Electric Vehicle Charging Socket Detection using YOLOv8s Model

Vladimir Tadic[1,2,3], Akos Odry[4], Zoltan Vizvari[3,5], Zoltan Kiraly[1,2], Imre Felde[1], Peter Odry[2,3]

[1]Óbuda University, John von Neumann Faculty of Informatics, Becsi ut 96/B, 1034 Budapest, Hungary; tadity.laszlo@uni-obuda.hu; kiru@uniduna.hu; felde.imre@uni-obuda.hu

[2]University of Dunaujvaros, Tancsics Mihaly u. 1/A, 2401 Dunaujvaros, Hungary; tadityv@uniduna.hu; kiru@uniduna.hu; podry@uniduna.hu

[3]Symbolic Methods in Material Analysis and Tomography Research Group, Faculty of Engineering and Information Technology, University of Pecs, Boszorkany u. 6, H-7624 Pecs, Hungary; vizvari.zoltan@mik.pte.hu

[4]Faculty of Engineering, University of Szeged, Mars tér 7, 6724 Szeged, Hungary; odrya@mk.u-szeged.hu

[5]Department of Environmental Engineering, Faculty of Engineering and Information Technology, University of Pecs, Boszorkany u. 2, H-7624 Pecs, Hungary; vizvari.zoltan@mik.pte.hu

Abstract: This paper introduces the utilization of the latest small You Only Look Once version 8 – YOLOv8s convolutional neural network in an automatic electric vehicle charging application study. The employment of a deep learning-based object detector is a novel and significant aspect in robotic applications, since it is both, the initial and the fundamental step in a series of robotic operations, where the intent is to detect and locate the charging socket on the vehicle's body surface. The aim was to use a renowned and reliable object detector to ensure the reliable and smooth functioning of the deployed robotic vision system in an industrial environment. The experiments demonstrated that the deployed YOLOv8s model detects the charging socket successfully under various image capturing conditions, with a detection rate of 97.23%.

Keywords: YOLOv8s; Electric vehicle charging socket; Image processing; Object detection; Robotic applications; Automotive applications

1 Introduction

Electric vehicles (EVs) are electrifying the transportation landscape, projected to dominate roads worldwide [1-4]. The accelerated advancement of electric vehicles will lead to a growing demand for associated applications in the coming years. [4-6]. Robotic charging represents a crucial step towards seamless and user-friendly electric vehicle charging. This technological leap provides the basis for a future where recharging becomes effortless and efficient, further propelling the electric mobility revolution [7-8]. Seamless, hands-free charging is electrifying the future of EVs. As customer demand rises and autonomous driving advances, automated charging robots are prepared to revolutionize the recharging experience.

This initial study investigates the application of object detection techniques in a robotic charging system for electric vehicles within an automotive project. As a result, a novel approach for electric vehicle charging socket detection using the novel state-of-the-art You Only Look Once version 8 (YOLOv8) [9] object detection framework will be introduced.

The main task of this research is the deployment of a lightweight and efficient object detector for the detection of the Combined Charging System 2 (CCS2) socket of electric vehicles using a renowned and reliable object detector model. Thus, the socket detection procedure in complex image scenes is based on the latest YOLOv8 framework introduced in 2023 by Ultralytics, which is the sequel of the earlier versions of well known YOLO object detectors. The main demand of this job is to initially detect the charging socket in compound scenes in order, that in the later stage of the operation, the robotic arm with a specialized short range 3D camera will approach the detected socket in order to accurately determine its position in space. Further, one of the main project requirements was to use a well known, reliable and fast object detector framework intending to ensure the smooth operation of the whole system. It should be noted, that in the subsequent stages, the Universal Robot 10e (UR10e) equipped with a built-in force-torque sensor will be employed as the robotic arm for the implementation of an autonomous charging application [1-3]. A comprehensive description of the robot and its operations falls outside the scope of this paper, a complete exploration will be provided in a separate publication.

Finally, the developed YOLOv8-based procedure successfully fulfilled the project's objectives, demonstrating accurate and reliable CCS2 socket detection.

This research contributes significantly to industrial research by developing a novel, trained, and reliable object detector for charging socket detections in automated EV charging applications. Notably, using YOLOv8 for this purpose is unreported in both, scientific literature and industrial research projects focused on charging socket detection. This new approach offers a promising solution to a key challenge in upcoming industrial EV charging applications.

The paper can be summarized as follows. The first section is the introduction, the second section is the literature overview, the third section introduces the YOLOv8 framework and the proposed method. Section four shows the experiments and results. Finally, the conclusions are drawn with the future works plan.

2 Related Works

Isolating individual objects from cluttered scenes and pinpointing their location remains a key challenge for robot vision systems [15-26]. Object segmentation, critical for isolating specific shapes, suffers from a research gap in EV charging socket detection [27-36]. YOLOv8's popularity in building accurate, precise, and flexible object detectors makes it a promising candidate for addressing this gap. This work leverages YOLOv8's broad applicability and acceptable speed to develop a novel solution for EV charging socket detection. This section will provide a concise overview of the relevant studies in this domain, however the description of YOLOv8 is poorly represented in the scientific literature.

Hussain [37] delves into the YOLO family's evolution, from its inception to the latest YOLOv8, analyzing its architectural strengths within the industrial context. Slimani et al. [38] leveraged YOLOv8 for efficient and precise rust disease classification in fava bean field images. Their transfer learning-based model outperformed others, achieving an impressive 95.1% detection accuracy. Sharma et al. [39] present a real-time parking violation detection system using YOLOv8 for vehicle identification and a tracking algorithm for persistence. Future work will explore multi-camera synchronization. Talaat and ZainEldin [40] leverage YOLOv8's deep learning strengths for a real-time, camera-based fire detection system in smart cities, achieving a state-of-the-art 97.1% precision rate. Bai et al. [41] comprehensively investigated YOLOv8n object detection improvements, integrating Wasserstein Distance Loss, FasterNext, and Context Aggravation. They individually and collectively evaluated each approach, demonstrating YOLOv8n's superior balance between accuracy, complexity, and inference speed compared to other frameworks. Finally, it should be noted, that numerous deep learning models [38-44] are analyzed and tested in recent years for various applications and the development of many deep learning models is expected in the upcoming period.

3 Methods

3.1 YOLOv8 Framework

Launched in 2023, YOLOv8 takes object detection and image segmentation to the next level. While comprehensive documentation awaits, Ultralytics offers essential info on the framework and architecture.

YOLOv8 redefines object detection with unmatched speed and precision, thanks to cutting-edge deep learning and computer vision [45]. Unlike the Region-based Convolutional Neural Network (R-CNN) and Fast R-CNN models, which use a multi-stage process to detect objects in image, all YOLO models use a single neural network (single shot detection) to predict both, the bounding boxes and class probabilities of objects in images [46]. Trading accuracy for speed, YOLO excels in real-time tasks, potentially requiring slight adjustments for optimal accuracy in some scenarios [46]. YOLO models employ direct prediction of class probabilities and bounding boxes, obviating the need for region proposal algorithms. This single-stage architecture facilitates object detection in a single network pass, leading to significant speed advantages compared to multi-stage approaches [45-46]. YOLO adopts a grid-based approach, dividing the input image into cells and predicting object presence within each. Each cell subsequently predicts bounding boxes and class probabilities for potential objects within its bounds, facilitating multi-object and multi-scale detection in a single pass [37], [46].

YOLOv8 can be used for the three main computer vision tasks:

1. Classification; where the objective of the object detector model is to determine the predominant class present in the input image.
2. Object detection; where the goal is to not only identify various classes within an image, but also to precisely locate them inside the image.
3. Segmentation; where the goal is to identify individual pixels belonging to each object in the image.

Earlier YOLO models use anchor boxes [45], which are predefined bounding boxes in order to enhance the accuracy of its predictions. YOLOv8 departs from traditional anchor-based methods, directly predicting bounding box centers and class probabilities for objects within the input image [45]. This anchor-free architecture eliminates the need for predefined anchor boxes, simplifying the model and enabling greater flexibility in detecting objects of diverse sizes and aspect ratios. Additionally, reduced box predictions enhance Non-Maximum Suppression (NMS) efficiency, a post-processing step critical for accurate object localization [45-46].

YOLOv8 distinguishes itself by leveraging the powerful mosaic augmentation technique [45]. Data augmentation artificially expands training data through manipulation of existing samples, enhancing model robustness and generalizability.

One of the well-known and powerful annotation and augmentation tools is the Roboflow Framework [47] that is used in this research. While mosaic augmentation offers significant advantages, YOLOv8 strategically disables it during the final 10 training epochs [45]. This fine-tuning step allows the model to focus on the original data distribution, potentially mitigating performance decline observed with prolonged mosaic augmentation [45].

The activation function used in YOLOv8 is the Sigmoid Linear Unit (*SiLU*) function [45-46]:

$$SiLU(x) = x\sigma(x) \tag{1}$$

where the $\sigma(x)$ is the Sigmoid function defined as [46]:

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{2}$$

The activation function plays a prominent role in determining whether a neuron should be activated, or not. It is accomplish by computing the weighted sum and incorporating bias. The primary goal of the activation function is to inject non-linearity into the neuron's output [45-46, 48].

Further, the loss of the YOLOv8 model is obtained with two functions, the Binary Cross Entropy (*BCE*) calculates the classification loss, while for the bounding box loss the Complete Intersection over Union (*CIoU*) and the Distribution Focal Loss (*DFL*) is calculated [46, 48]. A loss function serves as a metric to evaluate the disparity between the predicted and target output values, quantifying how efficiently the neural net represents the training data. During the training procedure, the aim is to minimize this loss, thereby enhancing the network's ability to accurately predict target outputs [45-46, 48].

The *BCE* and *CIoU* functions are defined as:

$$BCE = -\frac{1}{N}\sum_{i=1}^{N} y_i \log\big(p(y_i)\big) + (1-y_i)\log\left(1-p(y_i)\right) \tag{3}$$

$$CIoU = 1 - IoU + \left|\frac{\rho^2(b,b^{gt})}{c^2}\right| + \beta v \tag{4}$$

where *b* and b^{gt} are the central point of the predicted bounding box *B* and the central point of the ground-truth box B^{gt}, where *gt* labels the ground-truth and *N* is the number of object classes. The ρ parameter represents the Euclidean distance, $y \in \{1,0\}$ specifies the ground-truth class, $p \in [0,1]$ is the probality for the class label $y = 1$ and *c* represents the diagonal length of the smallest enclosing box that masks the two boxes [46, 48]. The β parameter labels the trade-off and the *v* parameter measures the consistency of the aspect ratio. The *v* and β are represented as follows [46]:

$$v = \frac{4}{\pi}\left(arctan\frac{w^{gt}}{h^{gt}} - arctan\frac{w}{h}\right)^2 \tag{5}$$

$$\beta = \frac{v}{1-IoU+v} \tag{6}$$

where *w* and *h* are the width and the height of the bounding box respectively. The Intersection over Union (*IoU*) is obtained with the predicted bounding box *B* and the ground-truth box B^{gt} with the following equation [46]:

$$IoU = \frac{B \cap B^{gt}}{B \cup B^{gt}} \tag{7}$$

The *CIoU* in equation (4) considers the overlapping area, the aspect ratio and the central point distance into account. It is an ameliorated variant of *IoU*, and its consequence is, that it converges faster and it is more effective in executing the bounding box regression [45-46].

Further, the problem with a classic loss functions such as the *BCE* loss function is that suchlike functions handle the missclassifications equally. In the object detection task this can be an issue since the huge majority of the image regions do not contain any object/shape and this could lead to a class inequity problem. Thus, the focal loss function handles this issue by down-weighting the loss allotted to well-classified samples. Further, the *DFL* is built upon by including the class distribution information into the focal loss function. The goal is to learn a dynamic weighting scheme for the loss function built on the distribution of classes in the training data. This permits the YOLOv8 model to assign more weights to the low-represented classes and less weights to the over-represented classes which can lead to more precise bounding box assessment [46, 48].

When the continuous distribution of the regression value is transformed to the discrete domain, the assessed regression value can be expressed as follows [46]:

$$\hat{y} = \sum_{i=0}^{n} P(y_i)\, y_i \tag{8}$$

where the $\hat{y}$ is the estimated regression value and the *n* is the number of classes in this case.

Using the Softmax functions [48]:

$$S_i = \frac{y_{i+1} - y}{y_{i+1} - y_i} \tag{9}$$

$$S_{i+1} = \frac{y - y_i}{y_{i+1} - y_i} \tag{10}$$

$\hat{y}$ can be written as follows [46,48]:

$$\hat{y} = \sum_{i=0}^{n} P(y_i)\, y_i = S_i y_i + S_{i+1} y_{i+1} = \frac{y_{i+1} - y}{y_{i+1} - y_i} y_i + \frac{y - y_i}{y_{i+1} - y_i} y_{i+1} = \mathrm{y} \tag{11}$$

Finally, the *DFL* can be expressed as [46, 48]:

$$DFL(S_i, S_{i+1}) = ((y_{i+1} - y)(S_i) + (y - y_i)\log(S_{i+1})) \tag{12}$$

As an estimation metric, the image recognition community decided to use the mean Average Precision (*mAP*) for object detector models [45-46]. The *mAP* is a combination of recall and precision values determined over multiple confidence thresholds, the IoU. The variation of the IoU threshold will result in different True Positives (*TP*) and False Positives (*FP*) predictions in image.

The precision is determined as the fraction of *TP* detections among all detections made at a specific IoU threshold [46]:

$$Precision = \frac{TP}{TP+FP} \tag{13}$$

The recall is determined as the fraction of *TP* detections found among all possible detections made at a specific threshold [46]:

$$Recall = \frac{TP}{TP+FN} \tag{14}$$

where *FN* are False Negatives predictions in the image.

Finally, the formula for *mAP* is determined as [46]:

$$mAP = \frac{1}{N}\sum_{n=1}^{N} AP_i \tag{15}$$

where AP_i is the average precision for the *i*-th class and *N* is the number of object classes.

The architecture of YOLOv8 is built upon the previous models of YOLO object detectors. The description of the YOLOv8 model architecture is provided by RangeKing [49]. The architecture is presented in Figure 1. At the core of YOLOv8 lies a convolutional neural network (CNN) architected into two distinct modules: the feature extraction backbone and the object detection head [45-49]. The backbone leverages a modified variant of the Cross Stage Partial (CSP) Darknet53 architecture, comprising 53 convolutional layers and specializing in extracting salient features from the input image [45, 49]. The CSP architecture employed in the backbone facilitates robust information flow by introducing cross-stage partial connections between network layers [45, 49]. Following this, the YOLOv8 head comprises a cascade of convolutional layers culminating in a sequence of fully connected layers. These layers work in tandem to perform critical object detection tasks [45, 49]. YOLOv8 distinguishes itself by incorporating a self-attention mechanism within its head [45, 49]. This mechanism endows the model with the ability to dynamically attend to different regions of the input image, selectively allocating emphasis to features based on their pertinence to the object detection task [45, 49]. YOLOv8 further distinguishes itself by its multi-scale object detection capability. This is achieved through its integration of a feature pyramid network (FPN), which enables the model to effectively localize objects across a wide range of sizes and scales within the input image. The FPN architecture comprises multiple feature levels, each specializing in the detection of objects at specific scales. This allows YOLOv8 to simultaneously identify both large and minute objects present in the image, leading to comprehensive and accurate detection performance [45, 49].

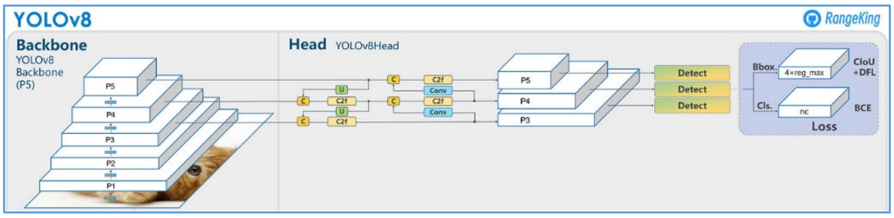

Figure 1

YOLOv8 model architecture (provided by RangeKing [48])

Further, YOLOv8 supports five different neural network sizes that vary in the amount of parameters present in the neural network: YOLOv8n (n-nano), YOLOv8s (s-small), YOLOv8m (m-medium), YOLOv8l (l-large) and YOLOv8x (x-extra large) [45]. The "n" model has the smallest number of parameters, while the "x" model has the largest number of parameters. The parameters are referring on the number of biases and weights in the network [45-49]. While YOLOv8n is the smallest and the fastest model, on the other hand the YOLOv8x is the most accurate and slowest among the YOLOv8 models. Based on the requirements of the application itself, the appropriate model could be selected [45]. Since the aim of the research is to detect the charging socket of the e-vehicle with reliable accuracy in a short time, the YOLOv8s model has been chosen to accomplish this task. According to Ultralytics test on Microsoft Common Objects in Context (MS COCO) dataset [45-48], the YOLOv8s model is to a lesser extent slower than the YOLOv8n model, and it is significantly faster than the other YOLOv8 models according to Open Neural Network Exchange (ONNX) results, while the mean Average Precision on the validation dataset (mAP^{VAL}) is not significantly lower compared to larger models (Table 1). Thus, based on the recommendations [45-46] and test results, the decision fell on the YOLOv8s model in this research.

Table 1

Comparison of YOLOv8 models on COCO dataset [45]

Model	Image Size (pixels)	mAP^{VAL}	Speed CPU ONNX (ms)
YOLOv8n	640	37.3	80.4
YOLOv8s	640	44.9	128.4
YOLOv8m	640	50.2	234.7
YOLOv8l	640	52.9	375.2
YOLOv8x	640	53.9	479.1

3.2 Dataset Preparation and YOLOv8s Training

The overall pipeline of the dataset preparation, training and deployment of the YOLOv8 models is presented in Figure 2. The first step is the dataset preparation which includes the collecting of images and their annotation which can include some arbitrary pre-processing such as contrast adjustment, resizing, etc. [45, 47].

After annotation and pre-processing, the next step involves augmentation, where training examples are generated based on selected augmentation options such as mosaics, rotation, shear, bounding box orientation, etc. [45, 47]. As it was noted, the Roboflow Framework and its annotation tool was used in this research which includes optional pre-processing and augmentation options [47]. Later, the annotated images can be split in training, validation and testing folders, while the annotated labels are saved in *.txt files, according to the YOLOv8 models requirements [47]. Finally, the prepared dataset can be exported to YOLOv8 models format and used for training, validation and testing [47]. The second step is the training of the YOLOv8s model object detector according to Ultralytics guidelines [45] in this research and finally the third step is the deployment of the trained network on sample images.

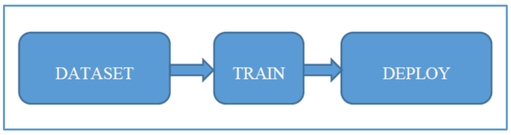

Figure 2
The overall pipeline of the YOLOv8 model [45]

The dataset used in this initial study contains 1125 self-created images in order to avoid any authorship issues. Further, all the images were captured with a modest quality camera since the industrial camera will be procured with the UR10e cobot in future steps of the project. Since the goal was to train a robust object detector, the images were captured under various capturing conditions which includes various lighting, shadow, distance, background, etc. It should be mentioned, that the CCS2 socket is black and its near background is very dark, mostly black. This fact is important since it is obvious that the detection of this kind of socket is a difficult task for all object detection frameworks. Thus, even the training of the YOLOv8s is a challenging job.

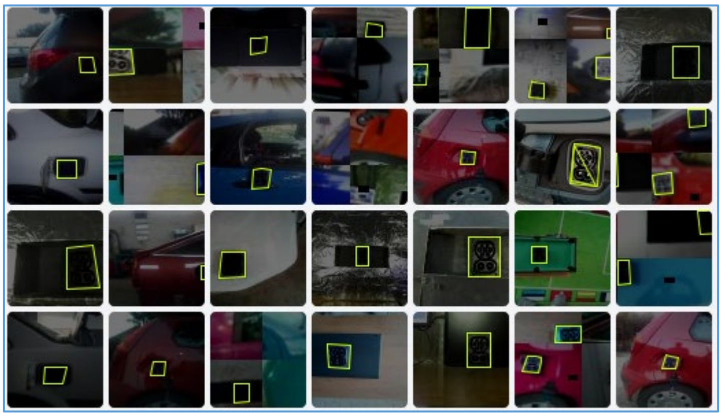

Figure 3
The examples of annotated images [47]

After the images were uploaded to Roboflow, the annotation is performed with the frameworks built-in annotation tool [47]. The part of the annotated image dataset can be seen in Figure 3. All the annotated images are labelled appropriately and the labelling data is saved in the corresponding *.txt file [47]. The annotation was followed by an optional pre-processing steps that are included in the online platform [47]. The resizing to 128x70 pixels, automatic contrast adjustment and automatic orientation options were chosen for the image dataset pre-processing and it was executed according to the Roboflow's built-in pre-processing algorithm [47].

The pre-processing is followed by the augmentation step. As it was mentioned, the data augmentation is an addition of new artificially derived data from existing training data with a goal to enhance the training of the neural network [45, 47]. In this study, 10 augmentation options were included as it can be seen in Table 2. All these augmentation operations are randomly applied to the whole dataset according to Roboflow's built in algorithm [47]. Finally, the dataset was split in three folders according to Roboflow's recommended splitting procedure. After the split is performed, the new folders are: the training folder, validation folder and test folder. Hence, the training folder contains 975, the validation 93 and the test 57 images respectively. The next step is the exportation of the dataset in ZIP file in the appropriate YOLOv8 format provided by Roboflow with the corresponding *.yaml file that contains the information related to dataset folders, number of object classes ("nc") and the objects names ("names") [47].

Table 2
Data augmentation using Roboflow [47]

Augmentation	**Options/Values**
Rotation	Between -5° and +5°
Shear	±5° Horizontal, ±5° Vertical
Hue	Between -25° and +25°
Saturation	Between -25% and +25%
Brightness	Between -10% and +10%
Exposure	Between -10% and +10%
Cutout	1 box with 5% size each
Bounding Box: Orientation	Between -5° and +5°
Bounding Box:Shear	±5° Horizontal, ±5° Vertical
Bounding Box: Exposure	Between -25% and +25%
Mosaic	Applied

Later, the ZIP file should be extracted in a main folder, where the training process will be performed. All this information is required during the training, validation and testing process of the YOLOv8s model, thus the *.yaml file should be included in the main folder with the dataset folders [45-49]. If the *.yaml file is not provided, or it is incorrectly configured, the YOLOv8s network will not find the suitable data for the training. Finally, after the YOLOv8 framework is installed and set up in the main folder, the training process can start, followed by the validation and testing steps in later stages [45-49]. The training process is performed according to Ultralytics instructions, mainly with the default settings provided by the framework itself in Command Line Interface (CLI) in Command Prompt on Windows 10 platform [45]. The used hardware platform is CPU Intel CORE(TM) i7-10700 2.90 GHz with 16 GB RAM (without Graphical Processing Unit - GPU), while the software platform is Ultralytics Yolov8.0.2.212, Python-3.10.9, torch-1.13.1. The training process was done with the YOLOv8s model, with 100 epochs, image size of 640 pixels and number of images per batch 32 [45-48], while all the other parameters were remained default [45]. The one epoch is when an entire dataset is passed forward and backward through the neural network only once, while the number of batches is a divided dataset into smaller sets or parts (called batches) that are passed through the neural network [45, 47]. The batch size is the total number of training examples present in a single batch [45, 47]. The training time was 21.133 hours and the achieved mAP50 was 0.928, the mAP50-95 was 0.745, the recall was 0.889, the precision was 0.947 and the inference time was 100.6 ms on the training dataset. The mAP50 refers to the calculation of the average precision across different levels of recall, up to a limit of 50 detections per image. This metric helps assess how well a model is performing in terms of both precision and recall, with a focus on the top 50 predictions [45]. The mAP50-95 is an extension of the mean mAP metric, specifically considering a range of IoU thresholds. The mAP50-95 is calculated by averaging the Average Precision (AP) values over a range of IoU thresholds, typically from 0.5 to 0.95, in increments of 0.05. The diagrams of the training results are shown in Figure 4, where the X axis represents the number of epochs, and the Y axis represents the corresponding parameter: the precision, recall, mAP50 and mAP50-95 respectively. Obviously, the training time would be shorter with an available GPU [45].

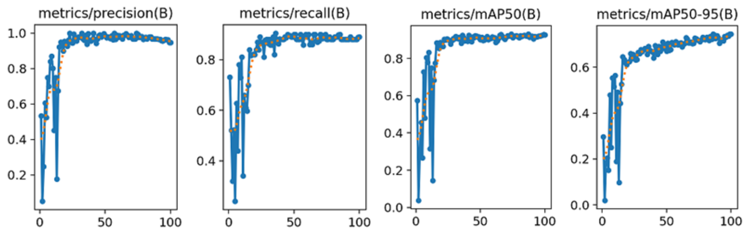

Figure 4

The training results diagrams [45]

In order to show the detecting capabilities of the trained YOLOv8s model, Fig. 5 presents several detecting results from the validation dataset. As it can be noticed, the detection results are considerably accurate and acceptable, even in the examples were the CCS2 socket is poorly visible and distinguishable from the background.

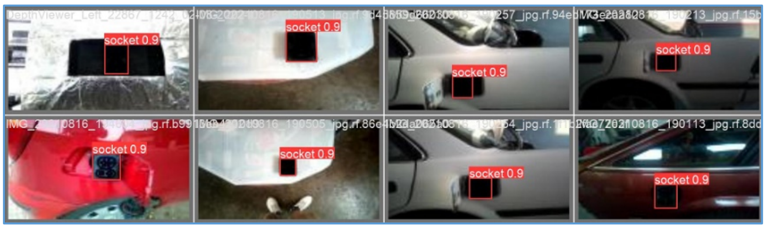

Figure 5
The results from the validation dataset

Further, Figure 6 presents several detecting results from the test dataset. As it can be seen, the detection results are also very accurate and acceptable, although the images are of poor quality.

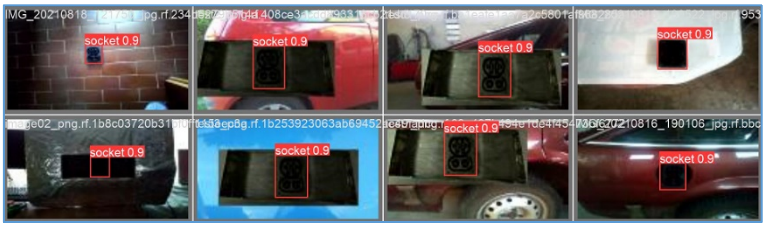

Figure 6
The results from the test dataset

In the end, the training results of the YOLOv8s are considered acceptable and they fulfilled the goal of the study. In later developments, an industrial computer with an GPU should be acquired for the work and the control of the whole robotic system that should provide an adequate hardware support for the possible novel YOLOv8s model training. Also, it should be noted that when the UR robot arrives, the industrial camera will be installed on the robotic arm and the testing will be executed in real time on the vehicle body model with a built-in CCS2 socket and with e-vehicles. In the final stage when the system is verified and certified, the testing will be extended to other available e-vehicles.

4 Experiments and Results

In this section, a comprehensive explanation of the conducted experiments and the corresponding results will be presented. As it was highlighted in the Introduction, the primary objective of this initial study is to exclusively employ established and verified CNN object detector in the development of the CCS2 socket detection procedure. Hence, the trained YOLOv8s model's performance was assessed using artificial vehicle body model equipped with the socket. A diverse range of capturing conditions were deliberately examined to thoroughly assess and delineate the capabilities and constraints inherent in the implemented object detector. It should be noted, that one of the main aspects in the development of the object detector model is the generation of a high-quality, usable input image appropriate for industrial applications that would be ensured in the real application. In addition to the use of high-quality capturing devices, the main requirement is the formation of an appropriately illuminated environment without disturbing effects, which can ensure the repeatability of a quality image capturing later in the commercial use on the parking lot. Herein, in the experiments, a modest quality camera was used in order to determine and examine the possibilities of the trained and deployed YOLOv8s model for initial study purposes. The image dataset was generated internally using a test vehicle body model equipped with an original CCS2 socket, and it contains 975 images. The images were captured across diverse recording conditions, deliberately including instances under less-than-ideal capturing conditions in numerous examples in order to examine the robustness and the limitations of the trained YOLOv8s object detector. Different capturing conditions includes: various camera distance and angle position, intentional shading, various illumination conditions, hazy images, etc.

Since this is an initial research, using the self-created internal image database serves the purpose of avoiding potential legal repercussions that may arise from the use of images depicting proprietary vehicles. Thus, the testing on real e-vehicles will be arranged at a later stage of the project, when all the equipment with the constructed charging station will be available, with legally rented and insured vehicles by the project management. The self-created dataset is not public since it is a part of a commercial industrial project, and it can be provided only with the permission of the project management and the project client.

The testing and the prediction with YOLOv8 object detector has been executed automatically on the whole dataset placed in a custom folder, with a proposed prediction options [45]. The prediction was launched with the command line interface (CLI), where the confidence threshold was set to 0.2, the image augmentation to prediction sources was turned on during the testing and the result saving option was activated [45]. All the other parameters remained the default and the detection results were saved in a separate folder for evaluation purposes [45].

The testing and prediction results achieved a considerable accuracy since in 948 images the detection was correct while in 27 images the detection failed (in 13 samples were no detection, and in 14 samples were false detection). This detection resulted with a 97.23% accuracy on the available self-developed image dataset. Figure 6 displays 25 samples with correctly detected CCS2 sockets (from the 948 correct samples). As it can be noticed, all the samples are captured under various capturing conditions, mostly with poor quality since the aim was to assess the capabilities and limitations of the trained YOLOv8s object detector. Further, it can observed, that even under excessive slant, hazy image, or shadow the socket is successfully detected with a lower confidence level, however the bounding box is correctly drawn around the detected object. Also, in examples where the small part of the CCS2 is missing due to the excessive slant, a correct detection is achieved. One of the reasons of successful detection under inordinate image capturing conditions is the activated augmentation option during the prediction according to the related documentation [45]. Therefore, this is also a great advantage of the YOLOv8s model in applications where incorrect input images are expected during the work. Naturally, in the real application an appropriate lighting source will be mounted on the parking lot, and an industrial AD/3D camera is intended to be used with a special speckle-free blue laser. All these will contribute to a much better-quality input image, which will certainly facilitate and improve the detection result of the trained YOLOv8s object detector.

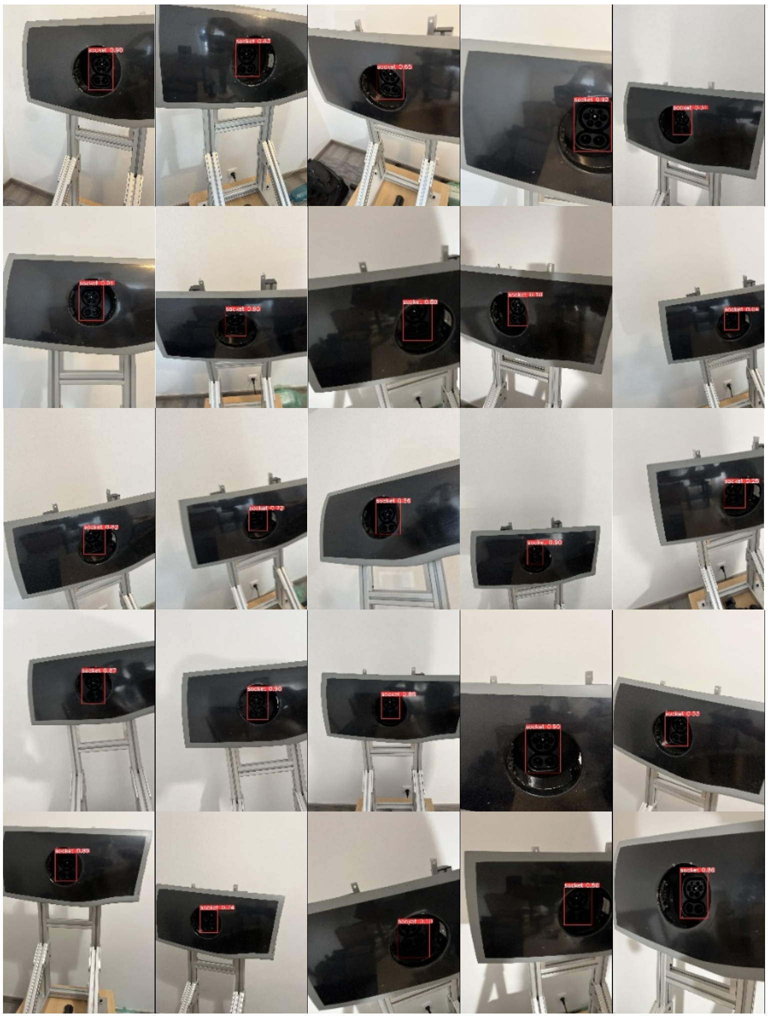

Figure 7

The detection results from the self-created dataset

In the end, this initial study entirely fulfilled the aim of the project, and the detection of the charging socket with a renowned YOLOv8s object detector was achieved. The practical application of the obtained results will be tested in the future, where the experiments will be executed with an UR10e robot on electric vehicles with adequate industrial vision and sensing equipment. The future managerial implications are the legal rent of a certain number of electric vehicles for testing purposes and the construction of an adequate real-world parking lots for experiments with the robot in an industrial environment.

Conclusions

This study introduced the training and deployment of YOLOv8s for charging socket detection in automated EV charging. Key steps in dataset preparation, training, and deployment were presented. Aiming for high robot reliability, a well-known object detector was chosen. Experiments on a custom vehicle model demonstrated a successful detection, with YOLOv8s showing good accuracy, robustness, and resistance to lighting variations and slant. Limitations under certain lighting and slant conditions were identified and will be addressed in future work with a high-quality industrial camera. Deployment incorporated all project requirements and achieved the set goal. Further development will involve industrial camera integration with both the collaborative robot and EVs on a built parking lot.

Acknowledgement

This research is a part of projects GINOP_PLUSZ-2.1.1-21-2022-00249 of Óbuda University and 2020-1.1.2-PIACI-KFI-2020-00173 of University of Dunaujvaros.

References

[1] P. Akella et al., "Cobots for the automobile assembly line", Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C), 1999, pp. 728-733, Vol. 1, doi: 10.1109/ROBOT.1999.770061

[2] Asif, Seemal, Philip Webb, "Realtime Calibration of an Industrial Robot", Applied System Innovation 5, No. 5: 96, 2022, https://doi.org/10.3390/asi5050096

[3] https://www.universal-robots.com/, Accessed: 08.09.2023

[4] K. W. E. Cheng, "Recent development on electric vehicles", 2009 3rd International Conference on Power Electronics Systems and Applica-tions (PESA), 2009, pp. 1-5, ISBN:978-1-4244-3845-7

[5] X. Zhou et al., "The current research on electric vehicle," 2016 Chinese Control and Decision Conference (CCDC), 2016, pp. 5190-5194, doi: 10.1109/CCDC.2016.7531925

[6] H. S. Matharu, V. Girase, D. B. Pardeshi and P. William, "Design and Deployment of Hybrid Electric Vehicle, "2022 International Con-ference on

Electronics and Renewable Systems (ICEARS), 2022, pp. 331-334, doi: 10.1109/ICEARS53579.2022.9752094

[7] W. Luo and L. Shen, "Design and Research of an Automatic Charging System for Electric Vehicles", 2020 15th IEEE Conference on In-dustrial Electronics and Applications (ICIEA), 2020, pp. 1832-1836, doi: 10.1109/ICIEA48937.2020.9248188

[8] H. Wang, "A New Automatic Charging System for Electric Vehicles", 2021 2nd International Conference on Computing and Data Science (CDS), 2021, pp. 19-26, doi: 10.1109/CDS52072.2021.00011

[9] https://docs.ultralytics.com/

[10] Tadic, V.; Odry, A.; Burkus, E.; Kecskes, I.; Kiraly, Z.; Klincsik, M.; Sari, Z.; Vizvari, Z.; Toth, A.; Odry, P., "Painting Path Planning for a Painting Robot with a RealSense Depth Sensor", Appl. Sci. 2021, 11, 1467

[11] Tadic, V.; Odry, A.; Burkus, E.; Kecskes, I.; Kiraly, Z.; Vizvari, Z.; Toth, A.; Odry, P., "Application of the ZED Depth Sensor for Painting Robot Vision System Development", IEEE Access 2021, 9, 117845-117859

[12] Tadic, V.; Toth, A.; Vizvari, Z.; Klincsik, M.; Sari, Z.; Sarcevic, P.; Sarosi, J.; Biro, I., "Perspectives of RealSense and ZED Depth Sensors for Robotic Vision Applications", Machines 2022, 10, 183, https://doi.org/10.3390/machines10030183

[13] R. C. Gonzales and R. E. Woods, Digital Image Processing, 4th ed., Pearson, NJ, USA, 2018

[14] R. C. Gonzales, R. E. Woods, and S. L. Eddins, Digital Image Processing Using MATLAB, 3rd ed. Knoxville, TN, USA: Gatesmark, 2020

[15] Fabrizio Flacco, Torsten Kroger, Alessandro De Luca, Oussama Khatib, "A Depth Space Approach to Human-Robot Collision Avoid-ance", 2012 IEEE International Conference on Robotics and Automation RiverCentre, Saint Paul, Minnesota, USA May 14-18, 2012

[16] Ashutosh Saxena, Sung H. Chung, Andrew Y. Ng, "3-D Depth Reconstruction from a Single Still Image", International Journal of Computer Vision, 2008, Volume 76, Issue 1, pp. 53-69

[17] Vladimiros Sterzentsenko, Antonis Karakottas, Alexandros Papachristou, Nikolaos Zioulis, Alexandros Doumanoglou, Dimitrios Zarpalas, Petros Daras, "A low-cost, flexible and portable volumetric capturing system", 14th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), 2018, doi: 10.1109/SITIS.2018.00038

[18] Nicole Carey, Radhika Nagpal, Justin Werfel, "Fast, accurate, small-scale 3D scene capture using a low-cost depth sensor", 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), DOI: 10.1109/WACV.2017.146

[19] Mathieu Labbé, François Michaud, “RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation”, J Field Robotics. 2018; 1-31, DOI: 10.1002/rob.21831

[20] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Dolha, Michael Beetz, “Towards 3D Point cloud based object maps for household environments”, Robotics and Autonomous Systems 56, 2008, 927_941, doi:10.1016/j.robot.2008.08.005

[21] Tobias Schwarze, Martin Lauer, “Wall Estimation from StereoVision in Urban Street Canyons”, Proceedings of the 10th International Conference on Informatics in Control, Automation and Robotics, pp. 83-90, DOI: 10.5220/0004484600830090

[22] Jean-Emmanuel Deschaud, François Goulette, “A Fast and Accurate Plane Detection Algorithm for Large Noisy Point Clouds Using Filtered Normals and Voxel Growing”, 3DPVT, May 2010, Paris, France, hal-01097361

[23] Aghi, D.; Mazzia, V.; Chiaberge, M., “Local Motion Planner for Autonomous Navigation in Vineyards with a RGB-D Camera-Based Algorithm and Deep Learning Synergy”, Machines, 2020, 8, 27, https://doi.org/10.3390/machines8020027l

[24] Kin-Choong Yow1, Insu Kim, “General Moving Object Localization from a SingleFlying Camera”, Applied Sciences, 2020, 10, 6945; doi:10.3390/app10196945

[25] Xianyu Qi, Wei Wang1, Ziwei Liao, Xiaoyu Zhang, Dongsheng Yang, Ran Wei, “Object Semantic Grid Mapping with 2D LiDAR and RGB-D Camera for Domestic Robot Navigation”, Applied Sciences, 2020, 10, 5782; doi:10.3390/app10175782

[26] Vladimir Tadic, Akos Odry, Attila Toth, Zoltan Vizvari, Peter Odry, “Fuzzified Circular Gabor Filter for Circular and Near-Circular Object Detection”, IEEE Access, DOI: 10.1109/ACCESS.2020.2995553

[27] Mingqiang Pan, Cheng Sun, Jizhu Liu, Yangjun Wang, “Automatic recognition and location system for electrlc vehicle charging port in complex environment”, IET lmage Processing, 2020, Vol. 14, 188.10, pp. 2263-2272, doi: 10.1049/iet-ipr.2019.1138

[28] Hui Zhang, Xiating Jin, “A Method for New Energy Electric Vehicle Charging Hole Detection and Location Based on Machine Vision”, 5th International Conference on Environment, Materials, Chemistry and Power Electronics, EMCPE, 2016, Atlantis Press

[29] Justinas Mišeikis, Matthias Rüther, Bernhard Walzel, Mario Hirz and Helmut Brunner, “3D Vision Guided Robotic Charging Station for Electric and Plug-in Hybrid Vehicles”, Proceedings of the OAGM&ARW Joint Workshop, 2017, doi: 10.3217/978-3-85125-524-9-13

[30] P. Quan, Y. Lou, H. Lin, Z. Liang and S. Di, "Research on Fast Identification and Location of Contour Features of Electric Vehicle Charging Port in Complex Scenes", in IEEE Access, Vol. 10, pp. 26702-26714, 2022, doi: 10.1109/ACCESS.2021.3092210

[31] Quan, P.; Lou, Y.; Lin, H.; Liang, Z.; Wei, D.; Di, S., "Research on Fast Recognition and Localization of an Electric Vehicle Charging Port Based on a Cluster Template Matching Algorithm", MDPI Sensors 2022, 22, 3599. https://doi.org/10.3390/s22093599

[32] Y. Lou and S. Di, "Design of a Cable-Driven Auto-Charging Robot for Electric Vehicles", in IEEE Access, Vol. 8, pp. 15640-15655, 2020, doi: 10.1109/ACCESS.2020.2966528

[33] Lin, H.; Quan, P.; Liang, Z.; Lou, Y.; Wei, D.; Di, S., "Collision Localization and Classification on the End-Effector of a Cable-Driven Manipulator Applied to EV Auto-Charging Based on DCNN–SVM", MDPI Sensors, 2022, 22, 3439, https://doi.org/10.3390/s22093439

[34] Li, T.; Xia, C.; Yu, M.; Tang, P.; Wei, W.; Zhang, D., "Scale-Invariant Localization of Electric Vehicle Charging Port via Semi-Global Matching of Binocular Images", Appl. Sci. 2022, 12, 5247, https://doi.org/10.3390/app12105247

[35] Damien Chablat, Riccardo Mattacchione, Erika Ottaviano, "Design of a robot for the automatic charging of an electric car", ROMANSY 24 - Robot Design, Dynamics and Control, Springer, 2022, hal-03624780 Vladimir

[36] Vladimir Tadic, "Study on Automatic Electric Vehicle Charging Socket Detection using ZED 2i Depth Sensor", MDPI Electronics, 2023, 12, 912, https://doi.org/10.3390/electronics12040912

[37] Muhammad Hussain, "YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection", MDPI Machines 2023, 11, No. 7: 677, https://doi.org/10.3390/machines11070677

[38] Hicham Slimani, Jamal El Mhamdi, Abdelilah Jilbab, "Artificial Intelligence-based Detection of Fava Bean Rust Disease in Agricultural Settings: An Innovative Approach", International Journal of Advanced Computer Science and Applications, Vol. 14, No. 6, 2023

[39] Nabin Sharma, Sushish Baral, May Phu Paing, and Rathachai Chawuthai, "Parking Time Violation Tracking Using YOLOv8 and Tracking Algorithms", MDPI Sensors, 2023, 23, No. 13: 5843, https://doi.org/10.3390/s23135843

[40] Talaat, F. M., ZainEldin, H., "An improved fire detection approach based on YOLO-v8 for smart cities", Neural Computing & Application, 35, 20939-20954, 2023, Springer, https://doi.org/10.1007/s00521-023-08809-1

[41] Ruihan Bai, Feng Shen, Mingkang Wang, Jiahui Lu, Zhiping Zhang, "Improving Detection Capabilities of YOLOv8-n for Small Objects in Remote Sensing Imagery: Towards Better Precision with Simplified Model Complexity", 22 June 2023, Preprint Version 1 available at Research Square [https://doi.org/10.21203/rs.3.rs-3085871/v1]

[42] Fatemeh Rashidi Fathabadi, Janos L. Grantner, Ikhlas, Abdel-Qader, Saad A. Shebrain M. D., "Box-Trainer Assessment System with Real-Time Multi-Class Detection and Tracking of Laparoscopic Instruments, using CNN", Acta Polytechnica Hungarica, Vol. 19, No. 2, 2022, DOI:10.12700/APH.19.2.2022.2.1

[43] Lei Kou, "A Review of Research on Detection and Evaluation of the Rail Surface Defects", Acta Polytechnica Hungarica, Vol. 19, No. 3, 2022, DOI:10.12700/APH.19.3.2022.3.14

[44] Man-Wen Tian, Ardashir Mohammadzadeh, Jafar Tavoosi, Saleh Mobayen, Jihad H. Asad, Oscar Castillo, Annámaria R. Várkonyi-Kóczy, "A Deep-learned Type-3 Fuzzy System and Its Application in Modeling Problems", Acta Polytechnica Hungarica, Vol. 19, No. 2, 2022, DOI:10.12700/APH.19.2.2022.2.9

[45] https://docs.ultralytics.com/, Accessed: 05.10.2023

[46] Frederik Fritsch, "Deep Neural Networks for Object Detection in Satellite Imagery", UPTEC IT23 014 Master Thesis, Uppsala Universitet, 2023

[47] https://roboflow.com/, Accessed: 05.08.2023

[48] Xiang Li, Wenhai Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang and Jian Yang, "Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection", arXiv:2006.04388, 2020

[49] Yolov8 model architecture layout. https://github.com/RangeKing, Accessed: 08.10.2023