


Article

Instance Segmentation Method Based on Improved Mask R-CNN for the Stacked Electronic Components

Zhixian Yang, Ruixia Dong * , Hao Xu and Jinan Gu *

School of Mechanical Engineering, Jiangsu University, Zhenjiang 212000, China; hiyoo@mail.ujs.edu.cn (Z.Y.); xhvincent18@163.com (H.X.)

* Correspondence: 2211803054@stmail.ujs.edu.cn (R.D.); gujinan@tsinghua.org.cn (J.G.)

Received: 10 May 2020; Accepted: 26 May 2020; Published: 27 May 2020



Abstract: Object-detection methods based on deep learning play an important role in achieving machine automation. In order to achieve fast and accurate autonomous detection of stacked electronic components, an instance segmentation method based on an improved Mask R-CNN algorithm was proposed. By optimizing the feature extraction network, the performance of Mask R-CNN was improved. A dataset of electronic components containing 1200 images (992×744 pixels) was developed, and four types of components were included. Experiments on the dataset showed the model was superior in speed while being more lightweight and more accurate. The speed of our model showed promising results, with twice that of Mask R-CNN. In addition, our model was 0.35 times the size of Mask R-CNN, and the average precision (AP) of our model was improved by about two points compared to Mask R-CNN.

Keywords: autonomous detection; electronic components; deep learning; instance segmentation; Mask R-CNN

1. Introduction

In the industrial assembly field, plug-in electronic components are often manually inserted due to their complicated shape and fragile nature. Because of the massive workload, low efficiency, and high cost, it is difficult to ensure the assembly quality. With the rapid development of the electronics industry, the higher demand for assembly speed and accuracy of plug-in electronic components is prioritized. Traditional manual assembly does not satisfy the development needs of the electronic industry anymore. Therefore, the automatic assembly of electronic components has become an inevitable trend.

Detecting the category and location of some objects is a necessary condition for automatic assembly. For human beings, recognizing and grabbing some specific objects in a stacked scenario is an intuitive behavior. However, for robots, finishing a series of motions smoothly, including identifying, locating, and grasping an object, is not an easy task. With the development of convolutional neural networks, object-detection methods based on deep learning have been greatly improved in speed and accuracy compared with traditional detection methods [1–5]. The self-adjusting ability of deep neural networks can effectively enhance the robots' autonomy in terms of object detection. Among all detection methods, instance segmentation can identify object contours at the pixel level and achieve higher location accuracy [6]. Differing from semantic segmentation, instance segmentation mostly focuses on the differences among the instances. In recent years, instance segmentation, as a critical technology of artificial intelligence, has been widely used in the medical field, the engineering field, and so on. He et al. [6] put forward the Mask R-CNN algorithm and used it for human pose estimation. The average precision of segmentation on the COCO dataset can reach 64.7%. Hang et al. [7] proposed a mammography quality detection and segmentation system based on the Mask R-CNN algorithm, which can effectively detect the quality of mammography without human intervention. Dai et al. [8]

implemented the segmentation of prostate and intraprostatic lesions based on Mask R-CNN, which is of considerable significance to help radiologists in clinical practice. Chiao et al. [9] utilized Mask R-CNN to segment the ultrasound breast images for lesion detection and diagnosis of benign and malignant, which provides a non-invasive method for breast-lesion detection. Furthermore, to analyze the environment inventory, Xu et al. [10] used the method of instance segmentation to segment trees from the urban scenes, while the accuracy of semantic labeling of trees reaches around 0.9. Bert De et al. [11] proposed a loss function with two terms to determine the entity to which the embedded pixel belongs by the intra-cluster pull and intra-cluster push forces. This method uses the pixel embedding to solve the problem of semantic instance segmentation at the pixel level and promotes the development of autonomous driving technology. These research works indicate that instance segmentation possesses the ability to generate a high-quality segmentation mask for each object.

In general, instance segmentation includes two kinds of methods: detection-based methods and segmentation-based methods. The detection-based methods focus on generating region proposals and predicted bounding boxes, then masking the objects in the predicted bounding boxes [12]. Hariharan et al. [13] proposed a method of simultaneous detection and segmentation (SDS). By using the R-CNN algorithm to extract features of each region, this method can generate a rough estimate of the mask based on the bounding boxes, combined with the region proposals, and eventually obtain a fine mask. To improve the accuracy of detection and segmentation, they further proposed a pixel descriptor called hypercolumn, which can calculate the vector of activations of all the convolutional layer pixels above a specified pixel. By embedding the pixel descriptor into the classifier, the mAP (mean average precision) raises from 50.3% to 56.5% on the PASCAL VOC 2012 verification set [14]. Dai et al. [15] replaced the pixel-category classifier in fully convolutional networks (FCN) [16] with the relative-position classifier of the pixel object instance, and the local correlation of the image was used to estimate the instance. On this basis, they designed a convolutional feature masking (CFM) method that extracts segmented features directly from feature maps instead of from original images [17]. The mAP increases from 56.5% to 61.5% on the PASCAL VOC 2012 validation set. To further improve accuracy, they divided the instance segmentation task into three sub-tasks: distinguishing instances, estimating masks, and classifying objects. After that, the multi-task network cascades (MNCs) [18] method was used to enhance the information flow among sub-tasks to accomplish fast and accurate instance-aware semantic segmentation. The mAP can reach 63.5% by using this method. Li et al. [19] put forward a fully convolutional instance-aware semantic segmentation (FCIS) instance segmentation method, which could make the mAP reach up to 65.7%. This method accomplished the detection and segmentation of the two sub-tasks by executing the inside score maps and outside score maps in parallel. The fully connected layers were replaced with a softmax classifier, thus reducing the possibility of overfitting. Based on FCIS, feature maps of different scales were used by Pham V Q et al. to generate score maps, which were fused with skip structure to produce segmentation results. Bayesian inference was put in to optimize the segmentation results, further increasing the mAP to 67.3% [20]. Another instance segmentation method is segmentation-based. Compared with the detection-based methods, the segmentation-based methods, on the contrary, first get a pixel-level segmentation map from an input image, then identify the object instances based on this segmentation map obtained [12]. Pinheiro [21] estimated the probability that an object is wholly contained in an image patch by using the object-proposal method, while the segmentation masks and correlation scores are generated simultaneously by giving the image input patch. Based on this approach, they further proposed a method called SharpMask [22]. Firstly, a coarse mask encoding was output in the feedforward process, and then a module which fuses the features extracted from the lower layer in the way of top-down was used to refine segmentation. To obtain the feature map with higher resolution, path enhancement was used in these references [12,23,24] to strengthen the information flow between network layers. The verification of these improved methods was performed on public datasets. These research works indicate that instance segmentation is an effective object-detection method, which can immensely improve the accuracy.

Compared with the bounding boxes of object detection, instance segmentation can get a more definite edge of an object. In the meantime, instance segmentation possesses better performance than semantic segmentation in labeling different instances among the same kind of objects. Generally speaking, the classification task is only to identify an image containing some objects, but when segmenting some instances, it will become a more complex procedure. Especially in a stacked scenario with multiple overlapping objects, it is necessary to not only classify diverse objects but also determine the boundaries, divergence, and relationships among all objects.

Nicholas et al. [25] used action primitives to touch multiple objects simultaneously. Object detection and grasp planning can be performed in a cluttered environment with multiple objects. Guo et al. [26] proposed a shared convolutional neural network that can detect target objects from stacked objects in real-time. Zhang et al. [27] put forward a multi-task convolutional neural network for automatic robotic grasping, focusing on object detection problems in the case of different object-stacking situations, which is suitable for grasping tasks in multi-object stacking scenarios. However, due to the small size of the electronic components, these multi-object detection methods are not available. In the case of stacked electronic components, accurately detecting their categories and locating their positions for the subsequent grasping operation has come to be an inevitable problem in need of urgent solutions. Instance segmentation can effectively detect all the objects from an input image, and at the same time, generate a high-quality segmentation mask for each instance to get a delicate position of the detected object [6].

Until now, the relative research on instance segmentation has been rarely published, especially in the assembly field of electronic components. The diversity of electronic components poses challenges in the investigation process. The key to the realization of autonomous detection of electronic components is to enhance the generalization ability of instance segmentation methods, such as the robustness of detection and recognition in complex scenarios, and the balance between samples. In this paper, an instance segmentation method based on an improved Mask R-CNN algorithm was proposed to detect stacked and occluded electronic components. The experiments were performed on a dataset of electronic components. The results were analyzed and discussed.

2. Dataset

2.1. Image Collection for the Dataset

Three kinds of datasets, which were training set, validation set, and testing set, were included in this study. Three hundred images (744×992 pixels) were collected for building the dataset. Four types of plug-in electronic components were included: tantalum capacitor, resistor, electrolytic capacitor, and potentiometer. Their shape features were representative in the electronic assembly field, as shown in Figure 1.

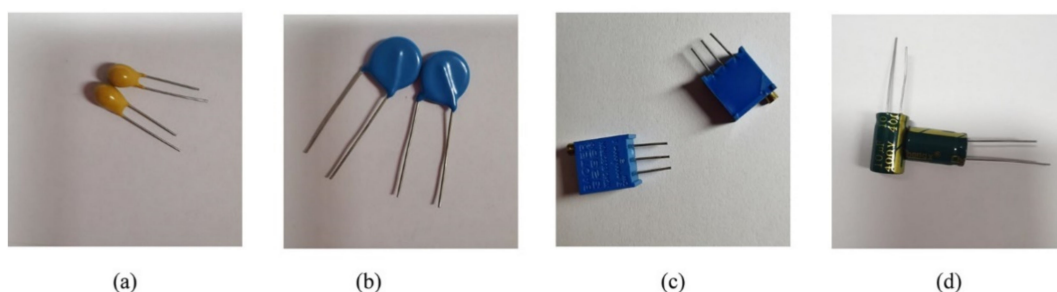


Figure 1. Plug-in electronic components used in the dataset. (a) Tantalum capacitor. (b) Resistance. (c) Potentiometer. (d) Electrolytic capacitor.

2.2. Data Augmentation

The number of images in the dataset was augmented to prevent overfitting, enhance the generalization ability, and improve the robustness of the model. In this work, 1200 images were obtained by using four data-augmentation methods, which were flipping, rotating, random cropping, and color jittering (see Figure 2). The number of images in the training set, validation set and testing set was 600, 240, and 360, respectively. The composition of the training set, validation set, and testing set is shown in Table 1.

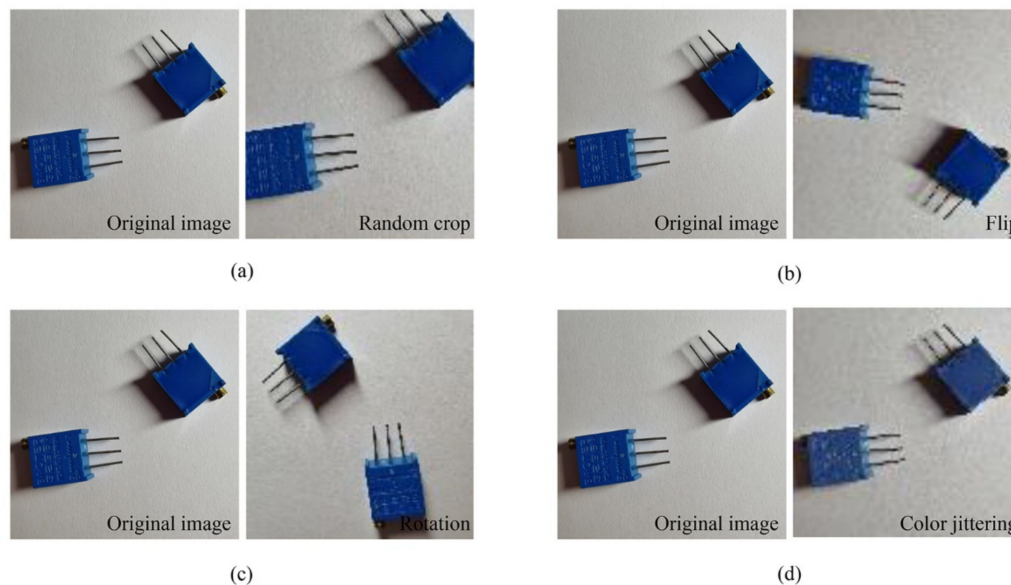


Figure 2. Four data-augmentation methods. (a) Random crop. (b) Flip. (c) Rotation. (d) Color jittering.

Table 1. The composition of training, validation, and testing sets.

Class	Training Set	Validation Set	Testing Set
	Number of Objects	Number of Objects	Number of Objects
Electrolytic Capacitor	365	120	204
Resistance	329	168	180
Tantalum Capacitor	340	130	220
Potentiometer	343	174	210

2.3. Image Annotation

Images were annotated with the VGG Image Annotator (VIA), which is an open-source annotation tool developed by the Visual Geometry Group. In the labelling process, a total of 2783 targets in 1200 images were labelled (examples of labelling can be seen in Figure 3). We used polygons from the VIA to label the region shapes. The region attribute was set to “electronic.” The identities of the four types of electronic components were 1, 2, 3, and 4, respectively, and the corresponding descriptions were “Capa,” “Resis,” “Tcapa,” and “Poten.” We show the corresponding labels for the four types of electronic components in Table 2.

Table 2. The corresponding labels for four types of electronic components.

Class	Identity (id)	Description
Electrolytic Capacitor	1	Capa
Resistance	2	Resis
Tantalum Capacitor	3	Tcapa
Potentiometer	4	Poten

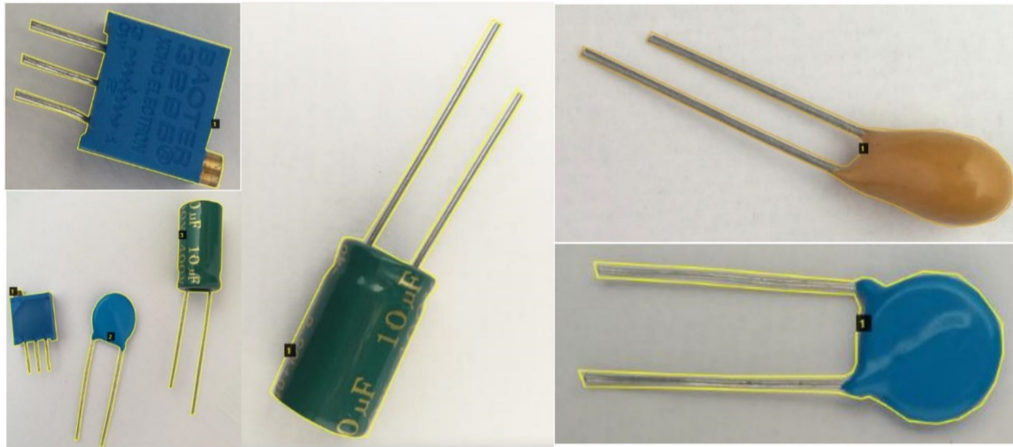


Figure 3. Examples of labelled images.

3. Structure

Mask R-CNN [6] is an extension of Faster R-CNN [1]. By adding a mask branch, masks can be generated to cover the objects based on the location and classes of the detected objects. As shown in Figure 4, Mask R-CNN is a two-stage architecture. In the first stage, the region proposal network (RPN) is used to generate object region proposals and determine the foreground and background of input images. In the second stage, convolutional neural network extracts features from candidate proposals, classifies the proposals, and generates bounding boxes and masks for possible objects. With the development of instance segmentation methods, the accuracy of detection is constantly improved. However, the improvement of accuracy does not necessarily make the network more advantageous in terms of speed and model size. As accuracy increases, the complexity and computational burden also increase. In many real-world applications such as automatic drive and robotics, detection tasks need to be performed in a timely manner on a computationally limited platform. Our goal was to optimize the network, reduce the calculation parameters, and speed up the detection while ensuring accuracy.

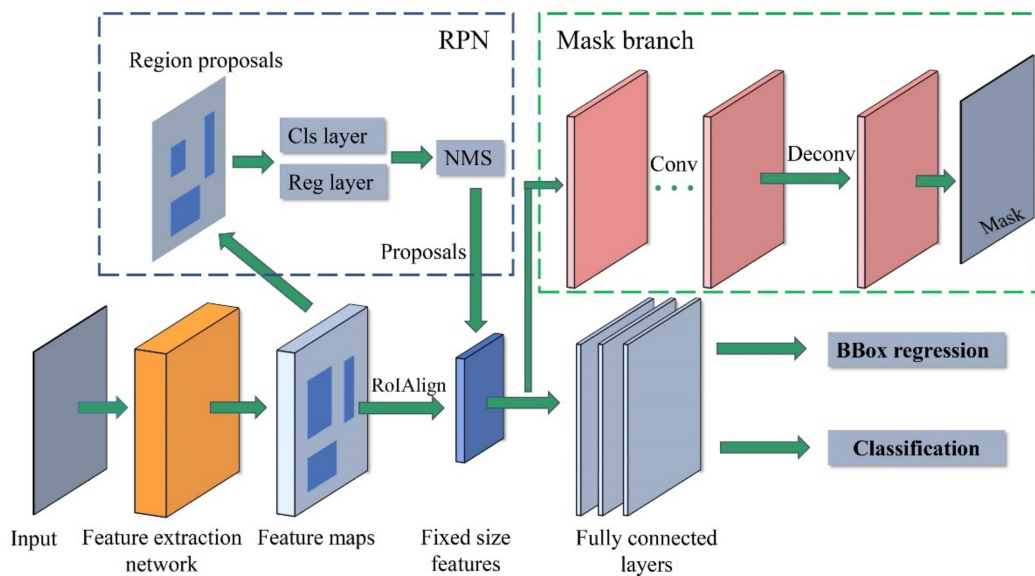


Figure 4. The schematic architecture of Mask R-CNN. “Cls layer” denotes classification layer, “Reg layer” denotes regression layer, “Conv” denotes convolution operation, and “Deconv” denotes deconvolution operation, “NMS” denotes non-maximum suppression, “BBox regression” denotes bounding box regression, “RoIAlign” denotes region of interest align. RPN: region proposal network.

3.1. Backbone

Two backbones are proposed as feature extractors in Mask R-CNN: deep residual networks (ResNets) [28] and feature pyramid networks (FPNs) [29]; each backbone corresponds to a mask head architecture. To make the network lightweight, we optimized the feature extraction network of Mask R-CNN. MobileNets [30], as one of the representatives of the lightweight neural network, can narrow a model, decrease the number of parameters, and improve the detection speed of a model while ensuring accuracy. Mask R-CNN is known for its high segmentation accuracy, and MobileNets can simplify the model and enhance the speed of detection while ensuring detection accuracy. In order to achieve a balance between accuracy and speed, we used MobileNets as part of the feature extractor of Mask R-CNN for the instance segmentation of electronic components.

The architecture of MobileNets (see Figure 5a) is based on depthwise separable convolution, which factorizes a standard convolution into a depthwise convolution and a pointwise convolution (see Figure 5b). The depthwise convolution uses a single convolution kernel to each input channel. The pointwise convolution uses a 1×1 convolution kernel to linearly combine the outputs of the depthwise convolution. Each depthwise convolution and pointwise convolution is followed by a batch normalization layer and the rectified linear unit (ReLU) [31] activation function. In addition, two hyperparameters are introduced by MobileNets: width multiplier and resolution multiplier. The width multiplier is used to control the number of channels for input and output, and the resolution multiplier is used to control the resolution of the input. The use of these two hyperparameters greatly reduces the computation load and expedites the speed of calculation.

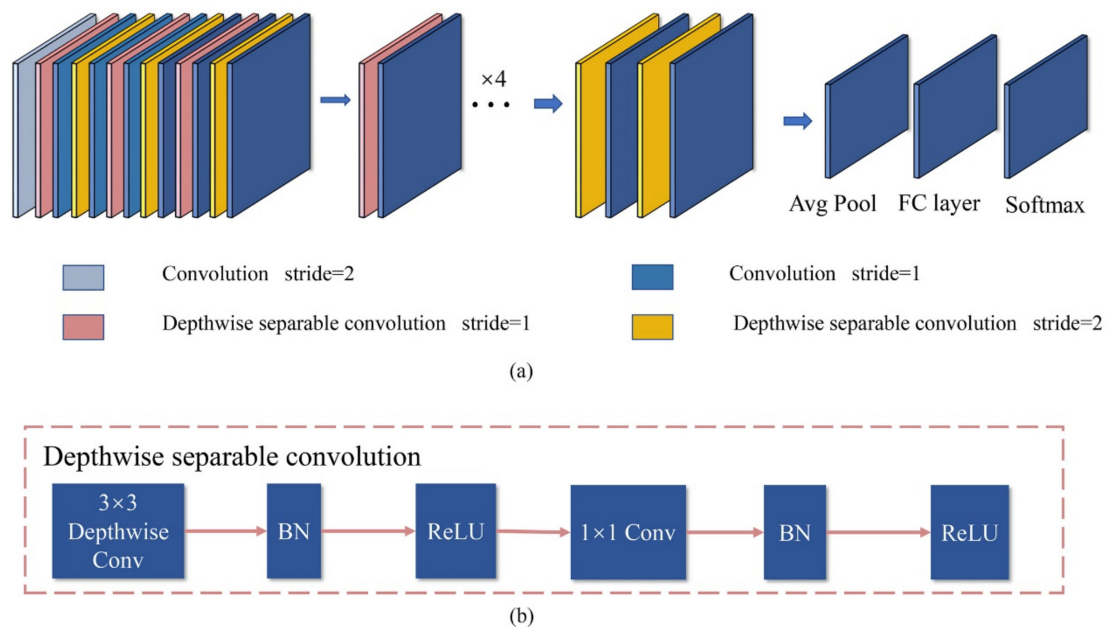


Figure 5. The structure of MobileNets and depthwise separable convolution. (a) The schematic architecture of MobileNets. (b) Depthwise separable convolution. “Avg Pool” denotes average pool, “FC layer” denotes fully connected layer, and “BN” denotes batch normalization. ReLU: rectified linear unit.

Multi-feature fusion aims to aggregate features of different resolutions. In FPNs, different levels of feature maps are efficiently fused through three ways of bottom-up, top-down, and lateral connection. It is worth noting that FPNs use not only deep but also shallow feature maps to extract features, which are very helpful for the detection of small objects like electronic components. Combining MobileNets and the FPN, we developed an improved Mask R-CNN, which consistently achieves better accuracy with much fewer parameters and faster speed than Mask R-CNN.

Firstly, the last average pooling layer, fully connected layer, and softmax layer of MobileNets are deleted, then the structure of MobileNets is divided into five stages (see Table 3). Stage 1 contains a standard convolution and a depthwise separable convolution. Both stage 2 and stage 3 include two depthwise separable convolutions. Six depthwise separable convolutions are included in stage 4, and two depthwise separable convolutions are included in stage 5. S1 to S5 represent the output of each stage of MobileNets, respectively. The feature fusion of MobileNets and the FPN is shown in Figure 6. The bottom feature layer obtains the same number of channels as the previous feature layer through 1×1 convolution. The upper feature layer gets the same length and width as the next feature layer through upsampling. To obtain a new fusion layer, add the length, width, and the number of channels; this fusion operation is shown in Figure 7. As a concrete example, the S4 layer gets the same number of channels as the FPN-P5 layer, and after upsampling, the length and width of the FPN-P5 layer are the same as that of the S4 layer. Finally, the two are added to get the fusion layer FPN-P4. Note that the FPN-P2 to FPN-P5 layers are used to predict the bounding boxes, position regression, and masks of objects, while the FPN-P2 to FPN-P6 layers are used to train the RPN, that is, the FPN-P6 layer is only used in the RPN.

Table 3. Five stages of MobileNets after modification [30].

Stage	Type	Filter Shape	Stride	Output
1	Conv	$3 \times 3 \times 3 \times 32$	2	S1
	Depthwise Conv	$3 \times 3 \times 32$	1	
	Conv	$1 \times 1 \times 32 \times 64$	1	
2	Depthwise Conv	$3 \times 3 \times 64$	2	S2
	Conv	$1 \times 1 \times 64 \times 128$	1	
	Depthwise Conv	$3 \times 3 \times 128$	1	
	Conv	$1 \times 1 \times 128 \times 128$	1	
3	Depthwise Conv	$3 \times 3 \times 128$	2	S3
	Conv	$1 \times 1 \times 128 \times 256$	1	
	Depthwise Conv	$3 \times 3 \times 256$	1	
	Conv	$1 \times 1 \times 256 \times 256$	1	
4	Depthwise Conv	$3 \times 3 \times 256$	2	S4
	Conv	$1 \times 1 \times 256 \times 512$	1	
	Depthwise Conv	$3 \times 3 \times 512$	1	
	Conv	$1 \times 1 \times 512 \times 512$	1	
5	Depthwise Conv	$3 \times 3 \times 512$	2	S5
	Conv	$1 \times 1 \times 512 \times 1024$	1	
	Depthwise Conv	$3 \times 3 \times 1024$	2	
	Conv	$1 \times 1 \times 1024 \times 1024$	1	

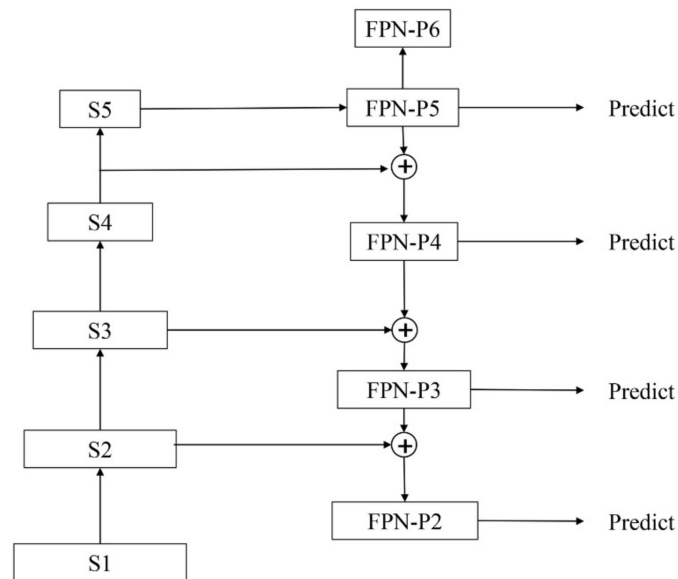


Figure 6. Data flow between MobileNets and the feature pyramid network (FPN).

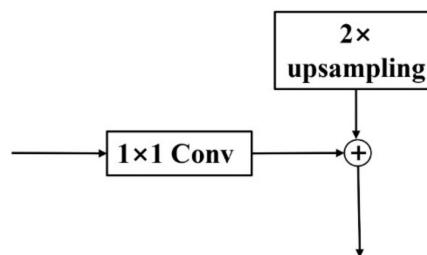


Figure 7. Feature fusion operation.

3.2. RPN

The RPN takes an image as input and outputs a set of rectangular object proposals, each with an objectness score [1]. It determines whether the anchor is the foreground or the background and performs the first coordinate correction for the anchors belonging to the foreground. The structure of the RPN is shown in Figure 8. The RPN uses sliding windows on shared convolutional feature maps to generate k object boxes ($k = 15$ in this paper) with a preset aspect ratio and a scale for each pixel, which are called anchor boxes. An anchor is centered at the sliding window in question and is associated with a scale and aspect ratio [1]. In Mask R-CNN, the number of region proposals fed to Region of Interest Align (RoIAlign) is very big, generally ranging from 100 to 300. In this case, the number of segmentation maps to be learned is large, which makes it difficult to extract features in the mask branch. To solve this problem, the threshold of non-maximum suppression (NMS) in the RPN is increased from 0.5 to 0.7, and the intersection over union (IoU) threshold for NMS is fixed at 0.7. The setting of anchors in Faster R-CNN contains three scales of anchor boxes, and each scale corresponds to three aspect ratios. In order to adapt to the size requirements of electronic components and obtain more precise region proposals, we used five scales with box areas of 32^2 , 64^2 , 128^2 , 256^2 , and 512^2 pixels, and three aspect ratios of 1:1, 1:2, and 2:1.

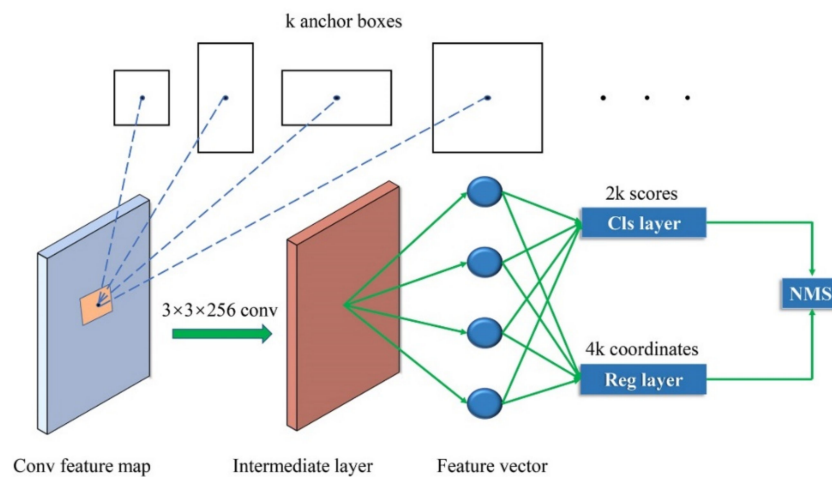


Figure 8. The structure of the RPN.

3.3. RoIAlign

Region of interest pooling (RoI-Pooling) is used to extract features from shared convolutional layers, and the features are input into fully connected layers for classification in Faster R-CNN [1]. Nearest-neighbor interpolation, which is a quantization operation, is used by RoI-Pooling when features are extracted from shared convolutional layers. Due to this quantization operation, the features corresponding to each RoI are converted into a fixed dimension, and the RoI of output feature maps after RoI-Pooling does not match the RoI of the input image. Different from RoI-Pooling, RoIAlign uses bilinear interpolation instead of nearest-neighbor interpolation to calculate the pixel value of each position and eliminate quantization operation. It firstly traverses region proposals and divides each region proposal into $k \times k$ units, leaving the boundaries of each unit unquantified. Then, the values of coordinates are calculated in each unit and the pixel values of positions are calculated by bilinear interpolation, and finally, the max-pooling operation is performed. The detection accuracy for small objects is more obvious owing to the elimination of the quantization operation.

3.4. Head Architecture

Two-head architectures are proposed in Mask R-CNN. We used one of them, as shown in Figure 9. In the mask branch, deconvolution operation [32] is used to increase the spatial dimension of the feature map. Finally, a mask of $28 \times 28 \times 80$ is output.

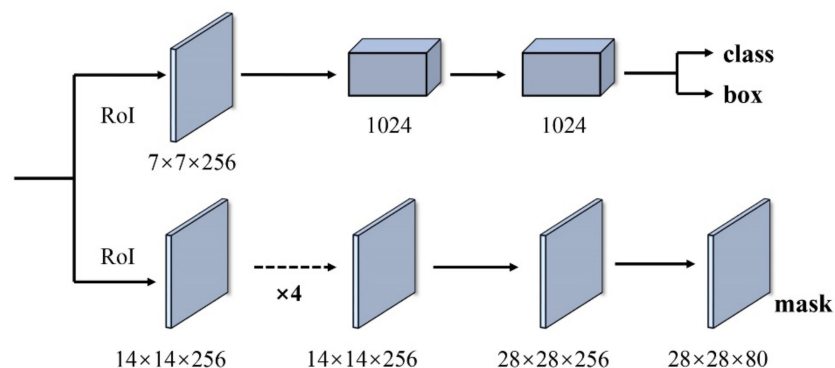


Figure 9. The head architecture we used in the improved Mask R-CNN [6]. RoI: region of interest.

3.5. Loss Function

Since a mask branch is added, the multi-task loss function of Mask R-CNN can be expressed as:

$$L_{final} = L_{RPN-clc} + L_{RPN-bbox} + L_{cls} + L_{bbox} + L_{mask}$$

where $L_{RPN-clc}$ is the classification loss function in the RPN, $L_{RPN-bbox}$ is the position regression loss function in the RPN, L_{cls} represents the classification loss function, L_{bbox} is the position regression loss function, and L_{mask} is defined as the average binary cross-entropy. The new mask branch is $k \times k \times m$ for each RoI output dimension, where $m \times m$ is the size of the mask, and k represents the number of classes, thus a total of k masks generated. After the predicted masks are obtained, a per-pixel sigmoid is used to classify the masks, and the obtained results are taken as one of the inputs of the L_{mask} . Note that only pixels that are considered foreground are used to calculate L_{mask} . The overall structure of the improved Mask R-CNN is shown in Figure 10.

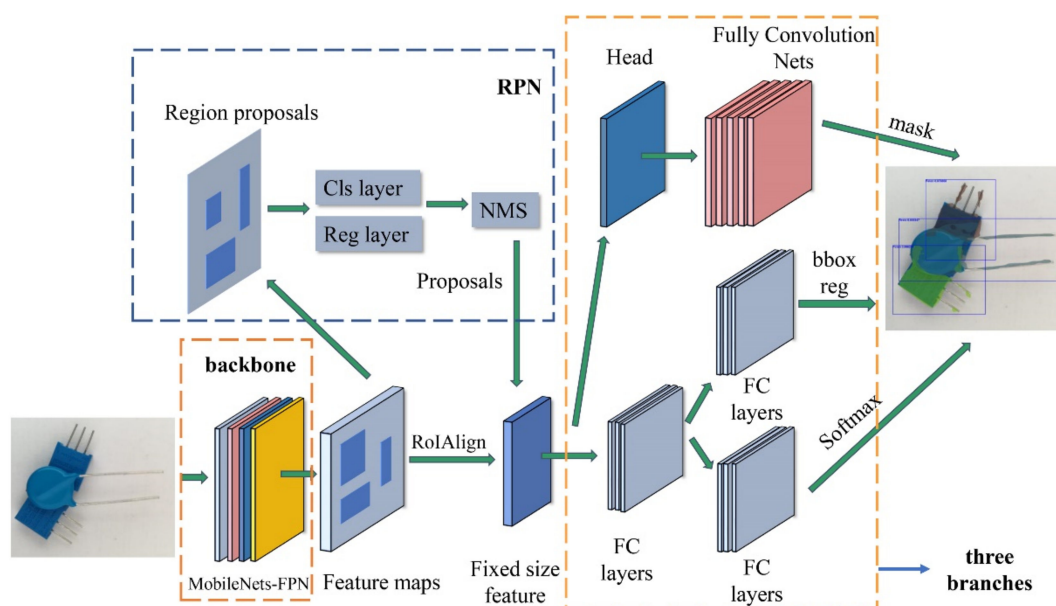


Figure 10. The overall structure of the improved Mask R-CNN. RoIAlign: Region of Interest Align.

4. Experiments and Discussion

4.1. Implementation Details

We used the open-source Mask R-CNN library to complete the experimental research. All experiments were performed on computers equipped with Intel Xeon(R) E5-1680 v4@3.40G Hz CPU and the Quadro M5000 graphics processing unit through Pycharm, CUDA 9.0 and CUDNN 9.0 realized.

We trained a total of 44 epochs with 200 steps each. We used a mini-batch size of 1 image per GPU and trained the model for 11k iterations, starting from a learning rate of 0.001. We used a weight decay of 0.0001 and a momentum of 0.9. It took four hours of training on a single 1-GPU machine under this setting.

The average precision (AP) is usually used to evaluate the performance of the object detector, and the precision/recall curve is summarized by calculating the area under the curve. For a given category, precision is used to account for the proportion of positive samples that are judged to be true, and recall is used to indicate the proportion of positive samples that are judged to be true in the classifier. The mAP is a performance metric for the algorithms that predict locations and categories of

objects, and it refers to the average of multiple classes of APs. In this paper, we used the standard COCO [33] metrics including AP, AP₅₀, and AP₇₅.

4.2. Training, Validation, and Test Results

We trained the improved network by using the weight file of COCO [33] in Mask R-CNN and evaluated its accuracy by using the testing set. The training time in GPU mode was four hours. The time required to evaluate an image of 744×992 pixels in GPU mode was 1.8 s. We recorded the APs of four types of electronic components. It can be seen from Figure 11 that the AP of tantalum was the highest, at 97.32%, and the AP of electrolytic capacitor, resistor, and potentiometer were 86.55%, 92.23%, and 96.36%, respectively.

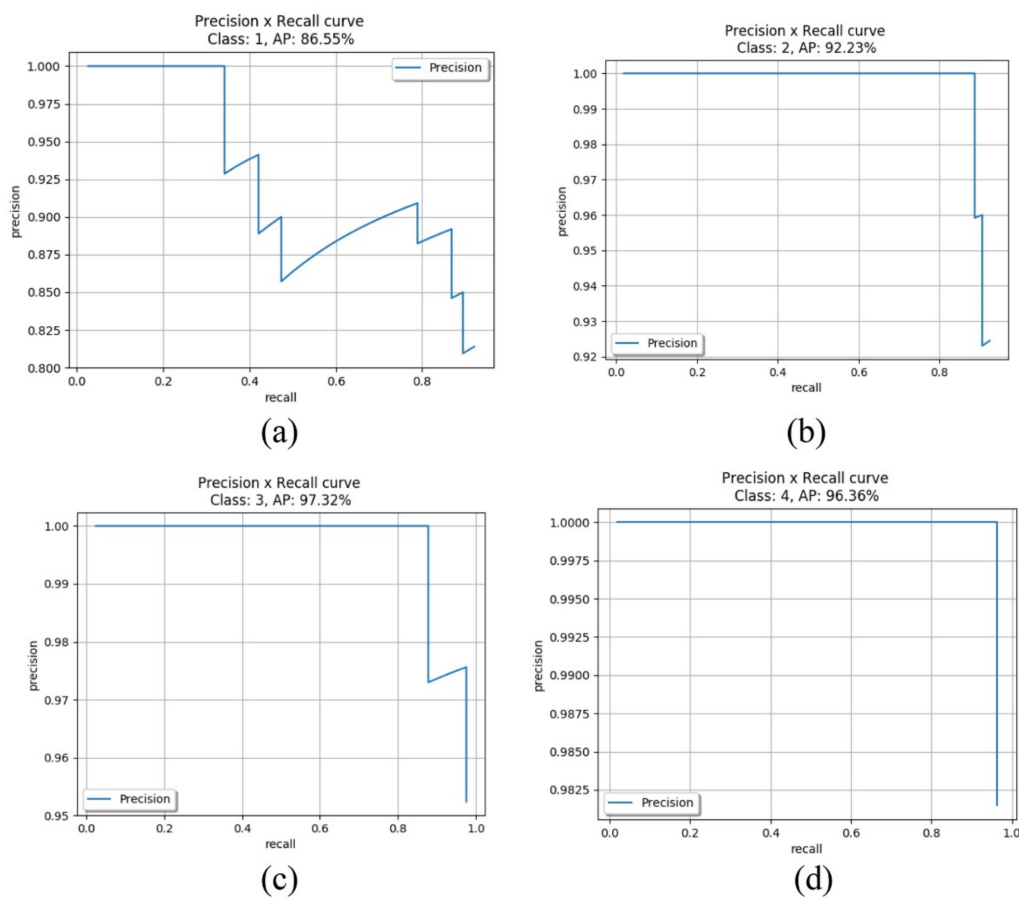


Figure 11. Precision/recall curves of various electronic components. (a) Electrolytic capacitor. (b) Resistance. (c) Tantalum Capacitor. (d) Potentiometer.

According to the method of sample-by-sample as the threshold dividing point, it can be seen from Figure 11a that the precision value of the electrolytic capacitor appears to oscillate. This is because as the threshold points are shifted to the left, the number of positive samples that are determined to be positive increases, and the number of negative samples that are determined to be positive also increases.

4.3. Testing New Images

We used 14 new images for testing to understand the performance of the improved Mask R-CNN in the instance segmentation of electronic components. These images were taken in a different environment than the images of previous training and testing. The distance from the camera was 5 cm, and the lighting conditions were different. The input image and output image are shown in Figure 12.

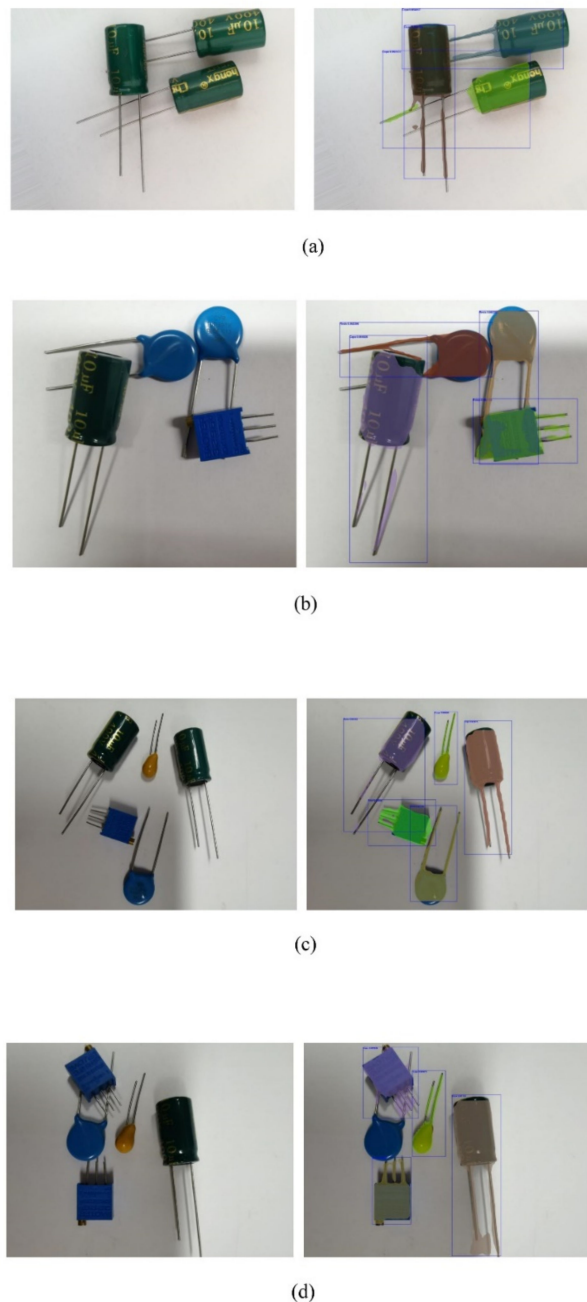


Figure 12. Electronic component segmentation under different light intensity; new testing images (left) and output of the improved Mask R-CNN (right). (a,b), detection of electronic components under intense light. (c,d), detection of electronic components under weak-light or shadow conditions.

Figure 12a,b are the images of electronic components collected under intense light. Some electronic components have the phenomenon of light reflection due to the problem of the surface material of electronic components, which is a factor that affects the detection accuracy. From Figure 12a,b we can see that the detection had a high success rate under intense light. Figure 12c,d shows the images of electronic components collected under weak-light or shadow conditions. The resistance in Figure 12d is not detected. Therefore, light intensity can affect the accuracy of detection, and the conditions for detection still require an appropriate light intensity.

In addition, during the testing process, it appeared the conditions that the electronic components were not detected in a stacked scenario, and multiple similar electronic components were considered

as one object. In Figure 13a, instance segmentation does not perform well on pins of electronic components. The thin characteristic of electronic components and cross placement are the reasons for this test result. Improving the performance of instance segmentation on the pins of electronic components is a meaningful direction for future research. In Figure 13b, the resistance in the middle of the image and the electrolytic capacitor are detected as one object. However, in other images, these electronic components can be successfully tested. The condition of false detection and missed detection is rare compared with the overall detection success rate.

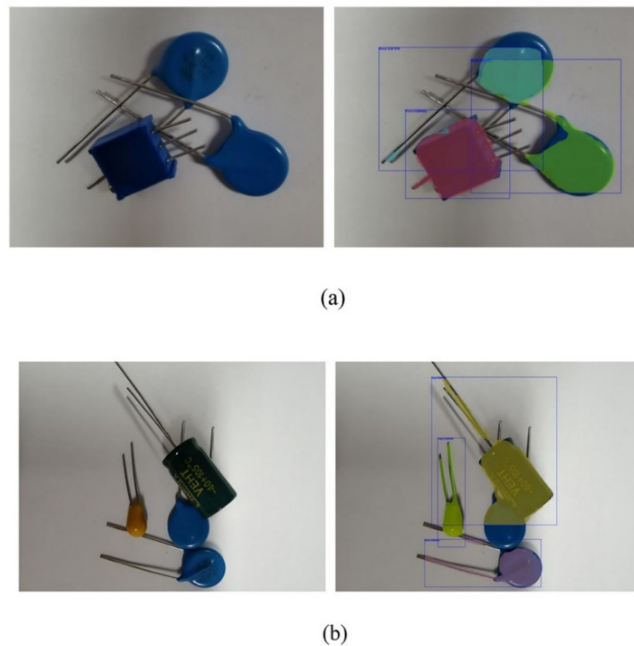


Figure 13. Electronic component segmentation in complex stacking scenarios; new testing images (left) and output of the improved Mask R-CNN (right). (a): detection of pins of electronic components in the case of cross placement. (b): the condition in which two objects are detected as one.

Despite some segmentation errors, the results demonstrate the superior performance of the improved Mask R-CNN on segmenting stacked objects. These errors may be caused by lighting problems, small amounts of training data, or overly complicated stacked scenes. Therefore, in future research, we will enrich our database, which has more types of electronic components, more lighting environments, and more images to improve the robustness and generalization of the model.

4.4. Comparative Study

In order to evaluate the performance of this method, it was compared with other methods under the same dataset. For the Mask R-CNN method, either the FPN or ResNet was used as the backbone network. In the feature extraction stage, the FPN and ResNet consumed a lot of time and slowed down the segmentation due to their deeper network layers and more calculation parameters.

We reported our improved Mask R-CNN on the testing set for comparison. As shown in Table 4, our improved Mask R-CNN with MobileNets-FPN trained on the dataset of electronic components already outperformed Mask R-CNN. Trained and tested with images of 992×744 pixels, our method outperformed the single model of Mask R-CNN with nearly two points under the same initial models. At the same time, the improved Mask R-CNN method reduces the model size from 255.9 MB to 91.1 MB, and the detection speed is twice that of Mask R-CNN, as shown in Table 5.

Table 4. Comparison of mean average precision (mAP) results in instance segmentation.

	AP	AP ₅₀	AP ₇₅	Backbone
Mask R-CNN	60.53	82.48	64.53	ResNet-101
Cascade Mask R-CNN	64.74	89.65	68.47	ResNet-101
Ours	62.63	91.04	67.78	MobileNets-FPN

Table 5. Comparison of detection speed and model size.

	Cascade Mask R-CNN	Mask R-CNN	Ours
Model size	615.7 MB	255.9 MB	91.1 MB
Test time per image	2.61 s	3.9 s	1.8s

The detection accuracy of Cascade Mask R-CNN reaches 64.74%, which is about two points higher than our model. But the model size of Cascade Mask R-CNN is 615.7 MB, which is 6.7 times the size of our model. Moreover, in terms of detection speed, the time to test per image is about 1.5 times that of our model.

We show the detection accuracy of the four types of electronic components in Table 6. The detection performance is better than Cascade Mask R-CNN and Mask R-CNN. The method proposed in this paper can be effectively used for the detection and segmentation of the four types of electronic components and can ensure the best detection accuracy and faster detection speed.

Table 6. Comparison of AP values in object detection.

	Cascade Mask R-CNN (%)	Mask R-CNN (%)	Ours (%)
Electrolytic Capacitor	82.47	94.60	86.55
Resistance	92.35	68.18	92.23
Tantalum Capacitor	86.92	84.74	97.32
Potentiometer	92.62	91.09	96.36
mAP	88.64	84.65	93.12

5. Conclusions

We proposed an improved Mask R-CNN model. We investigated some of the important factors leading to an efficient network. We then demonstrated how to optimize the feature extractor of Mask R-CNN to build a smaller and faster model. Finally, we compared the improved Mask R-CNN to popular models, demonstrating superior size and speed characteristics. The accuracy of instance segmentation surpassed the Mask R-CNN by two points.

In this paper, the method was applied to the detection and segmentation of four types of electronic components. In future research, we will increase the types of electronic components and the number of images in the dataset to improve the robustness of this network.

Author Contributions: Conceptualization, R.D.; Data curation, H.X.; Formal analysis, R.D. and H.X.; Investigation, H.X.; Methodology, R.D.; Resources, R.D. and H.X.; Software, H.X.; Supervision, Z.Y. and J.G.; Writing—original draft, R.D.; Writing—review & editing, Z.Y., R.D., H.X., and J.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (No.51875266).

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

References

1. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [[CrossRef](#)] [[PubMed](#)]
2. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. Presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016.
3. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. Presented at the Computer Vision—ECCV 2016, Cham, Switzerland, 17 September 2016.
4. Girshick, R. Fast R-CNN. *arXiv* **2015**, arXiv:1504.08083. Available online: <https://ui.adsabs.harvard.edu/abs/2015arXiv150408083G> (accessed on 1 April 2015).
5. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *arXiv* **2014**, arXiv:1311.2524.
6. He, K.M.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, 22–29 October 2017; pp. 2980–2988.
7. Hang, M.; Wilson, D.; Huang, Y.; Kelly, S.; Crozier, S.; Bradley, A.; Chandra, S. Fully automatic computer-aided mass detection and segmentation via pseudo-color mammograms and Mask R-CNN. *arXiv* **2019**, arXiv:1906.12118.
8. Dai, Z.; Carver, E.; Liu, C.; Lee, J.; Feldman, A.; Zong, W.; Pantelic, M.; Elshaikh, M.; Wen, N. Segmentation of the Prostatic Gland and the Intraprostatic Lesions on Multiparametric MRI Using Mask-RCNN. *arXiv* **2019**, arXiv:1904.02575.
9. Chiao, J.Y.; Chen, K.Y.; Liao, K.Y.; Hsieh, P.H.; Zhang, G.; Huang, T.C. Detection and classification the breast tumors using mask R-CNN on sonograms. *Medicine* **2019**, *98*, e15200. [[CrossRef](#)] [[PubMed](#)]
10. Xu, Y.; Sun, Z.; Hoegner, L.; Stilla, U.; Yao, W. Instance Segmentation of Trees in Urban Areas from MLS Point Clouds Using Supervoxel Contexts and Graph-Based Optimization. Presented at the 2018 10th IAPR Workshop on Pattern Recognition in Remote Sensing (PRRS), Beijing, China, 19–20 August 2018.
11. Brabandere, B.D.; Neven, D.; Gool, L.V. Semantic Instance Segmentation for Autonomous Driving. Presented at the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, USA, 21–26 July 2017.
12. Chen, K.; Pang, J.; Wang, J.; Xiong, Y.; Li, X.; Sun, S.; Feng, W.; Liu, Z.; Shi, J.; Ouyang, W.; et al. Hybrid Task Cascade for Instance Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
13. Hariharan, B.; Arbeláez, P.; Girshick, R.; Malik, J. Simultaneous Detection and Segmentation. Presented at the Computer Vision—ECCV 2014, Cham, Switzerland, 6–12 September 2014.
14. Hariharan, B.; Arbelaez, P.; Girshick, R.; Malik, J. Hypercolumns for object segmentation and fine-grained localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 15 October 2015.
15. Dai, J.; He, K.; Li, Y.; Ren, S.; Sun, J. Instance-Sensitive Fully Convolutional Networks. Presented at the Computer Vision—ECCV 2016, Cham, Switzerland, 17 September 2016.
16. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. Presented at the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015.
17. Dai, J.; He, K.; Sun, J. Convolutional Feature Masking for Joint Object and Stuff Segmentation. *arXiv* **2014**, arXiv:1412.1283.
18. Dai, J.; He, K.; Sun, J. Instance-aware Semantic Segmentation via Multi-task Network Cascades. *arXiv* **2015**, arXiv:1512.04412.
19. Li, Y.; Qi, H.; Dai, J.; Ji, X.; Wei, Y. Fully Convolutional Instance-aware Semantic Segmentation. *arXiv* **2016**, arXiv:1611.07709.
20. Pham, V.-Q.; Ito, S.; Kozakaya, T. BiSeg: Simultaneous Instance Segmentation and Semantic Segmentation with Fully Convolutional Networks. *arXiv* **2017**, arXiv:1706.02135.
21. Pinheiro, P.; Collobert, R.; Dollár, P. Learning to Segment Object Candidates. *arXiv* **2015**, arXiv:1506.06204.
22. Pinheiro, P.O.; Lin, T.-Y.; Collobert, R.; Dollár, P. Learning to Refine Object Segments. Presented at the Computer Vision—ECCV 2016, Cham, Switzerland, 17 September 2016.

23. Cai, Z.W.; Vasconcelos, N. Cascade R-CNN: Delving into High Quality Object Detection. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 17 December 2018; pp. 6154–6162.
24. Liu, S.; Qi, L.; Qin, H.F.; Shi, J.P.; Jia, J.Y. Path Aggregation Network for Instance Segmentation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; 8759–8768.
25. Nicholas, R.; Paul, N.; Siddhartha, S. Physics-Based Grasp Planning Through Clutter. In *Robotics: Science and Systems VIII*; MIT Press: Cambridge, MA, USA, 2013; pp. 57–64.
26. Di, G.; Tao, K.; Fuchun, S.; Huaping, L. Object discovery and grasp detection with a shared convolutional neural network. Presented at the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016.
27. Zhang, H.; Lan, X.; Bai, S.; Wan, L.; Yang, C.; Zheng, N. A Multi-task Convolutional Neural Network for Autonomous Robotic Grasping in Object Stacking Scenes. *arXiv* **2018**, arXiv:1809.07081.
28. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016.
29. Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. *arXiv* **2016**, arXiv:1612.03144.
30. Howard, A.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.
31. Nair, V.; Hinton, G. Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010.
32. Dumoulin, V.; Visin, F. A Guide to Convolution Arithmetic for Deep Learning. *arXiv* **2016**, arXiv:1603.07285.
33. Mask R-CNN. Available online: https://github.com/matterport/Mask_RCNN (accessed on 15 October 2019).



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).