WARD: Uma Aplicação de Reconhecimento e Descrição de Imagens para Deficientes Visuais

Lucas Izidorio Almeida[1], Rodrigo Gonçalves Ribeiro[1], Maria Inês Lage De Paula[1], Lesandro Ponciano dos Santos[1]

[1]PUC Minas em Contagem
Bacharelado em Sistemas de Informação

izidorio.lucas7@gmail.com, roxdrigo1883@gmail.com, milpaula@pucminas.br, lesandrop@pucminas.br

Resumo. *Muitos brasileiros são deficientes visuais. Em um mundo cada vez mais tecnológico, a inclusão de pessoas com problemas na visão se tornou um tópico importante a ser discutido. A experiência do usuário deve ser pensada para que todos possam utilizar os sistemas de informação, principalmente os sistemas em que os usuários são criadores de conteúdo, como em redes sociais. No contexto de imagens geradas e publicadas pelos usuários do sistema, a geração de descrição do conteúdo das imagens se torna ainda mais desafiadora. Este trabalho tem como objetivo desenvolver uma extensão para browser capaz de identificar imagens na Web e criar descrições acessíveis, para que os usuários com problemas na visão tenham acesso à informação sobre o conteúdo que é exibido dentro da imagem.*

1 INTRODUÇÃO

A deficiência visual é um problema que acomete milhares de brasileiros. Segundo as informações do último censo demográfico realizado pelo Instituto Brasileiro de Geografia e Estatística (IBGE), o Brasil possuía, em 2010, cerca de 6,5 milhões de pessoas deficientes visuais, sendo 528 mil dessas o número aproximado de indivíduos com ausência total da visão (IBGE, 2010). Tais pessoas, entretanto, acabam enfrentando dificuldades ao utilizar os sistemas que estão disponíveis na Web, principalmente quando se trata de redes sociais. Apesar de existirem funcionalidades de acessibilidade em algumas redes sociais, a maior parte do conteúdo visual publicado é feito pelos usuários e fica sem uma descrição acessível.

Existem softwares de acessibilidade para sistemas operacionais que auxiliam no uso de computadores pelo público que possui alguma deficiência visual. Dentre tais softwares, podemos citar ferramentas que fazem a leitura da tela para o usuário, como o *Non Visual Desktop Access* (NVDA)[1]. Entretanto, os leitores de tela não permitem que o usuário compreenda o contexto completo quando o conteúdo apresenta imagens. Esse problema pode ser contornado por meio de descrições das imagens, que podem ser providas pelo desenvolvedor do site durante sua criação.

Apesar da existência do atributo *alt* (texto alternativo) em páginas Web, muitos sites não o utilizam nas imagens e, quando falamos de redes sociais, o assunto fica ainda mais complexo. A maior parte dos usuários não sabem como criar uma descrição acessível ou simplesmente não pensam no público com deficiência na hora de fazer uma postagem. Ao pensar na experiência do usuário durante a utilização dos sistemas de informação, é necessário considerar quais são os requisitos de acessibilidade para aumentar o público da

[1] "NV Access." https://www.nvaccess.org/. Acessado em 12 set.. 2021.

aplicação. Mas e quando os desenvolvedores não são os únicos responsáveis pela acessibilidade? Como melhorar a experiência dos usuários com deficiência visual?

Grande parte do conteúdo das redes sociais mais utilizadas atualmente atrela fortemente a descrição (ou o texto da postagem) com alguma imagem associada (ou até mesmo mais de uma imagem). Para tornar a experiência de utilização das aplicações com mídia visual acessível é necessário prover descrições alternativas para as imagens.

O objetivo geral deste trabalho é desenvolver um sistema para funcionar em conjunto com o *browser* do usuário, denominado Web-Assistente de Reconhecimento Digital de Imagens (WARD), que seja capaz de acrescentar textos alternativos às imagens na tela, tornando-as acessíveis às pessoas que possuem alguma deficiência visual.

Para atingir o objetivo principal do trabalho é necessário cumprir alguns objetivos específicos, entre eles:

- Analisar as ferramentas atuais de acessibilidade para pessoas com deficiência visual e levantar as principais funcionalidades em comum entre elas.
- Criar uma extensão para *browser* capaz de ler e modificar o arquivo de Linguagem de Marcação de Hipertexto (HTML, do inglês *HyperText Markup Language*) de uma página Web.
- Criar uma Interface de Programação de Aplicação (API, do inglês *Application Programming Interface*) para ser consumida pela extensão capaz de gerar uma descrição para uma determinada imagem de entrada.
- Criar um algoritmo de descrição de imagens capaz de gerar descrições acessíveis.

A Seção 2 deste documento contém os fundamentos teóricos utilizados no trabalho. Na Seção 3, são apresentados os trabalhos relacionados. Em seguida, apresenta-se a metodologia utilizada no projeto, na Seção 4. As referências bibliográficas utilizadas estão na Seção 5.

2 REFERENCIAL TEÓRICO

Nesta seção são descritos os principais conceitos que estão relacionados a sistemas de acessibilidade e técnicas que serão utilizadas no trabalho.

2.1 ACESSIBILIDADE PARA DEFICIENTES VISUAIS

Quando pensamos em acessibilidade na Web, não se trata de um assunto completamente novo. Na verdade, é uma questão que vem sendo estudada há muitos anos, e uma das principais diretrizes a ser considerada nesse assunto é a *Web Content Accessibility Guidelines* (WCAG), um conjunto de recomendações a respeito de acessibilidade na Web que foi publicada em maio de 1999 pela *World Wide Web Consortium* (W3C), principal organização de padronização da internet.

Dentre as recomendações da WCAG, uma das principais é referente aos textos alternativos para conteúdo não-textual. De acordo com a W3C, é necessário que os textos alternativos estejam ligados através do código-fonte ao conteúdo não-textual de forma que as

tecnologias assistivas sejam capazes de lê-los e usá-los conforme o necessário (W3C, 2018, tradução nossa).[2]

Textos alternativos podem ser inseridos em imagens na página HTML através do atributo *alt*, e geralmente é esse o atributo analisado pelos leitores de tela para descrever a imagem para o usuário. Entretanto, nem sempre fica nas mãos do programador prover uma descrição para a imagem. Em redes sociais, onde o conteúdo é publicado por outros usuários e não pelo próprio programador, por exemplo, a falta de textos alternativos se torna um problema comum. A partir disso, existem algumas soluções que já foram criadas na Web.

O Twitter[3] e o Facebook[4] disponibilizam uma funcionalidade para que os usuários adicionem uma descrição de texto alternativo às imagens anexadas na postagem. Além disso, também existe um movimento conhecido como "#PraCegoVer"[5] (ou "#PraTodosVerem") nas redes sociais que utilizam *hashtags* que consiste em trazer uma descrição textual para postagens que têm imagens anexadas, provida pelo próprio autor da publicação. As publicações feitas utilizando alguma das *hashtags* são agrupadas dentro da rede social.

Sacramento *et al.* (2020) pontuam que, apesar das funcionalidades existentes nessas plataformas, muitos usuários não as usam por não conseguir encontrá-las ou por não saber o que escrever. Além disso, também falam sobre a falta dessas funcionalidades em outros sites e redes sociais, como o WhatsApp e o YouTube. Gleason *et al.* (2019) reforçam essa ideia em sua pesquisa, na qual em uma amostra de 9,22 milhões de tuítes, 1,09 milhões possuíam fotos anexadas e apenas 0,1% destes possuíam textos alternativos criados com a funcionalidade provida pela rede social.

2.2 FERRAMENTAS ASSISTIVAS, IDENTIFICAÇÃO E DESCRIÇÃO DE IMAGENS

Para atingir os objetivos do trabalho, é necessário utilizar tecnologias e técnicas já existentes para criar textos alternativos. Do ponto de vista técnico, a identificação de imagens se dá através da leitura do documento HTML de uma página Web em busca de imagens (pela tag *img*), enquanto a descrição de imagens está relacionada ao conceito de visão computacional e Inteligências Artificiais (IAs) capazes de identificar quais são os elementos presentes em uma determinada imagem.

O trabalho de Gleason *et al.* (2020) apresenta uma extensão para *browser* que identifica imagens no conteúdo HTML e extrai o *Uniform Resource Locator* (URL) para fazer uma requisição no servidor *backend*, sem a necessidade de uma ação do usuário. A resposta da requisição é o texto alternativo adequado e, após recebê-la, a extensão altera o atributo *alt* na página para possibilitar a leitura da descrição pelas ferramentas assistivas. Gleason *et al.* (2019) também especificam processos de descrição de imagens por pessoas e por robôs. Podemos citar os métodos de descrição de cores, determinação da importância de objetos na

---

[2] No original: *In order for people with disabilities to be able to use this text - the text must be "programmatically determinable." This means that the text must be able to be read and used by the assistive technologies (and the accessibility features in browsers) that people with disabilities use.*
[3] "Twitter. It's what's happening.." https://twitter.com/. Acessado em 12 set.. 2021.
[4] "Facebook - Log In or Sign Up." https://facebook.com/. Acessado em 12 set.. 2021.
[5] "Pra Cego Ver - Home | Facebook." https://www.facebook.com/PraCegoVer/. Acessado em 20 abr. 2021.

imagem, fundo e outros conteúdos que o usuário talvez não seja capaz de perceber somente através do texto original (tradução nossa).[6]

2.3 ALGORITMOS E TECNOLOGIAS DE DESCRIÇÃO DE IMAGENS

Existem diversas IAs disponíveis atualmente que fazem a identificação do conteúdo de uma imagem. A principal ferramenta que é utilizada neste trabalho para fazer o reconhecimento de imagens é o Google Cloud Vision API[7].

A criação de uma extensão de browser exige a utilização de HTML, Folha de Estilo em Cascatas (CSS, do inglês *Cascading Style Sheets*) e JavaScript. A lógica necessária para a extensão é bem simples, pois só é necessário fazer a manipulação do HTML da página Web que está sendo acessada pelo usuário. A parte principal da aplicação é feita através de um servidor *backend* utilizando Node.js[8].

Além disso, também existe o desafio da tradução da descrição do inglês para o português, já que a aplicação construída neste trabalho é voltada para o público brasileiro e todas as ferramentas utilizadas utilizam o inglês como língua base. Para fazer a tradução, a aplicação desenvolvida neste trabalho utiliza o Google Cloud Translation API[9].

3 TRABALHOS RELACIONADOS

Gleason *et al.* (2020) propôs uma extensão para *browser* capaz de ler e editar o HTML do Twitter. Sempre que uma nova imagem é carregada, a extensão extrai a URL da imagem e faz uma requisição no servidor *backend* da aplicação, onde a imagem é processada para gerar o texto alternativo adequado. A resposta da requisição com o texto alternativo é enviada de volta para a extensão, que edita o HTML do Twitter novamente para inserir o texto na propriedade *alt* da imagem processada. A extensão Twitter A11Y, no entanto, funciona apenas no Twitter, diferente da proposta deste trabalho.

Guinness *et al.* (2018) desenvolveu um *plugin* para *browser* que funciona de maneira semelhante ao Twitter A11Y, identificando as imagens na página HTML para inserir textos alternativos na propriedade *alt*. Todavia, o Caption Crawler utiliza o mecanismo de pesquisa reversa do Google para encontrar correspondências da imagem e reutilizar descrições que foram usadas anteriormente. O trabalho não é capaz de inserir descrições em qualquer imagem, pois é necessário que ela tenha sido descrita previamente por outra pessoa ou mecanismo, diferente da proposta deste trabalho.

Bodi *et al.* (2021) apresentam uma ferramenta para auxiliar deficientes visuais a assistirem vídeos com dois robôs diferentes: *NarrationBot*, um robô que descreve cenas nos vídeos a cada nova cena, e *InfoBot*, um robô que responde perguntas específicas sobre a cena

---

[6] No original: *One participant mentioned describing the colors that appeared in the image. Others described determining the importance of objects, background, and other content in the image that the reader may not be able to ascertain from the main text of the tweet.*
[7] "Vision AI | Derive Image Insights via ML ...." https://cloud.google.com/vision. Acessado em 20 abr. 2021.
[8] "Node.js." https://nodejs.org/. Acessado em 20 abr. 2021.
[9] "Translation API Basic - Google Cloud." https://cloud.google.com/translate. Acessado em 12 set.. 2021.

quando solicitado. Para fazer a descrição de cenas, a aplicação desenvolvida pelos autores escolhe o melhor frame do vídeo entre os frames repetidos onde a cena muda e analisa os elementos mais importantes da imagem, processo semelhante ao que é executado pela aplicação desenvolvida neste trabalho.

4 METODOLOGIA

Este trabalho tem como objetivo desenvolver uma extensão para o *browser* Google Chrome que seja capaz de ler o HTML das páginas acessadas pelo usuário e identificar imagens, criar descrições alternativas para elas através da API de descrição de imagens e editar o HTML da página inserindo as descrições na propriedade *alt* das imagens.

4.1 ETAPAS DO TRABALHO

Este trabalho será dividido nas seguintes etapas:

- Levantamento de requisitos
- Definição da arquitetura e modelos do sistema
- Desenvolvimento e testes da aplicação

Ao final do trabalho será feita a apresentação dos resultados obtidos com a pesquisa, análise dos resultados e conclusão.

4.2 LEVANTAMENTO DE REQUISITOS

Com o objetivo de entender as funcionalidades necessárias para atingir os objetivos deste trabalho, foram definidos os requisitos funcionais e não-funcionais da aplicação, representados a seguir nas tabelas 1 e 2, respectivamente.

ID	Requisito
RF-01	O sistema deve ser capaz de obter as *tags* HTML de imagem dos sites visitados pelo usuário
RF-02	O sistema deve gerar uma descrição para uma imagem captada da tela
RF-03	O sistema deve alterar a propriedade *alt* da imagem com a descrição gerada
RF-04	O sistema deve permitir a ativação/desativação da extensão através de atalhos no teclado

Tabela 1. Requisitos funcionais do projeto

ID	Requisito
RNF-01	As descrições devem ser geradas com um tempo máximo de 15 segundos
RNF-02	O sistema deve identificar as imagens na página HTML automaticamente, sem a necessidade de ações do usuário
RNF-03	A extensão será desenvolvida com as tecnologias JavaScript e HTML
RNF-04	A API será desenvolvida com a tecnologia JavaScript (Node.js)
RNF-05	O sistema deve ser compatível com os leitores de tela NVDA e JAWS
RNF-06	O sistema deve gerar log de erros para análises de problemas ou *bugs*
RNF-07	O sistema deverá ser compatível com o navegador Google Chrome

Tabela 2. Requisitos não-funcionais do projeto

4.3 ARQUITETURA E TECNOLOGIAS

O desenvolvimento da aplicação resultante deste trabalho será dividido em duas partes principais: a extensão para *browser* (módulo I) e a API *back-end* (módulo II). O diagrama apresentado na figura 1 representa como as partes devem interagir entre si para que a aplicação funcione corretamente.

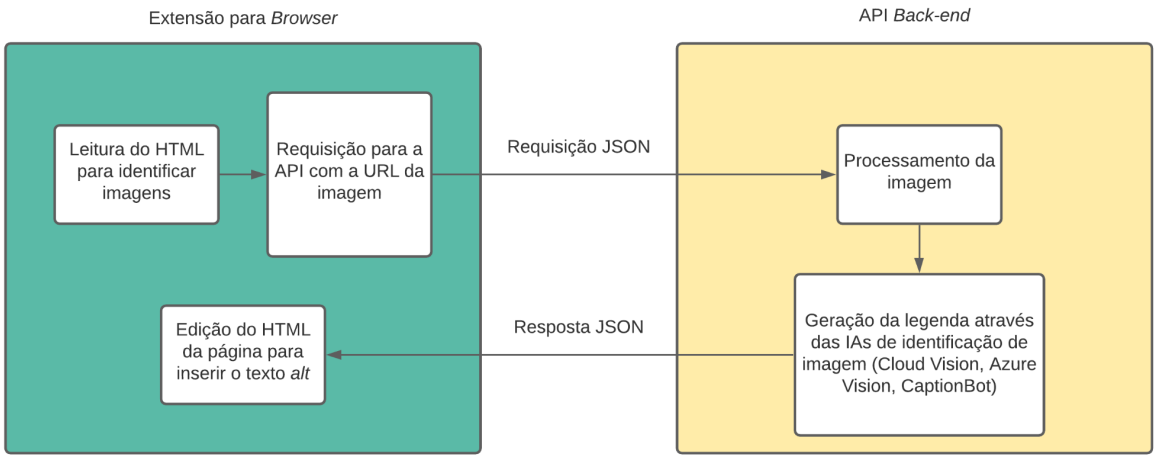

Figura 1. Diagrama de comunicação das partes da aplicação

Para desenvolver as partes será necessário utilizar algumas tecnologias e ferramentas já existentes no mercado. A tabela 3 especifica quais são essas tecnologias para cada parte da aplicação.

Módulo	Ferramenta
I	Linguagem de programação JavaScript
I	Linguagem de marcação HTML
I	Biblioteca Axios para execução de requisições REST
II	Linguagem de programação JavaScript
II	Framework Node.js + Express para desenvolvimento da API
II	Bibliotecas de reconhecimento de imagens: Google Cloud Vision API, CaptionBot API e Azure Computer Vision API

Tabela 3. Lista de ferramentas necessárias

Para desenvolver extensões para o *browser* Google Chrome é obrigatório o uso das tecnologias JavaScript e HTML. O framework Node.js foi escolhido devido à praticidade para construir aplicações escaláveis com a linguagem JavaScript, que já seria utilizada na extensão.

O módulo I será desenvolvido com a arquitetura MVC (Model-View-Controller), para facilitar o controle da aplicação e entregar o necessário para o desenvolvimento de uma extensão para o navegador Google Chrome. Já o módulo II será construído com a arquitetura padrão do Node.js com Express, constituído da separação dos arquivos por função: *config*, *controllers*, *models* e *routes*.

4.4 DESENVOLVIMENTO E TESTES

Os módulos I e II serão desenvolvidos paralelamente. Como o módulo I depende do módulo II para funcionar por completo, as requisições ficarão apenas estruturadas no código fonte. Dessa forma, a extensão poderá ser testada sem a necessidade de conexão com a API. O módulo II é independente da extensão, mas é necessário integrá-lo com as IAs de reconhecimento de imagens.

A API CaptionBot está disponível como biblioteca para JavaScript e as APIs do Google Cloud Vision e Azure Computer Vision podem ser chamadas através de requisições da aplicação Node.js.

Os testes da aplicação serão divididos em etapas. A primeira etapa é a de testes individuais nos módulos I e II, em que os requisitos funcionais da tabela 1 serão avaliados. Além dos requisitos funcionais, o funcionamento geral da aplicação também será validado através da primeira etapa de testes. Já a segunda etapa é a de testes de usabilidade com usuários.

Para a execução dos testes os usuários receberão acesso à extensão para utilizá-la durante a interação com sites e redes sociais. Os usuários também receberão acesso a um site que simula a experiência do Instagram com imagens selecionadas por nós e serão divididos em dois grupos:

- O grupo 1 terá acesso ao site sem modificações.

- O grupo 2 terá acesso ao site e à aplicação em funcionamento, fazendo com que as imagens exibidas no site tenham descrições alternativas.

5 CRONOGRAMA

O desenvolvimento do trabalho seguirá o cronograma apresentado na tabela abaixo.

Atividade	MAI	JUN	JUL	AGO	SET	OUT	NOV
Levantamento de requisitos							
Definição da arquitetura e modelos do sistema							
Desenvolvimento e testes da aplicação							
Conclusão							
Escrita do relatório							

Tabela 4. Cronograma de atividades

6 REFERÊNCIAS BIBLIOGRÁFICAS

Censo Demográfico de 2010. **Características gerais da população, religião e pessoas com deficiência**. Rio de Janeiro: IBGE, 2012.

GUIMARÃES, Ana Paula Nunes; TAVARES, Tatiana Aires. Avaliação de Interfaces de Usuário voltada à Acessibilidade em Dispositivos Móveis: Boas práticas para experiência de usuário. In: WORKSHOP DE TESES E DISSERTAÇÕES - SIMPÓSIO BRASILEIRO DE SISTEMAS MULTIMÍDIA E WEB (WEBMEDIA), 2014, João Pessoa. Porto Alegre: Sociedade Brasileira de Computação, 2014. p. 22-29. ISSN 2596-1683.

SACRAMENTO C. NARDI L, FERREIRA S. B. L., MARQUES J. M. S.. #PraCegoVer: Investigating the description of visual content in Brazilian online social media. In XIX Brazilian Symposium on Human Factors in Computing Systems (IHC ’20), October 26–30, 2020, Diamantina, Brazil. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3424953.3426489

GLEASON C., CARRINGTON P., CASSIDY C., MORRIS M.R., KITANI K. M. e BIGHAM J.P.. “It’s almost like they’re trying to hide it": How User-Provided Image Descriptions Have Failed to Make Twitter Accessible. In Proceedings of the 2019 World Wide Web Conference (WWW ’19), May 13–17, 2019, San Francisco, CA, USA. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313605

GLEASON C., PAVEL A., MCCAMEY E., LOW C., CARRINGTON P., KITANI K. M. e BIGHAM J.P.. Twitter A11y: A Browser Extension to Make Twitter Images Accessible. In Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20). Association for Computing Machinery, New York, NY, USA, 1–12. DOI:https://doi.org/10.1145/3313831.3376728

GUINNESS D., CUTRELL E. e MORRIS M. R.. Caption Crawler: Enabling Reusable Alternative Text Descriptions using Reverse Image Search. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18). Association for Computing Machinery, New York, NY, USA, Paper 518, 1–11.

BODI A., FAZLI P., IHORN S., SIU Y., SCOTT A. T., NARINS L., KANT Y., DAS A. e YOON I.. Automated Video Description for Blind and Low Vision Users. Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems. Association for Computing Machinery, New York, NY, USA, Article 230, 1–7. DOI:https://doi.org/10.1145/3411763.3451810