Evaluating *Speech-to-Text* Models for Brazilian Portuguese in *Égua Assist*: Supporting Programming Learning for People with Visual Impairments

Daniel Saavedra dos Santos[1], **Paula Christina F. Cardoso**[1], **Victor Hugo S. C. Pinto**[1]

[1]Instituto de Ciências Exatas e Naturais (ICEN)
Universidade Federal do Pará (UFPA)
Rua Augusto Corrêa, 01, Guamá – 66075-110 – Belém – PA – Brasil

daniel.santos@icen.ufpa.br, {pcardoso,victor.santiago}@ufpa.br

Abstract. *This paper presents a comparative evaluation of nine Speech-to-Text model configurations — including the Web Speech API, Gemini 2.0 Flash, and seven OpenAI Whisper variants — for Brazilian Portuguese in Égua Assist, an assistive tool supporting visually impaired students in early programming learning. Using 200 audio samples (100 synthetic, 100 real) from 20 participants under varied acoustic conditions, models were evaluated in terms of accuracy (WER, CER, BLEU), efficiency (RTF), and stability (error distribution and outlier analysis). Whisper via API achieved the best trade-off among accuracy, latency, and robustness, while compact local models showed high instability and hallucination-like failures on real audio. These findings provide empirical guidance for real-time ASR model selection in voice-driven assistive educational applications for Brazilian Portuguese.*

Resumo. *Este artigo apresenta uma avaliação comparativa de nove configurações de modelos de Conversão de Fala em Texto (Speech-to-Text), incluindo a Web Speech API, o Gemini 2.0 Flash e sete variantes do OpenAI Whisper, para o português brasileiro no contexto do Égua Assist, uma ferramenta assistiva voltada ao apoio de estudantes com deficiência visual na aprendizagem inicial de programação. Utilizando 200 amostras de áudio (100 sintéticas e 100 reais) provenientes de 20 participantes, em diferentes condições acústicas, os modelos foram avaliados quanto à acurácia (WER, CER e BLEU), eficiência (RTF) e estabilidade (distribuição de erros e análise de valores atípicos). O Whisper via API apresentou o melhor equilíbrio entre precisão, latência e robustez, enquanto modelos locais compactos demonstraram alta instabilidade e falhas semelhantes a alucinações em áudios reais. Esses resultados fornecem evidências empíricas para orientar a seleção de modelos de reconhecimento automático de fala (ASR) em tempo real para aplicações educacionais assistivas baseadas em voz no português brasileiro.*

1. Introdução

Na última década, o Reconhecimento Automático de Fala (ASR, do inglês *Automatic Speech Recognition*) passou por uma transformação fundamental, migrando de arquiteturas híbridas baseadas em Modelos Ocultos de Markov (HMM) e Misturas de Gaussianas (GMM) para sistemas *End-to-End* baseados em *deep learning* [Prabhavalkar et al. 2023].

Para o português brasileiro (PB), essa evolução foi acompanhada pela expansão de corpora de treinamento e pela adoção de grandes modelos pré-treinados multilíngues — como Wav2Vec 2.0 XLS-R [Babu et al. 2021] e OpenAI *Whisper* [Radford et al. 2022] —, que ampliaram a cobertura dialetal do idioma. Apesar desses avanços, a avaliação de modelos de ASR permanece concentrada em *benchmarks* gerais, com escassez de evidências sobre seu comportamento em aplicações interativas, assistivas e de domínio específico.

Essa lacuna é especialmente relevante em ferramentas de apoio ao aprendizado de programação para pessoas com deficiência visual (PcDV). Nesses sistemas, erros de transcrição propagam-se por todo o *pipeline* de interação: comprometem a interpretação da intenção do usuário, afetam a geração de código, podem provocar falhas na execução e degradam a qualidade do retorno acessível fornecido ao estudante. Por isso, a seleção de um módulo de ASR com base exclusivamente na minimização do WER (*Word Error Rate*) médio é insuficiente — o uso prático exige também comportamento previsível, robustez a ruídos e viabilidade de operação em tempo real.

Este artigo avalia comparativamente modelos de conversão de fala para texto no contexto da *Égua Assist* [Santos et al. 2025], ferramenta educacional assistiva para PcDV no aprendizado inicial de programação na linguagem Égua [Neves 2025]. O estudo concentra-se no bloco de transcrição — primeiro estágio crítico do *pipeline* e condicionante direto da qualidade de todas as etapas subsequentes — comparando Web Speech API, Gemini 2.0 Flash e sete variantes do OpenAI *Whisper* (Tiny, Base, Small, Medium, Large, Turbo e API). O PB apresenta desafios particulares para ASR, como variação de sotaque, nasalização, redução vocálica e *code-switching*, dado que comandos de programação frequentemente incorporam termos em inglês em enunciados em português. As principais contribuições deste artigo são: **(i)** avaliação comparativa de nove configurações de ASR em cenário educacional assistivo para programação por voz em PB; **(ii)** análise conjunta de precisão (WER, CER, BLEU), eficiência (RTF) e estabilidade em áudios sintéticos e reais; e **(iii)** identificação de perfis de aplicabilidade para cada família de modelos, com recomendações práticas para sistemas assistivos interativos.

2. Fundamentação Teórica

2.1. Evolução das Arquiteturas de ASR

Os sistemas de ASR migraram de arquiteturas híbridas HMM-GMM [Rabiner 1989, Wang et al. 2019] para modelos *End-to-End* (E2E) baseados em *Transformers* [Vaswani et al. 2017, Chan et al. 2015]. Atualmente, dominam o estado da arte o aprendizado auto-supervisionado (e.g., Wav2Vec 2.0 [Baevski et al. 2020]) e o treinamento supervisionado em larga escala, como o *Whisper* [Radford et al. 2022], que garante robustez a ruídos sem *fine-tuning* específico. Em aplicações assistivas interativas, a seleção do modelo deve transcender a precisão bruta, ponderando também latência, custo computacional e estabilidade.

2.2. Métricas de Avaliação, Normalização e Eficiência

A precisão de sistemas de ASR é medida pelo WER [Jurafsky and Martin 2026], baseado na Distância de Levenshtein [Levenshtein 1966], que contabiliza substituições (S), deleções (D) e inserções (I) em relação ao total de palavras de referência (N), conforme

a Equação 1. O *Character Error Rate* (CER) aplica a mesma lógica em nível de caractere [Wang et al. 2019], permitindo identificar erros ortográficos e fonéticos mais finos — relevantes em programação, onde pequenas alterações em identificadores, operadores ou valores numéricos comprometem a geração correta de código. O escore BLEU [Papineni et al. 2002] complementa a avaliação, medindo a preservação da estrutura sequencial das transcrições em relação às referências.

$$WER = \frac{S + D + I}{N} \tag{1}$$

A eficiência temporal é medida pelo *Real-Time Factor* (RTF = T_{proc}/T_{audio}) [Microsoft 2025]: valores $RTF < 1{,}0$ indicam processamento mais rápido que a duração do áudio — condição necessária para operação em tempo real. Antes do cálculo das métricas, aplica-se normalização textual uniforme: conversão para minúsculas, remoção de pontuação e padronização de espaços. Numerais e palavras-chave da linguagem Égua não são normalizados, pois possuem significado semântico direto no contexto de geração de código — sua alteração mascararia erros relevantes para o uso prático da ferramenta.

2.3. Estabilidade, Robustez e o Português Brasileiro

Métricas médias (como WER e CER) podem mascarar instabilidades e falhas severas, como alucinações [Kalai et al. 2025], que comprometem fatalmente o uso de ferramentas assistivas. Por isso, a análise de distribuição de erros (via *boxplots*) e o contraste entre áudios sintéticos (controlados) e reais (sujeitos a ruídos e hesitações) são essenciais para não superestimar o desempenho prático dos modelos.

Além disso, o Português Brasileiro (PB) impõe desafios acústicos ao ASR, como variação diatópica, nasalização [Cristófaro-Silva 2010] e *code-switching* [Sitaram et al. 2020], muito presente no jargão de programação. Embora o surgimento de grandes *corpora* (CORAA [Candido Junior et al. 2023], *Common Voice* [Ardila et al. 2020] e MLS [Pratap et al. 2020]) tenha impulsionado a área, avaliações direcionadas a cenários educacionais e assistivos continuam escassas na literatura.

3. Trabalhos Relacionados

A avaliação de sistemas de ASR para o português tem evidenciado que a seleção de um modelo constitui um desafio de otimização multidimensional. Fatores como precisão (WER, BLEU), custo, latência e impacto do pós-processamento são determinantes, conforme demonstrado pelas avaliações de APIs e modelos acadêmicos conduzidas por Magalhães et al. (2022) e Oliveira (2022). Carvalho et al. (2025) reforçam essa visão ao apresentar o CAMÕES, ilustrando que treinamentos com múltiplas variantes dialetais do português geram modelos mais robustos.

No domínio da acessibilidade, Zen et al. (2023) mapearam barreiras severas enfrentadas por estudantes com deficiência visual no ensino de programação. Embora a geração de código por voz tenha sido explorada por Nizzad e Thelijjagoda (2022) e Hossain et al. (2021), tais soluções restringem-se ao inglês ou a vocabulários rigidamente controlados. Nenhum desses trabalhos avalia modelos de ASR operando em tempo real com vocabulário técnico de programação em português brasileiro — uma lacuna assistiva e educacional que o presente artigo visa endereçar.

4. Égua Assist

A *Égua Assist* [Santos et al. 2025] é uma ferramenta web assistiva voltada ao ensino inicial de programação para PcDV por meio da linguagem Égua [Neves 2025]. Operada por voz, atalhos de teclado e *feedback* sonoro, dispensando leitores de tela externos, ela utiliza IA para converter comandos em código e auxiliar na correção de erros.

Conforme a Figura 1, sua arquitetura possui quatro etapas: o **Bloco de Transcrição**, foco deste estudo, que converte voz em texto via ASR, condicionando a qualidade de todo o *pipeline*; o **Bloco de Tradução**, que interpreta o *prompt* via LLMs (e.g., Gemini 2.0 Flash) e RAG; a **Validação**, que exige confirmação sonora do usuário antes de prosseguir; e a **Execução**, que roda o código validado e retorna uma explicação em áudio amigável sobre os resultados ou exceções geradas.

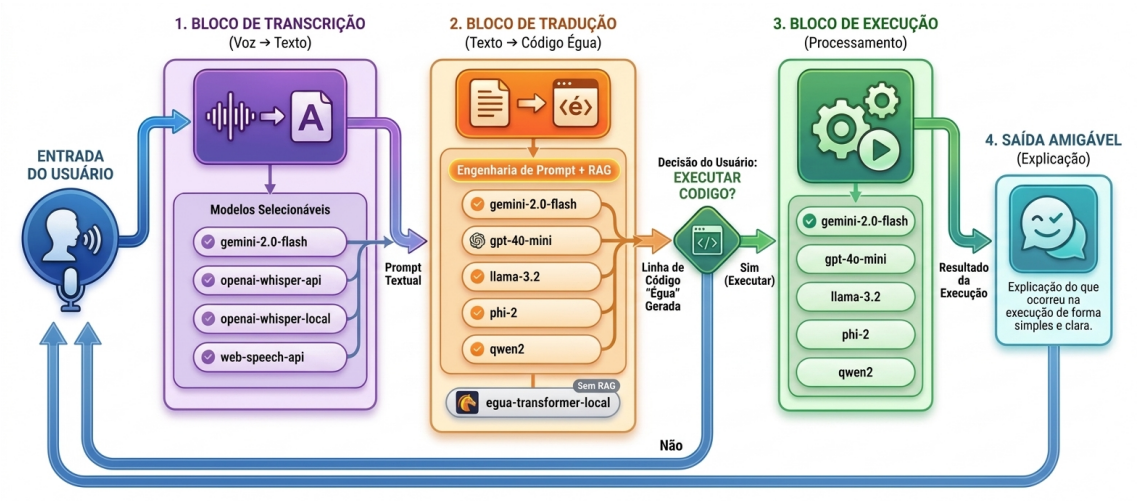


Figura 1. Estrutura geral da ferramenta *Égua Assist*, separada por blocos funcionais.

5. Metodologia

5.1. Coleta e Preparação dos Dados

O conjunto de dados é composto por 100 *prompts* representativos de comandos típicos da Égua Assist. Os comandos avaliados caracterizam-se por instruções de frase única capturadas pelo microfone, com vocabulário restrito e imperativo. O *dataset* foi estruturado a partir de uma taxonomia de tarefas fundamentais para a construção da lógica algorítmica inicial, cuja distribuição e exemplos reais extraídos do corpus encontram-se sumarizados na Tabela 1.

Esses enunciados traduzem-se em uma sequência de *tokens* (e.g., `['crie', 'uma', 'variável', 'x', 'com', 'valor', '4']`) processada por LLMs para a geração do código correspondente. Cada um dos 100 comandos gerou duas amostras de áudio: uma sintética e uma real, totalizando 200 gravações (formato .wav, 16 kHz, mono). As amostras sintéticas foram geradas via Google AI Studio[1] com voz computacional em português brasileiro (prosódia regular e ausência de ruído de fundo).

[1]Google AI Studio: https://aistudio.google.com/

Tabela 1. Taxonomia, distribuição e exemplos dos comandos (*prompts*) avaliados no conjunto de dados.

Categoria de Tarefa	Proporção	Exemplo de *Prompt* no *Dataset* (Referência)
Saída de Dados (Impressão)	35%	"concatenar as strings João e Souza e exibir o resultado"
Declaração e Atribuição de Variáveis	25%	"crie uma variável chamada prateleiras com o valor 6"
Estruturas Condicionais (Se / If)	18%	"crie uma condição que verifique se a variável x é maior que vinte e incremente a variável contador em um"
Blocos de Alternativa (Senão / Else)	12%	"crie um bloco senão que escreva saldo insuficiente para saque"
Operações Aritméticas	10%	"some os números 10 20 e 30 e armazene na variável total"

As amostras reais foram gravadas por 20 participantes voluntários (16 homens e 4 mulheres; 14 entre 20-30 anos, 4 entre 30-40 anos, 2 entre 40-50 anos), predominantemente das regiões Norte e Nordeste (19 do Pará e 1 do Maranhão). Para o contexto de gravação, a metodologia adotada consistiu na *leitura* dos comandos previamente definidos (Tabela 1). O objetivo dessa abordagem controlada foi estabelecer um marco de referência para verificar o desempenho dos modelos em transcrição direta, isolando variáveis cognitivas relacionadas à formulação do problema.

Com o intuito de maximizar a validade acústica e capturar uma ampla gama de condições ambientais, os participantes utilizaram o microfone de seus próprios *smartphones*, sem supervisão em laboratório. Como resultado, o corpus real engloba variações naturais de ruído de fundo, eco de ambiente e distorções de hardware.

Embora o protocolo tenha se baseado em leitura, observou-se a manifestação de fenômenos típicos da fala espontânea. Usuários sem experiência prévia na construção de enunciados computacionais apresentaram pausas, hesitações e reformulações pontuais ao ler comandos de maior extensão (como as estruturas condicionais). Esses fenômenos afetam diretamente o desempenho do reconhecimento automático e refletem os desafios de usabilidade frequentemente encontrados por usuários em estágios iniciais de letramento tecnológico ao interagirem com interfaces de voz [da Silva et al. 2024].

5.2. Configuração dos Modelos e Normalização

Foram avaliadas nove configurações organizadas em duas categorias. Os **modelos em nuvem** compreendem: (i) **Web Speech API**, API nativa do navegador baseada nos serviços do Google; (ii) **Gemini 2.0 Flash**, modelo multimodal do Google com suporte nativo ao português brasileiro; e (iii) **Whisper API**, versão hospedada do Whisper Large pela OpenAI via API REST. Os **modelos locais** compreendem seis variantes do OpenAI Whisper: (iv) **Tiny**; (v) **Base**; (vi) **Small**; (vii) **Medium**; (viii) **Large**; e (ix) **Turbo**. Todos os modelos locais foram executados em modo *zero-shot* com parâmetro de idioma fixado em

`pt`. Os experimentos locais foram conduzidos em Intel Core i7 12ª geração, 16 GB RAM e GPU Intel Alder Lake-UP3 GT2 (Iris Xe Graphics), executando Python 3.10 com PyTorch 2.1. Antes do cálculo das métricas, aplicou-se normalização uniforme: conversão para minúsculas, remoção de pontuação (exceto apóstrofos), padronização de espaços e remoção de marcadores de hesitação. Numerais e palavras-chave da linguagem Égua não foram normalizados, preservando seu significado semântico para geração de código.

5.3. Dimensões de Avaliação

A avaliação baseou-se em três dimensões independentes nos áudios sintéticos e reais, evitando o viés de métricas unidimensionais. A **precisão** foi mensurada por WER, CER e BLEU [Papineni et al. 2002], quantificando tanto erros lexicais quanto estruturais. A **eficiência temporal** adotou a mediana do *Real-Time Factor* (RTF) [Microsoft 2025] após três execuções por amostra, mitigando flutuações de carga do sistema. Por fim, avaliou-se a **estabilidade** via diagramas de caixa (*boxplots*) para mapear a variância e detectar alucinações [Kalai et al. 2025]. Valores de WER superiores a 1,0 (100%) foram interpretados como indicativos de alucinação (o modelo inventou mais palavras do que o existente na referência). Essa análise de *outliers* é vital em cenários assistivos, onde um erro severo isolado corrompe a experiência do usuário de forma mais drástica que um leve aumento na taxa média de erros.

6. Resultados e Discussão

Os resultados são apresentados em quatro dimensões analíticas: precisão (WER, CER e BLEU), estabilidade, eficiência computacional (RTF) e correlação entre métricas. Em seguida, discutem-se as implicações para a seleção de modelos de ASR em aplicações assistivas interativas.

6.1. Precisão: WER e CER

A Figura 2 apresenta os valores de WER e CER para todos os modelos avaliados, separados por tipo de áudio. O CER acompanha de forma consistente o WER em todos os modelos, indicando que os erros não se restringem a substituições lexicais superficiais, mas refletem imprecisões fonéticas e ortográficas que comprometem a estrutura interna das palavras transcritas.

O desempenho em áudio sintético é razoável mesmo para modelos compactos, enquanto o áudio real expõe limitações severas das arquiteturas menores. O *Whisper Tiny* e o *Whisper Base* apresentam degradação severa em áudio real, com taxas de WER superiores a 35%, tornando-se soluções pouco confiáveis para transcrição de fala natural. Em contraste, o *Whisper Turbo* demonstra forte capacidade de generalização, competindo diretamente com o *Whisper Large* e o *Whisper API*, ambos com WER inferior a 10% em áudio real.

A Web Speech API e o Gemini 2.0 Flash apresentaram desempenho intermediário: superior aos modelos Whisper compactos em áudio real, mas com maior variabilidade entre amostras do que as soluções de maior porte. Esses resultados sugerem que soluções comerciais em nuvem oferecem um ponto de equilíbrio viável para cenários com restrições de infraestrutura local, mas sem as garantias de estabilidade do *Whisper API*.

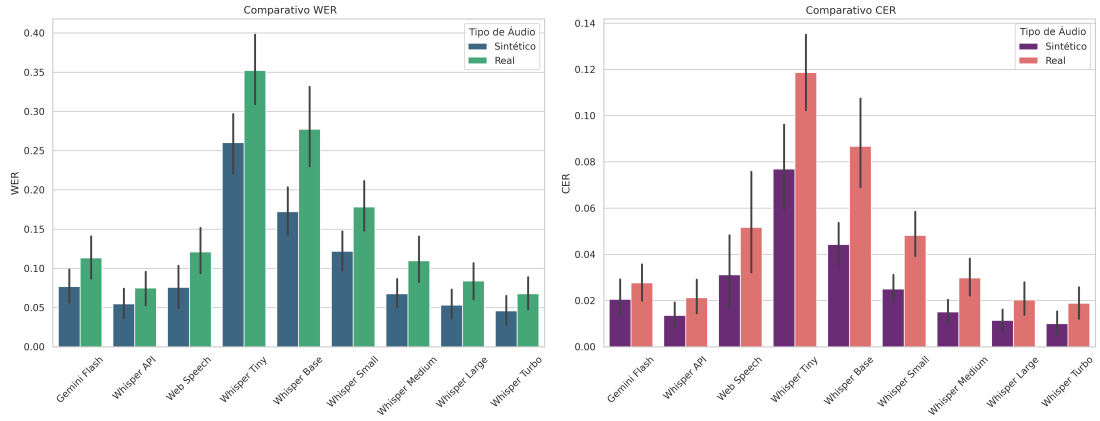


Figura 2. Comparação de WER e CER entre os modelos avaliados, por tipo de áudio (sintético e real).

6.2. Estabilidade e Consistência

A Figura 3 apresenta a distribuição dos erros WER por meio de diagramas de caixa, revelando comportamentos ocultos nas médias reportadas na subseção anterior.

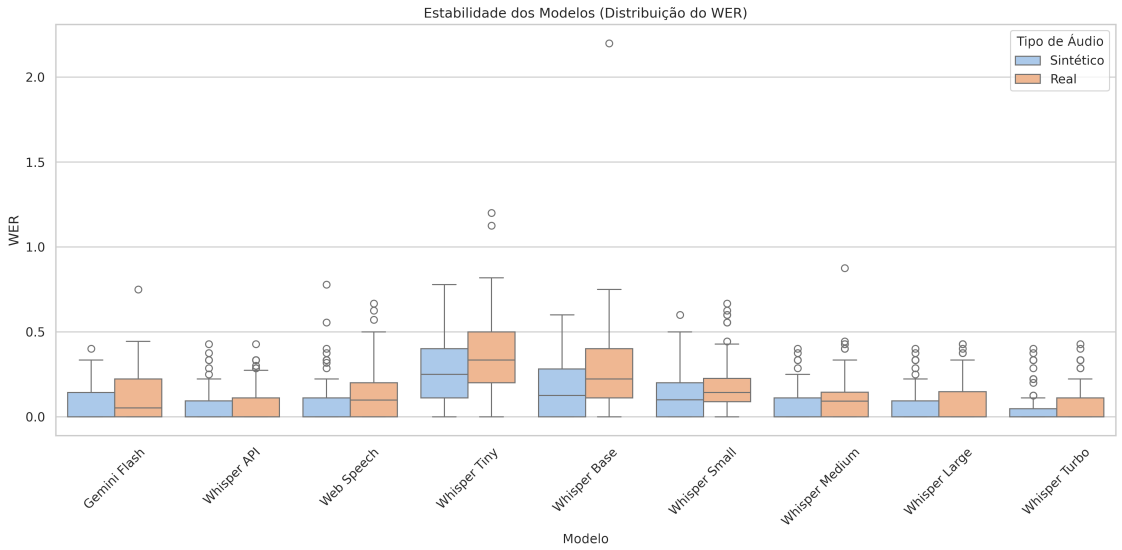


Figura 3. Distribuição do WER por modelo. A altura das caixas indica a variância inter-amostral; pontos acima dos limites representam *outliers*.

O *Whisper Base* apresenta *outliers* com WER superior a 2,0, compatíveis com episódios de alucinação — situações em que o modelo gerou mais palavras incorretas do que o total de palavras na referência, seja por repetição de frases ou por transcrição completamente desconectada do áudio de entrada. Em contraste, *Whisper API*, *Whisper Large* e *Whisper Turbo* apresentam distribuições extremamente compactas, com baixa variância e ausência de *outliers* expressivos, indicando alta previsibilidade de comportamento independentemente do perfil do locutor.

Esse contraste é crítico para aplicações assistivas. Em ferramentas como a *Égua Assist*, uma falha catastrófica isolada — mesmo em um modelo com boa média — pode

comprometer toda a sessão de aprendizado do usuário, tornando a estabilidade um critério de seleção tão relevante quanto a precisão média.

6.3. Eficiência Computacional e Viabilidade em Tempo Real

A Figura 4 apresenta os valores de RTF para cada modelo. Os modelos *Whisper Large* e *Whisper Turbo*, quando executados localmente em hardware convencional, apresentam $RTF > 2{,}0$ em áudio real, inviabilizando sua adoção em aplicações que exigem transcrição em tempo real sem acesso a GPUs de alto desempenho. O *Whisper Small* é o único modelo local que mantém $RTF < 1{,}0$ com qualidade de transcrição aceitável, representando o ponto de inflexão entre viabilidade temporal e precisão para implantações locais. O *Whisper Tiny*, embora computacionalmente barato, tem utilidade prática limitada pelo alto risco de alucinações observado na análise de estabilidade.



Figura 4. Fator de Tempo Real (RTF) por modelo. Valores acima de $1{,}0$ inviabilizam a operação em tempo real.

O *Whisper API* destaca-se por romper a correlação esperada entre tamanho do modelo e latência: ao delegar o processamento para infraestrutura otimizada em nuvem, oferece precisão equivalente ao *Whisper Large* com RTF significativamente inferior, posicionando-se como a solução de maior custo-benefício global no conjunto avaliado.

6.4. Correlação entre Métricas

A Figura 5 apresenta os coeficientes de correlação de Pearson [Schober et al. 2018] e de Kendall [Abdi 2007] entre as métricas. Observou-se forte associação positiva entre WER e CER ($r = 0{,}81$; $\tau = 0{,}83$), indicando que erros lexicais costumam acarretar degradação proporcional em nível de caracteres. Como esperado, a associação entre WER e BLEU foi forte e negativa ($r = -0{,}90$; $\tau = -0{,}88$), visto que métricas de erro minimizadas convergem naturalmente para uma maior preservação estrutural e sobreposição com a referência (maximizada pelo BLEU). Por fim, a correlação entre tempo de execução e WER apresentou-se negativa e fraca ($r = -0{,}32$; $\tau = -0{,}26$). Isso evidencia que o aumento do custo computacional não garante uma redução linear e proporcional na taxa de erros, reforçando que a eficiência empírica dos modelos deve ser avaliada pelo balanço conjunto entre precisão e latência, e não de forma isolada.

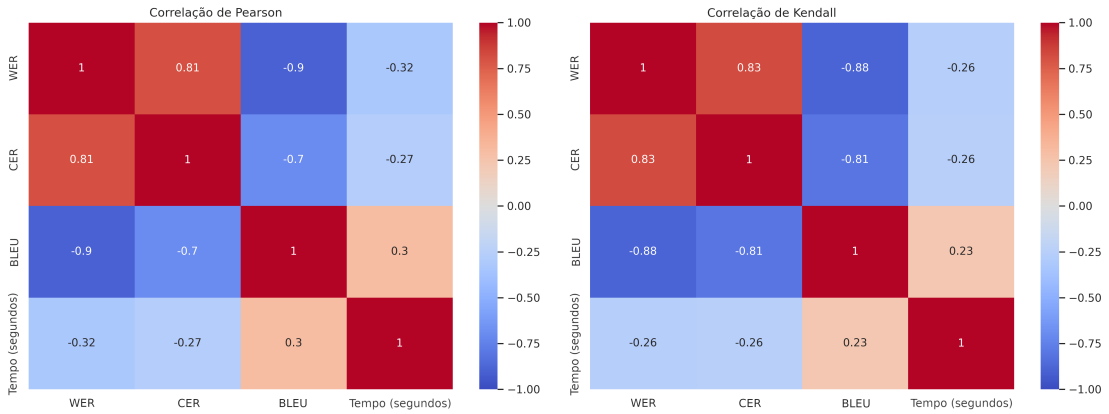


Figura 5. Matrizes de correlação de Pearson e Kendall entre as métricas de avaliação.

6.5. Perfis de Aplicabilidade e Limitações

A Figura 6 sintetiza o *trade-off* entre precisão (WER) e eficiência (RTF) para todos os modelos avaliados.

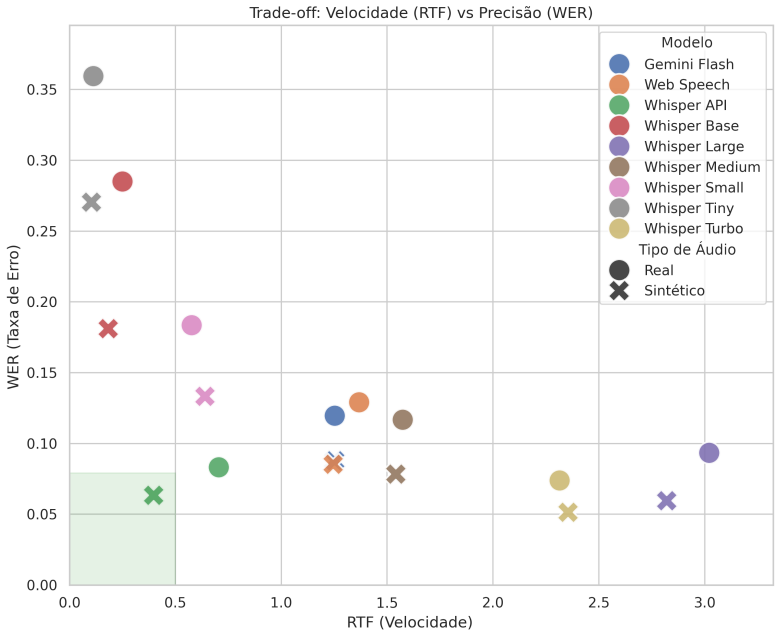


Figura 6. Trade-off entre precisão (WER) e eficiência (RTF). O quadrante ideal — baixo WER e baixo RTF — é ocupado exclusivamente pelo *Whisper API*.

Identificam-se três perfis de aplicabilidade: (1) *Whisper API* é a solução ideal sob conectividade estável, possuindo o melhor balanço entre precisão e latência (baixo WER e RTF); (2) *Whisper Turbo* otimiza o processamento local, entregando robustez equivalente ao *Large* com latência 30% menor; e (3) *Whisper Small* representa o limite de viabilidade para operação em tempo real sob restrições severas de hardware (CPU).

Quanto às limitações, a queda abrupta de desempenho dos modelos compactos no conjunto real evidencia dependência de padrões acústicos limpos e baixa generalização a ruídos. Adicionalmente, o tamanho da amostra (100 áudios, 20 locutores) restringe a representatividade dialetal, enquanto a ausência de testes estatísticos formais (e.g., Wilcoxon ou Friedman) demanda validações metodológicas em trabalhos futuros.

7. Conclusão

Este artigo apresentou uma avaliação comparativa de nove configurações de modelos de *Speech-to-Text* para o português brasileiro no contexto da *Égua Assist*, ferramenta educacional assistiva voltada ao apoio de PcDV no aprendizado inicial de programação. Os experimentos demonstraram que o desempenho dos modelos varia de forma expressiva entre áudio sintético e real, evidenciando que *benchmarks* baseados exclusivamente em fala controlada superestimam a capacidade de generalização dos modelos compactos. Os dados confirmam que a seleção de um modelo de ASR constitui um problema de otimização multidimensional: modelos com baixo WER médio podem apresentar alta variância e episódios de alucinação em áudio real, tornando estabilidade e latência critérios tão relevantes quanto a precisão. O *Whisper API* destacou-se como a solução mais adequada ao fluxo interativo da *Égua Assist*, por oferecer o melhor equilíbrio entre acurácia, latência e robustez; o *Whisper Turbo* mostrou-se a melhor alternativa para implantações locais; e o *Whisper Small* definiu o limite de viabilidade em tempo real em hardware convencional.

Este estudo apresenta limitações que devem ser consideradas: o subconjunto de áudio real (100 amostras, 20 locutores) restringe a representatividade dialetal do português brasileiro; a ausência de testes estatísticos formais impede afirmações de significância sobre as diferenças observadas; e a avaliação em modo *zero-shot* pode subestimar o potencial de modelos como o *Whisper Large* e o *Whisper Turbo* após especialização no domínio. Como trabalhos futuros, pretende-se: **(i)** ampliar o conjunto de dados com maior diversidade regional e de condições acústicas; **(ii)** realizar *fine-tuning* dos modelos locais com vocabulário da linguagem Égua; **(iii)** conduzir testes estatísticos formais (Wilcoxon, Friedman) para estabelecer a significância das diferenças de desempenho; **(iv)** investigar estratégias híbridas de execução local-nuvem; e **(v)** avaliar o impacto das configurações de ASR na experiência de aprendizado de PcDV com a *Égua Assist*.

Agradecimento

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

Referências

Abdi, H. (2007). The kendall rank correlation coefficient. *Encyclopedia of measurement and statistics*, 2:508–510.

Ardila, R., Branson, M., Davis, K., Henretty, M., Kohler, M., Meyer, J., Morais, R., Saunders, L., Tyers, F. M., and Weber, G. (2020). Common voice: A massively-multilingual speech corpus.

Babu, A., Wang, C., Tjandra, A., Lakhotia, K., Xu, Q., Goyal, N., Singh, K., von Platen, P., Saraf, Y., Pino, J., Baevski, A., Conneau, A., and Auli, M. (2021). Xls-r: Self-supervised cross-lingual speech representation learning at scale.

Baevski, A., Zhou, H., Mohamed, A., and Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations.

Candido Junior, A., Casanova, E., Soares, A., de Oliveira, F. S., Oliveira, L., Junior, R. C. F., da Silva, D. P. P., Fayet, F. G., Carlotto, B. B., Gris, L. R. S., and Aluísio, S. M. (2023). Coraa asr: a large corpus of spontaneous and prepared speech manually validated for speech recognition in brazilian portuguese. *Language Resources and Evaluation*, 57(3):1139–1171.

Carvalho, C., Teixeira, F., Botelho, C., Pompili, A., Solera-Ureña, R., Paulo, S., Julião, M., Rolland, T., Mendonça, J., Pereira, D., Trancoso, I., and Abad, A. (2025). Camões: A comprehensive automatic speech recognition benchmark for european portuguese.

Chan, W., Jaitly, N., Le, Q. V., and Vinyals, O. (2015). Listen, attend and spell.

Cristófaro-Silva, T. (2010). *Fonética e fonologia do português: roteiro de estudos e guia de exercícios*. Editora Contexto, São Paulo, 10 edition.

da Silva, T. H. O., Furtado, V., Furtado, E., Mendes, M., Almeida, V., and Sales, L. (2024). How do illiterate people interact with an intelligent voice assistant? *International Journal of Human–Computer Interaction*, 40(3):584–602.

Hossain, S., Emi, M. A., Mishu, M. H., Zannat, R., et al. (2021). Code generator based on voice command for multiple programming language. In *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, pages 01–05. IEEE.

Jurafsky, D. and Martin, J. H. (2026). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. 3rd edition. Online manuscript released January 6, 2026.

Kalai, A. T., Nachum, O., Vempala, S. S., and Zhang, E. (2025). Why language models hallucinate.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710.

Microsoft (2025). Evaluating performance of Embedded Speech. `https://learn.microsoft.com/en-us/azure/ai-services/speech-service/embedded-speech-performance-evaluations`. Última atualização em 07 agosto 2025; acessado em 09 janeiro 2026.

Neves, L. P. (2025). Linguagem Égua. `https://egua.dev/`. Programação simples e moderna em português. Acessado em: 1 de março de 2025.

Nizzad, A. and Thelijjagoda, S. (2022). Designing of a voice-based programming ide for source code generation: A machine learning approach. In *2022 International Research Conference on Smart Computing and Systems Engineering (SCSE)*, volume 5, pages 14–21. IEEE.

Oliveira, A. R. G. d. (2022). Avaliação de ferramentas de reconhecimento automático de fala para língua portuguesa. Trabalho de conclusão de curso (graduação em ciência da computação), Universidade Federal de Lavras. Defendido em 28 Apr. 2022; 43 p.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In Isabelle, P., Charniak, E., and Lin, D., editors,

Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Pires Magalhães, R., Rodrigues Vasconcelos, D. J., Sales Fernandes, G., Almada Cruz, L., Xavier Sampaio, M., Fernandes de Macêdo, J. A., and Linhares Coelho da Silva, T. (2022). Evaluation of automatic speech recognition approaches. *Journal of Information and Data Management*, 13(3).

Prabhavalkar, R., Hori, T., Sainath, T. N., Schlüter, R., and Watanabe, S. (2023). End-to-end speech recognition: A survey.

Pratap, V., Xu, Q., Sriram, A., Synnaeve, G., and Collobert, R. (2020). Mls: A large-scale multilingual dataset for speech research. In *Interspeech 2020*, interspeech_2020. ISCA.

Rabiner, L. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. (2022). Robust speech recognition via large-scale weak supervision.

Santos, D. S. d., Cardoso, P. C. F., and Pinto, V. H. S. C. (2025). Égua Assist: Uma ferramenta educacional assistiva para aprendizado de programação por pessoas com deficiência visual. In *Anais do Workshop de Informática na Escola*, pages 1–10. SBC.

Schober, P., Boer, C., and Schwarte, L. A. (2018). Correlation coefficients: Appropriate use and interpretation. *Anesthesia and Analgesia*, 126(5):1763–1768.

Sitaram, S., Chandu, K. R., Rallabandi, S. K., and Black, A. W. (2020). A survey of code-switched speech and language processing.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Wang, D., Wang, X., and Lv, S. (2019). An overview of end-to-end automatic speech recognition. *Symmetry*, 11(8).

Zen, E., Da Costa, V. K., and Tavares, T. A. (2023). Understanding the accessibility barriers faced by learners with visual impairments in computer programming. In *Proceedings of the XXII Brazilian Symposium on Human Factors in Computing Systems*, pages 1–11.