Avaliação Do Knn Para Reconhecimento De Placas Veiculares No Modelo Brasileiro

Resumo: *Pesquisa experimental e bibliográfica que avalia a eficiência do algoritmo KNN para o reconhecimento automático de placas veiculares segundo o modelo vigente no Brasil em 2018. Utiliza imagens veiculares retiradas da internet de forma aleatória e compara o resultado da classificação utilizando o algoritmo KNN com a inspeção visual. Foram utilizadas 3000 imagens pré-classificadas para treinar o algoritmo e 100 imagens para testes e avaliação. Descreve os problemas encontrados durante a avaliação e apresenta uma taxa de acerto de 45,04% para arquivos jpg e 94,24% para arquivos png.*

Palavras chaves: KNN, Visão Computacional, Reconhecimento de Caracteres.

Abstract: *Experimental and bibliographic research that evaluates the efficiency of the KNN algorithm for the automatic recognition of vehicular plates according to the model in force in Brazil in 2018. It uses vehicular images taken from the internet at random and compares the result of the classification using the KNN algorithm with the visual inspection. We used 3000 pre-classified images to train the algorithm and 100 images for testing. Describes the problems encountered during the evaluation and presents a hit rate of 45.04% for jpg files and 94.24% for png files.*

Keywords: KNN, Computational Vision, Character recognition.

1 Introdução

O uso da inteligência artificial neste começo de Século XXI está se tornando indispensável para a manutenção dos serviços prestados e levanta considerações referentes a criação de novos empregos e manutenção de empregos tradicionais (Korinek E Stiglitz, 2018). Esta evolução parece ser inevitável e, entre as técnicas utilizadas destacam-se as técnicas de aprendizado de máquina (*machine learning*). Uma consulta direta a sites de busca e distribuição de artigos científicos – Google Scholar – https://scholar.google.com, Microsoft Acadêmic Research - https://academic.microsoft.com/, Cornell University Library - Arxiv - https://arxiv.org/ - mostrou que 65% de todos os artigos escritos entre 2017 e 2018 contendo a expressão "artificial intelligence" estavam de alguma forma relacionados ao estudo de redes neurais artificiais.

Este artigo tem o objetivo de explorar de forma experimental o algoritmo *K-Nearest-Neighbor* (Knn). Um método de classificação que pode ser utilizado para reconhecimento de caracteres em imagens (Das, Sarkar, *et al.*, 2015). Para substituir técnicas baseadas em redes neurais (Zhang, Yin, *et al.*, 2018) (Hi, Bai E Yao., 2017) na esperança que o KNN seja eficiente e utilize menos recursos computacionais (Hazra, Singh E Daga, 2017).

Utiliza um conjunto de imagens personalizado obtido em sites de busca como o Google.com e o Bing.com para treinamento e teste do KNN de forma a avaliar o percentual de acerto do mesmo. Este artigo descreve este processo de avaliação e seus resultados.
Este artigo está dividido em quatro partes: **introdução**, esta seção; **referencial teórico;** descrevendo o conhecimento necessário a implantação e testes; **metodologia**, descrevendo as técnicas utilizadas na captação e classificação das imagens e **considerações finais**, onde estão descritos os próximos passos desta pesquisa.

2 Referencial Teórico

De acordo com Bradski e Kaehler (2018) (a visão computacional pode ser definida como "... a transformação de dados de uma câmera fixa ou de vídeo em uma decisão ou em uma nova representação.". Ou seja, através de ferramentas de visão computacional, é possível manipular imagens ou vídeo, a fim de obter informações relevantes para atingir determinado objetivo, mas para isso, é necessário entender como a máquina "enxerga". Este método pode ser utilizado para identificar objetos, pessoas ou caracteres em imagens ou vídeos (Das, Sarkar, *et al.*, 2015) (Hazra, Singh E Daga, 2017) (Zhang, Yin, *et al.*, 2018).

O algoritmo Knn ou ***K-Nearest Neighbour*** (vizinho mais próximo) é uma técnica de aprendizado supervisionado que pode resolver um problema de classificação de da amostra X baseado na distância que existe entre um determinado exemplo de teste e amostras previamente classificadas (Pellilo, 2014) (Zhang E Zhou, 2007).

Diversos tipos de medidas de distância podem ser utilizadas para efetuar o cálculo do algoritmo Knn, neste artigo optamos por usar a distância euclidiana, por ser simples, principalmente em casos bidimensionais (Russel & Norvig, 2003). Podemos descrever o Knn, ainda segundo Suguna e Thanushdodi (2010):

- Suponha a existência de j categorias de treinamento C_1, C_2, C_3 ... depois de um processo de redução de características para a criação de um vetor de características;

- Toma-se a amostra X na forma de um vetor contendo as mesmas características das amostras de treinamento;
- Calcula-se a distância (DIST) entre as amostras de treinamento d_i e o exemplo X na forma:

$$DIST_{(X,d_i)} = \frac{\sum_{j=1}^{m} X_j \cdot d_{ij}}{\sqrt{\sum_{j=1}^{m} X_j} \cdot \sqrt{\sum_{j=1}^{m} X d_{ij}}}$$

- Escolhe-se k amostras de $DIST_{(X,d_i)}$ e calcule a probabilidade de X pertencer a cada categoria, anteriormente definida segundo:

$$P_{(X,C_j)} = \sum_{s} DIST_{(X,d_i)} \cdot y(d_i, C_j)$$

Onde $y(d_i, C_j)$ é a função de atribuição de categoria dada por:

$$y(d_i, C_j) = \begin{cases} 1, & d_i \in C_j \\ 0, & d_i \notin C_j \end{cases}$$

- Consideramos X pertencente a categoria com a maior $P_{(X,C_j)}$.

Neste artigo foi utilizada a linguagem de programação Python para implementação do algoritmo como descrito por Zakka (2016) com a inserção da biblioteca OpenCV (https://opencv.org/) para tratamento de imagens.

Entre os formatos de arquivos disponíveis livremente na internet, e utilizados neste artigo, encontram-se o jpg e o png. O Primeiro, jpg (*Joint Photographic Experts Group*), representa arquivos que imagens estáticas codificadas segundo o padrão ISO/IEC 10918-1de 1994 (Wallace, 1992). Trata-se de um padrão de compressão, com perdas, que define os algoritmos de compressão e descompressão de dados utilizada para armazenamento de imagens (Wallace, 1992). Já o segundo formato, png (*Portable Network Graphics*), determina os algoritmo utilizados em um processo de compressão e descompressão sem perdas (Weinberger, Seroussi, & Sapiro, 2000) também para imagens estáticas. A conversão destes arquivos em tensores para aplicação do Knn foi realizada diretamente com o uso da biblioteca OpenCV (https://opencv.org/) tornando todos os processos de descompressão relacionados aos tipos de arquivos das imagens utilizadas tanto para treinamento quanto para avalição transparentes ao objetivo deste artigo.

A redução da entropia nas imagens que serão utilizadas para classificação parece ser um fator decisivo para a eficiência de algoritmos de classificação de imagens (Das, et al., 2015) (Hazra, Singh, & Daga, 2017) (Bradaski & Kaehler, 2018). Este artigo utiliza filtros disponíveis na biblioteca OpenCv para esta finalidade, a saber: redução a escala de cinza – remove a informação de cor conservando a informação de luminosidade diminuindo a dimensionalidade do tesor criado em memória; *gaussian blur* (Desfocagem gaussiana) - técnica de suavização de imagens, reduzindo o ruído de fundo da imagem e diminuindo sua entropia por meio de uma convolução, aplicada ao tensor em memódia, determinada por:

$$g(i,j) = \sum_{k,l} f(i+k, j+l) h(k,l)$$

onde o valor do novo pixel será $g(i,j)$ e $h(j,l)$ determina a função com os parâmetros que serão utilizados para a filtragem, conforme disponível na biblioteca OpenCv (OpenCV Team, 2018); *adaptative threshold* (limiarização adaptativa) função que transforma uma imagem em escala de cinza em uma imagem binária:

$$\mathtt{dst}(x,y) = \begin{cases} \mathtt{maxValue} & \text{if } \mathtt{src}(x,y) > T(x,y) \\ 0 & \text{otherwise} \end{cases}$$

Onde $dst(x,y)$ representa o novo valor do pixel após a aplicação do filtro (OpenCV Team, 2018).

3 Metodologia

O desenvolvimento da pesquisa se deu meio de pesquisa bibliográfica, que segundo é desenvolvida com base em material já elaborado, constituído principalmente de livros e artigos científicos (GIL, 2002) e, também segundo Gil (2002) trata-se de pesquisa experimental onde "*quando se determina um objeto de estudo, seleciona-se as variáveis que seriam capazes de influenciá-lo, define-se as formas de controle e de observação dos efeitos que a variável produz no objeto*".

Todos os scripts em Python foram escritos no ambiente integrado de desenvolvimento PyCharm disponível gratuitamente na internet e executados em um notebook pessoal rodando Windows 10 com processador I7 e 8 Gbytes de memória Ram sem configurações especiais além da instalação do PyCharm.

Durante o processo de reconhecimento dos caracteres referentes as placas de licença, encontradas nas imagens selecionadas, foi aplicado um conjunto específico de algoritmos em sequência para diminuir a informação contida em cada imagem diminuindo as características que serão avaliadas pelo Knn.

Partindo-se da premissa que a imagem foi transformada em dados de forma transparente pela biblioteca OpenCV. Podemos considerar tensor em memória como sendo uma representação oda imagem e sobre este tensor aplicar os filtros selecionados na biblioteca OpenCV visando diminuir a entropia da imagem. Neste intento, foram selecionados três filtros. São eles: redução a escala de cinza para retirar toda informação de cor da imagem conservando a informação de luminosidade. A figura 1, a seguir mostra o resultado obtido em uma das imagens de teste.

Figura 1- Imagem de testes reduzida a escala de cinza. Fonte:(os autores, 2018)

Em seguida foi aplicado o filtro *gaussian-blur,* cujo efeito pode ser observada no Figura 2 a seguir:

Figura 2 - Resultado da aplicação do filtro *gaussian-blur*. Fonte: (os autores, 2018)

Por fim foi aplicado o filtro *adaptative threshold,* cujo resultado é apresentado na Figura 3:

Figura 3 – Resultado da aplicação do filtro *adaptive thresold* . Fonte (os autores, 2018)

O próximo algoritmo seleciona áreas da imagem retangulares, sequenciais onde podem existir placas de licença veicular. Para isso, foi utilizado as funções ***findContours()*** e ***rectangle()*** da biblioteca OpenCv. A primeira encontra transições bruscas na imagem – contornos, enquanto a segunda recorta retângulos na imagem que incluam estes contornos.

Estas retângulos, fragmentos de imagens que podem conter as informações de uma placa veicular, ou não são aplicados ao algoritmo Knn previamente treinado com imagens que passaram pelo mesmo processo e foram classificadas manualmente para o reconhecimento de caracteres.

A análise de eficiência considerou o percentual de acerto por caractere lido por placa e a possibilidade de identificação, ou não de uma placa na imagem. Para tanto, todos os caracteres encontrados foram comparados com os caracteres identificados visualmente contando-se positivamente as coincidências e calculando-se o percentual de acertos em referência ao total de caracteres identificados.

Durante a avaliação observou-se que as imagens em formato jpg apresentaram um índice de acerto de 45,04% porcento contra um índice de acerto de 94,24% obtido nas imagens obtidas no formato png. Pode-se especular que esta diferença seja provocada pela perda de informação inerente ao formato de compressão jpg (Wallace, 1992).

4 Considerações Finais

O algoritmo Knn se mostrou eficiente na identificação de placas veiculares a partir de imagens estáticas, aleatoriamente obtidas mesmo sendo executado em um computador pessoal sem capacidades especiais. Apresentando um percentual de acerto de 94,24% para imagens em arquivos png.

A percepção do efeito da compressão sobre o reconhecimento dos caracteres requer maiores observações e pesquisa para identificar quais características da imagem, perdidas durante o processo de compressão foram decisivas para a diferença encontrada na identificação em arquivos jpg e png. Indicando que este deve ser o tema de uma pesquisa futura.

A maior dificuldade do projeto foi a localização de imagens estáticas com placas de veículo, segundo o padrão adotado no Brasil, nos motores de busca livres, como o Google e o Bing. Para aprofundar esta pesquisa será necessário recolher manualmente as imagens para treinamento e avaliação.

5 Referências bibliográficas

BRADSKI, G., & KAEHLER, A. (2018). *Learning OpenCV.* (1ª ed.). Los Angeles, CA. USA: O'Reilly.

Das, N., Sarkar, R., Basu, S., Saha, P. K., Kundu, M., & Nasipuri., M. (1 de Jun. de 2015). Handwritten Bangla character recognition using a soft computing paradigm embedded in two pass approach. *Pattern Recognition, 48*(6), .2054-2071.

GIL, A. C. (2002). *Como Elaborar Projetos de Pesquisa* (4ª ed.). São Paulo, SP Brasil: Editora Atlas.

Hazra, T. K., Singh, D. P., & Daga, N. (2017). Optical character recognition using KNN

on custom image dataset. *IndustriaAutomation and Electromechanical Engineering Conference (IEMECON), 2017 8th Annual*, (pp. 110-114).

hi, B., Bai, X., & Yao., C. (Abr. de 2017). An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence, 39*(11), 2298-2304.

Jr., C. N., Jr., J. D., & Kaestner, C. A. (2003). Detecção automática de sentenças com o uso de expressões regulares. *III Congresso Brasileiro de Computação – CBComp 2003* , (pp. 548-560).

Korinek, A., & Stiglitz, J. E. (Jan. de 2018). Artificial Intelligence and Its Implications for Income Distribution and Unemployment. *Economics of Artificial Intelligence*.

Liao, S., Devadas, S., Keutzer, K., Tjiang, S., & Wang, A. (1995). Code Optimization Techniques for Embedded DSP Microprocessors. *Design Automation, 1995. DAC'95*, (pp. 599-604).

Liu, W., Du, W., Chen, J., Wang, W., & Zeng, G. (31 de Mai. de 2014). Adaptive energy-efficient scheduling algorithm for parallel tasks on homogeneous clusters. *Journal of Network and Computer Applications*(41), 101-113.

OpenCV Team. (15 de Mai. de 2018). *Smoothing Images*. Acesso em 01 de Mai. de 2018, disponível em OpenCV: https://docs.opencv.org/2.4/doc/tutorials/imgproc/gausian_median_blur_bilateral_filter/gausian_median_blur_bilateral_filter.html

Pellilo, M. (2014). *Who invented the nearest neighbor rule?* Acesso em 12 de Mai. de 2018, disponível em Pattern Recognition Tools: http://37steps.com/4370/nn-rule-invention/

Pilavare, M. S., & Desai, A. (Jan. de 2015). A Survey of Soft Computing Techniques based Load. *International Journal of Computer Applications , 110*(14), 0975 – 8887.

Russel, S., & Norvig, P. (2003). *Articial Intelligence - A Moden Approach* (2ª ed.). London: Pearson Education Ltd.

Sang, E. T., Canisius, S., Bosch, A. v., & Bogers, T. (2005). Applying spelling error correction techniques for improving semantic role labelling. *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL),*, (pp. 229–232). Ann Arbor.

Suguna, N., & Thanushkodi, K. (Jul. de 2010). An Improved k-Nearest Neighbor Classification Using Genetic Algorithm. *International Journal of Computer Science Issues, 7*(4), 18-21.

Wallace, G. K. (Feb. de 1992). The JPEG still picture compression standard. *IEEE transactions on consumer electronics, 38*(1), xviii-xxiv. Fonte: IEEE transactions on consumer electronics.

Weinberger, M. J., Seroussi, G., & Sapiro, G. (Ago. de 2000). The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS. *EEE Transactions on Image processing, 9*(8), 1309-1324.

Zakka, K. (2016). *A Complete Guide to K-Nearest-Neighbors with Applications in Python and R*. Acesso em 20 de Mai. de 2018, disponível em Kevin Zakka's Blog: https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/