COMPARATIVO DE ALGORITMOS COM E SEM A UTILIZAÇÃO DE REDES NEURAIS SOBRE RECONHECIMENTO DE PLACAS AUTOMOTIVAS

COMPARISON OF ALGORITHMS WITH AND WITHOUT THE USE OF NEURAL NETWORKS IN CAR LICENSE PLATE RECOGNITION

Aryan L. Ferrari[1], Danilo C. Sano[2], Jefferson A. R. Passerini[3]

[1]Faculdade de Tecnologia Prof. José Camargo – Fatec Jales, danilo.sano@fatec.sp.gov.br
[2]Faculdade de Tecnologia Prof. José Camargo – Fatec Jales, aryan.ferrari@fatec.sp.gov.br
[3]Faculdade de Tecnologia Prof. José Camargo – Fatec Jales, jefferson.passerini@fatec.sp.gov.br

Informação e Comunicação
Subárea: Processamento de Imagens e Visão Computacional

RESUMO

Atualmente no cenário de extrema globalização, nota-se que o crescimento da população afeta diretamente o crescimento do número de veículos em circulação, o que desencadeia problemas de controle de tráfego. Tendo isso em vista, o presente trabalho objetiva comparar resultados de duas diferentes formas para o reconhecimento de placas veiculares e leitura dos caracteres dessas placas por meios de algoritmos a partir da entrada de imagens. Assim, foram comparados dois métodos diferentes de visão computacional juntamente com diversos tipos de processamentos digitais para que, no final, seja determinado o melhor e mais eficiente dentre eles. Em uma abordagem utilizou-se apenas bibliotecas de reconhecimento visual, em outra abordagem utilizou-se um algoritmo de redes neurais convolucionais, para extrair a região da placa na imagem. A partir destas imagens aplicou-se métodos de extração de caracteres com OCR (*Optical Character Recognition)* em ambas as imagens extraídas de cada abordagem utilizada. Com os resultados obtidos, pode-se concluir que o método mais eficiente para a visão computacional encontrado foi com a utilização de redes neurais convolucionais quando comparado com a biblioteca de reconhecimento visual, pois ele apresenta um melhor retorno de placas reconhecidas e caracteres lidos com êxito.
Palavras-chave: visão computacional; processamento digital; redes neurais convolucionais; reconhecimento ótico de caracteres; identificação de placas automotivas.

ABSTRACT

Currently, in the scenario of extreme globalization, it is remarked that population growth directly affects the growth in the number of vehicles in circulation, which triggers traffic control problems. In view of this, the present paper aims to compare results of two different ways for the recognition of license plates and reading the characters of these plates, by means of algorithms from the input of images. Thus, two different methods of computer vision were compared, along with different types of digital processing, so that in the end the best and most efficient one could be determined. In one approach, only visual recognition libraries were used, in another approach, a convolutional neural network algorithm was used to extract the plate region in the image. From these images, character extraction methods with OCR (Optical Character Recognition) were applied to both images extracted from each used approach. With the obtained results, it can be concluded that the most efficient method for computer vision found was the use of convolutional neural networks when compared to the visual recognition library, as it presents a better return of recognized plates and successfully read characters.
Keywords: computer vision; digital processing; convolutional neural networks; optical character recognition; identification of automotive plates.

1 INTRODUÇÃO

Nas últimas décadas houve um considerável aumento populacional em todo o planeta. Segundo Salas (2020), *"pulamos de 3,5* bilhões *de habitantes em 1960, para aproximadamente 7,9 bilhões em 2020";* a partir disso, outros aspectos cresceram de maneira paralela e, considerando o grande aumento da utilização de tecnologias, criaram-se alguns problemas com essa enorme quantidade de dados e de movimentação no mundo todo. Além disso, de 1984 a 2019, o número de condomínios residenciais aumentou em aproximadamente 321%, gerando assim, demanda por serviços essenciais de portaria. Porém, nem sempre esse serviço se mostra eficiente, contendo falhas humanas em reconhecimento de pessoas ou veículos, impossibilitando assim a entrada ou saída de moradores ou visitantes no recinto.

Nesse contexto, empresas buscam soluções baseadas em reconhecimento remoto ou presenciais de forma tecnológica e automática, objetivando mais eficiência e maior segurança no controle de acesso. Com isso, essas empresas procuram diversas maneiras de se obter uma boa tecnologia capaz do reconhecimento e leitura de placas automotivas, e acabam se deparando com alguns métodos diferentes, sendo eles no geral, utilizando bibliotecas específicas para visão computacional, ou chegando até à utilização de redes neurais.

Assim, o objetivo deste trabalho é mostrar a diferença de resultado entre essas tecnologias, montar e comparar o funcionamento de cada algoritmo e mostrar alguns de seus pontos, com seus resultados, desempenhos e requisitos para um bom funcionamento. Dessa forma, a comparação entre todos os pontos irá exibir uma melhor noção de como cada tecnologia funciona.

Os benefícios que podem trazer esse projeto incluem o conhecimento sobre visão computacional e extração de caracteres, abrangendo a área de reconhecimento visual aprendida através da leitura de artigos e projetos futuros, mostrando ao leitor as diferenças entre algumas das tecnologias que existem no mercado atualmente.

A partir disso, este trabalho está organizado em três seções: no referencial teórico, que possui toda a base teórica para a construção deste artigo; em seguida, a metodologia, pela qual são mostradas as maneiras como foram utilizadas as ferramentas para se obter o resultado esperado; e a análise e discussão dos resultados, onde se mostra o resultado do artigo a partir dos estudos feitos e métodos aplicados.

2 REFERENCIAL TEÓRICO

Conforme Antonello (2017), o ser humano possui cinco sentidos: a audição, visão, olfato, tato e paladar. A partir dessa afirmação, pode-se assegurar que cada um deles tem sua importância dentro do funcionamento do organismo para ter o conhecimento do que acontece ao seu redor. Por exemplo, o sentido da visão nos concede uma leitura de como está o ambiente ao seu redor por meio de uma leitura gráfica feita pelos olhos e assim pode-se ter uma melhor noção de como está o espaço ao redor para identificar objetos. Dessa maneira, se fosse implementado esse sentido em um computador, ele seria capaz de processar e entender tudo o que acontece ao seu redor, ou pelo menos em sua área de foco.

A visão computacional se objetiva permitir que computadores compreendam imagens como um ser humano faz a partir dos seu sentido da visão. Para que essa técnica funcione são utilizados os mais variados sensores como câmeras RGB (*Red, Green, Blue*) que possibilitam a captura de imagens coloridas dos três principais tons de cores: RGB-D (R*ed, Green, Blue, Depth*) que permite a captura de imagens coloridas, porém o mesmo permite capturar imagens com o efeito de profundidade, câmeras com sensores infravermelho, etc., e também faz-se uso de diversas técnicas de processamento para que se possa extrair características das imagens, processá-las e obter-se o conhecimento desejado (MARENGONI; STRINGHINI, 2009).

A partir dessa etapa em que já se possui a fonte de imagem, o computador consegue processar essa entrada para o reconhecimento e/ou fazer manipulações de acordo com o que foi programado (ANTONELLO, 2017).

De acordo com Marengoni e Stringhini (2009), juntamente da visão computacional também existe o processamento digital, que possuem suas diferenças. A visão computacional tem como entrada uma imagem com intuito de, no final de seu processo, sair uma detecção ou interpretação de tudo o que se tinha como área de interesse. No outro lado, tem-se o processamento de imagens, que também possui a entrada de imagens, mas na saída, têm-se valores que foram obtidos pelo algoritmo para compor ou não outra imagem estabelecida. Segundo Antonello (2017, p. 6) “Visão computacional é a ciência e tecnologia das máquinas que enxergam. Ela desenvolve teoria e tecnologia para a construção de sistemas artificiais que obtêm informação de imagens ou quaisquer dados multidimensionais”.

Como exemplo de utilização da visão computacional e processamento digital, pode-se citar que no Brasil, de acordo com o IBGE (2022), há um número de 58.016.405 de automóveis circulantes em vias públicas, e a partir desse grande número de veículos funcionando tem-se um grande problema com o rastreio ou a identificação de cada um desses automóveis. A partir dessa afirmação, utilizou-se a visão computacional e o processamento de imagens para se ter uma melhor gestão e uma maior eficiência na fiscalização de todo o tráfego de veículos no Brasil.

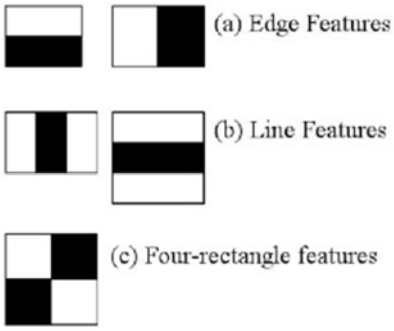
O reconhecimento visual e processamento digital podem ser caracterizados e utilizados de diferentes maneiras, mas apenas duas formas de reconhecimento visual foram abordadas neste trabalho. A primeira foi utilizando bibliotecas disponíveis na linguagem Python, sendo elas o OpenCV (*Open Source Computer Vision Library*), desenvolvido pela Intel no ano de 2000 para o reconhecimento visual; e o Pytesseract, desenvolvido pela Hewlett-Packard Company (comumente conhecida como HP) entre os anos de 1985 e 1994 para o reconhecimento ótico de caracteres. Em questões de redes neurais, são abordados métodos de detecção visual junto do YoloV4 (*You Only Look Once Version Four*) que funcionam a partir do framework Darknet que se utiliza de redes neurais profundas.

No algoritmo, sem a utilização de redes neurais, foi aproveitada a biblioteca OpenCV, que de acordo com a documentação da própria biblioteca (OPENCV.ORG, 2022) é voltada à visão computacional e se utiliza de funções de reconhecimento visual em cascata para melhores níveis de resultados e maiores taxas de acerto de acordo com seu arquivo de configurações.

De acordo com Marengoni e Stringhini (2009), neste arquivo existem duas funcionalidades gerais. A primeira é a terminologia de Classificadores que são utilizados para classificar imagens ou objetos dentro de uma imagem, e a partir do resultado dessa operação, retornar a área detectada junto de uma numeração do qual se determina as chances daquele objeto detectado ser realmente o que foi destinado a procurar.

A segunda funcionalidade é o Cascade, ou Cascata, que em sua principal ideia tem a função de separar toda a imagem fonte em várias janelas, e, dessa maneira, percorre todas elas iterativamente em diversas direções. Na Figura 1 (a), pode-se observar que a funcionalidade do modelo procura apenas na metade da imagem. Já na Figura 1 (b), procura ao centro a partir das divisões feitas. Na etapa da Figura 1 (c), divide-se a imagem em quatro e procura a partir de diagonais. Tudo isso funciona a partir das variações passadas como parâmetro anteriormente junto de seu objeto alvo, procurando assim em cada janela uma paridade com seus modelos de interesse, e se por acaso obter falso em todos as janelas, termina com um retorno negativo.

Figura 1 – Exemplificação do funcionamento do modelo em cascata em diversas posições



Fonte: GUALBERTO, 2018.

O modelo de cascata é previamente treinado com imagens positivas (imagens que contenham o objeto-alvo, ou objeto que está à procura) e negativas (imagens que não contenham o objeto-alvo), dessa maneira consegue-se entender a partir das variações de luzes se possui ou não seu objetivo de reconhecimento.

A ferramenta com rede neural a ser utilizada foi Yolo para a detecção visual, ou visão computacional. Este algoritmo funciona a partir dos pixels de uma fonte de imagem, filtrando-a e resultando nas probabilidades de classes existentes e pré-configuradas nas dimensões em volta do objeto, unindo isso tudo utilizando uma única rede neural. YOLO é um método de detecção de objetos de passada única *(single pass)* que utiliza uma rede neural convolucional como extrator de características *(features)*. A Rede Neural Convolucional é um algoritmo com foco em Aprendizado Profundo voltado a objetos a partir de imagens de entrada, atribuindo pesos e vieses que podem ser aprendidos, e diferenciar diversos objetos um do outro (REDMON et al., 2016).

A ideia principal de uma Rede Neural Artificial é funcionar de maneira similar a um cérebro humano, e com isso, possuem etapas parecidas em seu funcionamento para manter essa paridade funcional (BEZDAN; BACANIN, 2019).

São divididos em três etapas: a entrada de informação, a percepção e entendimento da informação, e por fim, a tomada de decisão. Na primeira etapa, entrada de informação, é onde toda a parte que irá ser processada ou visualizada para o entendimento será inserida para posteriormente ser enviada para a próxima etapa. Na segunda etapa, ocorre o processamento de tudo o que chegou através da comunicação vinda da primeira etapa, interpretando a fonte de informações de acordo com seus ensinamentos base, até enfim, chegar no resultado final, que será enviado para a terceira e última etapa. Por fim, na terceira etapa, ocorre a decisão final do que será a resposta de toda a informação que chegou. Foi processada e agora espera uma saída. Essa decisão ocorre a partir de dados previamente entendidos ou instaurados que, a partir disso, resultará numa resposta em cima de tudo o que se tem conhecimento (BEZDAN; BACANIN, 2019).

O Yolo utiliza uma rede neural profunda chamada Darknet, que é uma rede neural com aprendizado de máquina, possui código aberto e foi escrito na linguagem C. É uma arquitetura construída para suportar computação de CPU (*Central Processing Unit*) e GPU (Graphics Processing Units), com tudo isso já no seu primeiro ano de funcionamento ganhou grande parte dos pesquisadores da visão computacional pelo seu desempenho, agilidade, precisão e facilidade.

Primeiramente na questão das comparações, em ambos os casos, foram utilizados uma função de reconhecimento das placas como primeira função, obtida através da bilioteca OpenCV. O método funciona a partir de uma imagem colocada em um buffer, uma memória

física local que armazena dados temporariamente para uso, que a partir desta memória é lida e retornada a imagem passada para uma variável.

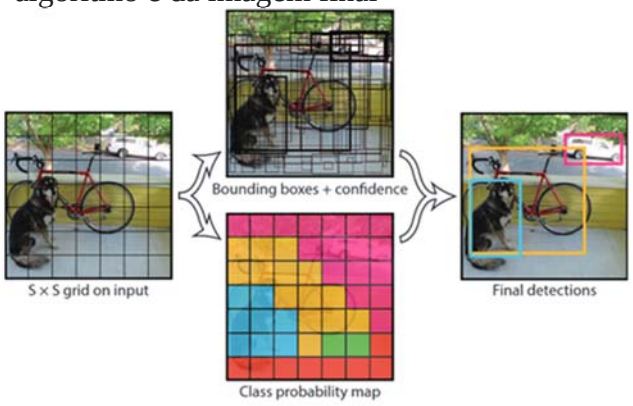
A partir deste passo o processo de cada um dos algoritmos se diferem. No caso do algoritmo sem a utilização de redes neurais, o processo de extração das placas começa com o carregamento prévio do arquivo pré-treinado em cascata para a localização de placas automotivas.

Após esta etapa ocorre encontrar efetivamente eventuais objetos-alvo nas imagens passadas e lidas anteriormente a partir deste modelo de cascata. Neste momento há o uso de uma função da biblioteca OpenCV que detecta objetos-alvo de diferentes tamanhos dentro da imagem passada e os possíveis resultados são passados em forma de lista junto de suas posições (ANTONELLO, 2017).

Já na visão computacional junto da utilização de redes neurais os passos para a extração da placa seguem passos semelhantes, mas com métodos totalmente diferentes. Em primeiro momento há o carregamento do peso de treinamento da rede neural, onde há os parâmetros para encontrar o possível objeto. Este peso de treinamento é anteriormente carregado a partir de uma base de dados massiva de imagens contendo conteúdos positivos e negativos como visto anteriormente, mas sua diferença é em seu treinamento, que se utiliza de rede neural convolucional. Dessa maneira, ela é informada e treinada utilizando funções e algoritmos que simulam uma rede neural, assim mantendo uma quantidade significativa de dados em um arquivo para uma utilização posterior (MARENGONI; STRINGHINI, 2009).

Em segundo momento é utilizado o algoritmo de detecção de imagens doYoloV4 junto do peso de dados para a detecção de possíveis objetos. Segundo Redmon et al. (2016), a função de detecção remodela a imagem e a define de um tamanho determinado, depois filtra toda a imagem a partir de quadrados pixelados de acordo com o tamanho configurado no peso (S x S), como pode ser visto na Figura 2; e, a partir desses pequenos quadrados, faz as detecções em toda a imagem obtendo as pontuações para cada quadrado de detecção. Se houver detecções de objetos de interesse nos quadrados, o algoritmo retorna um retângulo em volta da área e assim retornará uma pontuação de acordo com a chance de ter o objeto alvo dentro da imagem juntamente da localização.

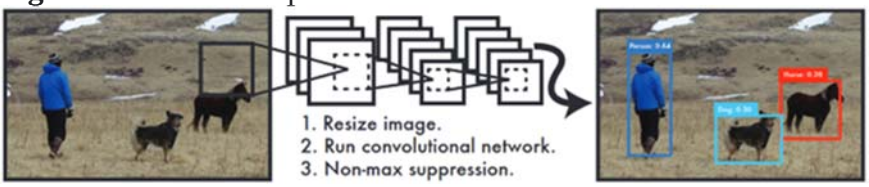
Figura 2 – Exemplificação das repartições das janelas criadas pelo algoritmo e da imagem final



Fonte: REDMON et al., 2016.

Após cada algoritmo utilizar suas devidas funções de reconhecimento visual, as imagens de retorno obtidas serão recortadas em áreas onde tem a possibilidade de existir um objeto-alvo, como mostra na Figura 3, e então, aplicar-se-ão filtros de processamento digital com a finalidade de melhorar a extração de caracteres no final do processo.

Figura 3 – Passos simplificados de como funciona o Yolo



Fonte: REDMON et al., 2016.

No processamento digital foram utilizados alguns filtros para uma melhor taxa de extração de caracteres nas imagens que foram obtidas através dos métodos anteriores. Cada filtro possui sua função específica nas imagens e assim deixando a área alvo o mais simplificado possível para uma melhor leitura dos caracteres.

O primeiro filtro a ser usado é uma transformação da imagem para tons de cinza, tirando suas tonalidades e deixando todas as cores em diferentes tonalidades, sendo elas constituídas de preto, branco e tons de cinza (OPENCV.ORG, 2021). A função pode ser vista na equação 1.

$$Y \leftarrow 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \qquad (1)$$

O segundo filtro é o Blur Gaussiano, que de acordo com Antonello (2017), se consiste numa suavização da imagem retirando os ruídos e diminuindo o nível de detalhamento, borrando assim alguns pontos e corrigindo muitas irregularidades. Dessa maneira a aplicação de filtros em sequência se consegue obter um melhor desempenho dentro da imagem. A fórmula do filtro gaussiano pode ser vista a seguir na equação 2.

$$G(x,y) = G(x).G(y)^t = \frac{1}{2\pi\sigma^2} e \frac{x^2+y^2}{2\sigma^2} \qquad (2)$$

No terceiro filtro, após a suavização da imagem utilizou-se a binarização, que consiste simplificadamente em deixar a imagem com apenas dois tons, o preto e o branco. Para sinalizar como cada cor será alterada, é passado um tom de parâmetro para que cada tom de cor que estiver acima desse parâmetro vá para a preta, e no tom abaixo vá para a cor branca. Dessa maneira, o contraste entre as cores mostra com maior probabilidade as informações necessárias sem o eventual problema de possíveis erros por cores parecidas (FERNANDES, 2019). Junto à binarização foi usado o método Otsu, ao qual se utiliza de funções que delimitam o melhor valor para a binarização e assim busca um meio termo entre todos os picos de cores. Dessa maneira consegue-se colocar uma melhor variância para melhorar a probabilidade de deixar apenas o essencial na imagem e retirar todo o resto (FERNANDES, 2019). O cálculo de Otsu pode ser encontrada na equação 3.

$$\sigma \frac{2}{w}(t) = q_1(t)\sigma \frac{2}{1}(t) + q_2(t)\sigma \frac{2}{2}(t) \qquad (3)$$

E por fim, o método de dilatar a imagem previamente binarizada consegue melhorar o arredondamento de bordas e retirar possíveis imperfeições ainda deixadas pela binarização. A dilatação da imagem tem em sua função aumentar o núcleo da imagem, expandindo-o e dando a impressão da "dilatação", com isso a parte branca ou brilhante da imagem se dilata em torno das regiões que a cercam (PINHO, 2022). A função utiliza a equação 4 como fonte para o resultado.

$$dst(x,y) = max(x^1,y^1): element(x^1,y^1) \neq 0 src(x+x^1, y+y^1) \qquad (4)$$

Na Figura 4 observa-se exemplos de imagens com todos os filtros sendo aplicados, desde a original, e depois passando pelos filtros de escala de cinza, Gaussian Blur, Threshold, e dilatação, respectivamente.

Figura 4 – Comparação da imagem original e aplicação dos filtros de processamento digital (Em ordem: Imagem Original, Escala de Cinza, Gaussian Blur, Threshold, Dilatação)

Fonte: Elaborada pelos autores.

Com todos os filtros aplicados ao recorte da imagem previamente feito, os algoritmos utilizaram os mesmos métodos de extração para uma melhor comparação. A função de reconhecimento dos caracteres da placa vem da biblioteca Pytesseract, que por sua vez, consegue encontrar as informações do texto pela mediana de espaços entre eles, e assim, encontra um padrão para a segmentação das linhas e caracteres (SMITH, 2007).

Segundo Smith (2007), após a separação o algoritmo segue o contorno de cada possível caractere e tenta decifrar quais letras são essas, montando um conjunto de caracteres no sentido de formar uma palavra. Após terminar o segmento faz-se uma análise se o que foi feito realmente é válido como uma palavra útil a partir do seu arquivo de configuração, montando as características e comparando com alguma possível palavra ou termo encontrado em sua base. Depois de toda a validação feita, retorna o possível resultado em formato de texto que mais se adequar a sua base de dados.

3 METODOLOGIA

O modelo proposto neste trabalho inicia-se com a captura das imagens. No caso desta pesquisa os algoritmos foram aplicados em uma base de imagens com 200 imagens, e a partir dela, faz-se o seu processamento obtendo-se a placa automotiva presente. Utilizando-se deste conceito i será realizada a extração inicial da placa da imagem por meio de duas abordagens e posteriormente realizar a operação de reconhecimento de caracteres (OCR) para a leitura da placa.

Para o desenvolvimento dos algoritmos foi utilizado a linguagem Python, pois segundo Baggio (2020), a linguagem possui diversas ferramentas e bibliotecas voltadas à visão computacional e ao processamento digital, facilitando no todo o desenvolvimento do projeto. Juntamente à linguagem foram utilizadas as bibliotecas de reconhecimento e processamento visual: Tesseract-OCR (PYPI.ORG, 2022) e OpenCV (OPENCV.ORG, 2022). Nas abordagens que utilizam redes neurais convolucionais aplicou-se a rede YoloV4 (REDMON et al., 2016) e para seu funcionamento foi necessário o uso do framework de rede neural chamado Darknet.

De ferramentas estão sendo utilizadas o Visual Studio Code, que é um editor de códigos oferecido pela Microsoft gratuitamente, o Jupyter, que é uma plataforma de desenvolvimento e automação para que o projeto consiga ser compilado dentro de áreas em desenvolvimento, e o Google Collaboratory, que é uma plataforma também voltada para o desenvolvimento em computadores remotos oferecidos gratuitamente pela Google.

Com o estudo da linguagem de programação juntamente das bibliotecas, também há o estudo das placas automotivas brasileiras, como: seu tamanho, cor e formato para um melhor desempenho e uma maior taxa de acerto com a utilização de processamento de imagens, como também, a modificação da luminosidade, binarização da imagem, captura de bordas e blur.

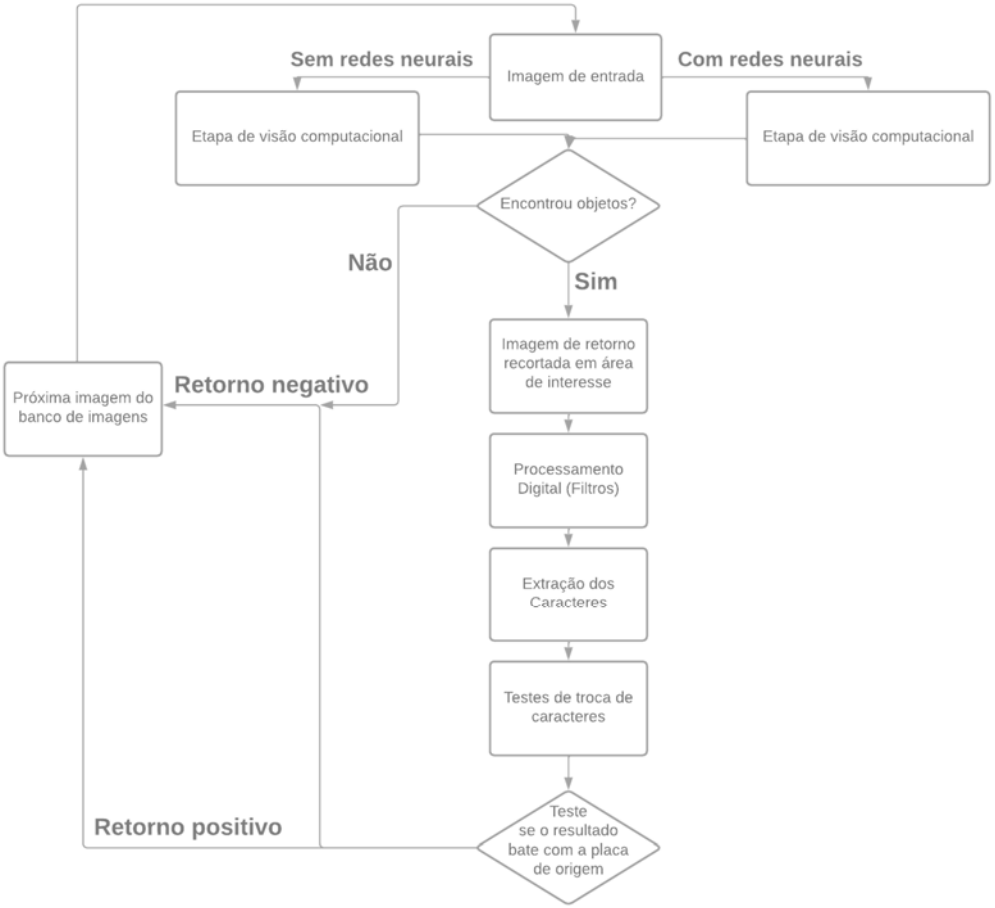
Tudo isso com o fim de se focar principalmente nos caracteres e deixar apenas a área de interesse em destaque na imagem obtida.

Para a construção da base de imagens retirou-se 200 imagens de um repositório público no GitHub do autor TheAIGuysCode (2022). Nesta base de imagens encontra-se placas capturadas por diversos usuários, com diversas câmeras, diversos ângulos de fotografia, diferentes ruídos e posições de sombreamento, tornando assim o processo de execução da comparação o mais realista possível, no intuito de obter os melhores resultados possíveis com o treinamento do algoritmo.

Na Figura 5, demonstra-se os passos de extração utilizando a abordagem com o uso de redes neurais convolucionais e sem a sua utilização, e também se visualiza o processo utilizado para o reconhecimento de caracteres após a identificação da placa veicular dentro da imagem.

Tudo começa na imagem de entrada, onde os processos realmente se diferenciam, após isso seguem para a etapa de visão computacional de acordo com o seu algoritmo. Após as detecções, há um teste para saber se foi encontrado algum objeto potencialmente positivo. Se por acaso não encontrou nada, pula para a próxima imagem do banco, e se encontrou, segue para o processamento digital, para posteriormente ser extraído os caracteres. E, no final, o teste para saber se o resultado da extração de caracteres está correto de acordo com a placa.

Figura 5 — Diagrama de etapas para os testes



Fonte: Elaborada pelos autores.

A primeira etapa é extração da placa veicular da imagem original que compõem a base utilizada para estes testes. Na etapa de extração propõe-se o uso de duas abordagens: uma com redes neurais convolucionais e a outra utilizando apenas processos de processamento de imagens.

Na abordagem com a utilização de redes neurais convolucionais a imagem é exposta a CNN YoloV4/Darknet que extrai as características e realiza o processamento de modo que retorna a placa extraída ao final do processo.

Na outra abordagem tem-se o emprego dos algoritmos da biblioteca OpenCV, que por sua vez procuram iterativamente por objetos que contenham formas ou contornos parecidos com seu objetivo. Após sua procura, retornam um contorno de uma possível placa caso seja encontrada.

Após o uso de uma das abordagens, verifica-se se o retorno possui alguma placa em potencial para se testar e extrair os caracteres. Se o resultado for verdadeiro, passa para o recorte da área prevista da placa, depois para o processamento digital, aplicando todos os filtros (de acordo com o demonstrado na Figura 5) para uma melhor extração de cada caractere.

Após o reconhecimento de caracteres realiza-se a verificação da taxa de acerto do modelo proposto, comparando-se a saída do modelo com o resultado esperado (placa correta).

Os testes realizados neste trabalho diferem-se no processo de encontrar-se a placa veicular nas imagens, utilizando-se de duas abordagens, uma vez encontrada a placa o processo de filtros e reconhecimento de caracteres é o mesmo para ambos os testes realizados.

4 ANÁLISE E DISCUSSÃO DOS RESULTADOS

Para esta etapa foram organizados os resultados obtidos da abordagem de extração da placa veicular da imagem original sem a utilização de redes neurais e da abordagem com a utilização de redes neurais convolucionais. Enfatiza-se que a base de imagens se mantém para ambos os testes, utilizando-se as 200 imagens do qual é composta. Durante os testes todos os parâmetros de extração de caracteres e leitura dos resultados foram os mesmos.

Na Tabela 1 visualiza-se as taxas de placas encontradas em cada uma das duas abordagens utilizadas.

Tabela 1 – Extração de Placa Veicular e taxas de acerto para as abordagens propostas

Placas (Base Imagem)	**Abordagem (sem CNN)**	**Abordagem (com CNN)**
200	78%	100%

Fonte: Elaborada pelos autores.

A abordagem utilizando CNN (*Convolutional Neural Network*) mostrou-se mais eficiente em encontrar as placas veiculares nas imagens da base e por meio deste método foram extraídas placas de todas as imagens utilizadas nos testes. Para a abordagem sem o uso de CNN obteve-se uma taxa de extração de 78%, existindo imagens em que não foi possível identificar a placa veicular.

Esse resultado pode ser explicado pelos métodos de cada um, pois as redes neurais possuem um processamento maior em quantidades de dados simultâneos, conseguindo assim, calcular o resultado com uma maior taxa de acerto junto de um processo com maior poder de dados, enquanto que a biblioteca OpenCV não consegue possuir um grande resultado devido à diferença da base de dados e treinamento entre os arquivos em cascata e os pesos treinados por redes neurais, como também pela forma como cada um tem seu funcionamento de reconhecimento visual.

Para os resultados do reconhecimento de caracteres desta comparação, a variação dos parâmetros segue o padrão de alternar os valores de entrada no filtro Gaussian Blur. A partir dele, com diferentes valores e resultados, seguir a aplicação dos demais filtros, que são a binarização e a dilatação explicados anteriormente, como pode ser visto na Tabela 2.

Esse processo pode ser explicado pois a suavização Gaussiana ou desfoque da imagem afeta diretamente no funcionamento de outros filtros que são aplicados em seguida. De acordo

com os diferentes níveis de suavização de ruídos e contornos, os métodos de binarização e dilatação retornam diferentes resultados da mesma imagem, assim, encontra-se a necessidade de realizar diversos testes seguindo esse padrão para que haja uma paridade com os parâmetros e seus resultados.

Tabela 2 – Testes da extração de caracteres em diferentes filtros e parâmetros e comparando os resultados de ambos os algoritmos

Escala Preto e Branco	Gaussian Blur	Threshold Binary + Otsu	Dilate	Taxa de Acerto sem Redes Neurais	Taxa de Acerto com Redes Neurais
Sim	X	X	X	32,00%	32,00%
Sim	1,1	X	X	32,00%	32,00%
Sim	3,3	X	X	26,00%	26,00%
Sim	5,5	X	X	25,00%	25,00%
Sim	7,7	X	X	22,00%	25,00%
Sim	9,9	X	X	28,20%	29,03%
Sim	1,1	Sim	x	15,92%	17,50%
Sim	3,3	Sim	X	21,79%	20,00%
Sim	5,5	Sim	X	19,35%	17,00%
Sim	7,7	Sim	X	20,50%	15,00%
Sim	9,9	Sim	X	20,50%	11,50%
Sim	1,1	Sim	Sim	12,73%	8,15%
Sim	3,3	Sim	Sim	12,65%	7,51%
Sim	5,5	Sim	Sim	14,01%	9,01%
Sim	7,7	Sim	Sim	13,92%	7.00%
Sim	9,9	Sim	Sim	14,10%	12,60%

Fonte: Elaborada pelos autores.

Ao observar a tabela 2, pode-se ver uma maior taxa de acerto quando a função opera sobre a imagem sem nenhum processamento digital. Esse processo pode ser explicado, pois a função de leitura dos caracteres da biblioteca está treinada para caracteres e palavras convencionais de forma com que seu arquivo de configuração está definido. Em sua função seguem um padrão de contorno, desenho, espaçamento e reta, mas, nas placas com o filtro adicionado, o conjunto de letras não segue nenhum padrão de espaçamento, curvas, bordas ou contorno, dificultando assim o reconhecimento de cada caractere.

Com isso pode-se notar que a taxa de sucesso ao reconhecimento das placas e extração dos caracteres é maior no algoritmo com a utilização de redes neurais. Assim, de acordo com todos os métodos estudados e utilizados neste artigo, tem-se o resultado que o reconhecimento visual deste algoritmo possui um retorno com uma maior taxa de acerto em comparação ao algoritmo utilizando apenas bibliotecas de visão computacional.

Outra questão que pode se levantar é o aumento de assertividade nos dois exemplos, mas com a mesma função de extração de caracteres. Para isso, há uma explicação de que a biblioteca OpenCV possui um reconhecimento visual às vezes falho, recortando uma parte da placa onde deveria existir o resto do objeto, invalidando assim a área de interesse.

Na Figura 6 pode-se notar que a placa foi recortada de maneira com que fosse impossibilitado de saber os 2 últimos números, e retirando metade do segundo número da sequência.

Figura 6 – Placa recortada erroneamente, retirando caracteres da área de interesse

Fonte: Elaborada pelos autores.

Nas Figuras 7 e 8 o recorte está retirando uma área maior do que a área de interesse, dando margem de erro para o algoritmo reconhecer bordas e aumentando as chances de reconhecer falsos números e letras.

Figura 7 – Exemplo de placa recortada erroneamente deixando espaços pelas laterais da placa

Fonte: Elaborada pelos autores.

Figura 8 – Exemplo de placa recortada erroneamente deixando espaços pelas laterais da placa

Fonte: Elaborada pelos autores.

5 CONCLUSÃO

Durante o desenvolvimento do projeto proposto neste trabalho, pode-se concluir que as redes neurais convolucionais possuem uma taxa de eficácia comprovadamente melhor em reconhecimento visual da localização da placa veicular em uma imagem quando se compara com um modelo de visão computacional, o que resulta em uma melhor detecção e extração dos caracteres com o OCR.

Além disso, junto de seu desempenho com redes neurais, também se consegue detectar muitos objetos-alvo dentro de uma fonte de imagem ao mesmo tempo, provando-se melhor em diversos pontos. A diferença de eficácia nas tecnologias acuradas neste trabalho, entre a utilização de redes neurais com YoloV4 e o reconhecimento visual oferecido pela biblioteca OpenCV, utilizando o mesmo método de extração e leitura dos caracteres, obteve um aumento nos resultados de 22% na visão computacional e 28% na extração e leitura dos caracteres das placas utilizando o algoritmo com o uso de redes neurais.

A vantagem vem devido ao seu grau de aprendizado e de reconhecimento em muitas de suas funções, utilizando de redes neurais artificiais e convolucionais. Em contrapartida, as redes neurais podem depender de mais recursos da máquina para funcionarem de forma adequada.

Para uma melhor implementação de uma rede neural demanda-se uma alta quantidade de recursos para assim conseguir o devido funcionamento e manter seu desempenho de acordo

com o esperado, aumentando o nível de requerimento para se adicionar um algoritmo com a utilização de redes neurais em um sistema.

Já na utilização da biblioteca OpenCV demanda apenas de um leve processamento de dados que pode ser feita em sistemas mais simples, diminuindo os requisitos e possuindo maiores facilidades para sua implementação, assim cada uma tendo suas preferências e eficácias em suas devidas áreas.

Como possíveis implementações futuras, o projeto pode seguir para a aplicação dos algoritmos em questão para o reconhecimento de veículos de pessoas autorizadas dentro de um recinto privado, ao acionar e abrir um portão eletrônico automaticamente com um carro anteriormente autorizado, e assim, melhorando toda a gestão de fluxo de veículos e aumentado a segurança interna do local.

REFERÊNCIAS

ANTONELLO, R. **Introdução a visão computacional com Python e OpenCV**. 2017. Disponível em: http://professor.luzerna.ifc.edu.br/ricardo-antonello/wp-content/uploads/sites/8/2017/02/Livro-Introdu%C3%A7%C3%A3o-a-Vis%C3%A3o-Computacional-com-Python-e-OpenCV-3.pdf. Acesso em: 15 mar. 2022.

BAGGIO, N. F. **Python**. 2020. Disponível em: https://cursos.alura.com.br/python-nelson-baggio-1589474259936-p33982. Acesso em: 13 out. 2022.

BEZDAN, T.; BACANIN, N. **Convolutional neural network layers and architectures**. 2019. Disponível em: https://www.researchgate.net/publication/333242381_Convolutional_Neural_Network_Layers_and_Architectures. Acesso em: 10 jun. 2022.

FERNANDES, L. S. **Reconhecimento automático de placas veiculares brasileiras incorporando Max-Margin Object Detection e redes neurais convolucionais**. 2019. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Universidade Federal do Ceará, Fortaleza, 2019. Disponível em: http://www.repositorio.ufc.br/handle/riufc/43285. Acesso em: 17 set. 2022.

GUALBERTO, A. **Detectando objetos com métodos clássicos**: OpenCV (cascades). 2018. Disponível em: https://medium.com/ensina-ai/detectando-objetos-com-m%C3%A9todos-cl%C3%A1ssicos-opencv-cascades-440e29913b1b. Acesso em: 17 set. 2022.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA – IBGE. **Projeção da população**. Disponível em: https://cidades.ibge.gov.br/brasil/pesquisa/22/28120. Acesso em: 15 abr. 2022.

MARENGONI, M.; STRINGHINI, D. Tutorial: **i**ntrodução à visão computacional usando OpenCV. **Revista de Informática Teórica e Aplicada**, v. 16, n. 1, p. 125-160, 2009. Disponível em: https://seer.ufrgs.br/index.php/rita/article/view/rita_v16_n1_p125/7289. Acesso em: 17 set. 2022.

OPENCV.ORG. **About OpenCV**. Disponível em: https://opencv.org/about/. Acesso em: 14 set. 2022.

OPENCV.ORG. **Docs**. Disponível em: https://docs.opencv.org/. Acesso em: 4 nov. 2021.

PINHO, S. M. **Processamento de imagens**. Disponível em: https://www.inf.pucrs.br/~pinho/CGII/PDFs/IntroducaoProcImg.pptx.pdf. Acesso em: 13 out. 2022.

PYPI.ORG. **Pytesseract 0.3.10**. Disponível em: https://pypi.org/project/pytesseract/. Acesso em: 15 set. 2022.

REDMON, J. *et al.* **You only look once**: infied, real-time object detection. 2016. Disponível em: https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/Redmon_You_Only_Look_CVPR_2016_paper.pdf. Acesso em: 10 maio 2022.

SALAS, J. **A humanidade não chegará aos 10 bilhões de pessoas**. 2020. Disponível em: https://brasil.elpais.com/ciencia/2020-07-16/a-humanidade-nao-chegara-aos-10-bilhoes.html. Acesso em: 15 abr. 2022.

SMITH, R. **An Overview of the Tesseract OCR Engine**. 2007. Disponível em: https://ieeexplore.ieee.org/abstract/document/4376991. Acesso em: 13 out. 2022.

THEAIGUYSCODE. **Yolov4-custom-functions**. Disponível em: https://github.com/theAIGuysCode/yolov4-custom-functions. Acesso em: 01 jun. 2022.