16º Congresso de Inovação, Ciência e Tecnologia do IFSP - 2025

DESENVOLVIMENTO DE UMA EXTENSÃO NO CHROME PARA CAPTAR TEXTO DE IMAGEM DE MANGÁ E TRADUZIR PARA PORTUGUÊS

MARIA LUIZA S. ARMANDO[1], PAULO HENRIQUE L. AZEVEDO[2], ROBSON F. LOPES[3]

[1] Cursando técnico em informática integrado ao Ensino Médio, IFSP, Campus Guarulhos, armando.maria@ifsp.edu.br
[2] Cursando técnico em informática integrado ao Ensino Médio, IFSP, Campus Guarulhos, lima.azevedo@ifsp.edu.br
[3] Mestrado em titulação, IFSP, Campus Guarulhos, rferreira@ifsp.edu.br

RESUMO: Este projeto tem como objetivo o desenvolvimento de uma extensão para o navegador Chrome, que possibilita a tradução de textos embutidos em imagens para o português, permitindo aos leitores de mangás acesso ético e simples ao conteúdo disponibilizado em sites oficiais, desta forma, contribuindo para o combatendo a pirataria dessas mídias. Com foco na leitura de mangás e outras mídias visuais, o projeto utilizará técnicas de Reconhecimento Óptico de Caracteres (OCR) integradas à API do Google Tradutor, buscando superar as limitações dos tradutores automáticos existentes. A metodologia adotada inclui pesquisa bibliográfica, desenvolvimento de protótipos e testes práticos para assegurar a funcionalidade e precisão da ferramenta. O código desenvolvido até o momento já permite capturar imagens da tela, identificar textos e traduzi-los para inglês, português, japonês, chinês tradicional, coreano e espanhol, representando um avanço importante para a solução proposta, que busca ser acessível, eficiente e expansível.

PALAVRAS-CHAVE: extensão; ocr; api; mangá; pirataria;

DEVELOPMENT OF A CHROME EXTENSION TO CAPTURE TEXT FROM mangá IMAGES AND TRANSLATE IT INTO PORTUGUESE

ABSTRACT: This project aims to develop an extension for the Chrome browser that enables the translation of text embedded in images into Portuguese, providing manga readers with ethical and easy access to content made available on official websites, thereby contributing to the fight against media piracy. Focusing on manga and other visual media, the project will utilize Optical Character Recognition (OCR) techniques integrated with the Google Translate API, seeking to overcome the limitations of existing automatic translators. The adopted methodology includes literature review, prototype development, and practical testing to ensure the tool's functionality and accuracy. The code developed so far already allows for screen image capture, text recognition, and translation into English, Portuguese, Japanese, Traditional Chinese, Korean, and Spanish — representing a significant advancement for the proposed solution, which aims to be accessible, efficient, and scalable.

KEYWORDS: extension; ocr; api; manga; piracy;

INTRODUÇÃO

O mercado de mangás carrega uma importância na sociedade, incentivando o hábito de leitura, aproximando pessoas e espalhando a cultura japonesa. Como destaca Giovana Santana na pesquisa "Os fãs da Cultura Pop Japonesa e a Prática de Scanlation no Brasil":

> "O mangá representa um novo momento para as histórias em quadrinhos no país (e em outros também), tanto por possibilitar o surgimento do "mangá brasileiro" ou histórias em quadrinhos brasileiras "em estilo mangá", quanto à grande oferta de títulos japoneses (seja via editoras nacionais ou scanlation) e à forte segmentação dos mangás, os quais geram mais possibilidades de atrair novos leitores [...]" (Santana, 2011, p. 17).

Entretanto, parte desses conteúdos é fornecida por *scanlators* , pessoas que realizam a distribuição de mangás traduzidos não oficiais, com o objetivo de facilitar a leitura dos fãs. O problema é que esse ato se enquadra como pirataria.

O objetivo desse projeto é desenvolver uma extensão que reconheça textos exibidos na tela do usuário por imagem e os traduza automaticamente para o idioma desejado, substituindo o texto original pela tradução, utilizando a *API* do *Google Tradutor*. Para isso, é necessário: desenvolver habilidades de programação voltadas para a criação de extensões; analisar as extensões de tradução existentes para identificar suas limitações e explorar soluções para superá-las; testar métodos de reconhecimento de texto em imagem através de códigos abertos, a fim de descobrir ou criar a abordagem mais eficaz para a extensão; desenvolver um protótipo funcional da extensão para garantir sua eficácia e identificar áreas de melhoria.

MATERIAL E MÉTODOS

Antes de começar a programação, foi realizada uma pesquisa por meio dos vídeos de Attekita (2023) e Dütra (2021) sobre o *Git*, uma ferramenta que possibilita o trabalho colaborativo em códigos, e o *GitHub*, plataforma destinada ao armazenamento e compartilhamento de projetos. Essas ferramentas foram estudadas por sua importância no desenvolvimento colaborativo de *software*.

Foram realizados estudos técnicos acerca dos principais recursos de programação que seriam utilizados no decorrer da codificação. O principal meio de estudo foram os vídeos do canal Hashtag Programação sobre as seguintes ferramentas: *Tesseract*, um programa capaz de capturar texto em imagens; *Pillow*, uma biblioteca destinada à edição de imagens; e *PyAutoGUI*, uma biblioteca utilizada para realizar capturas de tela. Essas ferramentas, em conjunto, possibilitaram tanto a captura de tela quanto a extração de textos de imagens.

Também foram consultados os documentos oficiais referentes ao *JSON*, com o objetivo de compreender sua utilização na serialização de objetos em um formato compatível com a transmissão e interpretação entre diferentes sistemas e linguagens.

Além disso, realizaram-se estudos sobre o desenvolvimento de extensões para o navegador *Chrome*, identificando suas limitações e requisitos. O principal arquivo identificado nesse processo foi o *manifest.json*, obrigatório em todas as extensões do *Chrome*, que contém informações como versão, nome, descrição, permissões necessárias e ações executadas pela extensão.

Para otimizar o progresso, o grupo de trabalho foi dividido em duas frentes: uma dedicada ao desenvolvimento do código e outra à pesquisa de formas de integrar o *Python* às extensões do *Chrome*, uma vez que seria necessário que o usuário tivesse o *Python* instalado em sua máquina.

Após a etapa de estudos e divisão de tarefas, deu-se início ao desenvolvimento do projeto, que ocorreu de maneira contínua, por meio de diferentes versões do protótipo:

Versão 1 – Utilizou-se o *PyAutoGUI* para capturar imagens da tela e o *Tesseract* para identificar os textos. Os textos identificados eram exibidos diretamente no terminal. Essa versão, por ser apenas um teste das ferramentas, não possuía interface gráfica e restringia-se ao ambiente de desenvolvimento *Visual Studio Code*.

Versão 2 – Foi implementada a integração com o *Google Translate*®, permitindo a tradução para o inglês dos textos extraídos. Apesar do avanço na funcionalidade, essa versão também não contou com interface gráfica.

Versão 3 – Descobriu-se o *Tesseract.js*, uma alternativa em *JavaScript* mais adequada para aplicações *web*. Nesse estágio, o *PyAutoGUI* foi substituído pelo recurso *captureVisibleTab* e o *Google Translate*® pelo *LibreTranslate*. Foi criada uma interface simples que permitia selecionar os idiomas de origem e destino, além de iniciar a tradução. Contudo, a extensão apresentou dificuldades técnicas, pois não permitia conexão direta com o *Tesseract.js* hospedado na web, e a tentativa de incluir os arquivos no diretório da extensão não obteve sucesso. Diante disso, optou-se por retomar o uso do *Python*.

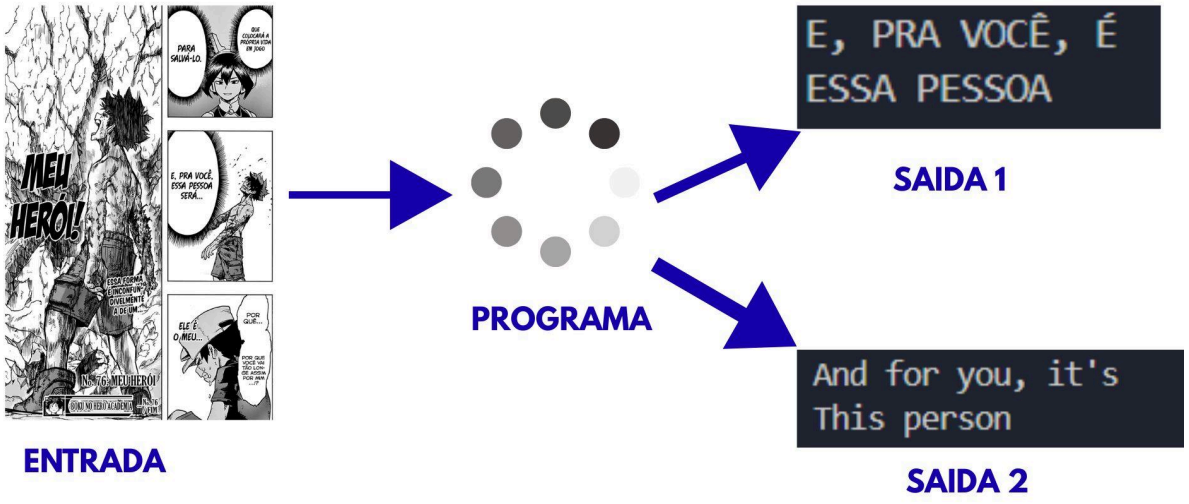
Versão 4 – Retomou-se a utilização do *Flask*, agora usado para a interação entre *Python* e *JavaScript*, permitindo testes em ambiente web, desde que o *Python* estivesse instalado na máquina. O *JavaScript* ficou responsável pela captura de tela e coleta dos idiomas, enviando os dados ao *Python*, que processava a imagem e realizava a tradução. A interface foi aprimorada, permitindo a escolha entre cinco idiomas e exibindo, no próprio popup da extensão, tanto a imagem quanto o texto traduzido.

Durante esta versão, dois desafios foram solucionados: (I) a diferença entre os formatos de códigos de idioma, o *Tesseract* utiliza siglas de dois caracteres (ex.: “en”), enquanto o *Google Translate*® emprega três (ex.: “eng”) — resolvido por meio de uma função de conversão; e (II) o envio de dados entre *JavaScript* e *Python*, feito por *IP* local, o que exigia atualização manual em caso de alteração de *IP*.

RESULTADOS E DISCUSSÃO

FIGURA 1.

Identificação de texto por imagem e tradução



A imagem de entrada é capturada pelo programa, e o *Tesseract* processa essa imagem para extrair o texto identificado. Após concluir a extração, na saída 1, o resultado é transmitido para o terminal. Na saída 2, antes de ser exibido no terminal, o texto é traduzido pela *API* do *Google Translate*®

A versão 3 não apresenta saída, pois não obteve sucesso em sua execução em razão de problemas não solucionáveis na conexão com o *Tesseract.js*.

A versão 4 passou a possuir uma interface para capturar a tela do navegador. Além disso, o texto extraído e traduzido substitui o texto original da imagem, que é exibida ao usuário. No entanto, isso causou um erro na tradução, que foi feita palavra por palavra, resultando em frases como “eu amor você” em vez de “eu amo você”.

Ao longo da pesquisa, identificou-se um alto índice de perda de palavras. Para solucionar esse problema, desenvolveu-se um código que processa a imagem, convertendo-a para preto e branco e reduzindo o ruído. Além disso, aplicou-se um realce de bordas, que ajuda o Tesseract a distinguir melhor os contornos dos caracteres, aumentando a quantidade de caracteres reconhecidos corretamente e diminuindo as chances de interpretar desenhos como texto.

FIGURA 2.

Aplicação de filtro antes da identificação do texto

CONCLUSÕES

A versão atual da extensão funciona hospedada localmente. Ela é capaz de capturar prints de múltiplas páginas de mangá, devolvendo as imagens com o texto traduzido e substituindo o texto original. Isso representa um grande avanço para a pesquisa, pois, com esta versão, já dispomos de uma forma funcional da extensão, alcançando a estrutura final do código. As versões seguintes se concentraram apenas em otimizar e/ou aprimorar os processos já existentes.

Atualmente, uma das melhorias no desenvolvimento é a aplicação de filtros para melhorar o desempenho do Tesseract e a correção dos erros de tradução causados pelo processo de substituição do texto original.

REFERÊNCIAS

ATTEKITA, Carol. **GIT para programadores INICIANTES | Introdução e fundamentos (O que são GIT e GITHUB?)**. YouTube, 2022. Disponível em: https://youtu.be/P9xXbEhqhqA?si=Yn45iKNn5bxR81Cm . Acesso em: 23 fev. 2025.

DÜTRA, Grazy - It-oficial. **Como trabalhar com Git e GitHub no Visual Studio - 2021**. YouTube, 2021. Disponível em: https://youtu.be/2aw4TRR5Xfw?si=orP_0COQv5cRLjXc . Acesso em: 23 fev. 2025.

GOOGLE. *Extensões|Chrome for Developers*. Disponível em: https://developer.chrome.com/docs/extensions/get-started?hl=pt-br. Acesso em: 14 mar. 2025.

HASHTAG PROGRAMAÇÃO. **Pillow Python - Processamento de Imagens no Python [Aula Completa]**. YouTube, 2022. Disponível em: https://youtu.be/6ceYcae4Ois?si=4Cs0L3PxH9mSHLKz . Acesso em: 1 mar. 2025.

HASHTAG PROGRAMAÇÃO. **Como Ler Textos Dentro de Imagens com Python [Introdução ao OpenCV e Tesseract]**. YouTube, 2021. Disponível em: https://youtu.be/Wx3SyNwZtsA?si=L5FZE0TFIlFcb3BL . Acesso em: 1 mar. 2025.

JSON.ORG. *JSON*, [s.d.]. Disponível em: **https://www.json.org/json-en.html.** Acesso em: 1 mar. 2025.

OBOLIÉNSKAIA**,** Iulia Leonárdovna; SALES, Denise Regina de; GARAY, Rodrigo Garcia. **A tradução literária no diálogo entre culturas e literaturas.** *Cadernos de Tradução*, Porto Alegre, RS: Nesp, 2016, p. 179-200. Disponível em: http://hdl.handle.net/10183/249198; Acesso em:1 mar. 2025

PROGRAMAÇÃO NA PRÁTICA. **Alternativa ao Heroku! Deploy render.com (Python e Flask)**. YouTube, 2022. Disponível em: https://youtube.com/watch?v=n_9FQVcqf4I&si=PkLcIDvmUrnbXZJJ . Acesso em: 1 mar. 2025.

SANTANA, Giovana Carlos. *O(s) fã(s) da cultura pop japonesa e a prática de scanlation no Brasil.* 2011. Dissertação (Mestrado em Comunicação e Linguagens) — Universidade Tuiuti do Paraná, Curitiba, 2011. Disponível em: https://www.academia.edu/1181174/_Dissertação_de_mestrado_O_s_fã_s_da_cultura_pop_japonesa_e_a_prática_de_scanlation_no_Brasil. Acesso em: 14 mai. 2025.

TESSERACT-OCR. *ImproveQuality*. Tessdoc: GitHub. Disponível em: https://github.com/tesseract-ocr/tessdoc/blob/main/ImproveQuality.md. Acesso em: 24 jul. 2025.

VICO, Ronan. **Como transformar imagem em texto usando OCR em Python com OpenCV , Tesseract reconhecendo caracteres.** YouTube, 2022. Disponível em: https://youtu.be/GMqFZ7f0dy4?si=XMeufCV5t3kxFask . Acesso em: 1 mar. 2025.