JOÃO OTAVIO GONÇALVES CALIS

APLICAÇÃO DE REDES NEURAIS CONVOLUCIONAIS PARA RECONHECIMENTO AUTOMÁTICO DE PLACAS DE VEÍCULOS

Monografia apresentada ao Departamento de Ciências de Computação e Estatística do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", como parte dos requisitos necessários para aprovação na disciplina.

São José do Rio Preto
2018

JOÃO OTAVIO GONÇALVES CALIS

APLICAÇÃO DE REDES NEURAIS CONVOLUCIONAIS PARA RECONHECIMENTO AUTOMÁTICO DE PLACAS DE VEÍCULOS

Monografia apresentada ao Departamento de Ciências de Computação e Estatística do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", como parte dos requisitos necessários para aprovação na disciplina.

Orientadora:
Profa. Dra. Inês Aparecida Gasparotto Boaventura

São José do Rio Preto
2018

C154a

Calis, João Otavio Gonçalves

Aplicação de redes neurais convolucionais para reconhecimento automático de placas de veículos / João Otavio Gonçalves Calis. -- São José do Rio Preto, 2018

67 p. : il., tabs., fotos

Trabalho de conclusão de curso (Bacharelado - Ciência da Computação) - Universidade Estadual Paulista (Unesp), Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto

Orientadora: Inês Aparecida Gasparotto Boaventura

1. Redes neurais (Computação). 2. Processamento de imagens. 3. Reconhecimento de padrões. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca do Instituto de Biociências Letras e Ciências Exatas, São José do Rio Preto. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

JOÃO OTAVIO GONÇALVES CALIS

APLICAÇÃO DE REDES NEURAIS CONVOLUCIONAIS PARA RECONHECIMENTO AUTOMÁTICO DE PLACAS DE VEÍCULOS

Monografia apresentada ao Departamento de Ciências de Computação e Estatística do Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista "Júlio de Mesquita Filho", como parte dos requisitos necessários para aprovação na disciplina Projeto Final.

_____
Profa. Dra. Inês Aparecida Gasparotto Boaventura

_____
João Otávio Gonçalves Calis

Banca Examinadora:
Prof. Dr. Adriano Mauro Cansian
Prof. Dr. Geraldo Francisco Donegá Zafalon

São José do Rio Preto
2018

Dedico esse trabalho primeiramente a Deus, porque dele, por ele e para ele são todas as coisas. Dedico também à todas as pessoas que de alguma forma contribuíram para a conclusão da minha graduação.

AGRADECIMENTOS

Agradeço, aos meus familiares, em especial a minha mãe por todo apoio ao longo do curso. A minha orientadora, profa. Dra. Inês pelo grande auxílio e paciência ao longo da confecção desta monografia. Aos meus colegas de classe e professores, especialmente a Amanda, que me apoiaram durante os anos de graduação.

RESUMO

Com o crescente tráfego de veículos nos centros urbanos, torna-se cada vez mais difícil o controle não automatizado da entrada desses veículos em ambientes restritos. A automatização de processos tem contribuído para resolução de problemas nos mais diversos segmentos da indústria, de sorte que também tem se mostrado promissora para o controle de tráfego automatizado. Dado este contexto, este trabalho tem por objetivo propor um sistema de baixo custo e pequenas dimensões capaz de aplicar técnicas de reconhecimento de padrões para controlar a entrada de veículos de forma automática em ambientes privados. Para isso, utilizou-se redes neurais convolucionais (CNNs), uma das técnicas de aprendizado profundo (*Deep Learning*), para a detecção do objeto "placa de veículo" em uma imagem digital, bem como o reconhecimento dos caracteres que compõem a placa, tais algoritmos formam o núcleo do sistema de reconhecimento automático de placas de veículos. Os resultados mostram que algoritmos de aprendizado profundo baseados em redes neurais convolucionais, apesar de exigirem um grande conjunto de imagens para a fase de treinamento, são bastante eficazes para detecção de placas e também dos caracteres presentes em uma imagem digital.

Palavras-chave: redes neurais convolucionais, reconhecimento de placas, visão computacional, *Raspberry Pi 3*.

ABSTRACT

With the increasing traffic of vehicles in urban centers, it has become increasingly difficult to control the entry of these vehicles into restricted environments without automation. Processes automation has contributed to solve the most diverse problems on several industries, so it has also shown promise for automated traffic control. Given this context, this work proposes a low cost and small size system capable of applying pattern recognition techniques to control the entry of vehicles automatically in private environments. For that, convolutional neural networks (CNNs) were used, which is one of the techniques of deep learning (Deep Learning) for the detection of the "vehicle plate" object in a digital image, as well as the recognition of the characters that make up the plate, which form the core of the automatic vehicle plate recognition system. The results show that deep learning algorithms based on convolutional neural networks, although requiring a large set of images for the training phase, are quite effective for detecting plaques and also the characters present in a digital image.

Keywords: convolutional neural network, plate recognition, computer vision, security, *Raspberry Pi 3*.

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

OpenCV: *Open Source Computer Vision Library*

VPN: *Virtual Private Network*

IDE: *Integrated Development Environment*

USB: *Universal Serial Bus*

HDMI: *High Definition Multimedia Interface*

GPIO: *General Purpose Input/Output*

SD: *Secure Digital (Card)*

WiFi: *Wireless Fidelity*

CPU*: Central Processing Unit*

RNA: Rede Neural Artificial

CNN: *Convolutional Neural Network*

ReLU: *Rectified Linear Unit*

VGG: *Visual Geometry Group*

ResNet: *Residual Neural Network*

FC: *Fully Connected*

CAPÍTULO 1 - Introdução

1.1 Considerações iniciais

O crescente interesse pela automatização de processos e a introdução de novas tecnologias são fatores motivadores para diversas pesquisas no ramo de reconhecimento de padrões em imagens. Atualmente, o avanço da tecnologia digital, associado ao desenvolvimento de novos algoritmos, tem permitido um número de aplicações cada vez maior para resolver problemas em diversas áreas, tais como medicina, biologia, automação industrial, sensoriamento remoto, astronomia, militar, segurança e vigilância [1].

Essa demanda crescente na automatização de tarefas, tradicionalmente executadas por seres humanos, tem favorecido o desenvolvimento de sistemas baseados em reconhecimento de padrões. Por exemplo, sistemas de reconhecimento facial, sistemas de reconhecimento de impressões digitais e, em especial, o reconhecimento óptico de caracteres permite a identificação automática de caracteres escritos em diversos idiomas a partir de imagens de documentos e, também, o reconhecimento automático de placas de veículos a partir de imagens [2], [3], [4], [5], [6].

Por outro lado, com o aumento do número de veículos trafegando pelos centros urbanos, torna-se cada vez mais necessário o controle dos automóveis que passam por

determinados lugares e, uma maneira eficaz de se empregar tal controle é a fiscalização de quais veículos trafegaram por determinados locais durante um dado momento [7].

Assim, a necessidade de um sistema capaz de reconhecer placas de veículos automotores é cada vez maior. A literatura especializada na área de reconhecimento de padrões traz experiências recentes no desenvolvimento de tais sistemas, como por exemplo, identificação de veículos permitindo ou negando o acesso a áreas restritas em condomínios fechados [8], o controle de veículos em estacionamentos [9], sistema de cobrança de pedágio por meio da placa do veículo [10], [11].

Existe atualmente grande interesse em aplicações em que um sistema baseado no reconhecimento automático de placas de veículos, embarcados ou não, pode ser desenvolvido e comercializado. Além disso, com o avanço do desenvolvimento tecnológico e com o preço cada vez mais acessível das filmadoras, máquinas fotográficas digitais e *WebCams*, tornam-se viáveis aplicação de reconhecimento por imagens aplicadas a problemas do cotidiano [12], [9], [13], [14].

1.2 Objetivo

O objetivo deste projeto foi desenvolver um protótipo de um sistema de controle de acesso baseado em visão computacional capaz de reconhecer a placa de um veículo e verificar se o mesmo possui ou não permissão de acesso a um determinado local, buscando uma solução de baixo custo, com resposta em um tempo viável para esse tipo de sistema.

O sistema desenvolvido foi integrado a um computador de pequeno porte, servindo como plataforma de controle de acesso a ambientes reservados, de forma que a solução automatize toda a tarefa de controle de acesso. Para isso, o sistema captura a imagem da placa do carro, faz a detecção e o reconhecimento por meio de uma combinação das estratégias de janela deslizante e pirâmide de imagens, que, por sua vez, utilizam uma rede neural convolucional para classificar cada um dos recortes obtidos da imagem de entrada.

Caso a localização da placa ocorra com sucesso e os caracteres nela contidos sejam adequadamente classificados, o sistema compara os dados obtidos com os dados

registrados em uma base de dados e, se o veículo estiver registrado, a cancela/portão é acionada automaticamente, permitindo sua passagem.

1.3 Justificativa

Considerando que o sistema possa ser instalado em diversas localidades e, devido as particularidades de cada ambiente, é importante que o hardware do sistema seja versátil, de pequenas dimensões e tenha poder computacional suficiente para garantir que as operações sejam feitas em um tempo aceitável.

Dadas estas necessidades, um microcontrolador atende à necessidade das dimensões reduzidas, porém sua capacidade de armazenamento e processamento não são suficientes para armazenar o banco de dados com os usuários autorizados e também não é capaz de realizar os cálculos matriciais necessários devido a suas limitações de *hardware*.

Em outro extremo, um computador convencional seria desnecessário para o escopo da aplicação, além de maiores dimensões, a capacidade de armazenamento e processamento de um computador pessoal seriam subutilizados pelo sistema, uma vez que a complexidade dos dados e dos algoritmos de reconhecimento treinados podem ser tratados por um dispositivo menos complexo.

Dessa forma, a placa controladora Raspberry Pi 3 modelo B foi elegida para este propósito. Ela possui as seguintes configurações [15]:

- Processador quad core de 64 bits com 1.2 Giga Hertz;
- 1 Gigabyte de memória RAM
- Porta CSI para conexão da câmera

As configurações desse modelo suprem as necessidades de hardware do sistema. Para o armazenamento, foi utilizado um cartão de memória com capacidade de 32 Gigabytes, o suficiente para instalação dos softwares e bibliotecas do ambiente e acomodação do banco de dados.

Para fazer a captura das imagens analisadas pelo sistema, uma câmera apropriada para a placa foi instalada, ela possui resolução de 8 Megapixels e é capaz de registrar imagens com a definição necessária para o reconhecimento dos caracteres.

A implementação do software do sistema foi feita na linguagem Python utilizando a biblioteca openCV, esta linguagem foi elegida devido ao seu poder de integração com as funcionalidades da Raspberry. Além disso, a união da linguagem com a biblioteca de visão computacional openCV permite vários testes e operações em alto nível que em outras linguagens, como por exemplo C ou C++, levariam um tempo consideravelmente maior, pela maior complexidade de implementação das chamadas de funções de tal biblioteca.

1.4 Motivação

É notório o grande aumento do número de veículos automotores não só no Brasil como também em todo o mundo. Este crescimento torna cada vez mais impraticável o controle dos fluxos de entrada de automóveis em recintos privados tomando-se apenas recursos humanos. Outro ponto a ser considerado é a falha e negligência humana, uma vez que em diversas situações e contextos os responsáveis pela autorização da entrada podem permitir a entrada de veículos não autorizados num dado momento.

Um sistema automatizado capaz de controlar a entrada desses veículos em tempo hábil, permitindo um fluxo maior de entrada em menos tempo, é capaz de reduzir a necessidade de recursos humanos para este controle, além de minimizar os erros cometidos ao se permitir entradas não autorizadas.

Na área de processamento de imagens e reconhecimento de padrões, o principal desafio, é aumentar as taxas de acerto dos classificadores atualmente empregados para a classificação de padrões. Tal desafio motivou o estudo de aprendizado profundo (*Deep Learning*) aplicado na tarefa de reconhecimento de caracteres e reconhecimento de objetos dentro de uma imagem que, deste trabalho, são as placas dos veículos.

1.5 Metodologia

Para que o sistema fosse capaz de fornecer resultados confiáveis e em tempo hábil para dar a vazão adequada ao fluxo de veículos, o enfoque do trabalho foi abordar

as principais tecnologias, no que se refere as técnicas de visão computacional para o reconhecimento de caracteres de placa, bem como os algoritmos necessários para identificar a posição da placa na imagem.

Na primeira etapa do desenvolvimento deste trabalho, o ambiente computacional foi preparado. Foi necessário compreender as particularidades do funcionamento da placa controladora, bem como da câmera acoplada a ela, alguns cuidados tiveram que ser tomados para que o equipamento não fosse danificado. O sistema operacional Raspbian em sua versão atual, bem como o interpretador da linguagem Python e a biblioteca openCV foram instalados.

Na segunda etapa, um levantamento do estado da arte foi feito. Foi realizada uma análise sistemática em artigos, periódicos e livros que tratam dos temas de processamento de imagens e reconhecimento de padrões para o problema em questão, que é o reconhecimento de caracteres e reconhecimento da localização da placa do veículo em uma imagem capturada da frente ou traseira dele. Essa etapa teve como objetivo o aprendizado a respeito do tema e a escolha de uma técnica adequada ao problema.

Na terceira etapa estudou-se as tecnologias de software utilizadas: os comandos do sistema operacional a serem utilizados, bem como as funcionalidades da biblioteca openCV e a linguagem Python.

A quarta etapa foi a implementação do software do sistema. A integração das chamadas dos procedimentos do sistema operacional que controlam o hardware com o software do sistema também foi realizada nesta etapa bem como a construção do banco de dados das placas licenciadas.

E finalmente, na quinta etapa, os testes para comprovar a eficácia do sistema foram realizados.

1.6 Organização da monografia

Este documento está dividido em cinco capítulos. No capítulo 2 será detalhada a fundamentação teórica necessária para a compreensão do funcionamento do sistema proposto. No capítulo 3 será descrita a arquitetura do sistema bem como a descrição do seu funcionamento como um todo. No capítulo 4 serão apresentados os testes e

analisados os resultados obtidos após a implementação do sistema. No capítulo 5 serão apresentadas as conclusões gerais sobre este trabalho bem como possíveis aperfeiçoamentos em trabalhos futuros.

CAPÍTULO 2 - Fundamentação teórica

2.1 Considerações iniciais

Neste capítulo serão apresentados os conceitos necessários para a compreensão do trabalho.

Inicialmente são apresentados na seção 2.2 os principais conceitos de processamento de imagens para servir como base aos demais tópicos expostos neste capítulo. Na seção 2.3 é feita uma breve introdução sobre visão computacional.

O hardware utilizado é parte importante do sistema, de modo que seu custo e arquitetura dependem da utilização da controladora Raspberry. A seção 2.4 possui informações relevantes sobre a mesma.

Na seção 2.5 apresenta-se a biblioteca *opensource* de visão computacional, a openCV, que implementa os algoritmos básicos de processamento digital de imagens.

As redes neurais artificiais foram escolhidas para compor o núcleo de reconhecimento de placas e caracteres do sistema. Mais especificamente, o algoritmo de redes neurais convolucionais foi aplicado para este fim. Dessa forma, as seções 2.6 e 2.7 tratam de redes neurais artificiais e redes neurais convolucionais respectivamente. CNNs são classificadas como algoritmos de aprendizagem profunda, por este motivo, os principais conceitos sobre este tema são apresentados na seção 2.8

O sistema não deve apenas classificar as placas e os caracteres dos veículos, mas também detectar automaticamente sua posição na imagem adquirida. Dos métodos

encontrados na literatura, os algoritmos de janela deslizante e o algoritmo *single shot multibox detector* são apresentados nas seções 2.9 e 2.10.

Na seção 2.10 abordam-se os trabalhos relacionados na literatura para a detecção de placas de automóveis.

2.2 Conceitos de processamento de imagens

A necessidade dos métodos para processar imagens digitais vem de duas necessidades principais: melhorar sua representação visual para os humanos e processar estas imagens para fins de armazenamento, transmissão e também para o reconhecimento das informações nelas contidas por meio de máquinas [1]. Destaca-se o processamento para o reconhecimento de máquina, foco deste trabalho.

Entre as técnicas aplicadas no contexto de detecção de placas, a binarização e equalização de histograma se destacam por constituírem a etapa de pré-processamento necessária para alguns métodos de detecção.

2.2.1 Transformações geométricas

Em diversas situações é necessário que uma imagem em sua forma original seja transformada. As operações realizadas visam alterar as características da imagem original de modo que sua posição, orientação, forma ou tamanho mudem [16].

As principais transformações geométricas são:

- Translação: consiste em deslocar a imagem no eixo XY. Este deslocamento faz com que a imagem se desloque para cima, para baixo, para a esquerda, para a direita ou para qualquer uma destas combinações;
- Rotação: rotacionar uma imagem é o mesmo que girá-la de acordo com algum ângulo. Também é necessário que um ponto seja especificado para que a rotação seja feita, geralmente escolhemos o centro da imagem;

- Redimensionamento: muitas aplicações exigem um tamanho fixo para a imagem de entrada. Por esse motivo é necessário redimensionar a imagem de entrada, de forma que sua altura e largura sejam aumentadas ou diminuídas. Esta alteração pode ser feita proporcionalmente ou definindo uma nova dimensão fixa para a altura e a largura;
- Inversão: é um tipo de rotação da imagem entorno de seus eixos, dessa forma, podemos inverter a imagem horizontalmente ou verticalmente.
- Recorte: recorta-se uma imagem para obter apenas a fatia de interesse e descartar as demais partes.

2.2.2 Binarização

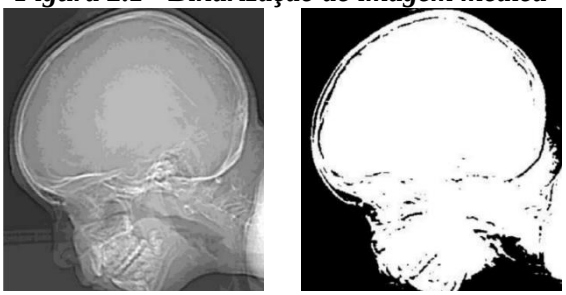
O processo de binarização ou limiarização consiste em separar as regiões da imagem original em duas classes, uma representando o fundo e a outra o objeto. Na prática, a imagem resultante desse processo é composta apenas pelos valores 0, que representa a cor preta e 255, que representa a cor branca. A possibilidade de apenas dois valores de intensidade justifica o nome binarização [17].

Esta separação é feita com base em um valor de limiar, de modo que todos os pixels da imagem que não atingem esse limiar são classificados como fundo e os demais como objeto.

A binarização de imagens contendo placas de carro pode ser interessante pois reduz drasticamente a informação presente na imagem. Com a determinação correta do valor de limiar os caracteres da placa podem ser facilmente detectados.

Na Figura 2.1 é apresentado um exemplo de binarização de imagem médica de uma radiografia craniana.

Figura 2.1 - Binarização de imagem médica



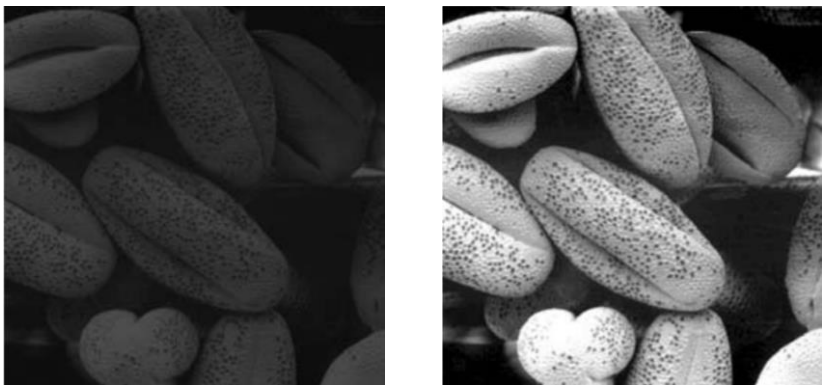
Fonte: retirada de [11].

2.2.3 Equalização de histograma

Um histograma nada mais é que uma representação, geralmente gráfica, do número de ocorrências de cada intensidade luminosa de uma imagem [1].

O processo de equalização de histograma altera os valores de intensidade de uma imagem de forma a redistribuir esses valores visando tornar o histograma da imagem aproximadamente uniforme. A equalização de histograma geralmente corrige deficiências de luminosidade em imagens, permitindo que os métodos de detecção tenham uma taxa de erro menor na detecção. Um exemplo de aplicação da equalização de histograma pode ser visto nas Figuras 2.2 (a) e 2.2 (b).

Figura 2.2 – Equalização de imagem



(a)Imagem original ***(b)imagem equalizada***

Fonte: retirada de [11].

2.2.4 Filtragem espacial

As técnicas de processamento de imagens relacionadas a filtragem espacial são implementadas no domínio espacial, que se trata simplesmente do plano contendo os pixels de uma imagem. Em geral, as técnicas no domínio espacial são computacionalmente mais eficientes e requerem menos processamento ao serem realizadas [1].

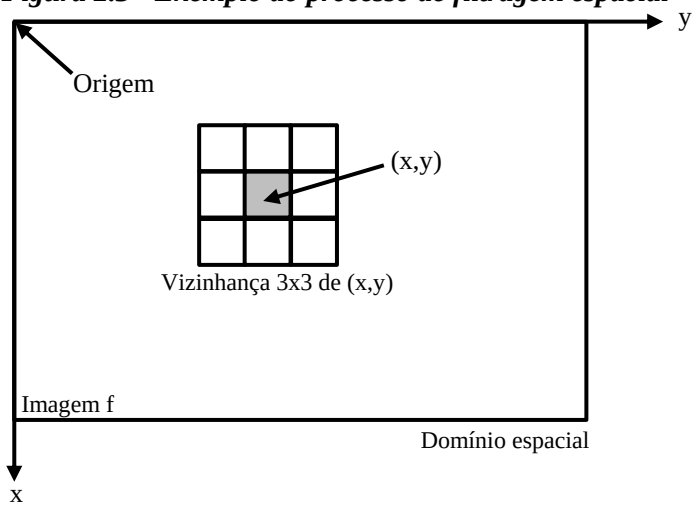
Em termos matemáticos, a filtragem espacial pode ser expressa pela Equação 1:

$$g(x,y) = T[f(x,y)] \quad (1)$$

onde f(x,y) é a imagem de entrada, g(x,y) a imagem de saída e T um operador em f definido em uma vizinhança do ponto (x,y).

Por meio da Figura 2.4 pode-se observar como o processo de filtragem espacial ocorre.

Figura 2.3 - Exemplo do processo de filtragem espacial



Fonte: adaptada de [11].

O processo de filtragem ocorre com uma matriz, geralmente de dimensões ímpares, iterando sobre a vizinhança de cada pixel da imagem f. A matriz f, juntamente com a definição do operador T, compõe o que é chamado de filtragem espacial. O operador T é definido como o filtro espacial. Ao final do processo, a imagem de saída g é obtida.

Quando se considera uma vizinhança de 1x1, cada ponto g(x,y) da imagem de saída *g* será gerado a partir de seu ponto correspondente f(x,y) da imagem de entrada *f*. Neste caso especial, tem-se uma função de transformação de intensidade, pois a vizinhança não é considerada. Existem três tipos básicos de funções de transformação de intensidade: linear (transformações de negativo e de identidade), logarítmica (transformações de log e log inverso) e de potência (transformações de n-ésima potência e n-ésima raiz).

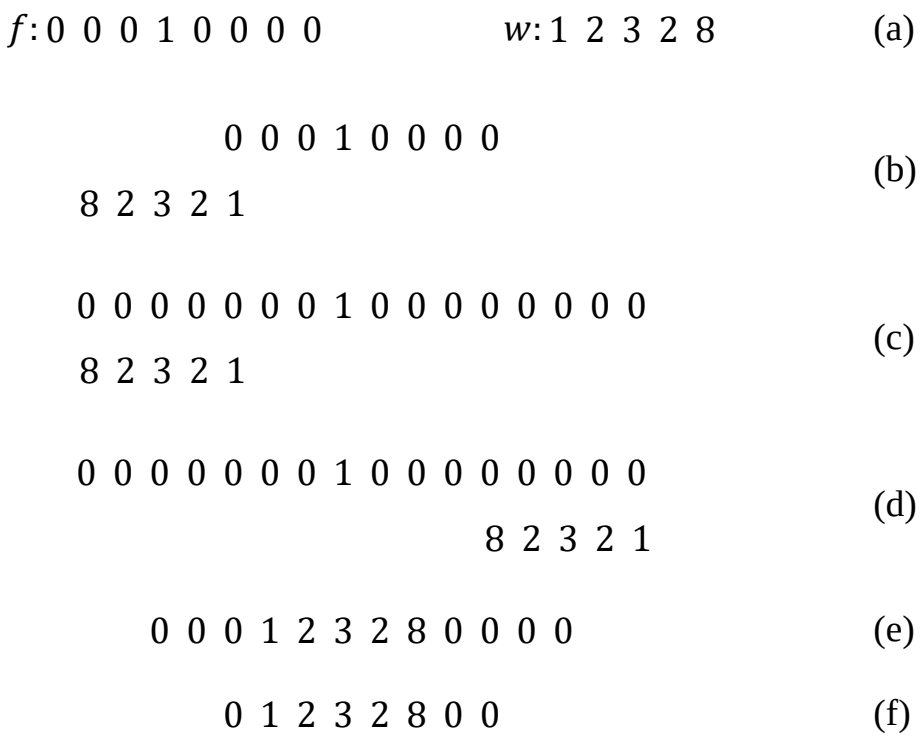
Os filtros são essencialmente importantes em redes neurais convolucionais pois constituem as camadas convolucionais dessa rede como será exposto adiante.

2.2.5 Operador de convolução

O funcionamento do operador de convolução é um dos conceitos de processamento de imagens mais importantes para a compreensão deste trabalho dado que o tipo de rede neural empregada leva seu nome.

Antes de apresentar a aplicação do operador em funções bidimensionais, consideraremos inicialmente uma função unidimensional f com um filtro unidimensional w, ambos ilustrados na Figura 2.5 (a) [1].

Figura 2.4 - Operação de convolução unidimensional



Fonte: adaptada de [1].

O primeiro passo da convolução consiste em rotacionar w em 180º (Figura 2.5 (b)). Feito isso, alinha-se seus valores com os valores de f configurando a posição inicial. Este alinhamento está ilustrado na Figura 2.5 (c). Deve-se adicionar zeros em f para garantir que cada pixel em w percorra todos os pixels em f, isto nos leva a uma função f temporariamente modificada, como pode ser visto na Figura 2.5 (c).

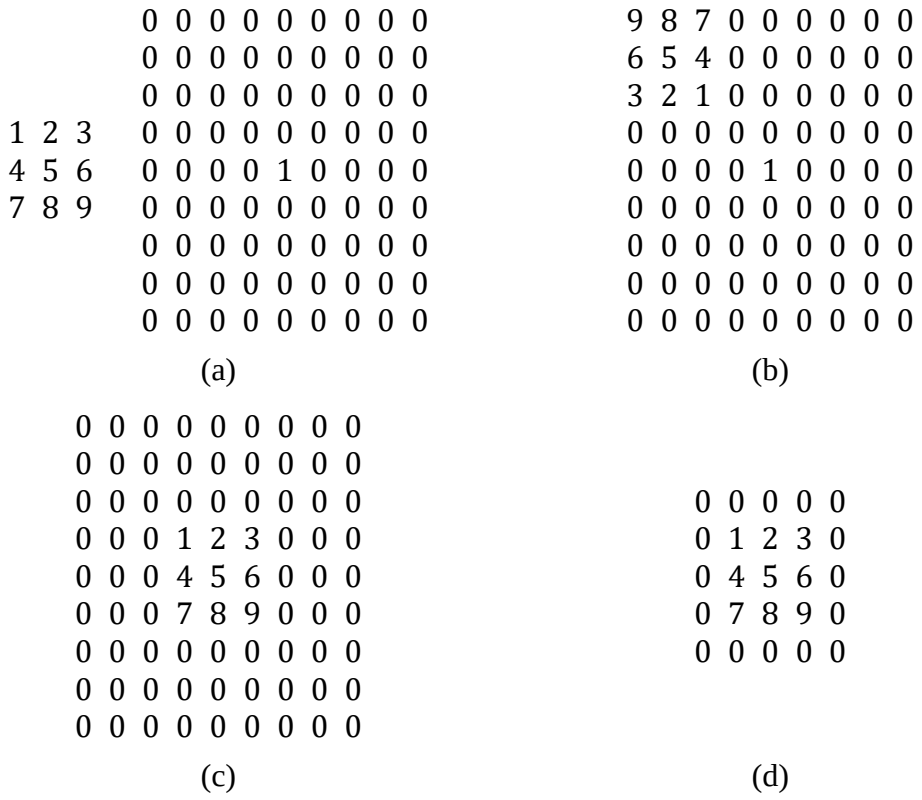
Com as devidas adequações, o processo de convolução consiste em computar a soma dos produtos dos pixels de w e f a cada posição de w. A função w se desloca para a direita até que seu primeiro pixel tenha sido multiplicado pelo último pixel da função f original, chegando a posição final ilustrada na Figura 2.5 (d).

No exemplo da Figura 2.5 foram necessários 12 deslocamentos da função w para completar a convolução. O resultado dessa operação pode ser visto na Figura 2.5 (e). A fim de que o resultado seja do mesmo tamanho da função original, recorta-se a função obtida, resultando na função final da Figura 2.5 (f).

O caso bidimensional é uma simples extensão do caso unidimensional anterior. Considerando-se que a função w tenha dimensões m × n, devemos completar f com m-1 linhas de zeros acima e abaixo e n-1 colunas à esquerda e a direita [1]. Um exemplo de convolução bidimensional pode ser visto na Figura 2.6.

Na Figura 2.6 (a) w é a matriz à esquerda enquanto f é a matriz à direita já preenchida com zeros, isto significa que a função original possuía dimensão 5 × 5.

Figura 2.5 - Operação de convolução bidimensional



Fonte: adaptada de [1]

Assim como no caso unidimensional, deve-se rotacionar w em 180º, no entanto, esta rotação deve ser feita nos dois eixos, na Figura 2.6 (b) tem-se w rotacionada e, na posição inicial para realizar a convolução. O processo ocorre de maneira semelhante ao exposto anteriormente, de modo que, o resultado final da convolução pode ser visto na Figura 2.6 (c) e seu recorte para as mesmas dimensões da função e f original pode ser visto na Figura 2.6 (d).

Pode-se representar matematicamente o processo descrito anteriormente, para o caso bidimensional, por meio da Equação 2, onde w(x,y) é o filtro de tamanho m × n aplicado a imagem f(x,y), a = (m-1)/2 e b=(n-1)/2.

$$w(x,y) \star f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) \cdot f(x-s, y-t) \qquad 2$$

Essa equação itera sobre todos os pixels da imagem f garantindo que todos os elementos de w percorram todos os pixels de f, admitindo que f foi previamente preenchida com os zeros necessários.

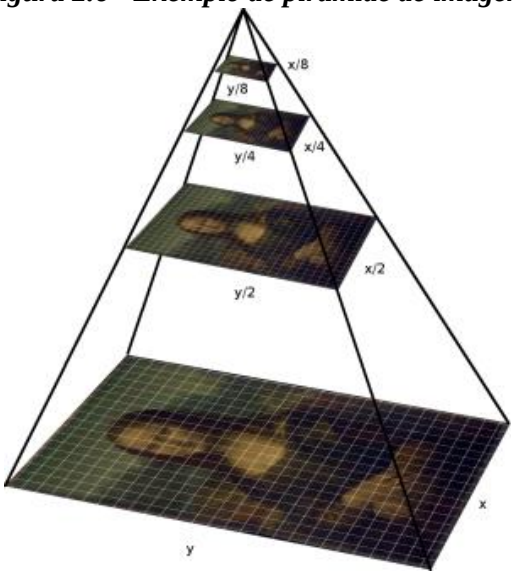
2.2.6 Pirâmide de imagens e janela deslizante

Após treinar um classificador de imagens é necessário que algum método seja empregado para apresentar as fatias da imagem que sejam potenciais objetos de interesse para que o classificador faça a predição da entrada.

Um método simples para esta tarefa consiste na combinação das técnicas de pirâmide de imagens e janela deslizante, na busca de objetos numa imagem.

Uma pirâmide de imagens é uma representação de uma imagem em múltiplas escalas [16], um exemplo da aplicação dessa técnica pode ser visto na Figura 2.7.

Figura 2.6 - Exemplo de pirâmide de imagens



Fonte: www.pyimagesearch.com/2015/03/16/image-pyramids-with-python-and-opencv/

A imagem original forma a base da pirâmide, as próximas camadas são formadas redimensionando a imagem anterior de acordo com um fator de escala pré-definido. A geração de novas camadas continua até que algum critério de parada seja atingido, que por sua vez pode ser um número fixo de iterações ou um tamanho mínimo atingido.

Uma janela deslizante é uma caixa retangular de dimensões fixas que itera sobre uma imagem [18]. O nome vem da impressão de deslizamento da janela sobre a imagem quando se tem uma representação gráfica do algoritmo.

Para cada uma das janelas, o recorte obtido pelas dimensões e posição da janela é aplicado ao classificador para determinar se naquela região há um objeto de interesse. Uma vez que o tamanho da janela é fixo, combina-se os algoritmos de redução de escala e de janela deslizante para que seja possível reconhecer objetos em escalas e locais variados na imagem.

Embora estas técnicas sejam de fácil implementação, sua complexidade computacional no pior caso é aproximadamente da ordem de $O(n^3)$, considerando o custo de percorrer toda a imagem com a janela deslizante multiplicado a cada camada gerada pela pirâmide de imagens.

2.3 Conceitos de visão computacional

A visão computacional é a área da computação por meio da qual são estudas formas para que as máquinas tenham uma percepção próxima do mundo real de maneira similar ao sistema de visão humana [1].

Os seres humanos são criaturas dotadas de alta capacidade de visão, o que leva a pensar que estas tarefas são fáceis. O cérebro humano divide o sinal da visão em muitos canais que transmitem diferentes tipos de informação deste sinal. Nosso cérebro tem um sistema de atenção que identifica, de uma maneira dependente da tarefa, partes importantes de uma imagem para examinar enquanto ignora outras áreas. Há um *feedback* massivo na corrente visual que ainda é pouco compreendida. Existem amplas contribuições associativas dos sensores de controle muscular e de todos os outros sentidos que permitem ao cérebro recorrer a associações cruzadas feitas a partir de anos de experiência vivida. Os ciclos de retroalimentação no cérebro retornam a todos os estágios do processamento, incluindo os próprios sensores de hardware (os olhos), que controlam mecanicamente a iluminação através da íris e sintonizam a recepção na superfície da retina.

Em um sistema de visão mecânica, no entanto, um computador recebe uma matriz de números da câmera ou do disco, e é apenas esta informação que se tem em mãos. Na maioria das vezes, não há reconhecimento de padrões integrado, nem controle automático de foco e abertura, nem associações cruzadas, com anos de experiência vivida. Os ciclos de retroalimentação no cérebro retornam a todos os estágios do processamento, incluindo os próprios sensores de hardware (os olhos), que controlam mecanicamente a iluminação através da íris e sintonizam a recepção na superfície da retina.

Em um sistema de visão mecânica, no entanto, um computador recebe uma matriz de números da câmera ou do disco, e é apenas esta informação que se tem em mãos. Na maioria das vezes, não há reconhecimento de padrões integrado, nem controle automático de foco e abertura, nem associações cruzadas com anos de experiência. Na verdade, os sistemas de visão ainda são bastante ingênuos. Considere a imagem de um automóvel, enquanto se tenta detectar uma série de objetos que o compõe, o que o computador “vê” é apenas uma matriz de números. Qualquer número

dado dentro dessa matriz tem um componente de ruído bastante grande, o que dificulta ainda mais as tarefas de detecção e classificação computacional.

Os problemas descritos até aqui são formalmente impossíveis de resolver e, em adição, tem-se ainda outro agravante: a mesma imagem bidimensional poderia representar qualquer combinação infinita de cenas tridimensionais, mesmo se os dados fossem perfeitos. No entanto, como já mencionado, os dados podem ser corrompidos por ruídos e distorções. Tal corrupção decorre de variações no mundo (clima, iluminação, reflexos, movimentos), imperfeições na lente e configuração mecânica, tempo de integração finito no sensor, ruído elétrico no sensor ou outros componentes eletrônicos após captura de imagem.

Estes problemas só podem ser contornados a partir do conhecimento contextual do ambiente onde o sistema será implantado. Estas informações podem ser previamente modeladas assumindo certas condições e restrições ao domínio da aplicação ou também podem ser modeladas explicitamente com técnicas de aprendizado de máquina. Variáveis ocultas como tamanho, orientação da gravidade e assim por diante podem ser correlacionadas com seus valores em um conjunto de treinamento rotulado. Alternativamente, pode-se tentar medir variáveis de polarização ocultas usando sensores adicionais. O uso de um laser para medir a profundidade permite medir com precisão o tamanho de um objeto [19].

As aplicações da área de visão computacional são diversas, as mais conhecidas compreendem: controle de processos (robôs industriais, veículos autônomos e máquinas do tipo), modelagem de objetos ou ambientes e interação.

Embora os sistemas de visão computacional possam ser bastante distintos entre si, algumas etapas em comum podem ser citadas:

- Aquisição de imagem: compreende a obtenção de um sinal do mundo real por meio de sensores específicos capazes de gerar uma representação computacional do sinal de entrada;
- Pré-processamento: as imagens obtidas da etapa anterior podem não estar nas condições adequadas para etapas de processamento propriamente ditas, dessa forma, pode ser necessário ajustá-las de modo a evitar erros de processamento posteriores;

- Extração de características: características que podem ser expressas matematicamente podem ser extraídas em níveis de complexidade elevados;
- Detecção e segmentação: consiste em discriminar regiões de interesse e separá-las para posterior análise;
- Processamento de alto nível: consiste em verificar a qualidade dos dados, fazer estimativas sobre a imagem, bem como classificar objetos detectados em diferentes categorias.

O estudo dos métodos de visão computacional ainda é uma área em constante evolução, ainda existem muitos problemas não solucionados e que, portanto, permanecem em aberto [7], [20].

2.4 Raspberry Pi

A Raspberry Pi pode ser considerada um computador de baixo custo. Seu desempenho e dimensões (aproximadamente as mesmas de um cartão de crédito), permitem que muitos projetos de automação antes inviáveis, seja por custo de hardware mais potente, seja por incapacidade de microcontroladores comuns se tornassem possíveis [15].

A versão utilizada no presente trabalho, a saber, 3 modelo B, conta com quatro portas USB 2.0, quarenta pinos GPIO, saída de áudio, entradas ethernet, HDMI, cartão micro SD, para câmera e tela de vídeo, além de conexões Bluetooth e WiFi. Na Figura 2.8 pode ser vista uma fotografia da controladora.

Figura 2.7 - Fotografia de uma Raspberry Pi 3 model B

Fonte: https://www.raspberrypi.org/app/uploads/2017/05/Raspberry-Pi-3-Ports-1-1833x1080.jpg

2.5 Open CV

Imagens e vídeos estão inseridos por toda a parte, especialmente com a crescente inserção de dispositivos computacionais cada vez mais potentes e com dimensões reduzidas. A biblioteca aberta de visão computacional Open CV é um importante instrumento para o desenvolvimento de aplicações que requerem processamentos que envolvam algum tipo de função de visão computacional que tal biblioteca implementa.

A Open CV contém mais de 500 algoritmos otimizados para a análise de imagem e vídeo. Desde a sua introdução em 1999, tem sido largamente adotada como a principal ferramenta de desenvolvimento pela comunidade de pesquisadores e desenvolvedores em visão computacional. A biblioteca foi originalmente desenvolvida na Intel por uma equipe liderada por Gary Bradski como uma iniciativa para avançar na pesquisa de visão e promover o desenvolvimento de aplicativos robustos, baseados em visão computacional e uso intensivo da CPU. Após uma série de versões beta, a versão 1.0 foi lançada em 2006. Um segundo grande lançamento ocorreu em 2009 com o lançamento do OpenCV 2 que propôs mudanças importantes [19].

Atualmente a biblioteca possui interface com as linguagens C++, Python e Java, sendo suportada por sistemas Linux, Mac OS, Windows iOS e Android. A biblioteca

foi escrita na linguagem C++ de forma otimizada, inclusive para processamento em mais de um núcleo. Seu uso é direcionado aos mais diversos fins, que vão desde arte interativa até robótica avançada [21], [19].

2.6 Redes Neurais Artificiais

As redes neurais artificiais são um conjunto de algoritmos de aprendizado supervisionado utilizado para diversos desafios de inteligência artificial. Sua concepção é inspirada nas redes de neurônios do cérebro de animais, que aprendem através da experiência.

A motivação por trás do surgimento destes algoritmos surge do fato dos computadores serem extremamente rápidos e eficientes em várias áreas que exigem cálculos matemáticos que seres humanos levariam muito tempo para realizar, mas por outro lado são ineficientes no reconhecimento de padrões, especialmente aqueles obtidos por meio de visão [22].

Estas redes, devido a sua origem, compartilham uma série de características com o sistema nervoso humano:

- A menor unidade de processamento de informações ocorre em um conjunto de elementos simples chamados neurônios;
- Esses neurônios podem receber e enviam estímulos de e para outros neurônios ou para o ambiente;
- Informações/sinais são transmitidas entre neurônios através de conexões chamadas sinapses;
- A eficiência de uma sinapse, representada por um peso ou força associada, corresponde à informação armazenada no neurônio, e por consequência na rede;
- O conhecimento é adquirido do ambiente por um processo conhecido como aprendizado, que é basicamente responsável por adaptar as forças de conexão, ou valores de peso para o caso artificial, aos estímulos externos.

Dessa forma, o aprendizado se resume em encontrar as forças de conexões apropriadas para produzir padrões de ativação satisfatórios sob certas circunstâncias.

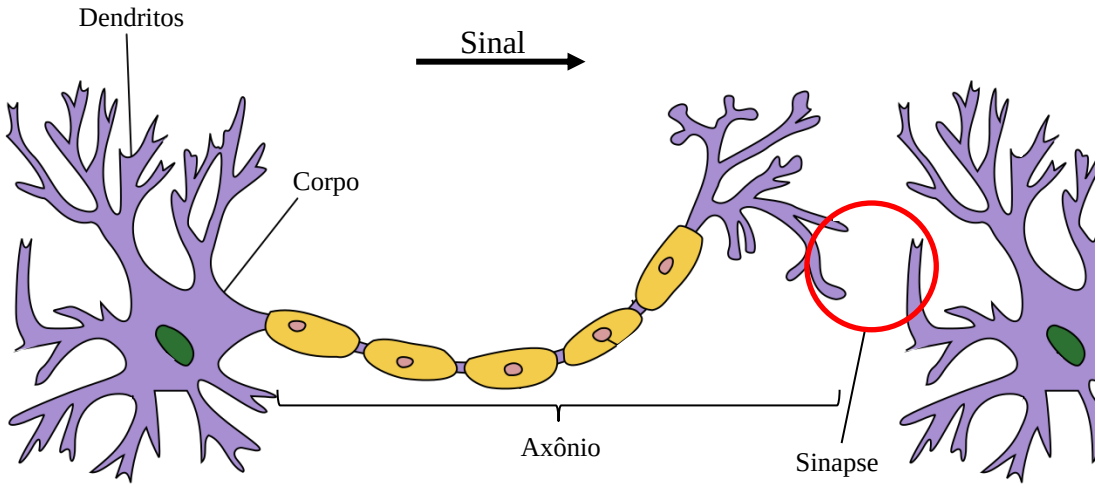
2.6.1 Neurônio Artificial

O funcionamento de um neurônio biológico ocorre da seguinte forma: o estímulo é recebido por canais localizados nas sinapses, permitindo que os íons fluam para dentro e para fora do neurônio. Um potencial de membrana aparece como resultado da integração das entradas neurais e, então determinará se um dado neurônio produzirá um pico (ação potencial) ou não.

Esse pico faz com que os neurotransmissores sejam liberados no final do axônio, formando sinapses com os dendritos de outros neurônios. O potencial de ação só ocorre quando o potencial de membrana está acima de um nível de limiar crítico. Entradas diferentes podem fornecer diferentes quantidades de ativação, dependendo de quanto neurotransmissor é liberado pelo emissor e quantos canais no neurônio pós-sináptico são abertos.

Uma ilustração de um neurônio biológico pode ser vista na Figura 2.9, onde suas principais partes citadas são apontadas.

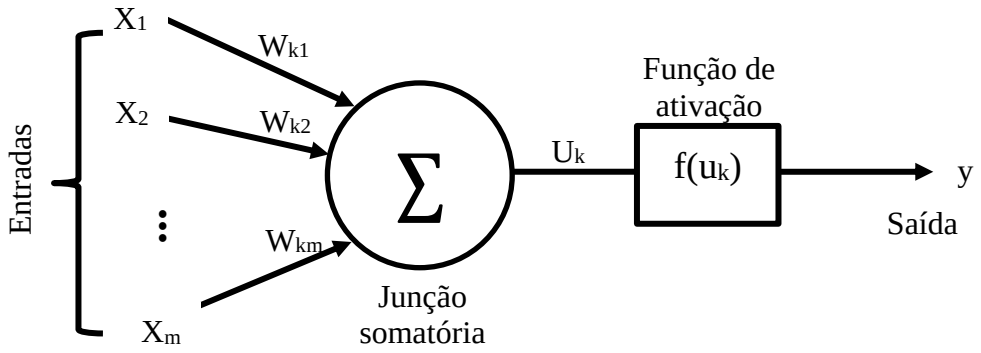
Figura 2.8 – Representação de neurônio biológico



Fonte: adaptado de https://pt.wikipedia.org/wiki/Axónio#/media/File:Neuron_Hand-tuned.svg

A partir desta representação, sucessivas melhorias no modelo de um neurônio artificial levaram a representação generalizada da Figura 2.10.

Figura 2.9 – Representação genérica de um neurônio artificial



Fonte: adaptada de [27].

Para cada um dos k neurônios, as entradas x_1, x_2, ..., x_m, são multiplicadas pelos respectivos pesos w_{k1}, w_{k2}, ..., w_{km}, e então somadas na junção somatória, a soma, denotada por u_k, é passada como entrada para uma função de ativação definida pelo projetista da rede, esta função gera um valor específico que é a saída final y do neurônio.

Um neurônio artificial também poder ser representado por uma função matemática, como a descrita na Equação 3.

$$y_k = f(u_k) = f\left(\sum_{j=0}^{m} w_{kj} x_j\right) \qquad 3$$

A saída y_k de um neurônio nada mais é que o resultado de sua função de ativação, que recebe como entrada um valor u_k, que por sua vez é o somatório das entradas multiplicadas pelos seus respectivos pesos.

2.6.2 Funções de ativação

As funções de ativação desempenham um papel importante nas redes neurais artificiais pois elas controlam a amplitude do sinal de saída de um neurônio. A escolha de função errada prejudica a eficiência de toda a rede, pois impede que ela seja capaz

de generalizar adequadamente os dados da fase de treinamento e, consequentemente, não seja capaz de classificar novos dados de maneira adequada.

Existe uma variedade de funções de ativação, algumas são utilizadas em camadas especificas de tipos específicos de redes neurais. Dessa forma, as principais funções são [22]:

- Função linear: a saída é igual a entrada.

$$f(u_k) = u_k \tag{4}$$

- Função limiar: a saída é um quando um limiar μ é atingido e zero caso contrário. Existe uma variação dessa função, conhecida como função bipolar, que retornam um e menos um ao invés de zero.

$$f(u_k) = \begin{cases} 1 & se\ u_k \geq \mu \\ 0 & caso\ contrário \end{cases} \tag{5}$$

- Função sigmoide: é estritamente crescente, apresentando saturação e equilíbrio entre comportamento linear e não linear. Possui forma de s e pode ser obtida por funções como a função logística, arco tangente ou tangente hiperbólica.

$$\text{Logística: } f(u_k) = \frac{1}{1+\exp(-u_k)}\ ,\ 0 \leq u_k \leq 1 \tag{6}$$

- Função de base radial: é uma função não monotônica que é simétrica ao redor de um valor base.

$$f(u_k) = \exp(-u_k^2) \tag{7}$$

- Função linear retificada: também conhecida como ReLU, é a função mais utilizada em redes neurais convolucionais. Uma das suas vantagens é a facilidade de otimização.

$$f(u_k) = \max\{0, u_k\} \tag{8}$$

- Função softmax: é uma generalização da função logística. Assim, como a função linear retificada, a função softmax é amplamente empregada em redes neurais convolucionais. É particularmente útil na classificação multiclasse pois normaliza as saídas entre zero e um. A probabilidade do elemento y_i de um vetor pertencer a uma classe depende do somatório de todos os elementos do vetor, e é o número de Euler.

$$f(u_k) = \frac{e^{y_i}}{\sum_j e^{y_j}} \quad (9)$$

2.6.3 Arquitetura e aprendizado de redes neurais

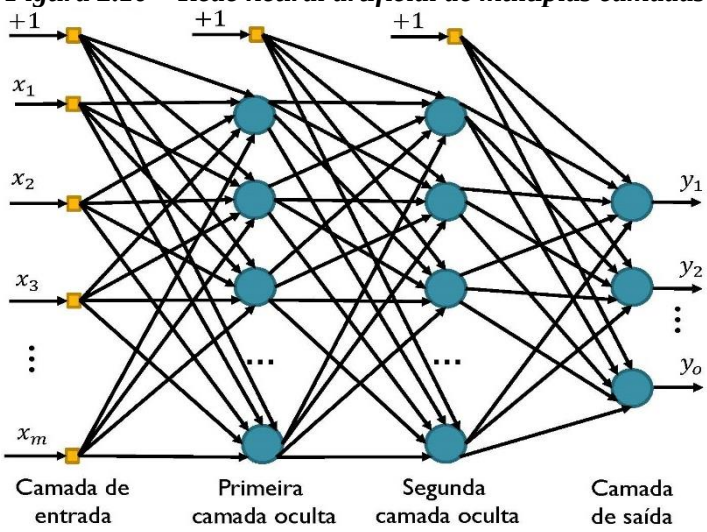
O funcionamento do sistema nervoso dos animais ainda não foi plenamente compreendido, não se sabe com riqueza de detalhes como as interconexões entre os neurônios são feitas. No entanto, sabe-se que há uma grande quantidade de neurônios que interagem entre si para realizar as tarefas cognitivas do indivíduo. Certamente um único neurônio isolado não seria capaz de realizar grandes feitos.

Por esse motivo, as redes neurais artificiais são organizadas em conjuntos de neurônios que são chamamos de camadas. Geralmente nomeia-se a primeira camada de uma rede como camada de entrada, responsável por receber o estímulo externo, uma ou mais camadas intermediárias ou ocultas, e uma camada de saída, que externa o resultado da rede.

Com exceção das redes recorrentes, as redes neurais, em geral, são do tipo *feedforward*, isto é, a propagação do sinal ocorre sempre da direção da entrada para a saída.

Na Figura 2.11 observa-se a ilustração de uma rede *feedforward* de múltiplas camadas.

Figura 2.10 – Rede neural artificial de múltiplas camadas



Fonte: Adaptado de [22]

Considerando uma rede neural como a apresentada na Figura 2.11, o processo de aprendizagem consiste em apresentar os padrões de entrada para a rede, isto é, os dados de treinamento. Estes dados permitem que a rede altere seus parâmetros livres de modo a produzir padrões de resposta desejados para os dados de entrada. É importante destacar que estes dados iniciais são rotulados, ou seja, sabe-se de antemão a qual classe o exemplo atual pertence, isto permite obter medidas de desempenho da rede. Esse processo de aprendizagem, com dados rotulados, é chamado de aprendizagem supervisionada. O desempenho da rede é medido ao se verificar a taxa de aprendizado, ou seja, a capacidade de generalização da rede.

Os parâmetros livres, em geral, são pesos ajustados por algum algoritmo de aprendizado. Se o treinamento for bem-sucedido, isto é, a rede é capaz de generalizar o padrão em questão, novos exemplos daquele padrão podem ser apresentados para que a rede possa agora os classificar.

2.7 Redes neurais convolucionais (CNNs)

As redes neurais convolucionais são atualmente um dos melhores classificadores de imagens conhecidos [23]. Enquanto nas RNAs tradicionais cada neurônio na camada de entrada é conectado a cada neurônio da camada de saída na camada seguinte, nas CNNs estas camadas, também conhecidas por camadas totalmente conectadas, são as últimas.

As camadas mais importantes são as camadas convolucionais, uma função de ativação não linear, como a função de base radial ReLU, é então aplicada à saída dessas convoluções e o processo de convolução, junto com uma mistura de outros tipos de camadas para ajudar a reduzir a largura e a altura do volume de entrada e ajudar a reduzir *overfitting*, que é a perda de capacidade de generalização da rede, é empregado até que, finalmente, se chegue ao final da rede e se aplique uma ou duas camadas completamente conectadas (também conhecidas como *fully connected* – FC) onde pode-se obter as classificações finais de saída.

Cada camada em uma CNN aplica um conjunto diferente de filtros, normalmente centenas ou milhares deles, e combina os resultados, alimentando a saída na próxima

camada da rede. Assim como nas redes tradicionais, ao longo do treinamento, uma rede neural convolucional aprimora automaticamente os valores desses filtros.

Existem dois benefícios principais no uso destas redes: invariância local e composicionalidade. O conceito de invariância local permite classificar uma imagem como contendo um objeto específico, independentemente de onde o objeto aparece na imagem. Obtemos essa invariância local através do uso de camadas de *pooling* (explicada em detalhes na subseção 2.7.2) que identifica regiões do volume de entrada com uma alta resposta a um filtro específico.

O segundo benefício é a composicionalidade. Cada filtro compõe um fragmento local de características de baixo nível em uma representação de alto nível, semelhante à forma como pode-se compor um conjunto de funções matemáticas que se baseiam na saída de funções anteriores: f (g (x (h (x))). Essa composição permite que a rede aprenda recursos mais ricos na rede. Por exemplo, ela pode construir arestas a partir de pixels, formas de arestas e objetos complexos a partir de formas - tudo de forma automatizada que ocorre naturalmente durante o processo de treinamento. O conceito de construir recursos de nível mais alto a partir dos de nível inferior é exatamente o motivo pelo qual as redes neurais convolucionais são tão poderosas na visão computacional.

Para construir uma CNN algumas camadas específicas são necessárias, estas serão descritas a seguir.

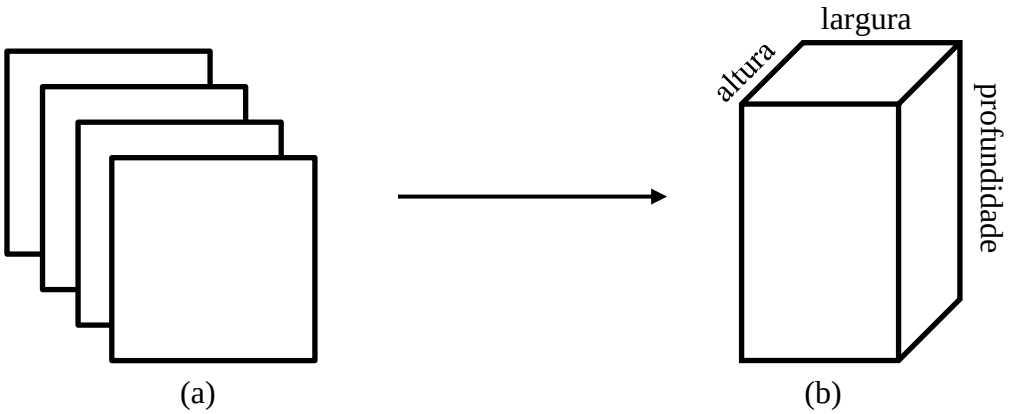
2.7.1 Camada convolucional

Esta é a camada central de toda CNN. A parte mais importante de todo processo está nesta camada pois é ela a responsável por aprender quais os filtros devem ser aplicados na imagem de entrada para que a classificação seja correta. Essa camada funciona como um extrator automático de características das imagens, de forma que as características inferidas pela rede neural são as mais relevantes para o processo de classificação.

Os parâmetros de entrada desta camada são a quantidade de filtros que podem ser aprendidos bem como suas dimensões, o tamanho do passo a ser dado na convolução, isto é, quantos pixels serão desconsiderados a cada iteração do operador.

Depois de aplicar todos os n filtros ao volume de entrada, tem-se n mapas de ativação bidimensionais, como pode ser visto na Figura 2.12 (a). Em seguida, empilha-se os mapas de ativação ao longo da dimensão de profundidade da matriz para formar a saída final na Figura 2.12 (b).

Figura 2.11 – Mapas de ativação empilhados para próxima camada



Fonte: adaptado de [28]

Cada entrada no volume de saída é uma saída de um neurônio que examina apenas uma pequena região da entrada. Dessa maneira, a rede aprende os filtros que são ativados quando eles veem um tipo específico de recurso em um determinado local espacial no volume de entrada.

Nas camadas inferiores da rede, os filtros podem ser ativados quando eles visualizam regiões semelhantes a bordas ou cantos, as quais são características de baixo nível da imagem, que são obtidas por meio de filtros que detectam bordas e cantos da imagem. Por outro lado, nas camadas mais profundas da rede, os filtros podem ser ativados na presença de características de alto nível, como partes do rosto, a pata de um cachorro, a frente de um carro, etc [23].

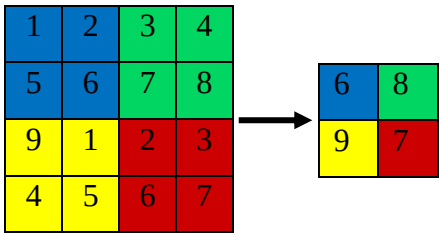
Tais elementos na abordagem tradicional de processamento de imagens digitais necessitariam que algoritmo específicos fossem implementados para realizar a detecção dessas características na imagem.

2.7.2 Camada de pooling

Para se resumir a quantidade de informações geradas ao final de uma camada de convolução utiliza-se uma camada de *pooling*. Esta camada diminui a dimensão da entrada dividindo-a em subconjuntos, onde apenas um valor resultante da operação escolhida irá compor a matriz de saída. Quando se escolhe o máximo entre os elementos do subconjunto, a camada é denominada *maxpooling*, se a operação for a média dos elementos, a camada será de *averagepooling*. Outras variações como *minpooling*, embora sejam teoricamente possíveis, dificilmente são utilizadas na prática.

Na Figura 2.13 pode-se observar um exemplo de funcionamento de uma camada de *pooling* máximo, em que cada janela de quatro pixels, torna-se o valor do maior pixel.

Figura 2.12 – Operação de pooling máximo



Fonte: elaborado pelo autor

2.7.3 Camada de *dropout*

A função de *dropout* é uma técnica recente que ajuda a controlar o sobreajuste da rede. A técnica consiste em desconectar aleatoriamente os valores de alguns neurônios da rede, a probabilidade para que isso ocorra geralmente é configurada em 0,5.

2.8 Aprendizagem profunda (*Deep Learning*)

Antes de expor o conceito de aprendizagem profunda, do inglês *deep learning*, é importante que se destaque em que área ela está inserida. Segundo [23], no conjunto de conhecimento relativo a inteligência artificial tem-se o subconjunto de aprendizado de máquina, que por sua vez possui o subconjunto de aprendizagem profunda.

Os algoritmos convencionais para reconhecimento e classificação de padrões ou objetos em imagens geralmente dependem de uma definição manual para quantificar e codificar um aspecto particular de uma imagem, os pixels servem apenas como entrada para o processo de extração de características, que por sua vez, resulta em um vetor de características, que é a entrada para o modelo de aprendizado de máquina.

Na abordagem de aprendizado profundo, especialmente com as redes neurais convolucionais, as características que antes eram definidas manualmente, agora são automaticamente aprendidas no processo de treinamento.

O sucesso dessa abordagem, principalmente para aplicações em visão computacional, decorre do fato de sua semelhança com o sistema visual humano. Usando o aprendizado profundo, tenta-se entender o problema em termos de uma hierarquia de conceitos. Cada conceito se baseia em cima dos outros. Conceitos nas camadas de nível inferior da rede codificam algumas representações básicas do problema, enquanto camadas de nível mais alto usam essas camadas básicas para formar conceitos mais abstratos. Esse aprendizado hierárquico permite remover completamente o processo de extração de características desenhado à mão e fazer as CNNs aprenderem a classificar uma imagem do começo ao fim [23].

Não há um consenso a respeito de quantas camadas uma rede neural deve conter para que seja considerada profunda, no entanto, é sabido que conforme a profundidade aumenta, também aumenta a precisão da classificação. A mesma afirmação é válida para o conjunto de dados de treinamento. Isso não ocorre em algoritmos de aprendizado de máquina tradicionais como por exemplo máquina de vetores de suporte (SVM) [23].

A necessidade de um grande conjunto de dados no aprendizado profundo é um fato. A eficiência do classificador pode ser prejudicada tanto pela quantidade quanto

pela qualidade dos dados apresentados a rede na etapa de treinamento. Para contornar esse problema aplica-se a técnica de aumento de dados sobre o conjunto.

Esta técnica permite que perturbações aleatórias sejam aplicadas às amostras originais de modo que novas amostras de mesmo rótulo sejam geradas. Esse procedimento permite que a generalização do modelo aumente, pois varia-se as amostras do conjunto.

No contexto de reconhecimento de placas de veículos por exemplo, se o conjunto de dados original não possuir placas de carro em diversos ângulos de rotação, provavelmente o modelo terá dificuldades em classificar novas placas nessa condição. Por este motivo, é interessante aplicar algumas transformações aleatórias sobre os dados, estas geralmente consistem em operações de recorte, rotação, translação, escala, distorções, filtros de aguçamento e suavização entre outros.

2.9 Single shot multibox detector

Uma rede neural convolucional é um classificador, ou seja, dada uma imagem de entrada, a rede é capaz de informar se esta entrada pertence ou não a uma das classes que a rede foi treinada para classificar. Em um sistema de detecção de placas de veículos, dificilmente teremos a posição e dimensão exatas de onde a placa está localizada na imagem. Dessa forma, é necessário que algum método adicional seja capaz de identificar, ou ao menos propor regiões onde existem possíveis regiões de interesse na imagem de entrada.

A combinação de pirâmide de imagens e janelas deslizantes é uma alternativa inviável se o objetivo for a detecção em tempo real. Em um hardware menos potente como uma *Raspberry Pi*, o tempo de execução seria ainda maior, impedindo que o sistema desse uma resposta em tempo hábil.

Em busca de um método capaz de realizar esta tarefa, diversas abordagens foram propostas [23]. A primeira, batizada de *region*-CNN (R-CNN), foi capaz de incorporar um algoritmo de busca seletiva para diminuir o número de possíveis regiões de interesse em uma imagem. Uma melhoria denominada *Fast* R-CNN tornou o processo significativamente mais rápido, mas não o suficiente, uma vez que a própria busca

seletiva era um gargalo. A abordagem *Faster* R-CNN eliminou a necessidade deste algoritmo, de modo que o mapa de recursos gerado pelas camadas convolucionais da rede era aproveitado para realizar a detecção.

Mesmo com estes avanços em relação ao desempenho, estes algoritmos não são, na prática, capazes de fornecer resultados em tempo real. Dessa forma, em [24] os autores propuseram uma abordagem denominada *Single Shot Multibox* Detector para esse fim.

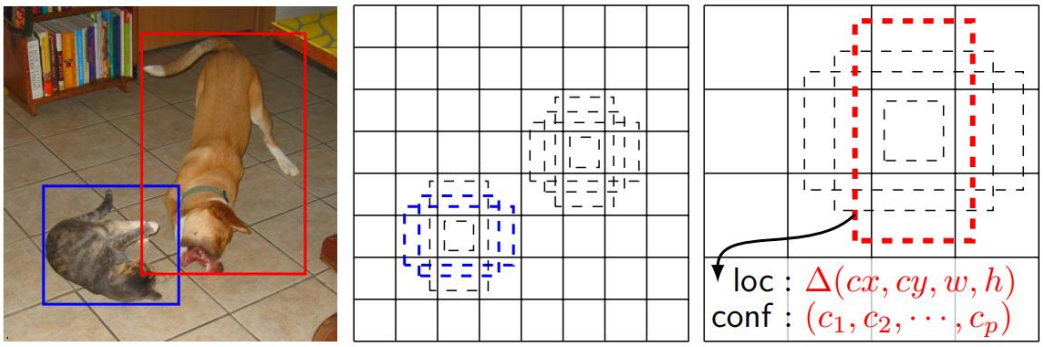
O termo disparo único, do inglês *single shot*, implica que tanto a localização quanto a detecção são realizadas em uma única passagem direta da rede durante o tempo de inferência, dessa forma, a rede precisa processar a imagem uma única vez para obter as previsões finais.

A arquitetura proposta consiste somente de uma rede neural convolucional que encapsula toda a computação dentro de si e que atuará não só como classificador, mas também como detector dos objetos. A primeira parte desta rede geralmente consiste em uma rede base, isto é, uma rede pré-treinada já conhecida como a do Visual Geometric Group (VGG), *ResNet*, *MobileNet* entre outras. Estas redes são classificadas como redes muito profundas pois possuem um número elevado de camadas e podem classificar centenas de classes.

O algoritmo *MultiBox* começa com antecedentes, que são caixas delimitadoras de tamanho fixo cujas dimensões foram pré-calculadas com base nas dimensões e localizações das caixas delimitadoras de referência para cada classe no conjunto de dados. O termo antecedentes decorre do fato que uma distribuição de probabilidades anterior é necessária para se determinar onde os objetos aparecerão na imagem. Os antecedentes são selecionados de forma que sua intersecção sobre a união com as caixas de referência seja maior que 50%.

Na Figura 2.14 tem-se uma ilustração do funcionamento dos antecedentes. A imagem da Figura 2.14(a) contém as caixas delimitadoras de referência. Deve-se gerar mapas de características que discretizam a imagem original em células, como nas imagens das Figuras 2.14(b) e 2.14(c).

Figura 2.13 – Geração de antecedentes da imagem de entrada



(a) (b) (c)

Fonte: retirada de [29]

Cada célula, em um mapa de características, possui quatro caixas delimitadoras padrão de proporções variadas. Ao discretizar a imagem de entrada em mapas de características de tamanhos diferentes, podemos detectar objetos em diferentes escalas na imagem.

No mapa de características de tamanho 8 x 8 da Figura 2.14(b) tem-se antecedentes pequenos em cada caixa delimitadora, eles são capazes de localizar pequenos objetos. Já no mapa de características de tamanho 4 x 4 da Figura 2.14(c), os antecedentes são maiores, permitindo que objetos maiores sejam localizados. Esta variação no tamanho do mapa de recursos é importante pois permite que objetos de tamanhos diferentes sejam localizados, como é caso do gato e do cachorro da Figura 2.14(a).

2.10 Sistemas de detecção e reconhecimento de placas de veículos

A literatura para sistemas de detecção e reconhecimento de placas de veículos é bastante ampla, e pode-se verificar que existem diversos trabalhos relacionados a este como por exemplo [2], [3], [4], [5], [25], [26], [6], [8], [9], [12], [13], [14], [27] e [28].

O estudo das técnicas de visão computacional aplicadas à detecção de placas de automóveis vem sendo cada vez mais estudada nos últimos anos [9]. Em cada uma das etapas que compõem o processo pôde-se encontrar várias abordagens para a resolução do problema.

Em [3] o método proposto consiste em converter a imagem inicial obtida para escala de cinza e então é aplicado o filtro da mediana para reduzir o ruído. O filtro da mediana é interessante nesse caso pois deseja-se que a imagem seja desfocada o mínimo possível para que o processo de reconhecimento dos caracteres não perca eficiência. Na etapa de detecção de bordas, os autores propõem o método de Canny para fazer a segmentação da imagem, este também é um método interessante em relação aos demais, uma vez que a presença de ruído dificilmente se torna uma falsa borda e o método possui uma boa taxa de detecção de bordas fracas [21]. Abordagens como as propostas em [25] e [29] também podem ser interessantes para este tipo de detecção. O próximo passo é a detecção da posição da placa do veículo. Primeiro uma função para preencher buracos é aplicada, fazendo com que a placa esteja entre os buracos preenchidos, a placa é selecionada procurando o buraco com maior área. Posteriormente os caracteres são segmentados por um processo semelhante ao anterior, onde cada caractere é separado por retângulos que os delimitam. Por fim, os caracteres são reconhecidos pela técnica de correspondência de modelos.

O método proposto em [17] também inicia com a conversão da imagem de entrada para níveis de cinza bem como a aplicação do filtro da mediana para a minimização de ruído. Os autores também aplicam a equalização de histograma para que a imagem se torne mais homogênea. Como o sistema proposto utiliza o vídeo para a aquisição de imagens foi necessário utilizar uma técnica de rastreamento de veículo para detectar a presença de veículos fazendo uma separação entre o objeto e plano de fundo. A localização da placa também é feita com o auxílio do método de detecção de bordas Canny. O método escolhido para a extração dos caracteres também foi a correspondência de modelos uma vez que as placas analisadas possuem caracteres do oriente médio que podem ser muito parecidos e facilmente confundidos por outros métodos.

Para a etapa de identificação de placa em meio a imagem adquirida, em [11] tem-se uma comparação entre diferentes wavelets para esse propósito, sendo a melhor delas, segundo as medições feitas pelos autores, a wavelet Biortogonal 1.3.

Uma abordagem diferente das demais pode ser vista em [29]: o pré-processamento é feito através da conversão da imagem em escalas de cinza e a atenuação do ruído por meio do filtro da mediana. A segmentação é feita pelo algoritmo de detecção de bordas Sobel. Os autores se destacam ao proporem para o reconhecimento de caracteres o uso de redes neurais de propagação reversa. Inicialmente a rede recebe dados de entrada para aprender as características dos caracteres que deverá reconhecer, esse processo ocorre até que a taxa de erro seja menor que um valor pré-determinado.

De início pode-se pensar que todo processo de reconhecimento de placas de veículos deve obrigatoriamente possuir uma etapa de segmentação para extrair a placa do veículo do resto da imagem, também é comum que a imagem seja convertida para escala de cinza antes de avançar para as próximas etapas. No método proposto em [26] uma rede neural convolucional (ConvNet) é utilizada para a extração de características da imagem e uma rede neural recorrente (RNN) é usada no sequenciamento. O único pré-processamento é redimensionar a imagem para 240x120 pixels, otimizando o tempo total de processamento. Dessa forma, as etapas até então utilizadas não são necessárias nessa abordagem, inclusive a segmentação.

2.11 Considerações finais

Neste capítulo foram apresentados os principais conceitos e tecnologias desse trabalho. O conceito de redes neurais convolucionais e os algoritmos que podem ser combinados para que a detecção das placas de veículos e caracteres ocorra formam a base desse trabalho. No próximo capítulo é exposto quais foram os passos para a implementação do sistema, compreendendo deste a preparação dos softwares básicos necessários para que a Raspberry operasse até o treinamento das redes neurais empregadas no sistema.

CAPÍTULO 3 - Implementação do sistema

3.1 Considerações iniciais

Neste capítulo serão apresentadas as etapas de desenvolvimento do sistema. Na seção 3.2 faz-se a descrição de como o ambiente proposto foi preparado. Na seção 3.3 apresenta-se a arquitetura do sistema bem como o fluxograma do seu funcionamento.

Para que se pudesse verificar o real funcionamento do sistema idealizado um protótipo foi construído, a descrição do seu funcionamento é feita na seção 3.3.

Uma vez que a porção mais importante de processamento do sistema são as redes neurais convolucionais, descreve-se na seção 3.4 os detalhes dos procedimentos para o seu treinamento.

3.2 Preparação do ambiente

A *Raspberry Pi* vem de fábrica sem nenhum software instalado. Por conta disso, deve ser instalado seu sistema operacional baseado em Linux, denominado *Raspbian,* no cartão SD para que a placa opere. É importante que o cartão de memória seja de alta velocidade para não comprometer o desempenho do sistema.

Algum dispositivo de aquisição de imagens deve ser conectado a placa. Para minimizar erros e problemas com relação a compatibilidade, a câmera do mesmo

fornecedor da *Raspberry* foi inserida. O software responsável pelo interfaceamento com a câmera, o *raspistill*, é parte do sistema operacional.

O modelo da rede neural convolucional foi implementado na biblioteca *TensorFlow*, sendo necessário que esta também fosse instalada na *Raspberry* para que o processo de reconhecimento fosse possível.

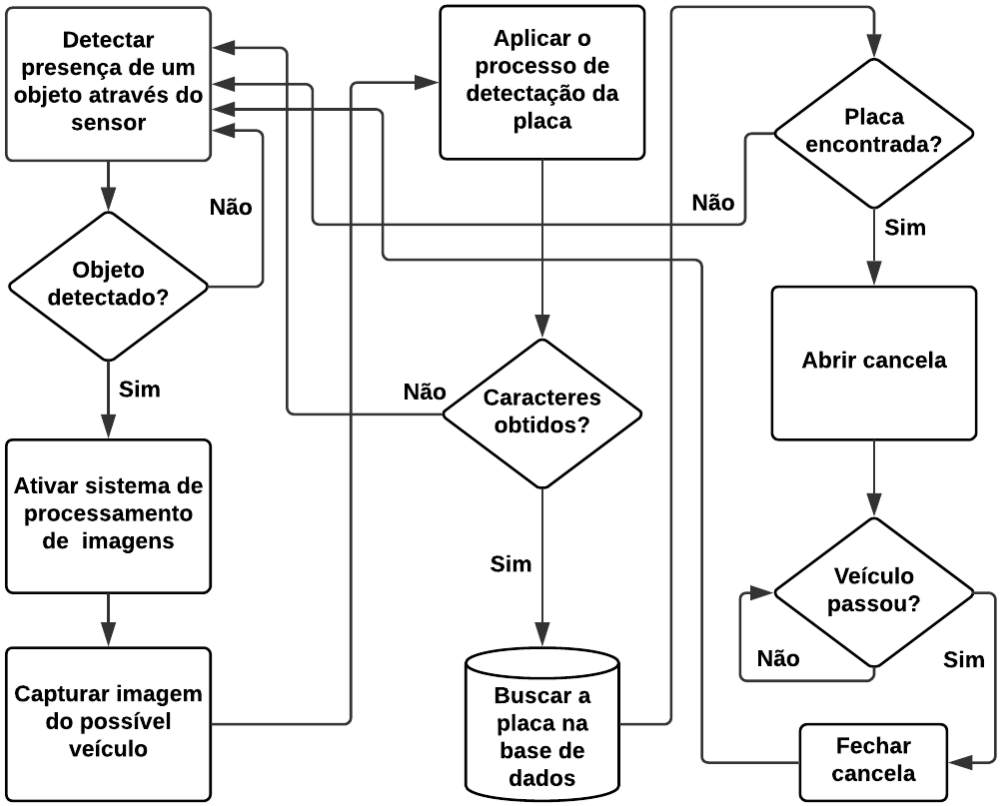
3.3 Arquitetura do sistema

Inicialmente, o sistema é mantido em modo de baixo consumo de energia, apenas os sensores de proximidade ficam em constante funcionamento. Quando estes detectam a proximidade de algum objeto o sinal é enviado para que a *Raspberry Pi* inicialize a câmera e fotografe a cena atual. Esta imagem é a entrada para o algoritmo de janela deslizante, que deverá iterar sobre a imagem para localizar a placa do veículo aplicando a classificação da rede neural.

É possível que o objeto detectado pelo sensor não seja necessariamente um veículo, no caso de animais ou folhas voando por exemplo, nessa situação a rede não detectará nenhuma placa, fazendo com que o sistema negue a passagem, uma vez que não há nenhum veículo. Quando um veículo for detectado o mesmo processo ocorre. Desta vez o processo de reconhecimento entrará em ação efetivamente e obterá os caracteres da placa em questão, posteriormente a cadeia de caracteres obtida é comparada com as placas autorizadas na base de dados. Se o veículo estiver cadastrado sua entrada será autorizada e a cancela abrirá.

O fluxograma representado na Figura 3.1 representa o funcionamento do sistema proposto.

Figura 3.1 - Diagrama do funcionamento do sistema



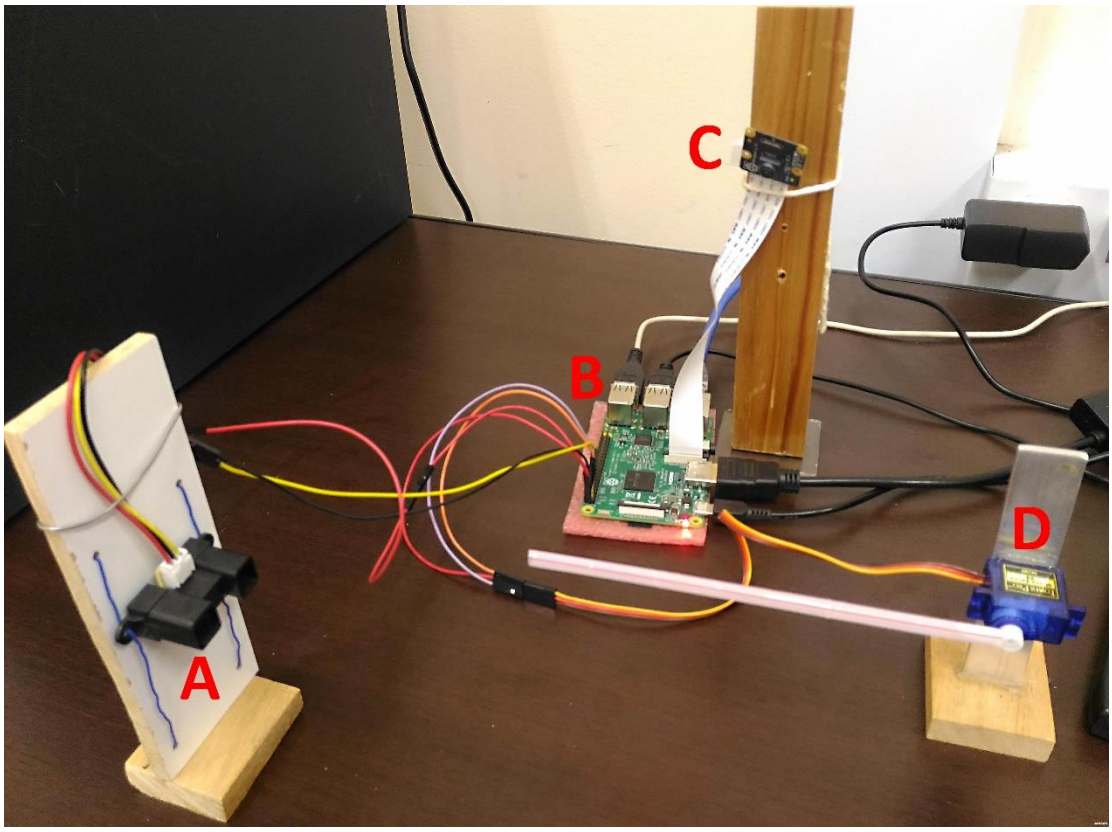
Fonte: elaborado pelo autor

3.4 Funcionamento do protótipo

O protótipo foi montado de modo a simular o posicionamento dos sensores em um sistema real. Para dar suporte aos três componentes externos empregados, foram construídas bases de sustentação com madeira, acrílico, metal e barbante de modo que nenhum componente fosse danificado.

Na Figura 3.2 podemos observar a disposição dos componentes físicos do protótipo do sistema.

Figura 3.2 – Disposição dos componentes físicos do sistema



Fonte: elaborado pelo autor

O componente com o rótulo A é um sensor de proximidade infravermelho. Enquanto este não detecta um objeto próximo o sinal zero é constantemente enviado a placa. Assim que um objeto passa pelo seu campo de detecção o sinal um é enviado.

Assim que recebe o sinal um, a placa, com o rótulo B, inicializa a câmera, com rótulo C para obter uma captura da cena atual. A imagem obtida é então processada pela rede neural a fim de localizar uma placa, caso haja sucesso a cadeia de caracteres gerada é então comparada na base de dados. Se a placa for encontrada, o servo motor, com o rótulo D move o canudo 90 graus para cima simbolizando que o veículo obteve permissão de acesso.

Neste protótipo o tempo em que a cancela permanece aberta foi fixado em cinco segundos. Após esse tempo o motor move o canudo para sua posição inicial. O sistema retorna ao estado inicial, esperando uma nova entrada.

3.5 Treinamento da rede neural

Para o treinamento da rede neural foram utilizadas duas bases de imagens de [27] e [28]. A base dos autores de [27] é denominada UFPR-ALPR e foi montada pelo laboratório Visão Robótica e Imagem da Universidade Federal do Paraná, ela contém 4500 imagens com resolução de 1920x1080 pixels de carros fotografados no trânsito. A base dos autores de [28] é denominada SSIG-SegPlate e foi construída pelo laboratório *Smart Sense* da Universidade Federal de Minas Gerais, ela possui 2500 imagens com resolução de 1920x1080 pixels de vários tipos de veículos fotografados em um ponto fixo de uma rua.

Cada imagem das bases de dados contém um arquivo texto com as localizações das placas e caracteres, estas informações foram utilizadas para gerar os recortes de exemplos de placas e não placas e exemplos dos 36 caracteres possíveis para compor os dados de treinamento e teste das CNNs. A Figura 3.3 é um exemplo de imagem obtida das bases de dados.

Figura 3.3 – Exemplo de imagem de treinamento

Fonte: base de imagens de [30]

Para a implementação do sistema proposto, é necessário que placas sejam extraídas de uma imagem e, posteriormente, caracteres desta placa sejam extraídos e reconhecidos. Assim, foram treinadas duas redes neurais. A primeira é responsável por classificar a imagem de entrada como uma placa ou algo que não seja uma placa e a segunda é responsável por identificar os caracteres.

A arquitetura final das duas redes, isto é, sua configuração, foi elaborada com base em modelos mais robustos como o da rede VGG16 [24]. Estes modelos são utilizados para fins de classificação mais genéricos do que os presentes neste trabalho, isso permitiu que fosse feita uma expressiva redução na quantidade de filtros e camadas da rede, tornando sua classificação ainda mais rápida.

A arquitetura da rede VGG 16 consiste em cinco blocos contendo de duas a três camadas convolucionais e uma camada de *maxpooling* ao final de cada um e um bloco com três camadas FC para a classificação final.

Fez-se a redução removendo gradativamente as camadas convolucionais e também a quantidade de filtros em cada uma delas. A cada alteração o processo de treinamento foi novamente realizado para garantir que não houvesse perdas significativas na detecção.

A arquitetura final após as reduções da primeira rede neural pode ser vista na Tabela 3.1.

Tabela 3.1 – Arquitetura da primeira rede neural

Camada	Neurônios / Filtros	Dimensão do Kernel	Passo	Camada de ativação
Convolucional	16	4x8	1	Relu
MaxPooling	-	2x2	2	-
Convolucional	32	2x4	1	Relu
MaxPooling	-	2x2	1	-
Planificação	-	-	-	-
Densa	700	-	-	Relu
Densa	350	-	-	Relu
Densa	2	-	-	Softmax

Fonte: Elaborado pelo autor

A segunda rede neural convolucional é a responsável por fazer o reconhecimento individual de cada caractere obtido na etapa anterior. São 36 classes de imagens de 0 a 9 e de A a Z. Sua arquitetura é praticamente a mesma da rede anterior, com exceção da última camada, que deve estar de acordo com o número de classes, ela está representada na Tabela 3.2.

Tabela 3.2 - Arquitetura da segunda rede neural

Camada	Neurônios / Filtros	Dimensão do Kernel	Passo	Camada de ativação
Convolucional	16	4x8	1	Relu
MaxPooling	-	2x2	2	-
Convolucional	32	2x4	1	Relu
MaxPooling	-	2x2	1	-
Planificação	-	-	-	-
Densa	700	-	-	Relu
Densa	350	-	-	Relu
Densa	36	-	-	Softmax

Fonte: elaborado pelo próprio autor

Nas Tabelas 3.1 e 3.2 a coluna Neurônios/Filtros contém o número de filtros para cada camada de convolução e o número de neurônios para cada camada densa.

Para as camadas convolucionais e de *pooling*, a dimensão do kernel é a definição do tamanho da janela que desliza sobre a imagem para realizar a operação de determinada camada.

A coluna de passo contém o valor do incremento que a janela dará a cada operação em uma camada convolucional ou de *pooling*. A coluna da camada de ativação descreve qual função de ativação foi utilizada ao final de cada camada que necessita desse recurso.

A última camada densa é a responsável por retornar o resultado da classificação. Ela deve conter exatamente o número de classes que a rede deve classificar e é importante que a função de ativação utilizada retorne probabilidades, como por exemplo a *softmax*. Dessa forma, para a primeira rede são necessários dois neurônios

ao final, uma vez que a rede classifica placas e não placas. Para a segunda rede são necessários 36 neurônios na camada final, pois a rede classifica 36 tipos diferentes de caracteres.

3.6 Considerações finais

Neste capítulo foi descrita a implementação do sistema proposto. A preparação consistiu em instalar os softwares e componentes de hardware na placa de forma que o protótipo se aproximasse de um sistema real. Foi feita a descrição da arquitetura e do funcionamento do protótipo, de modo a indicar o relacionamento entre os componentes físicos. Além disso, foi feita a descrição das bases de dados utilizadas e como os dados nelas contidos foram preparados para o treinamento da rede. O modelo de rede neural proposto foi projetado de modo a conciliar as dimensões de suas configurações com uma boa taxa de detecção. No próximo capítulo, os resultados do sistema proposto serão apresentados.

CAPÍTULO 4 - Resultados

4.1 Considerações iniciais

Neste capítulo serão apresentados os resultados referentes aos testes deste projeto.

Inicialmente, na seção 4.2 faz-se a descrição de como as imagens originais das bases de dados utilizadas foram preparadas para o treinamento das redes neurais.

Na seção 4.3 é descrita a metodologia dos testes, visando comprovar a eficiência do sistema quanto a classificação. Na seção 4.4 apresenta-se os resultados obtidos a partir dos testes.

4.2 Preparação dos dados

Para que os testes pudessem ser realizados foi necessário preparar os dados de entradas das redes neurais.

Primeiramente as duas bases foram combinadas em uma única base maior. Isto permite uma melhor generalização das redes visto que as imagens de cada base possuem algumas diferenças.

A princípio, na fase de treinamento, para que as redes pudessem aprender o que é uma placa e o que são os caracteres de uma placa de veículos, foram fornecidas apenas imagens já recortadas para a rede. Desse modo, tanto as placas quanto os caracteres foram recortados. Na Figura 4.1 observa-se alguns desses recortes.

Figura 4.1 – Exemplos de recortes

Fonte: elaborado pelo autor

A técnica de aumento de dados foi aplicada também com o intuito de variar as amostras da base combinada. As imagens da base não possuem uma boa variação no ângulo de rotação original. Dessa forma, uma das principais operações nesta etapa foi a rotação, uma vez que, num sistema real, a aquisição de imagens pode sofrer as mais diversas variações.

4.3 Metodologia dos testes

A validação cruzada foi utilizada para avaliar a capacidade de generalização das redes, a partir de um conjunto de dados disponível. Desta forma pôde-se estimar a precisão das redes neurais na prática, ou seja, o seu desempenho para um novo conjunto de dados.

Utilizou-se validação Cruzada 5 - folders, em que o conjunto de dados foi particionado em 5 subconjuntos mutualmente exclusivos, e posteriormente, foram utilizados 4 destes subconjuntos para o treinamento e o restante foi empregado na validação da rede. Este processo foi feito 5 vezes, com o rodizio do subconjunto utilizado para os testes, e os demais para o treinamento.

Com este procedimento pôde-se calcular a acurácia média do modelo e, portanto, aumentar a confiabilidade nos resultados pois além de embaralhar os dados, garantimos o rodízio dos mesmos.

Como as duas bases possuem os rótulos da localização da placa e dos seus caracteres foi possível treinar as duas redes separadamente.

A principal desvantagem em se utilizar redes neurais convolucionais é a necessidade de uma máquina com grande poder de processamento para o treinamento. Todas as imagens do conjunto devem ser carregadas em memória, a cada imagem apresentada as redes são mais de um milhão de parâmetros a serem otimizados em cada uma delas. Por esse motivo, é necessário que a máquina utilizada na fase de treinamento tenha grande quantidade de memória e um bom processador. Placas de vídeo, quando disponíveis, fornecem um excelente desempenho para este fim.

O computador utilizado para o treinamento das redes possui a seguinte configuração:

Processador Intel Core i7-4770K de 4ª geração com 8 núcleos de 3.5 Ghz cada;

Memória RAM DDR3 de 32 GB no total;

HD de 1 TB;

Sistema operacional Ubuntu 18.04.1

4.4 Resultados dos testes

O conjunto total de placas e não placas, incluindo as amostras geradas por aumento de dados, totalizaram 30183 itens. O conjunto de caracteres, considerando as 36 classes possíveis, incluindo as amostras com aumento de dados, somaram 35961 itens. Em ambos os casos a distribuição de amostras entre as classes foi aproximadamente a mesma.

Na rede responsável para o reconhecimento de placas, todas as entradas foram redimensionadas para 32 x 128 pixels. A acurácia média dos dados de treinamento foi de 99% com pequenas variações em cada *fold*. Foram necessárias 7 épocas de treinamento para atingir estes resultados. A acurácia média dos testes foi de 98% com

pequenas variações. Os valores obtidos em cada rodízio dos *folds* está representado na Tabela 4.1.

Tabela 4.1 - Resultado dos testes da rede de reconhecimento de placas

Fold	Acurácia de treinamento	Tempo	Acurácia de teste
1	99,84%	8,42 min	98,93%
2	99,41%	8,10 min	98,69%
3	99,42%	8,30 min	98,63%
4	99,62%	8,60 min	98,83%
5	90,49%	8,49 min	98,36%

Fonte: Elaborado pelo autor

Na rede responsável para o reconhecimento de caracteres, todas as entradas foram redimensionadas para 32 x 16 pixels. A acurácia média dos dados de treinamento foi de 99% com pequenas variações em cada *fold*. A invariância da acurácia permitiu que apenas uma época de treinamento fosse necessária. A acurácia média dos testes foi de 99% com pequenas variações. Os valores obtidos em cada rodízio dos *folds* está representado na Tabela 4.2.

Tabela 4.2 - Resultado dos testes da rede de reconhecimento de caracteres

Fold	Acurácia de treinamento	Tempo	Acurácia de teste
1	99,65%	45,43 s	99,54%
2	99,65%	45,59 s	99,61%
3	99,70%	45,86 s	99,64%
4	98,65%	44,93 s	99,61%
5	98,66%	45,01 s	90,62%

Fonte: Elaborado pelo autor

Apesar da performance dos algoritmos propostos, o método não pôde ser efetivamente implantado na Raspberry pois, como exposto na seção 2.2.6 a combinação do método de janela deslizante e pirâmide de imagens é lento. Os testes realizados na máquina de treinamento levavam em torno de um minuto para iterar sobre a imagem uma única vez, desconsiderando os redimensionamentos da pirâmide de imagens.

Considerando que o poder de processamento da Raspberry é consideravelmente menor, podemos afirmar com segurança que aplicar o mesmo algoritmo na placa levaria um tempo ainda maior, impossibilitando que a cancela fosse aberta em tempo hábil.

4.5 Considerações finais

Neste capítulo foram apresentados os resultados obtidos com os métodos implementados. As duas redes neurais empregadas obtiveram uma alta taxa de acertos. O algoritmo de janela deslizante para procurar por objetos de interesse na imagem de entrada, embora apropriado para encontrar objetos de interesse na imagem, mostrou-se inadequado para um sistema de tempo real, inviabilizando a sua inserção na Raspberry. No próximo capítulo as considerações finais desse projeto são apresentadas.

CAPÍTULO 5 - Conclusões

5.1 Conclusões gerais

Os sistemas de detecção de placas de carro continuam a ser um assunto importante e desafiador nas pesquisas de visão computacional. Diversos trabalhos atuais puderam ser encontrados, o que mostra a importância do tema. Diferentes métodos e abordagens para a implementação desses sistemas foram propostos na literatura, alguns com maiores taxas de acerto, porém com tempo de processamento também dispendioso, outros possuem uma taxa de detecção menor, porém também são menos custosos computacionalmente.

O sistema desenvolvido obteve bons resultados na classificação de placas de veículos e seus caracteres utilizando redes neurais convolucionais, uma nova tecnologia que vem sendo utilizada na detecção de objetos e classificação de imagens digitais. Embora estes algoritmos exijam computadores com alto poder de processamento, grandes quantidades de memória RAM e muitos exemplos de imagens na fase de treinamento em relação aos demais, os resultados obtidos compensam essa necessidade.

Se o tempo de execução do algoritmo de janela deslizante e pirâmide de imagens não fosse impeditivo, certamente o sistema poderia ser utilizado para controlar a entrada de forma automatizada de veículos nos mais diversos ambientes e situações.

A utilização de sensores de proximidade com o intuito de poupar o sistema de estar completamente ligado a todo tempo permite que energia seja economizada e que

os componentes, em especial a câmera, não sofram desgaste desnecessário estando constantemente ligados.

Dessa forma, o protótipo do sistema desenvolvido não obteve sucesso pleno devido a escolha do algoritmo de busca da placa na imagem, por ser demasiadamente custoso em termos de processamento . Mesmo assim, pôde-se concluir que as redes neurais convolucionais obtiveram altas taxas de acerto nas tarefas de classificação em que foram empregadas, sendo altamente recomendadas para este e outros tipos de classificação.

5.2 Dificuldades encontradas

As principais dificuldades encontradas foram a compreensão dos algoritmos envolvendo as redes neurais convolucionais e a busca por um método que fosse capaz de operar em tempo real sobre a imagem de entrada na arquitetura da *Raspberry*. A literatura apresenta algumas possíveis soluções, entre elas o método *Single Shot multibox Detector* que seriam capazes, na teoria, de fazer com que o sistema operasse em tempo real, no entanto, estes algoritmos são complexos tanto para a compreensão quanto para a implementação, mesmo utilizando linguagens de alto nível como Python e bibliotecas de suporte.

5.3 Trabalhos futuros

A ineficiência do método implementado automaticamente para a detecção da placa em uma imagem gera a necessidade da sua substituição. Trabalhos futuros teriam como foco principal a alteração deste método de janelas deslizantes por um método mais eficiente, em termos de tempo de processamento. Um forte candidato seria o método Single Shot Detector.

Outra melhoria em potencial para o sistema consiste em elaborar uma aplicação móvel híbrida para servir de interface entre o sistema e possíveis usuários responsáveis

pelo seu controle. Esta interface permitiria cadastrar novos usuários, fazer consultas, gerar relatórios e inclusive abrir a cancela incondicionalmente em caso de defeitos no sistema.

Sabe-se que o formato das placas brasileiras em breve mudará para um padrão de placas do Mercosul. Uma outra contribuição consiste em adicionar amostras dessas novas placas aos conjuntos de treinamento das redes neurais para que o sistema seja capaz de reconhecer os dois formatos de placas. No entanto, ainda não existem bases de dados disponíveis de placas do padrão Mercosul dado que poucos carros circulam atualmente com elas. Com o tempo, a elaboração de uma base de imagens desse tipo também será uma contribuição importante.

Referências Bibliográficas

[1] GONZALEZ, Rafael C.; WOODS, R. C. Processamento digital de imagens. tradução: Cristina Yamagami e Leonardo Piamonte. 2010.

[2] ABIRAMI, N.; JASMINE, JS Leena. Accurate Vehicle Number Plate Recognition And Real Time Identification Using Raspberry Pi. In: **International Research Journal of Engineering and Technology.** IRJET, 2018.

[3] AGARWAL, Anu; GOSWAMI, Sudhir. An Efficient Algorithm for Automatic Car Plate Detection & Recognition. In: **Computational Intelligence & Communication Technology (CICT), 2016 Second International Conference on**. IEEE, 2016. p. 644-648.

[4] XIE, Lele; AHMAD, Tasweer; JIN, Lianwen; LIU, Yuliang; ZHANG, Sheng. A New CNN-Based Method for Multi-Directional Car License Plate Detection. **IEEE Transactions on Intelligent Transportation Systems**, v. 19, n. 2, p. 507-517, 2018.

[5] BARNOUTI, Nawaf Hazim; NASER, Mustafa Abdul Sahib; AL-DABBAGH, Sinan Sameer Mahmood. Automatic Iraqi license plate recognition system using back propagation neural network (BPNN). In: **New Trends in Information & Communications Technology Applications (NTICT), 2017 Annual Conference on**. IEEE, 2017. p. 105-110.

[6] CALDERON, José Alejandro Franco; VARGAS, Javier Soto; PÉREZ-RUIZ, Alexander. License plate recognition for Colombian private vehicles based on an embedded system using the ZedBoard. In: **Robotics and Automation (CCRA), IEEE Colombian Conference on**. IEEE, 2016. p. 1-6.

[7] NORVIG, Peter; RUSSELL, Stuart. **Inteligência Artificial: Tradução da 3a Edição**. Elsevier Brasil, 2014.

[8] WIJNHOVEN, Rob G. J.; DE WITH, Peter H. N.. Identity verification using computer vision for automatic garage door opening. **IEEE Transactions on Consumer Electronics**, v. 57, n. 2, 2011.

[9] AALSALEM, Mohammed Y.; KHAN, Wazir Zada; DHABBAH, Khalid Mohammed. An automated vehicle parking monitoring and management system using ANPR cameras. In: **Advanced Communication Technology (ICACT), 2015 17th International Conference on**. IEEE, 2015. p. 706-710.

[10] LIN, L. R.; HSU, G. H.; JAN, R. H.; CHEN, C.. A novel non-payment vehicle searching method for multilane-free-flow electronic-toll-collection systems. In: **Advanced Communication Technology (ICACT), 2012 14th International Conference on**. IEEE, 2012. p. 904-909.

[11] TJANDRA, Lundy Orlando; NUGROHO, Saptadi; UTOMO, Darmawan. Electronic road pricing system prototype. In: **Technology of Information and Communication (ISemantic), International Seminar on Application for**. IEEE, 2016. p. 126-129.

[12] MCDONALD, Gregor J. et al. Real-Time Vehicle Identification Performance Using FPGA Correlator Hardware. **IEEE Trans. Intelligent Transportation Systems**, v. 13, n. 4, p. 1891-1895, 2012.

[13] TASHK, Ashkan; HELFROUSH, MohammdSadegh; KARIMI, Vahid. An automatic traffic control system based on simultaneous Persian license plate recognition and driver fingerprint identification. In: **Telecommunications Forum (TELFOR), 2012 20th**. IEEE, 2012. p. 1729-1732.

[14] WEBER, Henrique; JUNG, Claudio Rosito. Low-cost license plate detection using a calibrated camera. In: **Image Processing (ICIP), 2015 IEEE International Conference on**. IEEE, 2015. p. 4763-4767.

[15] Raspberry Pi 3 Model B. Disponível em: www.raspberrypi.org/products/raspberry-pi-3-model-b. Acesso em: 02 julho 2018.

[16] Image Pyramids with Python and OpenCV. Disponível em: <https://www.pyimagesearch.com/2015/03/16/image-pyramids-with-python-and-opencv>. Acesso em: 10 outubro 2018.

[17] MARQUES FILHO, Ogê; NETO, Hugo Vieira. **Processamento digital de imagens**. Brasport, 1999.

[18] Sliding Windows for Object Detection with Python and OpenCV. Disponível em: <https://www.pyimagesearch.com/2015/03/23/sliding-windows-for-object-detection-with-python-and-opencv/>. Acesso em: 10 outubro 2018.

[19] BRADSKI, Gary; KAEHLER, Adrian. **Learning OpenCV: Computer vision with the OpenCV library**. " O'Reilly Media, Inc.", 2008.

[20] PITERI, Marco Antônio; RODRIGUES, José Carlos. Fundamentos de visão computacional. **Presidente Prudente: FCT/UNESP-PP**, 2011.

[21] Find edges in intensity image - MATLAB edge. Disponível em: <https://www.mathworks.com/help/images/ref/edge.html>. Acesso em: 02 julho 2018.

[22] DE CASTRO, Leandro Nunes. **Fundamentals of natural computing: basic concepts, algorithms, and applications**. Chapman and Hall/CRC, 2006.

[23] ROSEBROCK, ADRIAN. **Deep Learning for Computer Vision with Python.** 1. ed. New York: Pyimageseach, 2017.

[24] LIU, Wei; ANGUELOV, Dragomi; ERHAN, Dumitru; SZEGEDY, Christian; REED, Schott; FU, Cheng-Yang; BERG, Alexander C.. **Ssd: Single shot multibox detector**. In: European conference on computer vision. Springer, Cham, 2016. p. 21-37.

[25] BOAVENTURA, Ines A. Gasparotto; GONZAGA, Adilson. Border detection in digital images: An approach by fuzzy numbers. In: **isda**. IEEE, 2007. p. 341-346.

[26] CHEANG, Teik Koon; CHONG, Yong Shean; TAY, Yong Haur. Segmentation-free Vehicle License Plate Recognition using ConvNet-RNN. **arXiv preprint arXiv:1701.06439**, 2017.

[27] LAROCA, Rayson; SEVERO, Evair; ZANLORENSI, Luiz A.; OLIVEIRA, Luiz S.; GONÇALVES Gabriel R.; SCHWARTZ, William R.; MENOTTI, David. **A Robust Real-Time Automatic License Plate Recognition based on the YOLO Detector**. arXiv preprint arXiv:1802.09567, 2018.

[28] GONÇALVES, Gabriel R.; SILVA, Sirlene P. G.; MENOTTI, David; SCHWARTZ, William R.. **Benchmark for license plate character segmentation**. Journal of Electronic Imaging, v. 25, n. 5, p. 053034, 2016.

[29] BOAVENTURA, Inês AG; GONZAGA, Adilson. Edge detection in digital images using fuzzy numbers. **International Journal of Innovative Computing and Applications**, v. 2, n. 1, p. 1-12, 2009.

[30] BARNOUTI, Nawaf Hazim; NASER, Mustafa Abdul Sahib; AL-DABBAGH, Sinan Sameer Mahmood. Automatic Iraqi license plate recognition system using back propagation neural network (BPNN). In: **New Trends in Information & Communications Technology Applications (NTICT), 2017 Annual Conference on**. IEEE, 2017. p. 105-110.