http://journal.unoeste.br/index.php/ce/index
DOI: 10.5747/ce.2021.v13.n1.e353
ISSN on-line 2178-8332

Colloquium Exactarum

Submetido: 15/03/2021 Revisado: 20/05/2021 Aceito:15/07/2021

DETECÇÃO E RECONHECIMENTO DE PLACAS DE LICENCIAMENTO VEICULAR EM TEMPO REAL USANDO CNN

BRAZILIAN LICENSE PLATE RECOGNITION IN REAL TIME USING CNN

Marcelo Eidi Imamura[1], Francisco Assis da Silva[1], Leandro Luiz de Almeida[1], Danillo Roberto Pereira[1], Almir Olivette Artero[2], Marco Antonio Piteri[2]

[1]Faculdade de Informática de Presidente Prudente, Unoeste - Universidade do Oeste Paulista, Presidente Prudente
marcelo_eidi12@hotmail.com, chico@unoeste.br, llalmeida@unoeste.br, danilopereira@unoeste.br
[2]Faculdade de Ciências e Tecnologia, UNESP - Universidade Estadual Paulista
Departamento de Matemática de Computação, Presidente Prudente
almir.artero@unesp.br, marco.piteri@unesp.br

RESUMO – O Brasil possui uma grande frota de veículos trafegando diariamente pelas vias urbanas e estradas, o que se faz necessário o uso de alguma solução computacional para auxiliar no controle e gerenciamento. Neste trabalho foi desenvolvida uma aplicação para detectar e reconhecer placas de licenciamento veicular em tempo real com várias possibilidades de aplicações. A metodologia desenvolvida neste trabalho possui três etapas principais, sendo a detecção da placa, a segmentação dos caracteres e o reconhecimento. Para a etapa de detecção foi utilizada a biblioteca YOLO, que faz uso de técnicas de aprendizagem de máquina para detectar objetos em tempo real. A YOLO foi treinada utilizando um *dataset* com imagens de placas em diferentes ambientes. Na etapa de segmentação foi realizada a separação dos caracteres individualmente que estão contidos na placa, fazendo o uso de métodos de processamento de imagem. Na última etapa, foi realizado o reconhecimento dos caracteres utilizando duas redes neurais convolucionais, obtendo uma taxa de acerto de 83,33%.
Palavras-chave: reconhecimento de placas de licenciamento veicular, aprendizado de máquina, visão computacional.

ABSTRACT - Brazil has a large fleet of vehicles running daily along urban roads and highways, which requires the use of some computational solution to assist in control and management. In this work we developed an application to detect and recognize real-time license plates with various application possibilities. The methodology developed in this work has three main stages: plate detection, character segmentation and recognition. For the detection step we used the YOLO library, which makes use of machine learning techniques to detect objects in real time. YOLO was trained using a dataset with plate images in different environments. In the segmentation stage, the individual characters contained in the plate were separated, using image processing methods. In the last stage, character recognition was performed using two convolutional neural networks, obtaining a hit rate of 83.33%.
Keywords: license plates recognition, machine learning, computer vision.

1. INTRODUÇÃO

A frota de veículos licenciados no Brasil é muito ampla. Apenas no estado de São Paulo, no ano 2018, foram licenciados mais de 29 milhões de veículos (DETRAN SP, 2018). A identificação de veículos no Brasil é feita por meio dos caracteres contidos na placa de licenciamento veicular acoplada ao veículo em sua dianteira e outra, de conteúdo igual, na traseira. A placa é composta por sete caracteres alfanuméricos impressos ao centro, e o município e a sigla do estado ao qual o veículo pertence impressos na parte superior (Figura 1). As cores dos caracteres e das bordas são determinadas de acordo com a sua utilização, sendo a placa mais comumente encontrada a de fundo cinza e texto em preto, usada em veículos particulares. Ao total existem 13 tipos diferentes de placas (CONTRAN, 2018).

Figura 1. Placa de licenciamento veicular do Brasil.

Fonte: (CONTRAN, 2018).

Os veículos são utilizados pela população diariamente, para lazer e trabalho, o que gera um grande volume de veículos trafegando simultaneamente pelas vias urbanas e rodovias. Realizar uma fiscalização nessas vias com uma grande quantidade de veículos, sem o uso de algum recurso computacional, é algo dificultoso e ineficiente.

A detecção e o reconhecimento de placas de licenciamento veicular não é uma tarefa trivial de se realizar. Existe uma série de fatores que dificultam todo o processo, como, por exemplo, placa danificada e ou desgaste do material, variação de iluminação, distância em que a imagem é capturada, entre outros (SILVA et al., 2013) (WANG; ZHU; ZHANG, 2016) (CORNETO et al., 2017).

Algoritmos de reconhecimento de placas de licenciamento veicular (*License Plate Recoginition* – LPR) possuem, normalmente, três etapas, sendo a detecção, segmentação e reconhecimento (ISMAIL, 2017). A etapa de detecção é responsável pela localização da placa na imagem. A etapa de segmentação realiza um pré-processamento na imagem onde a placa está localizada. Em seguida é realizada a segmentação dos caracteres contidos na placa. A última etapa do processo é o reconhecimento, em que cada caractere segmentado na etapa anterior é classificado como alguma letra ou número (JAGTAP; HOLAMBE, 2018).

As aplicações LPR buscam ser as mais eficientes possíveis e ser executadas em tempo real. Mas, para alcançar essa eficiência não é muito comum utilizar métodos que fazem análise das imagens pixel a pixel. Um método muito utilizado quando se busca processamento em tempo real é o aprendizado de máquina, que é constituído de algoritmos que podem "aprender" a partir de um treinamento utilizando uma base dados. Em muitos dos trabalhos encontrados na literatura, algoritmos de LPR foram desenvolvidos para resolver problemas em ambientes controlados (LEE et al., 2018), podendo não funcionar adequadamente para veículos em movimento nas vias e rodovias. Nesses casos, além da questão de poderem ser capturados muitos quadros (*frames*) de vídeo, ocorre variação de iluminação e de pontos de vista.

Algoritmos de LPR podem ser de grande auxílio e utilizados em aplicações em vários tipos de ambientes, como por exemplo, rodízio de veículos em grandes cidades onde a permissão de circular nas vias é controlada com base no último dígito da placa, aplicação de multas por excesso de velocidade, busca de veículos roubados, entre outras.

Este trabalho contribui com uma metodologia para detecção e reconhecimento de placas de licenciamento veicular em ambientes não controlados, utilizando aprendizagem de máquina. Para a fase de detecção da placa foi utilizada a biblioteca YOLO (REDMON et al., 2016). Foi criado um *dataset* com imagens de placas de veículos obtidas a partir de quadros de vídeo, que foram gravados, para realizar o treinamento. A partir da detecção das placas nas imagens, foi realizada a segmentação da região de interesse da placa, e após isso, foi realizada a segmentação dos caracteres utilizando técnicas de processamento de imagem (conversão para tons de cinza, filtro gaussiano para suavização, filtro laplaciano para realce de bordas, *threshold* de Otsu, algoritmo para detecção de contornos e equalização de histograma). Para a fase de reconhecimento dos caracteres foram utilizadas duas redes neurais convolucionais, uma para letras e outra para números, sendo necessário

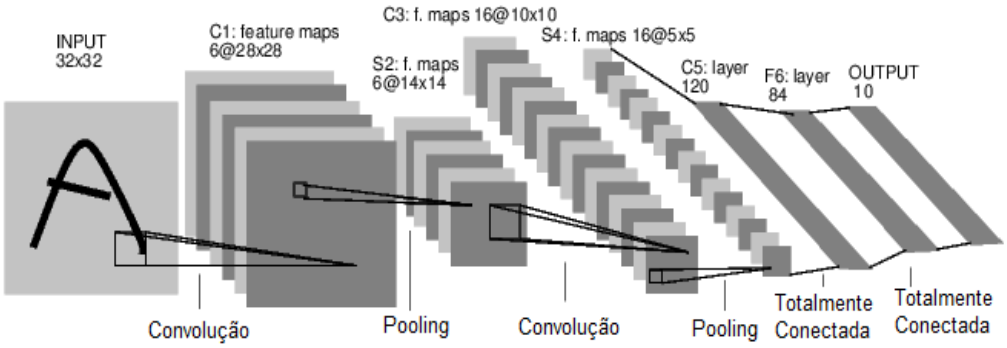
criar dois *datasets* para o treinamento das redes.

O trabalho está organizado da seguinte maneira. Na Seção 2 são tratados os conceitos a respeito de redes neurais convolucionais. A Seção 3 apresenta a biblioteca YOLO, utilizada para detecção das placas em tempo real. Na Seção 4 é apresentado o método proposto para resolver o problema de detecção e reconhecimento de placas de licenciamento veicular em tempo real. A Seção 5 apresenta os experimentos e resultados obtidos a partir da metodologia desenvolvida. Por fim, na Seção 6 são feitas as considerações finais do trabalho.

2. REDES NEURAIS CONVOLUCIONAIS

Uma rede neural convolucional (*Convolutional Neural Network* – CNN) é muito utilizada em visão computacional devido a sua facilidade de extrair padrões dos pixels sem a necessidade de se realizar algum tipo de pré-processamento. Um dos primeiros projetos utilizando CNN foi a LeNet (Figura 2), um método para reconhecer dígitos (SHEN; WANG, 2018). As CNNs são formadas por uma sequência de camadas, e cada uma delas possui uma função específica, sendo as três principais camadas: convolucionais, *pooling* e totalmente conectadas (ARAÚJO et al., 2017) (LECUN et al., 1989).

Figura 2. Ilustração da arquitetura de uma LeNet que classifica imagens de entradas em caracteres e suas três principais camadas: convolucionais, de *pooling* e totalmente conectadas.

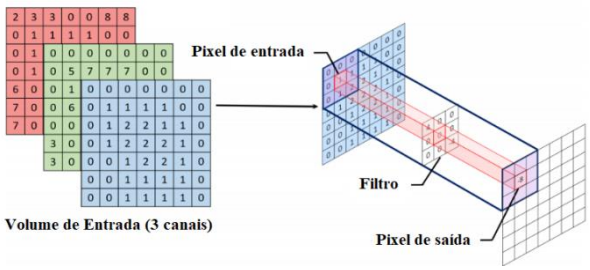


Fonte: (LECUN et al., 1989).

2.1 Camada de convolução

Na camada de convolução é realizado o mapeamento das características mais relevantes da imagem. São utilizados filtros para realizar as convoluções, gerando os mapas de características. Cada filtro possui dimensão reduzida, porém, ele se estende por toda a profundidade do volume de entrada. Por exemplo, se a imagem for colorida, então ela possui três canais (R, G, B) e o filtro da primeira camada convolucional terá o tamanho 5x5x3 (cinco de largura, cinco de altura e três de profundidade) (ALBAWI; MOHAMMED; AL-ZAWI, 2017). Durante a fase de treinamento da CNN, os filtros são ajustados de acordo com o que está sendo treinando para extrair as características mais relevantes daquela imagem (ALBUQUERQUE, 2001). Na Figura 3 encontra-se uma ilustração da operação de convolução, utilizando um filtro de dimensão 3x3.

Figura 3. Ilustração de convolução entre um filtro 3x3 e o volume de entrada.

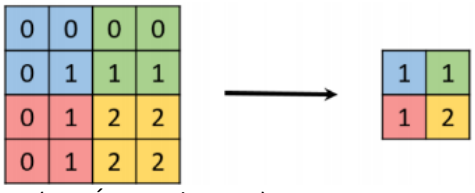


Fonte: (ARAÚJO et al., 2017).

2.2 Camada de *pooling*

Após a camada convolucional, geralmente existe uma camada de *pooling*. O objetivo dessa camada é reduzir progressivamente a dimensão espacial do volume de entrada, consequentemente a redução diminui o custo computacional da rede. A forma mais comum de *pooling* consiste em substituir os valores de uma região pelo valor máximo, essa operação é conhecida como *max-pooling* (Figura 4).

Figura 4. Aplicação de *max-pooling* em uma imagem 4x4 utilizando um filtro 2x2.



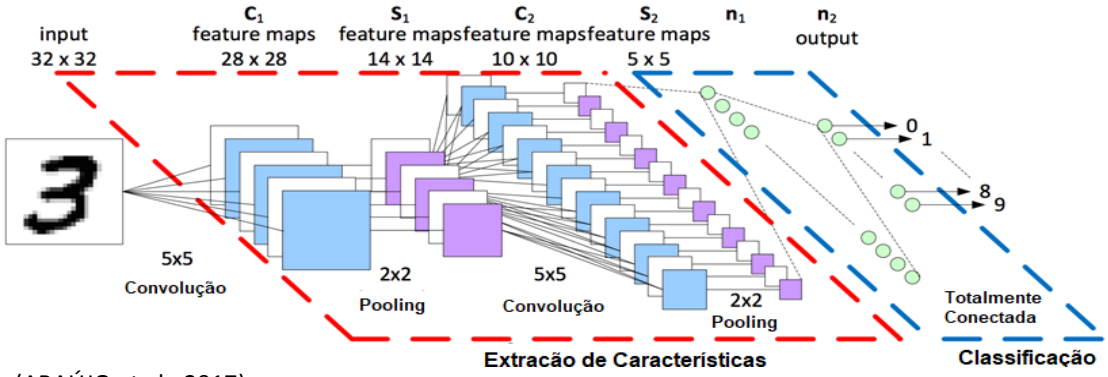
Fonte: (ARAÚJO et al., 2017).

Essa operação é útil para eliminar valores desprezíveis, reduzindo a dimensão da representação dos dados e acelerando a computação necessária para as próximas camadas, além de criar uma invariância a pequenas mudanças e distorções locais (GUO et al., 2017).

2.3 Camada totalmente conectada

A partir dos mapas de características da imagem, gerados pelas camadas anteriores, a camada totalmente conectada é responsável por traçar um caminho de decisão para cada classe de resposta. As camadas totalmente conectadas são exatamente iguais a uma rede neural artificial convencional MLP (*Multilayer Perceptron*). Na Figura 5 é ilustrado o processo de extração de características de uma imagem até que seja realizada a classificação pela camada totalmente conectada (VARGAS; PAES; VASCONCELOS, 2016) (ARAÚJO et al., 2017).

Figura 5. Ilustração da extração de características de uma imagem por uma CNN e sua posterior classificação



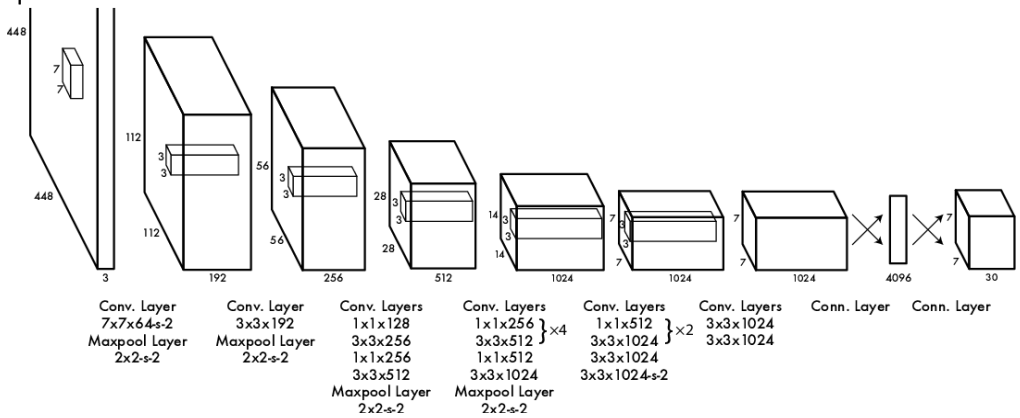
Fonte: (ARAÚJO et al., 2017).

3. *YOU ONLY LOOK ONCE* (YOLO)

You Only Look Once (YOLO) é uma biblioteca para detecção e reconhecimento de objetos em tempo real, possui uma alta eficiência para realizar o reconhecimento e detecção de objetos, para obter esse desempenho é utilizado uma única CNN. A arquitetura da YOLO possui 24 camadas para extração de características, sendo 18 camadas de convolução, 6 camadas de *pooling* e duas camadas totalmente conectadas, permitindo que possa ser utilizado para diferentes tipos de objetos (REDMON et al., 2016).

Na Figura 6 está mostrada a arquitetura utilizada na biblioteca YOLO.

Figura 6. Arquitetura da biblioteca YOLO.

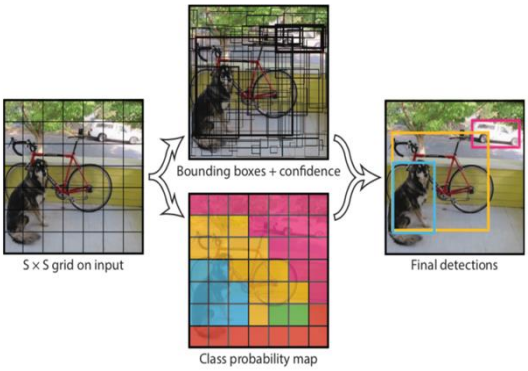


Fonte: (REDMON et al., 2016).

A biblioteca divide a imagem em uma grade *SxS* (Figura 7), cada cédula da grade é responsável por realizar a detecção do objeto contido nela, prever caixas que demarcam a localização do objeto, fazer a pontuações de confiança para cada classe de objeto a ser reconhecido.

As pontuações são obtidas pela multiplicação entre a probabilidade do objeto e IoU (união sobre a intersecção) da cédula da grade com a caixa, para verificar a precisão da caixa em conter um objeto, caso a pontuação seja zero, significa que não existe objeto dentro da caixa, caso exista algum objeto, a pontuação união sobre a intersecção entre a caixa prevista e a verdade básica. As caixas selecionas são aquelas que possuem maior pontuações. Em um cenário que exista mais de uma caixa sobrepondo o mesmo objeto, será selecionada aquela que possuir a maior pontuação (BENJDIRA et al., 2019).

Figura 7. Funcionamento da YOLO.

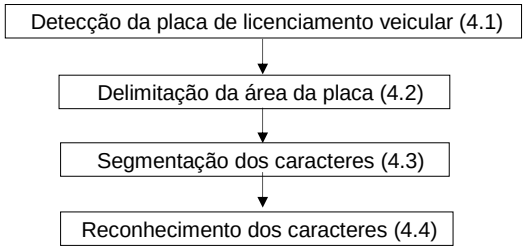


Fonte: (REDMON et al., 2016).

4. MÉTODO PROPOSTO

Nesta seção é descrito o funcionamento do método proposto, sendo dividido em quatro etapas: Detecção da placa de licenciamento veicular, delimitação da área da placa, segmentação dos caracteres, reconhecimento dos caracteres (Figura 8).

Figura 8. Fluxograma representando as etapas do método proposto.



Fonte: Autor (2019).

4.1 Detecção da placa de licenciamento veicular

Para realizar a detecção das placas de licenciamento veicular foi utilizada a biblioteca YOLO com um *dataset* de imagens de placas para o treinamento. As imagens do *dataset* foram obtidas a partir de quadros de vídeos capturadas com uma GoPro Hero 5 black, que permitiu capturar imagens de placas de carros (licenciamento veicular) em vários ambientes. Os vídeos foram gravados com resolução de 1920x1080 pixels. A câmera foi fixada em alguns ambientes com grande fluxo de veículos, e também fixada no capô de um veículo percorrendo diversas vias, permitindo gravar vídeos em movimento. O objetivo foi obter diferentes tipos de ambientes, diferentes variações de iluminação e de distância dos veículos. Foram gravados aproximadamente 10 horas de vídeos, e selecionadas as imagens que continham placas com um bom estado de conservação, obtendo 6.606 imagens de placas. Todas as placas utilizadas para o treinamento foram mapeadas manualmente, devido as variações de lugar onde as placas são encontradas na imagem. Na Figura 9 são mostradas algumas imagens de exemplo contidas no *dataset*. Não foi aplicada nenhuma técnica de aumento de dados (*data augmentation*) (PRASAD, 2017) nas imagens utilizadas no treinamento, deixando o modelo treinado mais próximo possível da realidade encontrada. A técnica de aumento de dados consiste em aplicar filtros nas imagens do *dataset,* para ampliar a quantidade de imagens usados no treinamento, o que pode provocar distorções nas imagens do *dataset* quando aplicado.

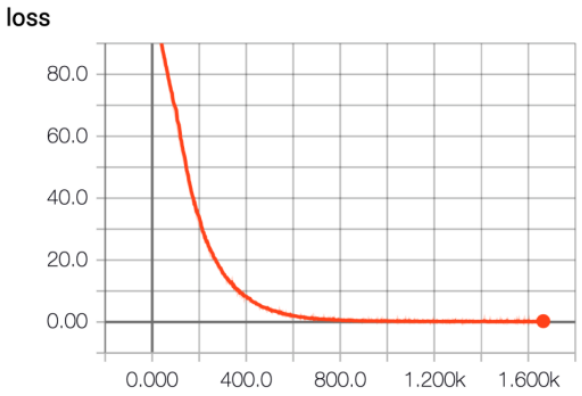
Figura 9. Exemplo de imagens contidas no *dataset* utilizado para realização do treinamento da YOLO.

Fonte: Autor (2019).

O erro da YOLO durante o treinamento foi de 0,2631, quanto menor esse valor significa maior o número de acertos. Esse erro é calculado com a soma dos erros: erro da classificação, erro de localização e erro de confiança. Na Figura 10 é apresentado um gráfico gerado no tensorboard

com 1600 etapas, em cada etapa foram utilizadas 32 imagens. Tensorboard é um conjunto de ferramentas da biblioteca TensorFlow para simplificar a visualização dos dados gerados durante o treinamento.

Figura 10. Gráfico gerado no tensorboard para o erro obtido pelo YOLO durante o treinamento, o lado esquerdo representa o erro e o lado inferior representa as etapas do treinamento



Fonte: Autor (2019).

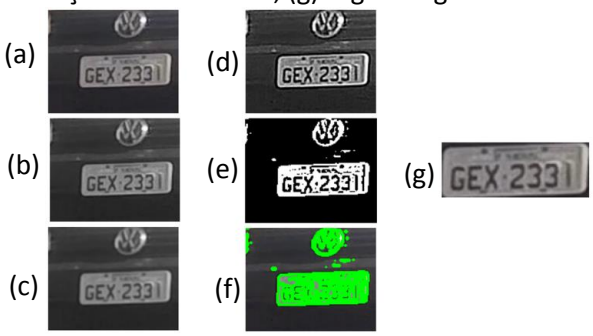
4.2 Delimitação da área da placa

Na etapa de detecção, o contorno da placa não é obtido de forma exata, sendo necessário delimitar a região antes da etapa de segmentação dos dígitos (Figura 11a).

Na etapa de delimitação da área da placa, inicialmente, a imagem colorida (RGB) é convertida para tons de cinza com apenas um canal (Figura 11b). Após, é aplicado um filtro gaussiano para suavizar a imagem com o objetivo de eliminar ruídos (JESUS; COSTA-JR, 2015) (Figura 11c), e o filtro laplaciano (DIAS; SOARES; FONSECA, 2011) para realçar as bordas da placa na imagem (Figura 11d). Com as bordas da placa realçadas, a imagem é binarizada utilizando o *threshold* (limiar) de Otsu (OTSU, 1979) para diminuir a variância entre os tons de cinza (Figura 11e). Na sequência é utilizado um algoritmo de detecção de contornos (LECUN, 1998), que faz os contornos de todos os objetos na imagem (Figura 11f). Para que algum contorno seja considerado a região da placa, é necessário que ocupe mais de 50% da largura da região detectada e mais de 30% da altura (Figura 11g).

Em alguns casos, quando a tonalidade da cor do veículo é muito similar a cor da placa, ocasiona falhas na borda da placa (Figura 12a), durante a etapa do pré-processamento da delimitação da área da placa.

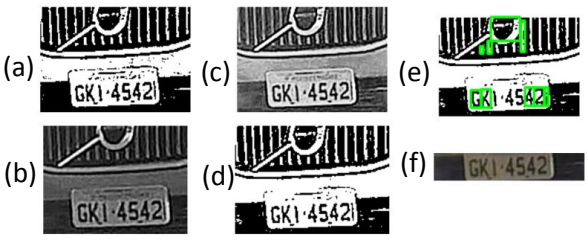
Figura 11. Processo para delimitar a região da placa. (a) região detectada pela YOLO, (b) imagem em tons de cinza, (c) filtro gaussiano, d) filtro laplaciano, (e) *threshold* de Ostu, (f) detecção de contornos, (g) região segmentada.



Fonte: Autor (2019).

Para resolver esses problemas foi necessário realizar outro processo, utilizando a imagem em que foi aplicado o filtro laplaciano (Figura 12b). A partir dessa imagem é realizada uma equalização de histograma, que é uma técnica que procura redistribuir os valores de tons de cinza dos pixels em uma imagem, de modo a obter um histograma uniforme (Figura 12c) (VIEIRA-NETO; MARQUES-FILHO, 1999). Em seguida, a imagem é binarizada utilizando o *threshold* de Otsu (Figura 12d). Para localizar a placa, é utilizado o algoritmo de detecção de contornos, excluindo os contornos muito pequenos e agrupando os contornos por altura e localização (Figura 12e). É considerado o grupo com o maior número de contornos onde se localiza os caracteres da placa, utilizando as coordenadas das alturas, é realizada a segmentação (Figura 12f).

Figura 12. Processo para encontrar a região da placa. (a) exemplo de placa que não possui o contorno por completo, (b) filtro laplaciano, (c) equalização de histograma, (d) *threshold* de Otsu, (e) detecção de contornos aplicado para excluir contornos pequenos, (f) região segmentada.



Fonte: Autor (2019).

4.3 Segmentação dos caracteres

Com a região da placa segmentada (Figura 13a), é realizado um pré-processamento, aplicando o *threshold* de Otsu nessa região, destacando os caracteres, e eliminando alguns ruídos (Figura 13b). Em seguida é utilizado o algoritmo para detecção de bordas para detectar os contornos dos caracteres. Os contornos que possuem uma altura muito diferente da maioria dos contornos são excluídos, sendo muito pequenos ou muito grandes.

Figura 13. (a) Região segmentada, (b) *threshold* de Otsu, (c) contorno dos caracteres.

Fonte: Autor (2019).

Com a localização dos sete contornos detectados (Figura 13c), é realizada a segmentação individual dos caracteres.

Existem casos em que não foi possível detectar todos os sete caracteres, devido ao desgaste da placa, caracteres apagados, entre outros fatores. Os caracteres que não se completam, fazem com que os contornos sejam divididos em dois (Figura 14a). Nesse caso é verificada a distância entre os caracteres (quantidade de pixels entre os caracteres). Quando a distância entres os caracteres for muito grande, significa que existe algum caractere naquela coordenada que não foi detectado, então é realizado o contorno tomando como base os outros caracteres já encontrados (Figura 14b).

Figura 14. (a) Placa com caractere "I" divididos em dois devido ao desgaste da placa, (b) correção aplicada possibilitando que todos os caracteres sejam segmentados.

Fonte: Autor (2019).

Os caracteres com desgaste podem ocorrer nas laterais da placa (Figura 15a). Nesse caso é verificando se não foi detectado três contornos antes do espaço que separa as letras dos números. São buscados contornos com a altura e largura da média dos outros caracteres detectados, até que sejam completados três contornos. Caso os sete contornos ainda não tenham sido detectados, é realizado o mesmo processo para o lado direito do espaço, até que sejam completados os quatro números (Figura 15b).

Figura 15. Imagem com desgaste dos números "7" e "1", (a) *threshold* de Otsu, (b) detecção de todos os caracteres.

Fonte: Autor (2019).

Outro problema encontrado na etapa de segmentação dos caracteres é que existe a possibilidade das bordas laterais da placa ser confundidos com a letra "I" e o número "1", sendo necessário realizar a exclusão desses falsos caracteres (Figura 16a). Sabendo que a maior distância entre os caracteres está no espaço entre a última letra e o primeiro número, caso a quantidade de contornos for mais que três antes do espaço, o algoritmo entende que foi detectada uma borda do lado esquerdo, sendo necessário excluir o contorno dessa borda. Caso o algoritmo seja encontrado mais do que quatro contornos do lado direto, é excluído o último contorno à direita.

Os caracteres eventualmente são unidos, devido aos furos dos parafusos existentes na parte inferior da placa (Figura 16a). A maioria dos caracteres possuem tamanhos similares, e para verificar se ocorreu a união de caracteres, é verificado se a largura desse contorno é o dobro da largura dos demais contornos, caso aconteça esse problema, o contorno é a divido ao meio obtendo dois contornos (Figura 16b).

Figura 16. (a) placa com caracteres unidos e bordas da placa detectada como sendo uma letra, (b) placa com caracteres divididos e borda excluída.

(a)

(b)

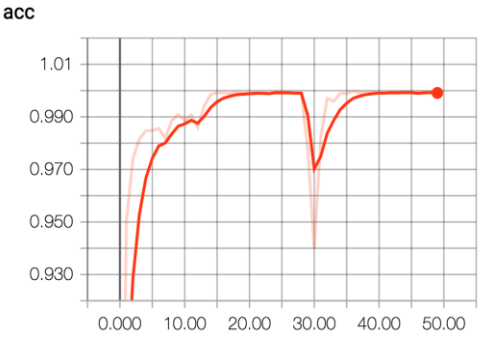
Fonte: Autor (2019).

4.4 Reconhecimento dos caracteres

Com os caracteres segmentados individualmente, é realizado o reconhecimento utilizando duas redes neurais convolucionais (*Convolutional Neural Network* – CNN), uma rede para as letras e outra rede para reconhecer os números, ambas com a mesma arquitetura. O fato de usar duas redes ao invés de usar apenas uma, evita erros como confundir o número "0" com a letra "D". Os *datasets* de letras e números foram construídos a partir do recorte dos caracteres de 1780 imagens, obtidos dos vídeos gravados (Seção 4.1).

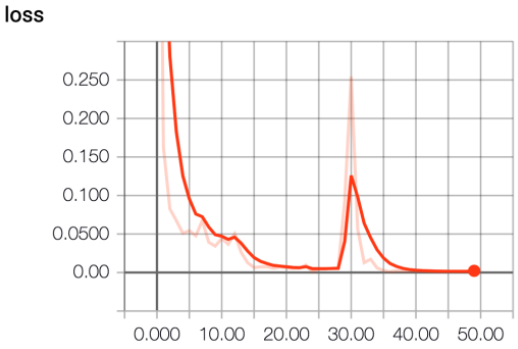
A arquitetura dessas redes é composta por duas camadas de convolução e uma camada totalmente conectada. Não foi necessário possuir muitas camadas, devido ao tamanho dos caracteres serem pequenos, 7x7 pixels. Para realizar o treinamento da rede para reconhecer letras, foi utilizado um *dataset* de 5109 imagens de letras do alfabeto de A-Z, divididas em 26 classes, em que cada classe possui em média 100 imagens. Para o treinamento, o *dataset* foi divido 90% utilizado para o treinamento e 10% para validação. O treinamento obteve uma acurácia de 99,98 % (Figura 17) e um erro de 0,002% (Figura 18), nas 50 épocas de treinamento, cada época representa a utilização de todo o *dataset* uma vez.

Figura 17. Gráfico gerado no tensorboard, representando a acurácia durante as 50 épocas do treinamento para reconhecer letras.



Fonte: Autor (2019).

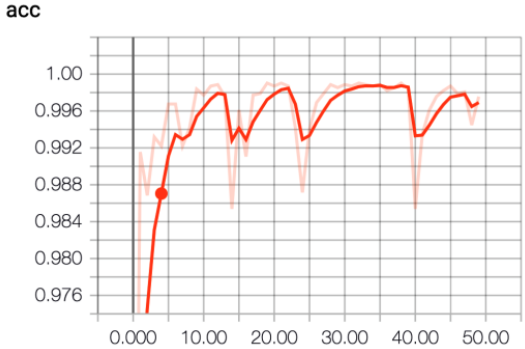
Figura 18. Gráfico gerado no tensorboard, representando o erro durante as 50 épocas do treinamento para reconhecer letras.



Fonte: Autor (2019).

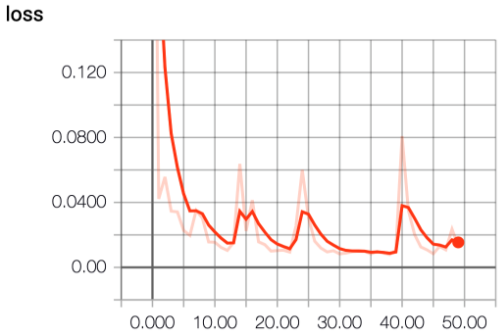
Para o treinamento da rede para reconhecer números, foi utilizado um *dataset* de 6532 imagens, dividido em dez classes, representando os números de 0 a 9, em cada classe possui em média 500 imagens. Esse é um *dataset* maior em relação ao *dataset* de letras, devido ao fato de que as placas possuírem mais números do que letras. O treinamento obteve uma acurácia de 99,76% (Figura 19) e um erro de 0,013% (Figura 20).

Figura 19. Gráfico gerado no tensorboard, representando a acurácia durante as 50 épocas do treinamento para reconhecer números.



Fonte: Autor (2019).

Figura 20. Gráfico gerado no tensorboard, representando o erro durante as 50 épocas do treinamento para reconhecer números.



Fonte: Autor (2019).

5 EXPERIMENTOS

Os experimentos foram executados utilizando um laptop com processador core i5 com 4 GB de memória RAM. Foi utilizado um vídeo de 10 minutos com resolução de 1920x1080 pixels (diferente do vídeo utilizado para obter as imagens utilizadas nos treinamentos das redes), gravado em um local movimentado com um grande fluxo de veículos, na proximidade de uma quebra-molas, para que os veículos passassem com velocidade reduzida.

A análise dos experimentos foi realizada utilizando o seguinte critério: quando em dois quadros seguidos do vídeo forem obtidos os mesmos resultados, o é considerado um resultado válido. No vídeo gravado passaram 60 veículos, em que todos os 60 veículos foram detectados pela YOLO, ocasionando em 100% de acertos para detecção. A segmentação obteve 86,66% de acertos, segmentando 52 placas corretamente. Para avaliar o reconhecimento da placa foram utilizados todos os frames do vídeo que o processo detecção e segmentação funcionaram corretamente, ao total foram 170 imagens de placas. A porcentagem de acertos no reconhecimento das letras foi de 93,69% e no reconhecimento dos números foi de 97,97% (Tabela 1). Todo o processo de detecção, segmentação e reconhecimento foi executado em média 3,9 frames por segundo. Foi obtido 83,33% de taxa de acerto, realizando todo o processo corretamente em 50 placas.

Tabela 1. Resultados obtidos pelo reconhecimento dos caracteres.

Tipo	Quantidade	Acerto	Erro	Acerto [%]
Letras	510	482	28	93,68 %
Números	680	668	12	97,97%

Fonte: Autor (2019).

6. CONSIDERAÇÕES FINAIS

Os resultados obtidos neste trabalho mostram que a metodologia proposta funciona para ambientes não controlados, podendo ser aplicado em fiscalizações de trânsito em vários ambientes. Utilizando um computador com uma placa vídeo, o algoritmo poderá ser executado de forma mais eficiente. De acordo com Redmon et al. (2016), a biblioteca YOLO é capaz de executar 45 frames por segundo.

Este trabalho utilizou de técnicas de processamento de imagem e técnicas de *deep learning,* fazendo a combinação de ambos, para se obter um bom desempenho. As técnicas de processamento de imagem foram aplicadas para diminuir a quantidade de informação e, consequentemente diminuir o tempo de processamento e aumentar a quantidade de frames por segundo a ser processado.

Algumas limitações podem ocorrer, principalmente na etapa de reconhecimento dos caracteres. Nessa etapa podem ocorrer previsões erradas quando os caracteres possuem características muito semelhantes, como é o caso da letra “O” e a letra “Q”, e letras com fontes de formatação semelhantes da placa de licenciamento veicular brasileira, são detectadas como sendo placas (Figura 21).

Figura 21. Conjunto de caracteres detectados (contorno amarelo) como sendo uma placa, devido a semelhança com fonte de formatação das placas.

Fonte: Autor (2019).

Não foi considerado para o treinamento o novo modelo de placa no padrão Mercosul

(Figura 22), isto porque durante a coleta das imagens para a montagem do *dataset* não foram encontradas muitas dessas placas em circulação.

Figura 22. Novo modelo de placa no padrão Mercosul.

Fonte: (Revista Quatro Rodas, 2019).

REFERÊNCIAS

ALBAWI, S.; MOHAMMED, T. A.; AL-ZAWI, S. Understanding of a convolutional neural network. *In*: IEEE INTERNATIONAL CONFERENCE ON ENGINEERING AND TECHNOLOGY (ICET), Antalya, 2017. https://doi.org/10.1109/ICEngTechnol.2017.8308186

ALBUQUERQUE, M. P. **Processamento de imagens:** métodos e análises. Rio de Janeiro: FACET, 2001.

ARAÚJO, F. H. D.; CARNEIRO, A. C.; SILVA, R. V.; MEDEIROS, F. N. S.; USHIZIMA, D. M. Redes Neurais Convolucionais com Tensorflow: Teoria e Prática. In: III ESCOLA REGIONAL DE INFORMÁTICA DO PIAUÍ. **Anais – ERI 2017**, v. 1, n. 1, Piauí, p. 382-406, 2017.

BENJDIRA, B.; KHURSHEED, T.; KOUBAA, A.; AMMAR, A.; OUNI, K. Car Detection using Unmanned Aerial Vehicles: Comparison between Faster R-CNN and YOLOv3. In: 1ST INTERNATIONAL CONFERENCE ON UNMANNED VEHICLE SYSTEMS (UVS), Oman, 2019. https://doi.org/10.1109/UVS.2019.8658300

CONTRAN. Resolução 231. Disponível em: http://www.denatran.gov.br/download/Resolucoes/RESOLUCAO_231.pdf. Acesso em: 15 ago. 2018.

CORNETO, G. L.; SILVA, F. A.; PEREIRA, D. R.; ALMEIDA, L. L.; ARTERO, A. O.; PAPA, J. P.; ALBUQUERQUE, V. H. C.; SAPIA, H. M. A New Method for Automatic Vehicle License Plate Detection. **IEEE Latin America Transactions**, v. 15, p. 75-80, 2017. https://doi.org/10.1109/TLA.2017.7827890

DETRAN SP: Frota de Veículos em SP – por tipo de veículo. Disponível em: https://www.detran.sp.gov.br/wps/wcm/connect/portaldetran/detran/detran/estatisticastransito/safrotaveiculos/d28760f7-8f21-429f-b039-0547c8c46ed1. Acessado em 17 ago. 2018.

DIAS, F. G., SOARES, H. C.; FONSECA, L. M. G. Estudo da aplicação de filtros de detecção de bordas na identificação da frente termal da Corrente do Brasil. **Anais XV Simpósio Brasileiro de Sensoriamento Remoto - SBSR**, Curitiba, PR, p. 7086-7092, 2011.

GUO, T.; DONG, J.; LI, H.; GAO, Y. Simple convolutional neural network on image classification. In: IEEE 2ND INTERNATIONAL CONFERENCE ON BIG DATA ANALYSIS (ICBDA) Beijing, p. 721-724, 2017. https://doi.org/10.1109/ICBDA.2017.8078730

ISMAIL, M. License plate Recognition for moving vehicles case: At night and under rain condition. In: IEEE SECOND INTERNATIONAL CONFERENCE ON INFORMATICS AND COMPUTING (ICIC), Jayapura, Indonesia, 2017. https://doi.org/10.1109/IAC.2017.8280649

JAGTAP, J.; HOLAMBE, S. Multi-Style License Plate Recognition using Artificial Neural Network for Indian Vehicles. *In*: IEEE INTERNATIONAL CONFERENCE ON INFORMATION, COMMUNICATION, ENGINEERING AND TECHNOLOGY (ICICET), Pune, India, 2018. https://doi.org/10.1109/ICICET.2018.8533707

JESUS, E. O.; COSTA-JR, R. A utilização de filtros gaussianos na análise de imagens digitais. **Proceeding.** Series of the Brazilian Society of Computational and Applied Mathematics, v. 3, n. 1, 2015. https://doi.org/10.5540/03.2015.003.01.0118

LECUN, Y.; BOSER, B.; DENKER, J. S.; HENDERSON, D.; HOWARD, R. E.; HUBBARD, W.; JACKEL, L. D. Backpropagation applied to handwritten zip code recognition. **Neural Computation**, v. 1, n. 4, p. 541-551, 1989. https://doi.org/10.1162/neco.1989.1.4.541

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278-2324, 1998. https://doi.org/10.1109/5.726791

LEE, S.; SON, K.; YOON, B.; PARK, J. Video Based License Plate Recognition of Moving Vehicles Using Convolutional Neural Network. In: 18th INTERNATIONAL CONFERENCE ON CONTROL, AUTOMATION AND SYSTEMS (ICCAS), Daegwallyeong, South Korea, p. 1634-1636, 2018.

OTSU, N. A Threshold Selection Method from Gray-Level Histograms. **IEEE Transactions on Systems, Man and Cybernetics**, v. 9, n. 1, 1979. https://doi.org/10.1109/TSMC.1979.4310076

PRASAD, P. Data Augmentation Techniques in CNN using Tensorflow. 2017. Disponível em: https://medium.com/ymedialabs-innovation/data-augmentation-techniques-in-cnn-using-tensorflow-371ae43d5be9. Acessado em: 12 jan. 2019.

REDMON, J.; DIVVALA, S.; GIRSHICK, R.; FARHADI, A. You Only Look Once: Unified, Real-Time Object Detection. IN: IEEE CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION (CVPR), Las Vegas, NV, p. 779-788, 2016. https://doi.org/10.1109/CVPR.2016.91

REVISTA QUATRO RODAS. Placa do Mercosul: tire suas dúvidas e saiba o que já mudou no projeto. 2019. Disponível em: https://quatrorodas.abril.com.br/auto-servico/placa-do-mercosul-tire-suas-duvidas-e-saiba-o-que-ja-mudou-no-projeto/ . Acesso em: 03 set. 2021

SHEN, W.; WANG, W. Node Identification in Wireless Network Based on Convolutional Neural Network. In: 14th IEEE INTERNATIONAL CONFERENCE ON COMPUTATIONAL INTELLIGENCE AND SECURITY (CIS), Hangzhou, p. 238-241, 2018. https://doi.org/10.1109/CIS2018.2018.00059

SILVA, F. A.; ARTERO, A. O.; PAIVA, M. S. V. ; BARBOSA, R. L. ALPRS - A New Approach for License Plate Recognition Using the SIFT Algorithm. **Signal & Image Processing: An International Journal**, v. 4, p. 17-33, 2013. https://doi.org/10.5121/sipij.2013.4102

VARGAS, A. C. G.; PAES, A.; VASCONCELOS, C. N. Um estudo sobre redes neurais convolucionais e sua aplicação em detecção de pedestres. *In:* CONFERENCE ON GRAPHICS, PATTERNS AND IMAGES. 24., 2016. **Proceedings […],** 2016.

VIEIRA-NETO, H.; MARQUES-FILHO, O. **Processamento digital de imagens**. Brasport, 1999.

WANG, N.; ZHU, X.; ZHANG, J. License Plate Segmentation and Recognition of Chinese Vehicle Based on BPNN. In: 12th IEEE INTERNATIONAL CONFERENCE ON COMPUTATIONAL INTELLIGENCE AND SECURITY (CIS), Wuxi, China, p. 403-406, 2016. https://doi.org/10.1109/CIS.2016.0098