyOCR: Ready-to-use OCR with 80+ supported languages and all popular writing scripts including Latin, Chinese, Arabic, Devanagari, Cyrillic and etc. — github.com. https://github.com/JaidedAI/EasyOCR, Setembro.

Kanagarathinam, K. and Sekar, K. (2019). Text detection and recognition in raw image dataset of seven segment digital energy meter display. *Energy Reports*, 5:842–852.

Lima, F. and Silva, E. (2025). Enhancing text recognition in ocr systems through image processing with bsrgan. In *Anais do XXI Simpósio Brasileiro de Sistemas de Informação*, pages 497–505, Porto Alegre, RS, Brasil. SBC.

Silva, J., Resendo, L., Andrade, J., and Komati, K. (2021). Comparação de apis de ocr para reconhecimento de dígitos em imagens de mostrador de sete segmentos. In *Anais do VIII Encontro Nacional de Computação dos Institutos Federais*, pages 33–40, Porto Alegre, RS, Brasil. SBC.

Smith, R. (2007). An overview of the tesseract ocr engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 629–633.

Tafti, A. P., Baghaie, A., Assefi, M., Arabnia, H. R., Yu, Z., and Peissig, P. L. (2016). Ocr as a service: An experimental evaluation of google docs ocr, tesseract, abbyy finereader, and transym. In *International Symposium on Visual Computing*.

Vijayarani, D. S. and Sakila, M. A. (2016). Text extraction from tamil and hindi document images using open source optical character recognition tools. https://api.semanticscholar.org/CorpusID:212595888, Setembro.

Vijayarani, S. and Sakila, A. (2015). Performance comparison of ocr tools. *International Journal of UbiComp*, 6:19–30.

Wick, C., Reul, C., and Puppe, F. (2018). Comparison of ocr accuracy on early printed books using the open source engines calamari and ocropus. *Journal for Language Technology and Computational Linguistics*, 33:79–96.

Wick, C., Reul, C., and Puppe, F. (2020). Calamari - A High-Performance Tensorflow-based Deep Learning Package for Optical Character Recognition. *Digital Humanities Quarterly*, 14(1).