Xinyao Zhang
Environmental Engineering Sciences,
University of Florida,
Gainesville, FL 32611
e-mail: xinyaozhang@ufl.edu

Kareem Eltouny
Civil, Structural, and Environmental Engineering,
University at Buffalo,
Buffalo, NY 14260
e-mail: keltouny@buffalo.edu

Xiao Liang
Assistant Professor
Civil, Structural, and Environmental Engineering,
University at Buffalo,
Buffalo, NY 14260
e-mail: liangx@buffalo.edu

Sara Behdad[1]
Associate Professor
Environmental Engineering Sciences,
University of Florida,
Gainesville, FL 32611
e-mail: sarabehdad@ufl.edu

Automatic Screw Detection and Tool Recommendation System for Robotic Disassembly

Disassembly is an essential process for the recovery of end-of-life (EOL) electronics in remanufacturing sites. Nevertheless, the process remains labor-intensive due to EOL electronics' high degree of uncertainty and complexity. The robotic technology can assist in improving disassembly efficiency; however, the characteristics of EOL electronics pose difficulties for robot operation, such as removing small components. For such tasks, detecting small objects is critical for robotic disassembly systems. Screws are widely used as fasteners in ordinary electronic products while having small sizes and varying shapes in a scene. To enable robotic systems to disassemble screws, the location information and the required tools need to be predicted. This paper proposes a computer vision framework for detecting screws and recommending related tools for disassembly. First, a YOLOv4 algorithm is used to detect screw targets in EOL electronic devices and a screw image extraction mechanism is executed based on the position coordinates predicted by YOLOv4. Second, after obtaining the screw images, the EfficientNetv2 algorithm is applied for screw shape classification. In addition to proposing a framework for automatic small-object detection, we explore how to modify the object detection algorithm to improve its performance and discuss the sensitivity of tool recommendations to the detection predictions. A case study of three different types of screws in EOL electronics is used to evaluate the performance of the proposed framework.
[DOI: 10.1115/1.4056074]

Keywords: screw detection, robotic disassembly, consumer electronics, YOLOv4, EfficientNetv2, sustainable manufacturing, assembly

1 Introduction

The rapid consumption of consumer electronics and the subsequent waste generation rate have pushed corporates to consider remanufacturing as a promising strategy for the efficient recovery of end-of-life (EOL) products. Despite the importance of remanufacturing, the technology development to support product recovery operations has been very limited [1]. Often consumer electronics need to be disassembled before harvesting components. However, disassembly is a labor-intensive task performed by human operators under risky conditions.

Manual disassembly is widely used due to its high flexibility in handling traditional disassembly operations [2]. However, manual operations are costly and negatively impact human health. To address this issue, human–robot partnership has been investigated as an alternative. Disassembly automation has already been incorporated into waste management [3–5] and topics such as disassembly line balancing and work assignment among humans and robots to satisfy demand while minimizing cost have been investigated [6]. Besides cost-efficiency, disassembly robots can perform repetitive tasks to reduce human fatigue [7].

While the disassembly process can benefit from robots, equipping robots with accurate detection capabilities is very challenging, as robots need to be familiar with the structure of EOL products before any operation. A computer vision system can support robots in identifying and locating components before disassembly actions [8,9]. Nevertheless, current computer vision techniques are not robust in detecting small objects, especially screws and fasteners. Moreover, when recognizing real objects in industrial environments, lighting conditions or viewpoints are not always consistent, and slight changes can alter the detected features [10]. Besides, the contrast ratio between the screw head and other connected components is low, and there is less information about deeper features. In addition, imperfect manufacturing processes and daily use under different conditions can easily produce micro defects on the screw surface, affecting the identification of the screw.

Considering the challenge of detecting screws, the template matching approach allows detecting matches between the template database and the input images [11,12]. However, the screw conditions can vary from the original design due to various usage conditions and part deterioration over time. Therefore, a fixed template cannot detect the dynamic state of the screws [13]. Moreover, the miniaturized design of electronics and overlapping objects makes screw detection challenging. With the development of deep learning, multiple object detection algorithms, such as the faster region-based convolutional neural networks (R-CNN), are used to detect screws [14–17]. To achieve automatic disassembly, the detection process should be efficient, as the speed and accuracy of the detection directly determine the subsequent disassembly operation.

When it comes to robotic disassembly, it is also necessary to determine the relevant tools needed in each disassembly step. Recent studies have highlighted the importance of automatic screw detection and equipping industrial robots with computer vision techniques to facilitate nondestructive dismantling [18]. In this work, we further elaborate on the importance of screw detection as a step toward developing tool recommendation guidelines. Corresponding to the screw detection, unscrewing tools should be supported in time [19]. With the need for autonomous decision-making tools, the tool recommendation function allows the robot to perform disassembly tasks successfully.

To facilitate robotic-assisted disassembly, this paper proposes a framework for screw detection in EOL electronics using YOLOv4 (You Only Look Once) and disassembly tools classification using

[1]Corresponding author.
Manuscript received July 26, 2022; final manuscript received October 21, 2022; published online December 2, 2022. Assoc. Editor: Cheryl Xu.

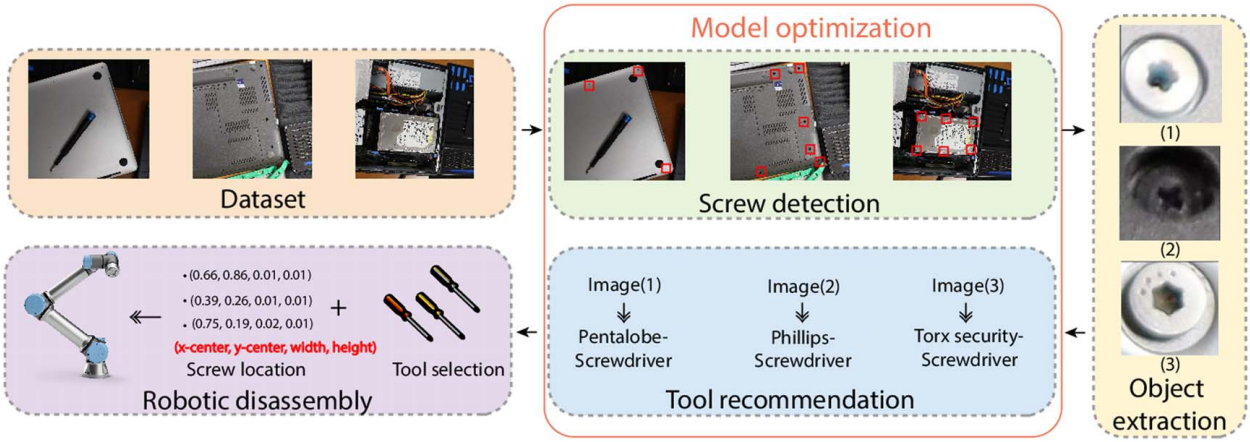

Fig. 1 Automatic screw detection and tool recommendation framework for robotic disassembly

EfficientNetv2. The paper is organized as follows: Sec. 2 provides an overview of the proposed automated framework. Section 3 presents the methodology consisting of deep learning and machine learning algorithms. The results and analysis are presented in Sec. 4, and Sec. 5 summarizes the conclusion and future work.

2 The Framework Overview

There are several challenges with robotic disassembling screws in EOL electronics. First, the size of screws is usually smaller than other components, requiring higher accuracy of position information. Second, the screws on EOL products may be deformed or damaged, making them missing or difficult to identify. Third, although screw heads look similar, only the specific tools that match the screws can remove them.

Therefore, we propose an image-based robotic screw disassembly framework as shown in Fig. 1. First, a YOLOv4 network is fine-tuned to achieve accurate detection of screws in consumer electronics. After obtaining the positioning coordinates of the screws, an image extractor is added to separate the screws from the EOL electronics to further refine the screw head features. Then, the EfficientNetv2 network was employed for screw classification and recommendation of disassembly tools. Several optimization strategies are applied, including modifying the detection algorithm and eliminating the sensitivity of the tool recommendation model to the detection prediction results, which could better inform the robot and prepare it for disassembly operations.

The proposed framework consists of two main modules. The target detection and target classification modules should be separated. The architecture from target detection should have the ability to detect small objects and provide accurate target locations. In addition, the detection architecture should not consider screw types to avoid the computational load. After the detection process, the targets can be completely separated from the EOL products by positioning information. Then, since disassembly tools depend on the screw head types, it is helpful to classify the screw heads. The classification module can focus directly on the features of the screw head and use the classification results to guide users on the relative screwdrivers.

3 Methodology

3.1 Object Detection Model Based on YOLOv4 Algorithm. Object detection refers to a series of tasks aimed at locating objects in images or videos. The purpose of object detection is to identify what an object is and where it is located by building computational models. In applying object detection in robotic disassembly, the first step for robots is to determine targets and their positioning information. There are two types of deep learning-based object detection algorithms: two-stage detectors, such as the R-CNN series, and one-stage detectors, such as the YOLO series. The two-stage detectors which are developed earlier first generate bounding box candidates and then implement object localization and classification. In subsequent developments, one-stage detectors omit the step of generating region candidates and are presented as a single convolutional network that simultaneously predicts bounding box localization and classification. The one-stage detectors are faster, while the accuracy of two-stage detectors is higher. However, starting from YOLOv3, YOLO series has achieved a better tradeoff between localization and recognition accuracy and speed [20].

In the present paper, we select the YOLOv4 algorithm as an object detection model to realize the task of detecting screws from EOL products. Compared with YOLOv3, YOLOv4 uses path aggregation network (PANet) as a parameter aggregation method for different detector levels instead of the feature pyramid network (FPN) used in YOLOv3, so the detection accuracy of YOLOv4 is higher than YOLOv3 in the MS COCO dataset [21,22]. Although YOLOv5 has recently been released, the YOLOv5 benchmark is not standardized yet, and more comparisons are needed. Furthermore, YOLOv4 is a good choice for adding more custom configurations than YOLOv5, which facilitates training and tuning the detection architecture on custom datasets.

YOLOv4 is a one-stage detector composed of CSPDarknet53 network (backbone), spatial pyramid pooling (SPP) layer, PANet, and three YOLO heads. The structure of YOLOv4 and its feature size corresponding to our dataset are shown in Fig. 2. The specific feature size parameters are useful for understanding how custom datasets work in the architecture. The detailed workflows of each module of YOLOv4 mainly include the following. First, CSPDarknet53, as the backbone of YOLOv4, is responsible for extracting deep features of the input image [12]. This convolution neural network consists of five residual blocks; each block contains convolution layers with 1×1 and 3×3 sizes and a Mish activation function. Second, the SPP layer participates in the convolution of the last feature layer of CSPdarknet53, followed by a maximum pooling with three kernel sizes of 5×5, 9×9, and 13×13 [14]. The SPP can increase the perceptual field and enhance feature extraction by separating the most important features from the backbone. Third, PANet adopts a bottom-up path, reducing the difficulty of extracting precise localization information. After shortening the information path between lower layers and topmost features, it is easier for the feature pyramid and solid localization features existing in the lower layer to propagate to the top [23]. Finally, three YOLO heads with sizes of 10×10, 20×20, and 40×40 are deployed to complete detection. The input of the heads contains rich semantic and spatial information from previous modules,

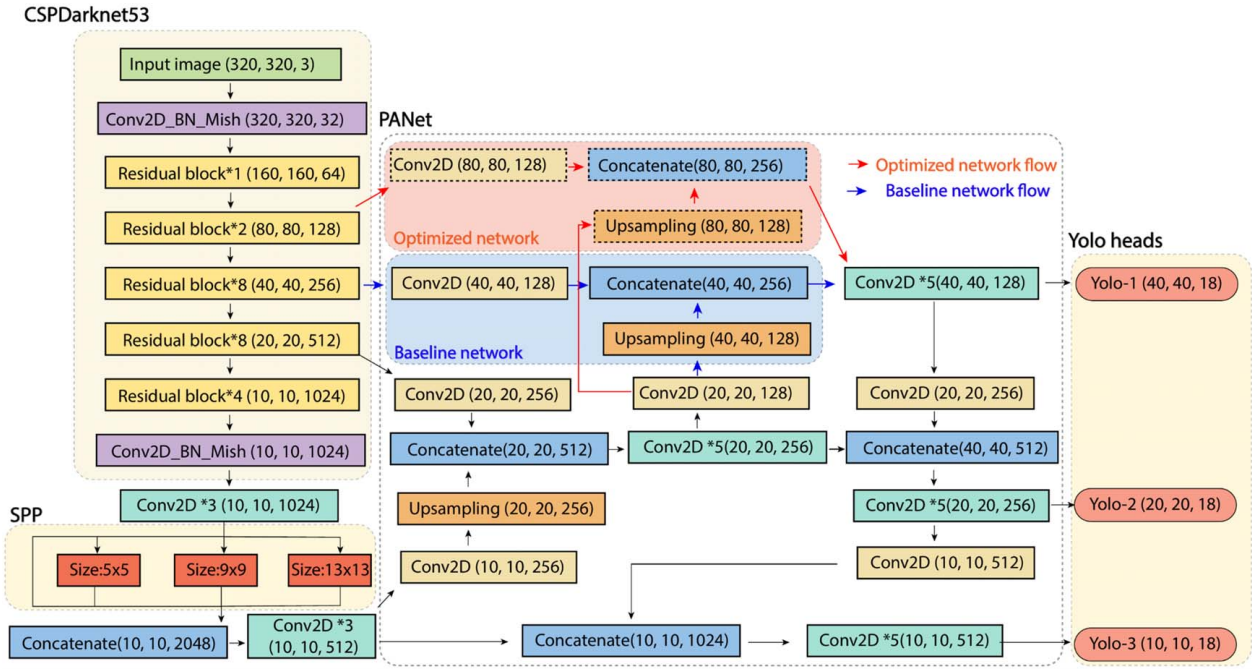

Fig. 2 Object detection model based on YOLOv4 algorithm

which guarantees a better performance for small target detection in complex backgrounds.

Although YOLOv4 performs satisfactorily in object detection, it is still not optimized for all scenarios. This affects the detection of small objects, as the average resolution of screws in the dataset is below 2×2 pixels. To improve the accuracy, we modify the baseline network with two adjustments. First, the network is optimized by using the low-level semantic information of the backbone. The convolutional layers of the backbone are rich in semantic information on the low-level feature maps and rich in spatial information on the high-level feature maps. The low-level layers in the backbone retain the high semantic values, which is partially beneficial for extracting the semantic information of small objects. The second adjustment is to prune the unimportant connections in PANet. Too frequent convolution reduces the spatial dimension and resolution, which is detrimental to detecting small objects. Since screws are the only target in our study, we can reduce the convolution operations. The difference between the baseline network and the optimized network is shown in Fig. 2. While recent work has already shown the application of YOLOv5s for screw detection [18], our focus on the adjustment and detailed optimization of the YOLO structure facilitates the adaptation of such algorithms to other small-sized objects not necessarily limited to screws.

3.2 Tools Recommendation Model Based on EfficientNetv2 Algorithm. The next stage in the proposed system is to utilize the results of the screw detection process and classify the screw types and further recommend a disassembly tool. It also fine-tunes the screw detection results by identifying false detections. The tool recommendation system consists of a screw image extraction mechanism and an image classifier. The two stages of the proposed framework are shown in Fig. 3.

Screws are inherently small relative to the input images. A single screw could occupy as low as 0.02% of the image area. To alleviate this disadvantage, we propose the following screw extractor. Leveraging the bounding boxes predicted by the YOLO network, the screw extractor can extract square regions each containing a single screw from the input image at full resolution.

Since YOLOv4's bounding boxes are rectangular, the region is first modified using a square bounding box with an equivalent area and centroid to the original one. Additionally, before extraction, the bounding box area is increased by a ratio, typically 20% of the original box area. This tolerance value is recommended as best practice in most settings. It can approximately accommodate the maximum off-center shift possible (one-sided shift of ~5%) for an intersection-over-union (IoU) of 0.9 based on square true and predicted boxes. This value can be adjusted for any unforeseen circumstances such as a sudden decrease in performance.

The extracted screw image is directly fed into a pretrained image classifier based on EfficientNetV2 [24]. EfficientNet is a novel architecture scaling convolutional neural networks to achieve decent classification performance and training efficiency. Specifically, the EfficientNetV2 network is an upgrade to the original EfficientNets [25] through utilizing the Fused-MBConv operator, applying training-aware neural architecture search toward parameter efficiency, and using a progressive learning strategy to speed up training. These upgrades focus on training efficiency while

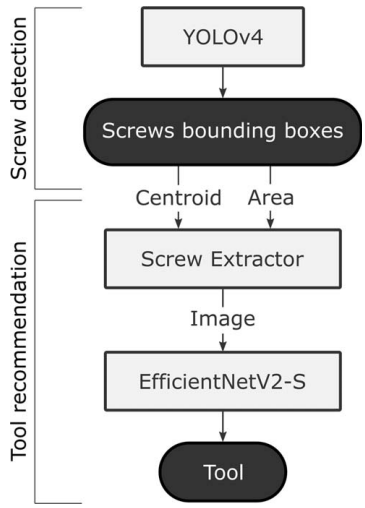

Fig. 3 The proposed framework for screw detection and tool recommendation

improving classification accuracy making it more adaptable to real-time robotic disassembly. As validation, EfficientNetV2 achieved competitive results on ImageNet [26] and CIFAR datasets while maintaining a relatively low number of parameters. In this framework, we use the EfficientNetV2-S variant, which has approximately 22 million parameters making it suitable for real-time application. The architecture of the EfficientNetV2-S is shown in Fig. 4.

The building blocks of the EfficientNetV2 network are the MBConv operator and its modified variant, the Fused-MBConv operator. MBConv, first introduced in MobileNetV2 [27] follows the concept of inverted residuals. It starts with narrow layers, expands them through a regular 1×1 convolution, applies a 3×3 depth-wise convolution, performs a squeeze-and-excitation operation [28], then squeezes the layers back to the original depth through another 1×1 convolution. The skip connections exist between the two starting and ending narrow layers, in contrast to regular residual blocks. The Fused-MBConv replaces the expansion and depth-wise convolution layers with a single regular 3×3 convolution as it was found that it boosts the training speed in the model's early stages.

Using transfer learning [29], the network is first pretrained with the ImageNet dataset and then fine-tuned to the screw images dataset. The benefit of using transfer learning is that they provide general model parameters which can be used in other deep learning applications. While ImageNet only contains over a thousand screw images, it encompasses almost 1.3 million images and 1000 classes, including various animals, plants, vehicles, and other objects. The images and classes are generic enough to provide a good start for the model parameters. In the fine-tuning phase, the network's final SoftMax layer is replaced with another corresponding to the number of screw types available in the fine-tuning dataset. Thus, the network can be trained by using a limited number of screw images by improving the network's parameter initialization procedure.

Besides classifying the screw types, two main aspects are desired from the tool recommendation system. First, it should identify false screw detections to fine-tune the YOLOv4 detection results. Second, the model classification predictions should be robust to deviated bounding boxes predicted by YOLOv4. We implement these aspects by (1) adding a "none" class which allows the model to flag false positives resulting from the screw detection

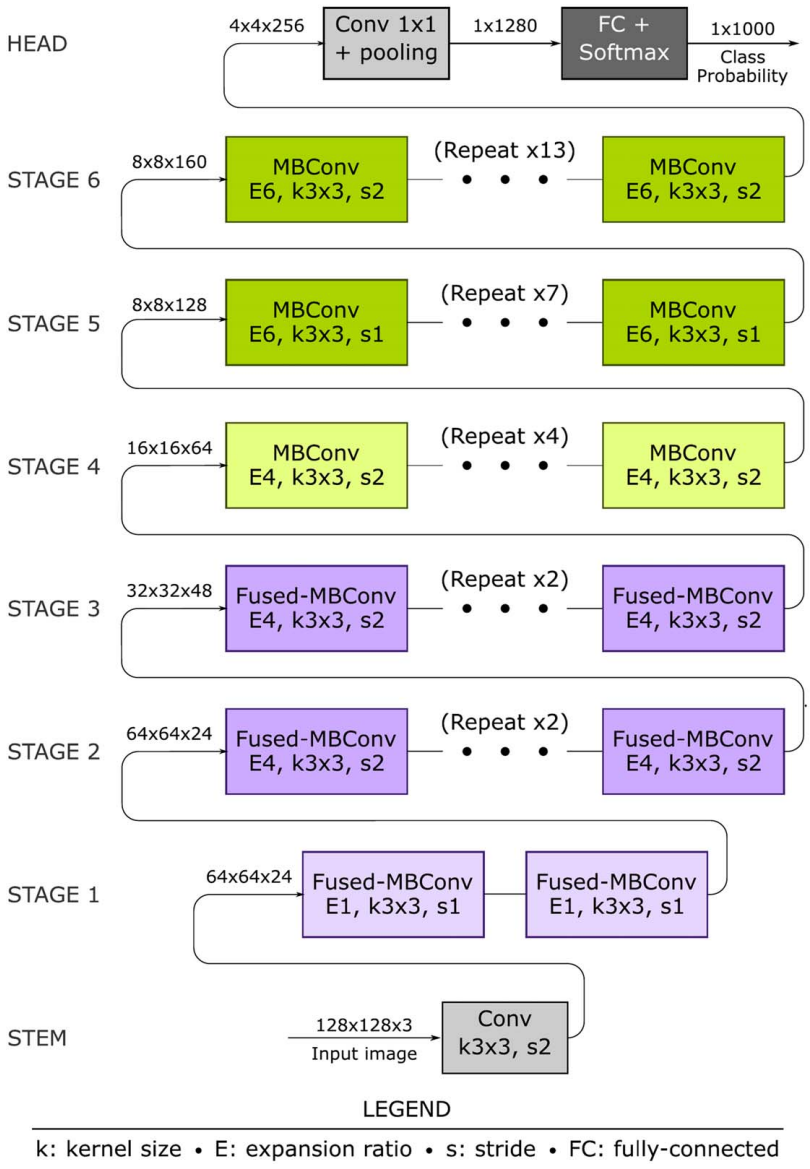

Fig. 4 Tool recommendation model based on EfficientNetV2 algorithm

Fig. 5 Data augmentation methods (hue = 0.1, exposure = 1.5, saturation = 1.5)

model and (2) implementing data augmentation techniques in model training that simulate the deviated bounding box predictions.

4 Experiments and Results

4.1 Experimental Environment and Dataset. To start training the models, we have taken photos of screws in different environments designed to be similar to a real remanufacturing workstation. To enhance the accuracy of the experiment and the complexity of detection, the screw photos should be as close as possible to the remanufacturing environment. Therefore, we created an experimental workstation with the EOL electronics, a disassembly toolkit, and multiple target-independent components to resemble a remanufacturing workstation.

The used electronics are supplied by the UF surplus office. The condition of the screws, such as damaged, tilted, twisted, or bent, will not be reported in advance. The information about the screws will be learned entirely by the model such that the trained model can be practically used in future remanufacturing scenarios without significant adjustment. A Canon EOS M200 camera equipped with a 15–45 mm lens has been used to take photos from any angle at 30 cm above the electronics.

The dataset includes three types of screws, Torx security screws on the desktop hard drive, Phillips screws on the back cover of a Dell laptop, and Pentalobe screws on the back cover of a Mac laptop. Three types of screws are available in different sizes, quantities, colors, and textures. A total of 300 images are recorded at a resolution of 4000×6000 pixels and divided into an 80% training set and a 20% test set. The training dataset is then manually annotated by LabelImg [30], a graphical image annotation tool. The rectangular bounding boxes only contain target objects and are labeled as a single class named "screw," which are stored as .txt files in YOLO format. Specifically, the object coordinates are the x–y coordinates of the center of each bounding box relative to the width and height of each image.

4.2 Object Detection Model Training and Evaluation Metrics. We have implemented the transfer learning technique to use the pretrained weights for YOLOv4. Beforehand training, YOLOv4 network adjusts images to a square format with 320×320 pixel resolution, where the cropped images do not maintain the aspect ratio of the original photos. During training, the network resizes input and output sizes for every ten iterations. Although the different images increase the complexity of the dataset, four data augmentation techniques amended in the network, including hue, exposure, saturation, and mosaic, are used to overcome the deficiency of the small dataset. The mosaic data augmentation is first introduced in YOLOv4. It combines four images into one single image, which leads to identifying objects on a smaller scale during the training process. Figure 5 visualizes the effect of employing the four data augmentation techniques on an original image.

A GPU NVIDIA GeForce RTX 3060Ti is used for training. The training hyperparameters are set as follows. The batch size is 32, which means 32 images are used in one iteration. The maximum training batches are initially set to 2000, suggesting the network terminates the training at 2000 batches. The adaptive moment estimation (Adam) optimizer in YOLOv4 is used for iterative parameter updates and fast training convergence. The initial learning rate is 0.001 and decays as training proceeds. The momentum is 0.949, which means the previous update strongly influences the current network update. The weight decay of 0.0005 is added as a regularization for the decreasing weight values. For anchor boxes, the number of anchors is determined by the pretrained YOLOv4 weights.

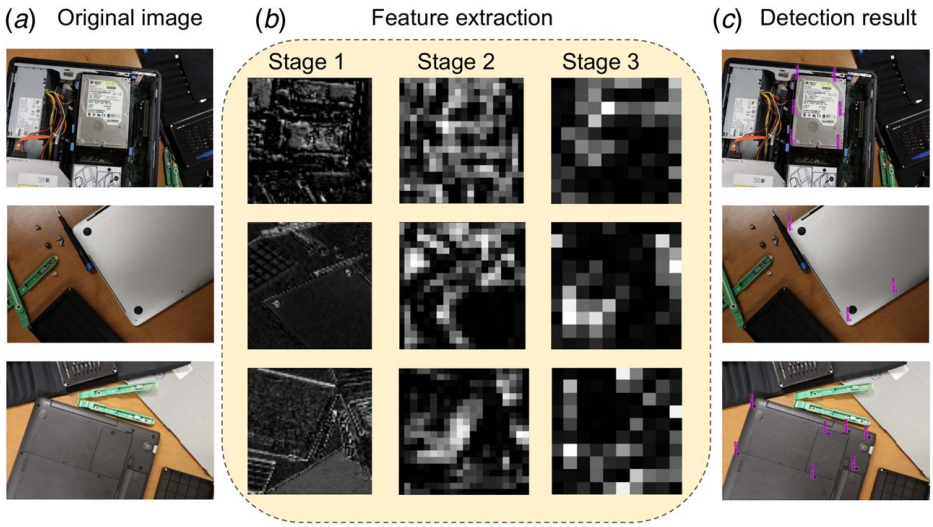

Fig. 6 Detection process

Table 1 Detection results of two models

	Precision	Recall	F_1 score	mAP
Baseline model	0.92	0.92	0.92	0.9132
Optimized model	0.94	0.95	0.94	0.9424

To evaluate the performance of the object detection model, *precision*, *recall*, F_1 *score*, and *mean Average Precision* (*mAP*) have been used as the evaluation indicators listed below. The IoU threshold of 0.5 is also set to classify whether the prediction is a true positive or a false positive.

4.3 Tool Recommendation Model Training and Evaluation Metrics. The dataset for training the tool recommendation model is built by extracting images of all screws in the original dataset using the actual labels and bounding boxes. For each screw, the extracted image has the same centroid of its corresponding bounding box and an increased area (e.g., 20%). The screw extractor also resizes the square images to 128×128 resolution. The dataset has three screw types: Pentalobe, Phillips, and Torx security. The total size of the result dataset is 1240 screw images which are split into 1,115 images for backpropagation and the remaining 125 for validation. Additionally, the images reserved for evaluating the screw detection model are also used for testing the tool recommendation system.

The EfficientNetV2-S variant is initially pretrained for the ImageNet dataset for the image classification model. Then, the input dimensionality was adjusted for images of size 128 and replaced the network's fully connected SoftMax layer with three nodes (or four), each representing a class probability. The model is then fine-tuned to the screw images using a categorical cross-entropy loss function with a batch size of five images. This fine-tuning process is optimized using an Adam optimizer [31] with a learning rate of 1E−5 and the exponential decay rates for the first and second-moment estimates of 0.9 and 0.999, respectively. In addition, the validation loss is monitored at each epoch through an early stopping criterion.

To evaluate the model performance, we use the F1-score, which is the harmonic mean of the recall and precision, representing a tradeoff between the two metrics.

4.4 Main Results. After training, the YOLOv4 model is used on the testing dataset. The detection results are shown in Fig. 6. The illustration of the feature extraction process leads to how YOLOv4 learns target features from big, middle, and small scales. Final detection results consist of object categories, bounding boxes, and detection accuracy.

After training the two models separately, the test results are listed in Table 1. Compared with the baseline model, the optimized model has better detection results in terms of evaluation metrics.

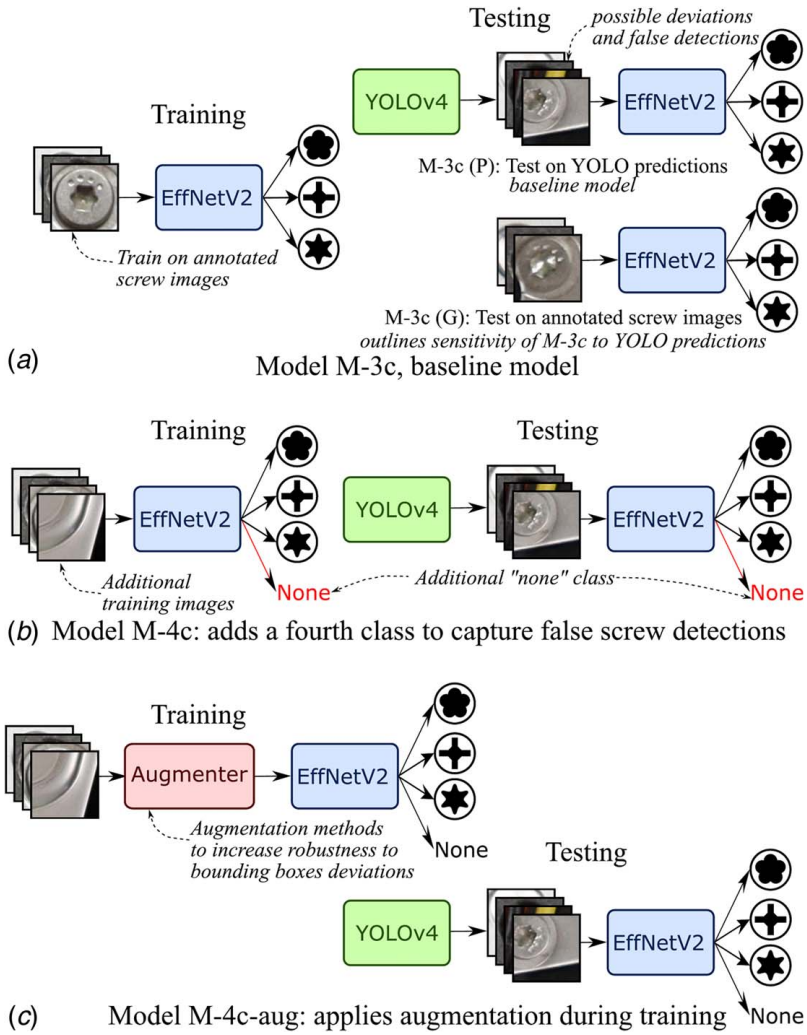

Fig. 7 Overview of tool recommendation models: (*a*) M-3c, (*b*) M-4c, and (*c*) M-4c-aug

Table 2 Test results of the tool recommendation models

Class	M-3c (G)	M-3c (P)	M-4c (P)	M-4c-aug (P)
Pentalobe	97.44%	96.72%	95.94%	99.17%
Phillips	99.24%	98.50%	99.62%	100.0%
Torx security	98.76%	98.33%	92.17%	99.59%
None	—	—	88.24%	98.36%
Average	98.48%	97.83%	93.99%	99.28%

Note: (G) and (P) indicate the usage of ground-truth and the predicted screw testing datasets, respectively.

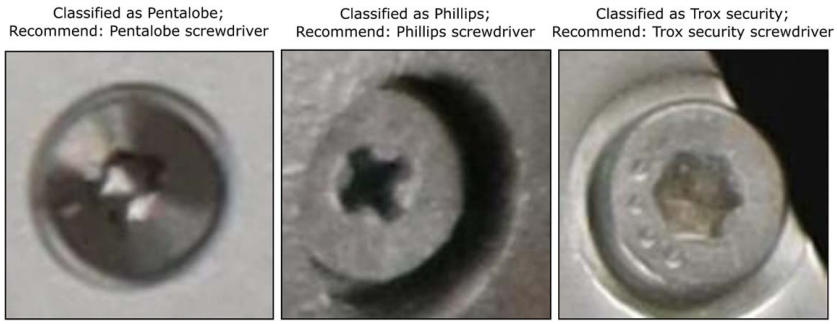

Fig. 8 Screw-type identification and tool recommendation results

Three models have been trained, tested, and compared for the tool recommendation model. The first model (M-3c) is tasked only with screw-type classification. The second model (M-4c) has an additional task of fine-tuning the screw detection results by identifying false detections. Finally, the third model (M-4c-aug) adds data augmentation techniques to account for deviations in screw bounding boxes predictions. A summary of these models is shown in Fig. 7. For the M-3c model, a single model is trained, but two tests are made. One is based on ground-truth screw detection and bounding boxes, and the other is based on detected screws and their bounding box predictions from the YOLOv4 model. The M-3c model's primary goal is to predict the screw type out of the available three classes and thus has three nodes at its SoftMax layer. The model is fine-tuned to the 1240 screw image dataset and converged after 83 epochs with a validation loss value of 2.4E−4. Note that the training dataset for this model is perfectly centered screw images with no false screws within the dataset as they purely rely on a manually labeled dataset.

The ground-truth testing dataset contains a total of 311 screw images, while the screw detection prediction dataset contains 314 screw images. The class F1 scores of M-3c for both datasets are presented in Table 2.

The average F1-scores for the ground-truth and predicted screws datasets are 98.48% and 97.83%, respectively. These results show good performance achieved by the M-3c model for both datasets.

There is a slight drop in all F1 scores in the predicted screw datasets due to possible bounding box off-center deviations, which makes the test set marginally different from what the model was trained on. The screw detection algorithm successfully avoided false screw detections in this dataset. However, M-3c will assign a screw-type to the empty bounding box if it happens during operation, resulting in an error. This is due to the nature of the three-class training dataset described earlier and because the model was built to predict the screw type out of the three given types. We label the false detections as "None," meaning that the network detected the screw, but no screw appeared in the image.

The second model, M-4c, is trained to detect false alarms, thus addressing one of the issues of M-3c and fine-tuning the screw detection model. This is done by introducing an additional 240 images to the training and 60 images to the testing dataset representing the false detections. These images are cropped at random locations with a fixed small window from the original components dataset such that they include no screws. We also replace the three-node SoftMax layer at the head with a four-node SoftMax layer, thus introducing a fourth class labeled "none." The M-4c model converged after 69 epochs with a validation loss value of 8.47E−4. While the model successfully identified the false screw detections, it achieved an average F1-score of 93.99% (Table 2). Nevertheless, the average F1 score of the M-4c(P) model is lower than that of the M-3c(P) model. Although the newly introduced images are labeled as false detections, some slightly deviated bounding boxes are still mainly of Torx security type, so the limited false detection data confuse the M-4c model to distinguish class None and class Torx security.

Finally, M-4c-aug introduces training data augmentation techniques, including random rotation, horizontal shift, vertical shift, and zoom. The main goal is to imitate possible bounding box deviations caused by the screw detection model. Thus, cases that are not present in the training data, such as partially visible screws and off-center screws, can be learned during the training process. This, however, made the training process lengthier, and the model converged after 155 epochs with a validation loss of 1.5E−4. This model addressed the issues of the other proposed models and achieved an average F1-score of 99.28% and improved class F1-scores, as shown in Table 2. Only two screws (out of 374) were mislabeled in this model. With the tool recommendation models, the final screw-type identification is shown in Fig. 8.

The case study outcomes show the application of object detection techniques for detecting small objects in consumer electronics recycling. The combination of computer vision algorithms can help robots recognize tiny objects and identify the tools required for disassembly based on the type of screws and fasteners used in designing electronics.

5 Conclusion and Future Work

This study proposes a framework for detecting small objects and recommending tools for robotic-assisted disassembly. The proposed framework consists of a YOLOv4-based screw detection and an EfficientNetv2-based tool recommendation. The modified

YOLOv4 algorithm can improve the accuracy of screw detection; the screw detection coordinates help crop and extract only screw images; the three models based on EfficientNetv2 can eliminate the effect of detection errors on screw classification. The application of the proposed work is demonstrated on a dataset of three different types of screws commonly used in consumer electronics. The proposed work opens the opportunity for better design of remanufacturing workflows to facilitate robotic disassembly and human–robot collaboration for waste stream management.

The proposed framework can be extended in several ways. First, the current complex electronics design and multilayer disassembly make it challenging to detect screws hidden in deeper spaces or covered by other components, so more efficient algorithms are needed to detect overlapping objects. Second, the appropriate disassembly tool should match not only the screw type but also the screw size. A method for determining screw size from visual data is needed to ensure the tool meets the screw removal requirements. Third, to achieve fully automated robot disassembly, the proposed framework can be integrated with other sensor-based technologies to collect data in real-time data while feeding it to robot control and planning algorithms.

Acknowledgment

This material is based upon work supported by the National Science Foundation–USA under Grant Nos. 2026276 and 2026533. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

References

[1] Wang, X. V., and Wang, L., 2018, "Digital Twin-Based WEEE Recycling, Recovery and Remanufacturing in the Background of Industry 4.0," Int. J. Prod. Res., **57**(12), pp. 3892–3902.
[2] Wurster, M., Michel, M., May, M. C., Kuhnle, A., Stricker, N., and Lanza, G., 2022, "Modelling and Condition-Based Control of a Flexible and Hybrid Disassembly System With Manual and Autonomous Workstations Using Reinforcement Learning," J. Intell. Manuf., **33**(2), pp. 575–591.
[3] Poschmann, H., Brüggemann, H., and Goldmann, D., 2020, "Disassembly 4.0: A Review on Using Robotics in Disassembly Tasks As a Way of Automation," Chem. Ing. Tech., **92**(4), pp. 341–359.
[4] Palmieri, G., Marconi, M., Corinaldi, D., Germani, M., and Callegari, M., 2018, "Automated Disassembly of Electronic Components: Feasibility and Technical Implementation," Proceedings of the ASME Design Engineering Technical Conference, Quebec City, Quebec, Canada, Aug. 26–29, vol. 51791, p. V004T05A006.
[5] Marconi, M., Palmieri, G., Callegari, M., and Germani, M., 2019, "Feasibility Study and Design of an Automatic System for Electronic Components Disassembly," ASME J. Manuf. Sci. Eng., **141**(2), p. 021011.
[6] Liu, B., Xu, W., Liu, J., Yao, B., Zhou, Z., and Pham, D. T., 2019, "Human–Robot Collaboration for Disassembly Line Balancing Problem in Remanufacturing," ASME 2019 14th International Manufacturing Science and Engineering Conference, MSEC 2019, Erie, PA, June 10–14, vol. 58745, p. V001T02A037.
[7] Li, K., Liu, Q., Xu, W., Liu, J., Zhou, Z., and Feng, H., 2019, "Sequence Planning Considering Human Fatigue for Human–Robot Collaboration in Disassembly," Procedia CIRP, **83**, pp. 95–104.
[8] Torres, F., Gil, P., Puente, S. T., Pomares, J., and Aracil, R., 2004, "Automatic PC Disassembly for Component Recovery," Int. J. Adv. Manuf. Technol., **23**(1–2), pp. 39–46.
[9] Bdiwi, M., Rashid, A., and Putz, M., 2016, "Autonomous Disassembly of Electric Vehicle Motors Based on Robot Cognition," Proceedings of the IEEE International Conference on Robotics and Automation, Stockholm, Sweden, May 16–21, pp. 2500–2505.
[10] Yildiz, E., and Worgotter, F., 2019, "DCNN-Based Screw Detection for Automated Disassembly Processes," Proceedings—15th International Conference on Signal Image Technology and Internet Based Systems, SISITS 2019, Sorrento, Italy, Nov. 26–29, pp. 187–192.
[11] Jiang, Z., Zhao, L., Li, S., Jia, Y., and Liquan, Z., 2020, "Real-Time Object Detection Method Based on Improved YOLOv4-Tiny," arXxiv.
[12] Li, Y., Wang, H., Dang, L. M., Nguyen, T. N., Han, D., Lee, A., Jang, I., and Moon, H., 2020, "A Deep Learning-Based Hybrid Framework for Object Detection and Recognition in Autonomous Driving," IEEE Access, **8**, pp. 194228–194239.
[13] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C., 2018, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4510–4520.
[14] Jiang, J., Fu, X., Qin, R., Wang, X., and Ma, Z., 2021, "High-Speed Lightweight Ship Detection Algorithm Based on YOLO-V4 for Three-Channels RGB SAR Image," Remote Sensing, **13**(10), p. 1909.
[15] Liu, S., Qi, L., Qin, H., Shi, J., and Jia, J., "Path Aggregation Network for Instance Segmentation." https://github.com/. Accessed November 28, 2021.
[16] Tan, M., and Le, Q. v., 2021, "EfficientNetV2: Smaller Models and Faster Training," Proceedings of the 38 th International Conference on Machine Learning, Virtual, July 18–24.
[17] Tan, M., and Le, Q. v., 2019, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," PMLR, pp. 6105–6114. https://proceedings.mlr.press/v97/tan19a.html. Accessed November 27, 2021.
[18] Mangold, S., Steiner, C., Friedmann, M., and Fleischer, J., 2022, "Vision-Based Screw Head Detection for Automated Disassembly for Remanufacturing," Procedia CIRP, **105**, pp. 1–6.
[19] Wegener, K., Chen, W. H., Dietrich, F., Dröder, K., and Kara, S., 2015, "Robot Assisted Disassembly for the Recycling of Electric Vehicle Batteries," Procedia CIRP, **29**, pp. 716–721.
[20] Jiang, Z., Zhao, L., Li, S., Jia, Y., and Liquan, Z., 2020, "Real-Time Object Detection Method Based on Improved YOLOv4-Tiny," arXiv.
[21] Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M., 2020, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv.
[22] Nepal, U., and Eslamiat, H., 2022, "Comparing YOLOv3, YOLOv4 and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs," Sensors, **22**(2), p. 464.
[23] Liu, S., Qi, L., Qin, H., Shi, J., and Jia, J., 2018, "Path Aggregation Network for Instance Segmentation," Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, June 19–21, pp. 8759–8768.
[24] EfficientNetV2: Smaller Models and Faster Training. http://proceedings.mlr.press/v139/tan21a.html. Accessed March 5, 2022.
[25] Tan, M., and Le, Q., 2019, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," In Proceedings of International Conference on Machine Learning (ICML), pp. 6105–6114.
[26] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L., 2010, "ImageNet: A Large-Scale Hierarchical Image Database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, June 20–25, pp. 248–255.
[27] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L. C., 2018, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, June 19–21, pp. 4510–4520.
[28] Hu, J., Shen, L., and Sun, G., 2018, "Squeeze-and-Excitation Networks," pp. 7132–7141. arXiv.
[29] Pan, S. J., and Yang, Q., 2010, "A Survey on Transfer Learning," IEEE Trans. Knowl. Data Eng., **22**(10), pp. 1345–1359.
[30] Tzutalin, 2015, "LabelImg," Git code. https://github.com/tzutalin/labelImg.
[31] Kingma, D. P., and Ba, J. L., 2014, "Adam: A Method for Stochastic Optimization," Third International Conference on Learning Representations, ICLR 2015—Conference Track Proceedings, San Diego, CA, May 7–9, pp. 1–13.