UNIVERSIDADE FEDERAL DA BAHIA - UFBA
INSTITUTO DE MATEMÁTICA - IM
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

LOCALIZAÇÃO DE PLACA VEICULAR COM BASE EM COVARIÂNCIA E REDES NEURAIS ARTIFICIAIS

TIAGO SILVA ARAÚJO

Salvador
2016

LOCALIZAÇÃO DE PLACA VEICULAR COM BASE EM COVARIÂNCIA E REDES NEURAIS ARTIFICIAIS

Tiago Silva Araújo

Monografia apresentada ao Curso de Sistemas da Informação, Departamento de Ciência da Computação, Instituto de Matemática e Estatística, Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Luciano Oliveira.

Salvador

2016

LOCALIZAÇÃO DE PLACA VEICULAR COM BASE EM COVARIÂNCIA E REDES NEURAIS ARTIFICIAIS

Tiago Silva Araújo

Monografia apresentada ao Curso de Sistemas de Informação, Departamento de Ciência da Computação, Instituto de Matemática e Estatística, Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Aprovado em 31 de Outubro de 2016

Banca Examinadora:

Prof. Dr. Luciano Rebouças de Oliveira (Orientador)
UFBA

M.Sc. Luiz Otávio de Oliveira Souza Júnior
UFBA

M.Sc. Kalyf Abdalla Buzar Lima
UFPI

À minha família

Agradecimentos

Agradeço a Deus por me dar o que preciso em minha caminhada. Nasci e fui criado em uma grande e amorosa família; sou indizivelmente grado por isso. Agradeço a Deus por me dar o poder, mesmo em minha ignorância, de imaginar sua grandeza e beleza. Agradeço a Deus por tudo.

Muito de minha capacidade, conhecimento e caráter devo à minha família, minha mãe Nádia, meu pai Ailton, que sempre me guiaram com atenção, carinho e responsabilidade. Sempre foram presentes, e participaram de minha vida com muito empenho, dedicação e amor. Meu irmão, Maurício, meu amigo, foi companheiro de infância, e hoje um exemplo de profissional e caráter. Foram essenciais no caminhar de minha graduação e no que sou.

Agradeço imensamente à minha esposa, Gracielle, pelo amor, companheirismo, compreensão e suporte que muito me ajudaram para a realização deste trabalho, assim como em toda minha graduação e em nossas vidas.

Agradeço às minhas filhas, Maria Eduarda e Amanda, por me presentearem com muito do todo amor que trago em meio peito, o que me confere maior precaução e atenção, com elas, e comigo mesmo. Me ensinaram também a olhar para a vida sob a perspectiva do amor e do tempo.

Dedico esse trabalho também aos meus familiares, em especial a meus avós, por todo o amor e atenção, e por terem construído nossas famílias dessa maneira. Aos tios e tias que são exemplos de caráter e dedicação.

Sou muito grato ao meu orientador, professor Luciano, por me apresentar uma área de estudo que colocou em meus olhos o brilho infantil da descoberta de algo novo e encantador. Agradeço a ele também por se mostrar sempre pronto a responder às minhas dúvidas e por reger a realização deste trabalho.

"Nem tudo que parece ser, é; mas
tudo que é aparenta ser."
Ditado popular.

Resumo

Um sistema de reconhecimento automático de placa veicular em imagens digitais normalmente é dividido conforme os principais problemas a serem resolvidos: a localização da placa; a segmentação da placa; e o reconhecimento dos caracteres. A localização da placa visa obter a região de interesse dentre os vários objetos presentes na imagem. A segmentação da placa, subdivide a região de interesse e separa os caracteres encontrados. O reconhecimento dos caracteres é o objetivo final, e tem como resposta a cadeia de caracteres que constitui a identificação do veículo. Neste trabalho propomos um método que resolva o primeiro problema: localizar placa de carro.

A localização de placa veícular tem sido alvo de muitos estudos de reconhecimento de padrão em imagens digitais. A dificuldade em reconhecer placas em ambientes não controlados, onde ocorrem grandes variações de luminosidade e outros ruídos, têm motivado o surgimento de vários métodos como proposta para resolver o problema. Neste trabalho, descrevemos as principais abordagens de processamento de imagens utilizadas em sistemas de localização de placas de veículos encontradas na literatura, e propomos um método utilizando um descritor de característica baseado na covariância dos gradientes e dos canais do espaço de cor RGB [1].

O método proposto neste trabalho utiliza informações estatísticas dos canais R, G e B, e dos gradientes horizontais e verticais, além de uma rede neural artificial de camadas múltiplas com retropropagação do erro, para classificar as subimagens. Uma janela deslizante percorre imagens obtidas de uma câmera; extrai os canais R, G e B, e os gradientes horizontais e verticais; calcula suas covariâncias; e as utiliza para treinar a rede neural artificial, na fase de treinamento. Na fase de teste, novamente são extraídos os canais de cor e gradientes de cada subimagem analisada pela janela deslizante. O descritor de covariâncias das características é utilizado para obter um resposta da rede neural: próximo a 1, se a região for uma placa; próximo a -1, caso contrário.

O método foi testado em ambiente real, em que uma câmera acessada via protocolo IP [2] provê as imagens de veículos, e o localizador proposto analisa os *frames* subsequentes

[1] Do inglês, *Red, Green, Blue*

[2] Do inglês, *Internet Protocol*

a uma detecção de movimento, destacando as placas encontradas. Para avaliar o classificador utilizamos as métricas taxa de acerto, taxa de alarme falso, exatidão e precisão. Foi obtida uma taxa de acerto de 84%. Com base nos resultados, consideramos que o objetivo deste trabalho foi alcançado.

Palavras-chave: Localização de placa de veículo. Visão computacional. Reconhecimento de padrão em imagem.

Abstract

A system of automatic recognition of licence plate in digital images is usually divided according to the main problems to be solved: the location of the licence plate; the segmentation of the licence plate; and recognition of the characters. The location of the licence plate aims to obtain the region of interest among the various objects in the image. The segmentation of the vehicle plate, subdivides the region of interest and separates the founds characters. The character recognition is the ultimate goal, and its response to the string which is the identification of the vehicle. In this paper we propose a method that solves the first problem: locate the car licence plate.

The location of the licence plate has been the target of many pattern recognition studies in digital images. The difficulty in recognizing license plates in uncontrolled environments, where there are large luminance variations and other noise, have motivated the emergence of several methods proposed to solve the problem. In this work we describe the main image processing approaches used in license plate localization systems found in literature, and we propose a method using a characteristic descriptor based on the covariance of the gradients and the color space of the RGB channels.

The method proposed here uses the statistical information of the R, G and B, and horizontal and vertical gradients, furthermore an artificial neural network of multiple layers with back propagation of error, to sort the subimages. A sliding window covers images taken from a camera; extracts R, G and B channels, and horizontal and vertical gradients; calculate its covariance; and used to train artificial neural network, the training phase. In the test phase, each subimage analyzed by sliding window, the color channels and gradients are extracted. The covariance descriptor of the features is used to obtain a response from the neural network: close to 1, if the region is one license plate; close to -1, otherwise.

The method was tested in a real environment, wherein a camera accessed by IP (Internet Protocol) provides the vehicle images, and the proposed locator analyzes the subsequent *frames* for a motion detection, highlighting the plaques found. To evaluate the classifier we use the hit rate metric, false alarm rate, accuracy and precision. for which values were obtained. It was obtained an hit rate of 84%. Based on these results,

we consider that the objective was achieved.

Keywords: License plate localization. Computer vision. Pattern recognition image.

Lista de Tabelas

Lista de Figuras

Sumário

Capítulo 1

INTRODUÇÃO

As pesquisas realizadas na área de inteligência artificial têm produzido conhecimentos de grande utilidade para os diversos âmbitos da sociedade. Nesse contexto surgem sistemas para, de forma automática, resolver problemas que por muito tempo só foi possível resolver com intervenção humana. O reconhecimento automático de placa de veículos (RAPV ou ALPR[1]), utilizando técnicas de reconhecimento de padrão em imagens, é um exemplo de problema para o qual se produz muito conhecimento, em busca de soluções satisfatórias, isto é, que aliem uma boa taxa de acerto a um bom desempenho.

O RAPV pode ser dividido em três problemas principais: a localização da placa, a segmentação da placa e o reconhecimento dos caracteres. Para Saha et al. [34], a localização da placa é a etapa mas importante e difícil de um sistema RAPV. Naturalmente, a identificação final do veículo depende do acerto na localização da placa.

A localização da placa de veículos em imagem tem motivado muitas abordagens de visão computacional. Neste texto descrevemos os principais métodos utilizados para localizar a placa, como a detecção de bordas, utilizado por Mohamed et al.[25], o descritor de bordas e RGB, utilizado por Mahini et al.[21]; o algoritmo Adaboost [39], utilizado por Dlangnecov [12]; as janelas concêntricas deslizantes, abordadas por Anagnostopoulos et al.[4]; o cálculo de média dos valores de gradientes, de Oliveira e Gonzaga [27]; a análise de componentes principais, utilizado por Passos et al. [29]; o descritor de covariância [30] e a operação morfológica de fechamento, método utilizado por Conci et al. [8]. Por fim, descrevemos o método do descritor de característica baseado em covariância de gradientes e RGB, utilizado neste trabalho para implementar um sistema de localização automática de placa de veículos (LAPV). Como classificador, optamos por uma rede neural artificial (RNA) de camadas múltiplas com treinamento *backpropagation* para realizar a classificação das janelas. Os resultados mostraram que o sistema é robusto a ruídos e outros fatores externos, como variações na iluminação, e que é possível utilizá-lo em situação real

[1]Do inglês, *automatic licence plate recognition* (ALPR).

após ajustes de configuração, e nos parâmetros de treinamento da rede neural artificial.

1.1 MOTIVAÇÃO

Um sistema para reconhecimento de placa de veículos é de grande utilidade para uma empresa, ou ambiente residencial, uma vez que ajuda a identificar e controlar o acesso de veículos nas áreas monitoradas. É comum encontrar tais sistemas em empresas e condomínios cujos moradores possuem um alto poder aquisitivo. No entanto, a implantação de tal serviço se torna viável, por exemplo, em um condomínio organizado ou estabelecimento comercial de pequena ou média empresa.

O acesso de veículos nas unidades da Universidade Federal da Bahia (UFBA) é feita hoje de maneira indiscriminada, ou utilizando mecanismos rudimentares de identificação. De tempos em tempos, o controle de acesso em alguns campi da UFBA é feito com a apresentação do comprovante de matrícula e um documento de identificação pessoal com foto. Este método costuma ser adotado durante um curto período de tempo, e após alguma ocorrência local que motive o controle, sendo abandonado após meses ou dias, quando então o acesso volta a ser permitido sem identificação.

O projeto apresentado neste texto tem por motivação contribuir com uma ferramenta automática de monitoramento e identificação dos veículos que acessam as unidades da UFBA, melhorando a segurança nos ambientes da universidade, e criando possibilidades de automatização de processos dos serviços prestados para a comunidade.

1.2 OBJETIVOS

Este trabalho tem por objetivo principal propor um método de localização de placa veicular em imagem digital obtida por câmera IP. O projeto deve localizar, destacar e persistir as placas de veículos que acessem o portão principal da Escola Politécnica da UFBA. Para isso, uma câmera IP foi instalada na guarita principal, provendo as imagens em tempo real. A câmera deve ser acessada através da rede para a obtenção dos *frames* a serem processados pelo algoritmo conforme segue:

- Analisar os *frames* de uma câmera;
- Detectar movimento;
- Obter placa de veículos, se houver.

Para realizar o objetivo principal, foi necessário atingir os seguintes objetivos específicos:

1. Desenvolver um módulo de análise de *frames* de vídeo para obter somente os *frames* com veículos a partir de uma câmera IP;

2. Criar uma base de imagens de placas e objetos que não sejam placas;

3. Analisar os principais métodos de localização automática de placas de veículo.

4. Definir uma solução baseada em abordagens utilizadas nos métodos estudados;

1.3 CONTRIBUIÇÕES

Este trabalho descreve a implementação de um sistema de localização automática de placa de veículos (LAPV) utilizando a abordagem do descritor de característica baseado na matriz de covariância. Descreve também as principais abordagens encontradas na literatura, utilizadas para resolver o problema de LAPV. Posteriormente, será proposto a implantação de um sistema de RAPV, cujo produto desse trabalho fará parte como o módulo de localização de placa.

Desse modo, esse trabalho pode contribuir como parte de um sistema de monitoramento que poderá ser implantado em qualquer ambiente, onde seja possível a instalação de uma câmera acessível através da rede de computadores utilizando o protocolo TCP/IP.

1.4 ORGANIZAÇÃO DO TEXTO

O trabalho está organizado como segue:

No Capítulo 2 são explicados os principais métodos de localização de placas de veículos. Quando presentes nos trabalhos, foram mostrados também os seus resultados.

No Capítulo 3 é descrito o localizador automático de placa de veículo proposto conforme os módulos que o compõe, detalhando os fundamentos e justificando as técnicas utilizadas.

No Capítulo 4 são feitos os experimentos e a análise de desempenho do método proposto, utilizando as métricas de avaliação também descritas.

No Capítulo 5 são feitas as considerações finais e sugestões para trabalhos futuros.

Capítulo 2

REVISÃO DE LITERATURA

A localização de placas veicular em processamento de imagens digitais é um problema bastante discutido em trabalhos acadêmicos. Os desafios intrínsecos ao problema vêm motivando pesquisadores e promovido uma extensa construção de literatura na área de visão computacional e reconhecimento de padrões. Neste Capítulo foram apresentadas as principais abordagens utilizadas em trabalhos divulgados e de ampla aceitação.

Aqui descrevemos as abordagens utilizadas por Mohamad et al. [25], usando detecção de bordas; a utilização do descritor de bordas e canais do espaço de cor RGB, no trabalho realizado por Mahini et al. [21]; o algoritmo Adaboost [39] utilizado por Dlangnecov [12]; as janelas concêntricas deslizantes, abordadas por Anagnostopoulos et al. [4]; o cálculo de média dos valores de gradientes, de Oliveira e Gonzaga [27]; a análise de componentes principais, utilizado por Passos, Maciel e Matos [29]; o descritor de covariância, de Porikli e Kocak [30]; e a operação morfológica de fechamento, método utilizado por Conci et al. [8].

2.1 LOCALIZAÇÃO DA PLACA

2.1.1 DETECTOR DE BORDAS

Para localizar a placa, Mohamad et al.[25] utilizou detecções de bordas e analisou os picos das resultantes, repetindo a análise em subimagens, até a localização completa. Primeiro, é aplicado o filtro sobel para detecção verticais. Com isso, é obtida a região que contém maior quantidade de bordas nessa direção. Esta heurística visa obter a região que contém os faróis, a placa e as estruturas do carro cuja projeção vertical salienta maiores quantidades de bordas em comparação com as outras regiões da imagem. A Figura 2.1 mostra o resultado da região obtida após o filtro sobel na direção vertical.

Tendo obtido a região de maior quantidade de bordas verticais (Fig. 2.1(a)), foi ne-

Figura 2.1: Resultados da aplicação do filtro Sobel vertical. (a) Imagem original, (b) sobel vertical, (c) projeção vertical da imagem aplicando threshold, (d) região de maior quantidade de bordas. Imagem retirada de [25].

cessário remover partes encontradas que não fazem parte da placa. Para isso, foi aplicado novamente o filtro sobel na direção vertical com um limiar determinado experimentalmente (Fig. 2.1(c)), a fim de diminuir a quantidade de arestas falsas. Então, é aplicado o filtro sobel na direção horizontal, seguido do filtro de média para minimizar os valores de picos, salientando a região de maior pico, e localizando a placa. A Figura 2.2 mostra o resultado da região obtida após o filtro sobel na direção horizontal e do filtro de média.

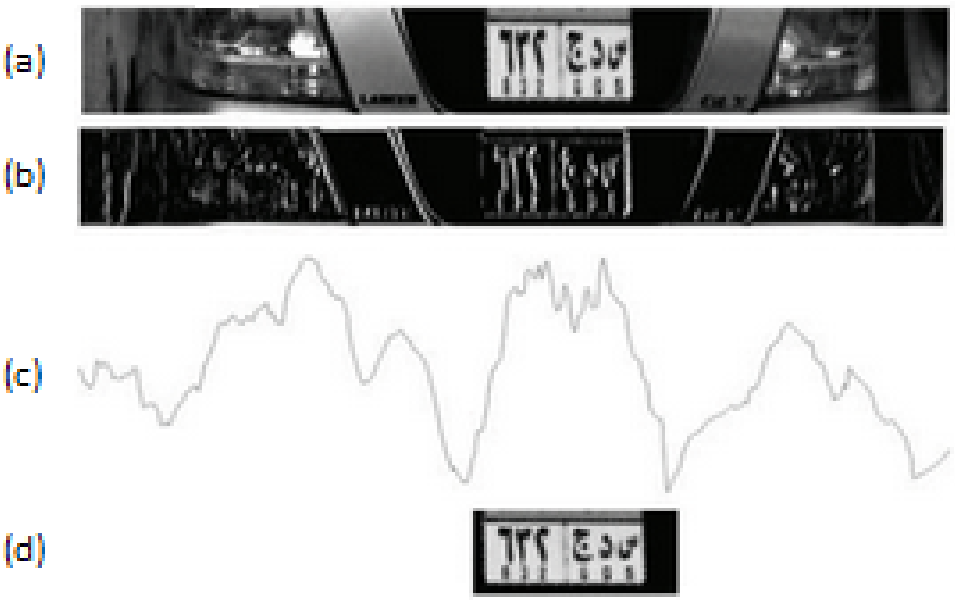


Figura 2.2: Resultados da aplicação do filtro sobel horizontal. (a) Região da imagem, (b) sobel vertical, (C) projeção horizontal de (b), (d) região correspondente ao pico das médias. Imgem retirada de [25]

A região contendo os faróis e a placa (Fig. 2.2(a)) possui maior quantidade de bordas verticais (Fig. 2.2(d)). Os picos do valor do filtro sobel na direção horizontal são suavizados com o filtro média, obtendo uma região com maior quantidade de picos

próximos (Fig. 2.2 (c)), uma vez que a suavização aproxima os vales de seus picos. Essa região, contendo maior quantidade de picos próximos, é a região onde a placa se encontra.

2.1.2 RGB e SOBEL

Tendo em vista a semelhança nos valores para cada componente do espaço de cor R, G e B das placas; a intensidade dos caracteres escuros sob o fundo claro; e a grande quantidade de bordas verticais, Mahini et al.[21] utilizaram análise de cor e bordas para localizar a placa.

A operação *bothat* é a subtração da operação de fechamento (dilatação seguida de erosão) pela imagem inicial [20]. Ela enfatiza as regiões escuras sob fundo claro. Partindo de tais características das placas, a operação foi utilizada para aumentar o poder da detecção de borda do filtro sobel. Feito isso, as bordas verticais foram detectadas e suavizadas. A Figura 2.3 (b) mostra o resultado do procedimento aplicado na Fig. 2.3 (a).

Figura 2.3: Aplicação do filtro bothat e sobel vertical. (a) imagem original, (b) resultado do filtro *bothat* seguido de sobel em (a). Imagem retirada de [21]

O próximo passo foi utilizar uma convolução para obter as primeiras regiões candidatas à placa. A Figura 2.4 mostra as imagens obtidas após cada etapa da busca pelas regiões candidatas (C) à placa. Em Fig. 2.4(a) está o resultado da convolução utilizando matriz identidade 3x30 na imagem contendo as bordas verticais. Segue a expressão que representa a obtenção de C1:

$$C1 = Edge * S_{3\times 30}\,, \tag{2.1}$$

onde C1 é uma imagem contendo candidatos à placa na primeira etapa; Edge é a imagem resultante do filtro sobel suavizada; e $S_{3\times30}$ é a matriz identidade usada na convolução.

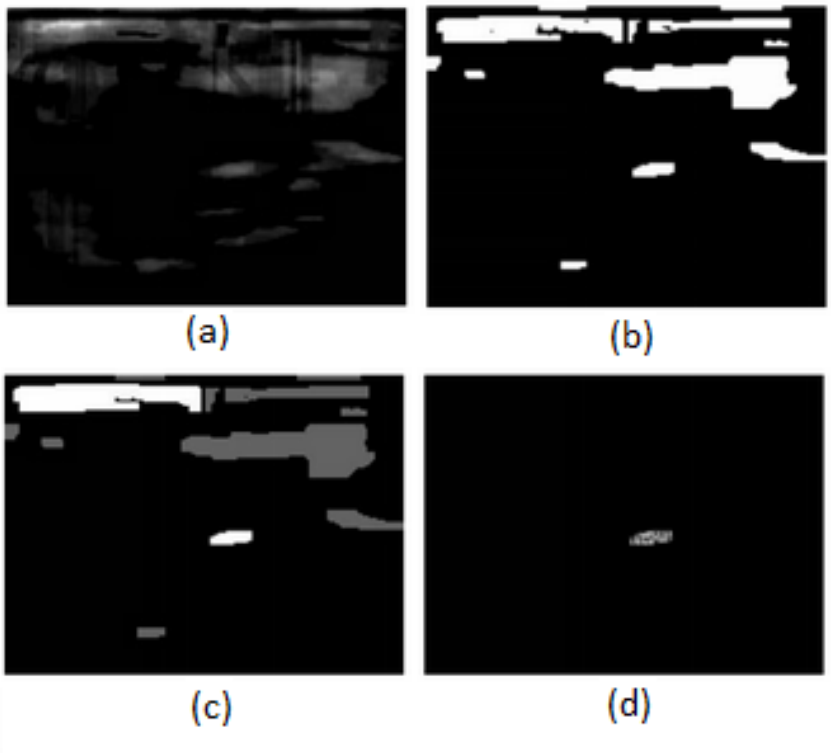


Figura 2.4: Buscas da placa por segmentação de regiões candidatas. (a) Convolução utilizando matriz identidade 3x30. (b) Binarização e operação morfológica de fechamento. (c) Resultado obtido apartir de heurísticas como dimensão, retangularidade e proporção. (d) Binarização Otsu em cada região candidata. Imagem retirada de [21]

O resultado visto em (b), é obtido conforme segue:

1. Binarização aplicando a regra: C2(i, j) é igual a 1 se a intensidade no pixel (i, j) tiver o mesmo valor nos componentes R, G e B, resultando em pixels de aparência cinza; caso contrário, C2(i, j) é igual a 0;

2. Aplicação da operação morfológica de fechamento utilizando matriz 3×3.

$$C3 = I.S_{3\times3}, \tag{2.2}$$

   onde I é a imagem em escala de cinza.

3. Fusão de C1, C2, C3:

$$\begin{gathered} Candidata(i,j) = C1(i,j) * C2(i,j) \\ *C3(i,j) * (C3(i,j) > 50) \\ Candidata2 = Candidata1 \Delta S_{3\times8}, \end{gathered} \tag{2.3}$$

   onde Δ é uma operação *tophat* utilizando uma matriz de convolução 3x8.

4. Por fim, a imagem Candidato2 é binarizada aplicando um limite de corte.

Tendo obtido o resultado como na Fig. 2.4 (b), as regiões encontradas são avaliadas conforme as seguintes heurísticas:

- Dimensão;
- Retangularidade e proporção;
- Intensidade;
- Localização referente às margens da imagem;
- Autovetores da PCA[1] deve ser quase horizontal, no máximo 35º.

Na Figura 2.4 (C) está o resultado de uma análise na Fig. 2.4 (b) conforme os preceitos listados acima.

A região final contendo a placa (Fig. 2.4 (d)) é obtida com a aplicação do método de binarização Otsu em cada região candidata (Fig. 2.4 (c)) e o histograma da soma dos valores de uma linha perpendicular ao primeiro autovetor da PCA, e que contenha entre 4 e 10 picos.

2.1.3 MATRIZ DE COVARIÂNCIA

Porikli e Kocak [30] descreveram um método de detecção de placas baseado na covariância entre características. A abordagem utiliza as covariâncias entre os valores dos gradientes e as coordenadas (x, y) dos pixels das imagens que compõem o *dataset*. Deslizando uma janela de tamanho fixo sob imagens de placas, amostras positivas; e de não placas, amostras negativas, são extraídas as matrizes de covariância dos valores de características em cada pixel da janela. As características foram a posição atual x e y; o nível de cinza I; o primeiro gradiente na direção x e na direção y (Ix e Iy, respectivamente); e o segundo gradiente na direção x e y (Ixx, Iyy, respectivamente). Desses valores são obtidos uma matriz de covariância para cada janela deslizante em uma amostra, positiva ou negativa. A Figura 2.5 ilustra o vetor das características extraído de uma placa e a matriz de covariância resultante.

[1]Do inglês, *Principal Component Analysis*.

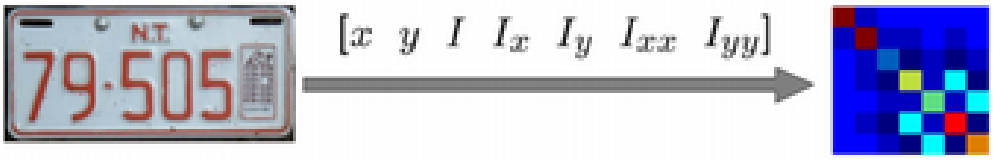


Figura 2.5: Vetor de características da placa e sua matriz de covariância. Imagem retirada de [30]

O *dataset* de imagens foi composto por 300 imagens de placas e 3000 imagens de não placas, de dimensões 105×32, obtidas através de uma câmera de alta resolução direcionada para uma faixa contendo sinalização de parada obrigatória. A Figura 2.6 mostra exemplos das imagens de placas usadas pelos autores.

Figura 2.6: Exemplos de imagens que compõem a base de dados utilizada por Porikli e Kocak. Imagem retirada de [30].

Obtidas as matrizes de covariâncias, uma rede neural é treinada tendo como entrada os valores na triangular superior e diagonal principal (valores únicos) de cada matriz, e um valor indicador do tipo da amostra: 1 para valores de matriz de janela em amostra positiva; e -1 para valores de matriz de janela em amostra negativa.

O teste da rede é feito de maneira similar, passando para a rede neural os valores únicos da matriz de covariância e obtendo como resposta da rede os valores 1, indicando que a janela está sob uma amostra positiva; ou -1, indicando que a janela está sob uma amostra negativa. Os resultados dos experimentos foram comparados com o resultado obtido através da aplicação de um descritor de histograma das imagens filtradas com o gradiente vertical e horizontal, método semelhante ao utilizado por Mohamad et al.[25].

Tabela 2.1: Lista de características retirada de cada pixel para compor diferentes vetores. Tabela retirada de [30]

Features	**Número**
g, r, I, $\|Ix\|$, $\|Iy\|$, $\|Ixx\|$, $\|Iyy\|$	C1
x, y, I, $\|Ix\|$, $\|Iy\|$, $\|Ixx\|$, $\|Iyy\|$	C2
x, y, I, , $\|Ix\|$, $\|Iy\|$	$C_{5\times5}$
histograma	Hϕ

Utilizando o método de covariância em diversos grupos de features organizados conforme a Tabela 2.1, os testes indicam que o descritor de covariância produz resultados muito melhores em comparação com o método de orientação de histograma. O gráfico da curva ROC na Fig. 2.7 compara os resultados obtidos com a aplicação dos quatro descritores de características listados na Tabela 2.1.

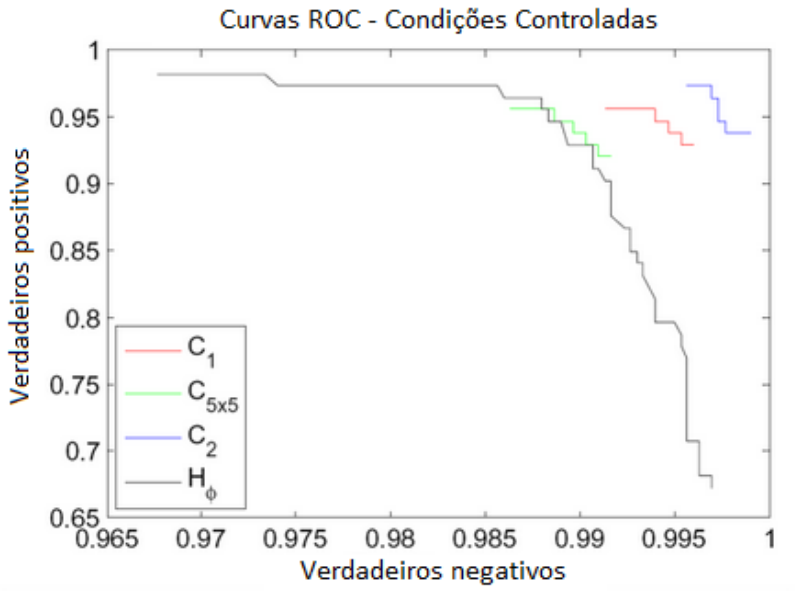


Figura 2.7: Gráfico de desempenho sob iluminação e ruído controlados para imagens posicionadas com precisão. Imagem retirada de [30].

Como regra geral, o desempenho degradou drasticamente à medida que a resolução das amostras foram diminuídas.

2.1.4 ADABOOST

Dlagnekov [12] utilizou o algoritmo Adaboost [39] para classificação e localização da placa. A abordagem envolve classificadores fortes em cascata criados a partir de classificadores fracos ponderados. Cada valor de característica é obtido da soma dos

valores de pixels das regiões claras subtraída pela soma dos valores de pixels obtidos das regiões escura. A Figura 2.8 ilustra as janelas de extração de característica do método Adaboost.

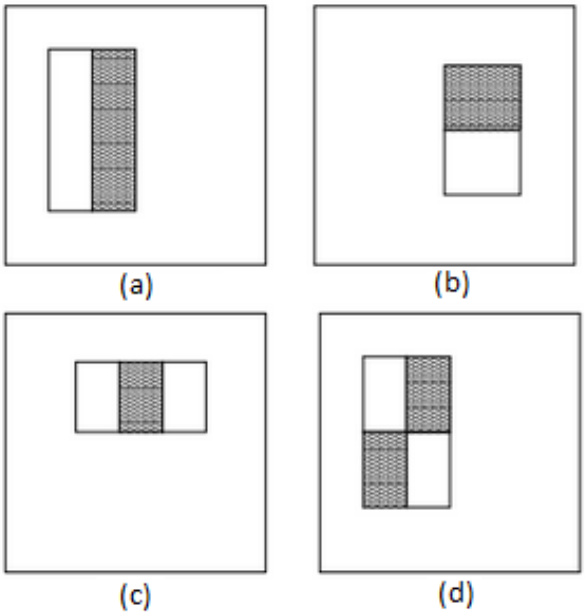


Figura 2.8: Templates Haar-like features. Imagem retirada de [39].

No trabalho de Dlagnekov [12], o algoritmo adaboost foi computado tendo como entrada os valores dos filtros de derivada horizontal e vertical (I_x e I_y, respectivamente), e variância, conforme a Fig. 2.9.

Figura 2.9: Tipos de características selecionadas pelo Adaboost. A soma computada nos regiões coloridas são subtraídos pelas regiões não coloridas. Imagem adaptada de [12].

Para o treinamento, Dlangnecov[12] usou 1500 imagens positivas, cortadas manualmente cortadas no tamanho de 45x15, e 10.000 imagens negativas. Todas as imagens foram obtidas através de uma câmera de vídeo de resolução 640x480. Foram selecionados 100 classificadores fracos para avaliar cada subregião durante o processo de aprendizado. Com essa abordagem, foi conseguido uma taxa de detecção de 95,6%; e uma taxa de falsos positivos de 5,7%.

2.1.5 FILTRO MÉDIA

Oliveira e Gonzaga [27] propuseram um método de localização baseado na busca pela região de maior média entre os valores de intensidade de pixel. Uma janela deslizante, de dimensão proporcional à placa, é passada sob imagens gradiente (filtros derivadas horizontais Gx, Gxx) calculando a média dos valores de intensidade dos pixels naquela região, e comparando o resultado com a maior média encontrada nas amostras de placas. A região encontrada contendo a maior média é considerada uma placa.

Para a realização do trabalho, os autores utilizaram cinco bancos de imagens. A Tabela 2.2 representa a qualidade das imagens em cada banco e os resultados do desempenho do método testado em cada um deles.

Tabela 2.2: Bancos de *datasets* e resultados.

Banco	**Quantidade**	**Qualidade**	**Acertos**	**Taxa de acerto**
BD1	75	Má	73	97,3%
BD2	77	Boa	77	97,5%
BD3	127	Boa	115	91,3%
BD4	17	Boa	14	82,3%
BD5	17	Boa	13	76,5%

2.1.6 OPERAÇÃO MORFOLÓGICA

Conci et al. [8] propuseram um método de localização da placa invariante a transformações geométricas, como a posição do carro e da câmera, e a variações na iluminação e nas dimensões da placa. A abordagem utiliza operações morfológicas, considerando o contraste entre o fundo da placa e as letras; e a relação entre as dimensões dos caracteres e distância entre eles. Segue os passos do algoritmo de localização:

1. Conversão das imagens coloridas para níveis de cinza;
2. Operação morfológica de fechamento;
3. Binarização da imagem pelo Método de Otsu;
4. Extração da região encontrada com altura e largura máximas e mínimas predefinidas.

Para o experimento, foram utilizadas imagens com resolução 320x240, com características variáveis, e de diferentes bases de dados. A Tabela 2.3 descreve o *dataset* utilizado.

Tabela 2.3: Características das bases de dados usadas. Tabela retirada de [8].

Número de imagens	**Base I**	**Base II**	**Base III**	**Base IV**
Total	180	100	32	12
Frontal	58	100	18	12
Traseira	122	0	14	0
Ângulo > 30	10	0	0	4
Com iluminação Heterogênia	37	4	7	0
Com iluminação Homogênia	143	96	25	12
Imagens claras	72	93	22	0
Imagens escuras	71	3	3	0
Distância da Câmera	2,0 a 3,5m	1,0 a 2,0m	1,5 a 2,5m	2,0 a 3,5m

Os resultados obtidos variaram de acordo com as características das imagens pertencentes a cada base de dados, conforme está descrito na Tabela 2.4.

Tabela 2.4: Resultado da localização das placas. Tabela retirada de [8].

Base	Correta	Parcial	Incorreta
I	63,88%	15,0%	21,11%
II	78,0%	5,0%	17,0%
III	84,38%	6,25%	9,38%
IV	66,67%	25,0%	8,3%

2.1.7 MÉDIA E DESVIO PADRÃO

Anagnostopoulos et al.[4] propôs um método de segmentação de placa utilizando duas janelas concêntricas deslizando na imagem e calculando o desvio padrão e o valor média de cada região. A Figura 2.10 (a) ilustra as janelas concêntricas SCW(do inglês, sliding concentric windows) e a direção em que elas percorrem a imagem, Fig. 2.10 (b). Sendo a razão entre os cálculos de medidas estatísticas das janelas maior que um valor de corte predefinido, o pixel central é considerado parte de placa.

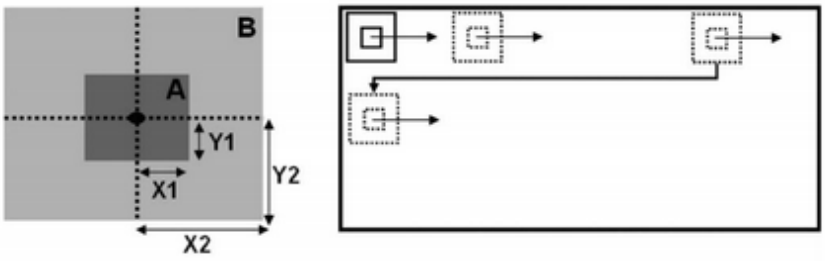


Figura 2.10: Ilustração da janela deslizante. (a) Janelas concêntricas e (b) percorrendo a imagem. Imagem retirada de [4].

Considerando o sistema de equação em 2.4, sendo I_1 um região da imagem representada em duas dimensões pelas coordenadas (x, y), e M a medida estatística (valor de média ou desvio padrão), um pixel central nas janelas concêntricas L1(x, y) é considerado pertencente à uma placa, assumindo o valor 1, quando a razão entre as medidas estatísticas das janelas for maior que o valor de corte T, caso contrário terá o valor 0, sendo considerado como pertencente a uma região fora da placa.

$$inI_1(x,y) \begin{cases} IAND(x,y) = 0, if \frac{M_B}{M_A} \leq T \\ IAND(x,y) = 1, if \frac{M_B}{M_A} > T. \end{cases} \tag{2.4}$$

O resultado da aplicação do SCW é uma imagem mapeada com valores 1 e 0. Então é utilizado um algoritmo de análise de vizinhança para determinar a real localização da placa. Com o método SCW aplicado em 1334 imagens contendo placas, foram detectadas 1287 placas verdadeiras, conferindo um resultado de 96,5% de acertos.

2.1.8 TRANSFORMADA DISCRETA DE FOURIER

Passos et al. [29] utilizaram uma abordagem de análise de textura, calculando a DFT[2], seguida do logarítmo sobre as magnitudes da DFT. O método é justificado pelas características estruturais dos elementos como asfalto, lataria, placa, borda, comumente presentes em imagens contendo veículos, serem bem descritas no domínio de frequência.

O logarítmo das magnitures de DFT é linearizado, resultando em um vetor de 16 posições, com muitos elementos redundantes e dependentes. Por esse motivo, utilizou-se os quatro primeiros autovetores e seus autovalores resultantes da PCA, após observado que eles correspondiam aos filtros gradiente horizontal, gradiente vertical, filtro passa baixas e filtro passa-altas, respectivamente. Nesses componentes foi aplicada uma função de transferência, com a finalidade de normalizar os valores no intervalo entre (0, 1). Segue a função sigmoide [19] utilizada na normalização:

[2]Do inglês, *Discrete Fourier Transform*

$$f(x) = \frac{1}{1+e^{-k(x+\theta)}}, \tag{2.5}$$

onde k é o ganho, e θ é o deslocamento.

A Figura 2.11 ilustra os vetores compostos pelos elementos dos quatro autovetores e seus autovalores da PCA. A Figura 2.11 (b) ilustra o resultado da normalização dos vetores mostrados na Fig. 2.11 (a).

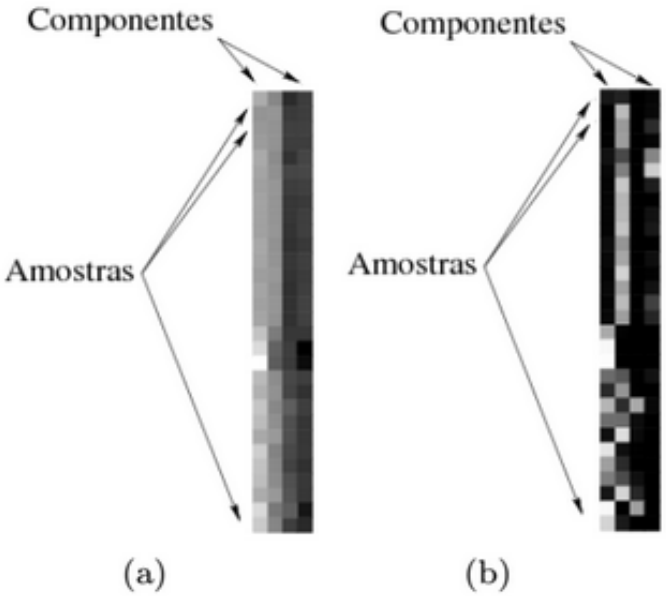
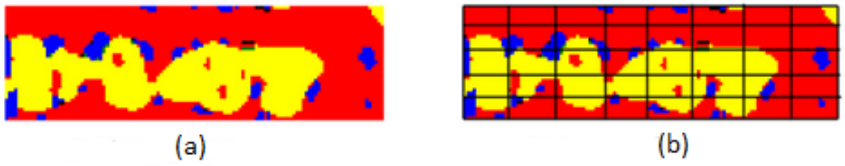


Figura 2.11: Exemplo de vetores de treinamento. (a) Subconjunto de vetores de treinamento. (b) Subconjunto após normalização. Imagem retirada de [29]

Os vetores normalizados foram submetidos a um rede neural auto-organizável (SOM)[3] cujos grafos resultantes do treinamento foram particionados pelo algoritmo Costa e Neto [11]. O resultado do mapeamento é ilustrado na Fig. 2.12. No retângulo à esquerda, uma subimagem qualquer, de dimensão 36 x 120, mapeada; à direita, a subimagem fatiada; e abaixo, uma matriz de contagem S que guarda a quantidade de pixels em cada quadro, e que corresponda ao rótulo que caracteriza a placa (rótulo amarelo). Como exemplo, Considerando o quadrado referente a posição 1x1 da matriz S, verifica-se 0 pixels com rótulos de placa; e na posição 2x1, foram encontrados 45 pixels com rótulos de placa.

[3]Do inglês, *Self Organization Maps*



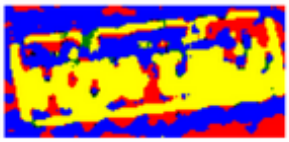
(a) (b)

$$S = \begin{vmatrix} 0 & 0 & 0 & 0 & 24 & 5 & 0 & 19 \\ 45 & 2 & 34 & 0 & 39 & 12 & 5 & 0 \\ 86 & 82 & 80 & 46 & 274 & 104 & 70 & 0 \\ 96 & 54 & 97 & 94 & 229 & 80 & 51 & 0 \\ 52 & 7 & 50 & 19 & 148 & 59 & 24 & 0 \end{vmatrix}$$

(c)

Figura 2.12: Exemplo do procedimento de rotulagem de subimagens.(a) Exemplo de uma subimagem de dimensão 36x120 pixels retirada de uma região qualquer da imagem, (b) fatiamento 5x8 dessa subimagem e (c) sua matriz de contagem S. Imagem retirada de [29]

Uma vez que qualquer região da imagem pode ser rotulada como uma região de placa, ainda que não pertença a uma, é necessário considerar o padrão da mancha em toda vizinhança para cada subimagem de dimensão 36 x 120. Para isso, a matriz de contagem de cada subimagem é comparada com a matriz de referência P, cujos valores correspondem a quantidade de pixels rotulados a partir de uma placa de referência (ver Fig. 2.13). Por fim, é atribuído notas no intervalo [0, 1] em cada partição da subimagem, de acordo com a similaridade entre sua matriz de contagem S e a matriz de referência P. Quanto mais S(i, j) for próximo de P(i, j), mais a nota de (i, j) se aproximará de 1. A nota global da subimagem é dada pela média aritmética de todos as notas das partições. A região que tiver a maior média é considerada placa.



$$P = \begin{vmatrix} 0 & 0 & 8 & 47 & 213 & 64 & 40 & 56 \\ 50 & 57 & 72 & 16 & 343 & 99 & 69 & 111 \\ 121 & 110 & 172 & 199 & 734 & 174 & 190 & 183 \\ 136 & 181 & 192 & 148 & 673 & 210 & 183 & 139 \\ 34 & 73 & 98 & 65 & 32 & 1 & 0 & 0 \end{vmatrix}$$

Figura 2.13: Exemplo de placa de referência e sua matriz de contagem P para um fatiamento 5 x 8. Imagem retirada de [29]

2.2 A RELAÇÃO ENTRE LITERATURA E MÉTODO PROPOSTO

Para a implementação do sistema LAPV proposto, decidimos aplicar o método do descritor de covariância similiar ao trabalho de Porikli e Kocak [30]. Visto que as imagens são obtidas em ambiente não controlado, e que os fatores externos como ruídos, iluminação e rotação influenciam na acurácia da localização da placa, a motivação para a adoção do método é o fato dele ser robusto a ruídos e variações de iluminação, o que é comprovado pela análise experimental feita no Capítulo 4.

Além dos gradientes utilizados no método de Porikli e Kocak[30], incluímos os valores de intensidade dos pixels para cada canal do espaço de cor RGB, já que as cores e suas variações nas placas obedecem um padrão predefinido de cores. Além dos diversos ruídos e variações causadas por mudanças climáticas, consideramos também que a posição das imagens obtidas estão sob rotação e inclinação em relação aos veículos, isso se deve a posição de instalação da câmera. Diante disso, corrigimos as angulações de forma fixa ao invés de utilizarmos as coordenadas polares, sugeridas por Porikli e Kocak para diminuir a sensibilidade a rotações, e que não foram citadas na Seção 2.1.3 por não terem feito parte do vetor de características (C_2, ver Tabela 2.1) que obteve o melhor resultado nos testes dos autores. O fato de optarmos por corrigir parte da rotação de maneira fixa, e não utilizando coordenadas polares, diminui a dimensão do vetor de característica, e melhora o desempenho do sistema durante a extração de características e durante a classificação das subimagens nas janelas deslizantes, já que, ao invés de um vetor de entrada na RNA contendo 28 elementos, adicionando mais dois elementos ao vetor de característas, teríamos um vetor de covariâncias resultantes de 45 elementos.

Capítulo 3

LOCALIZADOR AUTOMÁTICO DE PLACA DE VEÍCULO

Neste capítulo são abordados os detalhes dos procedimentos realizados para a implementação do sistema LAPV. Foi descrita desde a etapa de obteção das imagens à fase de teste da rede neural artificial. A descrição de cada procedimento e sua fundamentação teorica é feita em cada seção conforme a ordem de execução das fases nas quais se desenvolveu o projeto. A Figura 3.1 ilustra os módulos de desenvolvimento desse trabalho. Cada módulo se relaciona a uma fase do projeto e foram definidos com base nas atividades comumente encontradas em um sistema de visão computacional.

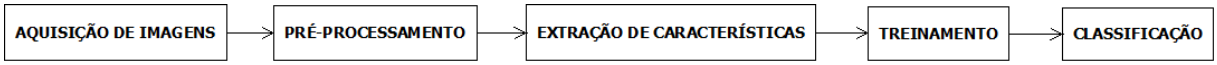


Figura 3.1: Módulos do sistema proposto.

Foi desenvolvido um sistema de localização automática de placa de veículos com a finalidade de localizar as placas que acessam a guarita principal da Escola Politécnica da UFBA. Através de uma câmera IP instalada na guarida de acesso, são obtidos os *frames* de vídeo para serem analisados pelo algoritmo de localização de placa, quando for detectado movimento de veículo. Visamos integrar a solução proposta neste trabalho em um projeto de reconhecimento de caracteres em placas veicular que será desenvolvido posteriormente.

O algoritmo aqui proposto desliza uma janela de dimensão fixa, obtendo os valores R, G e B de cada pixel, e calculando as derivadas parciais de primeira e segunda ordem, nas direções horizontais e verticais. De posse desses valores, é calculada a matriz de suas covariância, e os valores únicos da matriz, formados pela diagonal principal e uma das triangulares, são passados em forma de vetor para uma rede neural artificial. Por

fim, a RNA classifica a região sobreposta pela janela deslizante como positiva (placa) ou negativa (não placa).

3.1 FASES DO SISTEMA PROPOSTO

Um sistema de visão computacional costuma ser dividido pelas seguintes etapas: aquisição, pré-processamento, segmentação, extração de características, classificação e decisão [9].

Na etapa de aquisição, são adquiridas as imagens para compor o *dataset*. Nessa etapa foram obtidas imagens capturadas através de uma câmera IP com resolução de 640x480 pixels. Essas imagens foram recortadas em um processo manual de marcação de três pontos para extração das imagens de placas. As imagens de não placas foram obtidas utilizando um cortador automático que varre a imagem efetuando cortes com dimensões predefinidas e proporcionais às dimensões da placa. As atividades que envolvem esse etapa foram descritas na Seção 3.3.

As imagens de placas adquiridas são de posse da Universidade Federal da Bahia, laboratório IVision (Intelligent Vision Research Lab), e não serão divulgadas, obedecendo às leis do Código Penal Brasileiro.

A segmentação tem por objetivo separar nas imagens apenas as partes necessárias, e que contenham os objetos de interesse. O resultado dessa etapa é uma ou mais subimagens, ou regiões de interesse, consideradas candidatas à placa.

Na etapa pré-processamento, as imagens adquiridas sofrem alterações com a finalidade de melhorar aspectos, tais como intensidade de cor, angulação, bordas, e outras informações importantes para as etapas posteriores. Neste projeto as imagens tiveram uma correção na inclinação e rotação, visto que a localização e posição de captura da câmera obtém imagens com grandes alterações de angulação.

A extração de características é a etapa de obtenção dos dados a partir dos quais o computador possa distinguir os objetos a serem reconhecidos pelo sistema dos demais objetos que possam estar presentes nas imagens da aquisição. Nessa etapa, extraimos os canais RGB e os gradientes verticais e horizontais. Após isso, é calculada a matriz de covariância desses valores, e sua diagonal principal e triangular superior são transformadas em vetor de entrada para a rede neural.

Na etapa de classificação, as características de uma imagem, ou subimagem, são analisadas conforme uma técnica, modelo ou heurística a fim de reconhecer os objetos de interesse. Aqui, configuramos, treinamos e testamos a RNA com o descritor de covariâncias.

A última etapa, decisão, diz respeito às ações que devem ser tomadas pelo sistema

após o reconhecimento dos objetos. Os módulos do sistema proposto dão suporte à realização das etapas descritas acima, e que comumente constituem um sistema de visão computacional.

3.2 FERRAMENTAS UTILIZADAS

Para a realização desse trabalho, utilizamos a linguagem C/C++, o ambiente de desenvolvimento integrado (Integrated Development Environment - IDE) Eclipse, a biblioteca de visão computacional e aprendizado de máquina OpenCV (Open Source Computer Vision Library), versão 2.4.8, e a biblioteca FANN (Fast Artificial Neural Network Library), versão 2.2.0, que é uma biblioteca para redes neurais artificiais de múltiplas camadas.

3.3 AQUISIÇÃO DE IMAGENS

Para composição do *dataset*, foi criado um programa, escrito em linguagem C/C++, que acessa a câmera IP através do protocolo HTTP (Hypertext Transfer Protocol) e obtém os *frames* de vídeo para serem salvos em disco rígido. O programa foi instalado em um microcomputador desktop com processador de quatro núcleos AMD Phenon II 2.7 Ghz, e memória de 4GB, utilizando sistema operacional Debian 7. A Figura 3.2 contém exmplos de imagens obtidas acessando a câmera através da rede local.

Figura 3.2: Exemplos de imagens obtidas através da câmera.

Utilizando uma técnica de detecção de movimento por subtração de primeiro plano

(BS)[1], conforme descrita na Seção 3.3.1, apenas as imagens contendo movimento foram salvas para posteriormente serem tratadas e compor o banco de imagens. As placas são recortadas por um cortador de marcação de três pontos, e salvas para fazerem parte do conjunto de imagens positivas. O cortador, é um programa escrito em C/C++ que permite a marcação manual de três pontos, a partir dos quais é obtida a dimensão (largura e altura) e localização da região da placa a ser cortada. A Figura 3.3 ilustra os passos do corte de imagens positivas.

Figura 3.3: Exemplo de um procedimento de corte de imagem de placa utilizando o cortador de retângulos.

As regiões que não contém placas, foram cortadas automaticamente por uma janela deslizante de dimensão 40x13. O *dataset* foi composto 1043 imagens positivas, e 10.014 imagens de coisas diferentes de placas (ver a Tabela 3.1). A Figura 3.4 mostra exemplos de imagens obtidas.

Tabela 3.1: Descrição das imagens que compõem o dataset.

Tipo de Imagem	**Quantidade**	**Método de obtenção**	**Dimensão**
Placas	1043	corte manual	40x13
Não Placas	10.014	janela deslizante	40x13

[1]Do inglês, *background subtraction*.

Figura 3.4: Exemplos de imagens que fazem parte do *dataset*. (a) exemplo de imagens de placas recortadas, (b) exemplos de imagens de coisas que não são placas.

3.3.1 DETECÇÃO DE MOVIMENTO

Na etapa de composição do banco de imagens, consideramos que as imagens deviam fornecer subimagens contendo placas e objetos diferentes de placas, e que estes objetos deveriam fazer parte do contexto onde as placas seriam localizadas pelo sistema. Dessa forma, capturar *frames* apenas quando houver movimento de carro garante que todos esses objetos estejam presentes. Para isso, criamos um programa de captura de *frames* quando detectado algum movimento típico de veículo, utilizando a técnica BS.

3.3.2 SUBTRAÇÃO DE PRIMEIRO PLANO (BS)

Para detectar movimentos através da câmera, foi utilizado o método de subtração de primeiro plano, também chamada de detecção de primeiro plano [5]. Considerando um frame inicial sem movimento como segundo plano (*background*), os *frames* subsequentes são subtraídos dele até que o resultado da diferença seja maior que um valor de corte obtido empiricamente e conforme a aplicação. O resultado da subtração são as partes da segunda imagem que não estão presentes na primeira imagem.

Detalhando, de posse de um frame, extraímos uma região de interesse (ROI) [2], e a subtraímos pela mesma região de um frame obtido após dois segundos. Obtida a máscara do resultado da diferença, Figura 3.5(c), é realizada a contagem de pixels pertencentes à máscara, ou seja, de intensidade igual a 255. Caso o resultado da contagem seja superior a um valor de corte (obtido usando fuzzy, ver Seção 3.3.3), a imagem é considerada segundo plano, caso contrário, o processo é repetido até que o primeiro plano seja encontrado. Durante a execução do programa, o segundo plano é atualizado em intervalo de tempo predefinido. O valor de corte para definição do primeiro plano foi encontrado utilizando lógica fuzzy, conforme descrito na Seção 3.3.3. A Figura 3.5 ilustra o resultado da subtração do segundo plano 3.5(b) pelo primeiro plano 3.5(c). A mesma técnica é aplicada para encontrar, e atualizar, a imagem de segundo plano (*background*), e para detectar movimento (detecção de primeiro plano).

[2]Do inglês, *region of interest*.

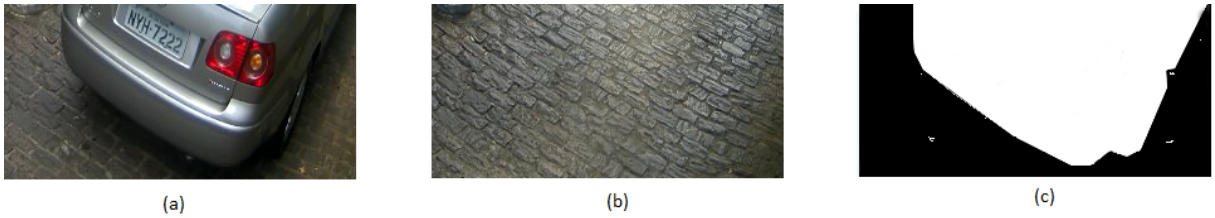


Figura 3.5: Ilustração do processo de subtração de imagem de segundo plano.(a) Imagem de primeiro plano, contendo veículo. (b) Imagem de segundo plano. (c) Imagem resultante da subtração de (b) em (a).

3.3.3 LÓGICA FUZZY

Zadeh [40], em 1965, propôs uma técnica capaz de obter informações não precisas, e convertê-las para um formato numérico que possa ser manipulado por computadores. Mais tarde, essa técnica passou a ser referenciada como lógica fuzzy, "lógica nebulosa" ou "lógica difusa". Neste trabalho foi utilizada a lógica fuzzy para determinar quando ocorre movimento de veículo, quando não ocorre movimento, e quando o movimento deve ser desconsiderado, como o movimento de pessoas, por exemplo.

Subtraindo duas imagens, obtém-se uma terceira imagem. A soma dos valores de pixel da terceira imagem resultante pode ser utilizada para calcular o quanto as duas imagens que participaram da subtração são diferentes. Utilizando lógica fuzzy, foi construída uma função de pertinência para identificar a ocorrência de movimento de veículo, e a ausência de movimento, a partir da soma dos valores de pixels de uma imagem subtraída de outra imagem. Jensen e Shen [18] definiram o conjunto fuzzy como um conjunto de pares ordenados A $= \{x, \mu A(x) | x \in U\}$, onde a função $\mu A(x)$ é chamada de função de pertinência de A, mapeando cada elemento do universo U a um grau de pertinência no intervalo [0, 1]. A especificação da função de pertinência é tipicamente subjetiva [18].

Uma forma de construir a função de pertinência é utilizando um modelo de votação [18]. Considerando o problema em questão, não existe uma fronteira bem definida que determine a medida exata da ausência de movimento, do movimento de veículos, e do movimento de outros objetos. Para essa situação, pode-se definir graus de pertinência utilizando conjuntos fuzzy. Seja $I_{m \times n}$ uma região de interesse na imagem I, U o conjunto universo pertencente ao conjunto dos números inteiros N, e *cont* uma função de contagem dos pixels de valor igual a 255 (branco), para $N \geq 0$ e $N \leq cont(I(m \times n))$:

$$\{U \in N | 0 \leq N \leq cont(I(m \times n) == 255)\}, \tag{3.1}$$

em uma região de interesse de dimensão 400x38, temos $0 \leq U \leq (400 * 38)$, pois supor a máxima diferença entre dois *frames* é considerar que cada um de seus pixels, na imagem

resultante, será igual a 255. No entanto, apenas nos interessa saber se a contagem dos pixels com valores 255 na imagem resultante da subtração de *frames* da região de interesse está nos intervalos $[0, 100], [100, 900], [900, 15200]$. Considerando os conjuntos B = {background}, MO = {movimento-objeto} e MV = {movimento-veículo}, temos a Tabela 3.2 para votação e definição da função de pertinência:

Tabela 3.2: Pertinência dos intervalos da contagem de pixels com intensidade igual 255 nas imagens resultantes da subtração de segundo plano.

$Cont(I(mxn) == 255)$	Pertinência B	Pertinência MO	Pertinência MV
$[0, 80]$	1	0	0
$]80, 90]$	0.8	0.2	0
$]90, 100]$	0.6	0.4	0
$]100, 700]$	0	1	0.2
$]700, 800]$	0	1	0.4
$]800, 900]$	0	0.8	0.6
$]900, 1000]$	0	0	0.8
$]1000, 15200]$	0	0	1

Com 15 imagens resultantes de subtrações de segundo plano, foi observado que a contagem dos pixels da região de interesse varia, aproximadamente, de 900 a 15200, durante a passagem de um veículo. Esses valores estão na Tabela 3.2 com pertinência maior em MV. O valor observado da quantidade de pixels na máscara dos *frames* considerados sem movimento foi a baixo de 100. Nesses casos, os movimentos observados foram de sombras das folhagens e, ou, de mudanças na iluminação. De 100 a 900, os movimentos comumente observados foram de pessoas transitanto.

3.4 PRÉ-PROCESSAMENTO

Para compor o *dataset*, as imagens foram submetidas a uma rotação e translação com o objetivo de diminuir a angulação das placas. Essas operações são transformações geométricas realizadas em cada pixel da imagem e em relação a uma outra, no caso, uma cópia dela mesma.

Considerando uma imagem como um conjunto de pontos P(x, y), sua translação P' é o resultado da soma de P por quantidades inteiras qx e qy, ou seja, uma função de translação T pode ser expressa por:

$$T(X_p, Y_p) = (X_p + qX, Y_p + qY). \tag{3.2}$$

Após aplicada a translação, a imagem sofre uma rotação através da aplicação da seguinte matriz:

$$\begin{bmatrix} \alpha & \beta & (1-\alpha).center.x - \beta.center.y \\ -\beta & \alpha & \beta.center.x + (1-\alpha).center, y \end{bmatrix}, \tag{3.3}$$

onde

$$\begin{aligned} \alpha &= scale.\cos angle, \\ \beta &= scale.\sin angle, \end{aligned} \tag{3.4}$$

e a escala e ângulo foram definidos empiricamente, com o valores 0.7, e 30º, respectivamente.

As Figuras 3.6 (c) e 3.6 (d) mostram o resultado das operações descritas acima.

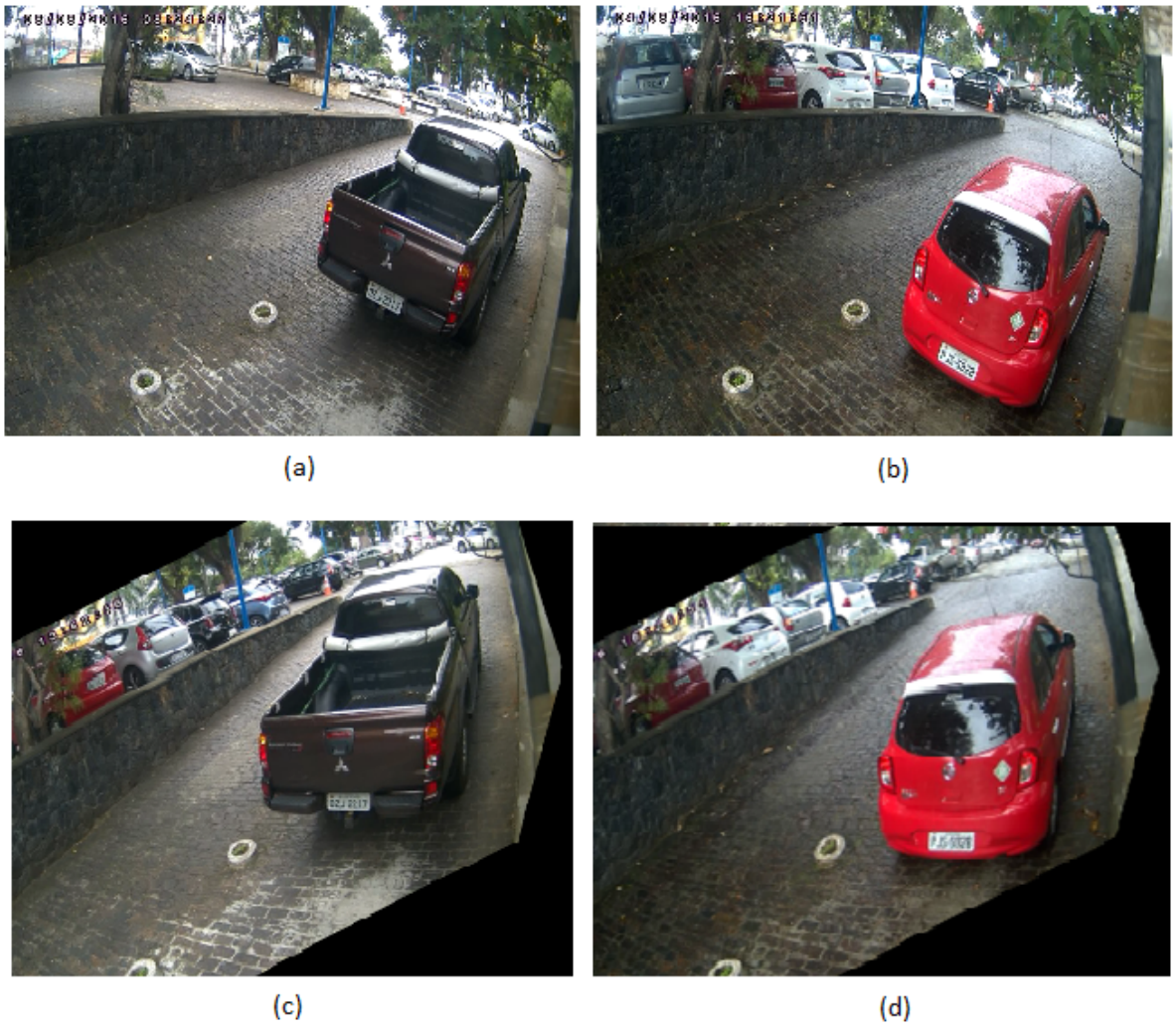


Figura 3.6: Exemplos de posicionamento da imagem. (a) e (b) Imagens originiais, (c) e (d) resultado de operação de translação e rotação de (a) e (b) , respetivamente.

3.5 EXTRAÇÃO DE CARACTERÍSTICAS

Em visão computacional e reconhecimento de padrões, para reconhecer objetos de interesse em imagens digitais, é necessário estabelecer parâmetros quantificáveis que possam ser interpretados pelo computador [8]. Segundo Gonzalez e Woods [13], representar uma região envolve duas opções: (1) podemos representá-las em termos de suas características externas como formato, dimensão, etc, ou (2) em termos de suas características internas (os pixels da imagem). Neste trabalho optamos por (2). O descritor de características baseado em matriz de covariância é uma forma de representação estatística sob os valores de pixels de uma região.

Após compor o *dataset*, as imagens foram submetidas a processos de extração dos valores dos canais de cor R, G e B, e das intensidades nas direções horizontais e verticais das bordas em cada pixel. A extração das intensidades das bordas na direção horizontal e vertical pode ser feita computando as derivadas de primeira e segunda ordem da imagem, ou aplicando as matrizes de convolução mostradas nas Figs. 3.7 e 3.8. Essa abordagem é justificada pelo realce das bordas dos elementos presentes na placa, já que as muitas bordas contidas no espaço de dimensão da placa costumam ser uma ocorrência singular no contexto das imagens analisadas.

Para Gonzales e Woods [13], a ferramenta ideal para encontrar a intensidade e a direção da borda na posição (x, y) de uma imagem, é o gradiente. Dentre os operadores utilizados para computar gradientes, Gonzales e Woods[13] afirmam que o sobel apresenta melhor supressão de ruído.

3.5.1 FILTRO SOBEL E LAPLACE

Os filtro Sobel e Laplace [13] são aplicações no domínio espacial, ou seja, realizada diretamente nos valores de pixels da imagem representada no plano, e tem por finalidade acentuar as regiões de maior taxa de variação nos valores de gradiente, e suavizar as regiões de menor variação. Isto produz um efeito de detecção de borda.

Supondo a representação de uma imagem digital como uma função $f(x, y)$, onde x e y correspondem à posição de um ponto em uma matriz bidimensional, e f a intensidade, ou nível de cinza, no ponto (x, y), uma borda é representada por uma mudança brusca e contínua em uma direção nos valores de intensidade dos pixels em relação aos pixels vizinhos.

Gonzalez e Woods [13] demonstraram a obtenção dos gradientes de uma imagem através das derivadas parciais em qualquer posição da imagem. As derivadas parciais de primeira e segunda ordem na direção (x, y), I_x, I_y, I_{xx} e I_{yy} podem ser implementadas utilizando as respectivas equações:

$$
\begin{aligned}
I_x(x,y) &= I(x+1,y) - I(x,y). \\
I_y(x,y) &= (i(x,y+1) - I(x,y+1). \\
I_{xx}(x,y) &= I(x+1,y) + I(x-1,y) - 2I(x,y). \\
I_{yy}(x,y) &= I(x,y+1) + I(x,y+1) - 2I(x,y).
\end{aligned} \tag{3.5}
$$

Resolvendo I_x e I_y (derivada de primeira ordem na direção horizontal e vertical, respectivamente), e I_{xx} e I_{yy} (derivada de segunda ordem na direção horizontal e vertical, respectivamente), obtém-se os valores de maior corte para cada orientação. Gonzalez e Woods [13] mostram que o filtro sobel pode ser obtido pelo cálculo das derivadas de primeira ordem, ou utilizando as matrizes de convolução ilustradas pela Fig. 3.7.

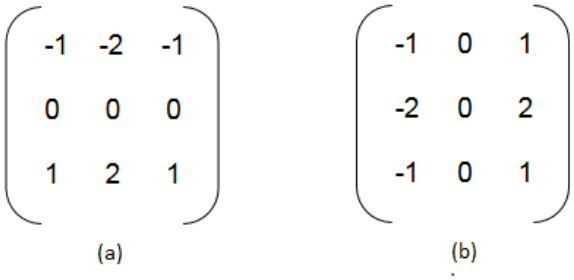


Figura 3.7: Máscaras de aproximação da primeira derivada na direção (a) horizontal e (b) vertical.

Sonka et al. [37], obtiveram o filtro Laplace através do cálculo das derivadas parciais de segunda ordem e através das matrizes de convolução ilustradas pela Fig. 3.8.

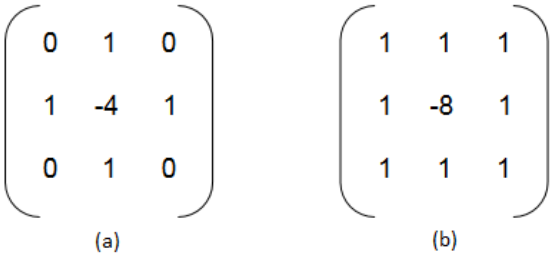


Figura 3.8: Máscaras de aproximação da segunda derivada na direção (a) horizontal e (b) vertical.

3.5.2 MATRIZ DE CONVOLUÇÃO

Convolução é uma operação linear frequentemente utilizada em processamento de imagem e expressa a relação entre duas imagens sobrepostas [37]. Sanchez e Canton [35] definiram convolução como um processo linear onde cada elemento do kernel é multiplicado por uma constante chamada de coeficiente de convoluçao (cc).

$$cc = \frac{f}{e}, \tag{3.6}$$

onde f é o multiplicador e e é o número de elementos no kernel. O fator f pode variar em cada elementos do kernel, enquanto o número de elementos e continua o mesmo. Uma representação de coeficientes em kernel de dimensão 3 x 3 pode ser:

$$\begin{matrix} a & b & c \\ d & e & f \\ g & h & i \end{matrix} \tag{3.7}$$

A matriz dos coeficientes de convolução é chamada de máscara ou kernel [35]. A máscara de convolução aplicada para obter o gradiente G_x (primeira derivada na direção horizontal), G_{xx} (segunda derivada na direção horizontal), G_y (primeira derivada na direção vertical), e G_{yy} (segunda derivada na direção vertical) são as já vistas na Seção 3.5.1. O cálculo consiste em multiplicar cada valor da máscara pelos valores de posição correspondente na imagem, posicionando o centro da máscara em um pixel da imagem filtrada, somar os resultados, e substituir o valor do pixel central pelo valor resultante. A Figura 3.9 ilustra o procedimento de convolução de um filtro em uma imagem I.

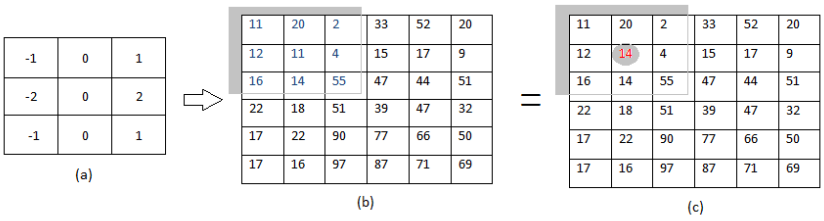


Figura 3.9: Convolução de matrizes. (a) Máscara de convolução aplicada sob a matriz (b) resultando na matriz (c).

Após multiplicar cada coeficiente da máscara pelo valor correspondente à cada posição na matriz imagem, da esquerda para a direita, e de cima para baixo, o pixel central na posição sobreposta da imagem é trocado pelo resultado das somas. A operação é feita conforme segue:

$$\begin{aligned} P(2,2) &= (11 \times (-1)) + (20 \times 0) + (2 \times 1) \\ &+(12 \times (-2)) + (11 \times 0) + (4 \times 2) \\ &+(16 \times (-1)) + (14 \times 0) + (55 \times 1) \\ &= 14 \end{aligned} \tag{3.8}$$

Utilizando a biblioteca de visão computacional e aprendizado de máquina OpenCV, tem-se a convolução implementada utilizando a equação matemática definida por Bradski e Kaehler [6]:

$$H(x,y) = \sum_{i=0}^{M_i-1} \sum_{j=0}^{M_j-1} I(x+i-a_i, u+j-a_j)K(i,j), \tag{3.9}$$

onde I(x, y) é uma imagem a ser filtrada, K(i, j) o kernel da convolução, com $0 < i < M_i - 1; 0 < j < M_j - 1$ e o pixel central como (ai, aj) [6].

3.5.3 MATRIZ COVARIÂNCIA

Milone [24] definiu covariância como a medida do grau de interdependência numérica entre duas variáveis aleatórias. Se duas variáveis aleatórias, x e y, são independentes, então cov(x, y) = 0 [31]. Para definir estatisticamente a covariância entre amostras, é computada a equação

$$\frac{1}{n-1} \sum_{i=1}^{n} (X_i^{(i)} - \overline{X}_1)(X_2^i - \overline{X}_2), \tag{3.10}$$

onde $(X_1^i, X_2^i), i = 1, ..., n$, são variáveis aleatórias e $\overline{X}_1, \overline{X}_2$ são suas médias aritméticas.

Neste trabalho foi utilizada a matriz de características conforme descrita por Porikli e Kocak [30]. Após a extração dos gradientes e a obtenção dos valores de R, G e B, é calculada a matriz de covariância entre as características fk:

$$fk = [r(x,y), g(x,y), b(x,y), I(x,y), Ix(x,y), Iy(x,y), Ixx(x,y), Iyy(x,y)], \tag{3.11}$$

onde fk representa o vetor de características referente a um pixel da janela W, r, g e b são os valores de cada canal do espaço de cor RGB, e I_x, I_{xx}, I_y e I_{yy} são os gradientes de primeira e segunda ordem na direção x, e de primeira e segunda ordem na direção y, respectivamente. A Figura 3.10 ilustra a matriz de covariância

$$\begin{aligned} \mathbf{C_W} &= \begin{bmatrix} c_W(1,1) & \cdots & c_W(1,d) \\ \vdots & \ddots & \\ c_W(d,1) & & c_W(d,d) \end{bmatrix} \\ &= \frac{1}{n-1} \sum_{k=1}^{n} (\mathbf{f_k} - \mu)(\mathbf{f_k} - \mu)^{\mathbf{T}} \end{aligned}$$

Figura 3.10: Matriz de covariância. Imagem retirada de [30]

Cw representa as covariâncias das janelas W(d, d), onde μ é o vetor média de todas as características.

Uma das propriedade da covariância é a simetria [16], ou seja Cov(X,Y) = Cov(Y,X). A simetria é também uma propriedade da matriz covariância [36] (Ct = C). Logo, Cw é

uma matriz simétrica em que seus elementos únicos encontram-se na diagonal principal com uma de suas triangulares.

A Figura 3.11. ilustra uma matriz de covariância 3x3. O valor de C(X, Y) terá o mesmo valor em C(Y, X), para qualquer X, Y. Portanto, sendo fk um vetor contendo as sete características, a saber, $r, g, b, I_x, I_{xx}, I_y, I_{yy}$, sua matriz de covariância gera vinte e oito elementos únicos. Isso não depende do tamanho da imagem (quantidade de pixels), mas sim da quantidade de características.

$$C_{mat} = \begin{pmatrix} cov(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & cov(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & cov(z,z) \end{pmatrix}$$

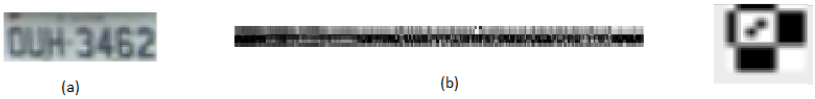
(a)

```
Matriz de covariancia:
[496.6636431006375, -90.71621461390247, 30.95114865866309,
  -90.71621461390247, 2281.308251815617, -36.19474581295372,
  30.95114865866309, -36.19474581295372, 3728.584719134427]
```

(b)

Figura 3.11: Ilustração de simetria em matrizes de covariância. Em (a) estão os relacionamentos das covariâncias em (x, y), e em (b) um exemplo de valores em matriz simétrica.

Na Figura 3.12, uma janela W_{dxd} extrai os gradientes e valores de R, G, e B em cada pixel da Fig. 3.12(a), obtendo um vetor de características (Fig. 3.12 (b)). O cálculo das covariâncias desse vetor dá origem a Fig. 3.12(c).



(a) (b) (c)

Figura 3.12: Extração das características (b) de uma placa (a) para obter a matriz de covariância (c) dos valores de (b).

Os valores únicos da matriz Cw foram transformados em um vetor de entrada e passado para o classificador. Neste trabalho, o classificador utilizado foi uma rede neural de múltiplas camadas com retropropagação de erro.

Sendo sete o número de características, cada pixel na imagem analisada gera uma matriz de covariância de dimensão 7x7. Com essa dimensão, foram utilizados 28 valores únicos (localizados em uma triangular com a diagonal) de covariância. Esses valores foram vetorizados e normalizar dentro do intervalo [-1, 1]. A normalização utilizada foi a norma L2, descrita em [17] conforme segue:

$$V(i) = \frac{Cov(x,y)}{\sqrt{\sum_{i=1}^{n} |v_i|^2}}, \tag{3.12}$$

onde $V(i)$ representa o elemento do vetor na posição i. Por fim, normalizados, os vetores são utilizados como entrada no nosso classificador.

3.6 REDES NEURAIS ARTIFICIAIS

Osório e Bittencourt [28] citaram vários tipos de tarefas que podem ser realizadas por redes neurais artificiais. Uma delas, é o reconhecimento de padrões em imagens digitais, ou classificação. Conci, Azevedo e Leta [8] definiram classificação como a tarefa de distinguir objetos na imagem agrupando seus descritores e objetos segmentados de acordo com suas semelhanças para cada região de pixels encontrada. Assim, neste trabalho, utilizamos uma rede neural para classificar as placas encontradas nas imagens a partir dos descritores de covariância conforme mostrado na Seção 3.5.3.

McCulloch e Pitts [22] propuseram um modelo matemático de funcionamento do neurônio biológico. A Figura 3.13 ilustra o modelo com i entradas, representando os dentritos em X1, X2, ... Xi, e uma saída, representando o axônio em Y. Cada entrada possui um peso W, representando a sinapse excitatória, valor positivo; ou inibitória, valor negativo. No corpo celular ocorre a soma $\sum$ dos valores do produto das entradas por seus pesos ($X_i.W_i$). Quando a soma for maior ou igual ao limiar, a saída é ativada com o valor 1 [7].

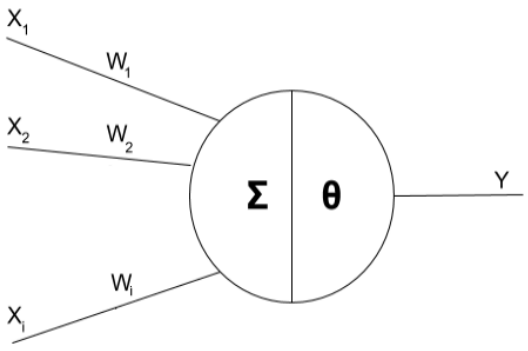


Figura 3.13: Modelo matemático de um neurônio biológico [22]. Imagem retirada de [7]

Após o modelo de McCulloch e Pitts [22], as principais alterações foram na função de ativação, ou função θ, adicionando um elemento chamado bias [7] que tem a finalidade de aumentar ou diminuir a entrada líquida da função de ativação, agindo como um peso extra nas conexões das unidades.

O modelo reproduz a estrutura e o funcionamento do cérebro humano tal como ele é descrito hoje pelas ciências naturais. A rede neural artificial recebe dados através de uma conexão que simula os dentritos, processa essas entradas ajustando a importância, ou peso, dos valores de cada dado em relação à característica a ser reconhecida como pertencente ou não a um padrão, e transmite dados para outra camada de neurônios que irão repetir o processo. O ajuste dos pesos deve aproximar a saída da rede à saída esperada, aprendendo a identificar a classe conforme os dados de entrada. A diferença entre o valor esperado e o valor atual é o valor do erro. Assim como o sistema cognitivo dos organismos inteligentes, a rede neural artificial aprende com os erros e acertos.

Gonzales e Woods [13] descrevem as redes neurais artificiais como o modelo de aprendizado de máquina de elementos básico de computação não linear, chamados de neurônios, organizados em uma estrutura semelhante ao que se acredita que seja a estrutura dos neurônios no cérebro humano. Os autores se referem ao modelo básico de uma rede neural artificial como perceptrons: tipos de redes neurais que aprendem uma função de decisão linear para conjuntos de treinamento linearmente separáveis.

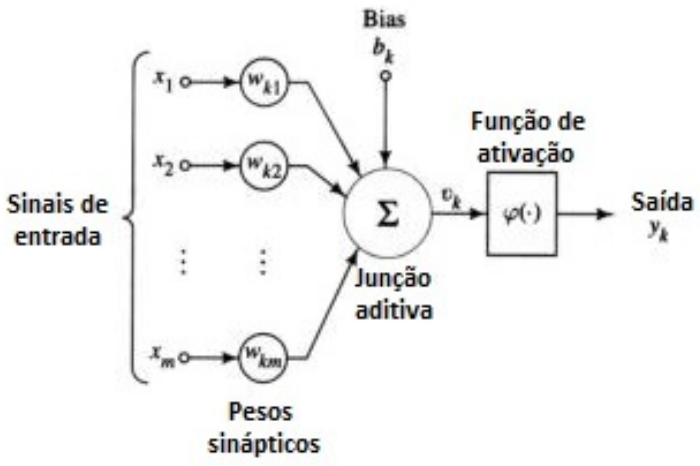


Figura 3.14: Modelo de neurônio base para projetos de RNA. Imagem retirada de [15]

No modelo representando um neurônio k, sinais xm partem através da sinapse m sendo multiplicado pelo peso sináptico W_{km}. Os produtos de cada sinal X_m por seus pesos W_{km} são somados, gerando o resultado U_k , que será passado para uma função de ativação.

A Figura 3.14 mostra os elementos básicos do modelo neural descrito por Haykin [15]. Um conjunto de sinapses, ou elos de conexão, são caracterizados por pesos W, presentes em intervalos positivos ou negativos, que são multiplicados pelas entradas x. Os produtos das entradas por seus respectivos pesos sinápticos participam de uma soma ponderada $\sum$ que pode ser expressa pelo seguinte combinador linear:

$$U_k = \sum_{j=1}^{m} W_{kj}.X_j. \tag{3.13}$$

A equação acima define o valor de U_k que será passado para a função de ativação junto com o bias, definindo o valor da saída y_k conforme segue:

$$Y_k = \varphi(U_k + b_k). \tag{3.14}$$

Haykin [15] chamou o resultado da soma ponderada U_k com a adição do bias b_k de potencial de ativação ou campo local induzido V_k:

$$V_k = U_k + b_k. \tag{3.15}$$

3.6.1 REDE NEURAL ARTIFICIAL DE ÚNICA CAMADA

A Figura 3.15 representa uma rede que aprende apenas funções para classes linearmente separáveis. Segundo Russell e Norvig [33], uma função pode ser representada por um perceptron se e somente se ela for uma função de separação linear, ou seja, caso ela sirva para classificar padrões linearmente separáveis.

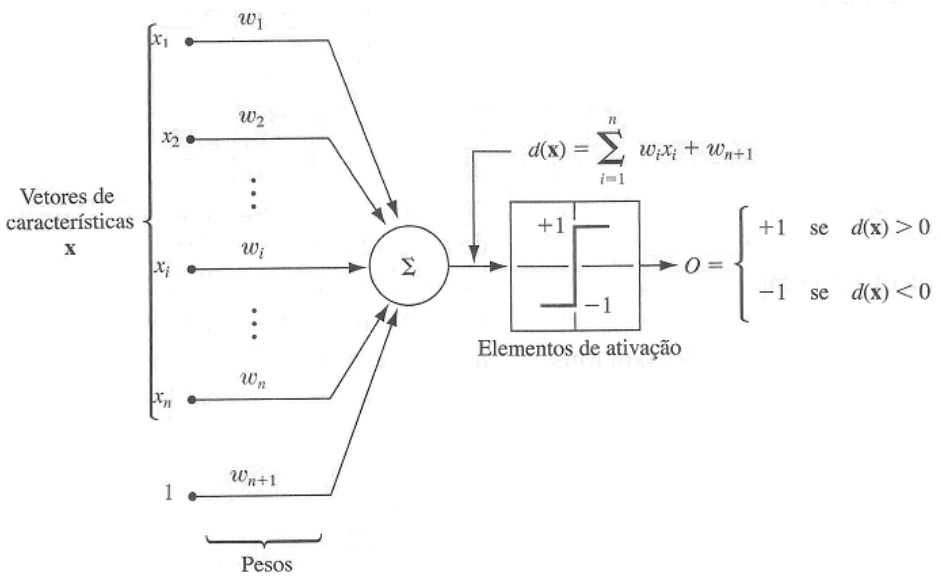


Figura 3.15: Modelo de perceptron para duas classes de padrões. Imagem retirada de [13]

As entradas na rede são vetores de características representados pelos valores de x. Os coeficientes w são os pesos de ajustes da soma ponderada das entradas. A soma ponderada pode ser expressa pela equação

$$d(x) = \sum w_i x_i + w_{n+1}. \tag{3.16}$$

O resultado da soma é passada para um função de ativação, que é a função que mapeia a saída da soma na saída final do dispositivo. Sendo d(x) > 0, o elemento de ativação faz com que a saída do perceptron seja +1, reconhecendo o padrão x como pertencente à classe; Quando d(x) > 0, o padrão é reconhecido como não pertencente à classe.

A equação d(x) do modelo descrito por Gonzalez e Woods [13] parece ter uma diferença do modelo utilizando bias. No entanto, Haykin [15] apresentou também um modelo similar em que um entrada X_0 sempre de valor +1 é multiplicada por um peso sináptico igual ao bias b_k [15]. Isso equivale a anotação W_{n+1} utilizada por Gonzalez e Woods [13].

3.6.2 REDE NEURAL ARTIFICIAL DE MÚLTIPLAS CAMADAS

Russell e Norvig [33] descreveram uma rede neural como uma estrutura composta de unidades conectadas contendo pesos numéricos associados a cada conexão. Cada unidade, ou neurônio, possui conexões com outros neurônios de entrada; conexões de saída; uma função de entrada; e uma função de ativação que fornece um valor na conexão de saída. Em redes neurais multicamadas, a saída de um neurônio pode ser fornecida como entrada para outro neurônio.

A estrutura de uma rede neural de múltiplas camadas foi descrita por Zhang e Gupta [41] como uma estrutura de neurônios agrupados em camadas, sendo a primeira e última camada, as camada de entrada e camada de saída, respectivamente, e as camadas intermediárias sendo chamadas de camadas escondidas. A Figura 3.16 ilustra a estrutura das redes multicamadas descritas por Zhang e Gupta [41].

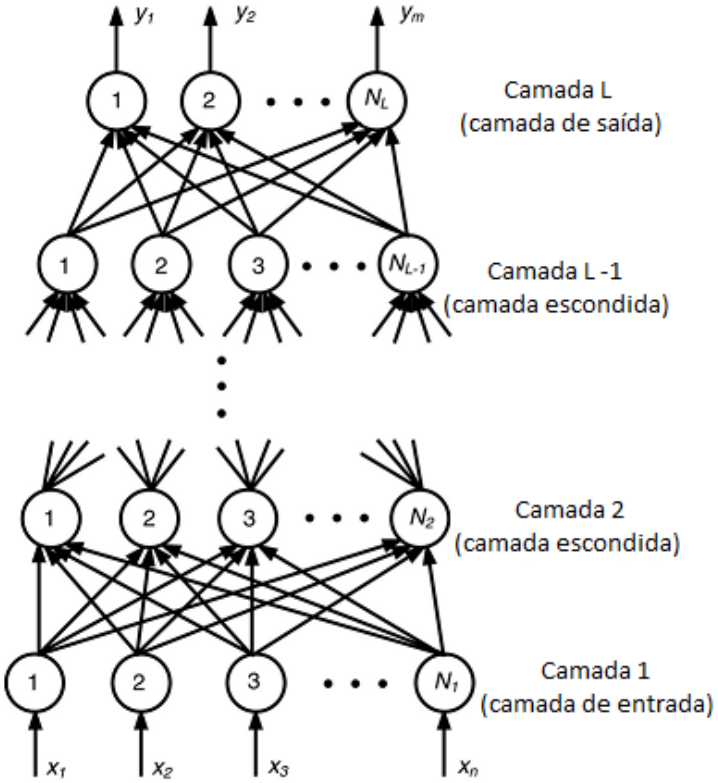


Figura 3.16: Estrutura de um perceptron multicamada. Imagem retirada de [41].

Na Figura 3.16, uma rede neural contendo n neurônios de entrada, é estruturada com L camadas de neurônios N, onde a primeira camada é a camada de entrada; a camada L2 à L - 1 são as camadas escondidas; e a camada L é a camada de saída. As xn entradas são computadas em cada neurônio da primeira camada e os valores resultantes são passados para a próxima camada, que repete o procedimento.

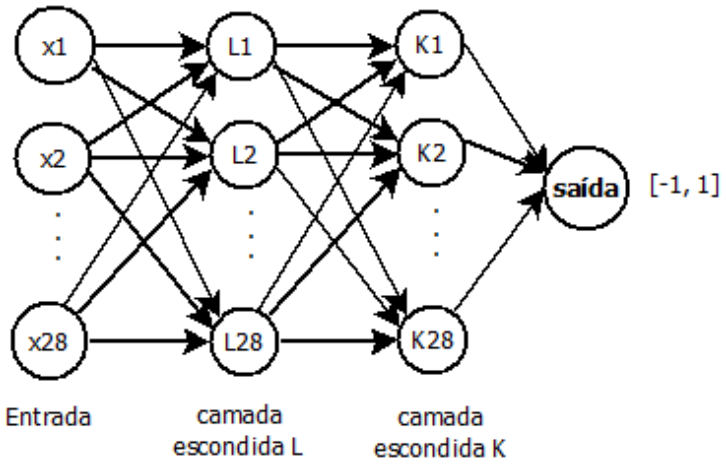


Figura 3.17: Estrutura da RNA utilizada no trabalho.

A RNA utilizada neste trabalho (ver Figura 3.17 possui 28 entradas, cujos valores são as covariâncias únicas e normalizadas em L2 do descritor de características (ver Seção

3.5.3); duas camadas escondidas, cada uma contendo 28 neurônios; e um elemento de saída com valores possíveis pertencentes ao intervalo [-1,1].

3.6.3 FUNÇÕES DE ATIVAÇÃO

A função de ativação tem a finalidade de limitar os valores de saída de um neurônio em um intervalo fechado [0,1] ou [-1, 1][13]. Haykin [15] definiu função de ativação como sendo uma aplicação de restrição da amplitude do sinal de saída de um neurônio em um intervalo permissível de valores finitos. Para a saída de um neurônio em termos do campo local induzido, ou força de ativação, o autor identificou três tipos básicos de função de ativação: a função de limiar; a função de limiar por partes; e a função sigmóide. A Figura 3.18 ilustra o gráfico da função de limiar.

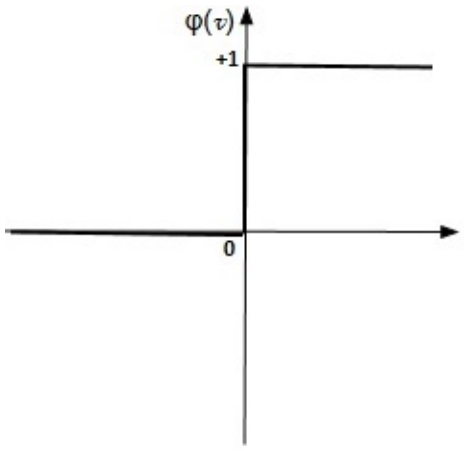


Figura 3.18: Função linear.

A função de limiar é aplicada em padrões linearmente separáveis. Neste modelo, a saída do neurônio assume o valor 1, para campos locais não-negativos, e valor 0, caso contrário:

$$\varphi(v) = \begin{cases} 1, & se \quad V \geq 0 \\ 0, & se \quad V < 0 \end{cases} \tag{3.17}$$

A função limiar por partes [15], ilustrada na Fig. 3.19, se comporta como a função limiar quando a operação no campo local induzido gerar uma amplitude infinitamente grande, caso contrário comporta-se como um combinador linear.

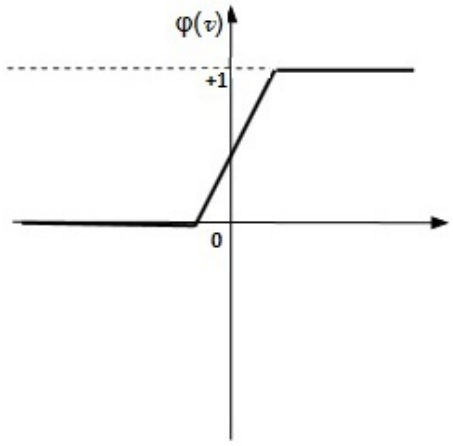


Figura 3.19: Função linear por partes.

Seque o sistema que define a função limiar por partes:

$$\varphi(v) = \begin{cases} 1, & V \geq \frac{+1}{2} \\ V, & \frac{+1}{2} > V > \frac{-1}{2} \\ 0, & V \leq \frac{-1}{2} \end{cases} \tag{3.18}$$

Segundo Heikins [15], a função sigmóide é a função mais utilizada em redes neurais artificiais. Para o autor, ela é uma função que exibe um balanceamento adequado entre comportamento linear e não-linear. A função sigmoidal tangente hiperbólica, ilustrada na Fig. 3.20 permite a normalização dos valores de saída no intervalo [-1, 1].

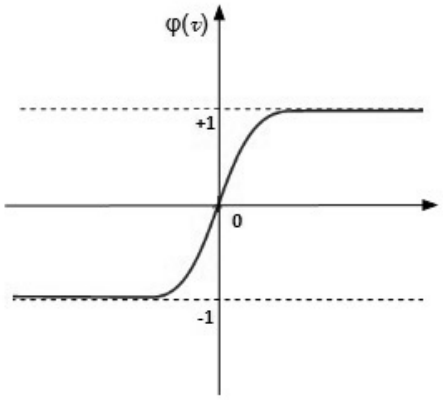


Figura 3.20: Função sigmóide simétrica.

Neste trabalho utilizamos a função sigmóide simétrica (Fig. 3.20), ou tangente hiperbólica por ela ser robusta ao comportamento não linear dos campos locais gerados do descritor de covariância. A biblioteca FANN [26] implementa a função sigmóide através da função fann_sigmoid_symmetric_real(sum), onde o argumento sum é a soma ponderada. Segue a equação que define a função utilizada:

$$\varphi(V) = \tanh(V) = \frac{1-e^{-v}}{1+e^{-v}}. \tag{3.19}$$

3.6.4 TREINAMENTO

Assim como o cérebro nos organismos biológicos inteligentes, as redes neurais aprendem com exemplos, acertos e erros. O aprendizado de uma RNA ocorre na fase de treinamento, primeira fase de uma solução utilizando RNA. De acordo com Mendel e Mclaren[23], nessa fase, a rede extrai informações relevantes de padrões a partir de dados apresentados a ela. Cada grupo de características é passado para a rede junto com suas saídas desejadas.

Supondo a utilização de n características, cada iteração na rede é o processamento de um grupo de valores das n características e suas respectivas saídas desejadas (normalmente -1 e 1 ou 0 e 1; -1 ou 0, para características de classes negativas; 1 para características de classes positivas). Cada valor de característica é multiplicado por um peso e participa de uma soma ponderada com o produto dos n valores de característica e seus pesos, como visto na Equação 3.16, Seção 3.6.1. O resultado da soma ponderada costuma ser passado para a funções de ativação presente em cada neurônio, definindo os valores que serão passados para a próxima camada de neurônios. Na saída de um neurônio ou iteração, é calculado o erro a partir da subtração da saída desejada e a saída atual conforme segue:

$$e_k(n) = d_k(n) - y_k(n)[15], \tag{3.20}$$

onde $e_k(n)$ é o erro no neurônio k e na iteração n, d é saída desejada, e y a saída atual.

O erro é utilizado no ajuste dos peso para a próxima iteração na rede, com a finalidade de aproximar os resultados de saída com os valores esperados. Haykin chamou de função de custo ou índice de desempenho $\beta(n)$, a equação utilizada para ajustar os pesos e diminuir os sinais de erro. O índice de desempenho foi definido como

$$\beta(n) = \frac{1}{2}e_k^2(n). \tag{3.21}$$

A fase de treinamento é interrompida com sucesso quando é alcançada uma taxa de acerto satisfatória. Segundo Gonzales e Woods [13], o principal objetivo dessa abordagem é encontrar os pesos que separe e classifique as amostras alcançando uma taxa de acerto que satisfaça a classificação.

O modelo de treinamento com ajuste dos pesos sinápticos descrito acima, foi chamado por Haykin[15] de aprendizagem por correção de erro. Existem vários modelos de aprendizado. Um texto descrevendo alguns desses modelos certamente teria uma grande

extensão e levaria este trabalho para além do que é proposto. O método de ajuste de pesos foi a abordagem utilizada no projeto de software descrito neste trabalho.

Através da câmera instalada na guarita de acesso ao estacionamento principal da escola Politécnica da UFBA, obtivemos os *frames* contendo imagens de veículos de onde cortamos manualmente, conforme descrito na Seção 3.3 as placas que foram analisadas. Foram utilizadas 1.043 imagens de placas e 10.014 imagens que não continham placa, somando um total de 11.057 imagens processadas na fase de teste. As imagens de placas foram redimensionadas para 40 x 13, que é uma dimensão proporcional à dimensão definida pelo Conselho Nacional de Trânsito [10] para placas de veículos no Brasil (ver Fig. 3.21), através da resolução 231 de 15 de março e 2007.

Figura 3.21: Modelo especificação da Placa de veículo no Brasil. Imagem retirada de [10].

A rede foi neural artificial teve seus componentes configurados conforme a Tabela abaixo.

Tabela 3.3: Parâmetros de configuração da rede neural artificial.

Componentes	**Valor**
Conjunto de entrada e saída	28 e 1, respectivamente
Número de neurônios em cada camada	28
Função de ativação	Sigmoide simétrica, normalizando entre [-1, 1]
Erro desejado	0.001
Número máximo de épocas	10.000
Número de camadas escondidas	2
Tipo de treinamento	supervisionado
Taxa de aprendizado	0.7
Momentum	0.0

3.6.5 TIPOS DE APRENDIZAGEM

O conceito de treinamento em redes neurais pode ser dividido conforme dois tipos de aprendizagem: a aprendizagem supervisionada, e a aprendizagem não supervisionada. Na aprendizagem supervisionada, conhece-se os valores desejados de saída para cada grupo de características passados à rede na fase de treinamento. Ou seja, os valores de treinamento são rotulados com os valores que devem ser aprendidos para as determinadas classes de padrões a serem reconhecidos. A aprendizagem não supervisionada é utilizada quando não se conhece as informações de classificação [14]. Para esses casos, separa-se as características em grupos, tendo como referência para os agrupamentos a medida de parâmetros dos objetos [9]

Sonka et al. [37] utilizam o mapa Kohonem como exemplo de uma rede neural de treinamento não supervisionado. O mapa Kohonem é uma técnica de agrupamento auto organizável (Self-Organizing Maps - SOM) que dispensa informações prévias sobre as classes de padrão. A Figura 3.22 ilustra uma rede neural utilizando o mapa Kohonem.

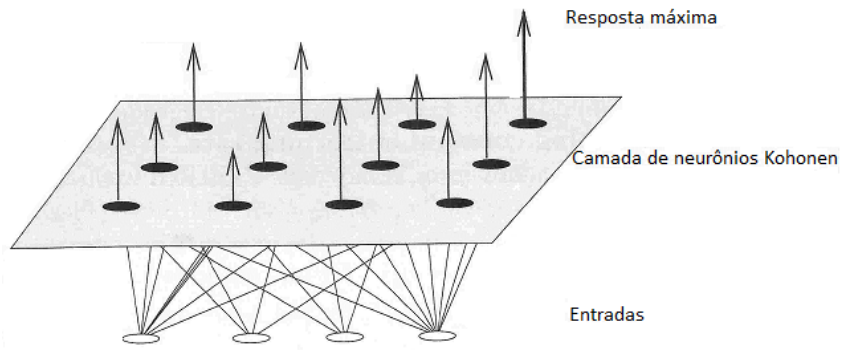


Figura 3.22: Mapa de agrupamento autoorganizável kohonen. Imagem retirada de [37].

Dada uma camada Kohonen em uma rede neural, serão admitidos como saída n-dimensional, os valores enquadrados nos grupos encontrados para os dados de entrada.

3.6.6 REDES NEURAIS DE RETROPROPAGAÇÃO

Devido a ampla utilização e casos de sucesso do algoritmo de retropropagação como método de treinamento da rede, resolvemos utilizá-lo em nossa solução.

No treinamento por retropropagação, o erro da camada de saída é utilizado para o ajuste dos pesos em cadeia, e para cada camada na direção inversa à entrada de dados. Assim, o erro é propagado da camada de saída à camada de entrada, ajustando os pesos em cada camada. Rojas [32] descreve o algoritmo retropropagação (do inglês, backpropagation) como a busca pelo erro mínimo no espaço de peso através do método de gradiente

descendente. O método requer o cálculo do gradiente da função de erro em cada iteração. Gonzales e Woods [13] descrevem o algoritmo como segue.

1. Definir um conjunto arbitrário de pesos de baixo valor;
2. Apresentar o vetor de treinamento à rede;
3. Calcular o erro, comparando o resultado da camada de saída com a saída desejada;
4. Passar o sinal de erro para cada nó das camadas anteriores;
5. Ajustar os pesos correspondentes a cada nó até a camada de entrada.

A biblioteca FANN, utilizada neste trabalho, implementa o algoritmo de retro-propagação na função fann_train(struct fann *ann, fann_type * input, fann_type * desired_output), onde

- ann - é uma estrutura de dados criada para representar uma rede neural e sua configuração;
- Input - é o conjunto de dados de treinamento;
- desired_ouput - é a saída desejada para o input conjunto de treinamento.

3.6.7 TESTE EM REDES NEURAIS ARTIFICIAIS

Na fase de teste, um conjunto de dados diferentes dos dados utilizados na fase de treinamento é utilizado para medir a performance da rede. Nessa fase é conhecida a qualidade da classificação da rede treinada. A capacidade de generalização da rede já pode ser vista aqui de modo muito próximo ao seu funcionamento quando já integrada ao sistema.

Neste trabalho, preparamos o ambiente de desenvolvimento para a realização dos testes conforme segue:

1. Obtenção de *frames* de vídeo a partir da câmera;
2. Correção de angulação das imagens;
3. Extração de características;
4. Classificação pela RNA;
5. Marcação da região reconhecida como placa.

O algoritmo de teste extrai as característas de cada frame percorrendo a janela deslizante de dimensão 40x13, gerando em cada pixel um vetor de 28 posições contendo as covariâncias dos gradientes e valores do RGB. O vetor é passado para a rede neural que classifica aquela região da imagem como positiva, retornando o valor 1; ou negativa , retornando o valor -1.

Neste capítulo foram explicados os módulos que compõem o sistema de LAPV proposto, sua composição e fundamentos teóricos. Também foram ditas as justificativas que motivaram a utilização de cada técnica explicada. No próximo Capítulo 4 serão mostrados os resultados do funcionamento do sistema na fase de análise.

Capítulo 4

ANÁLISE EXPERIMENTAL

Neste Capítulo serão descrito os procedimentos realizados na fase experimental do projeto. Fazem parte desta etapa a criação do *dataset* de imagens, para a realização dos testes; a definição das métricas de avalição do classificador com a finalidade de mensurar a qualidade da classificação aplicando o método proposto; e a análise dos resultados.

4.1 BASE DE IMAGENS DE TESTE

Para a fase de testes, foram utilizados 150 *frames* obtidos pelo mesmo método usado para compor as imagens de treinamento: capturando cada frame após detectado movimento de veículo (conforme descrito na Seção 3.3). Esses mesmos *frames* foram utilizados em dois tipos de testes definidos conforme a configuração do sistema localizador. Cada frame é percorrido pela janela deslizante gerando n quantidade de subimagens analisadas. Conforme as métricas de avaliação adotadas, essa quantidade, assim como a configuração de deslize (passos) da janela deslizante, influencia nos resultados.

4.2 MÉTRICAS DE AVALIAÇÃO

Para avaliar o sistema implementado, utilizamos algumas métricas comumente utilizadas em avaliação de classificadores. São elas: precisão, exatidão, taxa de acertos e taxa de alarme falso. Essas métricas são definidas em termo dos conceitos de verdadeiros positivos (VP), que são os objetos de interesse e corretamente classificados; falsos positivos (FP), objetos erroneamente classificados como elemento da classe de interesse; verdadeiro negativo (VN), objetos corretamente classificados como diferente da classe de interesse; e falso negativo (FN), que são objetos pertencentes a classe de interesse, mas erroneamente classificados como não pertencentes a ela. Seguem as definições das métricas.

- Precisão - Taxa de classificações de positivos considerando as subimagens que foram erroneamente classificadas como positivos. A precisão é dada por

$$P = \frac{VP}{VP+FP}. \tag{4.1}$$

- Exatidão - Também chamada de acurácia, expressa a relação entre as classificações corretas e o total de elementos classificados. A exatidão é dada por

$$E = \frac{VP+VN}{VP+VN+FP+FN}. \tag{4.2}$$

- Taxa de acerto - Taxa de exemplos positivos considerando as subimagens erroneamente classificadas como negativas. A taxa de acerto é dada por

$$TA = \frac{VP}{VP+FN}. \tag{4.3}$$

- Taxa de alarme falso - Taxa de errôneas classificações de positivos considerando as submagens que foram corretamente classificadas como negativas. A precisão é dada por

$$AF = \frac{FP}{FP+VN}. \tag{4.4}$$

4.3 AVALIAÇÃO

Para avaliar o desempenho da classificação da rede neural, foram realizados duas tomadas de testes com ajustes diferentes nos parâmetros de configuração do método. A Tabela 4.1 mostra os parâmetros que são configuráveis e seus respectivos valores utilizados em cada teste.

Tabela 4.1: Parâmetros de configuração do método proposto

Parâmetros	**Teste I**	**Teste II**
Largura da janela	40	40
Altura da janela	13	13
Deslize x	6	3
Deslize y	6	3
Classe *threshold*	0.95	0.95
Quantidade de *frames*	150	150
Tempo de uma análise em segundos	0.100	1.700

Onde, “Largura da janela” e “Altura da janela” compõem a dimensão da janela deslizante; o “Deslize x” é o passo da janela na direção horizontal; “Deslize y” é o passo da janela na direção vertical; e “Classe *threshold*”é o valor (de resposta da RNA) a partir do qual deve-se considerar a classe analisada como positiva. Os testes foram realizados utilizando 150 *frames*, cada um contendo uma placa.

As Tabelas que seguem (4.2 e 4.3) mostram os resultados da avaliação da classificação nos testes I e II. No teste I, analisando 810 subimagens em cada imagem, houve 89 reconhecimentos (verdadeiros positivos) de 150 placas; e 112 erros de classificação, sendo 51 o número de ocorrências de falsos positivos (objetos outros reconhecidos como placas), e 61 os falsos negativos (placas não reconhecidas).

No teste II, analisando 3240 subimagens em cada frame, houve 126 reconhecimentos (verdadeiros positivos) de 150 placas; e 73 erros de classificação, sendo 49 ocorrências de falsos positivos (objetos outros reconhecidos como placas), e 24 falsos negativos (placas não reconhecidas).

Tabela 4.2: Especificação dos testes I e II com relação a quantidade de *frames* contendo placa, quantidade de janelas (objetos) e tipos e quantidades de classificações (VP, FP, VN e FN).

Teste	**Quantidade de *frames***	**Quantidade de janelas**	**VP**	**FP**	**VN**	**FN**
I	150	810	89	51	121449	61
II	150	3240	126	49	485951	24

Tabela 4.3: Resultados da avaliação do classificador conforme as métricas já dadas.

Teste	**Precisão**	**Exatidão**	**Taxa de acertos**	**Taxa de alarme falso**
I	0.6357	0.9991	0.5933	0,0004197
II	0.72	0.9998	0.84	0.0001008

A Tabela acima mostra os resultados das métricas utilizadas para avaliar o classificador. Nota-se que o localizador é sensível a mudanças de configuração na janela deslizante. Analisando o mesmo banco de imagens, foi possível obter uma diferença entre os testes I e II de aproximadamente 24% , alterando apenas o deslize horizontal e vertical da janela deslizante. Porém, o tempo de análise de cada frame cresceu drásticamente. Utilizamos os deslizes de 1 pixel nas duas direções (x, y) para compor mais um teste e verificar o quanto esse tempo de análise cresceria. Em cada frame o tempo de análise subiu para 10 segundos. Esse teste foi descontinuado por causa do alto tempo de resposta e o pouco ganho observado em relação ao teste II.

A Figura 4.1 mostra exemplos de acertos e erros de classificação na fase de teste.

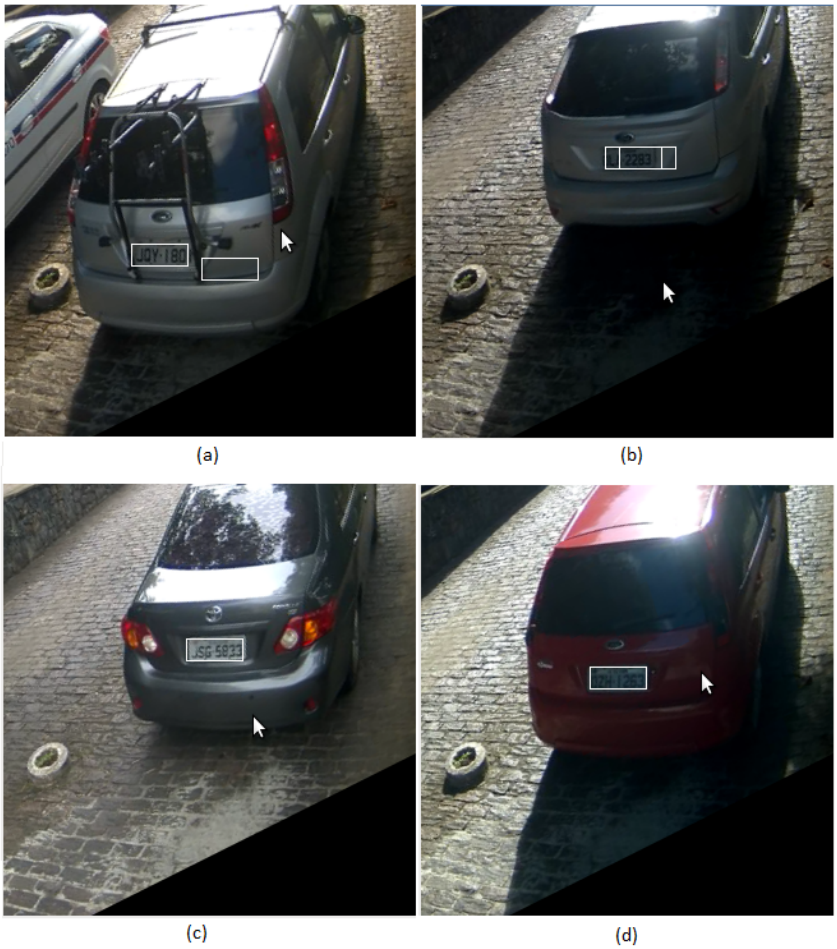


Figura 4.1: Exemplos de imagens da fase da classificação. Em (a) ocorre um reconhecimento e um erro, em (b) ocorre um reconhecimento total e um parcial, e em (c) e (d) ocorre reconhecimentos sem erro.

Para comparar o desempenho de nosso localizador com outros trabalhos, é necessário o acesso a uma base de imagens de veículos comum a ambos os trabalhos. Com o intuito de tornar possível as comparações entre pesquisas realizadas em sistemas de reconhecimento de placas veículares, Albuquerque [2] disponibilizou na internet uma base de imagens de veículos. No entando, hoje, essa base não está mais acessível. O último registro que encontramos de acesso a ela, ocorreu no ano 2011, e foi feita no trabalho de Souza e Passella [38]. Essa base não se encontra no endeço eletrônico divulgado pelo autor, e não foi encontrada em outro endereço. Diante disso, e da inviabilidade de implementação de um segundo método para comparação dos resultados, deixamos de fazê-lo.

Considerando os dados obtidos com os testes, o método proposto se mostra adequado para ser utilizado em um sistema completo de identificação de placa veicular por imagem digital. A variação da taxa de acerto em função do deslize da janela deslizante, que subiu de 59% para 84%, com aumento no deslize de 3 pixels nas direções x, y, e a

melhoria também no tempo de análise em relação ao deslize de 1 pixel em x e y, de 10 segundos para 2, mostram que o método é sensível a essas configurações e que pode atingir melhores desempenhos com ajustes na janela deslizante e, ou, em melhoria da qualidade e quantidade de amostras de imagens de treinamento. No trabalho de Porikli e Kocak [30] foram utilizadas 300 imagens de placas para treinar a rede. No entanto, considerando a Fig. 2.6 que representa a base de imagem utilizada por eles, as imagens de placas seria de melhor qualidade, e a base mais homogênia que a utilizada em nosso trabalho (ver Figura 3.4(a)). Diante do que foi dito, podemos considerar o método eficiente.

Capítulo 5

CONCLUSÃO

Neste trabalho descrevemos as principais técnicas de localização de placa veicular e apresentamos uma implementação do método que utiliza descritores baseado na covariância dos valores de gradientes e RGB. Nosso método é uma variação da abordagem utilizada por Porikli e Kocak [30]. Como classificador, utilizamos uma rede neural artificial de múltiplas camadas. O método escolhido é justificado por ser tolerante a ruídos e a variações de iluminação típicos de ambientes não controlados.

A base de imagens foi composta por *frames* de vídeo adquiridos através de uma câmera instalada na guarita de acesso ao estacionamento para funcionários da escola Politécnica da UFBA. A câmera é acessada pelo sistema para obter as imagens, assim como para localizar as placas na fase de testes.

Os testes indicaram uma boa taxa de acertos, considerando a complexidade das imagens analisadas e a pequena quantidade de amostras na fase de treinamento, em relação às variações de características entre elas. Outros métodos descritos neste trabalho alcançaram uma melhor taxa de acertos em imagens de melhor qualidade e, ou, em ambientes controlados. Porém, os testes indicaram que ocorre uma grande variação nessa taxa em função dos parâmetros configuráveis de nosso sistema. Esses parâmetros permitem uma grande quantidade de combinações para melhor ajustar o desempenho conforme os requisitos, como tempo de resposta, exigidos para a implantação do sistema.

Como sugestão para trabalhos futuros sugerimos o aumento do *dataset* de imagens, a repetição das fases de treinamento e testes, a construção de um segmentador de placa, e a utilização do método implementado neste trabalho para classificar os caracteres das placas localizadas. Sugerimos também a adição de placas de motocicletas no banco de imagens, assim como alterações nas dimensões da janela deslizante para corresponder à dimensão da placa que é de 187x136cm (LxA). Um classificador SVM[1] pode ser utilizado e ter seu desempenho comparado com o desempenho da RNA.

[1]Do inglês, *support vector machine*.

Por fim, consideramos que os objetivos foram alcançados, e que o trabalho realizado fornece outras possibilidades de utilização além do previsto. Não obstante, se faz necessário realizar mais ajustes e testes afim alcançarmos melhores resultados que já se mostram possíveis.

Referências Bibliográficas

[1] ABAR, CELINA. O Conceito Fuzzy. 2004. Disponível em: <http://www.pucsp.br/ logica/Fuzzy.htm.>.Acesso em: 10/09/2016.

[2] ALBUQUERQUE M. P. et. al. Projeto de Reconhecimento de Placas de Veículos Brasileiros. 2006. Disponível em: <http://www.cbpf.br/cat/pdsi/lpr/lpr.html/>. Acesso em: 22/05/2011.

[3] ALEKHIN, ALEXANDER. "Open Source Computer Vision Library". 2016. Disponível em: <http://http://docs.opencv.org/2.4.8/>. Acesso em: 18/9/2016.

[4] ANAGNOSTOPOULOS, C. N. E. et al. A License Plate-Recognition Algorithm for Intelligent Transportation System Applications. IEEE Transactions on Intelligent Transportation Systems, v. 7, n. 3, p. 377-392. 2006.

[5] BOUWMANS, THIERRY. et al. Background Modeling and Foreground Detection for Video Surveillance. Chapman and Hall/CRC. 2014.

[6] BRADSKI, GARY; KAEHLER, ADRIAN. Learning OpenCV. O'Reilly. United States of America. 2008.

[7] BRAGA, ANTÔNIO DE PÁDUA; LUDERMIR, T. B. Rede s Neurais Artificiais: teoria e aplicações. Editora LTC. Rio de Janeiro. 2000.

[8] CONCI, AURA; et al. A Complete System for Vehicle Plate Localization, Segmentation and Recognition in Real Life Scene. IEEE Latin America Transactions, v. 7, v. 5, p. 497-506. 2009.

[9] CONCI, AURA; et al. Computação Gráfica volume 2 (Processamento e Análise de Imagens Digitais. campus. 2009.

[10] CONTRAN, Conselho Nacional de Trânsito. Resolução 231. 11 p., Brasil, 2007.

[11] COSTA, JOSÉ ALFREDO F.; NETTO, MÁRCIO LUIZ A. Segmentação do SOM baseada em particionamento de grafos. Congresso Brasileiro de Redes Neurais, p.451-456. São Paulo. 2003.

[12] DLAGNEKOV, LOUKA. License Plate Detection Using AdaBoost. La Jolla. San Diego. 2005. Disponível em: <https://cseweb.ucsd.edu/classes/fa04/cse252c/projects/louka.pdf>. Acesso em: 01/09/2016.

[13] GONZALEZ, R. C.; WOODS, R. E. Processamento Digital De Imagens. Pearson Education - Br. 2011.

[14] HARRY, ANDREWS C. Introduction to Mathematical Techniques in Pattern Recognition. Wiley. 1972.

[15] HAYKIN, SIMON O. Neural networks and learning machines. Pearson Education, Inc. Hamilton, Ontario, Canada. 2009.

[16] HILDEBRAND, A.J. Introduction to Probability. Disponível em: <http://www.math.illinois.edu/ ajh/461/variance.pdf>. Acesso em: 01/10/2016.

[17] HORN, R. A; JONHSON, C. R. Norms for Vectors and Matrices. In:_____. Matrix Analysis. Cambridge, England: Cambridge University Press, Cap. 5. 1990.

[18] JENSEN, RICHARD; SHEN, QIANG. Computational Intelligence and Feature Selection: Rough and Fuzzy Approaches. Wiley-IEEE Press. 2008.

[19] JORDAN, M. I. Why the logistic function? a tutorial discussion on probabilities and neural networks. In: Technical report, Computational Cognitive Science 9503. Massachusetts Institute of Technology. 1995. Disponível em: <https://www.ics.uci.edu/ dramanan/teaching/ics273a_winter08/homework/jordan_logistic.pdf>. Acesso em: 02/10/2016.

[20] LOOMIS, JONH. Binary Image Morphology. 1997. Disponível em: <http://www.johnloomis.org/ece563/notes/BinaryImages/morph/morph.html>. Acesso em: 01/09/2016.

[21] MAHINI, HAMID. et al. An Efficient Features–Based License Plate Localization Method. ICPR, Hong Kong, China, 2006.

[22] MCCULLOCH, W. S.; PITTS, W. A Logical Calculus of the Ideas Immanent in Nervous Activity. Bulletin of Mathematical Biophysics, v. 5, p. 115-133. 1943.

[23] MENDEL, M. J.; MCLAREN, W. R. Reinforcement-Learning Control and Pattern Recognition Systems. Prentice Hall Press Upper Saddle River, NJ, USA, p. 287-318. 1970.

[24] MILONE, G. Estatística geral e aplicada. 1 ed. Centage Learning. São Paulo. 2003.

[25] MOHAMED, M. ABDELWAHAB. et al. Automatic Number Plate Recognition System. Annals of the University of Craiova, Mathematics and Computer Science Series. v. 38(1), p. 62–71. 2011.

[26] NISSEN, STEFFEN. Fast Artificial Neural Network Library. Disponível em: <http://leenissen.dk/fann/wp/>. 2016. Acesso em 18/9/2016.

[27] OLIVEIRA, LEONARDO AUGUSTO; GONZAGA, ADILSON. Localização, Segmentação e Reconhecimento de Caracteres em Placas de Automóveis. Neves et al. (Eds.), Avanços em Visão Computacional. p. 283. 2012.

[28] OSÓRIO, F. S.; BITTENCOURT, J. R. Sistemas Inteligentes baseados em Redes Neurais Artificiais aplicados ao Processamento de Imagens. In: Iº Workshop de inteligência artificial. Santa Cruz do Sul. p. 2-30. 2000.

[29] PASSOS, YURI TAVARES DOS; et al. Detecção de Placas de Licenca Veicular Utilizando Segmentação por Texturas. In: 4º Workshop de visão computacional, Bauru, São Paulo, p.47-52. 2008.

[30] PORIKLI, FATIH; KOCAK, TEKIN. Robust License Plate Detection Using Covariance Descriptor in a Neural Network Framework. In: Proceedings of the IEEE International Conference on Video and Signal Based Surveillance (AVSS'06), p. 107. 2006.

[31] PROKHOROV, A. Covariance. Encyclopedia of Mathematics. Disponível em: <http://www.encyclopediaofmath.org/index.php?title=Covariance&oldid=33783>. Acesso em: 20/09/2016.

[32] ROJAS, RAÚL. Neural Networks, A Systematic Introduction. Springer-Verlag. Berlin. 1996.

[33] RUSSEL, STUART J.; NORVIG, PETER. Artificial Intelligence A Modern Approach. Englewood Cliffs, New Jersey. 1995.

[34] SAHA, SATADAL; et al. Licence Plate Localization Using Vertical Edge Map and Hough Transform Based Technique. S.C. Satapathy et al. p. 649-656. India. 2012.

[35] SANCHEZ, JULIO; CANTON, MARIA P. Space Image Processing, Volume 1. CRC Press. London. 1999.

[36] SANIN, ANDRES. Improving Representation and Classification of Image and Video Data for Surveillance Applications. 2012. 148 f. PhD Tese. The University of Queensland, Austrália. 2012.

[37] SONKA, MILAN; et al. Image Processing, Analysis, and Machine Vision. 3th Edition. Thomson. 2008.

[38] SOUZA, GUILHERME S.; PASSELA, PAULO H. Reconhecimento Automático de Placas de Veículos Utilizando Processamento Digital de Imagens e Inteligência Artificial. In: Revista Científica Eletrônica UNISEB, Ribeirão Preto. v.1, n.1, p. 133.

[39] VIOLA, PAUL; JONES, MICHAEL. Rapid Object Detection using a Boosted Cascade of Simple Features. In: Computer vision and pattern recognition (CVPR). Cambridge, Inglaterra, p. 511-518, v. 1. 2001.

[40] ZADEH, L. A. Fuzzy Sets. Departamento de Engenharia e Laboratório de Pesquisa em Eletrônica, Universidade da California, Berkeley. Informação e Controle 8, p. 338-353. 1965.

[41] ZHANG, Q.J.; GUPTA, K. C. Neural Networks for RF and Microwave Design. In: IEEE Transactions on Microwave Theory and Techniques. v. 51, n. 4, p. 1339-1350. 2003.