Support system dedicated to monitoring the safety of workers on a construction site

Andrei-Cosmin MARIA

Resources: https://github.com/cosminacho/Helmet.AI

1. Introduction

Occupational health and safety is the field that deals with the regulation of dangerous activities that can endanger the product of work, and that can cause injury, illness, or even death of the employee. An important first step in preventing or mitigating the impact of accidents at work is the wearing of personal protective equipment (PPE). In my bachelor thesis, I present how computer vision can be used to solve this problem by creating a surveillance and assistance system dedicated to monitoring workers' activities on construction sites.

The main objective of the project is to identify in real-time some mandatory safety elements in workers' clothing, such as helmets or reflective vests. The secondary objective is the integration of the detection model with an object tracking algorithm (for better speed and flexibility) and with a face recognition application. In the future, I plan to test and install the system created within an NGO I am collaborating with.

The bachelor thesis combines both elements of the theory, necessary to understand how the algorithms work, and the practical part, with the explanation of the steps I followed in the implementation process. In this summary, I will focus more on the results obtained within the project and on some interesting aspects which are unique to the problem addressed that are different from the rest of the identification problems.

2. Methods

Object detection involves solving two computer vision problems at the same time: the classification; what object is the image, and the localization; where is the object positioned in the picture. Moreover, classification and localization must take place for all those objects of interest. A simple yet intuitive approach is the sliding window method that uses a classifier to scan different patches of an image in sequential order and at different scales. In order to increase the speed of this algorithm, various improvement strategies have been developed called two-stage detectors that propose regions with a high probability of representing an object, and then classify them. Two-stage detectors have the advantage of high accuracy but with the impediment of a low prediction speed.

The method I chose to use is called YOLO (You Only Look Once), which is a one-stage object detector that looks at the whole image at once and thus makes all the predictions simultaneously. This approach is possible due to the properties of the convolution operations in neural networks to learn features (such as lines, curves, textures, and shapes) and to share the knowledge across different regions of the image. YOLO divides the image into a grid shape and tries to assign for each box created the center of the corresponding object. If several objects occupy the same cell, YOLO uses a technique based on anchor boxes to distinguish objects by their shape and orientation. Finally, in a post-processing step, YOLO uses a Non-Max Suppression (NMS) algorithm to eliminate duplicate predictions based on some confidence and overlapping (IoU) thresholds.

I chose to use the YOLO detection algorithm because it's fast and has an accuracy very close to that of two-stage methods. According to the lead author of the latest version of the model, "YOLOv4 is the most accurate real-time object detector on the MS COCO dataset". The algorithm identifies small objects well because of the grid division strategy and the architecture that allows scanning the image at three different scales, which is an advantage on construction sites where the surveillance cameras will be placed somewhere far from the working area. It's robust, popular, well-documented, open-source, continuously updated, and in the context of my thesis subject, the classes I identify are easily recognizable due to their shape and color (yellow for helmets and green for vests).

3. Datasets

Data is the fuel of machine learning algorithms and the reason why this field has developed so much in recent years. For the PPE detection problem, I was unable to find a large enough dataset accessible on the internet. Fortunately, a group of researchers has already thought about the issue of a lack of unlabeled data for the detection problems and demonstrated in their paper how video games can be used to generate virtual datasets. Therefore, the datasets I used in the project were:

- 132 Gb in PNG format of virtually generated pictures in the game Grand Theft Auto V. Those are split into 126900 images for training and 13500 for validation. To make them easier to handle, I've reduced their volume using resizing operations and converted them to the JPG format.

Figure 3.1 Virtual dataset

- 220 real images provided by the same authors and used for fine-tuning and testing. The authors demonstrated in their paper that training on virtual data and fine-tuning on real data leads to good results.

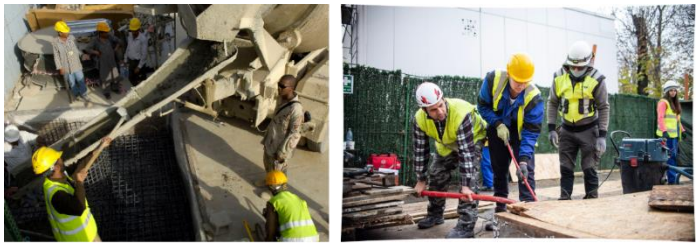

Figure 3.2 Real datasets

- 1850 images from a construction site in Romania that I labeled and used for fine-tuning and testing.
- a dataset containing only labels for helmets and their color that I used to train a smaller network, YOLOv3-tiny.

4. Problems encountered

Even though the images were generated and labeled automatically with a script, I identified two types of problems that could affect the performance of the detector.

The first problem is that of the almost identical images, as we can see in Figure 4.1. In this particular case, there are another 800 identical images with those in the figure in which the foreground character is locked in the same position. This is a problem that can lead to the phenomenon of overfitting, i.e. the model is over-achieving on the training data and no longer generalizes on the validation and test data. To solve this issue, I used classical methods in data augmentation, such as changing randomly the values of the parameters hue, saturation, exposure, and adding Gaussian noise or blur effect. In version 4 of the YOLO algorithm, it's also used a geometric augmentation technique called mosaic, which combines 4 different images into one.

Figure 4.1 Almost identical images

The second problem I encountered is related to overlapping objects. As we can see in Figure 4.2, the objects in the background are covered by those in the foreground. This is a problem as it can introduce False-Positive errors, i.e. objects that are labeled and should not. To solve this issue I created a script that based on an overlapping index (area of intersection over the area of the target object) and the area of the objects (those in the background having a smaller surface), it manages to eliminate all the overlapped objects as in the figure.

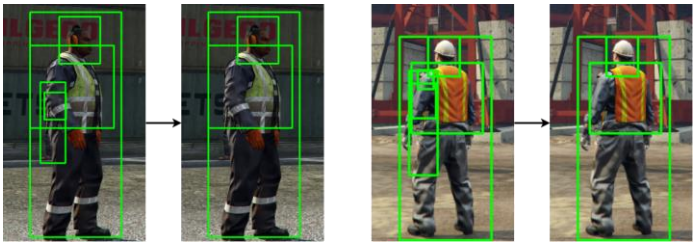

Figure 4.2 Overlapped objects

5. Results obtained and conclusions

For this project, I trained 6 YOLOv3 models, 4 YOLOv4 models, and one YOLOv3-tiny model in two different frameworks (Darknet and Tensorflow) summing up to a total of 1000 hours of training and 200 hours of testing and coding. In all the cases, I used the recommended hyper-parameters for the optimization algorithm and the Transfer Learning paradigm. I used the pre-trained weights on the MS COCO dataset and froze the first 52 convolutional layers out of 75 for the YOLOv3 model, and the first 72 convolutional layers out of 110 for the YOLOv4 architecture. Instead, I varied the parameters related to data augmentation and the size of the image in the input layer. The best results were obtained by training a YOLOv4 model with a 512x512 input size (not preserving the aspect ratio), 25% blur and noise, mosaic data augmentation, default anchor boxes, and the default random HSV values.

In Table 5.1, we can see a detailed comparison between the authors' model and mine. We can also observe that our hypothesis is confirmed; the model indeed better identifies the helmets and vests. It also has a good score on identifying persons due to having used transfer learning from a model already trained on a dataset that contains this class. In the end, the YOLOv4 model achieves a 5.9% better mAP and beats the authors' model with a two-on-one score. In conclusion, we can confirm that training on a virtual dataset and fine-tuning on real data gives good results, and by using state-of-the-art methods, we can achieve great models that are worth implementing in a production system.

For further practical results and the full thesis, please check the resources link above.

Table 5.1 Summary of the results[1]

Id	Train	Test	Head	Helmet	Mask	Headset	Chest	Vest	Person	mAP
Y3A	V	V	89.7%	86.7%	75.5%	89.0%	89.7%	90.0%	89.7%	87.2%
Y4*	V	V	96.6%	97.1%	86.1%	98.7%	96.5%	98.0%	95.0%	**95.4%**
Y3A	R	R	44.1%	52.2%	42.3%	62.0%	59.1%	60.7%	80.6%	**57.3%**
Y3A	V	R	36.3%	74.1%	27.3%	55.6%	45.7%	69.9%	76.9%	**55.1%**
Y4*	V	R	42.5%	58.2%	16.1%	69.9%	36.8%	60.3%	74.6%	51.2%
Y3A	VR	R	78.8%	73.3%	66.3%	74.0%	74.7%	78.6%	87.1%	76.1%
Y4*	VR	R	80.3%	82.5%	63.4%	92.5%	82.3%	84.0%	89.3%	**82.0%**

[1] Y3A: YOLOv3 authors' model; Y4*: my YOLOv4 model; V/R: trained/tested on virtual/real data; VR: train V + fine-tune R.