Uma Extensão chrome que filtra imagens usando redes neurais convolucionais

1st Samuel Cavalcanti
Escola de Ciências e Tecnologia
Universidade Federal do Rio Grande do Norte (UFRN)
Natal, Brasil
scavalcanti111@gmail.com

2nd Agostinho De Medeiros Brito Junior
Departamento de Engenharia de Computação e Automação (DCA)
Universidade Federal do Rio Grande do Norte (UFRN)
Natal, Brasil
agostinhobritojr@gmail.com

***Resumo*—Resumo da Pesquisa**

I. Introdução

Extensões para navegadores chrome são programas orientados a eventos usados para modificar ou melhorar a experiencia de navegação do usuário. Eventos são gatilhos do navegador que são disparados quando o usuário interage com o navegador, como o abrir ou fechar de uma página, remover ou adicionar uma página aos favoritos. Extensões são capazes monitorar esses eventos e reagir a eles citar [developer.chrome.com].

Esse trabalho é a explicação do funcionamento de uma extensão para o navegadores baseados no chrome que visa filtrar imagens caso ela pertença a uma classe que o usuário não queria ver, para isso é utilizado uma rede neural convolucional para classificar as imagens. Uma rede neural convolucional é uma classe de rede neural artificial que vem sendo aplicada com sucesso no processamento e análise de imagens digitais. A rede neural convolucional tira proveito das cores. Utilizar imagens RGB em vez de escala de cinza no treinamento, pode ocasionar em uma melhora significativa no resultado [1] [2].

O desenvolvimento dessa aplicação web é aberto e pode ser encontrado no github: *https://github.com/samuel-cavalcanti/an-chrome-extension*. O restante do artigo está dividido nas sessões: a sessão II explica o funcionamento de uma extensão chrome, a sessão III explica como um modelo de rede neural convolucional está sendo utilizado pelo sistema, a sessão V explica o funcionamento da extensão desenvolvida, a sessão VI mostra os testes e resultados realizados utilizando 2 modelos de redes neurais convolucionais e a ultima sessão concluí o artigo.

II. O funcionamento de uma Extensão chrome

Extensões para navegadores baseados no projeto chromium são construídos usando os componentes: background scripts, content scripts, options page, elementos de interface gráfica e um arquivo chamado manifest.json onde fica salvo os metadados da extensão. Para construir uma extensão não é necessário utilizar todos os componentes, mas nas extensão desenvolvida foi utilizado todos os componentes citar [developer.chrome.com]

A. manifest.json

Toda a extensão chrome possui esse arquivo, é nele que é informado ao navegador todas as permissões que a extensão necessita, nome da extensão, uma breve descrição da extensão, quais componentes serão utilizados nessa extensão e seus respectivos arquivos e outros metadados. citar [developer.chrome.com]

B. Elementos de interface gráfica

Toda Extensão pode possuir uma interação com usuário, essa interação acontece ao usuário clicar no ícone da extensão, dependendo da implementação o ato de clicar é toda interação com o usuário que aplicação pode ter ou que é mais comum é abrir uma página HTML com todas as informações sua aplicação. É possível customizar qual página HTML será exibida dependendo da página atual do usuário ou até mesmo impedir do usuário de acessar as funcionalidades da extensão. citar [developer.chrome.com]

C. Options page

Caso necessite que o usuário modifique ou customize o comportamento da extensão o desenvolvedor pode implementar uma página HTML listando todas as funcionalidades. Para acessa as opções o usuário precisa ir em **chrome://extensions**, localizar a extensão desejada, clicar em **Details** e clicar em **Extension options**. citar [developer.chrome.com]

D. Content scripts

Content scripts são arquivos que são executados em páginas web durante a navegação, esse arquivos são capazes de ler, modificar e passar informações da página atual para extensão, esses arquivos são executados isoladamente não conflitando entre si e nem com outros scripts presentes na página web apesar deles terem acesso ao mesmo Document Object Model (DOM). citar [developer.chrome.com]

E. background scripts

Como dito uma Extensão chrome é um programa orientado a eventos e o monitor desses eventos é o backgroud script, ele é inicializado sempre que a extensão for instalada, atualizada ou quando acontece um evento a qual ele esteja monitorando como: receber uma mensagem de outro componente ou uma determinada página começará a ser carregada. O background

script só fica ativo quando tiver executando alguma ação, assim que todas as ações forem finalizadas o script também é finalizado. citar [developer.chrome.com]

III. Rede neural convolucional

Uma Rede Neural Convolucional (ou Convolutional Neural Network - CNN) é uma variação das redes de Perceptrons de Múltiplas Camadas, tendo sido inspirada no processo biológico de processamentos de dados visuais. De maneira semelhante aos processos tradicionais de visão computacional, uma CNN é capaz de aplicar filtros em dados visuais, mantendo a relação de vizinhança entre os pixels da imagem ao longo do processamento da rede [3].

IV. Metodologia

A intenção desse trabalho era de desenvolver uma extensão chrome possui as seguintes funcionalidades:

- lista todos os modelos de CNNs do *https://tfhub.dev/* dado que o modelo serve para classificação de imagens e possui compatibilidade com tensorflowJS. O usuário pode selecionar e usar um desses modelos
- Exibe as classes que o modelo é capaz de classificar e permite o usuário selecionar qual classes ele quer ver ou não.
- Filtra as imagens dado que o usuário selecionou que não quer ver determinada classe de imagens e o modelo classificou como sendo dessa classe.
- carregar um modelo customizado localmente.

Apesar da aplicação cumprir com todas elas, por ser uma extensão foi escolhido o não armazenamento de modelos customizados uma vez que demanda o armazenamento local de um grande volume de dados e não foi encontrado suporte para isso.

V. Funcionamento da extensão chrome proposta

O objetivo da aplicação é monitorar todas as imagens recebidas pelo navegador e não exibir as imagens que provavelmente o usuário não deseja visualizar, então o primeiro passo para resolver o problema foi dividi-lo em problemas menores onde cada interface pudesse se especializar em e tentar prover uma boa solução. Então foi proposto as seguintes interfaces: gerenciador de classificadores, classificador, interface gráfica, observador de página, navegador-comunicador. Para o desenvolvimento dessa aplicação foi utilizado dois frameworks, tensorflow.js e Angular framework na versão 10.0. Angular é um framework JavasScript de código aberto com o objetivo de criar modernas aplicações web, mobile e desktop [4], tensorflow.js é uma biblioteca para construção e execução de algoritmos de aprendizado de máquina em JavaScript [5]. Para facilitar o entendimento foi criado sub-sessões cada uma explicando uma parte da solução proposta. É aconselhado ver os dois vídeos demostração antes de continuar a leitura, o segundo vídeo possuí conteúdo pornográfico explicito, nele é demonstrado como utilizar uma CNN customizada para filtragem de imagens, a CNN utilizada foi treinada para filtrar conteúdo pornográfico. Primeiro vídeo: *https://vimeo.com/454629223*, segundo vídeo: *https://vimeo.com/454627349*

A. navegador-comunicador

Com o objetivo de abstrair o navegador surgiu essa interface. O navegador-comunicador é responsável para realizar qualquer chamada da API do navegador chrome, é através dele que diferentes componentes da extensão se comunicam como a interface gráfica com o background script ou backgroud script com o content script, options page com background script, além também de ser responsável por armazenar informações no navegador e recuperar arquivos que foram instalados juntos com a extensão. Uma das principais motivações da criação dessa interface é isolar a funcionalidades presentes no navegador chrome, para caso houvesse um interesse de migrar ou adicionar suporte a outro navegador o código atual não precisaria ser modificado ou pouco modificado.

B. Observador de página

Essa interface é responsável por monitorar as páginas que o usuário está acessando e notificar para outros componentes todas as urls de imagens encontradas e espera ser notificado se as imagens notificadas podem ou não ser visualizadas, esse componente se localiza exclusivamente no content script da nossa aplicação, uma vez que é no content script que se tem acesso as páginas acessadas pelo usuário.

C. Interface gráfica

Para o desenvolvimento dessa aplicação foi construído uma página HTML que é exibida assim que o usuário clica no ícone da extensão. Essa página se comporta como uma single page application. Uma single page web application é exatamente o que o nome diz, uma aplicação web feita em JavaScript que necessita o carregamento de uma única página [6], no entanto essa página pode ter diferentes sub-páginas. Através dessa página HTML o usuário pode selecionar um modelo do *https://tfhub.dev/* e também marcar quais classes de imagens ele deseja ou não visualizar, por padrão ao escolher um modelo você pode visualizar todas as classes de imagens. Também é nessa interface que o usuário pode inserir um modelo customizado porém, como essa opção é para usuários avançados foi decidido coloca-lo no options page. Modelos customizados não são salvos, tendo que inseri-lo novamente a cada inicialização do navegador.

D. Classificador

Essa interface é exatamente o que o nome dele diz, ele recebe uma imagem e classifica se a imagem pode ou não ser exibida. Apesar de uma CNN poder classificar uma imagem em enumeras classes diferentes a decisão final é se essa imagem pertence ao conjunto de imagens que podem ser exibidas ou não. Apos realizar a classificação, ela notifica aos outras interface a sua decisão.

E. gerenciador de classificadores

Como surgiu a necessidade de introduzir diferentes formas de selecionar uma CNN, então foi feita essa interface que é responsável por carregar CNN e suas configurações adicionais como todas as classes que ela é capaz de classificar.

Atualmente só é possível carregar redes de origem do tfhub e localmente, no entanto se surgir a necessidade de adicionar outra origem, é nessa interface que se deve adicionar essa funcionalidade.

VI. Testes e Resultados

ideias para avaliar a extensão:

- mensurar o tempo da extensão de exibir as imagens
- mensurar o tempo de resposta das redes, isso varia com cada rede
- dizer as limitações da extensão, gifs, vídeos, lazing loading e outras variáveis que existem mas não tenho conhecimento devido poucos testes
- dizer que foi pouco testada
- devo automatizar os testes ? muito trabalho pra pouco resultado ?

VII. Conclusão

Referências

[1] R. d. P. Rosa *et al.*, "Método de classificação de pragas por meio de rede neural convolucional profunda," 2018.

[2] F. A. Wichmann, L. T. Sharpe, and K. R. Gegenfurtner, "The contributions of color to recognition memory for natural scenes." *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 28, no. 3, p. 509, 2002.

[3] A. C. G. Vargas, A. Paes, and C. N. Vasconcelos, "Um estudo sobre redes neurais convolucionais e sua aplicação em detecção de pedestres," in *Proceedings of the xxix conference on graphics, patterns and images*, vol. 1, no. 4, 2016.

[4] S. K. Kasagoni, *Building Modern Web Applications Using Angular*. Packt Publishing Ltd, 2017.

[5] D. Smilkov, N. Thorat, Y. Assogba, A. Yuan, N. Kreeger, P. Yu, K. Zhang, S. Cai, E. Nielsen, D. Soergel *et al.*, "Tensorflow. js: Machine learning for the web and beyond," *arXiv preprint arXiv:1901.05350*, 2019.

[6] D. Flanagan and G. M. Novak, "Java-script: The definitive guide," 1998.