

Training of YOLO Neural Network for the Detection of Fire Emergency Assets

A. Corneli^{a*}, B. Naticchia^a, M. Vaccarini^a, F. Bosché^b and A. Carbonari^a

^aPolytechnic University of Marche, DICEA, Ancona, Italy

^bInstitute for Infrastructure and Environment, University of Edinburgh, Edinburgh, UK

a.corneli@staff.univpm.it, b.naticchia@univpm.it, m.vaccarini@staff.univpm.it,
f.bosche@ed.ac.uk, a.carbonari@staff.univpm.it.

Abstract -

Building assets surveys are cost and time demanding and the majority of current methods still rely on manual procedures. New technologies could be used to support this task. The exploitation of Artificial Intelligence (AI) for the automatic interpretation of data is spreading throughout various application fields. However, a challenge with AI is the very large number of training images required for robustly detect and classify each object class.

This paper details the procedure and parameters used for the training of a custom YOLO neural network for the recognition of fire emergency assets.

The minimum number of pictures for obtaining good recognition performances and the image augmentation process have been investigated. In the end, it was found that fire extinguishers and emergency signs are reasonably detected and their position inside the pictures accurately evaluated.

The use case proposed in this paper for the use of custom YOLO is the retrieval of as-is information for existing buildings. The trained neural networks are part of a system that makes use of Augmented Reality devices for capturing pictures and for visualizing the results directly on site.

Keywords -

YOLO; Neural Network; Asset inventory

1 Introduction

Facility Management (FM) is the most costly phase of the building lifecycle, accounting for up to 80/90% of total costs [1]. For this reason, improving efficiency of FM processes can lead to significant savings. To establish an asset management system, component inventory has first to be conducted [2][3]. But, this process still relies on manual procedures that make it time-consuming, expensive and prone to errors and omissions. Construction industry is increasingly moving through digitization, consequently is growing the awareness about the value of integration of new technologies such as AI, in process automation. Component inventory is an area that could certainly bene-

fit from this. The automatic acquisition of geometric and semantic data of built assets has been pursued, principally through point cloud collecting technologies, photogrammetry and image processing. But computer vision, especially object detection using artificial intelligence (AI) has seen limited exploitation in that field, despite being aggressively pursued in others engineering fields: from autonomous driving to automated fruit picking. AI systems, and more specifically Deep Learning frameworks, generally require large datasets for training [4][5]. A challenge about this is that it is never obvious what the minimum number of pictures is for developing a well-performing neural network. You Only Look Once (YOLO) networks [6][7][8] are state-of-the-art real-time object detection and classification systems, demonstrating to be fast and accurate. The aim of this research is to investigate and detail the training process for customized YOLO Convolutional Neural Networks (CNNs).

This first development of a customized NN is part of an on-site application project which allows to automate surveys using mixed reality. NNs are exploited for automation of object detection while localization is performed by means of sensors and algorithms embedded in the augmented reality device. On site collected data can be immediately verified through the use MR device that shows information overlapped to real world and the possibility of adding semantic data directly on site avoiding long post processes phases.

2 Scientific Background

There is an increasing need to have structured and semantically enriched "as-is" 3D digital models of buildings in order to handle, more efficiently, maintenance, restoration, conservation or modification. Especially, as far as existing buildings are concerned, it is necessary to develop an efficient approach to generate a semantically enriched digital model. Various digital tools for building capture and auditing are available, such as 2D/3D geometrical drawings, tachometry, laser scanning or photogrammetry, but they need increased modelling and planning efforts of skilful personnel. Approaches that

process point clouds are beginning to appear for the semi-automatic identification of objects [3][9].

There are also systems like the one by [10] that exploit images rather than point clouds. However, these methods consist of complex processing operations that not only require long processing time and therefore high costs, but they are also pursued not on site with major difficulties in comparing collected data and real conditions. This leads to an error prone process and difficulties in the interpretation of gathered data.

Machine learning techniques have been widely applied in a variety of areas such as pattern recognition, natural language processing and computational learning [11]. Particularly, the field of image recognition, and object detection especially, has seen an increase in development in the recent years. In the automotive industry, for example, the use of deep learning algorithms has allowed self-driving cars to recognize lanes and obstacles without the need for more expensive and complex tools [12]. Object detection is a problem of importance in computer vision. Similarly to image classification tasks, deeper networks have shown better performance in detection. At present, the accuracy of these techniques is excellent. Hence, they are used in many, diverse applications, touching all engineering fields [13]. [14] studied an application of machine learning for the construction industry to categorize images of building designs. [15] proposed a similar approach towards the recognition of 3D BIM environments. [16] used NN for the automatic recognition of house spaces. Some interesting applications of ML regard diagnostic issues such as in [17] where semantic segmentation networks are used to recognize wall cracks on both stone and plastered walls. [18] proposed a system for the automatic detection of formworks in construction site images acquired with a Unmanned Aerial Vehicle (UAV).

Despite their huge potential NNs have not been wholly exploited in the AECO sector, partly due to the challenge of developing specific domain labelled datasets. In this paper the use of YOLO CNN for is considered along with the creation of a dataset for fire protection system components. In particular, an investigation is reported on the sufficient number of images for YOLO customization. Two objects classes have been implemented: fire extinguishers and emergency signs. This research work starts from the retrieval of specific images for the creation of multiple datasets with different number of pictures, then the setup of the whole training system has been specified. Finally, the trainings results are exposed and commented.

3 Methods

3.1 Training environment settings

In order to train the network, it is necessary to have a training environment. Since this project involves the use of YOLO neural network it has been decided to use the training platform advised by the developer of the network itself: Darknet-19 [13]. Darknet is an open source neural network framework written in C and CUDA. It supports CPU and GPU computation [8]. In order to install Darknet it is necessary to set the proper environment. The following ones are all the necessary requirements [7][19]:

- Windows or Linux;
- CMake ≥ 3.8 for modern CUDA support;
- CUDA;
- OpenCV ≥ 2.4 ;
- cuDNN ≥ 7.0 ;
- GPU with CC ≥ 3.0 ;
- on Linux GCC or Clang, on Windows MSVC 2015/2017/2019.

The development system is Visual Studio installed with its default options. The dependencies are CUDA, cuDNN and OpenCV. Starting from CUDA, the version installed is the 9.1. This installation requires also the installation of the NVIDIA Graphics Drivers if not yet on the pc. The second installation to be done is cuDNN version 7.0. Finally, it follows the installation of OpenCV 3.4.0. After having done all these installations, Darknet needs to be compiled with the following procedure:

1. Start Microsoft Visual Studio
2. Open the darknet.sln
3. set x64 and Release
4. Include cudnn.lib in your Visual Studio project
5. Build > Build darknet.

At this moment the darknet.exe is generated inside the folder. Finally, darknet needs to be prepared for using OpenCV, CUDA and cuDNN. The bin file has to be placed in the same folder of darknet.exe. Bin and include folders have to be inserted also in CUDA folder if they are not already there. Finally, a new Windows variable cudnn has to be created.

3.2 Dataset creation

The dataset to train the network to recognize a specific object must have specific features. Shape of the object, lighting conditions and varying viewpoints are aspects to take into consideration when gathering the images. According to the COCO dataset approach, choosing images with the object in context improves the recognition of it in real scenarios [20]. The presence of multiple objects in the same pictures is another parameter that improves the

performance of the network. For this reason, pictures with the objects in their common context have been preferred in this research.

The network capability of recognizing the object at first sight and with a high level of confidence depends, among other factors, upon the quantity of pictures in the dataset, although a minimum number is not suggested in the literature.

Collecting original images is always a time consuming task. Pictures can be gathered using various methods, including:

1. crowdsourcing;
2. web scraping.

Generally speaking, crowdsourcing regards all the possible processes to obtain information or input into a particular task or project by enlisting the services of a large number of people, either paid or unpaid, typically via the Internet. Unlike datasets shot in controlled environments crowdsourcing brings in diversity which is essential for generalization [21]. In this case, the task was not conducted using internet sources, but was spread among people in the department and acquaintances outside the university. In the case of web scraping three popular sites have been exploited: 1. Google; 2. Flickr (already used in other research like [22]); 3. Instagram. In this research, the three sources were exploited. Using keywords, images were manually searched and downloaded from Flickr and Instagram. For Google, a python script has been used for web scraping images through keywords as well [23]. To further increase the size of the dataset at low cost, a common technique consists in the usage of both original pictures, from real buildings in this case, and graphically re-edited photos; a process called Data Augmentation. Data augmentation introduces additional variety during training, producing robustness in the model to various inputs. The percentage of original images and modified images has been studied in order to obtain a trained network with good performances [24]. In this work the augmentation involves the modification of pictures with the help of a custom MatLab script, to automatically modify the photos, choosing what transformations must be performed, the starting dataset and the number of images to be created. The custom script applies the following transformations: resize, shifting, additive noise, rotate, zoom, crop. The creation of the dataset involves also labelling all the images. Creating the label involves both the design of the bounding box around the object to recognize and attaching the correct label to it. This operation is performed in this thesis using VoTT, a tool that supports the manual drawing of the bounding box [25]. This tool gives the possibility to choose the right output format according to the kind of network chosen. The output for the YOLO network is a .txt file, with the coordinates of the boxes and the label

attached to them. The final task to complete the dataset is the definition of the training and testing sets of images. We allocate 80% of the images to training, and 20% to testing. Anyway in this study the same testing dataset has been used so as to can better compare the data. Specifications about the testing process can be find in Section 5.

3.3 Training process settings

After having created the dataset there are some files to set before starting the training: the .cfg file of the network chosen; the .data file; the .names file; the .weights file. The parameters to customize in the cfg file are: batch = 64, this means we will be using 64 images for every training step; subdivision = 8, the batch will be divided by 8 to decrease GPU VRAM requirements. If one has a powerful GPU with loads of VRAM, this number can be decreased, or batch could be increased. The other parameters to change are classes = 1, the number of categories we want to detect; filters = (classes + 5)*5 [19][26].

The .data contains all the paths to the other necessary files for the training process. The .names file is the file that contains the name of the tag inside the images of the dataset.

The weights have to been chosen according to the network that one wants to train. Choosing the weight file means that the network used is a pre-trained network with a general dataset (CoCo, PascalVoc or others). It would be possible also not to choose any weights file and in that case the network will be trained from scratch and it will not profit from transfer learning, which is valuable particularly for low level feature learning.

The output of the training process is:

- the chart with the Mean Average Precision (mAP) progress and the Average loss progress;
- the log file with all the operations executed to train the network. It contains the report of all the epoch, with the avg and mAP values;
- the backup folder which contains the weights of the trained network saved at predefined stages.

The question “when the average loss is low enough?” does not find its answer in existing literature. As a rule of thumb, according to what is stated in other customizing network processes, when the first decimal digit reaches 0 it is low enough (e.g. 0.02).

3.4 Validation metrics

To compute the mAP, the Precision and Recall are required. Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. Recall is the ratio of correctly predicted

positive observations to all the observations in actual class. (Equation 2) [27][28][29][30][22].

$$Precision = TP/(TP + FP) \quad (1)$$

$$Recall = TP/(TP + FN) \quad (2)$$



Figure 1. From left to right: emergency sign door, emergency sign man, emergency sign.

True Positives (TP) and False Positives (FP) are the number of objects correctly and incorrectly predicted, respectively, as the object of interest. Similarly, True Negatives (TN) and False Negatives (FN) are the number of objects correctly and incorrectly recognized as background. Because the precision and recall rates cannot be reported for scenes without any actual positives, the images taken into consideration contain at least one instance of the objects of interest [28].

For each class, a Precision/Recall curve is obtained by varying the threshold parameter from 0 to 1. The average precision is defined as the area under the curve. The mAP is computed by averaging the AP value for all classes. This process is applied to obtain the AP for each class [24]. Besides the calculation of Precision and Recall also the F1 parameter has been measured. This metrics is frequently used in pattern recognition performance assessment [30]. F1-measure is a measure that combines Precision and Recall, using a sort of weighted average (Equation 3).

$$F1 = 2 * ((Precision * Recall)/(Precision + Recall)) \quad (3)$$

4 YOLO trainings

Training a customized neural network starts from the creation of the dataset which has been achieved through the method explain in 3.2. In this research two objects have been introduced into the datasets: fire extinguisher and emergency signal. Among the pictures referring to emergency signals there was a distinction between different types 1: Emergency sign; Emergency sign door and Emergency sign man. In 1 the original pictures have been reported.

The network chosen for the training sessions has been the tiny YOLOv2. This choice depends upon the further uses of the customized network, which has to be compatible with other components. The chosen network came from a training with CoCo dataset and thus pre-trained weights have been used. In 2 all the training sessions for fire extinguisher category have been detailed. In this table, TRAINING 2 uses the same dataset as TRAINING 1 but a network that is not pre-trained. 3 reports the list of all the training session for emergency signs.

Finally, a training (TRAINING 17) with a combined dataset has been done using 500 original pictures of fire extinguishers and 581 images of emergency signs. In this case, the mAP = 71,98%.

5 Testing the networks

The testing process of all the trainings has been pursued through the calculation of the metrics exposed in 3.4. This process has been done using the same test dataset, composed by 100 original pictures, so as to make the tests comparable. 2 shows the output of the tests and the evaluation about the result for the calculation of precision and recall. The third column shows the confidence score for every single object detected marked by its bounding box. At test time YOLO defines the confidence score as $P(\text{object}) * (\text{intersection}/\text{union})$ between the predicted box and the ground truth which provides class-specific confidence scores for each box. These The confidence score threshold for the testing process has been set equal to 60%.

Since the trainings produced a new weights file every 1000 iterations for every test it has been selected the weights closer to the number of iteration that had obtained the higher mAP. The test has been performed for the most relevant networks as shown in 4.

On the other hand, Figure 4 displays the values of precision, recall and F1 in a graph. Following these calculations it is possible to express some observations:

- it can be seen that all the F1 values are acceptable since they are higher than 80%;
- a high percentage of image augmentation deeply worsen the performances of the network, as can be seen in TRAINING 13;
- moderate percentage of image augmentation are still acceptable as suggested by TRAINING 15.

Moreover, from these tests it is evident that the higher the number of pictures the better the performances is not true, with or without augmentation.

Table 1. Dataset composed by original images.

Object	Dataset	Number and origin of images
FIRE EXTINGUISHERS	Dataset 1	175 from Polytechnic University of Marche and Flickr
	Dataset 2	118 original images from University of Edinburgh
	Dataset 3	286 images from Google
	Dataset 4	127 images from Google
	Dataset 5	270 images from Google
	Dataset 6	24 images from Instagram
EMERGENCY SIGNS	Dataset 7	45 images from Polytechnic University of Marche
	Dataset 8	536 images from Google

Table 2. Fire extinguishers training sessions.

Training	Dataset	mAP
TRAINING 1	1000 original pictures using all fire extinguisher images	89%
TRAINING 2	1000 original picture using all fire extinguisher images	90,80%
TRAINING 3	100 original pictures taken from Dataset 1	89,11%
TRAINING 4	200 original pictures taken from Dataset 1, 6 and 2	92,86%
TRAINING 5	300 original pictures taken from Dataset 1, 6 and 2	98,91%
TRAINING 6	400 original pictures taken from Dataset 1, 6, 2 and 3	99,30%
TRAINING 7	500 original pictures taken from Dataset 1, 6, 2 and 3	73,43%
TRAINING 8	600 original pictures taken from Dataset 1, 6, 2 and 3	82,73%
TRAINING 9	700 original pictures taken from Dataset 1, 6, 2, 3 and 4	82,73%
TRAINING 10	800 original pictures taken from Dataset 1, 6, 2, 3, 4 and 5	76,68%
TRAINING 11	900 original pictures taken from Dataset 1, 6, 2, 3, 4 and 5	78,13%
TRAINING 12	175 original pictures from Dataset 1 and 175 pictures coming from augmentation	94,95%
TRAINING 13	4000 pictures, only 175 original	16,43%

Table 3. Emergency signs training sessions.

Training	Dataset	mAP
TRAINING 14	1373 pictures, 539 original and 834 from re-edited photos	86,40%
TRAINING 15	581 original photos	80,98%.
TRAINING 16	310 original photos, 155 original and 155 from augmentation	84,46%.

Table 4. Training testing results.

Training	Precision	Recall	F1
TRAINING 1	97,83%	97,83%	97,83%
TRAINING 2	97,08%	96,03%	96,55%
TRAINING 3	89,51%	98,56%	93,81%
TRAINING 6	97,08%	96,03%	96,55%
TRAINING 7	97,09%	96,39%	96,74%
TRAINING 14	86,57%	93,98%	90,12%
TRAINING 15	84,80%	91,77%	88,15%
TRAINING 16	81,97%	89,29%	85,47%
TRAINING 17	83,19%	89,31%	86,14%

With the aim of testing the network in real world conditions it has been performed a test inside the Polytechnic University of Marche premises. The customized network was the one able to detect fire extinguishers only. The chosen building is a three-floor construction and the fire extinguisher were 32, 15 at the ground floor, 11 at the first floor and 6 at the basement floor. Performances with real and unfortunate light condition have been test in this case, especially using the Hololens for taking the pictures while in the aforementioned tests the pictures had been taken with phone camera. This real world test has been done

with an embedded system composed by the Hololens for taking the pictures, a raspberry and a neural compute stick to run the network locally. In this case the network has been always able to recognize the object although with different level of confidence. Less than 10% of the object have obtained a value of level of confidence lower than 60%. In 9 cases the recognition did not worked at the first attempt. This was due not to light condition but to the chosen point of view. When the white label usually on top of fire extinguisher was not visible the network struggled in recognize the object at the first attempt.

- [5] S.D. Kulik and A.N. Shtanko. Experiments with neural net object detection system yolo on small training datasets for intelligent robotics. In *Advanced Technologies in Robotics and Intelligent Systems*, pages 157–162. Springer, 2020.
- [6] J. Redmon, R. Girshick, A. Farhadi, and A. Dataset. You Only Look Once : Unified , Real-Time Object Detection. 2016.
- [7] J. Redmon and A. Farhadi. YOLO9000: Better, faster, stronger. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-Janua, pages 6517–6525, 2017. ISBN 9781538604571. doi:10.1109/CVPR.2017.690.
- [8] J. Redmon and A. Farhadi. YOLOv3: An Incremental Improvement. Accessed: 09/06/2020. URL <http://arxiv.org/abs/1804.02767>.
- [9] E. Valero, F. Bosché, and A. Forster. Automation in Construction Automatic segmentation of 3D point clouds of rubble masonry walls , and its application to building surveying , repair and maintenance. *Automation in Construction*, 96(August):29–39, 2018. ISSN 0926-5805. doi:10.1016/j.autcon.2018.08.018. URL <https://doi.org/10.1016/j.autcon.2018.08.018>.
- [10] Q. Lu, S. Lee, and L. Chen. Image-driven fuzzy-based system to construct as-is IFC BIM objects. *Automation in Construction*, 92(March):68–87, 2018. ISSN 09265805. doi:10.1016/j.autcon.2018.03.034. URL <http://linkinghub.elsevier.com/retrieve/pii/S0926580517306118>.
- [11] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi. A survey of deep neural network architectures and their applications. *Neurocomputing*, 234 (December 2016):11–26, 2017. ISSN 18728286. doi:10.1016/j.neucom.2016.12.038. URL <http://dx.doi.org/10.1016/j.neucom.2016.12.038>.
- [12] B. Huval, T. Wang, S. Tandon, J. Kiske, W. Song, J. Pazhayampallil, M. Andriluka, P. Rajpurkar, T. Migimatsu, R. Cheng-Yue, F. Mujica, A. Coates, and A. Y. Ng. An Empirical Evaluation of Deep Learning on Highway Driving. pages 1–7, 2015. URL <http://arxiv.org/abs/1504.01716>.
- [13] S. Shinde, A. Kothari, and V. Gupta. YOLO based Human Action Recognition and Localization. *Procedia Computer Science*, 133 (2018):831–838, 2018. ISSN 18770509. doi:10.1016/j.procs.2018.07.112. URL <https://doi.org/10.1016/j.procs.2018.07.112>.
- [14] F. Lamio, R. Farinha, M. Laasonen, and H. Huttunen. Classification of Building Information Model (BIM) Structures with Deep Learning. *Proceedings - European Workshop on Visual Information Processing, EUVIP*, 2018-November, 2019. ISSN 24718963. doi:10.1109/EUVIP.2018.8611701.
- [15] Z. K. Zhao, L. Wang, and N. Xu. Deep belief network based 3D models classification in building information modeling. *International Journal of Online Engineering*, 11(5):57–63, 2015. ISSN 18612121. doi:10.3991/ijoe.v11i5.4953.
- [16] T. Bloch and R. Sacks. Comparing machine learning and rule-based inferencing for semantic enrichment of BIM models. *Automation in Construction*, 91(July 2017):256–272, 2018. ISSN 09265805. doi:10.1016/j.autcon.2018.03.018. URL <https://doi.org/10.1016/j.autcon.2018.03.018>.
- [17] E. Cosenza, A. Salzano, C. Menna, D. Asprone, and M. Serra. Digitalizzazione del danno sismico di edifici su piattaforma BIM attraverso tecniche di intelligenza artificiale. *Ingenio*, 2:1–17, 2018.
- [18] A. Braun, K. Jahr, and A. Borrmann. Formwork detection in UAV pictures of construction sites. *eWork and eBusiness in Architecture, Engineering and Construction*, pages 265–271, 2019. doi:10.1201/9780429506215-33.
- [19] AlexeyAB. <https://github.com/AlexeyAB/darknet>, Accessed: 09/06/2020.
- [20] T. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft COCO: Common Objects in Context. pages 1–15, may 2014. URL <http://arxiv.org/abs/1405.0312>.
- [21] I. Laptev and A. Gupta. Hollywood in Homes : Crowdsourcing Data. 1:510–526, 2016. doi:10.1007/978-3-319-46448-0.
- [22] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, Jan 2015. ISSN 1573-1405. doi:10.1007/s11263-014-0733-5. URL <https://doi.org/10.1007/s11263-014-0733-5>.
- [23] Github. Darknet: yolov3workflow. https://github.com/reigngt09/yolov3workflow/tree/master/1_WebImage_Scraping, Accessed: 09/06/2020.

- [24] D. M. Montserrat, Q. Lin, J. Allebach, and E. J. Delp. Training object detection and recognition CNN models using data augmentation. *IS and T International Symposium on Electronic Imaging Science and Technology*, pages 27–36, 2017. ISSN 24701173. doi:10.2352/ISSN.2470-1173.2017.10.IMAWM-163.
- [25] Github VoTT. Labelling tool. <https://github.com/microsoft/VoTT>, Accessed: 09/06/2020.
- [26] Nils T. Customizing yolo. <https://timebutt.github.io/static/how-to-train-yolov2-to-detect-custom-objects/>, Accessed: 09/06/2020.
- [27] G. Li, Z. Song, and Q. Fu. A new method of image detection for small datasets under the framework of yolo network. In *2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pages 1031–1035, 2018.
- [28] H. Hamledari, B. McCabe, and S. Davari. Automated computer vision-based detection of components of under-construction indoor partitions. *Automation in Construction*, 74:78–94, 2017. ISSN 0926-5805. doi:10.1016/j.autcon.2016.11.009. URL <http://dx.doi.org/10.1016/j.autcon.2016.11.009>.
- [29] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Feifei. ImageNet : A Large-Scale Hierarchical Image Database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2–9, 2009. URL [10.1109/CVPR.2009.5206848](http://dx.doi.org/10.1109/CVPR.2009.5206848).
- [30] B. Quintana, S. A. Prieto, A. Adán, and F. Bosché. Door detection in 3D coloured point clouds of indoor environments. *Automation in Construction*, 85 (October 2016):146–166, 2018. ISSN 0926-5805. doi:10.1016/j.autcon.2017.10.016. URL <https://doi.org/10.1016/j.autcon.2017.10.016>.