Técnicas de Visão Computacional para a Deteção de Contentores de Resíduos

Computer Vision Approaches to Waste Containers Detection

Miguel Valente
Instituto Politécnico de Castelo Branco, Portugal
mvalente@ipcbcampus.pt

Hélio Silva
EVOX Technologies, Portugal
helio.silva@evox.pt

João M. L. P. Caldeira, Vasco N. G. J. Soares
Instituto Politécnico de Castelo Branco, Portugal
Instituto de Telecomunicações, Portugal
jcaldeira@ipcb.pt, vasco.g.soares@ipcb.pt

Pedro D. Gaspar
Universidade da Beira Interior, Portugal
dinis@ubi.pt

Resumo — **O trabalho apresentado neste artigo resulta de uma investigação preliminar que visa a utilização de técnicas de visão computacional para substituir o método atual de identificação de contentores de resíduos via identificação por radiofrequência. Comparativamente ao método atual, esta abordagem é mais ágil e diminui os recursos necessários para implementação. A abordagem aqui discutida é centrada no uso de redes neuronais convolucionais, especificamente a rede YOLO. Utilizando este método de identificação foi atingido uma precisão de deteção e classificação de 92% dos contentores de resíduos.**

Palavras Chave – contentores de residuos, identificação por radiofrequência, visão computacional, redes neuronais convolucionais, deteção de objectos, YOLO.

Abstract — **The work presented in this article is the result of a prelaminar investigation that aims at using computer vision techniques to replace the current method of performing detection of waste contains via radio-frequency identification. Comparatively to the current method, this approach is more agile and diminishes the resources needed for an implementation. The approach discussed is focused on the use of convolutional neural networks, specifically the network YOLO. Using this method of identification, it was attained an accuracy of 92% of the waste containers.**

Keywords - waste containers, radio-frequency identification, computational vision, convulutional neural networks, object detection, YOLO.

I. INTRODUÇÃO

A abordagem atual de identificação de contentores de resíduos utilizando identificação por radiofrequência (RFID) é ineficiente, de implementação e manutenção complexa e com um impacto ambiental considerável [1] [2] [3]. A utilização de métodos de identificação com base em visão computacional tem o potencial de reduzir a complexidade consideravelmente e aumentar a flexibilidade. No entanto, este método introduz um conjunto de problemas relacionados com o campo da visão computacional, tais como: oclusão parcial dos contentores, rotação, mudanças de perspetiva e iluminação, e diferenças no aspeto dos contentores. Estes problemas podem ser ultrapassados utilizando um grupo de imagens de teste apropriadas e tendo em conta o meio em que um sistema é instalado. Por exemplo, se o sistema para a deteção for colocado numa posição fixa num camião de recolha de resíduos, o problema de rotação seria eliminado.

Assim, este trabalho tem por objetivo analisar e avaliar o desempenho de redes neurais em visão computacional, com vista à proposta de uma solução inovadora, eficaz e eficiente para a deteção de contentores de resíduos.

Este artigo encontra-se estruturado da seguinte forma. Na Secção II apresenta-se o conceito de redes neurais convolucionais e em concreto a rede neuronal YOLO, discutindo a sua utilização no contexto deste trabalho. De seguida, na Secção III é apresentado o cenário para avaliação de desempenho da rede YOLO, discutindo-se os resultados obtidos. Finalmente, na Secção IV, apresentam-se as conclusões e trabalho futuro.

II. ESTADO DA ARTE

A atenção da comunidade de computação visual na última década tem estado centrada no campo de redes neuronais, uma denominação geral que representa os diferentes tipos de redes neuronais (RN) existentes. Uma rede neural convolucional (RNC) é um subtipo de RN, que tem sido utilizada frequentemente para deteção e identificação de objetos. Atualmente, estas dividem-se em dois ramos: redes de regressão/classificação e redes de propostas de regiões. O primeiro tipo de rede é conhecido pela sua superior rapidez de deteção, quando comparado com as redes de propostas de regiões. A aquisição de velocidade durante a fase de deteção tem por norma uma perda no valor da precisão, no entanto,

existem arquiteturas de rede de regressão/classificação que conseguem competir ao nível das redes de propostas de regiões no nível da precisão.

Uma rede em particular que obteve ótimos resultados em vários testes é a rede *you only look once* (YOLO) [4]. Esta consegue obter altos níveis de precisão, destacando-se a sua rapidez em realizar o processo de deteção, com resultados superiores aos de outras redes consideradas como estado da arte [5].

O trabalho apresentado neste artigo avalia o potencial da utilização da rede YOLO para realizar a deteção e classificação de contentores de resíduos, os quais podem ser classificados em diferentes categorias. Na prática esta rede mostra os resultados envolvendo o objeto detetado com uma caixa quadrada e a classe de objeto prevista. A sua abordagem é eficiente porque faz a previsão da caixa e da classe numa só avaliação. Isto significa que cada *frame* de um vídeo ou imagem é passada apenas uma vez na rede para se obter uma deteção. Os recursos computacionais requeridos pela rede YOLO, possibilitam a sua utilização na implementação futura de uma aplicação para uma plataforma móvel.

Ainda que a RNC esteja disponível nas versões YOLOv2 [6] e YOLOv3 [7], o foco deste trabalho incide sobre a YOLOv2 pois esta tem uma arquitetura menos profunda que a YOLOv3. Esta profundidade implica maior lentidão no processo de treino e de deteção. No entanto, para verificar o aumento de tempo, foi feita uma implementação da rede YOLOv3. A próxima secção descreve as condições de implementação e discute os resultados obtidos.

III. Avaliação de Desempenho

Esta secção começa por descrever o software e hardware utilizado para realizar a implementação do cenário de teste. Depois, centra-se na avaliação de desempenho.

A. Cenário de Teste

A implementação foi realizada num ASUS NJ750 standard com Ubuntu 18.04.1 LTS. As imagens usadas foram obtidas usando um Xiaomi Pocophone F1 e redimensionadas para 960x720 com formato *jpg*. A Tabela 1 contem a distribuição de imagens por grupo de uso e por contexto de iluminação.

Tabela I. Distribuição de imagens por uso e condições de iluminação.

	Noite	Dia	Total
Múltiplos Contentores por Imagem (treino da RNC)	60	84	144
Imagens sem contentores (treino da RNC)	63	81	144
Múltiplos Contentores por Imagem (teste da RNC)	9	22	31
Total	132	243	375

Para treinar uma RNC é necessário passar por um processo de *labelling*. Ou seja, colocar etiquetas com a classe e posição dos contentores em cada imagem. Para esse efeito utilizou-se o programa labelImg [8]. Após o processo de *labelling* estar completo, é possível treinar a RNC. Para tal, foi utilizada uma ferramenta disponibilizada pelos autores da rede YOLO, denominada *darknet* [9], tendo a mesma também sido utilizada durante a faze de teste. De modo a poder usufruir das capacidades de computação de uma *graphical processing unit* (GPU) foi necessário instalar: CUDA 9.0 [10] e cuDNN 7.1.4 [11]. A instalação da biblioteca OpenCV 3.2.0 [12] também é necessária pois é utilizada pela *darknet* para indicar resultados e produzir gráficos.

Para avaliação de desempenho considera-se a *mean average precison* (mAP) [13]. Esta métrica representa a precisão da identificação e classificação, tendo sido criada especificamente para avaliar o desempenho de RNCs.

B. Avaliação de Desempenho

Todas as implementações destas redes seguem os três passos essenciais: *labelling*, treino e teste. O processo da primeira fase pode encontra-se demonstrado na Figura 1. Os dois últimos passos podem ser feitos em simultâneo como demonstra a Figura 2 ou separadamente como foi feito na implementação do YOLOv3.

Figura 1. Processo de labbeling de imagens

Começando pela implementação da rede YOLOv2, para esta arquitetura de rede foram realizados dois treinos, o primeiro foi feito usando 288 imagens, metade destas não continham exemplos de contentores a detetar, isto numa tentativa de melhorar o valor da precisão da deteção e classificação. A segunda implementação da rede YOLOv2 usou 144 imagens, sendo estas aquelas que possuem contentores. Desta maneira é possível comparar como o acrescento de imagens sem contentores afeta o nível de precisão.

A Figura 2 demonstra o processo de treino da primeira implementação da rede YOLOv2, a azul encontra-se a perda média da rede e a vermelho está o nível de precisão calculado de 4 em 4 iterações. Este é baseado no grupo de imagens reservado para teste. É possível observar, que assim que a perda média atinge valores baixos, entre 0.5 e 0.06, esta começa a estabilizar. Isto significa que o processo de treino está prestes a ser concluído. Do mesmo modo, vemos o valor

da precisão a aumentar no final do treino, ficando perto dos 87%.

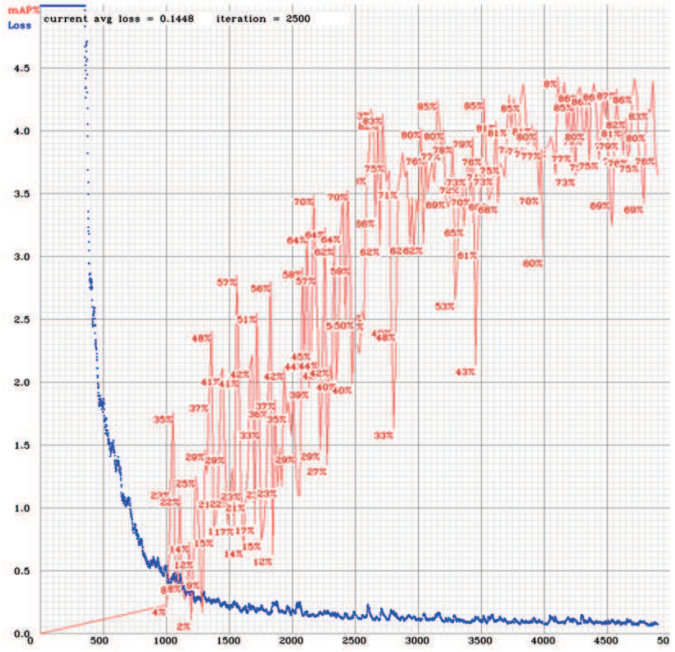

Figura 2. Gráfico de treino da primeira implementação do YOLOv2.

A Figura 3 demonstra o processo da segunda implementação do YOLOv2. Nesta a precisão começa a aumentar muito mais cedo e atinge valores de cerca de 92% perto da iteração 2000. Isto significa que a adição de imagens sem contentores abranda o tempo de aprendizagem e diminui o valor da precisão em 4%.

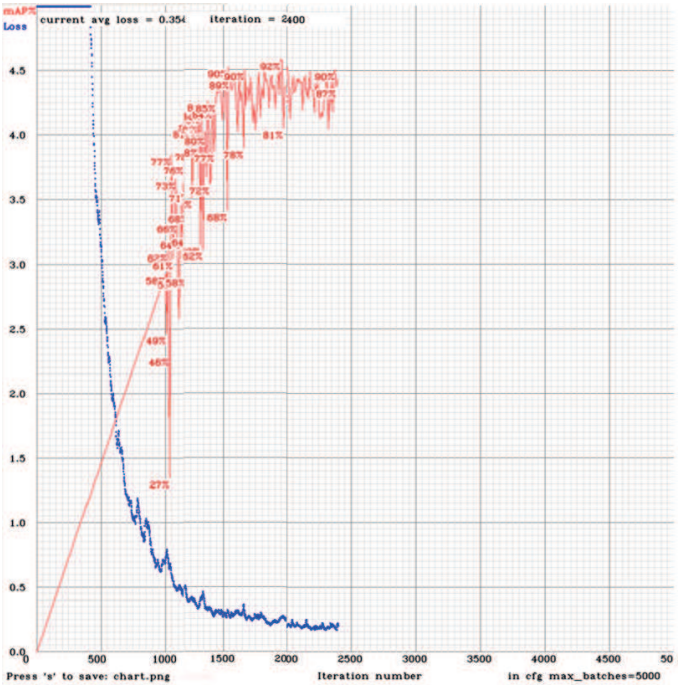

Figura 3. Gráfico de treino da segunda implementação do YOLOv2.

As Figuras 4 e 5 ilustram como a deteção e classificação do YOLOv2 funciona. Observando-as, verifica-se que a deteção funciona em ambientes com diferentes condições de iluminação, de escala, de perspetiva e até com obstrução. Esta robustez advém das características das imagens de treino, que continham diferentes perspetivas, condições de iluminação, escala e até oclusão. No entanto, foi observado num teste de vídeo que quando ocorria rotação, a deteção deixava de funcionar totalmente. Isto porque não existem imagens para treino contendo rotações. A solução passaria por adicionar exemplos contendo rotação ou rodar aleatoriamente todas as imagens do grupo de treino nas 4 direções, para ficar com uma distribuição homogénea.

Figura 4. Demonstração de deteção de contentores no período diurno.

Figura 5. Demonstração de deteção de contentores no período noturno.

Passando agora ao YOLOv3, como foi indicado acima, esta rede tem uma arquitetura mais profunda, requerendo mais tempo de treino para assegurar a deteção. De acordo com os resultados obtidos das implementações do YOLOv2 apenas foram usadas 144 imagens contendo as respetivas *labels*.

Como esperado é possível ver a complexidade da arquitetura da rede YOLOv3 na Figura 6.

Usufruindo do mesmo tempo de treino que a segunda implementação da rede YOLOv2, o treino apenas chegou à iteração número 670 com uma perda média de 1.6 que não se encontrava perto de uma fase de estabilização. Isto indica que o processo de treino necessitaria de mais tempo para chegar ao fim. Ainda assim, foi atingida uma precisão de deteção e classificação de 40% utilizando a última iteração da rede. Este valor foi obtido pós treino utilizando a *darknet*, visto que esta apenas começa a calcular a precisão durante o treino a partir da iteração número 1000.

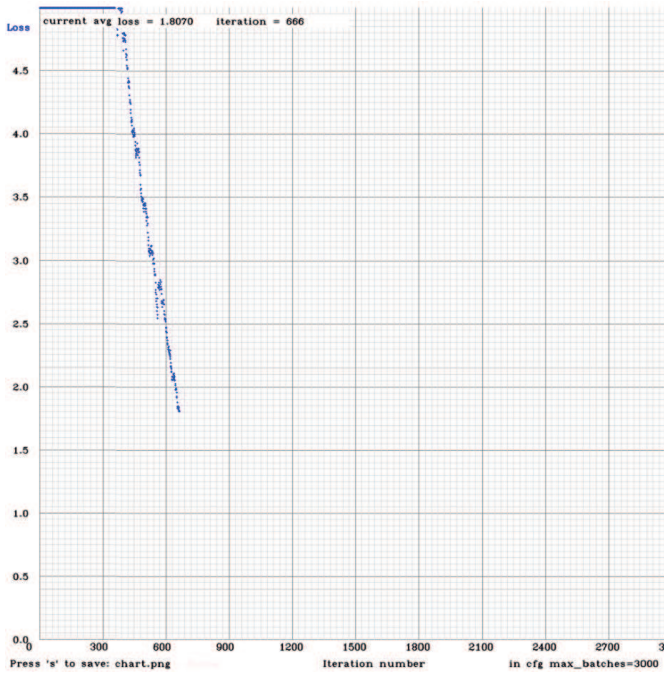

Figura 6. Gráfico de treino da implementação do YOLOv3.

IV. Conclusões e Trabalho Futuro

O trabalho aqui apresentado surgiu da necessidade da proposta de uma solução eficaz e eficiente para a deteção de contentores de resíduos. Para tal, considerou-se o uso de técnicas de visão computacional. Concretamente foi avaliado o desempenho de redes neuronais convolucionais, baseadas numa arquitetura de regressão e classificação, mais especificamente a rede YOLO.

Dos resultados observados concluiu-se que, apesar do desenvolvimento desta solução estar ainda numa fase inicial, a mesma aparenta ter um bom potencial. Verificou-se uma precisão máxima de identificação e classificação de 92% na segunda implementação da rede YOLOv2. Salientando-se que este valor foi alcançado com limitações de dados, de hardware e tempo. Estas são as principais limitações associadas ao uso de redes neuronais, influenciando a complexidade das implementações e a obtenção de resultados, ficando o seu impacto mais visível na implementação do YOLOv3.

Para trabalho futuro pretende-se aumentar o número de testes com recurso a hardware especializado, utilizando novas imagens contendo diferentes tipos e categorias de contentores. Estas imagens passarão por uma fase de *data augmentation*, de modo a expandir mais a informação disponível para o treino das redes. Deste modo, a rede conseguirá identificar mais contentores de categorias diferentes e com maior precisão. Estes testes permitirão avançar de seguida, para a implementação funcional deste conceito numa aplicação móvel.

Agradecimentos

Os autores expressam o seu agradecimento às empresas EVOX Technologies e InspiringSci, Lda pelo interesse e contribuição, determinantes para a concretização deste trabalho.

Referências Bibliográficas

[1] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 22–32, 2014.

[2] S. ABDOLI, "RFID application in municipal solid waste management system," 2009.

[3] M. G. Gnoni, G. Lettera, and A. Rollo, "A feasibility study of a RFID traceability system in municipal solid waste management," *Int. J. Inf. Technol. Manag.*, vol. 12, no. 1/2, p. 27, 2013.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[5] Z.-Q. Zhao, P. Zheng, S. Xu, and X. Wu, "Object detection with deep learning: A review," *arXiv Prepr. arXiv1807.05511*, 2018.

[6] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," *arXiv Prepr.*, 2017.

[7] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv Prepr. arXiv1804.02767*, 2018.

[8] T. Lin, "labelImg," *GitHub repository*. GitHub, 2019.

[9] A. Bezlepkin, "darknet," *GitHub repository*. GitHub, 2019.

[10] NVIDIA Corporation, *NVIDIA CUDA Compute Unified Device Architecture Programming Guide*. NVIDIA Corporation, 2007.

[11] S. Chetlur *et al.*, "cudnn: Efficient primitives for deep learning," *arXiv Prepr. arXiv1410.0759*, 2014.

[12] G. Bradski, "The OpenCV Library," *Dr. Dobb's J. Softw. Tools*, 2000.

[13] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.