applied sciences

MDPI

Article

Augmented Reality Maintenance Assistant Using YOLOv5

Ana Malta [1,*], Mateus Mendes [1,2,*] and Torres Farinha [1,3]

[1] Polytechnic of Coimbra, ISEC, 3045-093 Coimbra, Portugal; tfarinha@isec.pt
[2] ISR, University of Coimbra, 3004-531 Coimbra, Portugal
[3] Centre for Mechanical Engineering, Materials and Processes—CEMMPRE, 3030-788 Coimbra, Portugal
* Correspondence: a21230211@isec.pt (A.M.); mmendes@isr.uc.pt (M.M.)

Abstract: Maintenance professionals and other technical staff regularly need to learn to identify new parts in car engines and other equipment. The present work proposes a model of a task assistant based on a deep learning neural network. A YOLOv5 network is used for recognizing some of the constituent parts of an automobile. A dataset of car engine images was created and eight car parts were marked in the images. Then, the neural network was trained to detect each part. The results show that YOLOv5s is able to successfully detect the parts in real time video streams, with high accuracy, thus being useful as an aid to train professionals learning to deal with new equipment using augmented reality. The architecture of an object recognition system using augmented reality glasses is also designed.

Keywords: task assistant; YOLOv5; car engine dataset; car part detection; augmented reality

Citation: Malta, R.; Farinha, T.; Mendes, M. Augmented Reality Maintenance Assistant Using YOLOv5. *Appl. Sci.* **2021**, *11*, 4758. https://doi.org/10.3390/app11114758

Academic Editors: João Reis, Marlene Amorim and Yuval Cohen

Received: 28 April 2021
Accepted: 20 May 2021
Published: 22 May 2021

1. Introduction

Maintenance technicians often have to perform a large number of different jobs, involving numerous parts, manufactured by numerous different companies. The main objective of the present work is to build a system to recognize mechanical parts in car engines, and give directions, which come from a work order, to the technician. The technician wears augmented reality glasses during the procedure. Through the glasses, he sees the car parts and also the instructions on how to proceed, which are added in real time.

The field of object detection has garnered new attention with recent developments in deep neural network architectures, and it is constantly growing. More and more research is proposed, aiming to solve various problems to make people's lives easier. Many neural network architectures can detect a large number of objects, with high accuracy, in real time. The detection of engine parts of an automobile can bring major developments in the scope of manufacturing and maintenance, especially facilitating correct maintenance tasks, with the objective of maximizing the quality of maintenance procedures, aiming to increase engines' life and reliability.

To be able to build a good task assistant system, it is necessary to implement a good detection method. The present work relies on a YOLOv5 deep neural network, which is one of the fastest and most reliable detectors available nowadays. The YOLOv5 receives images in real time from the technician's augmented reality glasses and detects the parts. The task assistant suggests actions to perform, based on the sequence of actions of the work order and on which objects are detected in the technician's field of view.

To properly train the YOLOv5 neural network, it is essential to have a good dataset. The larger the dataset, the higher the chances of training the network to have good performance. For this first step, the dataset created consisted of 582 images taken from three videos with similar lighting conditions, where it was possible to identify a total of eight different types of parts: oil dipstick; battery; engine oil reservoir; wiper water tank; air filter; brakes fluid reservoir; coolant reservoir; and power steering reservoir. The images taken from each frame are converted to a 416 × 416 format, which is the format that the chosen architecture needs to use as input.

Two versions of YOLOv5 were tried, namely YOLOv5s and YOLOv5m. They are available at https://pytorch.org/hub/ultralytics_yolov5/ (accessed on 21 May 2021). The data were split into training, validation and testing sets and converted to YOLOv5 PyTorch format. Then, data were saved into a data.yaml file, which describes the classes to be used for training, validating and testing the model. This process was done using Roboflow, which is available online at http://www.roboflow.com (accessed on 21 May 2021). Then, using a Google Colab virtual machine, which is available at http://colab.research.google.com (accessed on 21 May 2021), the models were trained, with a total of 250 epochs. Different evaluation metrics were used, to evaluate the quality and reliability of the system. In a second experiment, the dataset size was increased to a total of 900 images with different lighting conditions, in order to optimize object recognition and test the system under more challenging situations.

Section 2 presents a brief review of the state of the art in the context of neural networks, mentioning some examples of the use of YOLOv5 architecture from other studies. Section 3 describes the proposed architecture for the augmented reality system. Sections 4 and 5 describe the materials used and the development of datasets for the intended work. Section 6 describes the YOLOv5 network and the models. The tests and results are described in Section 7 and discussed in Section 8. The conclusions and future work are presented in Section 9.

2. Related Work

2.1. Object Detection with YOLO

Deep learning-based object detection has been a research hotspot in recent years. For better image understanding methods, it is necessary not only to concentrate on classifying different images, but also to try to precisely estimate which objects are present in the images and their locations. This task is referred to as object detection [1], and the state of the art detectors are deep neural networks, namely convolution neural networks (CNN).

ImageNet is a dataset of more than 15 million high-resolution labeled images, which belong to about 22,000 categories. In the article "ImageNet classification with deep convolutional neural networks." [2], two error rates need to be analyzed: top-1 and top-5. The error rate of top-5 is the fraction of test images, where the caption is not one of the five captions considered most likely by the model. In the test data, error rates of 37.5% and 17% were achieved for top-1 and top-5, respectively, which is significantly better than the state of the art. In this context, a convolutional-based network appeared for the first time, which drastically reduced the error rates of the previous state-of-the-art methods. That new network is the now famous AlexNet.

Many modern detectors are usually pre-trained on ImageNet or on COCO datasets. Although the CNN architectures are still recent models, other works on object detection have already been developed using the pre-trained YOLOv5 architecture in the COCO dataset. In the article "Face Mask Detection using YOLOv5 for COVID -19" [3], the model correctly classified both people wearing a mask and people not wearing a mask. The model was tested using both YOLOv5s and YOLOv5x. The training process was completed by running the model through Google Colab. YOLOv5s is significantly better than YOLOv5x in terms of performance and speed. The mean average precision (mAP) is quite similar for both the processes, but when the processing speed is considered, YOLOv5s is slightly superior to YOLOv5x.

Another article purports to train the YOLOv5 architecture to recognize handguns in images. This project uses the pre-trained YOLOv5x model to determine the initial weights from which it started the training. The rest of the configuration settings are mostly preset: 50 epochs, 640 px image size for training including test set and stack size of 64. The model can successfully detect the presence of a handgun in an image, even if the handgun is not ideally oriented, does not have the traditional shape, or contains multiple handguns in the image. The results show 0.80 for precision, 0.89 for recognition, and 0.905 for mAP [4].

2.2. Augmented Reality Applications

Most of the existing augmented reality systems are able to understand the 3D geometry of the surroundings but lack the ability to detect and classify complex objects in the real world. Such capabilities can be enabled with deep convolutional neural networks [5]. The proposed architecture for these works combines deep learning-based object detection with AR technology to provide user-centered task assistance. In the paper [6], proposes two studies, matching a virtual object to a real object. in a real environment, and performing a realistic task, that is, the maintenance and inspection of a 3D printer. These tests are performed using HoloLens from Microsoft and combine object detection and instance segmentation and the results show the advantage, effectiveness, and extensibility to various real applications.

Augmented reality is being used in different industrial fields. Another example is the use of AR in conceptual prototyping processes in product design by overlaying different car interior mock-ups, which are usually only available as 3D models in the initial phases of development [7].

2.3. Task Assistance in Industrial Augmented Reality

Task assistance is one of the main areas of application of AR in the industry. Siam and Tonggoed [8] proposed a human–robot collaboration with augmented reality for virtual assembly task and conclude that the use of augmented reality in the assembly task can save cost and training time.

Augmented reality (AR) is considered to provide user-centric information easily in different environments, thus being able to reduce a worker's cognitive load and increase work efficiency [9]. Augmented reality is slowly gaining ground in industry by embedding visual information onto the real objects directly. In particular, the AR-based visualization of manufacturing information, called industrial AR, can provide more effective task assistance [10,11]. In some areas, industrial workers use AR glasses as an auxiliary tool or as a training simulator to be prepared for specific situations. Bosch's Common Augmented Reality Platform (CAP) is now also available for the new Microsoft HoloLens 2. The interactive training provides holographic step-by-step instructions that help workshop trainees understand new products and technologies and enable service technicians to conduct repair and maintenance tasks more efficiently and with fewer errors [12,13].

Kim et al. [9] propose an approach to industrial AR, which can complement existing AR methods and provide manufacturing information more effectively through the deep learning-based instance segmentation and depth prediction of physical objects in the AR scene. The proposed approach can provide consistent AR visualization regardless of the movement of the physical object. Therefore, manufacturing information can be provided to the worker more intuitively and situation-dependently based on the estimated 3D spatial relation and depth perception.

3. Proposed Architecture

The system proposed manages and processes work orders, which are sequences of procedures that need to be done by the maintenance technicians.

3.1. Workflow

Figure 1 represents, in a flowchart, the sequence of actions that are necessary to perform to go through a typical work order, to perform a planned maintenance task. The use of intelligent task assistants can facilitate the technician's job to go through the procedures, while reducing execution time and the risk of human errors.

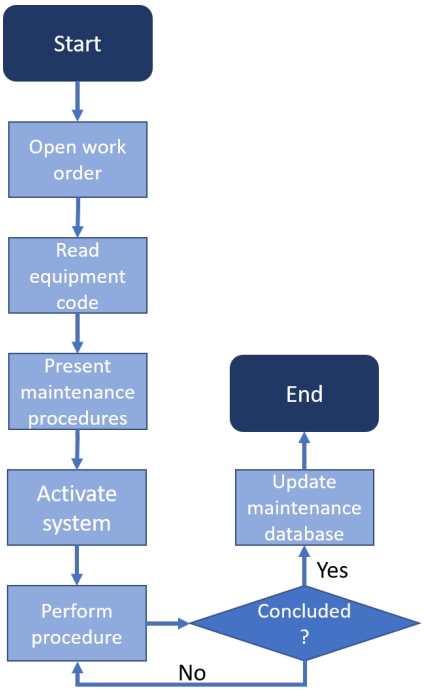

Figure 1. Flowchart showing the sequence of actions necessary to process a given work order, to perform maintenance tasks. The task assistant guides the maintenance technician through the steps shown.

The system proposed aims to guide the technician through all the steps, showing information and directions in the virtual reality glasses and audio messages. The steps illustrated in the flowchart are:

1. Open work order—This is the first step, where the technician opens the work order which was previously assigned to him by a manager. The work order is specific for an equipment, such as a given car or industrial machine.
2. Read equipment code—After opening the work order, the technician must read the equipment code. Each equipment has a unique code, and this step ensures that the technician is at the right equipment. The code can be a barcode, QR code, RF-id or other type of code used in the factory.
3. Present maintenance procedures—Once the equipment code is validated, the technician sees the sequence of procedures required for that particular work order. This gives an overview of the work that needs to be done. The technician can then choose to start working or cancel for some reason.
4. Activate system—After seeing the procedures, the technician must then activate the system to start working and going through each procedure.
5. Perform procedure—Once the system is activated, the task assistant guides the technician through all the procedures in a valid order. The technician is told to search for a particular part, and once that part is in his field of view, the assistant gives directions on how to correctly perform each task necessary. This is repeated for all procedures of the work order.
6. Once the work order is completed, the maintenance database is updated. This step may be done at the end for offline assistants, or in near real time for online assistants.

3.2. System Architecture

Figure 2 shows the architecture proposed for the system. The work orders (WO) are created and managed in a Computerized Maintenance Management System (CMMS), by a team manager. CMMS systems are special applications used to manage maintenance operations. It helps optimize the reliability and availability of physical assets [14]. The

present system adds new potential to execute maintenance interventions, both planned and non-planned. The final result will correspond to the reduction of the maintenance time of each intervention and, globally, to optimize the mean time to repair (MTTR).

For each intervention, the maintenance technician has an interface to the CMMS, where he picks the WO and downloads them to the augmented reality (AR) glasses, which can work offline or online with the CMMS. If they work online, the maintenance WO can be updated at the end of each procedure. If they work offline, the technician is only required to connect them again to the CMMS at the end of the task, to update the WO and send it back to the maintenance manager.

The picture also shows another user, who is the administrator, who has special privileges to manage the CMMS teams and tasks.

Farinha (2018) presents a holistic approach for the maintenance of physical assets, including technological solutions like AR [15].

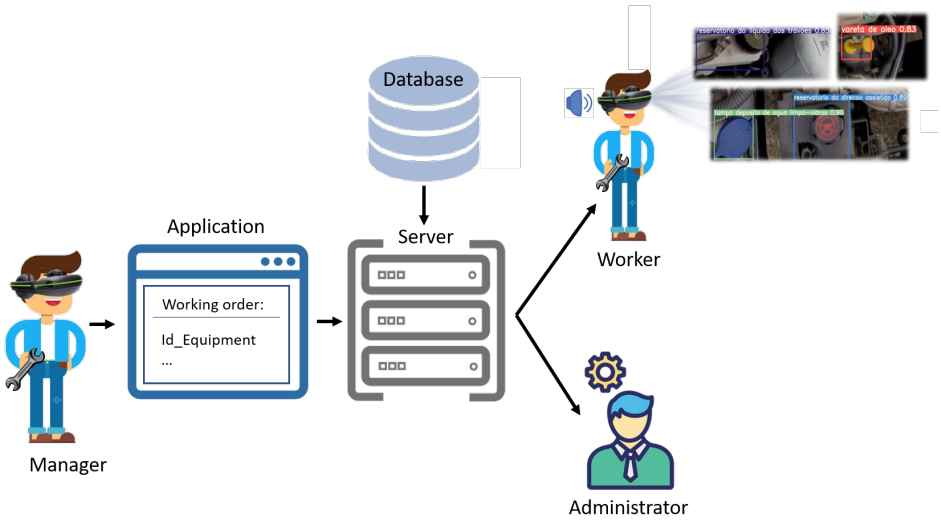

Figure 2. System architecture, showing the main components and interactions. The technician wears Augmented Reality glasses, where he receives directions to go through the procedures of the Working Order, that is created and managed through a CMMS application.

4. Materials and Methods

The complete system requires different types of software and hardware, for development and then for use when it is deployed.

4.1. Hardware Using for Development

The hardware used during development included computers, for running software, and cell phones, for capturing videos and pictures. The object detection model was trained using, laptop computer with access to a Google Colab virtual machine, which offers free GPU cloud service that allows one to obtain 0.007 second inference time. That is, 140 FPS on a TESLA P100 GPU.

To obtain the videos to construct the dataset, two mobile devices were used. The first mobile device was a Samsung Galaxy S10 and the second an IPhone 11. Both are devices with high quality 4K video recording. Table 1 summarizes the characteristics of the mobile devices' cameras.

Table 1. Mobile Devices Camera Characteristics.

Mobile Device	Camera Characteristics
Samsung Galaxy S10	Triple lens camera on the back with a 12 MP regular lens, 12 MP optical zoomed telephoto lens, and a brand new 16 MP ultra wide angle lens.
IPhone 11	Dual 12 MP Ultra Wide and Wide cameras.

4.2. Software Used for Development

The object detection model runs in Python and therefore requires a python interpreter. However, other applications were required for creating the dataset. The list of applications and libraries is as follows.

- **PyTorch**—An open source machine learning library for deep learning, used for applications such as computer vision and natural language processing. This framework was developed by Facebook and it was created to provide models that are easier to write than other frameworks such as TensorFlow [16]. The YOLOv5 architecture is available only for PyTorch at this point.
- **VoTT**—Visual Object Tagging Tool (VoTT) is an open source annotation and labeling tool for image and video assets. It is possible to import data from local or cloud storage providers and to export labeled data to local or cloud storage providers. There is an online version available at https://vott.z22.web.core.windows.net/#/ (accessed on 21 May 2021). VoTT was used to manually tag the images, marking the bounding boxes of each object for the training and testing sets.
- **Roboflow**—Hosts free public computer vision datasets in many popular formats. In the present project, Roboflow was used after VoTT, to tag the images, apply data augmentation and convert the dataset to YOLOv5 PyTorch format.
- **FFmpeg**—Platform to record, convert and stream audio and video. It was necessary to use it to convert the videos recorded through the mobile device to a format that VoTT would recognize. This framework is available at https://www.ffmpeg.org/ (accessed on 21 May 2021).

4.3. Software and Hardware Used for AR System

The proposed augmented reality system requires augmented reality glasses and a server to run the CMMS.

There are currently many augmented reality glasses, but none have been tested at this point. Microsoft HoloLens is a popular platform, which could serve for the proposed task assistant [17]. Another example is Vuzix smart glasses [18].

As for the server, it needs to run the CMMS software to manage the different equipment, maintenance procedures and work orders. The server and the augmented reality glasses can communicate in real time, via wireless network, or they can synchronize periodically according to the technician's needs.

5. Datasets

5.1. Dataset Search

For the present research, a dataset of the mechanical components of an automobile was necessary. However, publicly available datasets that were found were not compatible with the goals intended for the present research. Due to the increasing development of technology in the field of autonomous driving, there are a variety of datasets related to public roads for detecting pedestrians, traffic signs, pedestrian crossings, etc. One dataset widely used in this field is the "KITTI" [19]. KITTI was created mostly for autonomous driving platforms and contains a set of object detection data, including monocular images and object bounding boxes. However, KITTI is not useful for the development of the present project, which aims to detect car motor parts. Datasets of car components were also found, but again, those were not the most relevant components, for they were just

non-mechanical components such as the steering wheel or lighting [20]. After an exhaustive search, it was not possible to find a dataset which would be appropriate for the present research. Therefore, it was deemed necessary to build a custom dataset to proceed with the project.

5.2. Dataset Created

To create a custom dataset, the first step was to shoot three videos of the engine components of a car. The car model chosen was a Peugeot 206, built in 1998. The first videos were recorded with a Samsung Galaxy S10 cell phone. The quality was 4 K at 60 fps. It was necessary to produce multiple videos with different angles and distances, so that the algorithm could more easily identify the desired components. More videos were shot, using another cell phone, namely an iPhone 11. The video characteristics were maintained, to facilitate comparing the results. After the videos were produced using the mobile devices, the videos shot with the Samsung device were in "mp4" format. The videos shot with the iPhone were in "mov" format. This format is not well recognized by VoTT tool, which was used for the part tagging process. So, it was necessary to convert the video through a conversion application, which was "FFmpeg".

Once the videos were ready, they were tagged in VoTT. Eight car parts were chosen as targets for the present project. They are: 1—battery; 2—air filter; 3—power steering reservoir; 4—engine oil reservoir; 5—coolant reservoir; 6—brakes fluid reservoir; 7—water tank of the wiper; 8—oil dipstick. The first dataset had a total of 582 images. The second dataset had additional 318 images, for a total of 900 images, as shown in Table 2. As the table shows, there are a total of 510 targets in the first dataset, 335 targets in the second dataset and a total of 845 targets in the combined dataset.

The part class names were written in Portuguese, which is the official language of the application being developed. Table 3 shows the correspondence of each class name from English to Portuguese, so that in the test images, it is possible to perceive which classes are identified.

Table 2. Characteristics of the Datasets created for training and testing the object detector.

	Mobile Device	Videos	Total Images	Total Labels	Targets
	Samsung Galaxy S10	3	582	8	510
	IPhone 11	3	318	8	335
Total		6	900	8	845

Table 3. Target classes labels translation.

Labels in English	Labels in Portuguese
Battery	Bateria
Air filter	Filtro de ar
Power steering reservoir	Reservatório da direção assistida
Engine oil reservoir	Reservatório de óleo do motor
Coolant reservoir	Reservatório do líquido de arrefecimento
Brakes fluid reservoir	Reservatório do líquido dos travões
Wiper water tank	Reservatório de água dos limpa-vidros
Oil dipstick	Vareta de óleo

6. Object Detection with Deep Learning

6.1. YOLO Deep Neural Network

YOLO (You Only Look Once) is one of the most popular deep convolution neural models for object detection, due to its good performance and short time requirements.

The first model was proposed in 2016, and it is known as YOLOv1. Several updates were made since then, and the latest model is now YOLOv5, released by Glenn Jocher in 2020. To the best of the authors' knowledge, there is no peer-reviewed research paper

proposing the YOLOv5 architecture. Nonetheless, there are already more than 240 research papers referring to the architecture. YOLOv5 is based on the PyTorch framework. It is the latest version of the YOLO object recognition model, developed with the continuous efforts of 58 open source contributors [21]. There are a few model configuration files and different versions of the object detector. The present implementation uses YOLOv5s, which is the smallest model, and YOLOv5m, which is the next model in size. The other models available are YOLOv5l and YOLOv5x, the latter being the largest of all. As the network size increases, its performance may also increase, at the cost of additional processing times [22]. Therefore, the larger models may only be useful for complex problems where large datasets are available.

There are many other deep neural networks that can be used to detect objects. One of them is the mask-RCNN, which aims to solve the instance segmentation problem in machine learning or computer vision. The mask-RCNN is therefore, in theory, more precise, at the cost of additional processing time. In a comparison study, for the task of detecting a sports ball, both YOLO and Mask R-CNN with pre-trained weights show good precision and recall [23].

Because it is a good and faster detector, with high levels of performance [24], it was decided to choose the YOLOv5 network for the present project. Other architectures, such as the mask-RCNN, could provide similar detection performance and more precise location of the objects. However, the YOLO speed is a great advantage for real time operation of the task assistant. Moreover, in the present case, a bounding box is enough to give good directions to the technician.

6.2. Performance Evaluation

To evaluate the performance of an object detector, it is crucial to use appropriate metrics for each problem. Object detection is a very challenging problem because it is necessary to draw a bounding box around each detected object in the image. To evaluate the detection performance, some of the most common metrics are shown in Equations (1)–(3): precision, recall, and mAP.

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

$$\text{mAP} = \frac{1}{N} \sum_{N}^{i=1} \text{AP}i \tag{3}$$

True positive (TP) is a correct detection of an object that actually exists in the picture. False positive (FP) is an incorrect detection of an object, i.e., the network marks an object that is not there in the picture. False negative (FN) is an object that actually exists in the picture but is not detected by the network. In object detection, the intersection over union (IoU) measures the overlap area between the predicted bounding box and the ground truth bounding box of the actual object. Comparing the IoU with a given threshold, detection can be classified as correct or incorrect. Each value of the IoU threshold provides a different average precision (AP) metric, so it is necessary to specify this value.

7. Experiments and Results

The model was trained and tested with the datasets described in Section 5.

7.1. Train the Model

For the first experiment of the proposed system, the smallest and fastest YOLOv5 model was chosen (YOLOv5s). A notebook was adapted for the first tests [25], with all the necessary steps for training and validating the performance of the model recognizing the desired objects. The training procedure consisted of 250 epochs, which took 55 min 12 s for

the dataset. Moreover, 466 of the 582 images were used for training, and 116 were used for validation. The second test used 571 of the 900 images for training and 159 for validation. The training was completed with 250 epochs in 1 h 15 min 7 s.

For the second experiment of the system, the YOLOv5m was used. The test with YOLOv5m and the first dataset was completed with 250 epochs in 55 min 1 s. The test with the second dataset was completed with 250 epochs in 1 h 15 min 39 s.

7.2. Performance for Real Time Operation

Table 4 shows the results of those metrics for all classes, obtained on the first dataset with model YOLOv5s. Table 5 shows the same results for the same dataset but for the second model, YOLOv5m.

The tables show the performance for each of the eight classes and for the whole validation set. The third column shows the number of known targets to be detected. The fourth and fifth columns show the precision and recall of the detector. The sixth and seventh columns show the mean average precision for the IoU specified. As the tables show, YOLOv5s performs about the same as the larger network. So, for the volume of data and complexity of the problem, it is adequate and bigger models are not justified.

Table 4. Performance of the model YOLOv5s for first dataset (582 images).

Class	Images	Targets	Precision	Recall	mAP 0.5	mAP 0.5:0.95
All	116	510	0.98	0.987	0.992	0.813
1	116	81	0.987	0.967	0.993	0.892
2	116	72	0.972	0.977	0.982	0.873
3	116	46	0.968	1	0.995	0.801
4	116	68	0.985	0.956	0.993	0.826
5	116	56	1	1	0.995	0.883
6	116	79	0.986	1	0.99	0.792
7	116	43	0.976	1	0.994	0.777
8	116	65	0.963	1	0.994	0.656

Table 5. Performance of the model YOLOv5m for first dataset (582 images).

Class	Images	Targets	Precision	Recall	mAP 0.5	mAP 0.5:0.95
All	116	510	0.806	0.981	0.993	0.806
1	116	81	0.987	0.966	0.993	0.882
2	116	72	0.971	0.972	0.986	0.867
3	116	46	0.978	0.987	0.995	0.771
4	116	68	0.981	0.956	0.993	0.806
5	116	56	1	0.987	0.995	0.886
6	116	79	0.987	1	0.99	0.811
7	116	43	0.985	1	0.995	0.875
8	116	65	0.985	1	0.995	0.775

7.3. Prediction Examples

After the models were trained and tested, they were also tested against images that had not been used during training and some example results are shown in Figures 3 and 4. The figures show bounding boxes around objects detected. Each bounding box has a label identifying which object is detected inside the bounding box. Each bounding box is also pointed out by a red arrow, where instructions will be added for the technicians. In the figures, all the instructions are just “open here”, for that part of the software is left as future work.

Figure 3. Example of object detection on a test image, where the engine oil reservoir and oil dipstick are detected.

Figure 4. Example of object detection on a test image, where brakes fluid reservoir and battery are detected.

In test mode, the IoU parameter was lowered to 0.3. This means that a detection box is considered valid for $IoU \geq 30\%$. This was decided because to the human eye, it becomes difficult to distinguish between correct predictions considering a threshold of 0.5 and 0.3, according to [26–28]. When considering a lower IoU threshold, it will be possible to view a more significant number of valid detections, avoiding false negatives in the analysis of each image.

7.4. Comparison of Model YOLOv5s with Model YOLOv5m

In terms of speed, the two models are very similar. In terms of accuracy, the YOLOv5m model turned out to be slightly superior.

The two models were tested on both datasets. For the dataset with 900 images, the differences between the models in terms of mAP@0.5, mAP@0.5:0.9, precision and recall are shown in Figure 5. The color blue corresponds to model YOLOv5s and the color orange corresponds to the model YOLOv5m. As the graphs show, YOLOv5m is a bit more stable during the train. Nonetheless, both models converge to the same point.

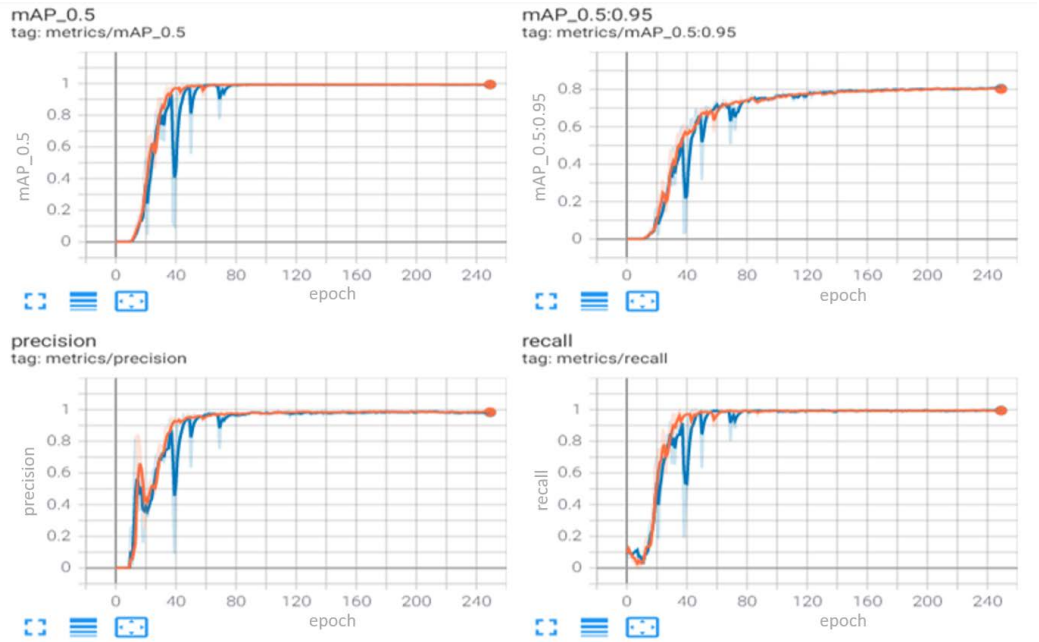

Figure 5. Comparison of performance metrics, during training, for YOLOv5s and YOLOv5m. The performance of YOLOv5s is shown in blue and YOLOv5m is shown in orange.

The plots show that the models are progressively learning through every epoch because the performance is increasing and 250 epochs are enough training, considering that the curves are stable at that point.

The loss function shows the performance of a given predictor in classifying the input data points in a dataset. The smaller the loss, the better the classifier is at modeling the relationship between the input data and the output targets. There are two different types of loss shown in Figure 6. The loss represented at the top is related to both the predicted bounding box and the loss related to the given cell containing an object during the training. The graphs of val Box and val Objectness represent their validation scores. Training loss is measured during each epoch while validation loss is measured after each epoch.

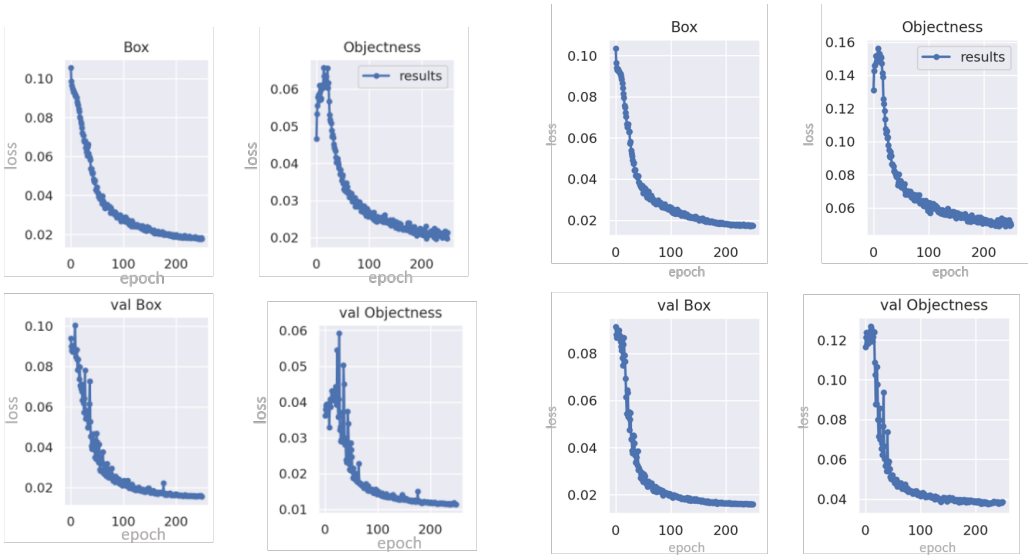

Figure 6. Loss during training, related to both the predicted bounding box and the loss related to the given cell containing an object, as well as their validation scores displayed as Box and Objectness. Results for the YOLOv5s are on the left hand side, for the YOLOv5m at the right hand side.

On the left-hand side, there are those values for the dataset with 582 images with the model YOLOv5s, and on the right are the same values for the model YOLOv5m. The charts show that both models are very similar. The YOLOv5m is faster stabilizing the learning process, but the model YOLOv5s seems to be enough.

Figure 7 shows the loss related to the predicted bounding box, a cell containing an object and the class loss for the two models YOLOv5s and YOLOv5m on the dataset with 900 images. YOLOv5m results are represented in orange and YOLOv5s results are in blue. Again, the results show that the evolution of loss for both models are very similar. Actually, the lines are practically overlapping.

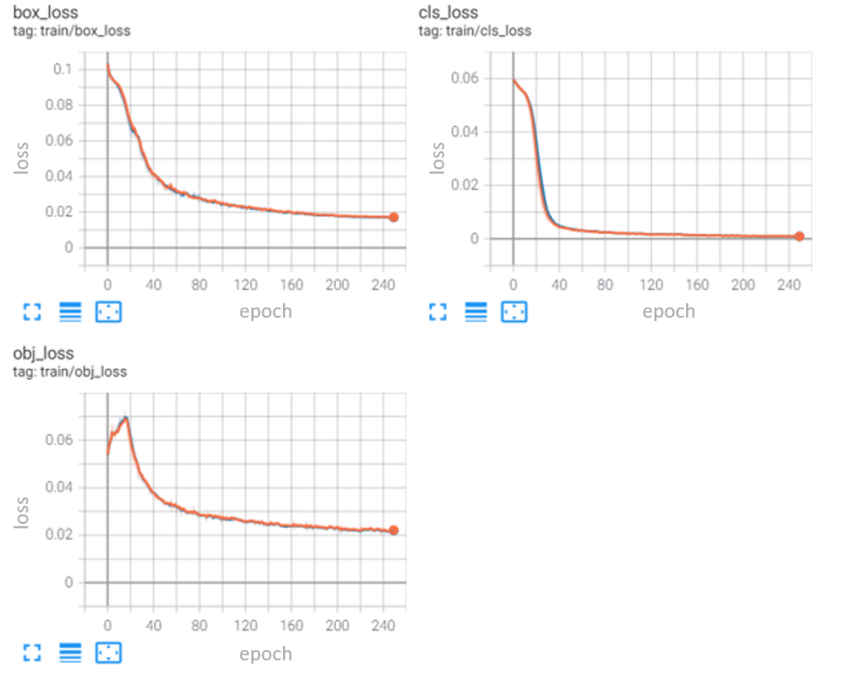

Figure 7. Loss functions for both the models YOLOv5s (blue) and YOLOv5m (orange).

8. Discussion

A task assistant was proposed, using a YOLO deep network for object detection and augmented reality glasses. The model of object detection trained learns quickly and gives a prediction in a fraction of a second, making it suitable for use in real time.

The precision obtained for the two models is in line with that obtained by other authors for similar problems, namely the "Artificial Intelligence for Real Time Threat Detection and Monitoring" [4] that has a precision of 0.803, 0.89 for recall and 0.905 for mAP 0.5. Prediction time is approximately 0.007 seconds, which allows up to 140 FPS on a TESLA P100 GPU. This shows that the system of detection can be integrated with the architecture proposed, to be used in real time, as intended, by maintenance professionals.

Table 6 compares the results for detection for all classes for both models, YOLOv5s and YOLOv5m, on both datasets. The first two lines show the results of training and testing the models with the largest dataset. A total of 571 images were used for training, 159 used for validation and 170 images were used for testing. The last two lines are the results obtained with the smaller dataset, as described in Section 7.1.

Table 6. Tests with YOLOv5s YOLOv5m for both datasets.

Model	Class	Test Dataset	Precision	Recall	mAP 0.5	mAP 0.5:0.95
YOLOv5s	All	dataset 2	0.975	0.992	0.994	0.797
YOLOv5m	All	dataset 2	0.985	0.994	0.994	0.8
YOLOv5s	All	dataset 1	0.969	1	0.992	0.818
YOLOv5m	All	dataset 1	0.974	0.997	0.994	0.829

9. Conclusions and Future Work

The goal of the present work was to train a neural network for deep learning that can serve as a basis for integrating into an augmented reality system that helps professionals in the field of maintenance.

To train the model, two different datasets were created, using two different devices in different illumination conditions. Two versions of YOLOv5 were also tested, and it was determined that YOLOv5s can be sufficient for the intended detection problem.

YOLOv5s demonstrated to be capable of identifying eight different mechanical parts in a car engine with high precision and recall always above 96.8% in the test sets, which, compared to the larger model, has almost the same results. All tests and results prove that the network is good and fast enough to be applied to the proposed system.

Future work includes the integration of the trained model in the CMMS, with the main objective of communicating with the virtual reality glasses, so that the technician is guided through the procedures of the working order in real time.

Author Contributions: Conceptualization, T.F. and M.M.; Methodology, A.M., T.F. and M.M.; software, A.M.; validation, T.F. and M.M.; formal analysis, T.F. and M.M.; investigation, A.M.; writing—original draft preparation, A.M.; writing—review and editing, T.F. and M.M.; visualization, A.M. and M.M.; supervision, T.F. and M.M. All authors have read and agreed to the published version of the manuscript.

Funding: The authors acknowledge Fundação para a Ciência e a Tecnologia (FCT) for the financial support to the project UIDB/00048/2020. FCT had no interference in the development of the research.

Institutional Review Board Statement: Not applicable

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

Ap	Average Precision
AR	Augmented Reality
CAP	Common Augmented Reality Platform
CMMS	Computerized Maintenance Management System
CNN	Convolutional Neural Network
COCO	Common Objects in Context
FFmpeg	Fast Forward MPEG
FN	False Negative
FP	False Positive
FPS	Frames per Second
GPU	Graphics Processing Unit
IoU	Intersection over Union
mAP	Mean Average Precision
MP	Megapixel
MTTP	Mean Time To Repair
RCNN	Region Based Convolutional Neural Networks
TP	True Positive
VoTT	Visual Object Tagging Tool
WO	Worker Order
YOLO	You Only Look Once

References

1. Zhao, Z.Q.; Zheng, P.; Xu, S.T.; Wu, X. Object detection with deep learning: A review. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 3212–3232. [CrossRef] [PubMed]
2. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]

3. Sharma, V. Face Mask Detection Using YOLOv5 for COVID-19. Mater's Thesis, Master of Science in Computer Science, California State University San Marcos, San Marcos, CA, USA, 2020.
4. Tolbert, S.W. Artificial Intelligence for Real Time Threat Detection and Monitoring. Available online: https://stevenwtolbert.com/pdf/threat_detection.pdf (accessed on 15 March 2021)
5. Liu, L.; Li, H.; Gruteser, M. Edge assisted real-time object detection for mobile augmented reality. In Proceedings of the 25th Annual International Conference on Mobile Computing and Networking, Los Cabos, Mexico, 21–25 October 2019; pp. 1–16.
6. Park, K.B.; Kim, M.; Choi, S.H.; Lee, J.Y. Deep learning-based smart task assistance in wearable augmented reality. *Robot. Comput. Integr. Manuf.* **2020**, *63*, 101887. [CrossRef]
7. Nee, A.Y.; Ong, S.K. Virtual and augmented reality applications in manufacturing. *IFAC Proc. Vol.* **2013**, *46*, 15–26. [CrossRef]
8. Charoenseang, S.; Tonggoed, T. Human–robot collaboration with augmented reality. In *International Conference on Human-Computer Interaction*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 93–97.
9. Kim, M.; Choi, S.H.; Park, K.B.; Lee, J.Y. A Hybrid Approach to Industrial Augmented Reality Using Deep Learning-Based Facility Segmentation and Depth Prediction. *Sensors* **2021**, *21*, 307. [CrossRef]
10. Kim, M.; Choi, S.H.; Park, K.B.; Lee, J.Y. User Interactions for Augmented Reality Smart Glasses: A comparative evaluation of visual contexts and interaction gestures. *Appl. Sci.* **2019**, *9*, 3171. [CrossRef]
11. Choi, S.H.; Kim, M.; Lee, J.Y. Situation-dependent remote AR collaborations: Image-based collaboration using a 3D perspective map and live video-based collaboration with a synchronized VR mode. *Comput. Ind.* **2018**, *101*, 51–66. [CrossRef]
12. Bosh. Bosch Connected Devices and Solutions GmbH—Interplay of Augmented Reality and IoT. Available online: https://www.bosch-connectivity.com/newsroom/blog/interplay-of-augmented-reality-and-iot/. (accessed on 15 April 2021).
13. Microsoft. Mixed Reality in der Automobilindustrie Oder Wie Bosch Mobilität Digitaler Gestaltet. 2019. Available online: https://news.microsoft.com/de-de/mixed-reality-in-der-automobilindustrie-oder-wie-bosch-mobilitaet-digitaler-gestaltet/ (accessed on 15 April 2021).
14. IBM. What Is a CMMS? Available online: https://www.ibm.com/topics/what-is-a-cmms (accessed on 23 April 2021).
15. Farinha, J.M.T. *Asset Maintenance Engineering Methodologies*, 1st ed.; CRC Press: Boca Raton, FL, USA, 2018; ISBN-10 1138035890, ISBN-13 978-1138035898.
16. Johns, R. Real Python—PyTorch vs. Tensorflow For Your Python Deep Learning Project. 2020. Available online: https://realpython.com/pytorch-vs-tensorflow (accessed on 10 February 2021).
17. Microsoft. Micrisoft-HoloLens2. Available online: https://www.microsoft.com/en-us/hololens/hardware (accessed on 21 April 2021).
18. Vuzix. Vuzix—Manufacturing Solutions. Available online: https://www.vuzix.com/solutions/manufacturing (accessed on 21 April 2021).
19. Andreas Geiger. Welcome to the KITTI Vision Benchmark Suite! Available online: http://www.cvlibs.net/datasets/kitti/ (accessed on 10 April 2021).
20. The Comprehensive Cars (CompCars) Dataset. Available online: http://mmlab.ie.cuhk.edu.hk/datasets/comp_cars/index.html (accessed on 10 April 2021).
21. Analytics India Magazine. Analytics India Magazine—Guide to Yolov5 for Real-Time—Object Detection. 2020. Available online: https://analyticsindiamag.com/yolov5/ (accessed on 10 February 2021).
22. Glenn Jocher. 2020. Available online: https://github.com/ultralytics/yolov5 (accessed on 21 April 2021).
23. Buric, M.; Pobar, M.; Ivasic-Kos, M. Ball detection using YOLO and Mask R-CNN. In Proceedings of the 2018 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 12–14 December 2018; pp. 319–323.
24. Simon, M.; Amende, K.; Kraus, A.; Honer, J.; Samann, T.; Kaulbersch, H.; Milz, S.; Michael Gross, H. Complexer-yolo: Real-time 3d object detection and tracking on semantic point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, CA, USA, 16–17 June 2019.
25. Jacob, S.; Nelson, J. How to Train YOLOv5 on a Custom Dataset. 2020. Available online: https://blog.roboflow.com/how-to-train-yolov5-on-a-custom-dataset/ (accessed on 21 April 2021).
26. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
27. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
28. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 12–14 December 2018; pp. 7263–7271.